2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

CPFG-SLAM:a robust Simultaneous Localization and Mapping based on LIDAR in off-road environment*

Kaijin Ji[1] and Huiyan Chen[1], Huijun Di[1], Jianwei Gong[1], Guangming Xiong[1], Jianyong Qi[1], Tao Yi[2]

***Abstract*— Simultaneous localization and mapping (SLAM), as an important tool for vehicle positioning and mapping, plays an important role in the unmanned vehicle technology. This paper mainly presents a new solution to the LIDAR-based SLAM for unmanned vehicles in the off-road environment. Many methods have been proposed to solve the SLAM problems well. However, in complex environment, especially off-road environment, it is difficult to obtain stable positioning results due to the rough road and scene diversity. We propose a SLAM algorithm based on grid which combining probability and feature by Expectation-maximization (EM). The algorithm is mainly divided into three steps: data preprocessing, pose estimation, updating feature grid map. Our algorithm has strong robustness and real-time performance. We have tested our algorithm with our datasets of the multiple off-road scenes which obtained by LIDAR. Our algorithm performs pose estimation and feature map updating in parallel, which guarantees the real-time performance of the algorithm. The average processing time of each frame is about 55ms, and the average relative translation error is around 0.94%. Compared with several state-of-the-art algorithms, our algorithm has better performance in robustness and location accuracy.**

I. INTRODUCTION

As an important component of positioning for the driverless vehicle, simultaneous localization and mapping on outdoor scenes has drawn more and more attention in recent years. SLAM can position the vehicle by the measured data of surrounding environment obtained from sensors, which can relatively complement with Global Navigation Satellite System (GNSS). It is particularly important to acquire an accurate localization of the vehicle by SLAM in situation where GNSS is inaccurate or even not available.

At present, many algorithms [1] have achieved good results in a specific environment, and most of the mainstream SLAM algorithms are based on camera [2] [3] or LIDAR [5] [10]. The Visual SLAM is sensitive to ambient lighting and optical texture and not stable in the outdoor environment, so it can not be used for all-weather unmanned vehicles within a short time. Lidar-based SLAM won't be affected by this challenge, meanwhile the cost of LIDAR is gradually reduced because of the maturity of sensor technology. So it has gradually become a hot spot of research.

There are many state-of-the-art SLAM algorithms that perform well in both real-time performance and location accuracy in urban environment. However, the off-road environment is more complex(please refer to Fig. 1). Some areas are open and have fewer features, which are challenging for feature based algorithms. At the same time, in the bushes, the irregularity of trees is difficult to describe by a specific feature. The bump of the road is also challenging the matching algorithm, because bumpy road is not conducive to initial pose estimation, and most of matching algorithms rely on initial pose. In summary, most of these algorithms might not be well compatible for the wide variety in the off-road environment. In order to improve the robustness for off-road environment, we develop a method of combining the point cloud feature with the occupancy probability in the grid.

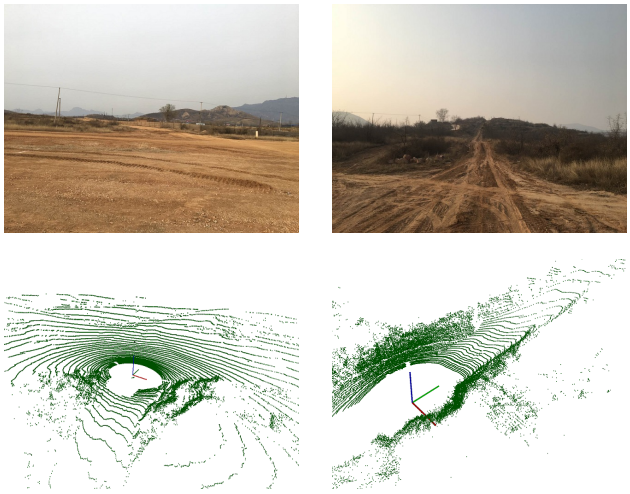


Fig. 1. experimental scenes in the off-road environment, the environment is complex, both open area and trees flourishing, and the rough road are not conducive to the matching of the point cloud.

This paper proposes the Closet Probability and Feature Grid (CPFG) SLAM algorithm which can still be robust and real-time in off-road environment. The algorithm combines feature with probability, extracts and updates point cloud feature and probability of each grid. Then EM algorithm is used to construct the optimization function for matching between point cloud and the grid map. The real-time performance of the algorithm is improved by using two threads to run the point cloud matching and the map updating in parallel. We make numerous experiments in many different off-road scenes. Our algorithm can get vehicle location results and update the map in real time. It costs about 55ms to obtain the LIDAR odometry results in every sweep. Meanwhile, our algorithm has better performance in robustness and location accuracy than several others.

*This study is supported by the National Natural Science Foundation of China (No.61703041).

[1]Kaijin Ji, Huiyan Chen, Huijun Di, Jianwei Gong, Guangming Xiong and Jianyong Qi are with the Intelligent Vehicle Research Center, Beijing Institute of Technology, Beijing 100081 China. jikaijin94@gmail.com,ajon@bit.edu.cn

[2]Tao Yi is with the Beijing Special Vehicle Academy, Beijing 100081 China. taotaoyiyi@163.com

The remainder of this paper is organized as follows. Section II presents related work and provides a brief introduction about some related algorithms. Section III introduces CPFG-SLAM and specifies the functionality in detail. Section IV describes the experiments performed in off-road scenes in order to evaluate our algorithm in positioning accuracy and real-time performance. Both conclusion and future work are presented in Section V.

II. Related work

The 3D LIDAR-based SLAM is mainly divided into two categories, one based on feature of point cloud and the other based on grid map. The former consists of the following steps. Extract the line and surface features [5] of point cloud or landmark feature [7], and then match the feature of current point cloud with the feature map. the latter, three-dimensional space is divided into voxels, is implemented by matching and registering point cloud onto the grid map [8]. In order to improve the robustness of the algorithm, most of this kind of methods apply multiple hybrid map [9] with different resolution. There are also some ways to combine the grid-based method and the feature-based one [10]. These algorithms mostly perform well in urban environment, but many methods are not adaptive to complex off-road scenes.

The evaluation of the SLAM algorithm mainly consists of the calculation performance and the positioning accuracy. It is crucial how to balance the relationship between both two. LOAM [5], as the state-of-the-art SLAM, presented the two-step algorithm which consists of Sweep-to-Sweep Motion Estimation and Sweep-to-Map Registration. The former can rapidly provide positioning result, but the accuracy is a little rough. The latter is quite the opposite, low efficiency but high precision. The two complement each other in order to ensure the accuracy and meet the requirement of real time. But at the same time, the algorithm also has the defect that it cannot create a map in real time. The frequency of mapping is only about 1Hz.

The matching algorithm plays an important role in LIDAR-based SLAM. The current mainstream algorithms include the ICP algorithm and the extension of ICP, Point to plane ICP [11] and so on [12]. The ICP algorithm needs the search algorithm to acquire the nearest neighbor points. Generally, the KD tree or octree [13] is used to manage the point cloud, so as to improve the search efficiency. Meanwhile, because of the large number of points from the 3D LIDAR, the filter should be used before the ICP algorithm to further improve the efficiency. NOctoSLAM [15] also extends the point to plane ICP further, and stores the surface normal vector of the point cloud directly through the octree-based map, which greatly improves the efficiency of computing. But it is only concerned with the plane feature, so it is difficult to be compatible with the environment with less features.

Normal Distribution Transform (NDT) [14] is another important point cloud matching algorithm, which calculates the normal distribution in the surface of the point cloud, and then matchs the point cloud the normal distribution by maximum likelihood estimation. Compared with ICP, the algorithm avoids to search a large number of nearest neighbor points.

The open source software cartographer is based on 3D probabilistic grid map. It achieves the localization and mapping by matching and registering point cloud onto two different resolution probabilistic grid maps in real time.

III. CPFG-SLAM

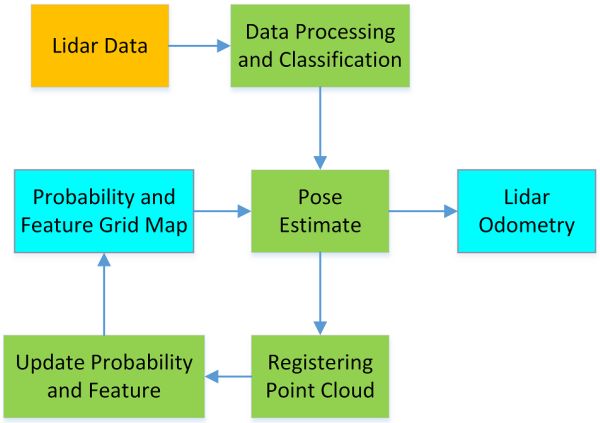
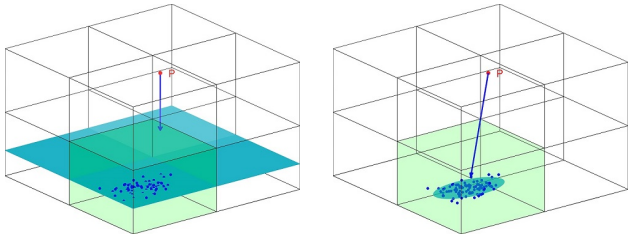


Fig. 2. algorithm flow for CPFG-SLAM

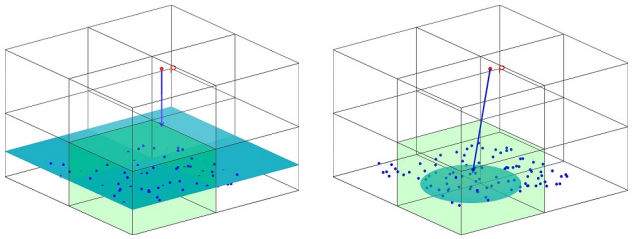
We propose the method based on point cloud feature and probability grid, which matches and registers point cloud onto the grid map for pose estimation by nearest neighbor grid. The algorithm further expands the point to plane ICP and NDT algorithm. The algorithm flow can be seen in Fig. 2.



(a) flat feature extraction (b) point cloud distribution feature extraction

Fig. 3. Comparing (a) and (b), it is found that the point cloud in the graph can describe the grid more accurately with the point cloud distribution than the surface features. The matching direction obtained by the point cloud distribution is more accurate, while the surface features are matched only in the direction of the surface.

Point to plane ICP searches for multiple nearest neighbors, and then obtains the plane by fitting several nearest neighbor points. Finally, based on the distance from the point to the plane, the optimization function is established for iterative optimization. It takes too much time to search for nearest neighbor points in point to plane ICP. Though using KD tree and octree to manage point cloud can improve search efficiency, there still is a large amount of computation, because of the large number of point clouds and searching for repetitions in every optimization. In order to improve



(a) flat feature extraction (b) point cloud distribution feature extraction

Fig. 4. Comparing (a) and (b), the flat feature is more accurate to describe the grid than the point cloud distribution. The matching direction obtained by the surface feature is more accurate, while the point cloud distribution leads to the redundant matching direction.

efficiency, our algorithm also uses NDT's normal distribution grid for reference. Instead of the nearest neighbor point, the nearest neighbor grid is used to reduce search cost and plane feature extraction cost, so as to improve real-time performance.

In addition, compared with NDT, the feature in our algorithm is not only limited to point cloud distribution (please refer to Fig. 3(b)), but also includes surface features (please refer to Fig. 4(a)) or line features, so that we can more accurately discribe the characteristic of point cloud in grid. In contrast of Fig. 3 and Fig. 4, some features are described more appropriately with normal distribution, and some should be expressed in surface. Meanwhile, we combine the feature and occupancy probability further to improve the robustness.

Our algorithm is mainly divided into three steps, data preprocessing, pose estimation, map updating. The main task of data preprocessing is the classification and filtering of the point cloud. In the second step, estimate the position and posture by matching each point cloud to the map. The main role of map updating is to extract point cloud features and update probability of grid. In order to improve the real-time performance of the algorithm, two threads are used to update the map and estimate the pose respectively.

A. Data preprocessing

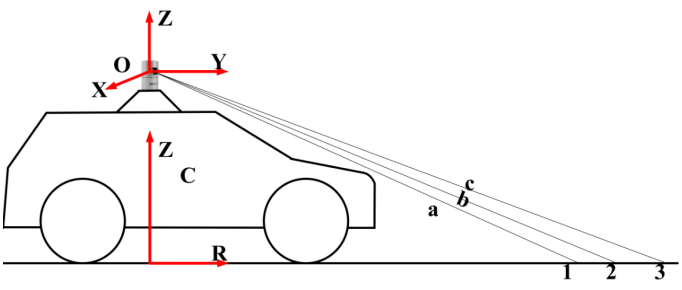


Fig. 5. multilayer LIDAR data acquisition and point cloud classification

A Velodyne 3D HDL-64 LIDAR is used as a perception sensor, which consists of 64 laser transmitters. The HDL-64 has a horizontal field of view (HFOV) of 360° , Vertical field of view (VFOV) of 26.8° , maximum range of up to 120m. We set its rotation frequency to 10Hz, then the horizontal angle resolution is 0.1728 degrees, and the number of points per frame is about 133333. The minimum angle resolution of the vertical direction is 1/3 degrees.

Because of the large number of points, it is necessary to sample the point cloud before matching and updating the map to reduce the operation cost. At the same time, in order to accompany the algorithm proposed in this paper, different laser numbers should be assigned to the points produced by different laser transmitters. As shown in the Fig. 5, C represents vehicle. O represents LIDAR. a, b and c stand for laser beams produced by different laser transmitters at the same time. 1, 2 and 3 represent the identifier of corresponding reflection points (lasernum). Meanwhile, the point clouds of different frames are also assigned different frame sequences (scannum). It is considered that one frame include the points obtained by a revolving circle of LIDAR.

B. Definition and update of the map

First of all, the map is managed by an octree, and the octree's leaf node consists of a point cloud feature and occupancy probability. Point cloud features are divided into three categories, point cloud distribution, plane and line feature. The Gauss distribution is used to approximate the point cloud distribution, as $X \sim N(\mu, \Sigma)$. μ represents the center of the point cloud in the grid. Σ represents the covariance of the point cloud distribution. To unify the format, the line and plane features are also expressed in the form of Gauss distribution.

After the matching of each frame, point cloud is registered and fused into the map which will be updated.

$$M_t = M_{t-1} \cup X^t \tag{1}$$

Where M_t represents the map after the fusion, M_{t-1} represents the map at the last moment, and X^t is the current frame point cloud.

For the start point of the update, we immediately begin to update the probability of each grid using a binary Bayes filter [16] when the grid is no longer empty. However the update of the feature is not, which needs to meet certain condition first. The LIDAR point cloud is sparse, especially in the vertical direction where the angle between adjacent laser beam is large, which results in so fewer points in some grids that we cannot obtain the actual feature. As shown in Fig. 6, the point cloud in the grid is derived from only one laser transmitter of one sweep at the beginning, which is not enough to get real feature. Therefore, feature extraction is not carried out. With the movement of the vehicle, more and more point clouds are filled into the grid, we can obtain the distribution feature of point cloud, then which is updated to the plane feature. Therefore, in order to avoid the negative effect from the sparsity of point cloud, the number of different point identifier (laser identifier and frame sequence) must run up to the threshold before extracting and updating feature grid.

$$\sum_{i=1}^{N_s} N_l^i > N_t \tag{2}$$

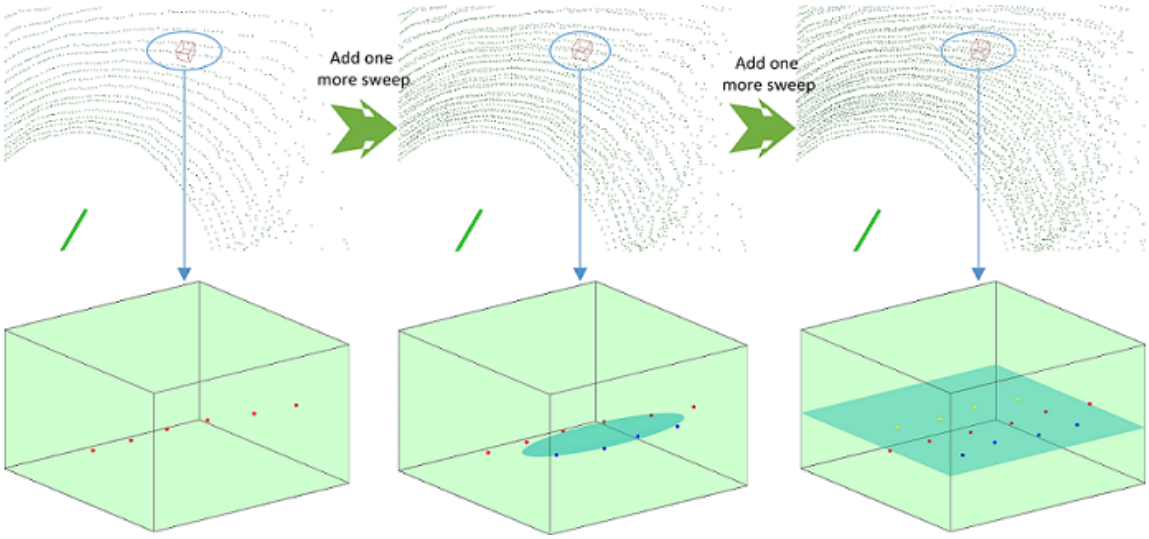


Fig. 6. Extract and update grid. When the point cloud in the grid is derived from only one laser transmitter of one sweep, feature extraction is not carried out. There is no feature in the grid at the beginning. With continuously registering the point clouds, the distribution feature of point cloud is extracted and then updates to the plane feature, meanwhile the occupancy probability is also increasing.

Where N_l^i represents the number of different laser identifiers which belong to the No.i frame point cloud in the grid. N_s represents the number of different frame sequences in the grid, and N_t represents the preset threshold.

For update frequency, only when there is the new data from new sweep or different laser emitter, will the occupancy probability be updated by Bayes. In addition, feature is updated after registering one whole frame of point cloud onto map.The extraction and update of the point cloud features are as follows.

Firstly, we obtain the distribution of point cloud in the grid as $X \sim N(\mu, \Sigma)$. Σ is a real symmetric matrix, the eigenvalue decomposition is easily carried out.

$$\Sigma = V^T \Lambda V \tag{3}$$

$$\Lambda = \begin{vmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{vmatrix} \tag{4}$$

Where Λ represents eigenvalue matrix, which is a diagonal matrix. V is eigenvector matrix. λ_1 λ_2 and λ_3 are eigenvalues. The eigenvalues of the covariance matrix represent the independent variance of the corresponding eigenvector direction.

Though using the feature of point cloud distribution can increase the constraint direction of feature matching, it also causes the phenomenon of feature degeneration of real line and surface features, and brings redundant direction constraint(Fig. 3(b)). For example, the surface feature is restricted by the normal direction, and if the constraints of other directions are added, optimization may be superfluous and wrong. Therefore, before using the characteristic of the point cloud distribution, it is also necessary to further determine whether it is a line or a surface feature. If both the $\lambda_1$andλ_2 are far greater than the λ_3, it is considered as the surface feature, then let the other $1/\lambda_1$and$1/\lambda_2$ equal to 0. Similarly, if the λ_1 is far greater than the $\lambda_2$andλ_3, it is considered as the line feature, then let $1/\lambda_1$ equal to 0.

C. Pose estimation

We firstly finds the nearest neighbour grid by calculating the distance between interest points and the expectation of the distributions in the surrounding grids. In this paper, Expectation-maximization (EM) is used to iteratively estimate the pose. The optimization function is given by

$$\min_T -\sum_{i=1}^{N} P(O_{c_i} \mid X_i, M_t, T) \log P(X_i \mid O_{c_i}, M_t, T) \tag{5}$$

Where N is the number of points chosen to match in the current frame, X_i is the coordinate of the match point, and T is the optimization target which is the pose of the current vehicle relative to the map coordinate system. c_i is the closet grid of point i. $P(O_{c_i} \mid X_i, M_t, T)$ represents the occupancy probability of this grid. $P(X_i \mid O_{c_i}, M_t, T)$ is the distribution of point cloud in this grid. We approximate the distribution to Gaussian distribution, then

$$-\log P(X_i \mid O_{c_i}, M_t, T) \propto (TX_i - \mu)^T V \Lambda^{-1} V^T (TX_i - \mu) \tag{6}$$

Which is similar to the Mahalanobis distance. The eigenvalue of the covariance matrix represents the variance of the point cloud in the corresponding eigenvector direction, that means, there are different optimal weights in different directions. However it is found that Mahalanobis distance exaggerates the effort of particular grids and has poor stability, so we will normalize the eigenvalue matrix and reassign the weight of grid.

$$\Lambda^{-1}_{normalized} = norm(\Lambda^{-1}) \tag{7}$$

Finally, we can get the following optimization problem, which can be solved by the Levenberg-Marquardt algorithm [17]. Where the initialization pose of the optimization function is obtained by the uniform motion model.

$$\min_{T} \sum_{i=1}^{N} \omega_{c_i} (TX_i - \mu)^T V \Lambda_{normalized}^{-1} V^T (TX_i - \mu) \quad (8)$$

Where ω_{c_i} is the matching weight of the c_i grid, which is related to the grid occupancy probability and origin eigenvalue of the covariance matrix.

IV. EXPERIMENTS

In the experiment, we obtain point cloud data from velodyne HDL64 LIDAR in the unknown off-road environment. The algorithm runs on the laptop computer which is equipped with Intel i7 CPU, 2.6GHz quad core, 8GB RAM and Linux platform. With the vehicle running, the LIDAR data and RTK-GPS data are collected at the same time, and we test the algorithm for the localization and real-time performance. In the test, the minimum resolution of the grid is selected as 0.5m.

A. localization and real-time evaluation

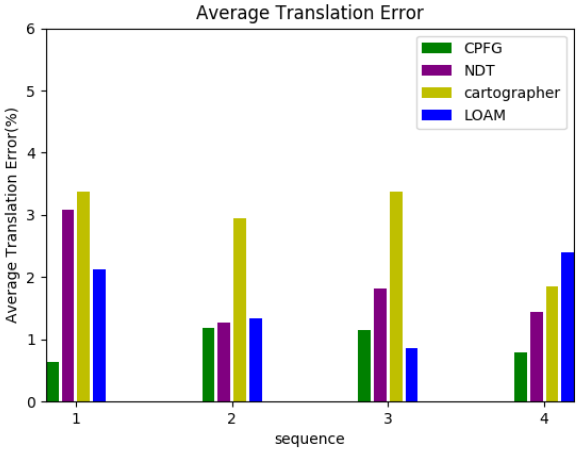


Fig. 8. Average Translation Error

We carry out experiments in a variety of off-road scenes. The vehicle passes through the forest, the open area, the steep slope, where road is also quite bumpy. We keep the speed of 15-20kph. With the vehicle running, there will be large area of dust floating.

We regard RTK-GPS data as the ground truth, then compare the average relative translation error and heading error of CFPG, LOAM, NDT and cartographer in many different scenes in Fig. 8 and Fig. 9. We can see that the location accuracy of both NDT and cartographer in these scenes is not as good as CPFG. Although LOAM sometimes(sequence 3) performs better in the environment with many features, its positioning accuracy is relatively poor in the open area. In general, our algorithm has the highest location accuracy. We don't use the closed loop detection in the four algorithms. Fig. 7 shows the contrast of RTK-GPS location and LIDAR odometry using a variety of algorithms in google earth.

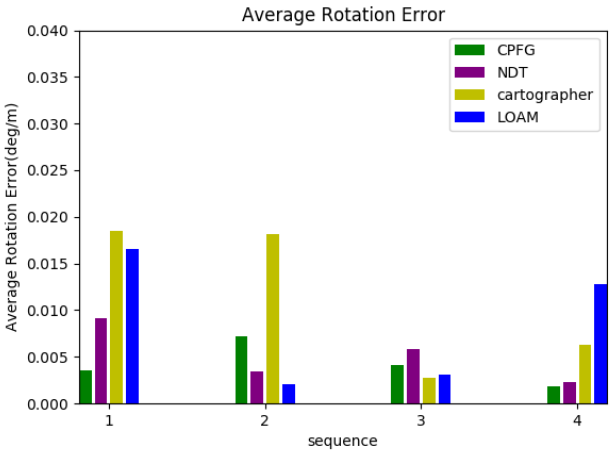


Fig. 9. Average Rotation(Heading) Error

We also test the real-time performance of the algorithm in these scenes, with an average running time of 55ms.

B. mapping

Fig. 7 are the 3D point cloud maps that are built by CPFG-SLAM. It can be seen that both the forest environment and the open environment have quite robust mapping results. In addition, the map can also be built and updated in real time, the update frequency of which is also 10Hz.

V. CONCLUSION AND FUTURE WORK

We proposed a robust SLAM algorithm for an unknown off-road environment. The algorithm is based on the point cloud feature and probability grid, meanwhile, the EM algorithm is used for the optimization function, which has strong adaptability to the different environment. We only use the HDL64 LIDAR for real-time localization and mapping in our algorithm, which is not fused with IMU or GPS data. We tested our algorithm in a number of off-road data sets, and observed accurate and stable positioning results and good real-time performance. But our algorithm is not optimized for dynamic object and will be affected by dynamic object. In the future, we will consider the tracking of dynamic objects to better adapt to the environment of the city and the highway. At the same time, we will also study the location error model of the algorithm, and get the relationship between location error distribution and environment, in order to integrate with other positioning systems, such as GPS.

REFERENCES

[1] C. Cadena *et al.*, "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age," in *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309-1332, Dec. 2016.

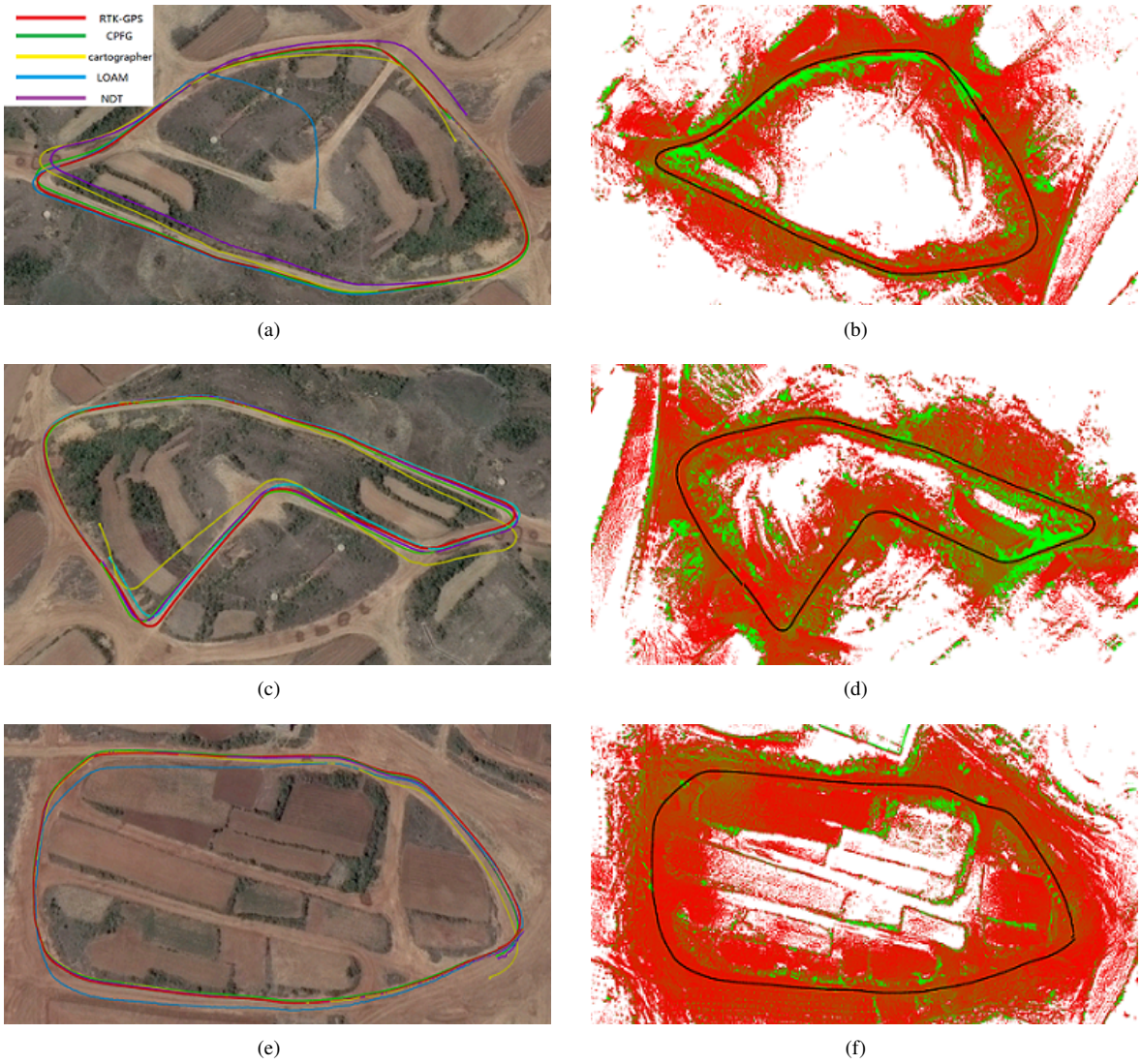


Fig. 7. (a)(c)(e) show the path estimated by CPFG, cartographer, LOAM, NDT and RTK-GPS positioning result in different off-road environment. AS a whole, our algorithm, CPFG, has the highest positioning accuracy. (b)(d)(f) are respectively the result of map reconstruction by using CPFG. The road are very bumpy in these scenes.

[2] Cvi̇šić, Igor, et al. "SOFT-SLAM: Computationally Efficient Stereo Visual SLAM for Autonomous UAVs." *Journal of Field Robotics* (2017).

[3] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," in *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255-1262, Oct. 2017.

[4] Wei-Hsin Chou. State estimation and vision-based occupancy mapping for off-road driving. *Masters thesis, Robotics Institute , Carnegie Mellon University, Pittsburgh, PA,* August 2017.

[5] Zhang, Ji, and Sanjiv Singh. "LOAM: Lidar Odometry and Mapping in Real-time." *Robotics: Science and Systems.* Vol. 2. 2014.

[6] Grisetti, Giorgio, Cyrill Stachniss, and Wolfram Burgard. "Improved techniques for grid mapping with rao-blackwellized particle filters." *IEEE transactions on Robotics* 23.1 (2007): 34-46.

[7] Montemerlo, M. "A Factored Solution to the Simultaneous Localization and Mapping Problem with unknown Data Association." Ph. D. thesis, Carnegie Mellon University (2003).

[8] Zayed Alsayed, Guillaume Bresson, Fawzi Nashashibi, Anne Verroust-Blondet. PML-SLAM: a solution for localization in large-scale urban environments. *PPNIV - IROS 2015*, Sep 2015, Hambourg, Germany. 2015.

[9] Choi, Jaebum, and M. Maurer. "Hybrid map-based SLAM with Rao-Blackwellized particle filters." *International Conference on Information Fusion IEEE*, 2014:1-6.

[10] Choi, Jaebum. "Hybrid map-based SLAM using a Velodyne laser scanner." *IEEE, International Conference on Intelligent Transportation Systems IEEE*, 2014:3082-3087.

[11] Low, Kok Lim. "Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration." *Chapel Hill* (2004).

[12] Segal, Aleksandr, Dirk Haehnel, and Sebastian Thrun. "Generalized-ICP." *Robotics: science and systems.* Vol. 2. No. 4. 2009.

[13] Hornung, Armin, *et al.* "OctoMap: An efficient probabilistic 3D mapping framework based on octrees." *Autonomous Robots* 34.3 (2013): 189-206.

[14] Merten, Heinz, ed. "The three-dimensional normal-distributions transform." threshold 10 (2008): 3.

[15] J. Fossel, K. Tuyls, B. Schnieders, D. Claes and D. Hennes, "NOctoSLAM: Fast octree surface normal mapping and registration," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, Canada, 2017, pp. 6764-6769.

[16] S. Thrun, W. Burgard, and D. Fox, Probabilistic robotics. *MIT press,* 2005.

[17] Moré, Jorge J. "The Levenberg-Marquardt algorithm: implementation and theory." *Numerical analysis. Springer, Berlin, Heidelberg*, 1978. 105-116.