A Real-Time Embedded Localization in Indoor Environment using LiDAR Odometry

Genghang Zhuang, Shengjie Chen and Kai Huang
Key Laboratory of Machine Intelligence and Advanced Computing
School of Data and Computer Science
Sun Yat-sen University
Email: {zhuanggh2,chenshj35}@mail2.sysu.edu.cn, huangk36@mail.sysu.edu.cn

***Abstract*—Localization for mobile robots in a pre-structured environment with a given prior map is considered as the essential problem to perform the further autonomous navigation. In indoor environments without the access to external localization system like GPS, we present a localization method based on the Monte Carlo Localization (MCL), only utilizing a modern 2D LiDAR of high update rate and low measurement noise, to locate the mobile robot in the prior map without giving a starting point. A LiDAR pseudo-odometry is proposed to compute pose changes in movements of the robot, in which the scan point clouds are matched against a locale map to reduce the cumulative errors. In localization iteration, the LiDAR odometry provides motion data to predict the position hypotheses, which is corrected by incorporating the current LiDAR observation to update and yield the localization estimates. The experiments performed on a car-like mobile robot in the real indoor environment demonstrate the accuracy and the real-time performance of the proposed localization system.**

I. Introduction

There has been increasing interest in developing autonomous operating ground vehicles in mobile robotics, which can be deployed in extensive scenarios to accomplish civilian and military tasks, including unknown area exploring, natural disaster rescue, and unmanned district patrol. In addition, it is also getting great attention in public because of its service applications in indoor environments.

The localization with a prior map is essential for a robot to perform autonomous navigation to a given destination. Many simultaneous localization and mapping (SLAM) methods and algorithms are successfully applied on ground vehicle robots to build a map for the environment and estimate the pose changes and the position relative to the origin of the built map. Different from SLAM, localization with a prior map is to determine where the robot is on the pre-charted map at the beginning of navigation and during the movements heading to the destination. In the urban environment, the problem is often solved with the Global Positioning System (GPS) and the Inertial Navigation System (INS) [1]. However, mobile robots in indoor environments without access to the external localization system such as GPS must rely on other approaches to locate itself.

In this paper, we propose a novel localization method based on Monte Color Localization (MCL) using a high update rate and low measurement noise 2D LiDAR, without other sensors like wheel encoders or IMU involved. Different from normal MCL methods deployed on ground robots, we utilize LiDAR as the only sensor to generate pseudo-odometry data of robot pose changes as the robot motion model in the prediction phase, in which locale maps is built during movements to reduce the accumulating error. In the next correction phase, we use the laser scan data as the measurement model to correct the prediction. After a number of iterations with particle re-sampling, a sufficiently accurate pose estimation will be yielded from the system.

Experimental results on our car-like mobile robot platform, running on an embedded ECU, demonstrate the accuracy of the localization, also the robustness and real-time capability of the system. The results running on the car-like robot further show that this method with only one main sensor can greatly endow the autonomous robot platform a high integration and portability.

II. Related Works

Many kinds of research attempted to solve the problem of mobile robot localization in indoor environments with various sensors. One of the common methods is path dead reckoning using internal odometry such as wheel encoders and gyroscopes [2], [3] with the given outset position from the beginning of robot movements. But it is hard to determine some physical parameters of the robot such as the scale factors of the wheels and gears, and the robot kinematics model with high accuracy. These problems can cause large accumulating errors in the dead reckoning localization process. The approach cannot be applied in cases starting from an unknown position in the map, without access to environment references.

Some methods focus on providing an external reference system in indoor environments, based on the electromagnetic signal strength like the WiFi [4]. Though the methods based on the wireless signal are successfully deployed on mobile robots [5], the requirements and effort of the pre-deployment and the accuracy subject to the variant medium and indeterministic interference are the key challenges for these solutions.

Monte Carlo Localization [6] is a probabilistic approach with particle-based density representations for robot position estimates, which is able to localize a mobile robot without knowledge of its starting location. Methods proposed in [7], [8] apply the Monte Carlo Localization to fuse both the odometry and laser scan data, which can greatly eliminate the

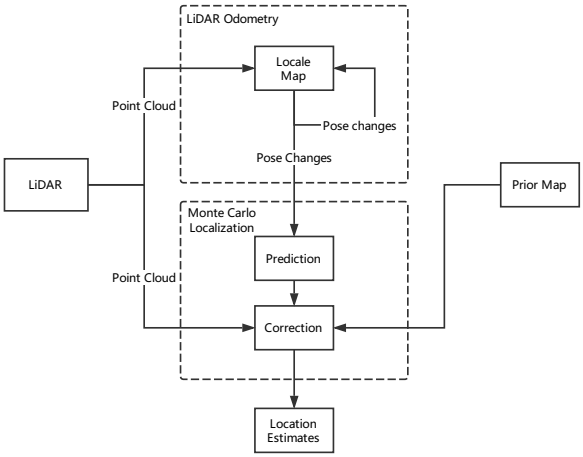


Fig. 1. Overview of System

cumulative errors of odometry by utilizing probabilistic correction approach with LiDAR measurement model. However, these methods still require the odometry of sufficient accuracy, which is subject to the external affection from the environment like driving wheels slipping and driven wheels stuck. In addition, the solution implementations of these systems are hardware platform specific because of their invasiveness in odometry pre-deployment.

III. System Overview

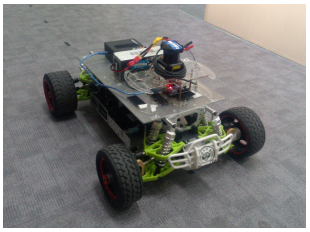
A. Software System Overview

The proposed localization system mainly consists of two components. The LiDAR odometry part takes in laser scan point cloud and estimates the pose changes from the robot motion, while the MCL part takes in the motion data from the LiDAR virtual odometry to update position estimation in the given prior map of the environment.

The Fig. 1 illustrates the framework of the localization system. LiDAR odometry processes the incoming laser scan data from a high update rate and low measurement noise 2D LiDAR. To reduce the accumulating errors in registering two consecutive scan point clouds in a long time period, registering scan data to a locale map is applied. At the beginning of robot movements, a locale map is initialized with the first scan. Every frame of scans in later will be registered to the locale map, to compute the pose change and update the locale map recursively for the next incoming scan.

Monte Carlo Localization is based on the Bayes' Theorem using a set of particles to represent the robot position hypotheses distribution. In recursive estimation process, MCL do the following works to update and approximate the position in reality:

- Prediction: Predict the new set of particles according to the robot motion model. In our method, the motion model is obtained from the LiDAR odometry above.



(a) Car-like Mobile Robot

(b) Hokuyo 2D LiDAR

Fig. 2. Hardware Platform

- Correction: Update the weight of each particle according to the real-time observation from the LiDAR.
- Re-sampling: Re-sample the particles in set depending the weight of each particle in a period of time.

The LiDAR odometry is involved in the prediction step to do coarse position update for every particle, by which a localization system taking advantage of modern LiDAR as the only sensor gets completed.

B. Mobile Robot Platform

The proposed method is tested and validated on a car-like mobile robot at a size of $0.70m \times 0.52m \times 0.35m$, which is motivated by a driving motor for the four wheels and a steering motor for two front wheels, controlled by an Arduino single-board microcontroller, as shown in the Fig. 2a.

A *Hokuyo UTM-30LX-EW* LiDAR of high update rate and accuracy is installed at the front of the upper platform, as the only main sensor of the localization system, which can provide measurement data in a range of $270°$ with a resolution of $0.25°$, per 25 milliseconds. The LiDAR is fixed on a metal platform with spring suspension to obtain 2D point clouds from the environment at a relatively stable height.

The software system is running on an Electronic Control Unit (ECU) with an Intel ultra-low power processor embedded in. The software platform is based on the Robot Operating System [9], which can provide highly efficient communication for local processes and also the remote computational nodes. For real-time robustness, the localization iteration is resolved on the ECU platform, and the status and position estimates are transmitted to a PC or laptop for display and interacting purposes.

IV. LiDAR Odometry

The pose of a robot in a 2D map can be defined as a three degree of free variable $\mathbf{x} = (x, y, \theta)^T$ indicating the position coordinate and the heading angle relative to the origin of the given prior map, which is finally estimated in the MCL iteration introduced in the next section.

The LiDAR odometry in this part serves as a pseudo-odometry to give transitions of robot state in motion model, fed with consecutive point clouds from LiDAR scans.

To compute pose change over two successive scans or a period of time, defined as $\Delta\xi = (\Delta x, \Delta y, \Delta\theta)^T$, in our approach the scan from the LiDAR observation is matched

against the locale map using the nonlinear optimization method, similar to the subproblem of SLAM. In further the state of the LiDAR odometry is defined as a tuple $O = \left\langle M, \xi, \{\langle \Delta\xi_t, t\rangle_{t=0,1,\dots}\}\right\rangle$, composed of maintained locale map which keeps tracking robot movements in period of time, accumulated pose change from local movement origin to the current robot position, and the pose changes sequence over time.

A. Pre-processing

The raw data from LiDAR of a full range scan consists of a sequence of values indicating distances to endpoints in ranger scope stepping by scanning angular resolution. Therefore, the each endpoint can be intuitively presented as $e = (d, \theta)$ in the polar coordinate form. For further processing, the endpoints are transformed into the $\mathbb{R}^2$ Cartesian coordinate form.

$$\mathbf{e} = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} d \cdot cos(\theta_o - \theta) \\ d \cdot sin(\theta_o - \theta) \end{pmatrix} \tag{1}$$

Note that the scan starting angular offset θ_o and increasing direction of θ is LiDAR and software platform specific, and should be taken into account in implementation. The transformed endpoints here are in the LiDAR reference frame, whose origin indicates the LiDAR center.

B. Locale Map

Point clouds of scans are registered to construct and update the locale map and yield the pose change estimates at the same time. The major motivation for adopting locale map is to track and keep surroundings details and features in a range of vicinity for further refinement in point cloud matching with less accumulated errors, rather than considering only two point clouds from the adjoint scans.

The locale map uses occupancy grids to represent the environment surroundings information, which is a prevalent approach in 2D SLAM for mobile robots [10]. The occupancy grid map down-sample endpoints into discrete grids to greatly reduce computational consumption, and tolerate observation noise with probabilistic strategy [11].

For a grid g, the occupancy to describe the probability of being occupied is usually given as a log-odds:

$$M(g) = log\frac{p(g)}{1 - p(g)} \tag{2}$$

where the $p(g)$ is the probability of being occupied for the grid g. At the beginning, the grids are initialize with $M(g) = 0$ inferring $p(g) = 0.5$ for the unknown area. When updating the grid map with a given scan point cloud, the occupancy values of grids which beams go through are updated with:

$$M(g) = M_{curr}(g) + log\frac{p_{free}}{1 - p_{free}} \tag{3}$$

while values of the grids where endpoints drop in are set to:

$$M(g) = M_{curr}(g) + log\frac{p_{occ}}{1 - p_{occ}} \tag{4}$$

where the $log\frac{p_{free}}{1-p_{free}}$ and $log\frac{p_{occ}}{1-p_{occ}}$ are constants referred as the measurement model beliefs.

In practice, a locale map only tracks a period of robot movements to reach agreement on both accuracy and memory consumption, especially in a large scale environment. The maintained part of the locale map is constrained in a size of window in our method, which shifts to keep around to the robot position in a long time movements, trimming the unconcerned far away part.

C. Pose Change Estimating

To compute pose change in LiDAR Odometry processing iteration, the scan point cloud is matched against the locale map, by solving the nonlinear optimization problem as follow:

$$\Delta\xi = \underset{\Delta\xi'}{\operatorname{argmax}} \sum_{i=0}^{N} M^{\star}(\mathbf{T}_{\Delta\xi'} \cdot \mathbf{T}_{\xi} \cdot \mathbf{e}_i) \tag{5}$$

This optimization is to find the best pose change $\Delta\xi$, which the point cloud $\{\mathbf{e}_i; i = 1 \dots N\}$ applies, getting most overlaid and coincided with the locale map around the pose in the last iteration.

Firstly, the scan point cloud gets projected to the last robot pose with $\mathbf{T}_{\xi}$, intuitively referred as a coarse match. Then the $\Delta\xi$ is optimized as the objective refinement for best matching. The $\mathbf{T}_{\Delta\xi}$ is defined as the transformation for the corresponding pose change $\Delta\xi$:

$$\mathbf{T}_{\Delta\xi} = \begin{pmatrix} cos(\Delta\theta) & -sin(\Delta\theta) & \Delta x \\ sin(\Delta\theta) & cos(\Delta\theta) & \Delta y \\ 0 & 0 & 1 \end{pmatrix} \tag{6}$$

Note there are implicit conversions between homogeneous coordinates and Cartesian coordinates.

The $\mathbf{0}$ will be a good initial estimate for $\Delta\xi$ with the belief that the pose change between two scans is minor, owing to the high update rate of the LiDAR, and also the gentle movements of the ground robot in indoor scenarios.

Note that since the accuracy of scan is highly superior to the locale grid map, a up-sampled map $M^{\star}$ for M is necessary for the optimization process. A bilinear interpolation introduced in [12] is adopted in this work.

Registering scans to locale map rather than taking account of only two consecutive scans like the typical ICP method, can greatly minimize the cumulative errors, and avoid unexpected immense deviation in results caused by symmetry in the environment, by giving a good initial estimate closed to the global minima yielded from the locale map approach.

The $\Delta\xi$ serves as the output of the LiDAR odometry in the iteration. Subsequently, the state of the LiDAR odometry, defined above as $O = \left\langle M, \xi, \{\langle \Delta\xi_t, t\rangle_{t=0,1,\dots}\}\right\rangle$, gets updated. The scan point cloud projected on the estimate is registered into the locale map M applying the method mentioned above, while the $\Delta\xi$ is inserted into the sequence of the pose changes

$\{\langle \Delta\xi_t, t\rangle_{t=0,1,\ldots}\}$, updating the pose change ξ relative to the movement origin:

$$\xi = \sum_{i=0}^{N} \Delta\xi_i = \xi_{last} + \Delta\xi \tag{7}$$

V. Monte Carlo Localization

With the input pose change estimates from the LiDAR odometry, the Monte Carlo Localization resolves the global localization problem based on the probabilistic method.

The Monte Carlo Localization approach can be considered as an implementation of the Bayesian filter, in which the pose of the robot at a moment t, defined as $\mathbf{x}_t = (x, y, \theta)^T$, is estimated based on the perception data in the past time, modeled as a posterior density:

$$p(\mathbf{x}_t|Z^t) \tag{8}$$

where the Z^t is the measurement knowledge up to the current moment.

The conditional density is solved in recursive iterations. In the prediction phase, the motion model $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ is involved to give the prediction density:

$$p(\mathbf{x}_t|Z^{t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) p(\mathbf{x}_{t-1}|Z^{t-1}) d\mathbf{x}_{t-1} \tag{9}$$

By applying the Bayes theorem, the posterior density in (8) is obtained recursively with the measurement model $p(\mathbf{z}_t|\mathbf{x}_t)$ incorporated in the correction phase.

$$p(\mathbf{x}_t|Z^t) = \frac{p(\mathbf{z}_t|\mathbf{x}_t) p(\mathbf{x}_t|Z^{t-1})}{p(\mathbf{z}_t|Z^{t-1})} \tag{10}$$

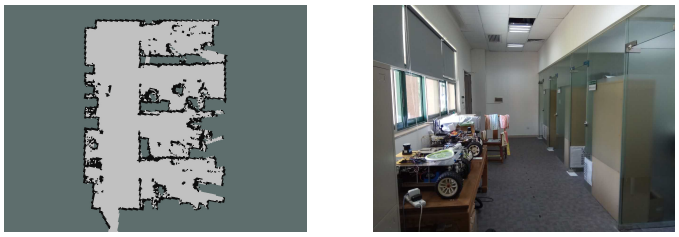
In the MCL method, the posterior density $p(\mathbf{x}_t|Z^t)$ is sampled as a set of particles:

$$S_t = \left\{ < \mathbf{x}_t^i, w_t^i >; i = 1 \ldots N \right\} \tag{11}$$

Each particle in the set represents a candidate hypothesis for robot pose $\mathbf{x}$, with a weight w which indicates the corresponding probability.

In the prediction phase, the pose change $\Delta\xi = (\Delta x, \Delta y, \Delta\theta)^T$ obtained from the LiDAR odometry serves as the motion model, to approximately motivate the particles in respect of the robot motion in reality. The set of particles S'_t describing the predictive density $p(\mathbf{x}_t|Z^{t-1})$ derives from the S_{t-1} in the last iteration, with the pose change $\Delta\xi$ applied.

In the correction phase, the weight of each particle in S'_t is updated according to the match level with the current measurement model. The LiDAR scan data is taken into account as the measurement model in this phase to compute the probability of particle belief with the given prior map. The weighted set then is re-sampled to yield the resulting sample set S_t as the distribution of the robot position hypotheses.



(a) Pre-built Map (b) Environmet Picture

Fig. 3. Experiment Environment

VI. Experiments

A. Experimental Setup

The proposed localization system in this paper has been tested on a car-like robot with a high frequency and low noise LiDAR in a real indoor environment. The method is implemented on the ROS framework and deployed on an ultra-low power processor.

The indoor environment in the experiments is a laboratory office composed of four rooms and a side corridor. For localization and reference purposes, an environment map was obtained preceding the further experiments by driving the mobile manually in the office. The pre-built map is an occupancy grid map generated by applying the 2D SLAM algorithms with LiDAR. The Fig. 3 shows the pre-built grid map and the environment picture in the camera scene. The size of the area is about $7m \times 9m$, represented in the prior map with the grids of $5cm$ resolution.

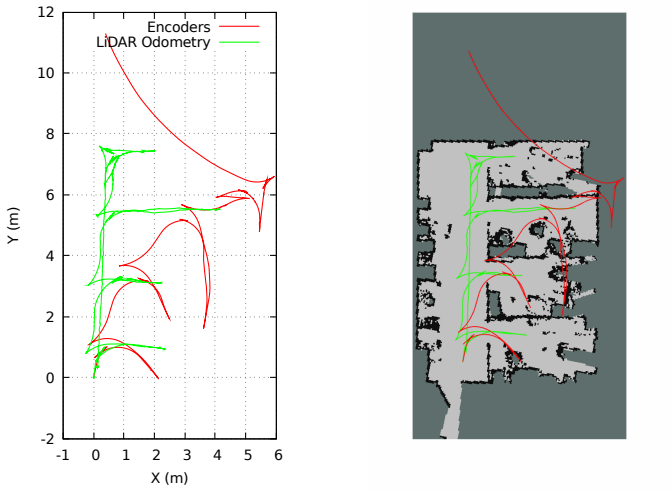
The experiments consist of three parts, focusing on the two subsystems in this work. The first part is intended for validating and testing the LiDAR odometry capacity of computing pose changes in robot movements. The second part verifies and demonstrates the proposed localization approach only utilizing the LiDAR sensor, and the third part presents the performance analysis of the localization system.

B. LiDAR Odometry

To evaluate the accuracy of the proposed LiDAR pseudo-odometry, we performed the path dead reckoning with the pose changes from the LiDAR odometry in the movements of patrolling in the experiment area. For comparison, a physical odometry embedded in the car-like robot including rotary encoders in the rear wheels and steering motor was used to draw an independent path in the same period of movements.

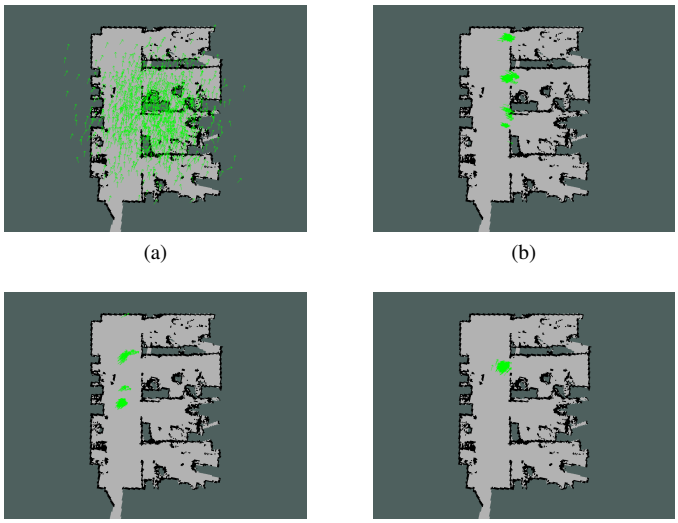
Fig. 4 shows the comparison of drawn paths obtained from reckoning of LiDAR and encoder odometry, with a given starting point in the reference map. In the patrolling movements of the car-like mobile robot in the four rooms, the trajectory path of LiDAR odometry reveals the relatively high accuracy even over a long time move, while the path of physical encoder odometry drafts and deviates quickly over time, showing a high cumulative errors of the physical odometry, especially in the angular pose estimates.

The deviation in the physical odometry could be caused by various factors, including wheel slipping, inaccuracy in



(a) The reckoned paths (b) The paths in reference map

Fig. 4. Comparison of paths reckoned with proposed LiDAR odometry and physical encoders. The path in green color is obtained by accumulating pose changes from the LiDAR odometry, while the red path indicates the trajectory drawn by collecting the encoder data from wheels and steering motor rotating.



(a) (b) (c) (d)

Fig. 5. The iterations of the localization process. The arrows in green color indicate the particles in MCL. (a) Initial distribution of particles. (b) Converged particles on several high probabilities of location. (c),(d) Particles distributed on the true position after further movements.

measurements of parameters like turning radius in this case and the internal errors in encoders, which further suggests the high invariance and portability in different hardware platforms the presented method has endowed.

C. Localization

This part carries out the localization experiments based on the LiDAR odometry, aiming to validate the ability of the localization system to locate the mobile robot in the prior map without a given starting position and to keep tracking the position of the robot in succeeding movements.

The experimental result was attained in the same laboratory office area. The Fig. 5 shows the iterations of the localization process, where the arrows indicate the particle hypotheses for the position and heading of the robot in the MCL.

Initially, the distribution of arrows is uniform over the space without giving a known position. After a while, in the period of movements, the particle arrows begin to converge on several position beliefs because of the symmetry in the environment. In further movements, the distribution of the arrows concentrate in the true position of the robot and trace the robot in the following moves.

TABLE I
AVERAGE ERRORS IN METERS ON PRESET REFERENCE POINTS

Point	A	B	C	D	E	F	G
Error(cm)	9.07	2.15	9.54	9.95	1.95	3.02	6.51

Fig. 6 shows the tracking path of another localization process after a converged distribution was obtained. To evaluate the accuracy of the localization in tracking, we drove the car-like robot passing by a series of preset label points, measuring

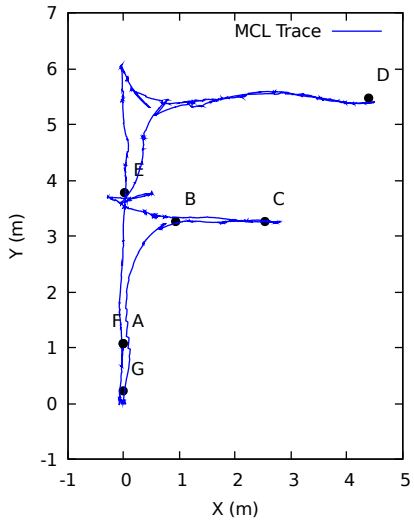


Fig. 6. The localization trajectory in MCL with preset reference points.

the distance errors between the trajectory and reference points, as shown in the TABLE I. Since the footprint size of the car-like mobile robot is $0.70m \times 0.52m$, the result of localization could be considered to be sufficiently accurate.

D. Performance Analysis

The experiments were running on an ECU with an ultra-low power processor *Intel i7 7500U*. To exclude the effects from irrelevant variations in the environment and other parameters, we perform the analysis based on simulations on the same recorded LiDAR scan data.

The statistics in TABLE II show the time cost of the pose change computation in the LiDAR odometry, which reveals the

TABLE II
AVERAGE TIME COSTS IN ITERATION OF LIDAR ODOMETRY

Resolution of Locale Map (cm)	3 - 15
Average Iteration Time Cost (ms)	2.33539

real-time capacity of the LiDAR odometry to provide motion data of the mobile robot at a sufficient update rate. Because of the up-sampling function in the nonlinear optimization process, the time costs remain steady in the limited range of available resolutions, which depends on the measurement noise of the LiDAR and the moving velocity of the mobile robots. In practice, since the MCL only corrects the localization errors after a certain distance movement, the correction rate is about 1Hz - 5Hz, depending on the moving velocity. Therefore, with the LiDAR odometry, the localization system can provide location estimates at an adequate accuracy and update rate at least 100Hz.

VII. CONCLUSION

In this paper, we propose a novel localization method in indoor environments based on the Monte Carlo Localization (MCL), only utilizing a high update rate and low measurement noise LiDAR. A LiDAR odometry is introduced to estimate the pose changes during the movements of the mobile robot, in which a locale map is adopted to reduce the accumulating errors by matching scan point clouds against the map. The LiDAR pseudo-odometry provides the motion model in the prediction phase of the MCL, to give a prediction of pose hypotheses distribution, which is corrected by incorporating the current LiDAR observation.

The experimental results suggest that, with the LiDAR odometry, the proposed localization system is able to locate the robot within a given map in real-time with a sufficient accuracy, without other sensors involved.

ACKNOWLEDGMENT

This work is supported by the Unmanned System Institute of the School of Data and Computer Science in Sun Yat-sen University. We are appreciated and grateful for the support and help.

REFERENCES

[1] H. Qi and J. B. Moore, "Direct kalman filtering approach for gps/ins integration," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 2, pp. 687–693, 2002.

[2] K. Komoriya and E. Oyama, "Position estimation of a mobile robot using optical fiber gyroscope (ofg)," in *Intelligent Robots and Systems' 94.'Advanced Robotic Systems and the Real World', IROS'94. Proceedings of the IEEE/RSJ/GI International Conference on*, vol. 1. IEEE, 1994, pp. 143–149.

[3] Y. Watanabe, "Estimation of position and its uncertainty in the dead reckoning system of the wheeled mobile robot," *Proc. 20th ISIR, Tokyo*, pp. 205–212, 1990.

[4] V. M. Olivera, J. M. C. Plaza, and O. S. Serrano, "Wifi localization methods for autonomous robots," *Robotica*, vol. 24, no. 4, pp. 455–461, 2006.

[5] J. Biswas and M. Veloso, "Wifi localization and navigation for autonomous indoor mobile robots," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 4379–4384.

[6] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 2. IEEE, 1999, pp. 1322–1328.

[7] L. Zhang, R. Zapata, and P. Lépinay, "Self-adaptive monte carlo localization for mobile robots using range sensors," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 1541–1546.

[8] N. Akai, S. Hoshino, K. Inoue, and K. Ozaki, "Monte carlo localization using magnetic sensor and lidar for real world navigation," in *System Integration (SII), 2013 IEEE/SICE International Symposium on*. IEEE, 2013, pp. 682–687.

[9] 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. Kobe, 2009, p. 5.

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

[11] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[12] S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *Safety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium on*. IEEE, 2011, pp. 155–160.