Received November 5, 2019, accepted November 25, 2019, date of publication November 28, 2019, date of current version December 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2956563

Conic Feature Based Simultaneous Localization and Mapping in Open Environment via 2D Lidar

JIAHENG ZHAO[1,2], SHOUDONG HUANG[2], LIANG ZHAO[2], YONGBO CHEN[2], AND XIAO LUO[1]
[1]Beijing Institute of Technology, Beijing 100081, China
[2]Centre for Autonomous Systems, Faculty of Engineering and IT, University of Technology Sydney, Sydney, NSW 2007, Australia

Corresponding author: Xiao Luo (luox@bit.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFC0803000 and Grant 2016YFC0803005, and in part by the China Scholarship Council.

ABSTRACT The conventional planar scan matching approach cannot cope well with the open environment as lacking of sufficient edges and corners. This paper presents a conic feature based simultaneous localization and mapping (SLAM) algorithm via 2D lidar which can adapt to an open environment nicely. The novelty of this work includes threefold: (1) defining a conic feature based parametrization approach; (2) developing a method to utilize feature's conic geometric information and odometry information since open environments are short of regular linear geometric features; (3) developing a factor graph based framework which can be adapted with the proposed parametrization. Simulation experiments and real environment experiments demonstrated that the proposed SLAM algorithm can get accurate and convincing results for the open environment and the map in our representation can express accurately the environment situation.

INDEX TERMS Conic equation, 2D lidar, open environment.

I. INTRODUCTION

Simultaneous localization and mapping (SLAM) is a fundamental research problem for autonomous robot navigation and map construction, comprising robot's or sensors' state estimation and corresponding map construction [1]. As series of researches indicated that different sensors equipped on a robot have significant influences on its potential application, it is worth mentioning that SLAM system based on lidar is considered as an effective and accurate way for robots to construct a map and locate themselves [2]–[5]. One important focus in the area of lidar-based SLAM is how to register scans and how to optimize the trajectories as precisely as possible. Two classes of scan points registration methods are adopted in recent years' research, namely, Iterated Closest Point (ICP) [6]–[8] based method and Gaussian Mixture Models (GMM) based method [9]. The latter one can be extended to a famous special case named Normal Distribution Transform (NDT) [10]–[12]. On this basis, some state-of-the-art 2D lidar-based SLAM algorithms have been developed for many indoor scenarios [13]–[17], especially those constituted by regular, obvious and sufficient lines or corners as is shown in Fig. 1(a).

The associate editor coordinating the review of this manuscript and approving it for publication was Chao Tan.

However, outdoor environment is complicated and cannot be treated as a general scenario (As shown in Fig. 1(b)). Compared to the general indoor environment, an open environment presents a series of challenges including:

1) Lacking of regular linear geometric information. Different from indoor environment, an open area is often composed of scarce trees, which has a significant influence on the performance of scan matching. Assume the robot moves around an object with cylindrical surface, then data acquired by lidar equipped on the robot is distributed on the robot-facing edge. In cases where the robot is operated in an indoor environment with boundaries and polygon features, such edge-distributed data will not increase the error apparently because its weight is diluted by other lines or corners. Once the scenario is lack of polygon features or boundaries, the performance of matching is dramatically declined.
2) Inconsecutive observations. Due to the sparse objects within open environments, observations can not be obtained steadily over time. Therefore, frame-to-frame scan matching cannot be executed.
3) Large scale maps. One common method to build map is via occupancy grid map (OGM). However,

constructing OGM in an open environment is ineffective since the large proportion of the environment consists of free space. Eventually, building OGM still causes a waste of computational memory because much data are stored as ''free grid''.

Currently, numerous approaches have been proposed to enhance the performance of scan matching. A common way is to consider lidar data as features via series of segmentation [18]–[20]. Zhang and Ghosh [21] employed 2D laser rangefinder to locate the robot and built a corresponding map via extracting line segments as basic elements. Coincidentally, Li *et al.* [22] proposed a point and line features based SLAM method. They firstly distinguished point and line features via a Split-and-Merge algorithm, then optimized poses by minimizing l_q-norm distance [23], [24]. However, line features are not suitable for an open environment. Some recent studies have considered the use of conic curves to calculate the location of features [25]. Zhang *et al.* [26] combined 2D lidar and gyroscope to navigate a robot in the forest. They utilized circles to fit scan points and estimated poses by feature-based extended Kalman filter. Unlike our method, their research focused on the initial application assuming all features as circles. Actually, these circular-like shapes are not standard circles. In other words, large accumulated error is unavoidable if scan points are fitted with circle alone. Also, the problem cannot be optimized globally because the feature-based extended Kalman filter is not able to close the loop and correct the accumulated error with time.

To avoid failure in an open environment, some researchers attempted to improve the performance of scan matching [27]. Besides, more researchers tried to seek aid from other sensors such as camera, wheel odometer, GPS, or ultrasonic [28]–[30]. Shin *et.al.* [31] proposed a sparse depth enhanced direct visual-lidar SLAM. Jiang *et al.* [32] proposed a SLAM framework fusing vision sensor and low-cost lidar. Xue *et al.* [33] fused IMU, wheel encoder and lidar simultaneously to estimated ego-motion for an mobile vehicle. However, non of these approaches proposed countermeasures for discontinuous laser scan, even [33] did not mention how to deal with inconsecutive scan while fusing more than three sensors.

Another inevitable aspect is the representation of maps. At present, some common representation ways in 2D are point clouds map [34]–[36] and occupancy grid map [37]–[39]. In addition to these two common representation ways, Einhorn and Gross [40] used Normal Distribtuion Transform (NDT) maps for a lifelong SLAM algorithm. Zhang and Ghosh [21] introduced a closed line segment map consisting of only line segments and defining a closed and connected region.

To overcome the above mentioned limitations, we proposed a novel conic feature based SLAM algorithm for mobile robot working in open environment. In contrast with existing planar SLAM systems or algorithms, we utilized conic characteristic of discrete scan points scattered on the edge of the object, while conventional methods only consider relationship between points or extraction of line segments. More than just utilizing geometric information, a further progress has been made that we defined a new parametrization approach for such conic feature and constructed corresponding factor graph optimization model. We also represented the map with conic features instead of occupancy grid map. In order to verify the feasibility of the proposed approach, we carried out experiments in simulated and real environments respectively. Also we evaluated our approach on a public dataset. The main contributions of this paper include:

1) Defining a conic feature based parametrization approach.
2) Developing a method to utilize feature's conic geometric information and odometry information since open environments are short of regular linear geometric features.
3) Developing a factor graph based framework which can be adapted with the proposed parametrization.

Remainder of this paper is organized as follows: Specific methodological theory and implementation are suggested in Section II. Section III presents detailed comparative experiments and analysis. Finally, conclusions and prospected future works are drawn in Section IV.

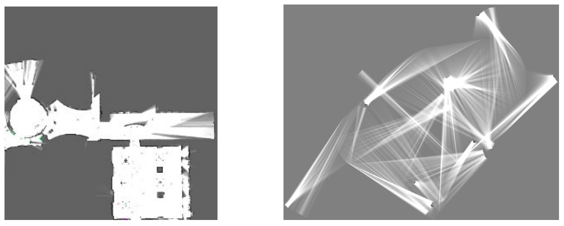


FIGURE 1. Two typical environments. (a) Indoor environment consisting of sufficient lines and corners; (b) Outdoor environment lacking of sufficient lines and corners.

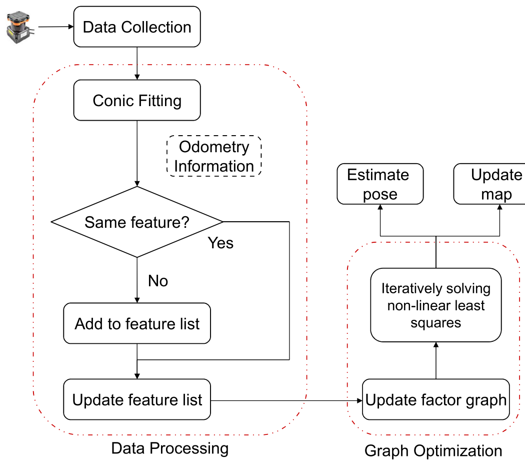


FIGURE 2. Flow chart of the proposed algorithm.

II. METHODOLOGY

The architecture of the proposed algorithm incorporates two main components as is illustrated in Fig. 2. Once data collection via lidar is finished, the first stage named data processing starts and the point sets are fitted by conic equation. The scan points are under the proposed parametrization after fitting. Noted that raw points are still stored although points are fitted into conic feature at this step. Then with the prior of odometry information, the feature list is to be updated if the current feature never occurred, otherwise a new edge between current step and this feature is linked to help close the loop. After data processing, the optimization problem is continued by iterative non-linear least squares method such as Gauss-Newton method or Levenberg-Marquardt algorithm. This stage is called graph optimization or the back-end. Each robot pose, feature parameters and the final map is eventually obtained after the back-end.

A. DATA PROCESSING

Conic feature parametrization and association are finished at this stage. We proposed a conic feature parametrization to model conic features for solving SLAM problem. The conic feature can be fitted on the basis of Ahn's work [41]. We also studied the uncertainty flow from sensor to parameterized feature which makes the fitted result reliable.

1) PARAMETRIZATION

There are four basic types of conics: circles, ellipses, hyperbolas, and parabolas. Fortunately, it is unnecessary to utilize all of the four types. In real world, circular or elliptical shaped objects such as trees and pillars appear more frequently, which means it is easy to implement circle or ellipse equation when denoting actual features. Further more, circle is the special case of ellipse where the major axis and the minor axis have the same dimension. Hence we can take advantage of ellipse equation to express features.

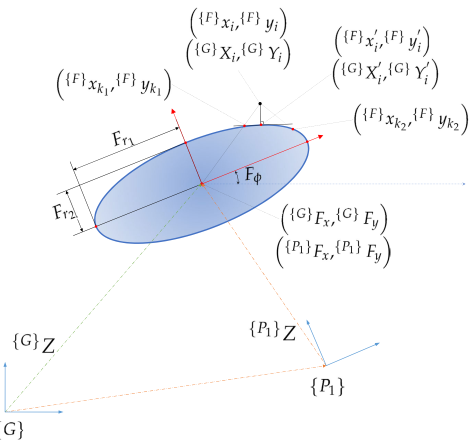

$\left({}^{\{F\}}x_i, {}^{\{F\}}y_i\right)$
$\left({}^{\{G\}}X_i, {}^{\{G\}}Y_i\right)$
$\left({}^{\{F\}}x_i', {}^{\{F\}}y_i'\right)$
$\left({}^{\{G\}}X_i', {}^{\{G\}}Y_i'\right)$
$\left({}^{\{F\}}x_{k_1}, {}^{\{F\}}y_{k_1}\right)$
$\left({}^{\{F\}}x_{k_2}, {}^{\{F\}}y_{k_2}\right)$
F_{r_1}
F_{r_2}
F_ϕ
$\left({}^{\{G\}}F_x, {}^{\{G\}}F_y\right)$
$\left({}^{\{P_1\}}F_x, {}^{\{P_1\}}F_y\right)$
${}^{\{G\}}Z$
${}^{\{P_1\}}Z$
$\{P_1\}$
$\{G\}$

FIGURE 3. Schematic diagram of conic feature parametrization.

Motivated by the above insight, the conic feature suggested in this paper is parameterized by $\Phi = [{}^{\{G\}}F_x, {}^{\{G\}}F_y, F_\phi, F_{r_1}, F_{r_2}]$. As is shown in Fig. 3, $\left({}^{\{G\}}F_x, \ {}^{\{G\}}F_y\right)$ is the central coordinate of the ellipse in the global frame, F_ϕ is the angle between ellipse's major axis and the world frame x-axis. $\left\{F_{r_1}, F_{r_2}\right\}$ are the absolute dimensions of the major axis and the minor axis respectively.

If the feature has circular shape, it is apparently confused to decide the specific position of the major or minor axis as well as the angle. Fortunately, we can still adopt this expression because of the elaboration of feasibility and validity of proposed feature parametrization in Section II-A.3 and Section III-B.

2) ELLIPSE FITTING

Many studies on fitting points into ellipses have been conducted. Our method is based on [41] and [42], supplementing studies on uncertainty flow from sensors to features. This section introduces the implementation approaches and uncertainty transmission.

- Polynomial fitting

The most common and widely used method to fit a ellipse should be polynomial fitting. Given a cluster of points $\mathbf{P} = \{p_i = (x_i, y_i) \mid i = 1, 2, 3, \cdots, n\}$ aligned on the surface of an arbitrary ellipse, obviously all of these points must satisfy the conic equation. With the parametrization mentioned above it is easy to give the equation at each point as follows:

$$f\left(p_i\right) = \frac{\left((x_i - F_x)\cos F_\phi + \left(y_i - F_y\right)\sin F_\phi\right)^2}{F_{r_1}^2} + \frac{\left((x_i - F_x)\sin F_\phi - \left(y_i - F_y\right)\cos F_\phi\right)^2}{F_{r_2}^2} - 1 \approx 0 \tag{1}$$

After series of simplification and like terms combination, a general polynomial form of conic equation is denoted by the following equation:

$$f\left(p_i\right) = Ax_i^2 + 2Bx_iy_i + Cy_i^2 + 2Dx_i + 2Ey_i + F \approx 0 \tag{2}$$

where A, B, C, D, E, F, are polynomial coefficients [42] .

Obviously a trivial solution that all of the coefficients are equal to 0 is good for nothing. To avoid such a situation, several normalization ways can be employed. In this paper, we normalized $A + C = 1$. Then for all the n points Eq. (2) can be revised into vector form:

$$f\left(\mathbf{v}\right) = \mathbf{W}\mathbf{v} - \mathbf{b} \tag{3}$$

where

$$\begin{aligned} \mathbf{W} &= [\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3, \cdots, \mathbf{W}_n]^T \\ \mathbf{b} &= [b_1, b_2, b_3, \cdots, b_n]^T \\ \mathbf{v} &= [A, B, D, E, F]^T \\ \mathbf{W}_i &= \left[x_i^2 - y_i^2, 2x_iy_i, 2x_i, 2y_i, 1\right]^T \\ b_i &= -y_i^2 \end{aligned} \tag{4}$$

Hence the linear least squares problem becomes

$$\min \mathcal{F}_{\mathbf{v}} = \frac{1}{2}\left(\mathbf{W}\mathbf{v} - \mathbf{b}\right)^T\left(\mathbf{W}\mathbf{v} - \mathbf{b}\right) \tag{5}$$

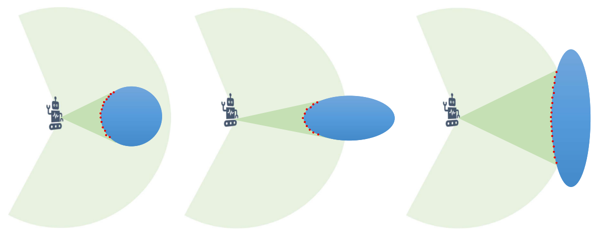

(a) Circular feature (b) Narrow end (c) Wide end

FIGURE 4. Different situations when the robot observes a conic feature.

This problem has a closed-form solution:

$$\mathbf{v}^* = \left(\mathbf{W}^T\mathbf{W}\right)^{-1}\mathbf{W}^T\mathbf{b} \tag{6}$$

Generally an ellipse can be fitted with series of points through the polynomial method talked above. Nevertheless, a robot can always "see" the object in one single direction as is shown in Fig. 4 which implies proper distributed points result in good ellipse fitting. For example, when the robot is in the position as is illustrated in Fig. 4(a), the results obtained by Eq. (6) are highly reliable. But if the robot happens to observe the extreme flat end or extreme narrow end of a ellipse (Fig. 4(b) and Fig. 4(c)), the observed points may contribute badly to the polynomial function, leading to a totally wrong result or even a complex solution. Therefore an enhanced method should be imposed.

- Orthogonal point fitting

In order to overcome the problem discussed in the previous section, one enhanced approach is to minimize the orthogonal distance which is invariant to rigid transformations in Euclidean space and which presents low curvature bias. Fig. 3 depicts various intermediate variables needed in the derivation process. The coordinate transformation of point cloud $\mathbf{P}$ between global coordinate $\{G\}$ and feature coordinate $\{F\}$ is defined by rotation matrix $\mathbf{R} = \begin{bmatrix} \cos F_\phi & \sin F_\phi \\ -\sin F_\phi & \cos F_\phi \end{bmatrix}$ and ellipse center ${}^{\{G\}}F_\mathbf{c} = \left({}^{\{G\}}F_x, {}^{\{G\}}F_y\right)$ by

$${}^{\{F\}}\mathbf{p} = \mathbf{R}\left({}^{\{G\}}\mathbf{P} - {}^{\{G\}}\mathbf{F_c}\right) \tag{7}$$

Because the feature coordinate $\{F\}$ is defined in the standard ellipse form, we can directly apply standard ellipse equation to the point cloud ${}^{\{F\}}\mathbf{p}$.

For any given point ${}^{\{F\}}\mathbf{p}_i = (x_i, y_i) \in {}^{\{F\}}\mathbf{p}$ in ellipse frame, it is easy to find the orthogonal point ${}^{\{F\}}\mathbf{p}'_i = \left(x_i{}', y_i{}'\right)$ located on the ellipse by solving tangent line equation and the standard ellipse equation as follows:

$$f\left({}^{\{F\}}\mathbf{p}'_\mathbf{i}\right): \begin{cases} f_1(x'_i, y'_i) = \frac{1}{2}\left(F_{r_1}^2 y_i'^2 + F_{r_2}^2 x_i'^2 - F_{r_1}^2 F_{r_2}^2\right) \\ f_2(x'_i, y'_i) = F_{r_2}^2 x'_i\left(y'_i - y_i\right) - F_{r_1}^2 y'_i\left(x'_i - x_i\right) \end{cases} \tag{8}$$

Given the point ${}^{\{G\}}\mathbf{p}_i : (X_i, Y_i)$ in coordinate $\{G\}$, transform the point to coordinate $\{F\}$ firstly obtaining ${}^{\{F\}}\mathbf{p}_i : (x_i, y_i)$, then the orthogonal point ${}^{\{F\}}\mathbf{p}'_i : \left(x_i{}', y_i{}'\right)$ can be found by adapting generalized Newton method iteratively to Eq. (8) through the following functions:

$$\begin{aligned} \mathbf{B} &= \left.\frac{\partial f}{\partial\left(\mathbf{x}\right)}\right|_{\mathbf{x}={}^{\{F\}}\mathbf{p}'_i} = \begin{bmatrix} \frac{\partial f_1}{\partial x'_i} & \frac{\partial f_1}{\partial y'_i} \\ \frac{\partial f_2}{\partial x'_i} & \frac{\partial f_2}{\partial y'_i} \end{bmatrix} \\ &= \begin{bmatrix} F_{r_2}^2 x'_i & F_{r_1}^2 y'_i \\ \left(F_{r_2}^2 - F_{r_1}^2\right) y'_i - F_{r_2}^2 y_i & \left(F_{r_2}^2 - F_{r_1}^2\right) x'_i + F_{r_1}^2 x_i \end{bmatrix} \\ \mathbf{B}\Delta &= -f\left(\mathbf{x}_k\right) \\ \mathbf{x}_{k+1} &= \mathbf{x}_k + \Delta \end{aligned} \tag{9}$$

The initial guess $\mathbf{x}_0$ for solving Eq. (9) can be given by approximately calculating the midpoint of two intersection points, where one point $\left({}^{\{F\}}x_{k_1}, {}^{\{F\}}y_{k_1}\right)$ is the intersection of line $\overrightarrow{{}^{\{F\}}F_\mathbf{c}{}^{\{F\}}\mathbf{p}_i}$ and the ellipse. The other point $\left({}^{\{F\}}x_{k_2}, {}^{\{F\}}y_{k_2}\right)$ is the intersection of the perpendicular line at ${}^{\{F\}}\mathbf{p}_i$ with respect to the ellipse's major axis and the ellipse. The three points are calculated by:

$${}^{\{F\}}\mathbf{x}_k = \frac{1}{2}\left({}^{\{F\}}\mathbf{x}_{k_1} + {}^{\{F\}}\mathbf{x}_{k_2}\right) \tag{10}$$

where

$$\begin{aligned} \mathbf{x}_{k_1} &= \begin{pmatrix} {}^{\{F\}}x_{k_1} \\ {}^{\{F\}}y_{k_1} \end{pmatrix} \cdot \frac{F_{r_1}F_{r_2}}{\sqrt{F_{r_2}^2\,{}^{\{F\}}x_{k_1}^2 + F_{r_1}^2\,{}^{\{F\}}y_{k_1}^2}} \\ \mathbf{x}_{k_2} &= \begin{cases} \begin{bmatrix} {}^{\{F\}}x_{k_2} \\ \operatorname{sign}\left({}^{\{F\}}y_{k_2}\right)\cdot\frac{F_{r_2}}{F_{r_1}}\sqrt{F_{r_1}^2 - {}^{\{F\}}x_{k_2}^2} \end{bmatrix} \\ \quad \textbf{if} \left|{}^{\{F\}}x_{k_2}\right| < F_{r_1} \\ \begin{bmatrix} \operatorname{sign}\left({}^{\{F\}}x_{k_2}\right)\cdot F_{r_1} \\ 0 \end{bmatrix} \\ \quad \textbf{if} \left|{}^{\{F\}}x_{k_2}\right| \geqslant F_{r_1} \end{cases} \end{aligned} \tag{11}$$

The orthogonal point ${}^{\{F\}}\mathbf{p}' : \left({}^{\{F\}}x'_i, {}^{\{F\}}y'_i\right)$ is finally obtained after iterative calculation of Eq. (9). At last the orthogonal error distance is minimized by the equation:

$$\min \sum \|e_i\|_\Sigma^2 = \sum \left\|{}^{\{G\}}\mathbf{p}_i - {}^{\{G\}}\mathbf{p}'\right\|_\Sigma^2 \tag{12}$$

after transferring the orthogonal point from feature frame to the global frame ${}^{\{G\}}\mathbf{p}' : \left({}^{\{G\}}X'_i, {}^{\{G\}}Y'_i\right)$. Σ is the covariance matrix of the intrinsic noise of the sensor.

Noted that we have defined ellipse parameters vector $\Phi = [{}^{\{G\}}F_x, {}^{\{G\}}F_y, F_\phi, F_{r_1}, F_{r_2}]$ in global frame, derivatives of Φ can be found through Eq. (7) and Eq. (8):

$$\begin{aligned} \mathbf{J}_{{}^{\{F\}}\mathbf{p}'_i,\Phi} &= \left.\left(\frac{\partial\mathbf{x}}{\partial\Phi}\right)\right|_{\mathbf{x}={}^{\{F\}}\mathbf{p}'_i} \\ &= \left.\begin{bmatrix} -\cos F_\phi & -\sin F_\phi & y'_i & 0 & 0 \\ \sin F_\phi & -\cos F_\phi & -x'_i & 0 & 0 \end{bmatrix}\right|_{\mathbf{x}={}^{\{F\}}\mathbf{p}'_i} \\ \mathbf{J}_{{}^{\{G\}}\mathbf{p}'_i,\Phi} &= \left.\left(\frac{\partial\mathbf{X}}{\partial\Phi}\right)\right|_{\mathbf{X}={}^{\{G\}}\mathbf{p}'_i} \end{aligned}$$

$$
\begin{aligned}
&= \mathbf{R}^{-1}\left(\frac{\partial \mathbf{x}}{\partial \Phi}\right)\Bigg|_{\mathbf{x}={}^{\{F\}}\mathbf{p}_i'} \\
&\quad + \begin{bmatrix} 1 & 0 & -x_i'\sin F_\phi - y_i'\cos F_\phi & 0 & 0 \\ 0 & 1 & x_i'\cos F_\phi - y_i'\sin F_\phi & 0 & 0 \end{bmatrix}\Bigg|_{\mathbf{x}={}^{\{F\}}\mathbf{p}_i'}
\end{aligned}
$$

$$
\begin{bmatrix} \dfrac{\partial f_1}{\partial \Phi} \\ \dfrac{\partial f_2}{\partial \Phi} \end{bmatrix} = \mathbf{0} \tag{13}
$$

The Jacobian matrix is to be derived after series of reductions by

$$
\mathbf{J} = \left(\mathbf{R}^{-1}\mathbf{B}^{-1}\mathbf{C}\right)\Big|_{\mathbf{x}={}^{\{F\}}\mathbf{p}_i'} \tag{14}
$$

where $\mathbf{B}$ is the Jacobian matrix from Eq. (9), and $\mathbf{C} = (\mathbf{C_1}, \mathbf{C_2}, \mathbf{C_3}, \mathbf{C_4}, \mathbf{C_5})$:

$$
\begin{aligned}
\mathbf{C}_1 &= \begin{bmatrix} F_{r_2}^2 x_i'\cos F_\phi - F_{r_1}^2 y_i'\sin F_\phi \\ F_{r_2}^2\left(y_i - y_i'\right)\cos F_\phi + F_{r_1}^2\left(x_i - x_i'\right)\sin F_\phi \end{bmatrix} \\
\mathbf{C}_2 &= \begin{bmatrix} F_{r_2}^2 x_i'\sin F_\phi + F_{r_1}^2 y_i'\cos F_\phi \\ F_{r_2}^2\left(y_i - y_i'\right)\sin F_\phi - F_{r_1}^2\left(x_i - x_i'\right)\cos F_\phi \end{bmatrix} \\
\mathbf{C}_3 &= \begin{bmatrix} \left(F_{r_1}^2 - F_{r_2}^2\right) x_i' y_i' \\ \left(F_{r_1}^2 - F_{r_2}^2\right)\left(x_i'^2 - y_i'^2 - x_i' x_i + y_i' y_i\right) \end{bmatrix} \\
\mathbf{C}_4 &= \begin{bmatrix} F_{r_1}\left(F_{r_2}^2 - y_i'^2\right) \\ 2F_{r_1} y_i'\left(x_i - x_i'\right) \end{bmatrix} \\
\mathbf{C}_5 &= \begin{bmatrix} F_{r_2}\left(F_{r_1}^2 - x_i'^2\right) \\ -2F_{r_2} x_i'\left(y_i - y_i'\right) \end{bmatrix}
\end{aligned} \tag{15}
$$

Finally, the value of ellipse parameters will be solved through iteratively minimizing the orthogonal error distance Eq. (12) among all of the given points. Noticed that an initial guess is still inevitable even using orthogonal point fitting method. Thus in this research the result of Eq. (5) is considered as initial guess. The algorithm of ellipse fitting is explicated in Algorithm 1. We have taken several steps to improve the fitting accuracy.

At first, enough input points are necessary to start fitting. The initial value $\tilde{F}_0$ is given by fitting with polynomial Eq. (6). Then the curvature difference (by calculating the curvatures in each end of the points and finding the difference) is sought and compared with an empirical criterion λ_ϕ. If the difference is acceptable, $\tilde{F}$ is to be solved by iteratively doing Eq. (14).

3) UNCERTAINTY TRANSMISSION

As we know, "ellipse fitting" is an approximation of the raw data, there is information loss during the procedure. It is untrusted to utilize fitted parameters without analyzing uncertainty transmission process. If we denote sensor's information matrix as $I_s = \Sigma^{-1}$, the information matrix of parameterized feature I_f can be calculated by:

$$
I_f = \mathbf{J}^T I_s \mathbf{J} \tag{16}
$$

Algorithm 1 Ellipse Fitting

```
  Input: Scan points $p_i$ in local frame
  Output: Parameters $\tilde{F}$ in global frame
1 if Enough points then
2     Solve initial guess $\tilde{F}_0$ by polynomial fitting via
      Eq. (6)
      if Curvature change of point sets > $\lambda_\phi$ then
          while Not Converged do
3             Calculate $\tilde{F}$ with $\tilde{F}_0$ by orthogonal point
              fitting via Eq. (12) iteratively
          end
      else
          Wait for the next observation
      end
  end
  Notation: $\lambda_\phi$ is an empirical criterion to filter near flat
  distributed points.
```

It should be noted that two cases will cause ill-condition problem of Jacobian $\mathbf{J}$, one is that the point $\mathbf{p_i}$ locates at the ellipse center, while the other case is when the ellipse has two similar axes (close to a circle, which is a special case of ellipse equation).

Fig. 5 compares errors and uncertainty of one feature's individual parameter. Firstly, considering the theoretical cases where no noise exists in the observation, error of each parameter is always zero without any doubt (See Fig. 5(a) - Fig. 5(d)). A remarkable part is that all the errors are strictly within the scope of their corresponding uncertainty except circular feature's angle which is not zero and the corresponding uncertainty is marked as infinite. It is caused by the same dimension of major and minor axis that deriving angle turns to be unreliable. Noise cannot be ignored when a robot is handled in real world (See Fig. 5(e) - Fig. 5(h)). As we have analyzed the uncertainty transmission above, uncertainty caused by sensors are evidently transmitted to the fitted parameters. All the errors are significantly limited in the range of calculated uncertainty. Even for a badly fitted result (Fig. 5(g)), the fitted ellipse diverges from the exact model but all the errors are reasonably in the range of uncertainty.

4) DATA ASSOCIATION

Data association is a difficult problem in SLAM, especially in certain complicated environment. When a robot works in an open environment, there are two cases where observations from lidar sensors do not always occur: one is no object exists within a valid lidar range, another is no acceptable feature parameters fitted in one single observation.

Due to the sparse observation distribution, a valid odometry information is needed to handle such no-observation situation. Then data association is easily done with the odometry information since features are widely dispersed. Also if features in one single observation are relatively dense, it is still

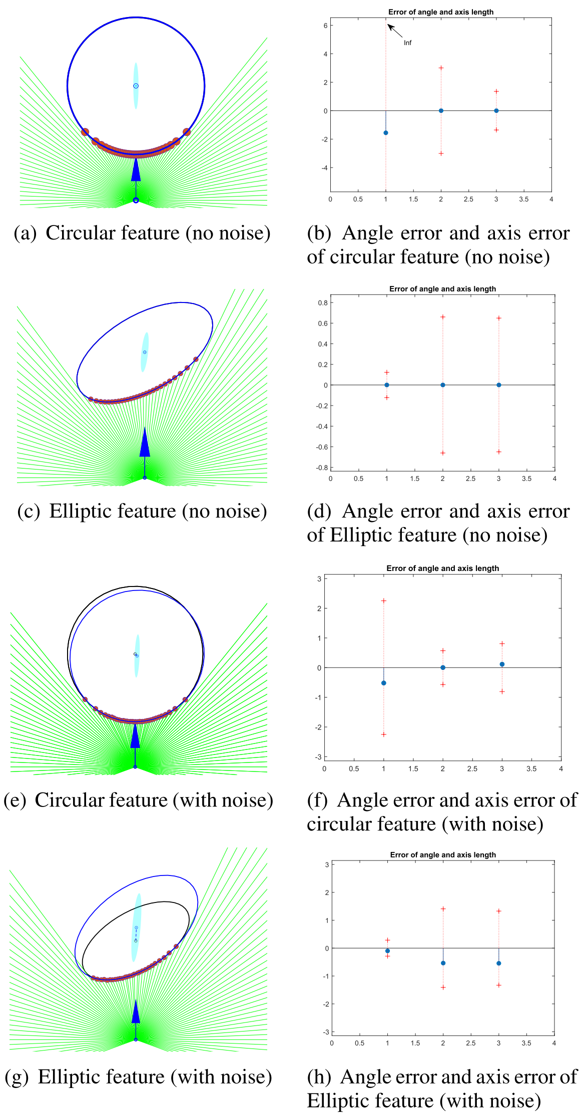


FIGURE 5. Uncertainty after fitting process. Left side figures show the error and uncertainty of translation (Error is depicted by dash lines, uncertainty is depicted by light blue elliptical range). Right side figures show the error and uncertainty of angle and axis dimension (From left to right each bar is corresponded of angle, major axis, and minor axis respectively).

not hard to make association. Different from conventional lidar SLAM, each feature defined in our parametrization possesses a center, an angle and a pair of geometry dimensions, and these parameters can be taken into account if correspondences are found, since using pure points is more complicated to solve nearest neighbor for the sake of large size and dense distribution. Noted that if no valid fitted features appear at a certain step, this step is marked by "no observation" and no edges are added between this step's node and other feature poses' node (discussed in Section II-B). Then our approach can overcome the challenge of no consecutive observations.

As is shown in Fig. 6, a newly fitted feature as well as odometry information is considered at the same time to determine whether this feature appeared or not. If this newly

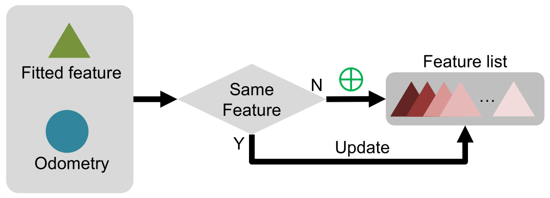


FIGURE 6. Flow chart of data association.

fitted feature appears for the first time (not the same feature in the feature list), it is added to the feature list. Otherwise, the feature list will update existing features only.

B. GRAPH OPTIMIZATION

This section focuses on the back-end optimization of the proposed algorithm. Thanks to the data processing section we are provided with an initial graph of robot poses and features. The remainder of this section briefly introduces factor graph SLAM for our problem.

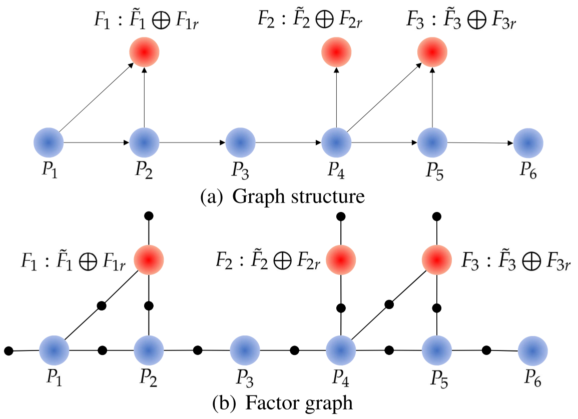

$F_1 : \tilde{F}_1 \bigoplus F_{1r}$
$F_2 : \tilde{F}_2 \bigoplus F_{2r}$
$F_3 : \tilde{F}_3 \bigoplus F_{3r}$
P_1
P_2
P_3
P_4
P_5
P_6
(a) Graph structure
(b) Factor graph

FIGURE 7. Optimization structure.

When a robot is carried in a 2D space, its state vector can be described by $P = (x, y, \theta)$. Remarkably, according to the parametrization discussed above, we form the first three parameters $[F_x, F_y, F_\phi]$ as the feature's "pose" $\tilde{F}$ with geometry properties F_r. With our feature parametrization approach each feature can be re-expressed as a feature pose and a dimensional part denoted by $F : \tilde{F} \bigoplus F_r$. Let us assume a simplified structure (Fig. 7(a)). Blue nodes are robot poses and red nodes are features, and each observation is represented by an arrow edge. When express this structure via factor graph (Fig. 7(b)), one observation are represented by an edge with a black point. Edges between robot pose follow a motion model with the input $u = (\delta x, \delta y, \delta\theta)$ and edges linking features and poses follow an observation model similarly. Noticed that every feature node is connected with an isolated factor, which means prior probability density for a feature should be provided if the robot needs to locate in a given map. In our case, such factors can be neglected.

In the factor graph $\mathcal{F} = (\mathcal{M}, \mathcal{X}, \varepsilon)$, we can denote factors, variables and edges as $\phi_i \in \mathcal{M}$, $x_i \in \mathcal{X}$ and $e_{ij} \in \varepsilon$, respectively. Writing all of the variables for an assignment to the set X_i, we can define the global factorized optimization problem of the example as:

$$\text{argmax } \phi(X) = \prod_i \phi_i(X_i) \tag{17}$$

where

$$\begin{aligned}\phi(X) &= \phi(F_1, F_2, F_3, P_2, P_3, P_4, P_5, P_6)\\ &= \phi_1(P_1)\,\phi_2(P_2, P_1)\,\phi_3(P_3, P_2)\\ &\quad \phi_4(P_4, P_2)\,\phi_5(P_5, P_2)\,\phi_6(P_6, P_2)\\ &\quad \phi_7(F_1, P_1)\,\phi_8(F_1, P_2)\,\phi_9(F_2, P_4)\\ &\quad \phi_{10}(F_3, P_4)\,\phi_{11}(F_3, P_5)\end{aligned} \tag{18}$$

It is easy to extend Eq. (17) to a general global optimization function which can be solved by general non-linear least squares method like Gauss-Newton method.

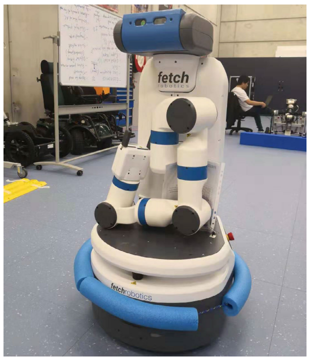


FIGURE 8. Fetch robot platform.

III. EXPERIMENTS

A. EXPERIMENT OVERVIEW

The experiment overview and results are interpreted in this section. Not only did we build the corresponding simulation environment, but we also experimented in the real scene. The platform in our experiment is the Fetch robot [43] (Fig. 8). It is equipped with a SICK 2D laser scanner at 15 Hz and has a 220 degree field of view with an angular resolution of 0.3323 degree and a 25 meter valid range. The simulator is designed for Fetch robot including Fetch model and working environment. All the simulation parameters are set the same as a real Fetch while we assumed the observation noise and odometry noise obey zero mean Gaussian distribution $n_s \sim \mathcal{N}(\mathbf{0}, \Sigma_\mathbf{s})$ and $n_o \sim \mathcal{N}(\mathbf{0}, \Sigma_\mathbf{o})$ respectively, which are supposed to be similar to the real robot's noise. During the simulation $\Sigma_\mathbf{s}$ was set to (0.02m, 0.02m) with regard to $(\delta x, \delta y)$ for the laser point in Cartesian coordinate, and $\Sigma_\mathbf{o}$ was set to (0.05m, 0.05m, 0.001rad) with regard to $(\delta x, \delta y, \delta\theta)$ for the odometry.

We built three different simulated environments and one real scenario to test our algorithm and compared with common algorithms, namely Cartographer [13], ICP [44] and NDT [10]. We also conducted experiments to analyze the uncertainty during fitting. Furthermore, we compared the uncertainty via different optimization methods. In our case, the proposed algorithm is adapted to EKF approach as a control group.

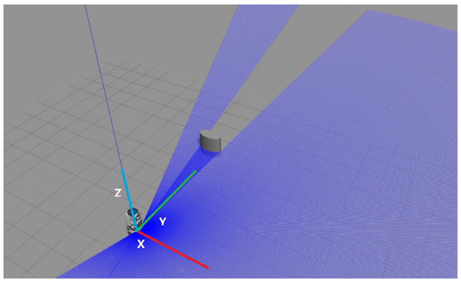


FIGURE 9. Schematic diagram of uncertainty analysis experiment.

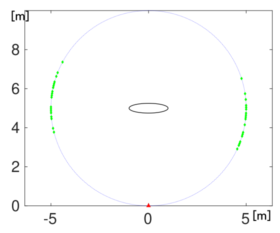
TABLE 1. Average error percentage of estimated axis dimension for different features. (Unit: m).

		Feature 1	Feature 2	Feature 3	Feature 4
F_{r1}	Groundtruth	1	1	1	1
	Estimate	1.0117	1.0027	0.9976	1.0018
	Error %	1.17%	0.27%	0.24%	0.18%
F_{r2}	Groundtruth	0.25	0.5	0.75	1
	Estimate	0.2518	0.5015	0.7489	0.9991
	Error %	0.72%	0.3%	0.15%	0.09%

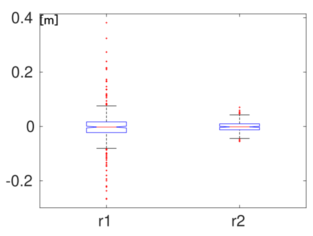
B. ANALYSIS OF FEATURE FITTING

We tested the fitting process at various observing angles since the moving robot cannot observe objects ideally. In this test, Fetch robot made a counter-clockwise circular motion around the object 3 meters from the robot on y-axis (As shown in Fig. 9). Because the dimension of two axes are able to judge the fitting performance intuitively, we only compare F_{r_1} and F_{r_2} in this case. Table 1 shows the average error percentage of different dimensional features: Feature 1 $\sim$ (F_{r_1} = 1m, F_{r_1} = 0.25m), Feature 2 $\sim$ (F_{r_2} = 1m, F_{r_2} = 0.5m), Feature 3 $\sim$ (F_{r_3} = 1m, F_{r_3} = 0.75m), and Feature 4 $\sim$ (F_{r_4} = 1m, F_{r_4} = 1m). The fitting process in each case was performed for 15 times before the final results were obtained.

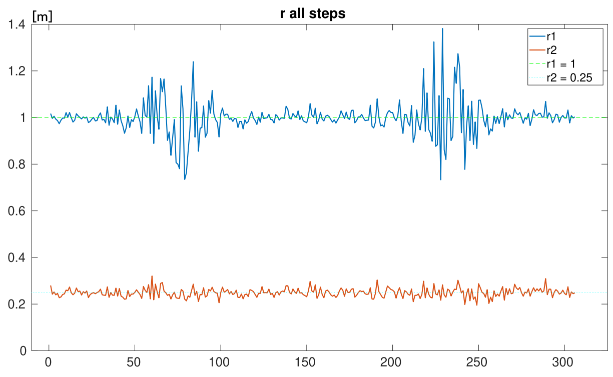
It can be seen errors of four features are reasonably small. The average error percentage of F_{r_1} and F_{r_2} descends with the decreasing of axis ratio $\tau = \frac{F_{r1}}{F_{r_2}}$. Feature 1 has the largest τ and the biggest error percentage. As is shown in Fig. 10(c), F_{r_1} has two distinct growth with the increase of steps, but F_{r_2} doesn't represent the similar regularity. According to Fig. 10(b), errors of F_{r_2} are not significantly large compared with F_{r_1}. If marking positions where the error percentage of F_{r_1} is over 10% with green diamond markers, we can obviously see that these large errors are mostly found at positions where the robot observes feature's narrow end. Fig. 10(d) illustrates error and 3-sigma bounds among all steps. As we can see, the error of F_{r_1} reaches the maximum around 1.5m at the 90*th* step but it is still located within 3-sigma bounds as well as errors of all the other steps.



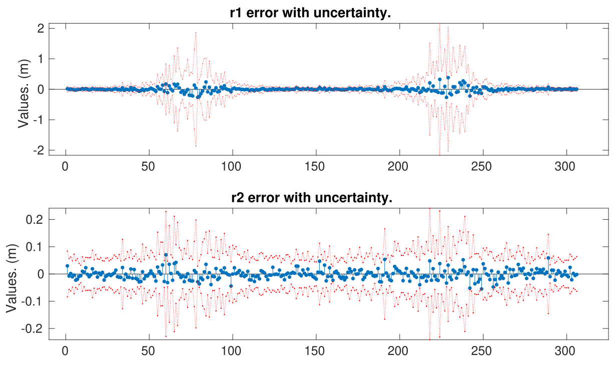
(a) Positions when error percentage is over 50%



(b) Error distribution for r1 and r2

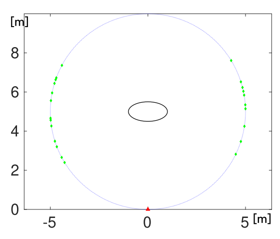


(c) Dimensions of F_{r_1} and F_{r_2} each step in one loop

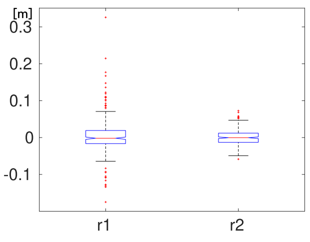


(d) Error and 3-sigma bounds at each step in one loop

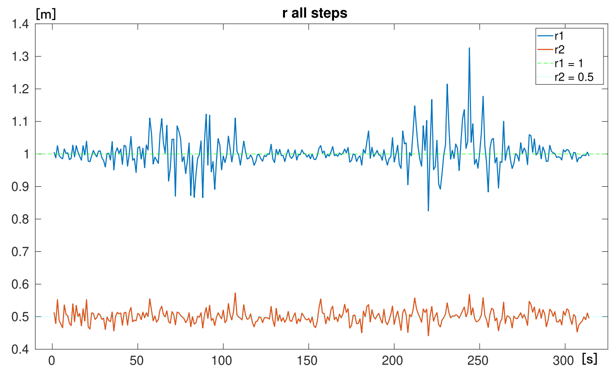
FIGURE 10. Error of F_{r_1} and F_{r_2} for Feature 1.



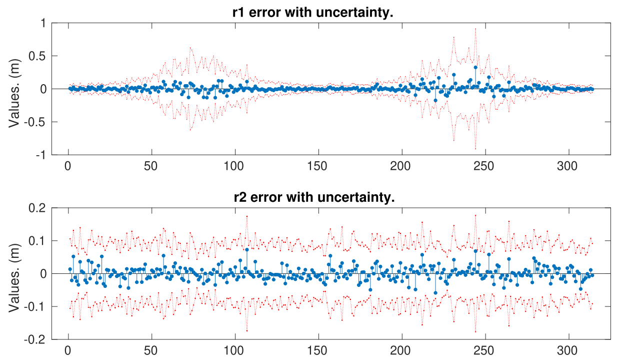
(a) Positions when error percentage is over 50%



(b) Error distribution for r1 and r2



(c) Dimensions of F_{r_1} and F_{r_2} each step in one loop



(d) Error and 3-sigma bounds at each step in one loop

FIGURE 11. Error of F_{r_1} and F_{r_2} for Feature 2.

Fig. 11(a), Fig. 12(a) and Fig. 13(a) show the error occurrence for Feature 2, Feature 3, and Feature 4 respectively. It can be seen that the error-prone positions are most likely occurred when observing the narrow end of features, but the probability of large-error occurrence and the value of errors descend with the decrease of τ. By comparing all the four features' error and 3-sigma bounds, it can be found that Feature 1 possesses the largest errors for F_{r_1} and F_{r_2} by around 0.5m and 0.1m (Fig. 10(d)), Feature 2's largest errors locate at 0.29m and 0.09m (Fig. 11(d)), Feature 3 possesses the largest errors for F_{r_1} and F_{r_2} by around 0.17m and 0.08m (Fig. 12(d)) and Feature 4 has the largest errors for 0.1m and 0.1m (Fig. 13(d)). All the errors are strictly limited within the 3-sigma bounds.

C. SLAM SIMULATION RESULTS

In this section, we constructed multiple expected working scenarios (as shown in Fig. 14), and compared our method with the state of the art 2D SLAM system Cartographer [13], and other widely used algorithm: ICP without an initial guess (set initial guess to zero, denoted by ^{0}ICP), ICP with a good initial guess (set initial guess to the odometry value, denoted by ^{1}ICP), NDT without an initial guess (set initial guess to zero, denoted by ^{0}NDT), and NDT with a good initial guess (set initial guess to the odometry value, denoted by ^{1}NDT). Noted that the results of Cartographer is under the configuration ''using odometry information'', while the other four approaches doesn't utilize odometry. We evaluated our algorithm with other methods by comparing difference with ground truth in x, y, θ via Root-Mean-Square-Error (RMSE) and error per step. In order to clear the advantages of factor graph optimization in our algorithm, a general extended Kalman filter based SLAM algorithm [45] was adopted as a controlled group. Different from common EKF SLAM, the state vector is composed of current pose and parameterized features under the parametrization of the proposed

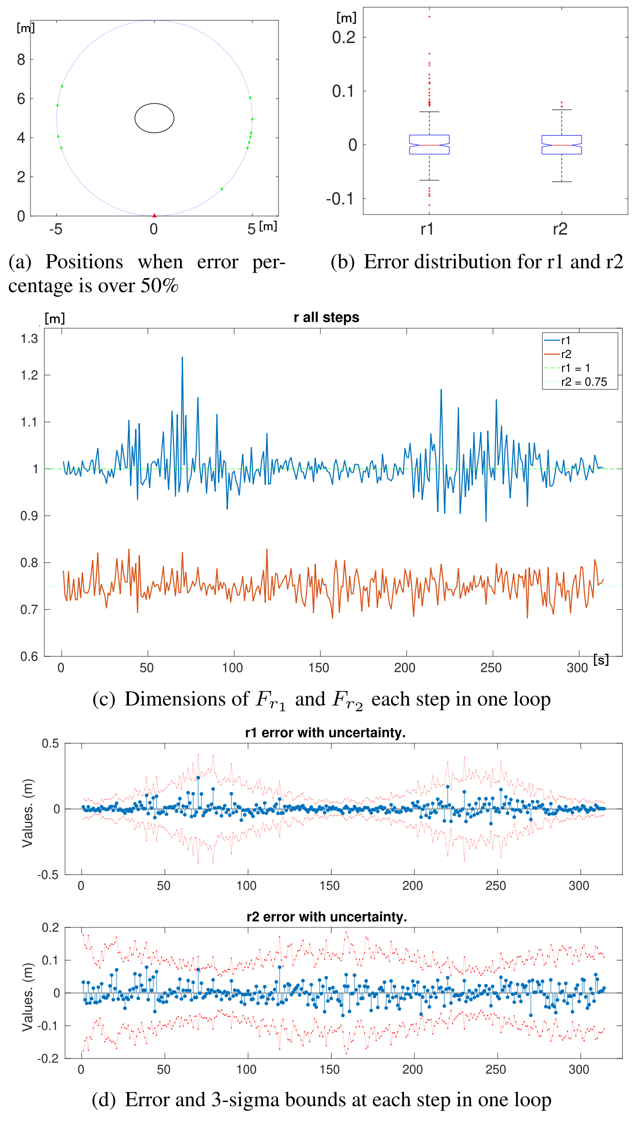

(a) Positions when error percentage is over 50%
(b) Error distribution for r1 and r2
(c) Dimensions of F_{r_1} and F_{r_2} each step in one loop
(d) Error and 3-sigma bounds at each step in one loop

FIGURE 12. Error of F_{r_1} and F_{r_2} for Feature 3.

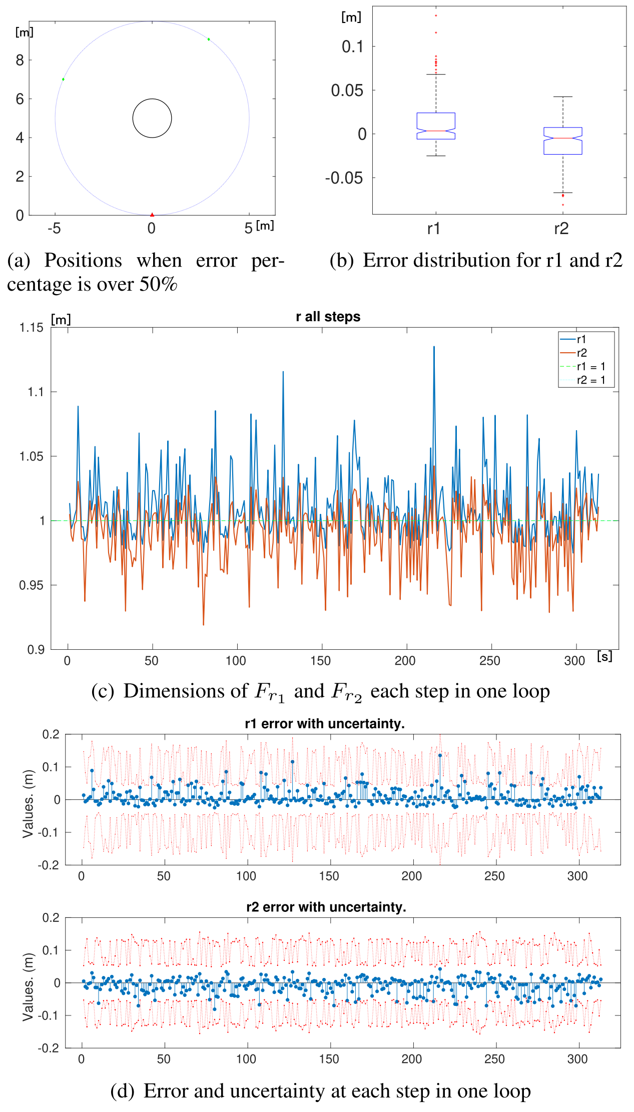

(a) Positions when error percentage is over 50%
(b) Error distribution for r1 and r2
(c) Dimensions of F_{r_1} and F_{r_2} each step in one loop
(d) Error and uncertainty at each step in one loop

FIGURE 13. Error of F_{r_1} and F_{r_2} for Feature 4.

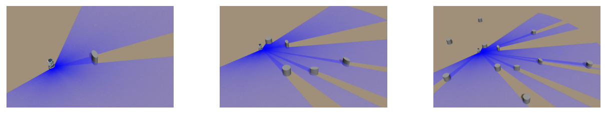

(a) Case 1: Single feature.
(b) Case 2: Five features.
(c) Case 3: Eleven features.

FIGURE 14. Simulation environment. Case 1 contains one single feature, the robot moves around the feature. Case 2 contains five features, the robot moves around all the features. Case 3 contains eleven features, the robot moves though the features.

algorithm (Denoted by EKF directly). Remarked that estimated features at the last step as well as estimated pose at each individual step are chosen for purposes of comparison.

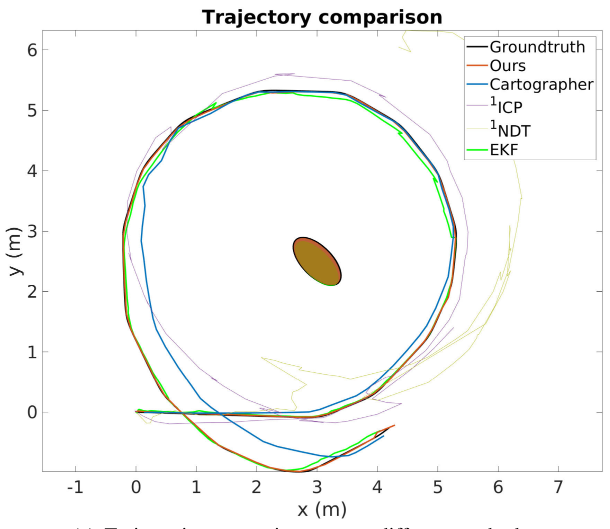
Table 2 gives groundtruth of feature parameters in each case. F_ϕ is denoted by "-" if that feature is circular shaped. All the features are built in Gazebo. To enhance the visualization performance, real feature is filled with black shadow and estimated result by our method is filled with orange color in the trajectories comparison figures (Fig. 15(a), Fig. 16(a), Fig. 17(a), and Fig. 21).

In Case 1 (Fig. 14(a)), the robot moved around one single elliptical feature. Trajectory comparison is illustrated in Fig. 15(a). ^{0}ICP and ^{0}NDT are not depicted for the sake of completely wrong results. A turn back exists in the trajectory of ^{1}NDT which is caused by the similar shape at both sides. ^{1}ICP is better than ^{1}NDT but is still worse than Cartographer. The trajectory of our method is the closest to the groundtruth. Fig. 15(b) demonstrates the estimated error of robot pose in δx, δy and $\delta\theta$ varying with time. Both ICP and NDT cannot provide reasonable result, the maximum errors of δx and δy exceed 4 meters and the variation of rotation error is even greater. Cartographer and our method can maintain the error within a small range. In the dash rectangle we enlarged part of the error curve from 50s to 60s. It can be found the absolute translation error of Cartographer is around 0.4m while ours is within 0.1m. The peak of rotation error of both Cartographer

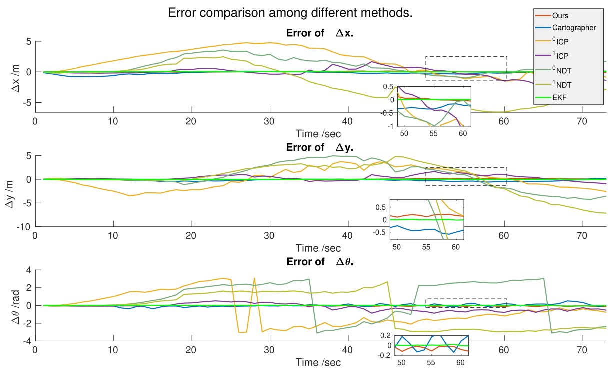
TABLE 2. Feature parameters ground truth.

		F_x /m	F_y /m	F_ϕ /rad	F_{r_1} /m	F_{r_2} /m
Case 1	F_1	3	2.5	2.3562	0.5	0.25
Case 2	F_1	3	2.5	2.3562	0.5	0.25
	F_2	6	-2.4	-	0.5	0.5
	F_3	9	-1	0.5	0.5	0.25
	F_4	12	2.4	2.7416	0.5	0.25
	F_5	0	2	-	0.5	0.5
Case 3	F_1	9	-1	0.5	0.5	0.25
	F_2	6	-2.4	-	0.5	0.5
	F_3	7	12	-	0.5	0.5
	F_4	3	2.5	2.3562	0.5	0.25
	F_5	0	2	-	0.5	0.5
	F_6	0	-8	-	0.5	0.5
	F_7	12	2.4	2.7416	0.5	0.25
	F_8	13	-6	-	0.5	0.5
	F_9	20	5.5	-	0.5	0.5
	F_{10}	-9	12	-	0.5	0.5
	F_{11}	-8	0	-	0.5	0.5

* F_ϕ is denoted by "-" if that feature is circular shaped.



(a) Trajectories comparison among different methods.

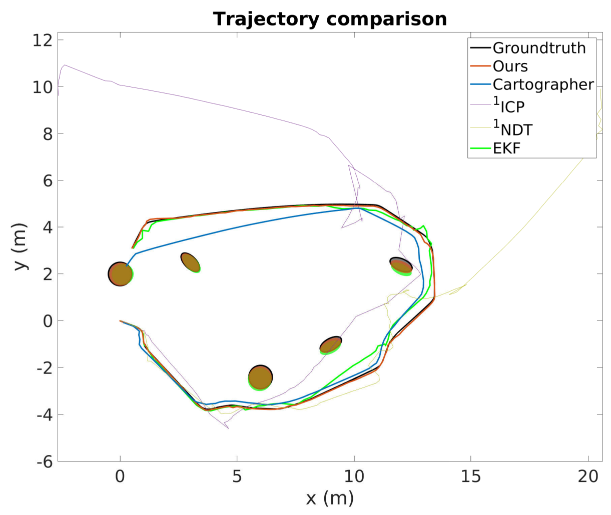


(b) Case 1: Error comparison among different methods.

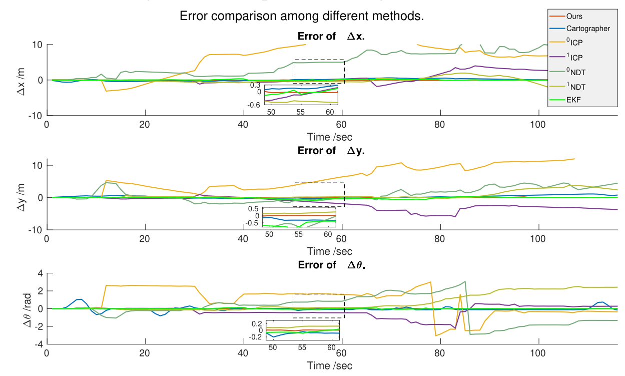
FIGURE 15. Case 1: Trajectory and error varying with time.

and our method can reach 0.2rad but it is clearly seen that ours has a lower average level than that of Cartographer.

In Case 2 (Fig. 14(b)), the robot moved around five features including circular and elliptical shape. Trajectory comparison



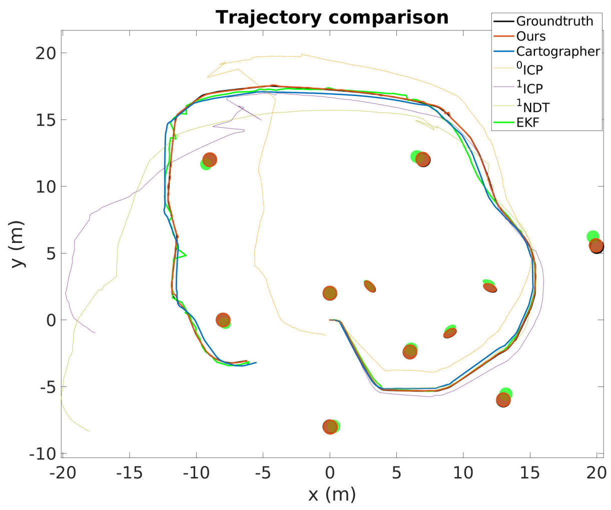
(a) Trajectories comparison among different methods.



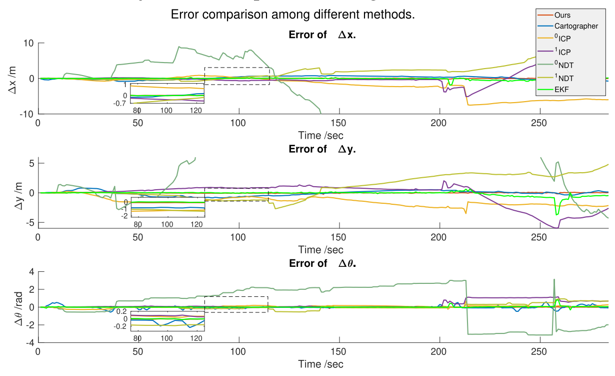
(b) Error comparison among different methods

FIGURE 16. Case 2: Trajectory and error varying with time.

is illustrated in Fig. 16(a). ^{0}ICP and ^{0}NDT are not depicted for the sake of completely wrong results. ^{1}ICP and ^{1}NDT still perform badly. The trajectory of our method is the closest to the ground truth. There is a relatively large jump in the Cartographer's trajectory, the reason of which is that the observed scan points are located on the other side of features in contrast to previous observation. Thus the registration process considers point sets on both side as the same during scan matching. Fig. 16(b) demonstrates the estimated error of robot pose in δx, δy and $\delta\theta$ varying with time. Both ^{0}ICP and ^{0}NDT cannot provide reasonable result, where the translation error and rotation error are too large. The results in the first half part of ^{1}ICP and ^{1}NDT are roughly near the real trajectory, but in the remainder part they diverged because of wrong matching. Cartographer and our method can maintain the error within a small range. In the dash rectangle we enlarged part of the error curve from 50s to 60s. It can be found clearly that our error is less than that of Cartographer. The peak of rotation error of Cartographer is even beyond 0.2rad while ours keep the error level stick to near 0. It is worth saying that ^{1}NDT possesses a smaller rotation error and δy error than Cartographer dramatically, due to the increased amount of features.



(a) Trajectories comparison among different methods.

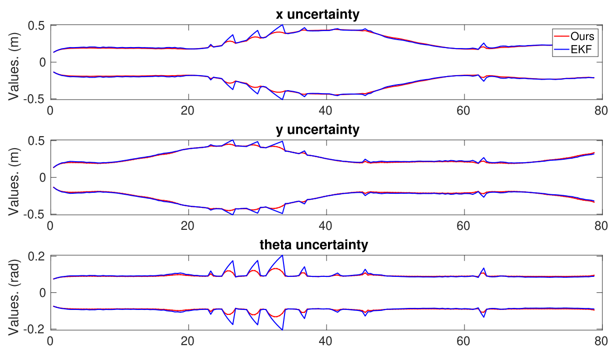


(b) Error comparison among different methods.

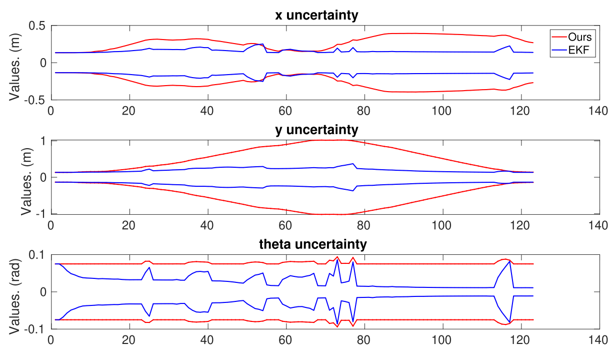
FIGURE 17. Case 3: Trajectory and error varying with time.

In Case 3 (Fig. 14(c)), the robot moved through and around eleven features including circular and elliptical shape. Trajectory comparison is illustrated in Fig. 17(a). ^{0}NDT is not depicted for the sake of completely wrong result. In this case ^{0}ICP, ^{1}ICP and ^{1}NDT are partly trusty when the robot can observe features from both left and right sides. Then the trajectories starts to drift since it can only observe features from a single side. Our trajectory is still the nearest to the groundtruth and Cartographer in this case performs the best compared with the other 2 cases. Fig. 17(b) demonstrates the pose error of the robot in $\delta x, \delta y$ and $\delta\theta$ varying with time. ^{0}NDT in this case is the method that generate the worst result. In the dash rectangle we enlarged part of the error curve from 80s to 120s. Our method's error is still the smallest one.

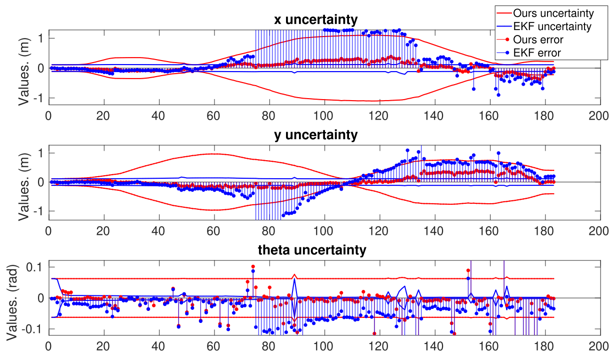
We also utilized EKF method to evaluate the accuracy of importing factor graph. From Fig. 15 to Fig. 17 it can be seen directly that our method via factor graph possesses a more accurate result compared with EKF. In order to demonstrate the superiority of factor graph, we depicted the uncertainty comparison between EKF frame and factor graph frame as is shown in Fig. 18. Obviously the uncertainty curve of factor graph performs more continuously and smoothly than that of EKF. It is worth noting that although in Case 2 and Case 3 the



(a) Case 1



(b) Case 2



(c) Case 3

FIGURE 18. Pose 3-sigma bounds comparison between EKF and factor graph. From top to bottom at each sub graph illustrates the uncertainty of x, y and θ.

uncertainty of factor graph exceeds that of EKF and the θ uncertainty of EKF is dramatically small in Case 3 compared with that of factor graph, we still cannot regard that EKF is more accurate than factor graph. The reason has been proved in [45]:

- *The inconsistency of EKF SLAM may cause the variance of the robot orientation estimate to be incorrectly reduced to zero.*
- *The linearization process of EKF SLAM can introduce errors to make the estimated uncertainty smaller or larger than true uncertainty.*

The error of each pose is depicted as is shown in Fig. 18(c). It can be found that errors of our approach are distributed within the uncertainty range, while EKF's errors exceed the corresponding uncertainty greatly even if the uncertainty looks fairly small.

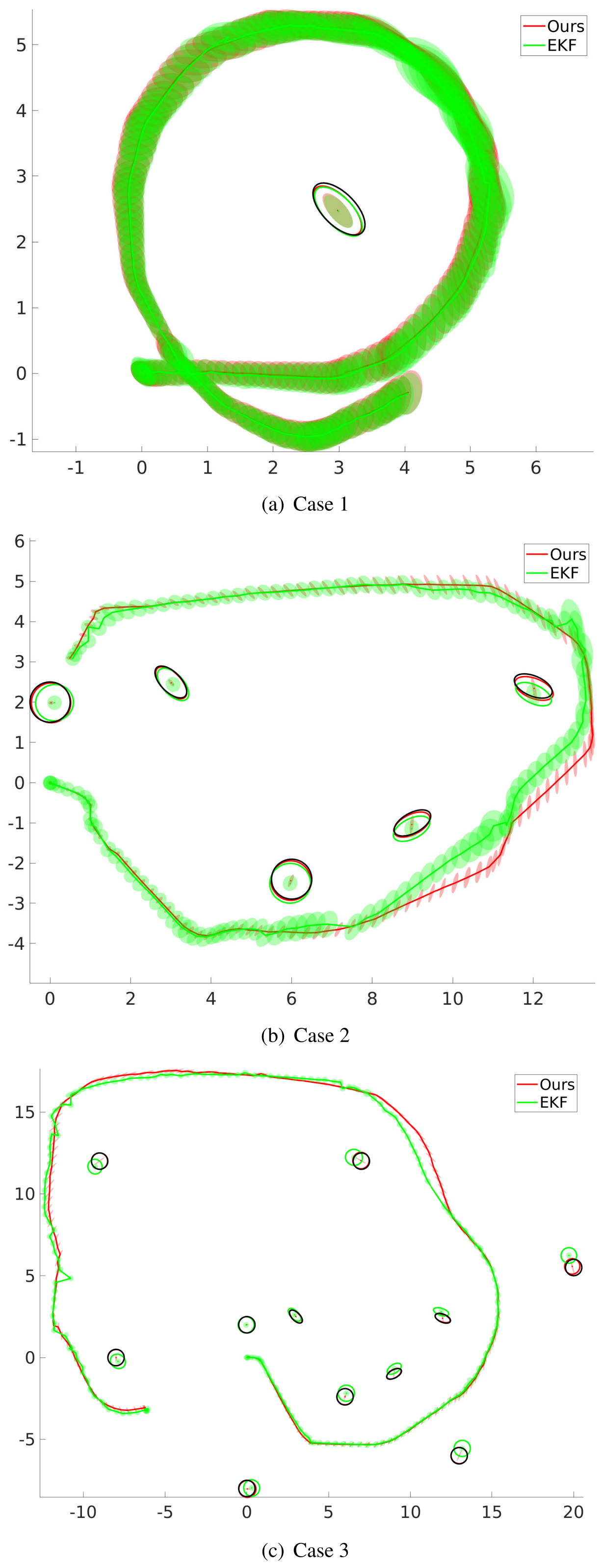


FIGURE 19. **Uncertainty comparison displayed in the map for each case.**

The uncertainty of poses and features in the map is displayed as is shown in Fig. 19 (black ellipses are true features). Two conclusions can be drawn: (a) If only one single feature exists, the performance of factor graph and EKF are slightly different, but the difference is indeed small. (b) If there are more than one feature in the environment, factor graph can obtain more accurate estimate compared with EKF especially when loop closure happens.

Table 3 provides the RMSE for all the methods with respect to three cases. Table 4 lists feature parameters error comparing estimated features obtained by our method with ground truth.

It can be concluded from Table 3 that our method possesses the minimal RMSE for three simulated cases except for θ in Case 1 and Case 2, while Cartographer performs better than the other four methods. Nevertheless, the difference between our method via factor graph and EKF for Case 1 and Case 2 are quite minor that the difference is no more than 0.007rad. Another conclusion is NDT method is less adapted to the open environment than ICP method because of sparsely distributed sensor data. Fortunately, a good initial guess for both ICP and NDT can improve the accuracy significantly, but they cannot reach Cartographer's accuracy. By comparing errors in Cartographer and our method from Case 1 to Case 3, the accuracy of our method enhanced with the increasing feature amounts, while Cartographer is not influenced. This phenomenon is due to the compact graph structure as we associate data before graph optimization, which makes the result robust and accurate.

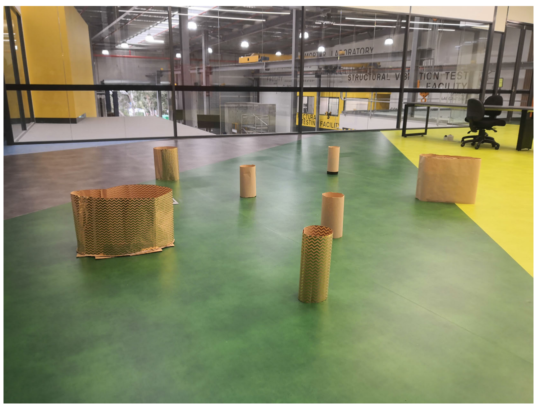


FIGURE 20. **Real world environment.**

D. REAL EXPERIMENT

In this section, we conducted a real world environment (Fig. 20) with 7 features surrounded by glass walls. Laser data is not reliable hitting transparent glass. The origin position is manually measured as well as features' positions with respect to the predetermined coordinate at the origin point, and the accuracy of measurement is within 0.1m. Because we only have odometry information and features' manually measured position, we did not compare pose errors. Instead, the odometry and measured features' position were used to roughly distinguish the trajectory and evaluate the mapping performance.

Fig. 21 depicts trajectories obtained by our method and Cartographer. Both ICP and NDT failed in obtaining an

TABLE 3. RMSE comparison.(Unit: m).

Dataset Method	Case 1 x/m	 y/m	 θ/rad	Case 2 x/m	 y/m	 θ/rad	Case 3 x/m	 y/m	 θ/rad
Ours	**0.0906**	**0.0918**	0.0552	**0.0771**	**0.0527**	0.0416	**0.0458**	**0.0486**	**0.0147**
Cartographer	0.3015	0.2778	0.1550	0.3176	0.4774	0.2546	0.4851	0.4311	0.1575
^{0}ICP	2.4685	2.1470	1.7136	7.0788	9.8747	1.4655	3.5194	1.6207	0.1391
^{1}ICP	0.8192	0.7204	0.4227	1.9770	3.4728	0.6113	3.7548	1.9173	0.4943
^{0}NDT	1.7986	2.7243	2.1332	5.2144	4.3799	1.3632	27.9214	17.6822	2.1198
^{1}NDT	0.7255	0.6489	0.3167	7.7872	1.9755	1.5936	4.1918	2.2244	0.3292
EKF	0.1074	0.1159	**0.0488**	0.1081	0.2866	**0.0400**	0.2932	0.4427	0.0834

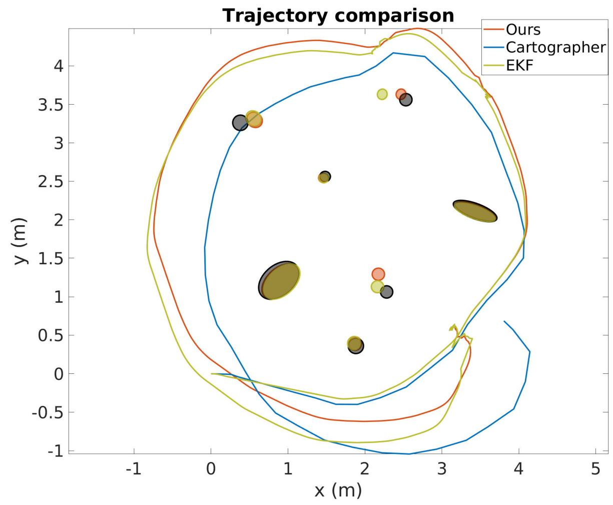
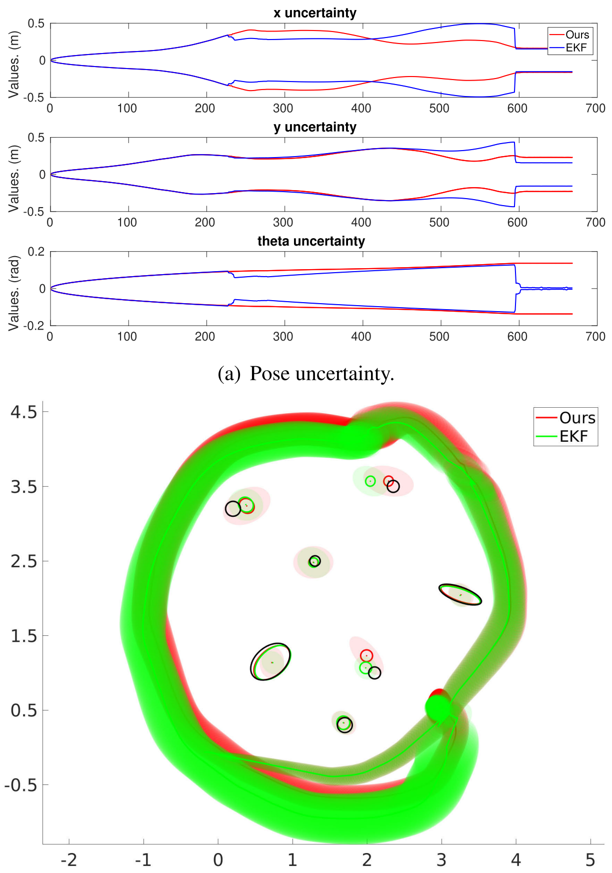


FIGURE 21. Real world: Trajectories comparison among different methods.

acceptable solution. Real features are plotted in grey shadow via rough measurement, and orange features are the estimated result by our method. Some estimated features are around 0.5 meter from the actual corresponding features, and the axes dimension of one elliptical feature shrinks. But the trajectory of Cartographer is obviously untrusted because it goes through a feature. Our method performs better in this case.

Fig. 22(a) depicts pose uncertainty comparison between our method and EKF approach. Same phenomenon occurs resembling simulation experiments. Fig. 22(b) shows the "real" features lie in the uncertainty range of ours, while one estimated feature by EKF exceeds the reliable range.

We also evaluated our approach on public dataset Victoria Park [46] as is shown in Fig. 23. However, cartographer cannot be adopted on this dataset. Hence, we only compared our method with a point feature based approach [47] which is marked by blue line. GPS data of the dataset is marked by black dot, and our method is expressed by red line. Red ellipses (which look like red points because of the scaled display) are estimated features by our approach. It is not easy to evaluate the accuracy of trajectory quantitatively since the GPS data is untrustworthy. Also, if looking at the turn on the right it can be seen that our method drifts a little compared with point feature SLAM. This is because features at that turn



(a) Pose uncertainty.

(b) Uncertainty comparison displayed in the map.

FIGURE 22. Uncertainty comparison between factor graph and EKF.

are rare which makes the ellipse fitting process unstable. But the performance is better in the middle part and the left part since features are trustful.

E. MAPPING QUALITY

In this section we compared the mapping performance between Cartographer and our method. Mapping by Cartographer is an occupancy grid map which is widely used in robotics algorithms. However, the accuracy of occupied grid map is affected by the size of the grid. The mapping representation of our method directly expresses features with conic equation. The advantages include three aspects: 1. Mapping is continuous so that mapping accuracy won't be influenced by

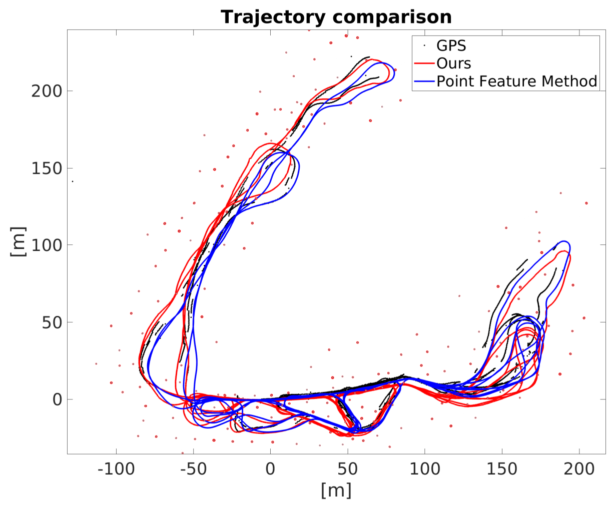
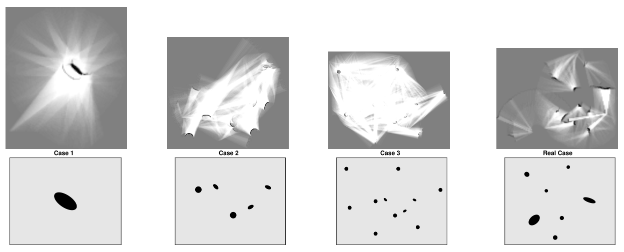


FIGURE 23. Evaluated on Victoria Park.

the grid size. 2. The map needs less memory and the memory required is only related with the number of features. 3. The representation is human friendly and easy for visualization.



(a) Case 1 (b) Case 2 (c) Case 3 (d) Real case

FIGURE 24. Maps by Cartographer (the first row) and our method (the second row). (a) Case 1; (b) Case 2; (c) Case 3; (d) Real world.

Fig. 24 compares mapping performance of Cartographer and ours. The first row is obtained by Cartographer and the second row is by our method. It is clear that our method can obtain a good map. In the first row of Case 1, Case 2, and Real Case, it is easy to find abundant duplicated and overlapping curves which should be assembled to the same feature. But Cartographer did well for Case 3 because peripheral features are observed on one side which will not introduce too much error into the optimization problem.

We also compared the estimated features with ground truth (Ground truth: Case 1 to Case 3. Rough measurement: Real experiment) to check the accuracy numerically and to make figures friendly to read we filled real features and estimated results with black and orange shadow respectively in the trajectories comparison figures (Fig. 15(a), Fig. 16(a), Fig. 17(a), and Fig. 21). As is shown in Table 4, for simulation experiments, the accuracy of majority estimated features is around 2 centimeters except a few large error terms. But even these large terms are within 10 centimeters. However, real experiment performs worse than simulations, the largest error term reaches 20 centimeters (As shown in Fig. 21,

TABLE 4. Features absolute error.

		ΔF_x /m	ΔF_y /m	ΔF_ϕ /rad	ΔF_{r_1} /m	ΔF_{r_2} /m
Case 1	F_1	0.0019	0.0036	0.0123	0.0060	0.0055
Case 2	F_1	0.0121	0.0148	0.0176	0.0018	0.0032
	F_2	0.0155	0.0356	-	0.0011	0.0002
	F_3	0.0127	0.0470	0.0045	0.0095	0.0025
	F_4	0.0102	0.0871	0.0228	0.0075	0.0189
	F_5	0.0283	0.0123	-	0.0029	0.0101
Case 3	F_1	0.0039	0.0284	0.0121	0.0021	0.0011
	F_2	0.0066	0.0137	-	0.0011	0.0017
	F_3	0.0426	0.0448	-	0.0043	0.0011
	F_4	0.0120	0.0187	0.0440	0.0045	0.0020
	F_5	0.0024	0.0149	-	0.0003	0.0051
	F_6	0.0293	0.0216	-	0.0146	0.0117
	F_7	0.0158	0.0276	0.0316	0.0139	0.0049
	F_8	0.0017	0.0310	-	0.0028	0.0061
	F_9	0.0684	0.0887	-	0.0079	0.0124
	F_{10}	0.0028	0.0048	-	0.0062	0.0015
	F_{11}	0.0073	0.0016	-	0.0003	0.0002
Real	F_1	0.1074	0.2297	-	0.0010	0.000
	F_2	0.0629	0.0708	-	0.0121	0.0169
	F_3	0.1838	0.0381	-	0.0062	0.0065
	F_4	0.0223	0.0180	-	0.0070	0.0074
	F_5	0.0210	0.0125	-	0.0150	0.0259
	F_6	0.0132	0.0297	-	0.0102	0.0127
	F_7	0.0089	0.0039	0.0107	0.0168	0.0128

two estimated features are far away from the measured true features). Also, mapping of the Victoria Park by Cartographer is not built due to the capability limitation, while our approach built the map shown in Fig. 23. Nevertheless, our method is able to provide a continuous, well-performed and robust map in contract to Cartographer.

IV. CONCLUSION

A conic feature based SLAM algorithm in open environment via 2D lidar was proposed in this paper. Tradition scan matching methods are not competent for working in an open environment where sufficient edges and corners do not exist. We proposed a conic feature based method to represent features and reformed corresponding graph structure instead of matching scan points in a traditional approach. First, the raw data was processed with the prior information of odometry to associate data. Then conic feature fitting was applied to transform points to the feature parametrization proposed in this paper. At last a factor graph optimization was adopted to obtain pose estimates as well as the map in our representation. Simulation experiments and real environment test demonstrated that our proposed SLAM algorithm can get accurate and convincing results for the open environment and the map in our representation can accurately express the environment situation.

In the future work, we will utilize more types of geometric information in a general form to solve SLAM problem and verify on more public datasets.

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

[2] J. Yin, L. Carlone, S. Rosa, M. L. Anjum, and B. Bona, "Scan matching for graph SLAM in indoor dynamic scenarios," in *Proc. 27th Int. Flairs Conf.*, May 2014.

[3] J. Song, J. Wang, L. Zhao, S. Huang, and G. Dissanayake, "MIS-SLAM: Real-time large-scale dense deformable SLAM system in minimal invasive surgery based on heterogeneous computing," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 4068–4075, Oct. 2018.

[4] J. Song, J. Wang, L. Zhao, S. Huang, and G. Dissanayake, "Dynamic reconstruction of deformable soft-tissue with stereo scope in minimal invasive surgery," *IEEE Robot. Autom. Lett.*, vol. 3, no. 1, pp. 155–162, Jan. 2018.

[5] E. Pedrosa, A. Pereira, and N. Lau, "Efficient localization based on scan matching with a continuous likelihood field," in *Proc. IEEE Int. Conf. Auto. Robot Syst. Competitions (ICARSC)*, Apr. 2017, pp. 61–66.

[6] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," *Proc. SPIE, Sensor Fusion IV, Control Paradigms Data Struct.*, vol. 1611, pp. 586–606, 1992.

[7] G. C. Sharp, S. W. Lee, and D. K. Wehe, "ICP registration using invariant features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 1, pp. 90–102, Jan. 2002.

[8] J. Yang, H. Li, and Y. Jia, "Go-ICP: Solving 3D registration efficiently and globally optimally," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1457–1464.

[9] B. Jian and B. C. Vemuri, "Robust point set registration using Gaussian mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1633–1645, Aug. 2011.

[10] P. Biber and W. Strasser, "The normal distributions transform: A new approach to laser scan matching," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, vol. 3, Oct. 2003, pp. 2743–2748.

[11] T. D. Stoyanov, M. Magnusson, H. Andreasson, and A. Lilienthal, "Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations," *Int. J. Robot. Res.*, vol. 31, no. 12, pp. 1377–1393, 2012.

[12] J. Zhao, S. Huang, and L. Zhao, "Constrained Gaussian mixture models based scan matching method," in *Proc. Australas. Conf. Robot. Autom.*, Dec. 2018.

[13] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d LIDAR SLAM," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2016, pp. 1271–1278.

[14] E. Olson, "M3RSM: Many-to-many multi-resolution scan matching," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 5815–5821.

[15] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent, "Efficient sparse pose adjustment for 2D mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 22–29.

[16] F. Martín, R. Triebel, L. Moreno, and R. Siegwart, "Two different tools for three-dimensional mapping: De-based scan matching and feature-based loop detection," *Robotica*, vol. 32, no. 1, pp. 19–41, Jan. 2014.

[17] L. Li, J. Liu, X. Zuo, and H. Zhu, "An improved MbICP algorithm for mobile robot pose estimation," *Appl. Sci.*, vol. 8, no. 2, p. 272, Feb. 2018.

[18] Z. Guo, B. Cai, W. Jiang, and J. Wang, "Feature-based detection and classification of moving objects using LiDAR sensor," *IET Intell. Transp. Syst.*, vol. 13, no. 7, pp. 1088–1096, Jul. 2019.

[19] C. Wang, Q. Shu, X. Wang, B. Guo, P. Liu, and Q. Li, "A random forest classifier based on pixel comparison features for urban lidar data," *ISPRS J. Photogram. remote Sens.*, vol. 148, pp. 75–86, Feb. 2019.

[20] S. A. Gargoum and K. El Basyouny, "A literature synthesis of lidar applications in transportation: Feature extraction and geometric assessments of highways," *GIScì. Remote Sens.*, vol. 56, no. 6, pp. 864–893, 2019.

[21] L. Zhang and B. K. Ghosh, "Line segment based map building and localization using 2d laser rangefinder," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, Apr. 2000, pp. 2538–2543.

[22] J. Li, R. Zhong, Q. Hu, and M. Ai, "Feature-based laser scan matching and its application for indoor mapping," *Sensors*, vol. 16, no. 8, p. 1265, 2016.

[23] G. Marjanovic and V. Solo, "On l_q optimization and matrix completion," *IEEE Trans. Signal Process.*, vol. 60, no. 11, pp. 5714–5724, Nov. 2012.

[24] G. Marjanovic and V. Solo, "ι_q sparsity penalized linear regression with cyclic descent," *IEEE Trans. Signal Process.*, vol. 62, no. 6, pp. 1464–1475, Mar. 2014.

[25] T. Bailey, J. Nieto, and E. Nebot, "Consistency of the fast SLAM algorithm," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2006, pp. 424–429.

[26] C. Zhang, L. Yong, Y. Chen, S. Zhang, L. Ge, S. Wang, and W. Li, "A rubber-tapping robot forest navigation and information collection system based on 2D LiDAR and a gyroscope," *Sensors*, vol. 19, no. 9, p. 2136, May 2019.

[27] R. Ren, H. Fu, and M. Wu, "Large-scale outdoor SLAM based on 2D LiDAR," *Electronics*, vol. 8, no. 6, p. 613, May 2019.

[28] R. Keicher and H. Seufert, "Automatic guidance for agricultural vehicles in europe," *Comput. Electron. Agricult.*, vol. 25, nos. 1–2, pp. 169–194, Jan. 2000.

[29] J. Tang, Y. Chen, A. Kukko, H. Kaartinen, A. Jaakkola, E. Khoramshahi, T. Hakala, and J. Hyyppä, M. Holopainen, and H. Hyyppä, "SLAM-aided stem mapping for forest inventory with small-footprint mobile LiDAR," *Forests*, vol. 6, no. 12, pp. 4588–4606, Dec. 2015.

[30] B. Benet and R. Lenain, "Multi-sensor fusion method for crop row tracking and traversability operations," in *Proc. Conf. AXEMA-EURAGENG*, 2017, p. 10.

[31] Y.-S. Shin, Y. S. Park, and A. Kim, "Direct visual SLAM using sparse depth for camera-lidar system," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, 2018, pp. 1–8.

[32] G. Jiang, L. Yin, S. Jin, C. Tian, X. Ma, and Y. Ou, "A simultaneous localization and mapping (SLAM) framework for 2.5D map building based on low-cost LiDAR and vision fusion," *Appl. Sci.*, vol. 9, no. 10, p. 2105, Aug. 2019.

[33] H. Xue, H. Fu, and B. Dai, "IMU-aided high-frequency lidar odometry for autonomous driving," *Appl. Sciences*, vol. 9, no. 7, p. 1506, Apr. 2019.

[34] M. Bosse and R. Zlot, "Continuous 3D scan-matching with a spinning 2D laser," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 4312–4319.

[35] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," *Robot. Sci. Syst.*, vol. 2, p. 9, Jul. 2014.

[36] W. Wu, C. Chen, Y. Cong, Z. Dong, J. Li, S. Li, W. Dai, and B. Yang, "Low-cost wheeled robot-borne laser scanning system for indoor and outdoor 3D mapping application," *Int. Arch. Photogram., Remote Sens. Spatial Inf. Sci.*, vol. 42, no. 2/W13, pp. 1–5, Jun. 2019.

[37] S. Thrun, "Learning occupancy grid maps with forward sensor models," *Auto. Robots*, vol. 15, no. 2, pp. 111–127, Sep. 2003.

[38] J. Strom and E. Olson, "Occupancy grid rasterization in large environments for teams of robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 4271–4276.

[39] G. Grisettiyz, C. Stachniss, and W. Burgard, "Improving grid-based SLAM with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2005, pp. 2432–2437.

[40] E. Einhorn and H.-M. Gross, "Generic 2D/3D SLAM with NDT maps for lifelong application," in *Proc. Eur. Conf. Mobile Robots*, Sep. 2013, pp. 240–247.

[41] S. J. Ahn, W. Rauh, and M. Recknagel, "Ellipse fitting and parameter assessment of circular object targets for robot vision," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 1, Oct. 1999, pp. 525–530.

[42] Z. Zhang, "Parameter estimation techniques: A tutorial with application to conic fitting," *Image Vis. Comput.*, vol. 15, no. 1, pp. 59–76, 1997.

[43] M. Wise, M. Ferguson, D. King, E. Diehr, and D. Dymesich, "Fetch and freight: Standard platforms for service robot applications," in *Proc. Workshop Autonomous Mobile Service Robots*, Jul. 2016, pp. 1–6.

[44] P. Bergström and O. Edlund, "Robust registration of point sets using iteratively reweighted least squares," *Comput. Optim. Appl.*, vol. 58, no. 3, pp. 543–561, Jul. 2014.

[45] S. Huang and G. Dissanayake, "Convergence and consistency analysis for extended Kalman filter based SLAM," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 1036–1049, Oct. 2007.

[46] J. Guivant, F. R. Masson, and E. M. Nebot, "Simultaneous localization and map building using natural features and absolute information," *Robot. Autonomous Syst.*, vol. 6, no. 1, pp. 581–586, Aug. 2000.

[47] S. Huang, H. Wang, U. Frese, and G. Dissanayake, "On the number of local minima to the point feature based SLAM problem," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 2074–2079.

JIAHENG ZHAO received the bachelor's degree in mechanical engineering and automation from Beijing Jiaotong University, Beijing, China, in 2014. He is currently pursuing the Ph.D. degree in mechanical engineering with the Beijing Institute of Technology. He is currently pursuing the Dual Ph.D. degree with the Centre for Autonomous Systems, University of Technology Sydney, Australia. His research interests include computer vision, localization, and simultaneous localization and mapping for autonomous robots.

SHOUDONG HUANG received the bachelor's and master's degrees in mathematics and the Ph.D. degree in automatic control from Northeastern University, China, in 1987, 1990, and 1998, respectively. He is currently an Associate Professor with the Centre for Autonomous Systems, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, Australia. His research interests include nonlinear control systems, and mobile robots simultaneous localization and mapping (SLAM) and exploration and navigation.

LIANG ZHAO received the Ph.D. degree in photogrammetry and remote sensing from Peking University, China, in 2013. He was a Postdoctoral Research Associate with the Hamlyn Centre for Robotic Surgery, Department of Computing, Faculty of Engineering, Imperial College London, U.K. He is currently a Lecturer with the Centre for Autonomous Systems, Faculty of Engineering and Information Technology, University of Technology Sydney, Australia. His research interests include mobile/surgical robots simultaneous localization and mapping (SLAM), optimization techniques, and image guide robotic surgery.

YONGBO CHEN received the Bachelor and Ph.D. degrees from the Beijing Institute of Technology, in 2012 and 2019, respectively. He is currently pursuing the Ph.D. degree with the Centre for Autonomous Systems, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, Australia. His research interests include UAV motion/mission planning, and mobile robots simultaneous localization and mapping.

XIAO LUO received the bachelor's degree in information and electronics from the Beijing Institute of Technology, Beijing, China, in 2006, and the Master and Ph.D. degrees in mechatronical engineering from the Beijing Institute of Technology, Beijing, China, in 2008 and 2012, respectively. She is currently a Lecturer with the Beijing Institute of Technology. Her current research interests include image processing, computer vision, simultaneous localization and mapping (SLAM), and automate robot.