2D Laser SLAM with General Features Represented by Implicit Functions

Jiaheng Zhao[1], Liang Zhao[1], Shoudong Huang[1] and Yue Wang[2]

***Abstract*— The main contribution of this paper is the problem formulation and algorithm framework for 2D laser SLAM with general features represented by implicit functions. Since 2D laser data reflect the distances from the robot to the boundary of objects in the environment, it is natural to use the boundary of the general objects/features within the 2D environment to describe the features. Implicit functions can be used to represent almost arbitrary shapes from simple (e.g. circle, ellipse, line) to complex (e.g. a cross-section of a bunny model), thus it is worth studying implicit-expressed feature in 2D laser SLAM.**

In this paper, we clearly formulate the SLAM problem with implicit functions as features, with rigorously computed observation covariance matrix to be used in the SLAM objective function and propose a solution framework. Furthermore, we use ellipses and lines as examples to compare the proposed SLAM method with the traditional pre-fit method (represent the feature using its parameters and pre-fit the laser scan to get the fitted parameter as virtual observations). Simulation and experimental results show that our proposed method has a better performance compared with the pre-fit method and other methods, demonstrating the potential of this new SLAM formulation and method.

***Keywords* —2D laser SLAM, general feature, implicit function, covariance, feature parametrization, performance analysis.**

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. In the last few years, one application that has been widely adopted by industry and academy is planar SLAM based on 2D lidar or laser rangefinder, and the number of approaches has increased [1], [2], [3], [4], [5].

Currently, the two main common approaches to 2D laser SLAM are scan matching based approach and feature based approach. In a typical scan matching based approach, nearby scans are registered to obtain the relative poses, and then a pose-graph optimization is performed to obtain the optimized poses. Finally, the map is built via the optimized poses and the laser scans. Although it is beneficial for scan matching not making assumption on environment, a prior knowledge of the geometrical information is helpful to improve the accuracy. In the industrial environment with multiple stacks, for instance, the boundary description of manufactured objects can be easily obtained from the manufacturer, which is workable to model the boundary via implicit functions for stacks. Furthermore, there is difficulty involved for scan matching method in accurately fusing information from consecutive scans.

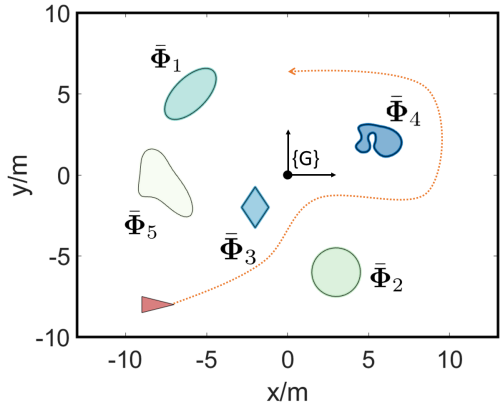


Fig. 1. Schematic diagram of SLAM with implicit functions. Red rectangle is the robot starting at $[-8, -8]^\intercal$. Features with implicit functions are defined by:
$\bar{\Phi}_1 \overset{\text{def.}}{=} 0.6\mathbf{x}^2 - 0.8\mathbf{x}\mathbf{y} + 0.6\mathbf{y}^2 + 11.2\mathbf{x} - 10.8\mathbf{y} + 59.6$;
$\bar{\Phi}_2 \overset{\text{def.}}{=} 0.4\mathbf{x}^2 - 2.7\mathbf{x} + 0.4\mathbf{y}^2 + 5.3\mathbf{y} + 19$;
$\bar{\Phi}_3 \overset{\text{def.}}{=} 1.2|\mathbf{x}+2| + 0.8|\mathbf{y}+2| - 1$;
$\bar{\Phi}_4 \overset{\text{def.}}{=} 3(1-5(\mathbf{x}-5))^2 e^{-(5(\mathbf{x}-5)^2)-(5(\mathbf{y}-2)+1)^2} - 10((\mathbf{x}-5) - 5(\mathbf{x}-5)^3 - 3(\mathbf{y}-2)^5)e^{-2(\mathbf{x}-5)^2-5(\mathbf{y}-2)^2} - \frac{1}{3}e^{-(5(\mathbf{x}-5)+1)^2-4(\mathbf{y}-2)^2} - 0.1$;
$\bar{\Phi}_5 \overset{\text{def.}}{=} 1.2\mathbf{x}^4 + 0.4\mathbf{y}^4 + 2\mathbf{x}\mathbf{y} + 2.3\mathbf{x}^3\mathbf{y} - 2$.
Noted that $\mathbf{x}$ and $\mathbf{y}$ are points belonging to each feature in global frame.

Feature based approach estimates the parameters of the feature present in the environment. One basic feature based SLAM is point feature SLAM [6] where the feature parameter is the position of the point feature. Other features used in laser SLAM include line feature [7], ellipse feature [8], curve feature [9] and so on. A unified formulation named *matchable* was employed in [10] to represent point, line and plain features. Zhang et al. [11] utilized remote and near feature parametrization to improve the robustness of rotation estimation. Holỳ [12] combined points and lines for the scan matching to decrease the computation time and increase the accuracy. Our previous work [8] utilized ellipse feature to reduce the number of points needed during calculation and to compensate for errors in observations from different perspectives. Rao et al. [13] extracted Bézier curves and used four control points to parameterize curve features, then the optimization problem was solved by Levenberg-Marquardt algorithm. Pedraza et al. [14] were the first to use spline

[1]The authors are with the Centre for Autonomous Systems, Faculty of Engineering and Information Technology, University of Technology Sydney (UTS), Sydney, Australia. jiaheng.zhao@student.uts.edu.au; {liang.zhao, shoudong.huang}@uts.edu.au

[2]Yue Wang is with the State Key Laboratory of Industrial Control and Technology, Zhejiang University, Hangzhou, P.R. China. ywang24@zju.edu.cn

to parameterize features and then optimized robot poses and control points simultaneously.

However, most of the existing feature based SLAM which fit features before optimization can only handle limited types of geometric features, which is possible to lose information. Zhang et al. [15] extracted circle features from lidar data and utilized circle centers to estimate navigation line. One limitation is that they assumed all the features are circle and possess similar radius. The navigation line is easily influenced by fitted centers since fitting sparse lidar points to circles introduced information loss.

In reality, laser scans reflect the boundary of an object, which could be of arbitrary shape and cannot be easily described with feature parameters. On the other hand, most of the boundaries can be expressed by implicit functions (every point on the boundary satisfies the function, referred to Section II-C). Thus we would like to ask the question: Is it possible to use implicit functions as features in SLAM?

It should be noted that implicit functions cover general geometric features as special cases. As is shown in Fig. 1, features like circle, ellipse, diamond, or even irregular closed curve can be expressed by implicit functions. Thus, SLAM with implicit functions as features is a very general feature based SLAM and has the potential to be applied in different scenarios (3D surfaces are implicit functions in 3D).

This paper studies the 2D laser SLAM problem utilizing implicit functions as features. To the best of our knowledge, no clear researches are made on formulating features as implicit functions. We clearly formulate the problem as an optimization problem, and (a) correctly compute the observation covariance matrix, (b) formulate a symmetry energy term for closed shapes. Then, we propose a potential framework which can be adopted for all the types of features represented by implicit functions. To illustrate the new proposed SLAM technique, we use ellipse and line features as two examples to demonstrate how the proposed problem can be solved by iterative methods. We compare the performance of our new technique with the traditional pre-fit method and clearly show the advantages of the proposed approach. Our main contributions are:

- We propose a framework for implicit function based SLAM problem by clearly formulating the problem with implicit functions to represent features, computing corresponding implicit covariance rigorously, and presenting a framework for solving the problem using iterative methods.
- By taking ellipse and line features as examples, we compared the proposed method with the tradition feature based method and proved the superiority of our method.
- We develop a novel logarithmic form objective function for features with closed shapes to enhance the convergence of the iteration based algorithms.

II. SLAM PROBLEM WITH GENERAL FEATURES DESCRIBED BY IMPLICIT FUNCTIONS

In this section, we formulate a general SLAM problem with features represented by implicit objective functions and elaborate the feasibility and approaches of solving such a problem.

A. Notation and conventions

In this paper, the semicolon is to represent vertical vector concatenation. An observed point is defined as $\mathbf{p} \in \mathbb{R}^2$, while $\hat{\mathbf{p}}$ is the same point in homogeneous coordinate. A 2D point set is denoted by $\mathbf{P} = [\mathbf{p}_1, \cdots, \mathbf{p}_w]^\intercal$, and the corresponding homogeneous point set is $\hat{\mathbf{P}}$. We also assume observed points have a zero-mean Gaussian noise $\mathbf{n}_z \in \mathbb{R}^2 \sim N(\mathbf{0}, \Sigma_z)$.

For an angle $\phi \in [-\pi, \pi)$, let $R(\phi) \in SO(2)$ be the corresponding rotation matrix which is abbreviated as R. The translation is denoted by $\mathbf{t}$. ${}^{\{G\}}R, {}^{\{G\}}\mathbf{t}$ means the rotation or translation defined in global frame. To simplify the formula, $\{G\}$ is usually omitted. Suppose a robot pose $\mathbf{\Xi}_j = [\mathbf{t}_j; \phi_j]$, we use $T(\mathbf{\Xi}_j, {}^{\{j\}}\chi)$ to represent the process of transforming a ${}^{\{j\}}\chi$ from frame $\{j\}$ to global frame $\{G\}$, and $T^{-1}(\mathbf{\Xi}_j, {}^{\{G\}}\chi)$ indicates the opposite process. χ can be a point, a point cloud, or a feature.

In order to evaluate approaches fairly, we assume that which feature the collected data belongs to is known in the following sections as well as in the simulation experiments. The same ground-truth data association is also adopted to all the other compared methods in simulation to ensure a fair comparison. The data association of real application is preprocessed and is discussed in Section IV-E.

B. General feature based SLAM problem

Consider a general feature-based SLAM problem (see Fig. 1). Assume a robot moves n steps in the scenario containing features $\mathbf{\Phi}_1, \mathbf{\Phi}_2, \cdots, \mathbf{\Phi}_q$. At each step, the robot collects laser points hitting on features. The state is defined by:

$$\mathbf{\Psi} \overset{\text{def.}}{=} \begin{Bmatrix} \mathbf{\Xi}_1 & \mathbf{\Xi}_2 & \cdots & \mathbf{\Xi}_n \\ \mathbf{\Phi}_1 & \cdots & \mathbf{\Phi}_q & \end{Bmatrix} \overset{\text{def.}}{=} < \mathbf{\Xi}, \mathbf{\Phi} > \tag{1}$$

under the assumption that initial pose $\mathbf{\Xi}_0 = [0; 0; 0]$. Thus, the problem is minimizing the energy function:

$$\underset{\mathbf{\Psi}}{\operatorname{argmin}} \; E_{\text{total}} = E_{\text{odom}} + \underbrace{\sum_{j=1}^{q} E_{\text{feature},j}}_{E_{\text{feature}}} \tag{2}$$

and each term in Eq. (2) is defined by:

$$\begin{aligned} E_{\text{odom}} &= \frac{1}{2} \sum_{i=1}^{n} \| f(\mathbf{Z}_{\text{odom},i}, \mathbf{\Xi}_{i-1}, \mathbf{\Xi}_i) \|^2_{\Sigma_{\text{odom},i}} \\ E_{\text{feature},j} &\overset{\text{def.}}{=} \frac{1}{2} \sum_{i=1}^{n} \| g(\mathbf{Z}_{\text{feature},i,j}, \mathbf{\Xi}_i, \mathbf{\Phi}_j) \|^2_{\Sigma_{\text{feature},i,j}} \end{aligned} \tag{3}$$

where $\mathbf{Z}_{\text{odom},i}$ is the observation vector of i^{th} odometry, $\mathbf{Z}_{\text{feature},i,j}$ is the observation vector of feature j at pose i.[1] $f(\mathbf{Z}_{\text{odom},i}, \mathbf{\Xi}_{i-1}, \mathbf{\Xi}_i)$ and $g(\mathbf{Z}_{\text{feature},i,j}, \mathbf{\Xi}_i, \mathbf{\Phi}_j)$ are the cost functions for the two entries, respectively. $\Sigma_{\text{odom},i}$ is

[1]Without loss of generality, we assume feature j is observed from all the poses 1 to n. For different SLAM formulations, the format of $\mathbf{Z}_{\text{feature},i,j}$ is different, as seen in Sections III-A and III-B (implicit function), Sections III-D and III-E (pre-fit).

the odometry covariance at the i^{th} step. $\Sigma_{\text{feature},i,j}$ is the covariance of feature j's observation from pose i.

The energy term of a typical odometry observation (measuring relative pose) is:

$$f(\mathbf{Z}_{\text{odom},i}, \mathbf{\Xi}_{i-1}, \mathbf{\Xi}_i) = \mathbf{Z}_{\text{odom},i} - \begin{bmatrix} R_{i-1}^{\intercal}(\mathbf{t}_i - \mathbf{t}_{i-1}) \\ \text{dist}(\phi_i - \phi_{i-1}) \end{bmatrix}_{3\times 1} \tag{4}$$

where $\text{dist}(\phi_i - \phi_{i-1})$ is the angle distance between the i^{th} and the $(i-1)^{th}$ robot orientation, and $\mathbf{Z}_{\text{odom},i}$ is the odometry observation at pose i in the form of $[\Delta x_i, \Delta y_i, \Delta\phi_i]^{\intercal}$. In this paper, geodesic distance is used to find the difference of angles, which is also known as "*wrap*".

C. SLAM with features represented by implicit functions

Fig. 2 illustrates the flow chart of SLAM with feature represented by implicit functions. Suppose an implicit function

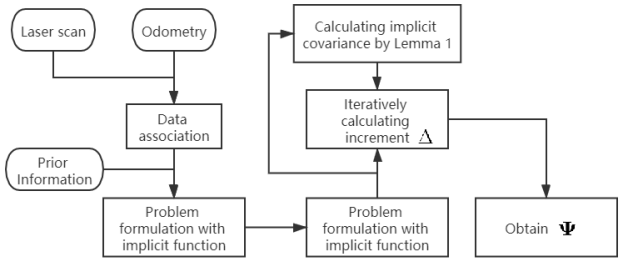


Fig. 2. The flow chart of SLAM with implicit functions.

$\bar{\mathbf{\Phi}}_j(\mathbf{P}) = \mathbf{0}$ holds[2] for a point set $\mathbf{P}$ that belongs to feature $\mathbf{\Phi}_j$ and $\mathbf{P}$ is in the global coordinate as well as $\mathbf{\Phi}_j$. As shown in Fig. 1, for example, feature $\mathbf{\Phi}_1$ to $\mathbf{\Phi}_5$ are in complex shapes, and the corresponding implicit functions $\bar{\mathbf{\Phi}}_1 = \mathbf{0}$ to $\bar{\mathbf{\Phi}}_5 = \mathbf{0}$ hold for every point locating on each feature respectively.

If the perfect observation of $\mathbf{\Phi}_j$ at pose i is a point set denoted by ${}^{\{i\}}\mathbf{P}$, it must satisfy the implicit function after transforming it to global frame, that is:

$$\bar{\mathbf{\Phi}}_j({}^{\{G\}}\mathbf{P}) = \bar{\mathbf{\Phi}}_j(T^{-1}(\mathbf{\Xi}_i, {}^{\{i\}}\mathbf{P})) = \mathbf{0}. \tag{5}$$

Eq. (5) is obviously an implicit function of $\mathbf{\Xi}_i$,$\mathbf{\Phi}_j$ and ${}^{\{i\}}\mathbf{P}$. Since observations always contain noises, still taking feature j as an instance, the energy term of feature j turns to be:

$$E_{\text{feature},j} = \frac{1}{2}\sum_{i=1}^{n} \left\| \bar{\mathbf{\Phi}}_j \left(T^{-1}(\mathbf{\Xi}_i, \mathbf{Z}_{\text{feature},i,j}) \right) \right\|^2_{\Sigma_{\mathbf{\Phi}_j,i}} \tag{6}$$

where $\mathbf{Z}_{\text{feature},i,j}$ are raw points belonging to feature j at pose i, and $\Sigma_{\mathbf{\Phi}_j,i}$ is the corresponding covariance, which will be computed in the next section.

[2]We use $\mathbf{\Phi}_j$ to represent the feature j and use $\bar{\mathbf{\Phi}}_j$ to represent the feature's implicit function.

D. Implicit covariance

In typical least squares problems, the energy term is $E = \|\mathbf{z} - f(\mathbf{x})\|^2_{\Sigma}$ and Σ is the covariance of the noise in $\mathbf{z}$. However, the covariance of implicit functions cannot be obtained directly from observations. Thus the following lemma is proposed to link the raw observation and the implicit items and calculate the covariance $\Sigma_{\mathbf{\Phi}_j,i}$ in Eq. (6).

Lemma 1: Consider a least squares problem with energy term $E = \|f(\mathbf{x}, \mathbf{z})\|^2_{\Sigma_f}$ with variables $\mathbf{x}$ and observations $\mathbf{z}$. Assume $\mathbf{z} = \mathbf{z}_0 + \mathbf{n}_{\mathbf{z}}$, where $\mathbf{n}_{\mathbf{z}} \sim N(\mathbf{0}, P_z)$ is a zero-mean Gaussian noise. Since $f(\mathbf{x}_0, \mathbf{z})$ approximately follows Gaussian distribution around $\mathbf{z} = \mathbf{z}_0$ as:

$$f(\mathbf{x}_0, \mathbf{z}) \sim N(f(\mathbf{x}_0, \mathbf{z}_0), (J_{\mathbf{z}} P_z^{-1} J_{\mathbf{z}}^{\intercal})^{-1}) \tag{7}$$

where

$$J_{\mathbf{z}} = \left.\frac{\partial f}{\partial \mathbf{z}}\right|_{\mathbf{z}=\mathbf{z}_0} \tag{8}$$

a good choice of Σ_f in the least squares problem is

$$\Sigma_f^{-1} = J_{\mathbf{z}} P_z^{-1} J_{\mathbf{z}}^{\intercal} \tag{9}$$

Proof: See Appendix A. ■

Remark 1: Since it is impossible to obtain $\mathbf{z}_0$ during practical experiments, which are the ground truth of $\mathbf{z}$. Because the noise influence of observed points is similar to that of ground truth points when the observations are near the exact positions, we use the observed points to approximately calculate the covariance in Eq. (6). The experiment in Section IV-B shows the validity.

E. Approaches to solve the SLAM problem

One important difference between the new SLAM problem and a traditional feature based SLAM is: Feature $\mathbf{\Phi}_j$ in Eq. (1) is expressed by an implicit function instead of a finite-dimensional vector. Thus $\mathbf{\Psi}$ in Eq. (1) is not a typical state vector and the problem cannot be directly solved using iterative methods.

However, if we can identify some "changeable parameters" in each feature $\mathbf{\Phi}_j$, then the problem is to find these changeable parameters together with the poses such that the total energy is minimized.

Suppose the "changeable parameters" in feature $\mathbf{\Phi}_j$ are defined by s elements in a vector form $\overrightarrow{\mathbf{\Phi}_j} = [\Phi_{j_1}, \Phi_{j_2}, \cdots, \Phi_{j_s}]^{\intercal}$, then standard iterative methods such as Gauss-Newton and Levenberg-Marquardt can be used to solve the problem.

Suppose the incremental Δ is defined by:

$$\begin{aligned} \Delta &\overset{\text{def.}}{=} \begin{Bmatrix} \Delta\mathbf{\Xi}_1 & \Delta\mathbf{\Xi}_2 & \cdots & \Delta\mathbf{\Xi}_n \\ \Delta\overrightarrow{\mathbf{\Phi}_1} & \cdots & \Delta\overrightarrow{\mathbf{\Phi}_q} & \end{Bmatrix} \\ &\overset{\text{def.}}{=} < \Delta\mathbf{\Xi}, \Delta\overrightarrow{\mathbf{\Phi}} > . \end{aligned} \tag{10}$$

Since we need to find Δ, an $\oplus$ operator is defined to apply the increment Δ to $\mathbf{\Psi}_{\text{old}}$ as:

$$\begin{aligned} \mathbf{\Psi}_{\text{new}} &\overset{\text{def.}}{=} \mathbf{\Psi}_{\text{old}} \oplus \Delta \\ &\overset{\text{def.}}{=} \begin{Bmatrix} \mathbf{\Xi}_1 \oplus \Delta\mathbf{\Xi}_1 & \cdots & \mathbf{\Xi}_n \oplus \Delta\mathbf{\Xi}_n \\ \mathbf{\Phi}_1 \oplus \overrightarrow{\mathbf{\Phi}_1} & \cdots & \mathbf{\Phi}_q \oplus \overrightarrow{\mathbf{\Phi}_q} \end{Bmatrix} \\ &\overset{\text{def.}}{=} < \mathbf{\Xi}_{\text{old}} \oplus \Delta\mathbf{\Xi}, \mathbf{\Phi}_{\text{old}} \oplus \Delta\overrightarrow{\mathbf{\Phi}} > . \end{aligned} \tag{11}$$

The step increment Δ can be calculated by LM method. The Jacobian of feature j alone is given as an example:

$$J_j = \begin{bmatrix} \frac{\partial \bar{\mathbf{\Phi}}_j}{\partial \mathbf{\Xi}} & \frac{\partial \bar{\mathbf{\Phi}}_j}{\partial \Phi_{j_1}} & \cdots & \frac{\partial \bar{\mathbf{\Phi}}_j}{\partial \Phi_{j_s}} \end{bmatrix} \tag{12}$$

the first element in J_j is the derivative with respect to all the poses.

It is worth noting that the covariance adaptively varies according to Lemma 1, and the implicit function allows very flexible representation of the features in the environments. The "changeable parameters" is a way to parameterize the feature and adjust an initial value for the general feature when the more detailed shape information of the feature becomes available. Currently, the number of parameters is determined manually to help optimization.

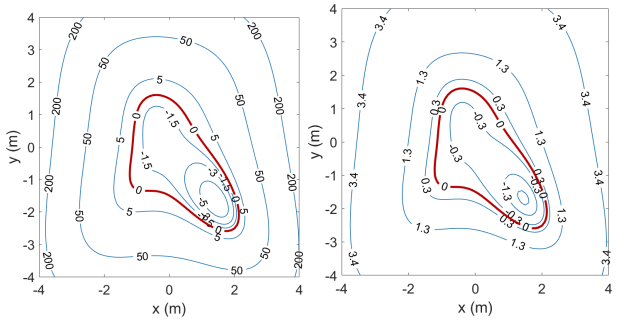
F. An improved objective function for closed shape

One inevitable problem of implicit functions is: for features with closed shapes, the value of implicit functions $\bar{\mathbf{\Phi}}_j$ in the error term $E_{\text{feature},j}$ various from $-\varrho$ (inside the boundary, where $\varrho > 0$, varying from different implicit functions) to 0 (at the boundary) and then from 0 to $+\infty$ (outside the boundary). ϱ is generally a small number. The value of $\bar{\mathbf{\Phi}}_j$ changes slightly within the boundary, while the change outside the boundary is dramatic. This will make it difficult for the energy term to quickly decline to the optimal solution when the virtual observation points are inside the boundary during iterations.

We propose to improve $\bar{\mathbf{\Phi}}_j$ to $\bar{\mathbf{\Phi}}_j^\star$, following:

$$\bar{\mathbf{\Phi}}_j^\star = \log(\frac{1}{\varrho}\bar{\mathbf{\Phi}}_j + 1) \tag{13}$$

then the value of $\bar{\mathbf{\Phi}}_j^\star$ varies from $-\infty$ (inside the boundary) to $+\infty$ (outside the boundary).



(a) Value of $\bar{\mathbf{\Phi}}_5$ (b) Value of $\bar{\mathbf{\Phi}}_5^\star$

Fig. 3. Comparison of implicit functions for $\mathbf{\Phi}_5$. Red line is the boundary of feature $\mathbf{\Phi}_5$. Blue contours are the values of $\bar{\mathbf{\Phi}}_5$ and $\bar{\mathbf{\Phi}}_5^\star$, respectively.

Fig. 3a shows a general implicit function defined by the following function, taking $\mathbf{\Phi}_5$ (in Fig. 1) as an example:

$$\bar{\mathbf{\Phi}}_5(\mathbf{P}) = 1.2\mathbf{x}^4 + 0.4\mathbf{y}^4 + 2\mathbf{xy} + 2.3\mathbf{x}^3\mathbf{y} - 2 \tag{14}$$

where a point set is $\mathbf{P} = [\mathbf{x}, \mathbf{y}] \in \mathbb{R}^{w\times 2}$ belonging to $\mathbf{\Phi}_5$. Obviously, the value of $\bar{\mathbf{\Phi}}_5$ outside the boundary grows rapidly (from 0 to 200), while that inside the boundary does not change much (from 0 to -5). Such value distribution can cause an inefficient descending problem.

Hence, a logarithmic function is applied to reconstruct the original implicit function as:

$$\bar{\mathbf{\Phi}}_5^\star(\mathbf{P}) = \log(\frac{1}{\varrho}\bar{\mathbf{\Phi}}_5(\mathbf{P}) + 1) \tag{15}$$

where $\varrho = 0.1543$ for $\mathbf{\Phi}_5$.

Fig. 3b depicts the improved implicit function. The value of $\bar{\mathbf{\Phi}}_5^\star$ inside and outside the boundary changes relatively evenly, showing a symmetry property and leading to a steady convergence.

III. ILLUSTRATION OF THE PROPOSED METHOD WITH TWO EXAMPLES

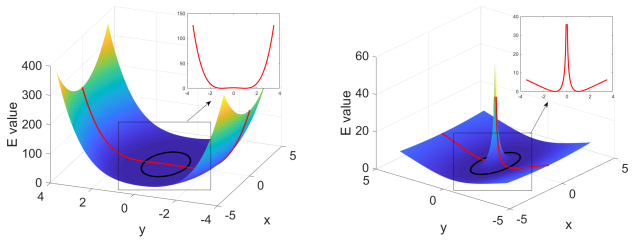
In this section, we use two examples (ellipse feature and line feature) to compare our method (called "post-count") with the traditional pre-fit method.

A. Post-count method: Ellipse feature

Suppose for an arbitrary point $\mathbf{p} = (x, y)^\intercal$ defined in the global frame. Then $\bar{\mathbf{\Phi}}_j$ is:

$$\begin{aligned} \bar{\mathbf{\Phi}}_j(\mathbf{p}) = &\frac{\left((x - F_x)\cos F_\phi + (y - F_y)\sin F_\phi\right)^2}{F_{r_1}^2} \\ &+ \frac{\left(-(x - F_x)\sin F_\phi + (y - F_y)\cos F_\phi\right)^2}{F_{r_2}^2} - 1 \end{aligned} \tag{16}$$

where (F_x, F_y) is the center of the ellipse, F_ϕ is the angle between the major axis of the ellipse and x axis, F_{r_1} and F_{r_2} are the major and minor axis respectively, and the vector form of changeable parameters for ellipse feature is $\overrightarrow{\mathbf{\Phi}}_j = [F_x, F_y, F_\phi, F_{r_1} F_{r_2}]^\intercal$. The reason of formulating $\overrightarrow{\mathbf{\Phi}}_j$ as the given way is to facilitate comparison with the pre-fit method, which requires reasonable feature parametrization (discussed in Section III-D and Section III-E).



(a) Value of $E_{\text{feature},j}$. (b) Value of $E_{\text{feature},j}^\star$.

Fig. 4. Comparison of energy terms utilizing Eq. (16) and Eq. (17) for ellipse feature.

We reduce the dimensions of Eq. (16) to 2 by fixing the last three parameters of $\overrightarrow{\mathbf{\Phi}}_j$ and robot poses in order to illustrate the convergence ability. Fig. 4a depicts the energy term $E_{\text{feature},j} = \left\|\bar{\mathbf{\Phi}}_j(\mathbf{p})\right\|_\Sigma^2$, and the black ellipse is the ground truth ellipse $[0, 0, 0, 2, 1]^\intercal$ depicted in xy-plane. It seems that the function can converge well for any point from a macro perspective. However, the fact is that $E_{\text{feature},j}$ is

hard to converge when the observed points locate within the ellipse area. As shown in the sub-figure in Fig. 4a, we drew a line where $x = 0$ as an instance. The value of $E_{\text{feature},j}$ varies slightly if $|y| \leq 1$, which means the gradient is too small. It is worth mentioning that this figure is only a 2-dimension example. The real function is in high-dimension, and it is far more difficult for the energy function to decline in the right direction when the observed points are inside the ellipse area.

As the result, we reconstruct the implicit function by:

$$\bar{\mathbf{\Phi}}_j^{\star}(\mathbf{p}_a) = \log(\bar{\mathbf{\Phi}}_j(\mathbf{p}_a) + 1). \tag{17}$$

Depict the new energy term in a similar way $E^{\star}_{\text{feature},j} = \left\|\bar{\mathbf{\Phi}}_j^{\star}(\mathbf{p})\right\|^2_{\Sigma}$ and the result is shown in Fig. 4b. In this example, the new objective function provides a sharp decline when points fall inside the ellipse. It is still easy to descend due to a large change in energy function.

The details of $g(Z_{\text{ep},i,j}, \mathbf{\Xi}_i, \bar{\mathbf{\Phi}}_j)$, $E^{\star}_{\text{feature},j}$ and $\Sigma_{\mathbf{\Phi}_j,i}$ can be found in Appendix C.

B. Post-count method: Line feature

Similarly, assume ${}^{\{i\}}\mathbf{P}_{\mathbf{\Phi}_k}$ is the homogeneous point set of k^{th} line feature in the pose i. The observations are defined by $Z_{\text{lp},i,k} = {}^{\{i\}}\mathbf{P}_{\mathbf{\Phi}_k}$. $\overrightarrow{\mathbf{\Phi}_k} = [l_\alpha, p]^\intercal$ is the vector form of changeable parameters normalized by $p \geq 0$. The implicit function of $\mathbf{\Phi}_k$ is:

$$\begin{aligned} g(Z_{\text{lp},i,k}, \mathbf{\Xi}_i, \mathbf{\Phi}_k) &= \bar{\mathbf{\Phi}}_k(T(\mathbf{\Xi}_i, Z_{\text{lp},i,k})) \\ &= \cos l_\alpha \mathbf{x} + \sin l_\alpha \mathbf{y} - p \end{aligned} \tag{18}$$

$\mathbf{x}$ and $\mathbf{y}$ are column vectors of $Z_{\text{lp},i,k}$. Since p is a scalar, the "-" operator represents p is subtracted from each element in the preceding term.

The details of $g(\mathbf{Z}_{\text{feature},i,k}, \mathbf{\Xi}_i, \mathbf{\Phi}_j)$, $E_{\text{feature},k}$ and $\Sigma_{\mathbf{\Phi}_k,i}$ are listed in Appendix C.

C. Traditional feature-based SLAM problem: Pre-fit

In traditional feature-based SLAM using pre-fit, the observation $\tilde{\mathbf{Z}}_{\text{feature},i,j}$ is obtained by fitting the raw data following a certain parametrization, and the covariance $\tilde{\Sigma}_{\text{feature},i,j}$ is from the fitting result. Hence, the optimization problem aims to minimize the energy cost $\tilde{E}_{\text{feature},j}$ between actual observation $\tilde{\mathbf{Z}}_{\text{feature},i,j}$ and theoretical observation $T^{-1}(\mathbf{\Xi}, \mathbf{\Phi})$. To obtain the theoretical observations, we need to transform feature states from global frame to the corresponding local frame. In order to distinguish annotations from post-count method, we use $\mathbf{F}$ to denote features. Taking feature j as an example, the energy cost can be expressed as:

$$\tilde{E}_{\text{feature},j} = \frac{1}{2}\sum_{i=1}^{n} \|\tilde{\mathbf{Z}}_{\text{feature},i,j} - T^{-1}(\mathbf{\Xi}_i, \mathbf{F}_j)\|^2_{\tilde{\Sigma}_{\text{feature},i,j}}. \tag{19}$$

It is worth noting that $\mathbf{F}_j$ depends on how the feature is parameterized, instead of only in the point form. And $\tilde{\mathbf{Z}}_{\text{feature},i,j}$ is always in the same format of the feature state. It is easy to find $\tilde{\Sigma}_{\text{feature},i,j}$ if features can be observed directly or fitted by raw data in advance. The comparison of pre-fit and post-count method is shown in Tab. I.

D. Pre-fit method: Ellipse feature

The raw data can be used to fit an ellipse, one method is presented in [8]. Denote $\breve{\mathbf{Z}}_{\text{ef},i,j}$ as the observation of the j^{th} ellipse feature at pose i, then $\breve{\mathbf{Z}}_{\text{ef},i,j} = {}^{\{i\}}\breve{\mathbf{F}}_j$, where ${}^{\{i\}}\breve{\mathbf{F}}_j$ is the actual observation of ${}^{\{i\}}\mathbf{F}_j$.

Hence the pre-fit ellipse observation function can be written as:

$$g(\breve{\mathbf{Z}}_{\text{ef},i,j}, \mathbf{\Xi}_i, \mathbf{F}_j) = {}^{\{i\}}\breve{\mathbf{F}}_j - \begin{bmatrix} T^{-1}(\mathbf{\Xi}_i, \mathbf{F}_{j_{xy}}) \\ \text{dist}(F_{j_\phi} - \phi_i) \\ \mathbf{F}_{j_{r1,r2}} \end{bmatrix}_{5\times 1} \tag{20}$$

where $\mathbf{F}_{j_{xy}}$,F_{j_ϕ} and $\mathbf{F}_{j_{r1,r2}}$ are the position, angle and axis dimensions of the j^{th} feature in global frame, respectively. An extra *wrap* step is still needed for the 3^{rd} element.

The ellipse feature uncertainty $\Sigma_{\text{ef},i,j}$ is easily computed by $\Sigma^{-1}_{\text{ef},i,j} = \mathbf{J}^\intercal \Sigma_z^{-1} \mathbf{J}$ (Discussed in our previous work [8, Eq. 16]).

E. Pre-fit method: Line feature

We replace $\mathbf{F}_k$ with $\mathbf{L}_k$ to represent the line feature to distinguish it from ellipse features. Suppose the k^{th} line state vector is parameterized by $\mathbf{l}_k = [\alpha_k, p_k]^\intercal (p_k \geq 0)$. Then the corresponding line feature $\mathbf{L}_k$ normalized by $\mathbf{l}_k$ is $\mathbf{L}_k = [\cos\alpha_k, \sin\alpha_k, -p_k]^\intercal$. It is worth noting that the line feature state includes normalization process. Suppose the k^{th} line state vector is parameterized by $\mathbf{l}_k = [\alpha_k, p_k]^\intercal (p_k \geq 0)$. Then the corresponding line feature $\mathbf{L}_k$ normalized by $\mathbf{l}_k$ is $\mathbf{L}_k = [\cos\alpha_k, \sin\alpha_k, -p_k]^\intercal$ which satisfies $\mathbf{L}_k^\intercal \hat{\mathbf{p}} = 0$ where $\hat{\mathbf{p}}$ is a point on the line feature in homogeneous coordinate. This mapping is denoted by $\mathbf{l}_k \mapsto \mathbf{L}_k$ and is reversible.

The observation of line features is obtained by intuitively minimizing the distance from discrete points to the line. Suppose a point ${}^{\{i\}}\mathbf{p}_w$ is defined in the local frame $\{i\}$ at pose i belonging to $\mathbf{L}_k$ and denote $\breve{\mathbf{Z}}_{\text{lf},i,k}$ as the observation of the k^{th} line feature at pose i. In order to avoid excessive mapping, the observations of line $\breve{\mathbf{Z}}_{\text{lf},i,k}$ are in the form of $\mathbf{L}_k$ and calculated by minimizing:

$$\underset{{}^{\{i\}}\mathbf{L}_k}{\text{argmin}} \sum_w {}^{\{i\}}\mathbf{L}_k^\intercal {}^{\{i\}}\hat{\mathbf{p}}_w \tag{21}$$

thus $\breve{\mathbf{Z}}_{\text{lf},i,k} = {}^{\{i\}}\mathbf{L}_k$.

The detailed derivation of pre-fit line model can be derived according to Lemma 2 at Appendix B. The model can be written as:

$$g(\breve{\mathbf{Z}}_{\text{lf},i,k}, \mathbf{\Xi}_i, \mathbf{l}_k) = {}^{\{l\}}\mathbf{L}_k - T^{-1}(\mathbf{\Xi}_i, \mathbf{L}_k) \tag{22}$$

Remarked here that an implicit mapping $\mathbf{l}_k \leftarrow \mathbf{L}_k$ is done to accomplish the state vector.

The uncertainty of line energy function $\Sigma_{\text{lf},i,k}$ can be developed by $\Sigma_{\text{lf},i,k} = \text{diag}(\Sigma_z, 0)$ according to Zhao et al. [17, Eq. (19)].

IV. EXPERIMENTS AND ANALYSIS

In this section several numerical examples were considered to analyze the performance of the proposed method: firstly, we investigated the validity of Lemma 1; secondly, we compared the results of our method and pre-fit method; then

TABLE I

COMPARISON OF TRADITIONAL FEATURE BASED METHOD (PRE-FIT) AND IMPLICIT FUNCTION FEATURE BASED METHOD (POST-COUNT)

		Traditional feature based SLAM: Pre-fit	**Implicit function feature based SLAM: Post-count**
Observation		Depending on feature parametrization $\tilde{\mathbf{Z}}_{\text{feature},i,j}$	Raw points $\mathbf{Z}_{\text{feature},i,j}$
Properties†		Need parametrization $\mathbf{F}_j$	Need Implicit function $\bar{\mathbf{\Phi}}_j(\mathbf{p})$
Objective function	Equation‡	$\frac{1}{2}\sum_{i=1}^{n}\|\tilde{\mathbf{Z}}_{\text{feature},i,j} - T^{-1}(\mathbf{\Xi}_i, \mathbf{F}_j)\|^2_{\Sigma_{\text{feature},i,j}}$	$\frac{1}{2}\sum_{i=1}^{n}\left\|\bar{\mathbf{\Phi}}_j\left(T^{-1}(\mathbf{\Xi}_i, \mathbf{Z}_{\text{feature},i,j})\right)\right\|^2_{\Sigma_{\mathbf{\Phi}_j,i}}$
	Frame change	From global frame $\{G\}$ to local frame $\{L\}$	From local frame $\{L\}$ to global frame $\{G\}$
Covariance	Notation	$\Sigma_{\text{feature},i,j}$	$\Sigma_{\mathbf{\Phi}_j,i}$
	Dependence	Depend on feature's fitting; fixed	Depend on feature's implicit function; vary in each iteration step
	Information loss	Accumulate with time and poses	No accumulation

† Pre-fit: Different feature-based SLAM methods differ in parametrization. Post-count: Each kind of features possesses a unique implicit function which holds for all the points belonging to the feature.
‡ Odometry part is omitted in the table.

we tested our method by fixing the covariance to evaluate Lemma 1; we also compared the performance between the improved implicit functions for ellipse feature with the original functions; and finally, we tested the robustness to observation noise level and checked the influence of fusing different types of features on both methods.

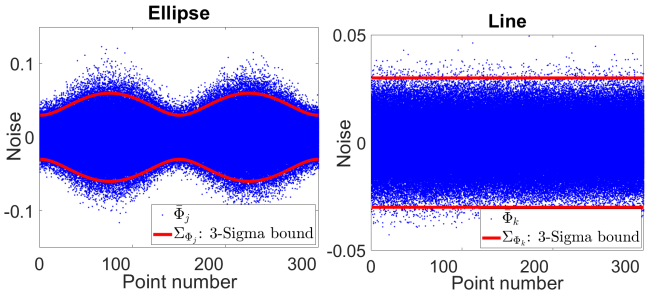
A. Simulation environment

The simulated environment is a 15 m×8 m space containing walls and ellipse features. The robot starts at $[0, 0, 0]^\intercal$ and odometry information is provided via a virtual wheel encoder with a random Gaussian noise $\text{diag}(0.4^2, 0.4^2, 3e^{-6})$. The initial observation noise is a random Gaussian noise $\mathbf{n}_z \sim N(\mathbf{0}, \text{diag}(0.05^2, 0.05^2))$. A 2D lidar is simulated with the valid range of 10m and the angle resolution of $0.33°$. Range-Azimuth model is adopted for simulation, but the range-bearing data is transferred to Cartesian coordinate to form the observation. Only points within valid range and hit on features can be observed.

Our algorithm was tested in multiple settings: pre-fit method with ellipse feature only (denoted as pfE), with line feature only (pfL), and with both ellipse and line feature (pfEL); post-count method with ellipse feature only (denoted as pcE), with line feature only (pcL), and with both ellipse and line feature (pcEL). All the three post-count methods implemented variable covariance for ellipse and line features according to Lemma 1. As a comparison, post-count method with a given unchanged covariance for both ellipse and line features is prepared (pcEL_fixCov). Another two comparisons are post-count method with original ellipse objective function ($E_{\text{feature},j}$ by Eq. (16)) (pcEL_oldFun) and the same configuration except fixing covariance matrix (pcEL_oldFun_fixCov).

B. Feasibility of implicit covariance

In this part, we used ellipse and line feature for verification of Lemma 1. We firstly verified whether $g(\mathbf{Z}_{\text{feature},i,j}, \mathbf{\Xi}_i, \mathbf{\Phi}_j)$ yields to $\Sigma_{\mathbf{\Phi}_j,i}$ via feature points by Monte Carlo experiment.



(a) Noise of ellipse feature (b) Noise of Line feature

Fig. 5. Uncertainty comparison. Red line is the 3-Sigma bound calculated by Lemma 1. Blue points are real values of implicit functions obtained by repeated experiments.

300 points on the edge of an ellipse and a line are selected respectively. Under the given noise, we repeatedly calculated both ellipse and line's implicit functions $\bar{\mathbf{\Phi}}_j$ and $\bar{\mathbf{\Phi}}_k$ with noisy observation and noisy $\mathbf{\Psi}$ by 1000 times and marked all results at each point as blue dots, as is illustrated in Fig. 5. The red line represents 3-sigma bound that obtained by Lemma 1. For both ellipse and line features, over 90% sampled data are strongly limited in 3-sigma bound, which verifies Lemma 1 statistically.

C. Results comparison

We compared the results of pcEL and pfEL, as shown in Fig. 6. Because the line parameters in this paper cannot represent line segments, the end points of each line at the first observation are maintained dependently (Do not participate in the optimization. We assume each observed line feature contains all the points.) and the resulted line features are drawn by transforming the end points to global frame. They are not accurate lines but to make the results look better.

Obviously, the trajectory of pcEL is much better than that of pfEL. Some sharp "jump" occurred for pfEL due to the badly-fitted observations. The Root Mean Square Error (RMSE) of pose is shown in the Tab. II. A main conclusion

TABLE II

RMSE COMPARISON OF MULTIPLE SETTINGS. THE DEFINITION OF ABBREVIATIONS IS IN SECTION IV-A.

	pfE	pfL	pfEL	pcE	pcL	pcEL	pcEL_fixCov	pcEL_oldFun	pcEL_oldFun_fixCov
x/m	0.0974	0.1038	0.0806	0.0940	0.0776	0.0749	0.0780	**0.0614**	0.0940
y/m	0.1713	0.1332	0.1128	0.0762	0.1695	**0.0520**	0.0693	0.0872	0.1608
$\mathbf{t}/m$	0.1971	0.1689	0.1386	0.1210	0.1864	**0.0912**	0.1043	0.1066	0.1863
θ/rad	0.0058	0.0068	0.0059	0.0067	0.0059	**0.0043**	0.0044	0.0078	0.0047

Remark: pf stands for pre-fit; pc stands for post-count; E stands for ellipse; L stands for line.

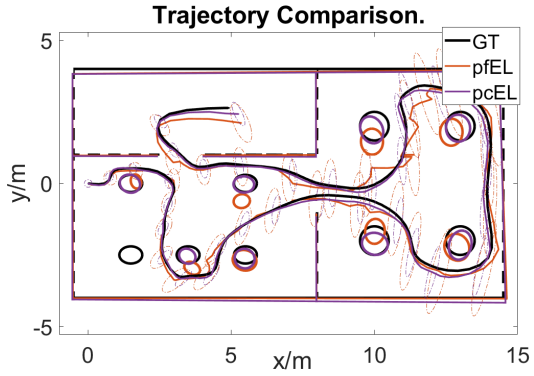


Fig. 6. Trajectory comparison. 3-sigma bound for robot's positions are depicted by shadowed ellipse in specific color.

is that the pre-fit model usually possesses a higher error level than post-count model. The position error of pfEL is 0.1386m, while that of pcEL is 0.0912m, which is much smaller than pfEL. Also, the final covariance of position of pcEL is smaller than that of pfEL according to Fig. 6.

Then we evaluated pcEL_fixCov in the same simulation environment. In Tab. II, it can be found that the RMSE of pcEL_fixCov is slightly bigger than pcEL, but smaller than any pre-fit approaches and post-count approaches.

D. Influence of feature fusion and robustness

A secondary conclusion can be derived from Tab. II is that the combination of ellipse and line features can effectively improve the accuracy of the results compared with settings only using one type of feature, whether it is pre-fit method or post-count method.

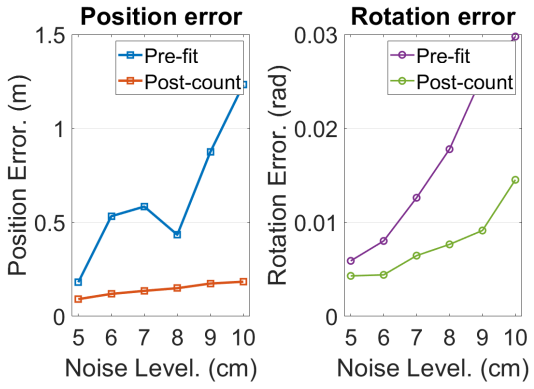


Fig. 7. Error changes with noise increasing.

In the last experiment, we tested both pre-fit model and post-count model with observing noise level increasing. The observing noise increases from 0.05 m to 0.1 m, and for each level we tested both algorithms by 50 Monte Carlo experiments and used the average error to depict Fig. 7.

It is clear that post-count method is more robust to noise than pre-fit method on both position error and rotation error. The reason that pre-fit model performs badly is that the larger the noise is, the less accurate the fitted features are.

E. Practical experiment

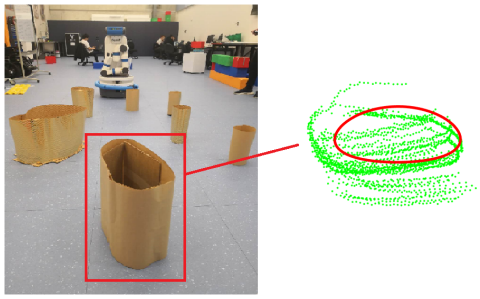
In this part we implemented our approach on a real scenario. The experiment is conducted in a lounge consisting of several near-elliptical features (7 of which are manually made and 1 of which is a round sofa) and glass wall. The data is collected via Fetch robot [18]. The data association is executed by two aspects: 1. we clustered the discrete points of elliptical features by roughly projecting points back to the initial frame via odometry information since the number of features is known and features are sparsely placed; 2. a simple way is used to associate lines. We first extract lines at each single scan and then projecting line parameters to the initial frame via odometry. As each line is represented by the distance to the line and the angle of its normal line, a threshold is selected to determine to merge the same lines.

The experiment compared our method and the state of the art Cartographer [4]. As is shown in Fig. 8, we chose 7 ellipse features and 1 irregular feature with quartic implicit function (the irregular feature) in order to evaluate our method on general shape features. The global guess of irregular feature's parameters is given by quartic parameters. The results of our method and Cartographer are depicted in Fig. 8c. Similar to simulation, line feature are depicted as segments for a better visualization. In the first row we depicted results of both Cartographer and our method together to show the difference. Since the ground truth in real scenario is not available, we cannot quantitatively evaluate the two methods. However, it is possible to compare the results by re-projecting scan points back to the initial frame via estimated poses of either method. Three rectangle areas are highlighted in the figure. In region A, more points of Cartographer exceed features' boundaries, while our method can maintain the basic shape of features. In region B and C, Cartographer's results show a clear dispersion compared with our method.

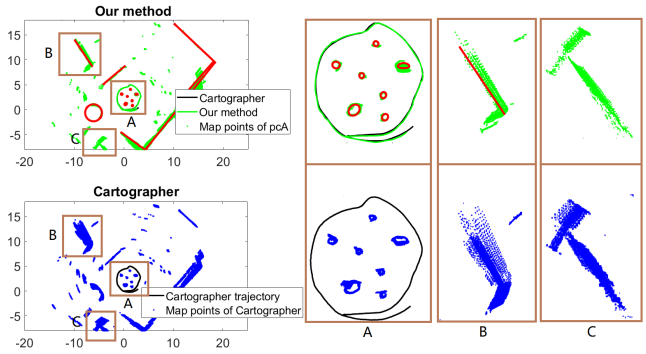
V. CONCLUSIONS AND FUTURE WORK

In this paper, a clear problem formulation and a solution framework for implicit function based SLAM problem are

(a) Practical scenario: a lounge consisting of several near-elliptical features



(b) The irregular feature. On the right is the estimated feature by our method



(c) Comparison of our method and Cartographer. Region A, B and C are highlighted to compare the results.

Fig. 8. Practical experiment.

proposed. Two challenges involved in this novel SLAM problem are addressed. One is finding the covariance of the noises involved in implicit energy terms. Another is handling the asymmetry of the energy terms for closed shape features. Simulation results using ellipse and line features as examples shows that the proposed method is more robust to observation noises and outperforms the traditional pre-fit method. It is also shown that using hybrid features can achieve better accuracy in SLAM compared with SLAM with only ellipses or lines. Practical experiment illustrates that our method has the ability to acquire accurate result.

This paper is the first step in investigating SLAM problem with implicit function as features. More experiments using actual laser data are required to further confirm the effectiveness and the performance of the proposed approach. Integrating feature identification methods such as machine learning into the proposed framework to build up a practical SLAM system is the next step of this research. The idea of this paper can be easily extended to 3D lidar-based SLAM. Effectively and accurately modeling complex 3D features using implicit functions and using them in practical 3D lidar-based SLAM and RGB-D based SLAM is our future work.

APPENDIX A

In this appendix, we give the proof for Lemma 1.

Proof: [Lemma 1] Let $f(\mathbf{x}_0, \mathbf{z}_0) = f_0$ and $J_\mathbf{x} = \frac{\partial f}{\partial \mathbf{x}}\Big|_{\mathbf{z}=\mathbf{z}_0, \mathbf{x}=\mathbf{x}_0}$. Expand $f(\mathbf{x}, \mathbf{z})$ around $\mathbf{z}_0$ and $\mathbf{x}_0$:

$$f(\mathbf{x}, \mathbf{z}) \approx f_0 + J_\mathbf{z}(\mathbf{z} - \mathbf{z}_0) + J_\mathbf{x}(\mathbf{x} - \mathbf{x}_0) \tag{23}$$

and $f(\mathbf{x}_0, \mathbf{z})$ around $\mathbf{z}_0$ is:

$$f(\mathbf{x}_0, \mathbf{z}) \approx f_0 + J_\mathbf{z}(\mathbf{z} - \mathbf{z}_0) = \mathbf{Z}. \tag{24}$$

Since $\mathbf{z}$ has zero-mean Gaussian noise $\mathbf{n}_\mathbf{z}$, the probability distribution of $\mathbf{Z}$ yields:

$$\mathbf{Z} \sim N(f_0, (J_\mathbf{z} P_z^{-1} J_\mathbf{z}^\intercal)^{-1}) \tag{25}$$

Hence, $f(\mathbf{x}_0, \mathbf{z})$ approximately follows:

$$f(\mathbf{x}_0, \mathbf{z}) \sim N(f_0, (J_\mathbf{z} P_z^{-1} J_\mathbf{z}^\intercal)^{-1}) \tag{26}$$

Then the minimizing problem can be rewriten by:

$$\begin{aligned} \underset{\mathbf{x}}{\operatorname{argmin}}\ F &\approx \|f_0 + J_\mathbf{z}(\mathbf{z} - \mathbf{z}_0) + J_\mathbf{x}(\mathbf{x} - \mathbf{x}_0)\|^2_{\Sigma_f} \\ &= \|\mathbf{Z} - A\Delta\mathbf{x}\|^2_{\Sigma_f} \end{aligned} \tag{27}$$

where

$$A = -J_\mathbf{x}, \quad \Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_0 \tag{28}$$

Thus a good choice of Σ_f^{-1} is $\Sigma_f^{-1} = J_\mathbf{z} P_z^{-1} J_\mathbf{z}^\intercal$. ■

APPENDIX B

In this appendix, a lemma useful for Line feature is provided.

Lemma 2: Assume an arbitrary line feature $\mathbf{L}_k$ is defined in the global frame $\{G\}$. Suppose $T_i \in \mathrm{SE}(2)$ denotes the transformation from local frame $\{i\}$ to global frame $\{G\}$, then the corresponding line feature in the frame $\{i\}$ is:

$${}^{\{i\}}\mathbf{L}_k = T_i^\intercal \mathbf{L}_k \tag{29}$$

Proof: Suppose an arbitrary point ${}^{\{i\}}\hat{\mathbf{p}}$ is allocated on the line ${}^{\{i\}}\mathbf{L}_k$ and both the point and the line are defined in the frame $\{i\}$. Then the point in the frame $\{G\}$ can be obtained by $\hat{\mathbf{p}} = T_i {}^{\{i\}}\hat{\mathbf{p}}$, which yields $\hat{\mathbf{p}}^\intercal \mathbf{L}_k = 0$. By direct calculation,

$$\hat{\mathbf{p}}^\intercal \mathbf{L}_k = (T_i {}^{\{i\}}\hat{\mathbf{p}})^\intercal \mathbf{L}_k = {}^{\{i\}}\hat{\mathbf{p}}^\intercal (T_i^\intercal \mathbf{L}_k) \tag{30}$$

Hence ${}^{\{i\}}\mathbf{L}_k = T_i^\intercal \mathbf{L}_k$. ■

APPENDIX C

In this appendix, we give the cost function and Jacobian equations for ellipse and line.

A. Ellipse

The final ellipse implicit function written in vector form and the energy function is:

$$\begin{aligned} g(Z_{\text{ep},i,j}, \boldsymbol{\Xi}_i, \mathbf{F}_j) &= g(Z_{\text{ep},i,j}, \boldsymbol{\Xi}_i, \boldsymbol{\Phi}_j) = \bar{\boldsymbol{\Phi}}_j^{\star}(T(\boldsymbol{\Xi}_i, Z_{\text{ep},i,j})) \\ &= \log\left(\text{sum}_2\left(A^{\intercal} M \odot A^{\intercal}\right)\frac{1}{F_{r_1}^2} + \right. \\ &\qquad \left. \text{sum}_2\left(A^{\intercal} N \odot A^{\intercal}\right)\frac{1}{F_{r_2}^2}\right) \\ E_{\text{feature},j}^{\star} &= \frac{1}{2}\sum_i^n \left\|\bar{\boldsymbol{\Phi}}_j^{\star}(T(\boldsymbol{\Xi}_i, Z_{\text{ep},i,j}))\right\|_{\Sigma_{\boldsymbol{\Phi}_j,i}}^2 \end{aligned} \tag{31}$$

where $\odot$ is the Hadamard product, sum_2 returns a column vector containing the sum of each row, and

$$\begin{aligned} A &= T(\boldsymbol{\Xi}_i, Z_{\text{ep},i,j}) - \mathbf{F}_{j_{xy}}\mathbf{1}_{w^{\star}}^{\intercal} \\ M &= \begin{bmatrix} c^2 & cs \\ cs & s^2 \end{bmatrix} \quad N = \begin{bmatrix} s^2 & -cs \\ -cs & c^2 \end{bmatrix} \end{aligned} \tag{32}$$

$\Sigma_{\boldsymbol{\Phi}_j,i}$ can be calculated according to Lemma 1:

$$\begin{aligned} \Sigma_{\boldsymbol{\Phi}_j,i}^{-1} &= \triangledown g_{ij}\Sigma_z^{-1}\triangledown g_{ij}^{\intercal} \\ \triangledown g_{ij} &= \left.\frac{\partial \bar{\boldsymbol{\Phi}}_j^{\star}(T(\boldsymbol{\Xi}_i, Z_{\text{ep},i,j}))}{\partial p_{\boldsymbol{\Phi}_j}}\right|_{\{i\}p_{\boldsymbol{\Phi}_j}, \boldsymbol{\Xi}_i, \boldsymbol{\Phi}_j} \\ &= \frac{1}{C_0}\left(\frac{2\Delta^{\intercal} MR}{F_{r_1}^2} + \frac{2\Delta^{\intercal} NR}{F_{r_2}^2}\right) \\ C_0 &= C|_{\{i\}p_{\boldsymbol{\Phi}_j}, \boldsymbol{\Xi}_i, \boldsymbol{\Phi}_j} \\ \Delta &= T(\boldsymbol{\Xi}_i, {}^{\{i\}}p_{\boldsymbol{\Phi}_j}) - \mathbf{F}xy \end{aligned} \tag{33}$$

We give a simplified example of one point in a feature. Suppose a single point $\mathbf{p} = [x_i; y_i]$. Also denote $\boldsymbol{\Phi}_{j_{xy}} = [F_x; F_y]$, $\cos(F_\phi)$ as c, and $\sin(F_\phi)$ as s. Thus, the updated Ellipse cost function turns to:

$$\begin{aligned} E_{\text{feature},j}^{\star} &= \frac{1}{2}\left\|g(Z_{\text{ep},i,j}, \boldsymbol{\Xi}_i, \boldsymbol{\Phi}_j)\right\|_{\Sigma_{\boldsymbol{\Phi}_j,i}}^2 = \\ &\frac{1}{2}\left\|\log\left(\frac{1}{F_{r_1}}\underbrace{\left(\mathbf{p}-\boldsymbol{\Phi}_{j_{xy}}\right)^{\intercal} M \left(\mathbf{p}-\boldsymbol{\Phi}_{j_{xy}}\right)}_{A} + \right.\right. \\ &\left.\left.\frac{1}{F_{r_2}}\underbrace{\left(\mathbf{p}-\boldsymbol{\Phi}_{j_{xy}}\right)^{\intercal} N \left(\mathbf{p}-\boldsymbol{\Phi}_{j_{xy}}\right)}_{B}\right)\right\|_{\Sigma_{\boldsymbol{\Phi}_j,i}}^2 \\ &= \left\|\log\left(\underbrace{\frac{A}{F_{r_1}^2} + \frac{B}{F_{r_2}^2}}_{C}\right)\right\|_{\Sigma_{\boldsymbol{\Phi}_j,i}}^2 \end{aligned} \tag{34}$$

Then the corresponding Jacobian can be obtained by partially differential g with related to the state vector.

$$\frac{\partial g}{\partial \boldsymbol{\Psi}} = \sum^n \frac{1}{C}\cdot\frac{\partial C}{\partial \boldsymbol{\Psi}} \tag{35}$$

where

$$\begin{aligned} \frac{\partial C}{\partial \mathbf{t}} &= \frac{1}{F_{r_1}^2}\frac{\partial A}{\partial \mathbf{t}} + \frac{1}{F_{r_2}^2}\frac{\partial B}{\partial \mathbf{t}} \\ &\Rightarrow \begin{cases} \frac{\partial A}{\partial \mathbf{t}} = 2\left(\mathbf{p}-\boldsymbol{\Phi}_{j_{xy}}\right)^{\intercal} M = 2\mathbf{D}^{\intercal} \\ \frac{\partial B}{\partial \mathbf{t}} = 2\left(\mathbf{p}-\boldsymbol{\Phi}_{j_{xy}}\right)^{\intercal} N = 2\mathbf{E}^{\intercal} \end{cases} \end{aligned}$$

$$\begin{aligned} \frac{\partial C}{\partial \phi} &= \frac{1}{F_{r_1}^2}\frac{\partial A}{\partial \phi} + \frac{1}{F_{r_2}^2}\frac{\partial B}{\partial \phi} \\ &\Rightarrow \begin{cases} \frac{\partial A}{\partial \phi} = 2\mathbf{D}^{\intercal}(dR)^{\{i\}}\mathbf{p} \\ \\ \frac{\partial B}{\partial \phi} = 2\mathbf{E}^{\intercal}(dR)^{\{i\}}\mathbf{p} \end{cases} \end{aligned}$$

where dR is the derivative of R

$$\begin{aligned} \frac{\partial C}{\partial \boldsymbol{\Phi}_{j_{xy}}} &= \frac{1}{F_{r_1}^2}\frac{\partial A}{\partial \boldsymbol{\Phi}_{j_{xy}}} + \frac{1}{F_{r_2}^2}\frac{\partial B}{\partial \boldsymbol{\Phi}_{j_{xy}}} = -\frac{\partial C}{\partial \mathbf{t}} \\ &\Rightarrow \begin{cases} \frac{\partial A}{\partial F_{xy}} = -2D \\ \\ \frac{\partial B}{\partial F_{xy}} = -2E \end{cases} \end{aligned} \tag{36}$$

$$\begin{aligned} \frac{\partial C}{\partial F_\phi} &= \frac{1}{F_{r_1}^2}\frac{\partial A}{\partial F_\phi} + \frac{1}{F_{r_2}^2}\frac{\partial B}{\partial F_\phi} \\ &\Rightarrow \begin{cases} \frac{\partial A}{\partial F_\phi} = \left(\mathbf{p}-\boldsymbol{\Phi}_{j_{xy}}\right)^{\intercal} dM \left(\mathbf{p}-\boldsymbol{\Phi}_{j_{xy}}\right) \\ \\ \frac{\partial B}{\partial F_\phi} = \left(\mathbf{p}-\boldsymbol{\Phi}_{j_{xy}}\right)^{\intercal} dN \left(\mathbf{p}-\boldsymbol{\Phi}_{j_{xy}}\right) \end{cases} \end{aligned}$$

where $dM = \begin{bmatrix} -2cs & -s^2+c^2 \\ -s^2+c^2 & 2cs \end{bmatrix}$

and $dN = \begin{bmatrix} 2cs & s^2-c^2 \\ s^2-c^2 & -2cs \end{bmatrix}$

$$\frac{\partial C}{\partial F_{r_1}} = -\frac{2A}{F_{r_1}^3} \quad , \quad \frac{\partial C}{\partial F_{r_1}} = -\frac{2B}{F_{r_2}^3}$$

B. Line

The line feature's implicit function can be formulated with the help of Lemma 2:

$$\begin{aligned} g(Z_{\text{lp},i,k}, \boldsymbol{\Xi}_i, \boldsymbol{\Phi}_k) &= \bar{\boldsymbol{\Phi}}_k(T(\boldsymbol{\Xi}_i, Z_{\text{lp},i,k})) \\ &= T(\boldsymbol{\Xi}_i, Z_{\text{lp},i,k})^{\intercal}\overrightarrow{\boldsymbol{\Phi}}_k^{\circ} \end{aligned} \tag{37}$$

where $\overrightarrow{\boldsymbol{\Phi}}_k^{\circ} = [\cos l_\alpha, \sin l_\alpha, -p]^{\intercal}$ is the normalized vector of $\overrightarrow{\boldsymbol{\Phi}}_k$. Hence, the energy functions of line is:

$$E_{\text{feature},k} = \frac{1}{2}\sum_i^n \left\|\bar{\boldsymbol{\Phi}}_k(T(\boldsymbol{\Xi}_i, Z_{\text{lp},i,k})\right\|_{\Sigma_{\boldsymbol{\Phi}_k,i}}^2 \tag{38}$$

The covariance $\Sigma_{\boldsymbol{\Phi}_k,i}$ is calculated as:

$$\begin{aligned} \Sigma_{\boldsymbol{\Phi}_k,i}^{-1} &= \triangledown h_{ik}\Sigma_z^{-1}\triangledown h_{ik}^{\intercal} \\ \triangledown h_{ik} &= \left.\frac{\partial h_{ik}}{\partial p_t}\right|_{\{i\}p_{\boldsymbol{\Phi}_k}, \boldsymbol{\Xi}_i, \boldsymbol{\Phi}_k} \\ &= \overrightarrow{\boldsymbol{\Phi}_{k_{12}}}^{\circ\intercal} \end{aligned} \tag{39}$$

$\overrightarrow{\mathbf{\Phi}_{k_{12}}}^{\circ}$ is the first two elements of $\overrightarrow{\mathbf{\Phi}_{k}}^{\circ}$.

Similarly, a simplified example is given based on the existing normalization. Let $\overrightarrow{\mathbf{\Phi}_k} = [l_\alpha; p]$ be the normalized feature state and $\overrightarrow{\mathbf{\Phi}_k}^{\circ} = [\cos l_\alpha; \sin l_\alpha; -p]$. Then the Jacobian matrix is derived by:

$$\begin{aligned}
\frac{\partial E_{\text{feature},k}}{\partial \mathbf{t}} &= \overrightarrow{\mathbf{\Phi}_{k_{12}}}_{\circ}^{\intercal} \\
\frac{\partial E_{\text{feature},k}}{\partial \phi} &= \overrightarrow{\mathbf{\Phi}_{k_{12}}}_{\circ}^{\intercal} (dR)^{\{i\}}\mathbf{p} \\
\frac{\partial E_{\text{feature},k}}{\partial l_\alpha} &= T^{-1}(\mathbf{\Xi}_i, {}^{\{i\}}\mathbf{p})^{\intercal} \begin{bmatrix} -\sin l_\alpha \\ \cos l_\alpha \end{bmatrix} \\
\frac{\partial E_{\text{feature},k}}{\partial p} &= -1
\end{aligned} \tag{40}$$

REFERENCES

[1] R. Ren, H. Fu, and M. Wu, "Large-scale outdoor slam based on 2d lidar," *Electronics*, vol. 8, no. 6, p. 613, 2019.

[2] G. Wilson, C. Pereyda, N. Raghunath, G. de la Cruz, S. Goel, S. Nesaei, B. Minor, M. Schmitter-Edgecombe, M. E. Taylor, and D. J. Cook, "Robot-enabled support of daily activities in smart home environments," *Cognitive Systems Research*, vol. 54, pp. 258–272, 2019.

[3] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. Jesus, R. Berriel, T. Paixão, F. Mutz, *et al.*, "Self-driving cars: A survey," *arXiv preprint arXiv:1901.04407*, 2019.

[4] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1271–1278.

[5] M. Labbé and F. Michaud, "Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *Journal of Field Robotics*, vol. 36, no. 2, pp. 416–446, 2019.

[6] J. Guivant and E. Nebot, "Simultaneous localization and map building: Test case for outdoor applications," in *IEEE Int. Conference on Robotics and Automation*, 2002.

[7] S. Kim and S.-Y. Oh, "Slam in indoor environments using omni-directional vertical and horizontal line features," *Journal of Intelligent and Robotic Systems*, vol. 51, no. 1, pp. 31–43, 2008.

[8] J. Zhao, S. Huang, L. Zhao, Y. Chen, and X. Luo, "Conic feature based simultaneous localization and mapping in open environment via 2d lidar," *IEEE Access*, vol. 7, pp. 173 703–173 718, 2019.

[9] M. Liu, S. Huang, and G. Dissanayake, "Feature based slam using laser sensor data with maximized information usage," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 1811–1816.

[10] I. Aloise, B. Della Corte, F. Nardi, and G. Grisetti, "Systematic handling of heterogeneous geometric primitives in graph-slam optimization," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2738–2745, 2019.

[11] H. Zhang, K. Hasith, and H. Wang, "A hybrid feature parametrization for improving stereo-slam consistency," in *2017 13th IEEE International Conference on Control & Automation (ICCA)*. IEEE, 2017, pp. 1021–1026.

[12] B. Holỳ, "Registration of lines in 2d lidar scans via functions of angles," *IFAC-PapersOnLine*, vol. 49, no. 5, pp. 109–114, 2016.

[13] D. Rao, S.-J. Chung, and S. Hutchinson, "Curveslam: An approach for vision-based navigation without point features," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 4198–4204.

[14] L. Pedraza, D. Rodriguez-Losada, F. Matia, G. Dissanayake, and J. V. Miró, "Extending the limits of feature-based slam with b-splines," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 353–366, 2009.

[15] 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, 2019.

[16] J. Zhao, L. Zhao, S. Huang, and Y. Wang. 2d laser slam with general features represented by implicit functions (full version). Github. [Online]. Available: https://github.com/JiahengZhao/ImplicitFunction

[17] L. Zhao, S. Huang, L. Yan, and G. Dissanayake, "A new feature parametrization for monocular slam using line features," *Robotica*, vol. 33, no. 3, pp. 513–536, 2015.

[18] M. Wise, M. Ferguson, D. King, E. Diehr, and D. Dymesich, "Fetch and freight: Standard platforms for service robot applications," in *Workshop on autonomous mobile service robots*, 2016.