# 2D Laser SLAM with General Features Represented by Implicit Functions (Full Version)

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 methods 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 outperforms the pre-fit method, 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].

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 using the optimized poses and the laser scans. The limitations of scan matching based approaches include (1) no prior knowledge of the geometry information of the environment is used, (2) heavy computational cost, (3) difficult to fuse information of consecutive scans.

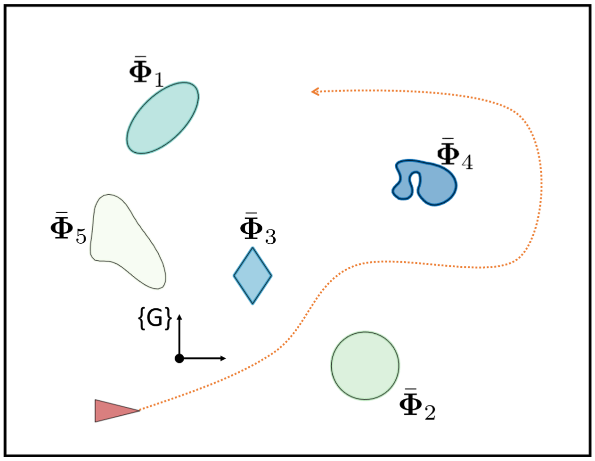


Fig. 1. Schematic diagram of SLAM with implicit functions. Red rectangle is the robot starting at [-8, -8]$^\intercal$. Features with implicit functions are defined by:
$\bar{\mathbf{\Phi}}_1 \overset{\text{def.}}{=} 0.6211\mathbf{x}^2 - 0.75\mathbf{x}\mathbf{y} + 0.6289\mathbf{y}^2 + 11.2028\mathbf{x} - 10.7891\mathbf{y} + 1400.4$;
$\bar{\mathbf{\Phi}}_2 \overset{\text{def.}}{=} \mathbf{x}^2 - 6\mathbf{x} + \mathbf{y}^2 + 12\mathbf{y} + 42.74$;
$\bar{\mathbf{\Phi}}_3 \overset{\text{def.}}{=} 1.2|\mathbf{x}+2| + 0.8|\mathbf{y}+2| - 1$;
$\bar{\mathbf{\Phi}}_4 \overset{\text{def.}}{=} 3(1-5(\mathbf{x}-5))^2 e^{-(5(\mathbf{x}-5)^2)-(5(\mathbf{y}-2)+1)^2} - 10((\mathbf{x}-5) - 5(\mathbf{x}-5)^3 - 3(\mathbf{y}-2)^5) e^{-2(\mathbf{x}-5)^2 - 5(\mathbf{y}-2)^2} - \frac{1}{3} e^{-(5(\mathbf{x}-5)+1)^2 - 4(\mathbf{y}-2)^2} - 0.1$;
$\bar{\mathbf{\Phi}}_5 \overset{\text{def.}}{=} 1.2\mathbf{x}^4 + 0.4\mathbf{y}^4 + 2\mathbf{x}\mathbf{y} + 2.3\mathbf{x}^3\mathbf{y} - 2$.
We abuse $\mathbf{x}$ and $\mathbf{y}$ as coordinates of points belonging to each feature.

Feature based approach estimates the parameters of the feature present in the environment. One basic feature based SLAM is point feature SLAM [5] where the feature parameter is the position of the point feature. Other features used in laser SLAM include line feature [6], ellipse feature [7], curve feature [8] and so on. A unified formulation named *matchable* was employed in [9] to represent point, line and plain features. Zhang et al. [10] utilized remote and near feature parametrization to improve the robustness of rotation estimation. Holỳ [11] combined points and lines for the scan matching to decrease the computation time and increase the accuracy. Our previous work [7] imported ellipse feature to reduce the number of points needed during calculation and to compensates for errors in observations from different perspectives. Rao et al. [12] extracted Bézier curves and used four control points to parameterize curve features, then the optimization problem is solved by Levenberg-Marquardt algorithm. Pedraza et al. [13] are the first to use spline to parameterize features and then optimize robot poses and control points simultaneously.

In most of the feature based SLAM approaches, the geometric features are represented by a small fixed number

[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`

of parameters, and the laser points associated to a particular feature are used to estimate the parameters of the feature (in the local frame), the estimated parameters are then used as virtual observations in the filter or optimization backend. Compared with scan matching based methods, feature-based method needs less resources and calculation, and some geometrical features can provide more credible constraints, which enhance the accuracy of localization and mapping.

However, most of the existing feature based SLAM can only handle limited types of geometric features. In reality, laser scans reflect the boundary of occupied area and unoccupied area, and the boundary of an object in the environment could be of arbitrary shape, which cannot be easily described using feature parameters. On the other hand, most of the boundary 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 functions with certain parameters. 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 overcome one challenge of correctly computing the observation covariance matrix. Then we propose a potential framework which can be adopted for all the types of features instead of regular shapes. 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 compared 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 proposed a framework for implicit function based SLAM problem by clearly formulating the problem with implicit functions to represent features, computing corresponding implicit covariance strictly, and present 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 pre-fit method and proved the superiority of our method.
- We developed a novel cost function for features with closed shapes to enhance the decline during iteration. We reconstruct the original objective function and use the logarithmic form to improve the convergence of the function.

## II. SLAM PROBLEM WITH GENERAL FEATURES DESCRIBED BY IMPLICIT FUNCTIONS

In this section, we illustrate a general SLAM problem with implicit objective function 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 point set with $n$ points are noted by $\mathbf{P} \in \mathbb{R}^{n\times 2}$, and the corresponding homogeneous point set is $\hat{\mathbf{P}}$. We also assume observed points have a zero-mean Gaussian noise $\mathbf{n}_z \sim N(\mathbf{0}, \Sigma_z)$ and $\Sigma_z \in \mathbb{R}^{2\times 2}$.

For an angle $\phi \in \mathbb{R}$, let $R(\phi) \in SO(2)$ be the corresponding rotation matrix $R(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$, 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. $T_{ij}$ means the transformation of pose $j$ relative to pose $i$. We always omit $i$ to represent the pose from pose $j$ to pose 0 (origin of the global coordinate frame) by $T_j$. For example, suppose a robot pose is $\mathbf{\Xi}_j = [\mathbf{t}_j; \phi_j]$, then $T_j = \begin{bmatrix} R_j & \mathbf{t}_j \\ \mathbf{0} & 1 \end{bmatrix}$. We use $T(\mathbf{\Xi}_j, {}^{\{j\}}\mathbf{p})$ to represent the process of transforming a point from $\{j\}$ frame ${}^{\{j\}}\mathbf{p}$ to global frame $\{G\}$, and $T^{-1}(\mathbf{\Xi}_j, {}^{\{G\}}\mathbf{p})$ indicates the opposite process[1].

### 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 environment is shown in Fig. 4. The state is denoted by:

$$\mathbf{\Psi} = \begin{Bmatrix} \mathbf{\Xi}_1 & \mathbf{\Xi}_2 & \mathbf{\Xi}_3 & \cdots & \mathbf{\Xi}_n \\ \mathbf{\Phi}_1 & \mathbf{\Phi}_2 & \cdots & \mathbf{\Phi}_q & \end{Bmatrix} \tag{1}$$

under the assumption that initial pose $\mathbf{\Xi}_0 = [0; 0; 0]$. Noting that the number of variables for each feature is different. Then the problem is formulated by 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:

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

[1] Sometimes we abuse the notation and use the same functions $T(\mathbf{\Xi}_j, {}^{\{j\}}\mathbf{P})$ and $T^{-1}(\mathbf{\Xi}_j, {}^{\{G\}}\mathbf{P})$ to indicate transformation of a point set.

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 $i$[2]. $f(\mathbf{Z}_{\text{odom},i}, \Xi_{i-1}, \Xi_i)$ and $g(\mathbf{Z}_{\text{feature},i,j}, \Xi_i, \mathbf{\Phi}_j)$ are the cost functions for the two entries, respectively. $\Sigma_{\text{odom},i}$ is 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:

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

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

In order to evaluate approaches fairly, we assumed that which feature the collected data belongs to is known in the following sections.

### C. SLAM with features represented by implicit functions

Suppose an implicit function $\bar{\mathbf{\Phi}}_j(\mathbf{P}) = \mathbf{0}$ holds[3] 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 $\bar{\mathbf{\Phi}}_1$ to $\bar{\mathbf{\Phi}}_5$ hold for every point locating on each feature respectively.

If the observation of $\mathbf{\Phi}_j$ at pose $i$ are raw points set, denoted by ${}^{\{i\}}\mathbf{S}$, it must satisfy the implicit function after transformed to global frame, that is:

$$\bar{\mathbf{\Phi}}_j({}^{\{G\}}\mathbf{S}) = \bar{\mathbf{\Phi}}_j(T^{-1}(\Xi_i, {}^{\{i\}}\mathbf{S})) = \mathbf{0} \tag{5}$$

Eq. (5) is obviously an implicit function. Still taking feature $j$ as an instance, the energy term of feature $j$ turns to be:

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

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.

### D. Implicit covariance

In typical least squares problems, the energy term is $E = \|\mathbf{z} - f(\mathbf{x})\|^2_{\Sigma}$ and $\Sigma$ is the covariance of $\mathbf{z}$. 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 $\Sigma_{\mathbf{\Phi}_j,i}$.

*Lemma 1:* Consider the problem of estimating the covariance of the noises on $f(\mathbf{x}, \mathbf{z})$ where $\mathbf{x}$ are variables and $\mathbf{z}$ are observations. Suppose the noise of $\mathbf{z}$ is $\mathbf{n}_{\mathbf{z}}$ which is zero-mean Gaussian $\mathbf{n}_{\mathbf{z}} \sim N(\mathbf{0}, P_z)$, then $f(\mathbf{x}, \mathbf{z})$ approximately follows Gaussian noise at $\mathbf{z} = \mathbf{z}_0, \mathbf{x} = \mathbf{x}_0$ as:

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

where

$$\Sigma_f^{-1} = J_{\mathbf{z}} P_z^{-1} J_{\mathbf{z}}^{\intercal} \ , \quad J_{\mathbf{z}} = \frac{\partial f}{\partial \mathbf{z}}\Big|_{\mathbf{z}=\mathbf{z}_0, \mathbf{x}=\mathbf{x}_0} \tag{8}$$

*Proof:* See Appendix A. ■

Since it is difficult to obtain ground truth of $\mathbf{z}$, we use observed points to approximately calculate the covariance in Eq. (6) (Please refer to Remark 1 and Section IV-B).

### 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 $\mathbf{\Phi}_j$ is defined by $s$ elements in 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-Marguardt can be used to solve the problem. Here we only consider feature energy function Eq. (6).

To be more simplified, only feature $j$ is considered in Eq. (3) and the abbreviation form is took as $E = \frac{1}{2}\|\bar{\mathbf{\Phi}}_j(\Xi, \mathbf{Z})\|^2_{\Sigma_{\mathbf{\Phi}_j}}$. The Jacobian matrix is formulated by:

$$J_j = \begin{bmatrix} \frac{\partial \bar{\mathbf{\Phi}}_j(\Xi,\mathbf{Z})}{\partial \Xi} & \frac{\partial \bar{\mathbf{\Phi}}_j(\Xi,\mathbf{Z})}{\partial \Phi_{j_1}} & \cdots & \frac{\partial \bar{\mathbf{\Phi}}_j(\Xi,\mathbf{Z})}{\partial \Phi_{j_s}} \end{bmatrix} \tag{9}$$

noting that the first element in $J_j$ is the derivative of all the poses.

Suppose an initial $\mathbf{\Psi}$ and the incremental $\Delta\mathbf{\Psi}$ are defined by:

$$\mathbf{\Psi} \overset{\text{def.}}{=} (\Xi, \overrightarrow{\mathbf{\Phi}_j}) \quad , \quad \Delta\mathbf{\Psi} \overset{\text{def.}}{=} (\Delta\Xi, \Delta\overrightarrow{\mathbf{\Phi}_j}) \tag{10}$$

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

$$\mathbf{\Psi}_{\text{new}} \overset{\text{def.}}{=} \mathbf{\Psi} \oplus \Delta\mathbf{\Psi} \overset{\text{def.}}{=} (\Xi \oplus \Delta\Xi, \mathbf{\Phi}_j \oplus \Delta\overrightarrow{\mathbf{\Phi}_j}) \tag{11}$$

The step increment $\Delta\mathbf{\Psi}$ can be calculated by LM method:

$$\Delta\mathbf{\Psi} = -(J_j^{\intercal} \Sigma_{\mathbf{\Phi}_j}^{-1} J_j + \mu I)^{-1} J_j^{\intercal} \Sigma_{\mathbf{\Phi}_j}^{-1} \bar{\mathbf{\Phi}}_j \tag{12}$$

Here $\mu$ is a damping parameter and $I$ is an identity matrix. Please note that during each iteration, the covariance adaptively varies according to the updated implicit function.

It is worth noting that the implicit function allows very flexible representation of the features in the environments. The "changeable parameters" of each feature can vary based on the prior knowledge and any updated information.

[2] 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 Section III-A and III-B (implicit function), and Section III-D (pre-fit)

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

TABLE I

COMPARISON OF PRE-FIT METHOD AND POST-COUNT METHOD

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

† Different feature-based SLAM methods differ in parametrization.

‡ Under the assumption that each kind of features possesses an unique implicit function which holds for all the points belonging to the feature.

†‡ Observation vector consists of single observations at each pose, at both functions are simplified by integrating $n$ poses.

‡‡ $\mathbf{\Phi}_j$ is not the same for two methods.

**Algorithm 1: The process to solve the problem**

```
Input: $\mathbf{Z}_{\text{odom}}$, $\mathbf{Z}_{\text{feature},j}$
Initialization;
$found = $ false; $k = 1$;
while (not $found$) & ( $k \leq k_{\max}$) do
    Solve $\Delta\mathbf{\Psi}$ according to Eq. (10) and Eq. (12);
    if Optimal then
        $found$ = true;
    else
        $\mathbf{\Psi} = \mathbf{\Psi} \oplus \Delta\mathbf{\Psi}$
    end
end
Output: $\mathbf{\Psi}^\star = \mathbf{\Psi}$
```

### F. An improved objective function for closed shape

One inevitable problem of features' implicit functions is: for features with closed shapes, the value of implicit functions $\bar{\mathbf{\Phi}}_j$ in the error term $E_{\text{feature},j}$ (We simplified the expression) various from $-a$ (where $a > 0$, varying from different feature's implicit functions) to 0 and then from 0 to $+\infty$ (see Fig. 2), which will make it difficult to quickly decline to the optimal solution when the virtual observation points are inside the boundary during iterations.

Denote the problem with the original energy term is $\bar{\mathbf{\Phi}}_j$ while the improved objective function as $\bar{\mathbf{\Phi}}_j^\star$. We improved the objective function by:

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

$\varrho$ is a compensation argument that serves two purposes: (a) to ensure that the domain of new function is $(0, +\infty)$; (b) to keep the minimal value at the boundary. Generally $\varrho$ is the reciprocal of $a$.

As is shown in Fig. 2, taking $\mathbf{\Phi}_5$ (in Fig. 1) as an example, the left figure is a general implicit function defined as:

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

where a point set is $\mathbf{P} = [\mathbf{x}, \mathbf{y}] \in \mathbb{R}^{n\times 2}$ 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, we apply a logarithm to reconstruct the original implicit function as:

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

where $\varrho = 6.4817$ for $\mathbf{\Phi}_5$.

The right figure depicts the improved objective function. The value of $\bar{\mathbf{\Phi}}_5^\star$ inside and outside the boundary changes relatively uniformly, showing a symmetry property and leading to a steady decline.

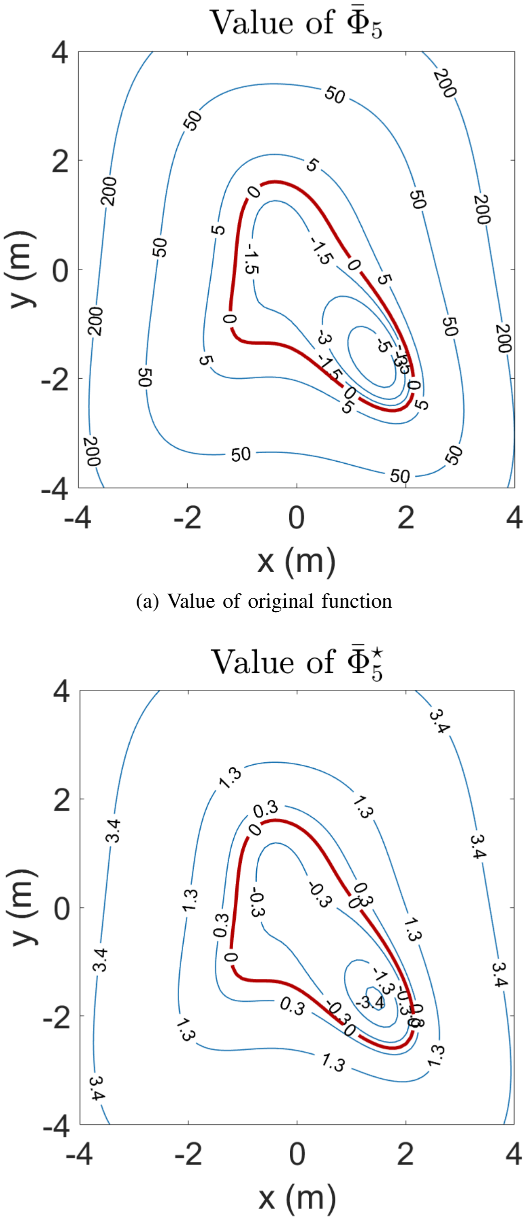
## III. EXAMPLES

In this section, we use two examples (ellipse feature and line feature) to illustrate the proposed method. Since ellipse and line can be represented using 5 and 2 parameters, respectively, we can also use the traditional pre-fit method. From now on, we call our method "post-count".

### A. Post-count Example 1: Ellipse feature

Assume ${}^{\{i\}}\hat{\mathbf{P}}_{\mathbf{\Phi}_j} \in \mathbb{R}^{w_{ij}^\star \times 3}$ is the homogeneous point set of $j$th ellipse feature in the $i$th frame. Noted that $w_{ij}^\star$ is the number of points of this point set. Then the observation is defined by $Z_{\text{ep},i,j} = {}^{\{i\}}\hat{\mathbf{P}}_{\mathbf{\Phi}_j}$.

Suppose for an arbitrary point $\mathbf{p}_a = (x_a; y_a)$ defined in the global frame and $E_{\text{feature},j} = \left\Vert\bar{\mathbf{\Phi}}_j(\mathbf{p}_a)\right\Vert^2_\Sigma$, then conventional



(a) Value of original function

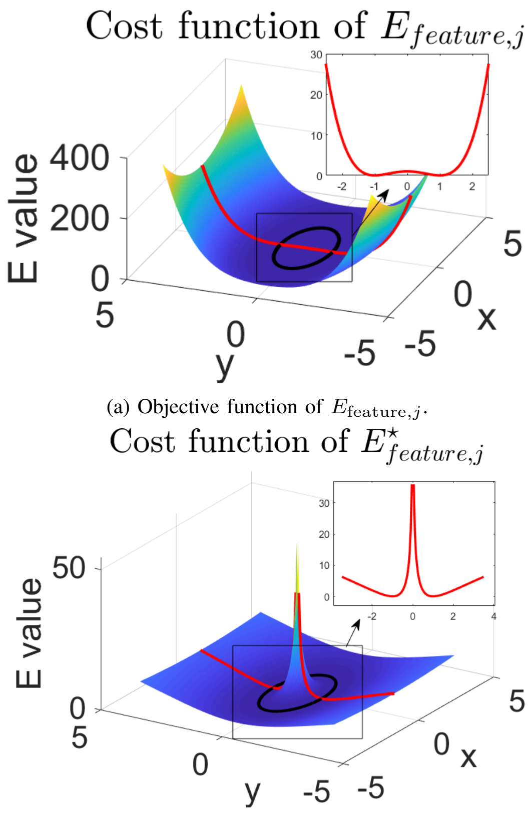
(b) Value of improved function

Fig. 2. Comparison of objective 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.

ellipse equation of $\mathbf{\Phi}_j$ is as follows:

$$\bar{\mathbf{\Phi}}_j(\mathbf{p}_a) = \frac{((x_a - F_x)\cos F_\phi + (y_a - F_y)\sin F_\phi)^2}{F_{r_1}^2} + \frac{(-(x_a - F_x)\sin F_\phi + (y_a - F_y)\cos F_\phi)^2}{F_{r_2}^2} - 1 \tag{16}$$

where $\overrightarrow{\mathbf{\Phi}_j} = [F_x, F_y, F_\phi, F_{r_1} F_{r_2}]^\intercal$ are the changeable parameters for ellipse feature. 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 **??** and Section III-E).



(a) Objective function of $E_{\text{feature},j}$.

(b) Objective function of $E^\star_{\text{feature},j}$.

Fig. 3. Objective functions of 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. As is shown in Fig. 3a, the black ellipse is the ellipse example $[0, 0, 0, 2, 1]^\intercal$ depicted in $\mathbb{R}^2$. It seems that the function can converge well for any point from a macro perspective. However, the fact is that $E_{\text{ellipse}}$ is hard to converge when the observed points locate within the ellipse area. As shown in the sub-figure, we drew a line where $x = 0$ as an instance. The value of $E_{\text{ellipse}}$ varies slighted 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:

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

The final implicit function $g(Z_{\text{ep},i,j}, \mathbf{\Xi}_i, \mathbf{\Phi}_j)$ and the energy function $E^\star_{\text{feature},j}$ is listed in Appendix B, as well as the covariance $\Sigma_{\mathbf{\Phi}_j,i}$.

Depict the new objective function in a similar way and the result is shown in Fig. 3b. In this example, the new objective function provides a sharp decline when points fall inside the ellipse. Although the gradient is not as significant as the original objective function for points $|x|, |y| \geq 1$, it is still easy to descend because there is still a large change in energy function.

### B. Post-count Example 2: Line feature

Similarly, assume ${}^{\{i\}}\hat{\mathbf{P}}_{\phi_k} \in \mathbb{R}^{v^{\star}_{ik}\times 3}$ is the homogeneous point set of $k$th line feature in the $i$th frame. Noted that $v^{\star}_{ik}$ is the number of points of this point set. And the observations are defined by $Z_{\mathrm{lp},i,k} = {}^{\{i\}}\hat{\mathbf{P}}_{\phi_k}$. $\overrightarrow{\mathbf{\Phi}_k} = [l_\alpha, p]^\intercal$ are the changeable parameters normalized by $p \geq 0$.

The line feature's implicit function $g(Z_{\mathrm{lp},i,k}, \mathbf{\Xi}_i, \mathbf{\Phi}_k)$, energy function $E_{\text{feature},k}$ and covariance $\Sigma_{\mathbf{\Phi}_k,i}$ are also listed in Appendix B.

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

In traditional feature-based SLAM using pre-fit, $\tilde{E}_{\text{feature}}$ aims to minimize the difference between actual observation $\tilde{\mathbf{Z}}_{\text{feature}}$ and theoretical observation $T^{-1}(\Xi, \mathbf{\Phi})$, and the feature can be uniquely represented by its parameter. To obtain the theoretical observations, we need to transform feature states from global frame to the corresponding local frame. Take feature $j$ as an example, and also integrate all poses for simplification, then generally the process can be expressed as:

$$\tilde{E}_{\text{feature},j} = \frac{1}{2}\sum_{i=1}^{n} \|\tilde{\mathbf{Z}}_{\text{feature},i,j} - T^{-1}(\mathbf{\Xi}_i, \mathbf{\Phi}_j)\|^2_{\tilde{\Sigma}_{\text{feature},i,j}} \tag{18}$$

noted that $\mathbf{\Phi}_j$ depends on how the feature is parameterized not just 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 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 [7]. As the correct data association is known, the observed points belonging to a certain ellipse feature at each successive timestamp are pre-fitted and reassembled into new observations.

In order to distinguish annotations from post-count method, we replace $\mathbf{\Phi}_j$ and $\mathbf{\Phi}_k$ with $\mathbf{F}_j$ and $\mathbf{L}_k$ to represent the ellipse and line feature, respectively. It is worth noting that 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.

Denote $\breve{\mathbf{Z}}_{\mathrm{ef},i,j}$ as the observation of the $j$th ellipse feature at the $i$th step, then $\breve{\mathbf{Z}}_{\mathrm{ef},i,j} = {}^{\{i\}}\breve{\mathbf{F}}_j, {}^{\{i\}}\breve{\mathbf{F}}_j$ is the observation of ${}^{\{i\}}\mathbf{F}_j$.

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

$$g(\breve{\mathbf{Z}}_{\mathrm{ef},i,j}, \mathbf{\Xi}_i, \mathbf{F}_j) = {}^{\{i\}}\breve{\mathbf{F}}_j - \begin{bmatrix} T^{-1}(\mathbf{\Xi}_i, \mathbf{F}_{j_{xy}}) \\ \mathrm{dist}(F_{j_\phi} - \phi_i) \\ \mathbf{F}_{j_{r1,r2}} \end{bmatrix}_{5\times 1} \tag{19}$$

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

The uncertainty of feature parameters is easy to compute based on existing researches. The ellipse feature uncertainty $\Sigma_{\mathrm{ef},i,j}$ is computed by $\Sigma^{-1}_{\mathrm{ef},i,j} = \mathbf{J}^\intercal \Sigma_z^{-1} \mathbf{J}$ (Discussed in our previous work Eq. 16 [7]).

### E. Pre-fit method: Line feature

Similarly, the observation of line features is obtained by intuitively minimizing the distance from discrete points to the line. Suppose a given point ${}^{\{l\}}\mathbf{p}_i = (x_i, y_i) \in \mathbb{R}^2$ is defined in the local frame $\{l\}$ and belongs to a line feature $\mathbf{L}_k$ and denote $\breve{\mathbf{Z}}_{\mathrm{lf},i,j}$ as the observation of the $k$th line feature at the $i$th step. In order to avoid excessive mapping, the observations of line $\breve{\mathbf{Z}}_{\mathrm{lf},i,j}$ are in the form of $\mathbf{L}_k$ and calculated by minimizing:

$$\underset{{}^{\{l\}}\mathbf{L}_k}{\operatorname{argmin}} \sum_i {}^{\{l\}}\mathbf{L}_k^\intercal \, {}^{\{l\}}\hat{\mathbf{p}}_i \tag{20}$$

thus $\breve{\mathbf{Z}}_{\mathrm{lf},i,j} = {}^{\{l\}}\mathbf{L}_k$.

According to Lemma 2 in Appendix B, the pre-fit line model can be written as:

$$g(\breve{\mathbf{Z}}_{\mathrm{lf},i,j}, \mathbf{\Xi}_i, \mathbf{l}_k) = {}^{\{l\}}\mathbf{L}_k - T^{-1}(\mathbf{\Xi}_i, \mathbf{L}_k) \tag{21}$$

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

According to Zhao et al. [14] (Eq. 19), the uncertainty of line energy function $\Sigma_{\mathrm{lf},i,k}$ can be developed by $\Sigma_{\mathrm{lf},i,k} = \begin{bmatrix} \Sigma_z & \\ & 0 \end{bmatrix}$.

## IV. EXPERIMENTS AND ANALYSIS

In this section we consider several numerical examples to analyze the performance of proposed method: firstly, we investigated the validity of Lemma 1; secondly, we compared the results of our method and pre-fit method; then we tested our method by fixing the covariance and changing the implicit function for ellipse feature; 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

A simulated environment was generated to test our algorithm 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 feature 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 objective function ($E^{\star}_{\text{Ellipse}}$, Eq. (16)) for both ellipse and

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 |
| $\theta$/rad | 0.0058 | 0.0068 | 0.0059 | 0.0067 | 0.0059 | **0.0043** | 0.0044 | 0.0078 | 0.0047 |

line features (pcEL_oldFun_fixCov) and the same configuration except letting covariance matrix be adaptively variable (pcEL_oldFun).

The simulated environment is a 10×10 space containing walls and ellipse features. The robot starts at $[0,0,0]^\top$ and odometry information is provided via a virtual wheel encoder with a random Gaussian noise $[0.02^2, 0.02^2, 1e^{-8}]$. The initial observation noise is a random Gaussian noise $\mathbf{n}_z : N(\mathbf{0}, \mathrm{diag}(0.05^2, 0.05^2))$. Because the line parameters used in this paper cannot represent line segments, the end points of each line feature at the first observation are maintained dependently (do not participate in the optimization. In this stage, 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.

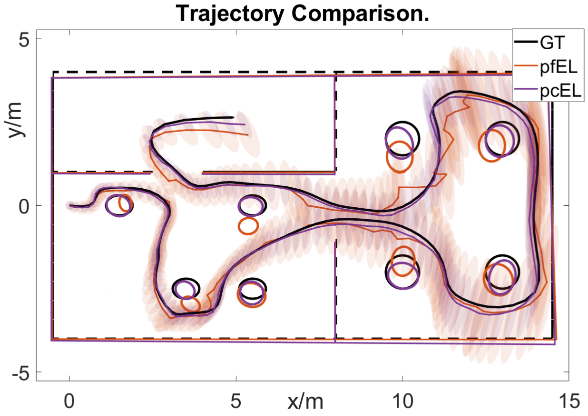
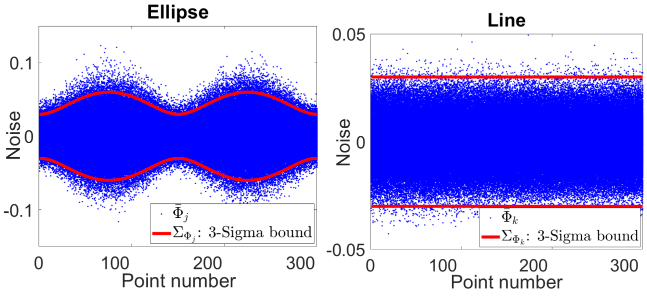


Fig. 4. Trajectory comparison. 3-sigma bound for robot's positions are depicted by shadowed ellipse in specific color.

### B. Feasibility of implicit covariance

Lemma 1 applies to any planar features, not only isolated and contrived examples such as ellipse and line features. In this part, we merely used ellipse and line feature for verification. We firstly verified Eq. (30) and Eq. (36) on points of the feature by Monte Carlo experiment. 300 points on the edge of an ellipse and a line are selected respectively. Given a random Gaussian noise $\mathbf{n}_z : N(\mathbf{0}, \mathrm{diag}(0.01^2, 0.01^2))$, we repeatedly calculated both ellipse and line's implicit functions $\bar{\mathbf{\Phi}}_j$ and $\bar{\mathbf{\Phi}}_k$ with noisy observation and $\mathbf{\Psi}$ 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 solved 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.



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

Fig. 5. Uncertainty comparison

*Remark 1:* For a real case, it is impossible to get the exact points' position by sensors. Because the noise influence of observed points is similar to that of ground truth points when the observations are near the exact positions, we choose the observed points to approximately calculate $\Sigma^{-1}_{\mathbf{\Phi}_j,i}$. As is shown in Fig. 5, although $\Sigma^{-1}_{\mathbf{\Phi}_j,i}$ cannot cover more than 99% error points, it is still an acceptable solution and the imported errors can be amended by the ellipse cost function we proposed.

### C. Results comparison

Firstly, we compared the results of the proposed method with changeable implicit covariance (pcEL) and a traditional pre-fit method with ellipse and line feature (pfEL), as shown in Fig. 4. Obviously, the trajectory of pcEL is much better than that of pfEL. Some sharp "jump" occurred for pfEL due to the badly-fitted observations. In Tab. II, the position error of pfEL is 0.1386, while that of pcEL is 0.0912, which is much smaller than pfEL. Also the final covariance of position of pcEL is smaller than that of pfEL.

Then we compared pcEL with fixing covariance (pcEL_fixCov) in the same simulation environment. In Tab. II, it can be found that each elements of pcEL_fixCov is slightly bigger than pcEL, but smaller than any pre-fit approaches and post-count approaches.

### D. Noise resistance and feature influence

The Root Mean Square Error (RMSE) of pose is shown in the Tab. II. A main conclusion is that the pre-fit model usually possesses a higher error level than post-count model. The second minor conclusion is that the combination of ellipse and line features can effectively improve the accuracy of the results, whether it is pre-fit model or post-count model.

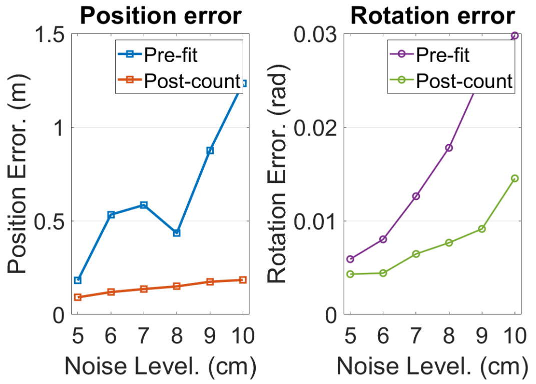


Fig. 6. Error changes with noise increasing.

In the last experiment, we tested both pre-fit model and post-count model with observing noise level increasing. The observing noise increases from $5E-2$ to $1E-1$, and for each level we tested both algorithms by 50 Monte Carlo experiments and used the average of the results to depict figure, as shown in Fig. 6.

It is clear that post-count method is more tolerant to noise than pre-fit method for 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.

## V. CONCLUSIONS AND FUTURE WORK

In this paper, a clear problem formulation and a solution framework for implicit function based SLAM problem are 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 experiment results using ellipse and line features as examples have shown that the proposed method is more resistant to observation noises and outperforms the traditional pre-fitting based method. It is also shown that using hybrid features with both ellipses and lines can achieve better accuracy in SLAM as compared with SLAM with only ellipses or lines.

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 laser-based SLAM. Effectively and accurately modeling complex 3D features using implicit functions and using them in practical 3D laser-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 $J_{\mathbf{x}} = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{z}=\mathbf{z}_0, \mathbf{x}=\mathbf{x}_0}$. Expand $f(\mathbf{x}, \mathbf{z})$ at $\mathbf{z} = \mathbf{z}_0$ and $\mathbf{x} = \mathbf{x}_0$:

$$f(\mathbf{x}, \mathbf{z}) \approx f_0 + J_{\mathbf{z}}(\mathbf{z} - \mathbf{z}_0) + J_{\mathbf{x}}(\mathbf{x} - \mathbf{x}_0) \tag{22}$$

Then the minimizing problem can be rewriten by:

$$\begin{aligned} \underset{\mathbf{x}}{\operatorname{argmin}}\ F &\approx \|f_0 + J_{\mathbf{z}}(\mathbf{z} - \mathbf{z}_0) + J_{\mathbf{x}}(\mathbf{x} - \mathbf{x}_0)\|^2_{\Sigma_f} \\ &= \|\mathbf{Z} - A\Delta\mathbf{x}\|^2_{\Sigma_f} \end{aligned} \tag{23}$$

where

$$\begin{aligned} \mathbf{Z} &= f_0 + J_{\mathbf{z}}(\mathbf{z} - \mathbf{z}_0) \\ A &= -J_{\mathbf{x}} \\ \Delta\mathbf{x} &= \mathbf{x} - \mathbf{x}_0 \end{aligned} \tag{24}$$

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

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

Thus $\Sigma_f = J_{\mathbf{z}} P_z J_{\mathbf{z}}^{\intercal}$. ■

## 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:

$${}^{\{i\}}\mathbf{L}_k = T_i^{\intercal} \mathbf{L}_k \tag{26}$$

*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,

$$\hat{\mathbf{p}}^{\intercal}\mathbf{L}_k = (T_i {}^{\{i\}}\hat{\mathbf{p}})^{\intercal}\mathbf{L}_k = {}^{\{i\}}\hat{\mathbf{p}}^{\intercal}(T_i^{\intercal}\mathbf{L}_k) \tag{27}$$

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:

$$\begin{aligned} g(Z_{\mathrm{ep},i,j}, \mathbf{\Xi}_i, \mathbf{\Phi}_j) &= \bar{\mathbf{\Phi}}_j^{\star}(T(\mathbf{\Xi}_i, Z_{\mathrm{ep},i,j})) \\ &= \log\left(\mathrm{sum}_2\left(A^{\intercal} M \odot A^{\intercal}\right)\frac{1}{F_{r_1}^2} + \right. \\ &\qquad \left. \mathrm{sum}_2\left(A^{\intercal} N \odot A^{\intercal}\right)\frac{1}{F_{r_2}^2}\right) \end{aligned} \tag{28}$$

$$E^{\star}_{\mathrm{feature},j} = \frac{1}{2}\sum_{i}^{n}\left\|\bar{\mathbf{\Phi}}_j^{\star}(T(\mathbf{\Xi}_i, Z_{\mathrm{ep},i,j}))\right\|^2_{\Sigma_{\mathbf{\Phi}_j,i}}$$

where $\odot$ is the Hadamard product, $\mathrm{sum}_2$ returns a column vector containing the sum of each row, and

$$\begin{aligned} A &= T(\mathbf{\Xi}_i, Z_{\mathrm{ep},i,j}) - \mathbf{F}_{j_{xy}} \mathbf{1}_{w^{\star}}^{\intercal} \\ M &= \begin{bmatrix} c^2 & cs \\ cs & s^2 \end{bmatrix} \quad N = \begin{bmatrix} s^2 & -cs \\ -cs & c^2 \end{bmatrix} \end{aligned} \tag{29}$$

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

$$\begin{aligned}
\Sigma_{\mathbf{\Phi}_j,i}^{-1} &= \nabla g_{ij}\Sigma_z^{-1}\nabla g_{ij}^{\intercal} \\
\nabla g_{ij} &= \left.\frac{\partial\bar{\mathbf{\Phi}}_j^{\star}(T(\mathbf{\Xi}_i, Z_{\text{ep},i,j}))}{\partial p_{\mathbf{\Phi}_j}}\right|_{{}^{\{i\}}p_{\mathbf{\Phi}_j},\mathbf{\Xi}_i,\mathbf{\Phi}_j} \\
&= \frac{1}{C_0}\left(\frac{2\Delta^{\intercal}MR}{F_{r_1}^2} + \frac{2\Delta^{\intercal}NR}{F_{r_2}^2}\right) \\
C_0 &= C|_{{}^{\{i\}}p_{\mathbf{\Phi}_j},\mathbf{\Xi}_i,\mathbf{\Phi}_j} \\
\Delta &= T(\mathbf{\Xi}_i, {}^{\{i\}}p_{\mathbf{\Phi}_j}) - \mathbf{F}_{xy}
\end{aligned} \tag{30}$$

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

$$\begin{aligned}
&E_{\text{feature},j}^{\star} = \frac{1}{2}\left\|g(Z_{\text{ep},i,j}, \mathbf{\Xi}_i, \mathbf{\Phi}_j)\right\|_{\Sigma_{\mathbf{\Phi}_j,i}}^2 = \\
&\frac{1}{2}\left\|\log\left(\frac{1}{F_{r_1}}\underbrace{\left(\mathbf{p}-\mathbf{\Phi}_{j_{xy}}\right)^{\intercal}M\left(\mathbf{p}-\mathbf{\Phi}_{j_{xy}}\right)}_{A} + \right.\right. \\
&\qquad\left.\left.\frac{1}{F_{r_2}}\underbrace{\left(\mathbf{p}-\mathbf{\Phi}_{j_{xy}}\right)^{\intercal}N\left(\mathbf{p}-\mathbf{\Phi}_{j_{xy}}\right)}_{B}\right)\right\|_{\Sigma_{\mathbf{\Phi}_j,i}}^2 \\
&= \left\|\log\left(\underbrace{\frac{A}{F_{r_1}^2} + \frac{B}{F_{r_2}^2}}_{C}\right)\right\|_{\Sigma_{\mathbf{\Phi}_j,i}}^2
\end{aligned} \tag{31}$$

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

$$\frac{\partial g}{\partial\mathbf{\Psi}} = \sum^{n}\frac{1}{C}\cdot\frac{\partial C}{\partial\mathbf{\Psi}} \tag{32}$$

where

$$\frac{\partial C}{\partial\mathbf{t}} = \frac{1}{F_{r_1}^2}\frac{\partial A}{\partial\mathbf{t}} + \frac{1}{F_{r_2}^2}\frac{\partial B}{\partial\mathbf{t}}$$

$$\Rightarrow\begin{cases}\frac{\partial A}{\partial\mathbf{t}} = 2\left(\mathbf{p}-\mathbf{\Phi}_{j_{xy}}\right)^{\intercal}M = 2\mathbf{D}^{\intercal} \\ \frac{\partial B}{\partial\mathbf{t}} = 2\left(\mathbf{p}-\mathbf{\Phi}_{j_{xy}}\right)^{\intercal}N = 2\mathbf{E}^{\intercal}\end{cases}$$

$$\frac{\partial C}{\partial\phi} = \frac{1}{F_{r_1}^2}\frac{\partial A}{\partial\phi} + \frac{1}{F_{r_2}^2}\frac{\partial B}{\partial\phi}$$

$$\Rightarrow\begin{cases}\frac{\partial A}{\partial\phi} = 2\mathbf{D}^{\intercal}(dR)^{\{i\}}\mathbf{p} \\ \\ \frac{\partial B}{\partial\phi} = 2\mathbf{E}^{\intercal}(dR)^{\{i\}}\mathbf{p}\end{cases}$$

where $dR$ is the derivative of $R$

$$\frac{\partial C}{\partial\mathbf{\Phi}_{j_{xy}}} = \frac{1}{F_{r_1}^2}\frac{\partial A}{\partial\mathbf{\Phi}_{j_{xy}}} + \frac{1}{F_{r_2}^2}\frac{\partial B}{\partial\mathbf{\Phi}_{j_{xy}}} = -\frac{\partial C}{\partial\mathbf{t}} \tag{33}$$

$$\frac{\partial C}{\partial F_\phi} = \frac{1}{F_{r_1}^2}\frac{\partial A}{\partial F_\phi} + \frac{1}{F_{r_2}^2}\frac{\partial B}{\partial F_\phi}$$

$$\Rightarrow\begin{cases}\frac{\partial A}{\partial F_\phi} = \left(\mathbf{p}-\mathbf{\Phi}_{j_{xy}}\right)^{\intercal}dM\left(\mathbf{p}-\mathbf{\Phi}_{j_{xy}}\right) \\ \\ \frac{\partial B}{\partial F_\phi} = \left(\mathbf{p}-\mathbf{\Phi}_{j_{xy}}\right)^{\intercal}dN\left(\mathbf{p}-\mathbf{\Phi}_{j_{xy}}\right)\end{cases}$$

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:

$$\begin{aligned}
g(Z_{\text{lp},i,k}, \mathbf{\Xi}_i, \mathbf{\Phi}_k) &= \bar{\mathbf{\Phi}}_k(T(\mathbf{\Xi}_i, Z_{\text{lp},i,k})) \\
&= T(\mathbf{\Xi}_i, Z_{\text{lp},i,k})^{\intercal}\overrightarrow{\mathbf{\Phi}_k}^{\circ}
\end{aligned} \tag{34}$$

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

$$E_{\text{feature},k} = \frac{1}{2}\sum_{i}^{n}\left\|\bar{\mathbf{\Phi}}_k(T(\mathbf{\Xi}_i, Z_{\text{lp},i,k})\right\|_{\Sigma_{\mathbf{\Phi}_k,i}}^2 \tag{35}$$

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

$$\begin{aligned}
\Sigma_{\mathbf{\Phi}_k,i}^{-1} &= \nabla h_{ik}\Sigma_z^{-1}\nabla h_{ik}^{\intercal} \\
\nabla h_{ik} &= \left.\frac{\partial h_{ik}}{\partial p_t}\right|_{{}^{\{i\}}p_{\mathbf{\Phi}_k},\mathbf{\Xi}_i,\mathbf{\Phi}_k} \\
&= \overrightarrow{\mathbf{\Phi}_{k_{12}}}^{\circ\,\intercal}
\end{aligned} \tag{36}$$

$\overrightarrow{\mathbf{\Phi}_{k_{12}}}^{\circ}$ 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:

$$\begin{aligned}
\frac{\partial E_{\text{feature},k}}{\partial \mathbf{t}} &= {\overrightarrow{\mathbf{\Phi}_{k_{12}}}^{\circ}}^{\intercal} \\
\frac{\partial E_{\text{feature},k}}{\partial \phi} &= {\overrightarrow{\mathbf{\Phi}_{k_{12}}}^{\circ}}^{\intercal} (dR)^{\{i\}}\mathbf{p} \\
\frac{\partial E_{\text{feature},k}}{\partial l_\alpha} &= T^{-1}(\mathbf{\Xi}_i, {}^{\{i\}}\mathbf{p})^{\intercal} \begin{bmatrix} -\sin l_\alpha \\ \cos l_\alpha \end{bmatrix} \\
\frac{\partial E_{\text{feature},k}}{\partial p} &= -1
\end{aligned} \tag{37}$$

## References

[1] J. M. Santos, D. Portugal, and R. P. Rocha, “An evaluation of 2d slam techniques available in robot operating system,” in *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2013, pp. 1–6.

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

[3] G. Grisetti, C. Stachniss, W. Burgard, *et al.*, “Improved techniques for grid mapping with rao-blackwellized particle filters,” *IEEE transactions on Robotics*, vol. 23, no. 1, p. 34, 2007.

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

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

[7] 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.

[8] 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.

[9] 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.

[10] 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.

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

[12] 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.

[13] 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.

[14] 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.