2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Invariant EKF based 2D Active SLAM with Exploration Task

Mengya Xu[1,2], Yang Song[1], Yongbo Chen[1], Shoudong Huang[1], and Qi Hao[2]

Abstract— **Right invariant extended Kalman filter (RIEKF) based simultaneous localization and mapping (SLAM) proposed recently has shown to be able to produce more consistent SLAM estimates as compared with traditional EKF based SLAM methods, including some improved EKF SLAM methods such as observability constrained-EKF (OC-EKF) SLAM. Latest results have demonstrated that its performance is very close to optimization based SLAM algorithms such as iSAM. In this paper, we propose to use RIEKF SLAM algorithm in active SLAM where both the predicted SLAM results for choosing control actions and the actual estimated SLAM results applying the selected control actions are computed using RIEKF algorithms. The advantages over traditional EKF based active SLAM are the more accurate and consistent predicted uncertainty estimates which result in robustness of the active SLAM algorithm. The advantages over optimization based active SLAM is the reduced computational cost. Simulation results are presented to validate the advantages of the proposed algorithm[3].**

I. INTRODUCTION

Active simultaneous localization and mapping (SLAM) is a decision making problem, in which the motion or path of the robot needs to be planned to achieve certain tasks and at the same time maintaining a good SLAM estimate. Motion planning is very important for the robot to explore an unknown environment. Good planning methods can help to improve the SLAM estimates (the estimation accuracy of map and localization), improve the map coverage, and reduce the exploration time. In this paper, we focus on active SLAM with exploration task, considering the performance in SLAM accuracy, coverage and the speed of exploration.

The study on active SLAM in unknown environments with exploration task was started two decades ago. Most of the earlier works built on extended Kalman filter (EKF) based SLAM algorithms. For example, Makarenko et al. [1] took into consideration the integrated tasks of coverage, SLAM accuracy and exploration speed. Each of them is evaluated using a utility function. The total utility of each candidate destination is calculated, and the one with the highest total utility will be the destination in the next step. Leung et al. [2] applied attractors to help guide the robot to the desired destination. There were three states, called $explore$, $improve\ map$ and $improve\ localization$ respectively, which were divided by a predefined threthold of uncertainty. According to the robot states, the attractor was placed in different places to guide the robot.

Usually, active SLAM problem is solved by minimizing a certain criterion in terms of the information gain, and the objective is to obtain more information. In [3], the greedy method was proposed, which aims to minimize the estimation error in the next step. Huang et al. [4] extended the greedy method into multi-step look-ahead method. A variant of nonlinear Model Predictive Control (MPC) was proposed to obtain a multi-step optimization in EKF based SLAM system within a finite time horizon.

Since 2006, it has been realized that EKF SLAM may result in inconsistent estimates [5], [6], [7], [8] due to linearization errors. In particular, the obtained covariance matrix in EKF SLAM is smaller than the actual estimate uncertainty, especially when the robot orientation error is large. Clearly, using the over confident covariance matrix obtained in EKF SLAM to guide the planning may lead to a poor robot trajectory in active SLAM.

Due to the inconsistency of EKF SLAM, optimization based SLAM methods become popular in the last decade and it has also been frequently used in active SLAM. Using all the information, the optimization based methods can achieve accurate SLAM estimate as well as consistent uncertainty estimate. However, the computational cost of optimization based SLAM is very high due to the large number of robot poses involved, especially when it is applied in active SLAM where the SLAM algorithm needs to be performed frequently to predict the performance of each candidate control action. Therefore, a mainstream idea for improving optimization based active SLAM is to increase its efficiency. For example, [14] investigated a sparsification method to reduce the computational complexity of decision making. Chen et al. [15] presented an efficient active SLAM approach to reduce the time for planning and estimation by utilizing submap joining, graph topology and convex optimization. In [17], a computationally efficient approach was proposed for belief space planning in high-dimensional state spaces via factor-graph propagation action tree.

Very recently, Right Invariant Extended Kalman Filter (RIEKF), designed on a specific Lie group structure, was found to have very good performance when it is applied in SLAM [9], [10], [11], [12], [13]. In particular, RIEKF SLAM can produce SLAM results with much improved consistency as compared with some improved EKF SLAM [10], such as observability constrained EKF (OC-EKF) SLAM [6]. Some nice convergence properties of RIEKF SLAM have been proved in [12]. Its performance was shown to be comparable to iSAM [20] and close to full optimization based SLAM in

[1] Mengya Xu, Yang Song, Yongbo Chen, and Shoudong Huang are with Centre for Autonomous Systems, University of Technology Sydney, NSW 2008, Australia. Mengya.Xu@student.uts.edu.au

[2] Mengya Xu and Qi Hao are with Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. hao.q@sustc.edu.cn

[3] The MATLAB source codes are made available at https://github.com/MengyaXu/RIEKF-based-active-SLAM

many cases [13], [18].

Motivated by these recent research results, we propose to use RIEKF in active SLAM where a robot needs to explore an unknown environment. This paper is the first research work along this direction and considers 2D case and applies greedy method in the planning. Simulation results demonstrate that the RIEKF based active SLAM is more robust than traditional EKF based active SLAM, and is computationally more efficient than optimization based active SLAM with acceptable performance in accuracy.

The paper is organized as follows. Section II reviews the RIEKF SLAM algorithm and its key advantages. Section III presents the details of the RIEKF based active SLAM framework. Experimental results using different simulation scenarios are provided in Section IV to compare the proposed RIEKF based active SLAM with EKF based and two optimization based active SLAM strategies. Finally, Section V concludes the paper and presents some future work.

II. RIEKF BASED 2D SLAM

SLAM problem has a nontrivial Lie group structure, as described and studied in [9], [10], [11]. Based on this discovery, [9], [10], [13] have proved that for feature based SLAM, the linearized system of RIEKF approach can automatically and correctly capture the unobservable direction of SLAM, ensuring strong consistency properties. Right Invariant EKF frame work shown in Alg. 1 is designed on this specific Lie group structure. Different with EKF method, RIEKF does the linearization on groups[10], [12], [13]. In this section, we will first review the specific Lie group structure of 2D SLAM. And then based on this proposed structure, RIEKF based 2D SLAM algorithm is briefly reviewed.

In the considered 2D SLAM problem, the n-th step state with N features is

$$\chi_n = (\boldsymbol{R}(\theta_n), \boldsymbol{x}_n, \boldsymbol{p}_n^1, \cdots, \boldsymbol{p}_n^N), \tag{1}$$

where $\theta_n \in (-\pi, \pi]$, $\boldsymbol{x}_n \in \mathbb{R}^2$ and $\boldsymbol{p}_n^i \in \mathbb{R}^2$ $(i = 1, \cdots, N)$ are respectively the robot orientation, robot position, and the coordinate of the i-th feature, all described in the fixed world coordinate frame. The notation $\boldsymbol{R}(\theta)$ is the rotation matrix related to the orientation θ:

$$\boldsymbol{R}(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}.$$

And we denote $\mathbb{SO}(2)$ as the set of all 2D rotation matrices.

And the set of all such possible states is denoted as

$$\mathcal{G}(N) = \left\{ \left(\boldsymbol{R}, \boldsymbol{x}, \boldsymbol{p}^1, \cdots, \boldsymbol{p}^N\right) | \boldsymbol{R} \in \mathbb{SO}(2), \boldsymbol{x} \text{ and } \boldsymbol{p}^i \in \mathbb{R}^2 \right\}. \tag{2}$$

It is a Lie group with the group action

$$\chi_1 \odot \chi_2 = \left(\boldsymbol{R}_1\boldsymbol{R}_2, \boldsymbol{R}_1\boldsymbol{x}_2 + \boldsymbol{x}_1, \cdots, \boldsymbol{R}_1\boldsymbol{p}_2^N + \boldsymbol{p}_1^N\right), \tag{3}$$

for all $\chi_1, \chi_2 \in \mathcal{G}(N)$.

Denote $\mathfrak{g}(N)$ as the associated Lie algebra of $\mathcal{G}(N)$, which is isomorphic to $\mathbb{R}^{2N+3}$. And the exponential mapping exp from $\mathfrak{g}(N)$ to $\mathcal{G}(N)$ is defined by

$$\exp(\boldsymbol{\xi}) = \left(\boldsymbol{R}(\delta\theta), \boldsymbol{B}(\delta\theta)\boldsymbol{\delta x}, \boldsymbol{B}(\delta\theta)\boldsymbol{\delta p}^1, \cdots, \boldsymbol{B}(\delta\theta)\boldsymbol{\delta p}^N\right), \tag{4}$$

where

$$\boldsymbol{\xi}^T = (\delta\theta, \boldsymbol{\delta x}^T, \boldsymbol{\delta p}^{1^T}, \cdots, \boldsymbol{\delta p}^{N^T}),$$
$$\boldsymbol{B}(\delta\theta) = \begin{bmatrix} \frac{\sin(\delta\theta)}{\delta\theta} & -\frac{1-\cos(\delta\theta)}{\delta\theta} \\ \frac{1-\cos(\delta\theta)}{\delta\theta} & \frac{\sin(\delta\theta)}{\delta\theta} \end{bmatrix}. \tag{5}$$

Then we can define the stochastic model on the proposed Lie group by

$$\chi = \exp(\boldsymbol{\xi}) \odot \hat{\chi}, \tag{6}$$

where $\hat{\chi}$ represents the mean, and $\boldsymbol{\xi} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{P})$ is the error with covariance matrix $\boldsymbol{P}$. Then χ is said to be log-Gaussian, denoted by $\chi \sim \mathcal{N}_{log}(\hat{\chi}, \boldsymbol{P})$.

Algorithm 1 Right Invariant EKF frame work

```
  Input: $\hat{\chi}_n$, $\boldsymbol{P}_n$, $\boldsymbol{u}_n$, $\boldsymbol{z}_{n+1}$
  Output: $\hat{\chi}_{n+1}$, $\boldsymbol{P}_{n+1}$
  Propagation:
$\hat{\chi}_{n+1|n} \leftarrow \boldsymbol{f}(\hat{\chi}_n, \boldsymbol{u}_n, \boldsymbol{0})$, $\boldsymbol{P}_{n+1|n} \leftarrow \boldsymbol{F}_n\boldsymbol{P}_n\boldsymbol{F}_n^T + \boldsymbol{G}_n\boldsymbol{Q}_n\boldsymbol{G}_n^T$
  Update:
$\boldsymbol{S}_{n+1} \leftarrow \boldsymbol{H}_{n+1}\boldsymbol{P}_{n+1|n}\boldsymbol{H}_{n+1}^T + \boldsymbol{O}_{n+1}$
$\boldsymbol{K}_{n+1} \leftarrow \boldsymbol{P}_{n+1|n}\boldsymbol{H}_{n+1}^T\mathbf{S}_{n+1}^{-1}$
$\boldsymbol{y}_{n+1} \leftarrow \boldsymbol{z}_{n+1} - h_{n+1}(\hat{\chi}_{n+1|n}, \boldsymbol{0})$
$\hat{\chi}_{n+1} \leftarrow \exp(\boldsymbol{K}_{n+1}\boldsymbol{y}_{n+1}) \odot \hat{\chi}_{n+1|n}$
$\boldsymbol{P}_{n+1} \leftarrow (\boldsymbol{I} - \boldsymbol{K}_{n+1}\boldsymbol{H}_{n+1})\boldsymbol{P}_{n+1|n}$
```

Considering the specific 2D SLAM problem, the process model is

$$\chi_{n+1} = \boldsymbol{f}(\chi_n, \boldsymbol{u}_n, \boldsymbol{w}_n) = \begin{bmatrix} \boldsymbol{R}(\theta_n + \omega_n + w_n^\omega) \\ \boldsymbol{R}(\theta_n)(\boldsymbol{v}_n + \boldsymbol{w}_n^v)^T \\ \boldsymbol{p}_n^1 \\ \vdots \\ \boldsymbol{p}_n^N \end{bmatrix}, \tag{7}$$

where $\chi_n \sim \mathcal{N}_{log}(\hat{\chi}_{n|n}, \boldsymbol{P}_{n|n})$, $\boldsymbol{u}_n = [\omega_n, \boldsymbol{v}_n]$ is the control input, and $\boldsymbol{w}_n = [w_n^\omega, \boldsymbol{w}_n^v] \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_n)$ is the noise.

And suppose there are K features $\left\{\boldsymbol{p}_{n+1}^{l_1}, \cdots, \boldsymbol{p}_{n+1}^{l_K}\right\}$ to be observed at the $n+1$-th step, the observation model can be considered as

$$\begin{aligned} \boldsymbol{z}_{n+1} &= \boldsymbol{h}_{n+1}(\chi_{n+1}, \boldsymbol{\epsilon}_{n+1}) \\ &= \begin{bmatrix} \boldsymbol{h}_{n+1}^1(\boldsymbol{R}(\theta_{n+1})^T(\boldsymbol{p}_{n+1}^{l_1} - \boldsymbol{x}_{n+1})) \\ \vdots \\ \boldsymbol{h}_{n+1}^K(\boldsymbol{R}(\theta_{n+1})^T(\boldsymbol{p}_{n+1}^{l_K} - \boldsymbol{x}_{n+1})) \end{bmatrix} + \boldsymbol{\epsilon}_{n+1}, \end{aligned} \tag{8}$$

where $\boldsymbol{\epsilon}_{n+1} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{O}_{n+1})$ is the observation noise.

Due to different linearization process (compared to the traditional EKF), the Jacobians in Alg.1, $\boldsymbol{F}_n$ is the identity matrix $\boldsymbol{I}_{2N+3}$, $\boldsymbol{G}_n$, and $\boldsymbol{H}_{n+1}$, are given by (see [10])

$$\boldsymbol{G}_n = \begin{bmatrix} 1 & \boldsymbol{0}_{1,2} \\ -\boldsymbol{J}\hat{\boldsymbol{x}}_{n|n} & \boldsymbol{R}(\hat{\theta}_{n|n}) \\ -\boldsymbol{J}\hat{\boldsymbol{p}}_{n|n}^1 & \boldsymbol{0}_{2,2} \\ \vdots & \\ -\boldsymbol{J}\hat{\boldsymbol{p}}_{n|n}^N & \boldsymbol{0}_{2,2} \end{bmatrix}, \quad \boldsymbol{H}_{n+1} = \begin{bmatrix} \nabla\boldsymbol{h}_{n+1}^1\boldsymbol{H}_{n+1}^1 \\ \nabla\boldsymbol{h}_{n+1}^2\boldsymbol{H}_{n+1}^2 \\ \vdots \\ \nabla\boldsymbol{h}_{n+1}^K\boldsymbol{H}_{n+1}^K \end{bmatrix}, \tag{9}$$

where $\boldsymbol{J}$ represents the skew symmetric matrix:

$$\boldsymbol{J} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix},$$

$\nabla \boldsymbol{h}_{n+1}^{j}$ is the Jacobian of $\boldsymbol{h}_{n+1}^{j}$ computed at $\boldsymbol{R}(\hat{\theta}_{n|n-1})^{T}(\hat{\boldsymbol{p}}_{n|n-1}^{l_j} - \hat{\boldsymbol{x}}_{n|n-1})$, and

$$\boldsymbol{H}_{n+1}^{j} = \left[\mathbf{0}_{2,1} \;\; -\boldsymbol{R}(\hat{\theta}_{n+1|n})^{T} \;\; \mathbf{0}_{2,2j-2} \;\; \boldsymbol{R}(\hat{\theta}_{n+1|n})^{T} \;\; \mathbf{0}_{2,2K-2j}\right].$$

III. Active SLAM using RIEKF

In this section, we present our active SLAM algorithm based on RIEKF.

A. The active SLAM problem

For the active SLAM problem considered in this paper, given an unknown environment of which the size is fixed and known, we assume there are some point features randomly distributed in the environment, both the number and the location of the features are unknown. The robot starts from a fixed location in the environment. The objective is to plan the robot trajectory for a given time horizon, such that it can explore the environment as much as possible (observe as many features as possible), and estimate the observed features and the robot poses with acceptable accuracy.

B. Proposed active SLAM method

In this paper, the RIEKF SLAM algorithm is used to predict the uncertainty of the SLAM estimate after taking a potential control action, it is also used to perform the SLAM estimation after a control action is taken.

We use one-step look-ahead strategy to maximize the information that will be obtained in the next step. In the RIEKF based method, the information gained in terms of the SLAM estimate can be described by the resulting covariance matrix after the control action is taken.

Concretely, given $\hat{\boldsymbol{\chi}}_n$ and $\boldsymbol{P}_n$, we would like to select the control vector $\boldsymbol{u}_n$ such that a certain metric (e.g. the trace) of the covariance matrix of the next step is optimized. That is, we want trace$(\boldsymbol{P}_{n+1})$ to be as small as possible where $\boldsymbol{P}_{n+1}$ is obtained by Alg. 1.

Note that the observation data $\boldsymbol{z}_{n+1}$ is not available when the planning is performed, so we assume that no new feature will be observed and the estimation will not be updated after the observation (zero-innovation) [15]. That is,

$$\begin{aligned} \boldsymbol{z}_{n+1} - \boldsymbol{h}_{n+1}(\hat{\boldsymbol{\chi}}_{n+1|n}) &= 0, \\ \hat{\boldsymbol{\chi}}_{n+1} &= \hat{\boldsymbol{\chi}}_{n+1|n}. \end{aligned} \tag{10}$$

In the proposed method, in order to take into account the exploration task in the planning, we consider not only the SLAM uncertainty, but also the distance, d, between the predicted robot position, $\hat{\boldsymbol{x}}_{n+1|n}$, and the goal point (how to select the goal point is presented in Section III-C).

Thus, the objective function to be minimized is a weighted sum of trace$(\boldsymbol{P}_{n+1})$ and d

$$obj = w_p \text{trace}(\boldsymbol{P}_{n+1}) + w_d d, \tag{11}$$

where the relative weights w_p and w_d are dependent and can be adjusted continuously or as an abrupt mode switch.

C. Goal point selection

In each step of the planning, there is one goal point selected to be the next destination which is used in the objective function (11).

We first generate a list of exploration points based on the size of the environment called L_e. For example, the exploration points can be uniformly distributed in the environment. The number of exploration points depends on the sensor range of the robot.

The goal point is selected according to the current state, similar to [2]. There are three states in total, called $explore$, $improve\,localization$ and $improve\,map$.

The three states are described as follows.

- **Explore.** When the uncertainty is blow a threshold $lowerbound$, the state is transformed to $explore$. The goal point is set to be the closest exploration point selected from the exploration point list L_e. We write the selected point as $\boldsymbol{p}_{explore}$. When the robot has approached the destination, $\boldsymbol{p}_{explore}$ is labeled to be explored and deleted from the exploration point list. Once all the exploration points are explored, the $explore$ state is no longer available and the $lowerbound$ becomes invalid. The task of the robot is to improve localization or improving map according to the $upperbound$.
- **Improve localization.** When the uncertainty is exceeding the threshold $upperbound$, the state is changed to $improve\,localization$ and we want the robot to re-visit a good feature, $\boldsymbol{p}_{good}$, whose uncertainty is low enough. To keep the re-visiting efficiency, $\boldsymbol{p}_{good}$ should not be too far away from the robot. Thus, the goal point is set to be the feature with the lowest uncertainty within a predetermined distance from the robot.
- **Improve map.** Otherwise, the state is changed to $improve\,map$ and a poor feature, $\boldsymbol{p}_{poor}$, is selected to be the goal point, of which the uncertainty is the highest within a given distance from the robot.

With the goal point selected as above, we can obtain the distance d in (11) as:

$$d = \begin{cases} d_{explore}, & \text{trace}(\boldsymbol{P}) < lowerbound \text{ and } L_e \neq \emptyset \\ d_{poor}, & lowerbound \leq \text{trace}(\boldsymbol{P}) < upperbound \\ & \text{ or } (\text{trace}(\boldsymbol{P}) < upperbound \text{ and } L_e = \emptyset) \\ d_{good}, & \text{trace}(\boldsymbol{P}) \geq upperbound, \end{cases} \tag{12}$$

where $d_{explore}$, d_{poor} and d_{good} are the distances from the estimated robot pose to $\boldsymbol{p}_{explore}$, $\boldsymbol{p}_{poor}$ and $\boldsymbol{p}_{good}$ respectively. And $\boldsymbol{P}$ is the covariance matrix.

The thresholds are constantly adjusted according to the number of the observed features and the current steps:

$$\begin{aligned} upperbound &= w_k k + w_n n, \\ lowerbound &= w_k k + w_n n - c, \end{aligned} \tag{13}$$

where w_k is the weight of the number of the observed features k, w_n is the weight of the current total step n, and c is a constant. As the values of the objective function are different in different algorithms, the values of w_k and w_n are adjusted according to the environment and the algorithms.

IV. Simulation

In this section, we evaluate the proposed RIEKF based active SLAM by comparing it with EKF based active SLAM and two optimization based active SLAM.

A. Observation model and simulation settings

The process model for the active SLAM is shown in Section II. And the specific observation map $\boldsymbol{h}^j_{n+1}$ for the j-th observed feature $\boldsymbol{p}^{l_j}$ at the $(n+1)$-th step is

$$\boldsymbol{h}^j_{n+1}(\boldsymbol{q}) = \begin{bmatrix} \sqrt{q_1^2+q_2^2} \\ \text{atan2}(q_2, q_1) \end{bmatrix}, \tag{14}$$

where $\boldsymbol{q} = [q_1, q_2]^T = \boldsymbol{R}(\theta_{n+1})^T(\boldsymbol{p}^{l_j}_{n+1} - \boldsymbol{x}_{n+1})$.

In our simulation, there are 50 features generated randomly in the range of 100 m × 100 m, and the total step of the robot is set to be 500. The initial robot pose is $[0, 0, 0]$, and the sensor range is 20 m. The covariance matrix of control noise $\boldsymbol{w}_n$ is $\text{diag}[(0.02\text{rad})^2,\ (0.03\text{m})^2, (0.03\text{m})^2]$. And the covariance matrix of observation noise $\boldsymbol{\epsilon}^j_n$ is set as $\text{diag}[(0.04\text{m})^2,\ (0.04\text{rad})^2]$.

The compared EKF based active SLAM is similar to that in [2]. It uses the same approach as RIEKF based active SLAM but replacing RIEKF with EKF. In the first compared optimization based method, to calculate the next control, the information matrix of the next step can be directly obtained and maximized. Thus, its objective function is set to be:

$$obj = w_p \log(\det(\boldsymbol{\Lambda}_{n+1})) + w_d d, \tag{15}$$

where $\boldsymbol{\Lambda}_{n+1}$ is the information matrix.

The dimension of information matrix is very large due to the many robot poses involved. To reduce the computational complexity, in the second compared optimization based method, we use the lower bound based optimization method proposed in [15] and [16]. Instead of processing the large information matrix, a lower uncertainty bound in terms of the log determinant of a weighted Laplacian matrix is calculated. We write the predicted lower uncertainty bound as $\mathcal{LB}_{n+1}$, then the objective function becomes:

$$obj = w_p \log(\det(\mathcal{LB}_{n+1})) + w_d d. \tag{16}$$

The detailed formula about $\mathcal{LB}_{n+1}$ can be found in [15] and [16].

We compare the performance of these algorithms in terms of coverage, accuracy and processing time. The coverage is compared by counting the number of the unexplored features. The accuracy is measured by calculating the maximal/average error between the estimated values and the ground truth, including the robot pose error and the feature position error. We record the processing time of both the decision making part and the SLAM part, to compare the speed of determining the next control and the speed of estimating the current state, respectively. For each algorithm, we perform the simulation several times and present a representative result.

B. Simulation results

1) Simulation with a given path: We first present the different SLAM results using a predetermined path based on the EKF, optimization and RIEKF. The predetermined path is set to be a circle of which the radius is 45m.

As Fig. 1a shows, the result of EKF SLAM is clearly inconsistent, as the actual feature positions of most of the features are out of the 99% confidence ellipses. The nonlinear least squares optimization based SLAM (NLS) and RIEKF based SLAM can both obtain good quality estimate as shown in Fig. 1b and Fig. 1c, respectively. However, because the path is predetermined, 9 features remain undetected.

Fig. 2 shows the robot pose error of using the predetermined path. Because of the inconsistency issue, the error of the EKF based method is very large. The RIEKF based method achieves much better result and the pose error is similar to that of the optimization based method.

Table I shows the pose and feature estimation error of using the different approaches. The estimation error of optimization based algorithm is the smallest among the three algorithms. 'Max error robot' and 'Ave error robot' are

TABLE I: Estimation error comparison

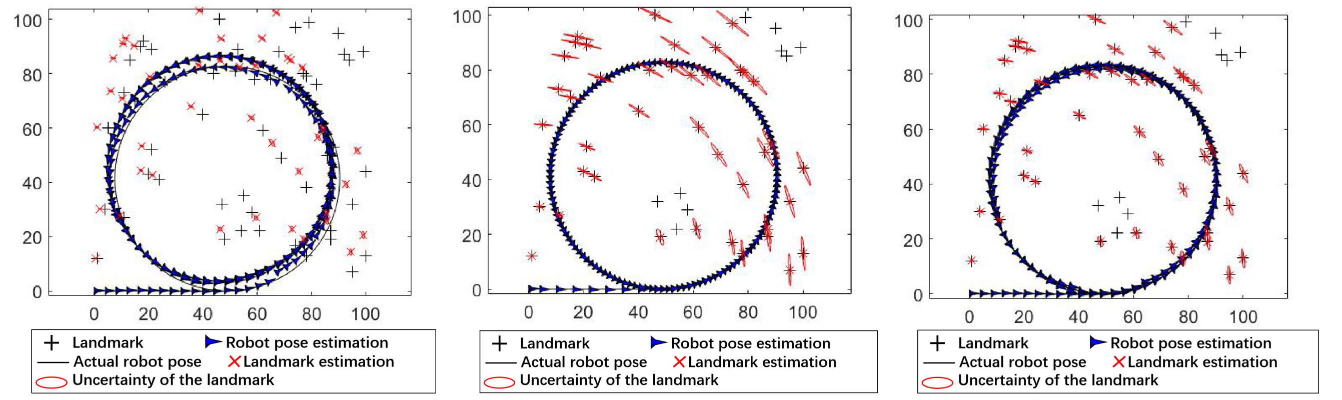
	Predetermined path		
	EKF	***NLS***	***RIEKF***
Max error robot (m)	7.8303	0.3450	2.4603
Ave error robot (m)	4.7247	0.1631	0.5094
Max error feature (m)	8.9188	0.3182	0.3548
Ave error feature (m)	5.9575	0.1777	0.2032

respectively the maximum and average errors of the robot pose. 'Max error feature' and 'Ave error feature' represent the maximum and average errors of the features, respectively. The errors are the distances between the ground truth and the estimation.

2) Comparison of the different active SLAM methods: **Coverage.** In this part, we show the coverage performance of the different methods under the same planning environment. The ground truth of the robot trajectory and the features and the results based on different methods (including the estimated poses and the estimated features, and the covariance ellipse of the features) are shown in Fig. 3, Fig. 4, Fig. 5, and Fig. 6.

The performance of the coverage task is much better when the planning method is used, compared with the results given by the predetermined path. Fig. 3 shows the result of the EKF based active SLAM algorithm. The accuracy is improved a lot by using the active SLAM algorithm. In the general cases, there are 3 to 8 features remaining undetected, also better than using the predetermined path. Fig. 4 and Fig. 5 show the results of using the traditional optimization method and the lower bound based optimization method respectively. There are 3 features left unseen in both of them. As shown in Fig. 6, when applying the RIEKF based active SLAM algorithm, all features can be detected. Usually, all features can be detected within 290 steps.

Accuracy. In this part, we compare the accuracy performance of the obtained active SLAM results using the



(a) EKF (b) NLS (c) RIEKF

Fig. 1: Result of using the predetermined path based on different SLAM methods in the environment with 50 randomly distributed landmarks/features.

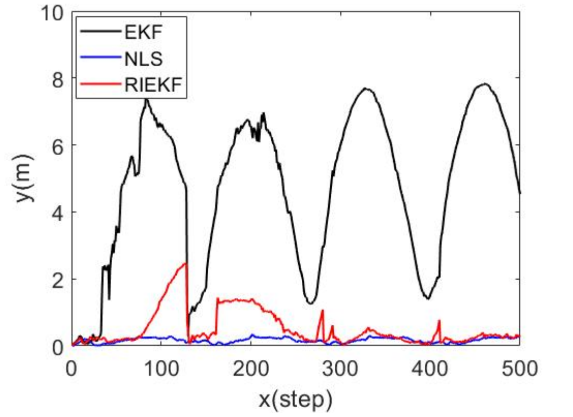


Fig. 2: The pose error of using the predetermined path based on EKF, the optimization algorithm and the RIEKF.

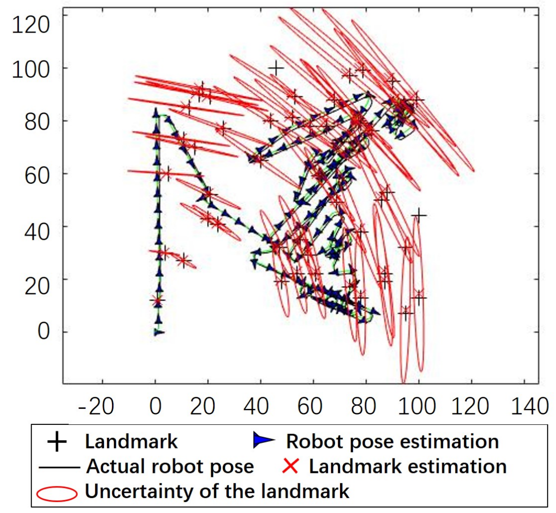


Fig. 4: Result of using optimization based active SLAM using information matrix as criteria (NLSI).

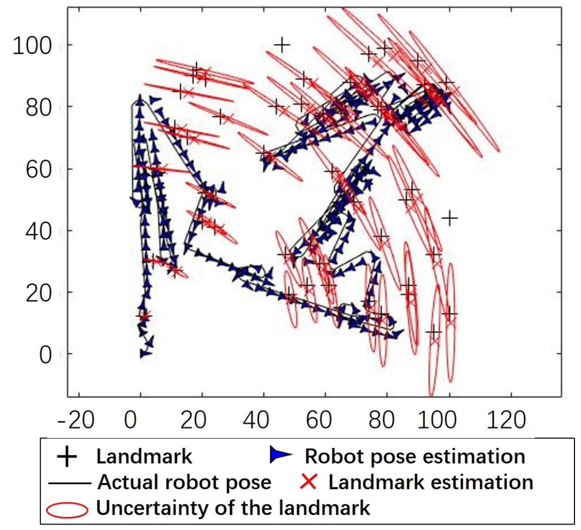


Fig. 3: Result of using EKF based active SLAM.

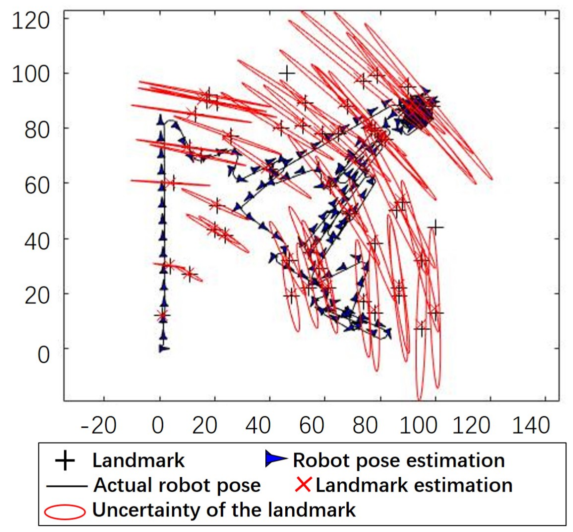


Fig. 5: Result of using optimization based active SLAM using lower bound as criteria (NLSlb).

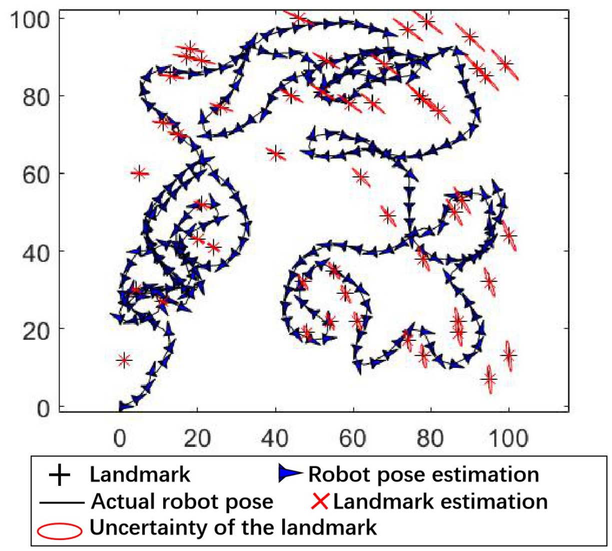


Fig. 6: Result of using RIEKF based active SLAM.

different methods. The results of the pose error based on different methods are shown in Fig. 7.

Two optimization algorithms achieve the best accuracy. The estimation error of the RIEKF based method is much smaller than the EKF based method in most cases. The average robot pose error and maximum robot pose error shown in Table II suggest that the RIEKF based method can obtain much smaller errors than EKF.

Besides the robot pose error, Table II also shows the maximum feature estimation error and the average feature estimation error in the last step. Similar to Section IV-B.1, the optimization algorithms have some advantages. Compared with the EKF based algorithm, the RIEKF based one can get smaller maximum error and average error.

By comparing Table I and Table II, we can see the improvement on the accuracy by using planning algorithms.

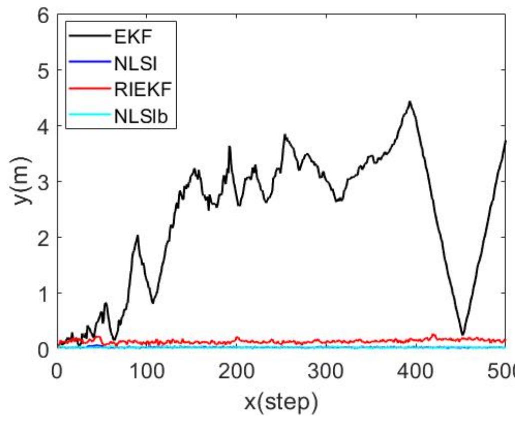


Fig. 7: The robot pose error of using different active SLAM methods. The results of the two optimization based methods (NLSI and NLSlb) are very similar.

Processing time. For efficiency, we firstly compare the processing time in the SLAM part. The result is as expected. The optimization based method takes much longer time than the EKF and RIEKF based method. EKF based method and RIEKF based method take almost the same time for the SLAM part. Then, our focus is on the processing time in the decision making part. We can see in Fig. 8, the EKF based method costs the least time, and the traditional optimization method NLSI costs much more time than others. The average value shows that the RIEKF based method costs about two times longer than the EKF based method. The lower bound based optimization method NLSlb is remarkable in reducing the computation time. Its speed in the first 300 steps is almost the same as the RIEKF based method. However, with the increase of the number of steps, its computation time increases gradually.

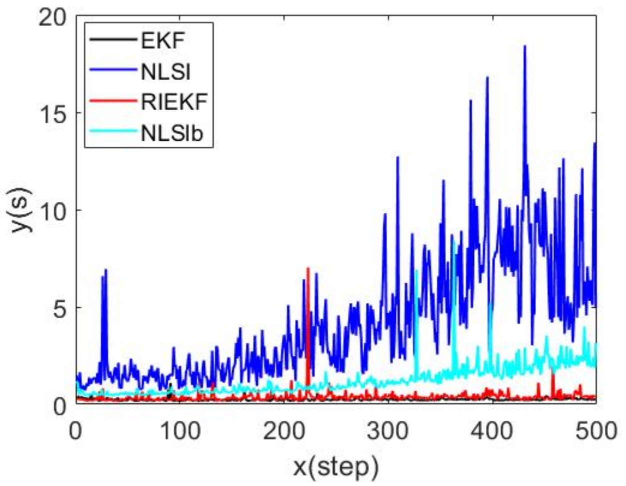


Fig. 8: The comparison of the decision making time.

TABLE II: Estimation error comparison

	Active SLAM			
	EKF	***NLSI***	***NLSlb***	***RIEKF***
Max error robot	4.4331	0.0592	0.0459	0.2634
Ave error robot	2.3257	0.0186	0.0161	0.1254
Max error feature	4.5722	1.8556	1.1315	0.1629
Ave error feature	2.8808	1.1821	0.7541	0.1175

The MATLAB source codes used in this paper are made available at https://github.com/MengyaXu/RIEKF-based-active-SLAM to help the readers to understand our framework more clearly. Some simulation results in other scenarios can be found there as well.

V. Conclusion and Future work

This paper proposes an RIEKF based active SLAM algorithm. Because of the improved consistency, the proposed algorithm shows superior performance in accuracy as compared with EKF based active SLAM. It is also demonstrated that the proposed algorithm has acceptable performance in accuracy and much lower computational cost as compared with optimization based active SLAM algorithms.

This research shows a great potential of using the RIEKF method in active SLAM. In the future, we will test our approach in practical 2D and 3D environments with obstacles. Some other planning strategies, such as multiple steps look ahead, will be investigated and evaluated. Future research work will also include other active SLAM problems where not only exploration task is needed.

References

[1] A. A. Makarenko, S. B. Williams, F. Bourgault, and H. F. Durrant-Whyte, "An experiment in integrated exploration," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 534-539, 2002.

[2] C. Leung, S. Huang, and G. Dissanayake, "Active SLAM using model predictive control and attractor based Exploration," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5026-5031, 2006.

[3] H. Feder, J. Leonard, and C. Smith, "Adaptive mobile robot navigation and mapping", International Journal of Robotics Research, vol. 18, no. 7, pp. 650-668, 1999.

[4] S. Huang, N.M. Kwok, G. Dissanayake, Q. P. Ha, and G. Fang, "Multi-Step Look-Ahead Trajectory Planning in SLAM: Possibility and Necessity," IEEE International Conference on Robotics and Automation (ICRA), pp. 1091-1096, 2005.

[5] S. Huang and G. Dissanayake, "Convergence and consistency analysis for extended Kalman filter based SLAM," IEEE Transactions on Robotics, vol. 23, no. 5, pp. 1036-1049, 2007.

[6] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "Analysis and improvement of the consistency of extended Kalman filter based SLAM," IEEE International Conference on Robotics and Automation (ICRA), pp. 473–479, 2008.

[7] J. A. Castellanos, R. Martinez-Cantin, J. Tardos, and J. Neira, "Robocentric map joining: Improving the consistency of EKF-SLAM," Robotics and Autonomous Systems, vol. 55, no. 1, pp. 21- 29, 2007.

[8] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "Observability-based rules for designing consistent EKF SLAM estimators," The International Journal of Robotics Research, vol. 29, no. 5, pp. 502–528, 2010.

[9] S. Bonnabel, "Symmetries in observer design: Review of some recent results and applications to EKF-based SLAM," in Robot Motion Control. London, U.K.: Springer, Jan. 2012, pp. 3–15.

[10] A. Barrau and S. Bonnabel, "An EKF-SLAM algorithm with consistency properties," CoRR, abs/1510.06263, 2015.

[11] A. Barrau and S. Bonnabel, "The invariant extended Kalman filter as a stable observer," IEEE Transactions Autonomous Control, vol. 62, no. 4, pp. 1797–1812, 2017.

[12] T. Zhang, K. Wu, J. Song, S. Huang, and G. Dissanayake, "Convergence and consistency analysis for a 3-D invariant-EKF SLAM, IEEE Robot," IEEE Autonomous Lettter, vol. 2, no. 2, pp. 733–740, 2017.

[13] M. Brossard, A. Barrau, and S. Bonnabel, "Exploiting symmetries to design EKFs with consistency properties for navigation and SLAM," IEEE Sensors Journal, vol. 19, no. 4, pp. 1572-1579, 2019.

[14] V. Indelman, "No correlations involved: decision making under uncertainty in a conservative sparse information space," IEEE Robotics and Automation Letters, vol. 1, no. 1, pp. 407-414, 2016.

[15] Y. Chen, S. Huang, R. Fitch, and J. Yu, "Efficient active SLAM based on submap joining, graph topology, and convex optimization," IEEE International Conference on Robotics and Automation (ICRA), 2018, pp. 5159-5166.

[16] Y. Chen, S. Huang, and R. Fitch, "Active SLAM for mobile robots with area coverage and obstacle avoidance," IEEE/ASME Transactions on Mechatronics, vol. 25, no. 3, pp. 1182-1192, 2020.

[17] K. Dmitry and V. Indelman, "General-purpose incremental covariance update and efficient belief space planning via a factor-graph propagation action tree," The International Journal of Robotics Research vol, 38, no. 14, pp. 1644-1673, 2019.

[18] Y. Zhang, T. Zhang and S. Huang, "Comparison of EKF based SLAM and optimization based SLAM algorithms," IEEE Conference on Industrial Electronics and Applications (ICIEA), pp. 1308-1313, 2018.

[19] K. Khosoussi, S. Huang, and G. Dissanayake, "Novel insights into the impact of graph structure on SLAM," in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2014, pp. 2707– 2714.

[20] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," The International Journal of Robotics Research, vol. 31, no. 2, pp. 216–235, 2012.