Comparison of EKF based SLAM and Optimization based SLAM Algorithms

Yanhao Zhang, Teng Zhang, Shoudong Huang
Center for Autonomous Systems, University of Technology Sydney, Sydney, Australia
Yanhao.Zhang@student.uts.edu.au, Teng.Zhang@student.uts.edu.au, Shoudong.Huang@uts.edu.au

***Abstract*—This paper compares the recent developed state-of-the-art extended Kalman filter (EKF) based simultaneous localization and mapping (SLAM) algorithm, namely, invariant EKF SLAM, with the nonlinear least squares optimization based SLAM algorithms. Simulations in 1D, 2D, and 3D are used to evaluate the invariant EKF SLAM algorithm. It is demonstrated that in most 2D/3D scenarios with practical noise levels, the accuracy of invariant EKF is very close to that of nonlinear least squares optimization based SLAM. In the simple 1D case, the Kalman filter results and the linear least squares results are exactly the same (for any noise levels) due to the linear motion model and linear observation model involved.**

***Index Terms*—EKF, optimization, SLAM, Lie group, performance.**

I. Introduction

SLAM problem [1] asks whether it is possible for a robot to build a map of an unknown environment and simultaneously work out its own location within the map, using information gathered from sensors mounted on the robot. Reliable solutions to SLAM underpin successful robot deployment in many application domains especially when an external location reference such as a global positioning system (GPS) is not available. In point feature based SLAM, the map is represented by a sparse set of point features, and the solution to SLAM is an estimate of the observed point features and the latest robot pose, together with the associated estimate uncertainty.

Both EKF [1] and least squares optimization [2] have been used extensively in SLAM research in the past. Earlier SLAM research has used EKF where the state vector contains the latest robot pose and the positions of the features observed. However, it has been shown that EKF SLAM could result in inconsistent estimate [3] [4], here inconsistency means that the estimated covariance from the algorithm can violate the theoretical achievable lower bounds [1] [5].

Optimization based SLAM uses a state vector containing all the robot poses and all the features observed. Because relinearization is performed during each iteration step, there is no inconsistency issue in optimization based SLAM and thus the quality of the estimate is higher than that of EKF SLAM. However, when the robot trajectory is very long, the dimension of the state vector in optimization based SLAM is very high.

Recently, in [6], a 2D invariant EKF SLAM algorithm was proposed and it was demonstrated to perform much better than traditional EKF SLAM. Furthermore, in [7], some basic convergence properties of the 3D invariant EKF SLAM algorithm were proved, without the requirement that the Jacobians were evaluated at the ground truth as in [5]. It is also pointed out that the invariant EKF SLAM satisfies two necessary conditions that a consistent SLAM algorithm should satisfy, namely, (i) the estimate (relative position and orientation) value is invariant to the initial robot pose, and (ii) the estimate value is invariant to the initial robot pose uncertainty. This new version of EKF SLAM appears to overcome the fundamental limitation of the traditional EKF SLAM algorithms.

Although optimization based SLAM algorithms are becoming popular recently due to the high quality performance and the efficiency of the modern sparse solvers [8] [9], they may not always be the best algorithm to use in practice due to the accumulated length of the state vector, especially for the scenarios when a robot continuously operates in an environment with fixed size. Thus it is interesting to compare the performance between the state-of-the-art invariant EKF SLAM with optimization based SLAM algorithm for different scenarios.

This paper focuses on the accuracy comparison between invariant EKF SLAM and optimization based SLAM. We started from the 1D linear case using Kalman filter (KF) and linear least squares (LLS), then we analyzed 2D/3D nonlinear cases using right invariant EKF (RI-EKF) method [7] and non-linear least squares (NLLS) method. We numerically confirmed that the KF and LLS results for 1D SLAM are always exactly the same no matter how big the noise level is and no matter what operation scenario is. For 2D/3D cases, in most practical scenarios with reasonable noise levels, the RI-EKF results are very close to that of the NLLS results.

This paper is organized as follows. Section II discusses some related work. In Section III, the motion model and observation model of SLAM problem in 1D/2D/3D are stated, and the RI-EKF SLAM algorithm and least squares optimization based SLAM algorithm are briefly reviewed. In Section IV, the RI-EKF SLAM and optimization based SLAM are compared using 1D/2D/3D simulations for different noise levels and different simulation environments. Finally, Section V concludes the paper and outlines the future work.

Throughout this paper, bold lower-case and upper-case letters are reserved for vectors and matrices, respectively. $\mathbb{R}$ represent the set of real numbers. $\mathbb{SO}(2)$, $\mathbb{SO}(3)$ are the special orthogonal groups. The square of Euclidean norm is denoted by $\|\cdot\|^2$. The square of weighted Euclidean norm of vector $\mathbf{e}$ with a positive definite matrix $\mathbf{W}$ is denoted by $\|\mathbf{e}\|^2_{\mathbf{W}} := \mathbf{e}^\intercal \mathbf{W}^{-1}\mathbf{e}$.

978-1-5386-3758-6/18/$31.00 ©2018 IEEE

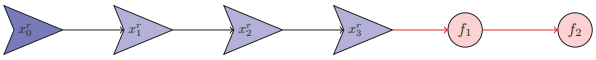


Fig. 1. An example of 1D SLAM problem. x_i^r $(i = 0, 1, 2, 3)$ represent robot poses and f_j $(j = 1, 2)$ represent feature/landmark positions.

II. Related Work

EKF SLAM has been the major technique when the SLAM problems were emerged in 1990s [1]. However, after a few years, a few researchers have pointed out a fundamental problem of EKF based SLAM due to linearization errors and the fact that Jacobians are evaluated at the estimated state values, [3] [4] [5] [10], that is, EKF SLAM algorithm can produce inconsistent estimates, especially when the robot orientation error is not small.

It was shown in [11] that the inconsistency in EKF SLAM is closely related to the partial observability of SLAM problem [12] [13]. This insight resulted in a number of EKF SLAM algorithms which significantly improve the SLAM consistency, such as first Jacobian [11] and observability-constrained EKF SLAM [14]. However, the potential inconsistency is still not completely avoided.

Recently, in [6], a 2D invariant EKF SLAM algorithm was proposed that exploits a Lie group based error representation. Some important convergence and consistency properties of invariant EKF SLAM were proved in [7]. These convergence and consistency properties naturally extend the proofs in [1] [5], without the unrealistic assumptions of linear model [1] or Jacobians are evaluated at the ground truth [5]. This new version of EKF SLAM has motivated researchers to reconsider the pros and cons of filter based and optimization based SLAM algorithms [15], while previous evaluation and comparisons had considered traditional EKF SLAM [16].

This paper aims to answer the question of under what conditions, the state-of-the-art invariant EKF based SLAM algorithm can produce results with similar accuracy as the optimal optimization based SLAM algorithms.

III. Invariant EKF SLAM and Optimization Based SLAM

A. Motion model and observation model in SLAM

In feature based SLAM, there are two kinds of information available. One is the odometry information described by the motion model, another is the observation information described by the observation model.

1) 1D SLAM: Consider the 1D SLAM problem as shown in Fig. 1. The robot moves in 1D and the robot position at time n is denoted as $x_n^r \in \mathbb{R}$. There are a number of features/landmarks such as $f_1,\ f_2$.

Assume the control input (e.g. odometry) at time n is $u_n \in \mathbb{R}$. Then the robot motion model can be written as

$$x_{n+1}^r = x_n^r + u_n + w_n \tag{1}$$

where $w_n \in \mathbb{R}$ is the process noise and is assumed to be zero mean Gaussian with variance σ_{od}^2.

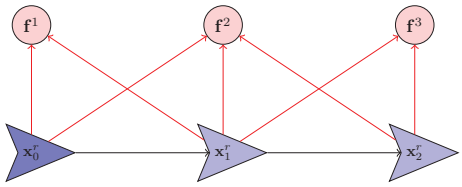


Fig. 2. An example of 2D/3D SLAM problem with three robot poses and three features/landmarks.

Assume the sensor on-board the robot can measure the relative position between the observed feature and the robot. Then the observation model (observing feature f_i at time step n) is

$$z_n^i = f_i - x_n^r + v_n^i \tag{2}$$

where f_i denotes the position of the stationary feature, $v_n^i \in \mathbb{R}$ is the observation noise and is assumed to be zero mean Gaussian wth variance σ_{ob}^2.

Note that in 1D SLAM, both the motion model and observation model are linear.

2) 2D/3D SLAM: For 2D/3D SLAM as shown in Fig. 2, the motion model can be described by

$$\begin{aligned}\mathbf{x}_{n+1}^r &= f^r(\mathbf{x}_n^r, \mathbf{u}_n, \epsilon_n) = \\ &(\mathbf{R}_n \exp(\omega_n + \epsilon_\omega), \mathbf{p}_n + \mathbf{R}_n(\nu_n + \epsilon_\nu))\end{aligned} \tag{3}$$

where the robot pose at time n is denoted by

$$\mathbf{x}_n^r = (\mathbf{R}_n, \mathbf{p}_n), \tag{4}$$

which includes the rotation matrix $\mathbf{R}_n$ and the robot position $\mathbf{p}_n$. $\mathbf{u}_n = (\omega_n, \nu_n)$ is the control input in which ω_n and ν_n represent the angular increment and linear translation between time step n and $n+1$, respectively, $\exp(\omega_n)$ is the rotation corresponding to ω_n and $\epsilon_n = (\epsilon_\omega, \epsilon_\nu) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Phi}_n)$ is control noise at the time step n [1].

We use $\mathbf{z}_n^i$ to denote the relative position observation made from pose $\mathbf{x}_n^r$ to feature $\mathbf{f}^i$. The observation model is given by

$$\mathbf{z}_n^i = \mathbf{R}_n^\intercal(\mathbf{f}^i - \mathbf{p}_n) + \xi_n^i, \tag{5}$$

where $\mathbf{f}^i$ is the feature position and $\xi_n^i \sim \mathcal{N}(\mathbf{0}, \mathbf{\Psi}_n^i)$ is the observation noise [2].

B. Invariant EKF SLAM

For 1D SLAM, the problem can be easily solved by KF. For 2D/3D SLAM, it has been shown that the RI-EKF outperforms other EKF SLAM algorithms [6] [7]. Now we briefly review the RI-EKF SLAM algorithm. Let the state in EKF SLAM be denoted as

$$\mathbf{X} = \left(\mathbf{R}, \mathbf{p}, \mathbf{f}^1, \cdots, \mathbf{f}^N\right). \tag{6}$$

[1] In 2D case, $\mathbf{R}_n \in \mathbb{SO}(2)$, $\mathbf{p}_n \in \mathbb{R}^2$, $\omega_n \in \mathbb{R}$, $v_n \in \mathbb{R}^2$, $\mathbf{u}_n \in \mathbb{R}^3$, $\exp(\omega_n) \in \mathbb{SO}(2)$, $\epsilon_n \in \mathbb{R}^3$, and $\mathbf{\Phi}_n \in \mathbb{R}^{3\times3}$. In 3D case, $\mathbf{R}_n \in \mathbb{SO}(3)$, $\mathbf{p}_n \in \mathbb{R}^3$, $\omega_n \in \mathbb{R}^3$, $v_n \in \mathbb{R}^3$, $\mathbf{u}_n \in \mathbb{R}^6$, $\exp(\omega_n) \in \mathbb{SO}(3)$, $\epsilon_n \in \mathbb{R}^6$, and $\mathbf{\Phi}_n \in \mathbb{R}^{6\times6}$.

[2] In 2D case, $\mathbf{z}_n^i, \mathbf{f}^i, \xi_n^i \in \mathbb{R}^2$, $\mathbf{\Psi}_n^i \in \mathbb{R}^{2\times2}$, while in 3D case, $\mathbf{z}_n^i, \mathbf{f}^i, \xi_n^i \in \mathbb{R}^3$, $\mathbf{\Psi}_n^i \in \mathbb{R}^{3\times3}$.

According to (3) and (4), the motion model of state $\mathbf{X}$ can be written as

$$\begin{aligned}&\mathbf{X}_{n+1} = f(\mathbf{X}_n, \mathbf{u}_n, \epsilon_n) = \\ &\left(\mathbf{R}_n \exp(\omega_n + \epsilon_\omega), \mathbf{p}_n + \mathbf{R}_n(\nu_n + \epsilon_\nu), \mathbf{f}_n^i, \cdots, \mathbf{f}_n^N\right).\end{aligned} \tag{7}$$

Considering that the robot observes different landmarks at each time step, we use O_{n+1} to denote the set of the landmarks observed at time step $n+1$. We can write the observation model at time step $n+1$ as follows

$$\mathbf{Z}_{n+1} = h_{n+1}(\mathbf{X}_{n+1}) + \xi_{n+1}, \tag{8}$$

where $\mathbf{Z}_{n+1}$ is a column vector obtained by stacking all entries (see (5)) $h^i(\mathbf{X}_{n+1}) = \mathbf{R}_{n+1}^\intercal(\mathbf{f}_{n+1}^i - \mathbf{p}_{n+1})$ for all $i \in \mathrm{O}_{n+1}$, $\xi_{n+1} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Psi}_{n+1})$ is the noise vector obtained by stacking all entries $\xi_{n+1}^i \sim \mathcal{N}(\mathbf{0}, \mathbf{\Psi}_{n+1}^i)$ for all $i \in \mathrm{O}_{n+1}$ and $\mathbf{\Psi}_{n+1} = \mathrm{diag}(\cdots, \mathbf{\Psi}_{n+1}^i, \cdots)$. With the motion model (7) and observation model (8), EKF formula could be applied to obtain a solution of 2D/3D SLAM.

The key difference between RI-EKF and conventional EKF is that a novel error state and retraction operation is defined (for more details see [6] [7]).

The retraction $\oplus$ of RI-EKF is chosen such that $\mathbf{X} = \hat{\mathbf{X}} \oplus \mathbf{e} := \exp(\mathbf{e})\hat{\mathbf{X}}$, where exp is the exponential mapping on the Lie group $\mathcal{G}(N)$ [7], $\mathbf{X} \in \mathcal{G}(N)$ is the actual pose and landmarks, $\hat{\mathbf{X}} \in \mathcal{G}(N)$ is the *mean* estimate and the uncertainty vector $\mathbf{e} = \begin{bmatrix} \mathbf{e}_\theta^\intercal & \mathbf{e}_p^\intercal & (\mathbf{e}^1)^\intercal & \cdots & (\mathbf{e}^N)^\intercal \end{bmatrix}^\intercal$ follows the Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{P})$ where $\mathbf{P}$ is the covariance matrix in RI-EKF SLAM. In 2D SLAM $\mathbf{e} \in \mathbf{R}^{2N+3}$, in 3D SLAM $\mathbf{e} \in \mathbf{R}^{3N+6}$.

C. Optimization based SLAM

The 1D SLAM problem can also be formulated as a LLS problem which has a closed-form solution, where the parameters to be estimated are all the poses (except pose x_0^r which is served as an anchor) and feature positions. For 2D/3D SLAM, the problem can be formulated as a NLLS problem [2] as below.

The parameters to be estimated include all the robot poses (except pose $\mathbf{x}_0^r$ which is assumed to be known) and all the feature positions

$$\mathbf{X} = \left(\mathbf{R}_1, \mathbf{p}_1, \cdots, \mathbf{R}_m, \mathbf{p}_m, \mathbf{f}^1, \cdots, \mathbf{f}^N\right), \tag{9}$$

The optimization problem is to minimize the negative log-likelihood function

$$\begin{aligned}F(\mathbf{X}) =& \sum_{i=0}^{m} \sum_{j \in O_i} \|\mathbf{z}_i^j - \mathbf{R}_i^\intercal(\mathbf{f}^j - \mathbf{p}_i)\|_{\mathbf{\Psi}_i^j}^2 \\ &+ \sum_{i=0}^{m-1} \|\mathbf{u}_i - \Delta\mathbf{x}_i^r\|_{\mathbf{\Phi}_i}^2.\end{aligned} \tag{10}$$

where $\Delta\mathbf{x}_i^r$ represents the relative pose between pose $\mathbf{x}_{i+1}^r$ and $\mathbf{x}_i^r$.

This NLLS problem could be solved by iterative methods such as Gauss-Newton. The covariance matrix of NLLS result can be obtained by inverting the information matrix given by $J^T\mathbf{\Sigma}^{-1}J$, where J is the Jacobian and $\mathbf{\Sigma}$ is obtained by stacking $\mathbf{\Psi}_i^j$ and $\mathbf{\Phi}_i$.

IV. Performance comparisons

For evaluating the performance of invariant EKF based SLAM and optimization based SLAM, we compare their estimation results in 1D/2D/3D SLAM [3]. We use KF and LLS in 1D SLAM, and RI-EKF and NLLS in 2D/3D SLAM. Monte Carlo simulations under different noise levels are performed assuming the noises of odometries and observations are from Gaussian distributions. The noise levels of odometries and observations are represented as σ_{od} and σ_{ob}.

A. Simulation setup

1) 1D SLAM simulation setup: In 1D SLAM, the robot moves along a straight line which allows only 1-DOF motion. Since it can be proved that the KF and LLS results are the same, we only confirm the results numerically using two simulations. In the first small simulation, the robot moves 20m, 10 landmarks are generated on the same line, and are not far from the robot positions. Noises of odometries and observations are randomly generated, and the noise level is $\sigma_{od} = 0.20$m, $\sigma_{ob} = 0.20$m. Only landmarks inside robot's sensor range (set as 3m) can be observed. The number of steps is 5, and the initial covariance of robot is set as zero. In the second simulation, other parameters are the same, but 50 landmarks are generated and the number of steps is 100 (2 loops), and the noise level is set as $\sigma_{od} = 0.07$m, $\sigma_{ob} = 0.07$m.

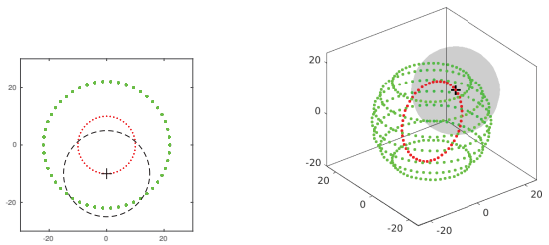
2) 2D SLAM simulation setup: In 2D SLAM, the robot moves along a circular trajectory which allows sufficient 3-DOFs motion. 50 landmarks are generated on a circle around the trajectory. Noises of odometries and observations are randomly generated, and robot sensor range is 15m and 2π FoV. The left figure of Fig. 3 shows the ground truth setup. Red dots represent the robot's position and the green dots represent the landmarks. The robot starts at position '+', and moves counterclockwise. In a single Monte Carlo run, the number of steps is 100 (2 loops). In order to evaluate the performance of RI-EKF and NLLS in 2D SLAM, we perform the Monte Carlo analysis with 50 runs for each noise level. The covariance matrices of odometries and observations are $\mathbf{\Phi}_n = \sigma_{od}^2\mathbf{I}_3$ and $\mathbf{\Psi}_n^i = \sigma_{ob}^2\mathbf{I}_2$.

3) 3D SLAM simulation setup: In 3D SLAM, the robot moves a trajectory (inside a cubic of 20m×20m×20m) which allows 6-DOFs motion. 350 landmarks are generated on circles around the trajectory. Robot sensor range is 15m, and the number of steps is 60 (more than one loop). The covariance matrices of odometries and observations are $\mathbf{\Phi}_n = \sigma_{od}^2\mathbf{I}_6$ and $\mathbf{\Psi}_n^i = \sigma_{ob}^2\mathbf{I}_3$, and initial robot pose covariance is set as zero. We perform a Monte Carlo simulation with 50 runs, and noise level (unit: radian and meter) is $\sigma_{od} = 0.07$, $\sigma_{ob} = 0.07$. The right figure of Fig. 3 shows the ground truth setup of 3D SLAM simulation.

B. Metrics used for comparison

Proper metrics are needed when comparing different estimation methods [17]. Firstly, it should be noticed that RI-EKF

[3]The Matlab code of the algorithms are available at: https://github.com/YanhaoZhang.



(a) Ground truth of 2D simulation (b) Ground truth of 3D simulation

Fig. 3. The ground truth of 2D/3D SLAM simulation. Red dots: robot trajectory, green dots: landmarks. Robot starts at position '+' and moves counterclockwise along each trajectory, respectively. Black circle/sphere: the sensor range (15m) of initial robot pose.

is an incremental estimation method, while NLLS is a batch estimation method. To be specific, EKF uses as much information as NLLS only after its final update process. Therefore, when evaluating the performance of the two methods, only the estimation of final state by EKF and the corresponding estimation by NLLS is comparable, namely the robot's final pose and all the landmarks. To compare the estimation of previous poses, we need to perform NLLS incrementally to make sure the two estimates are compared when they are using the same amount of information.

Root mean square error (RMSE) is a classical indicator used for evaluating the performance of different SLAM methods. The error $\mathbf{e}_n$ of current pose at time n is defined as [4]

$$\mathbf{e}_n^{\mathbf{R}} = \log\left(\hat{\mathbf{R}}_n \mathbf{R}_n^T\right), \quad \mathbf{e}_n^{\mathbf{p}} = \hat{\mathbf{p}}_n - \hat{\mathbf{R}}_n \mathbf{R}_n^T \mathbf{p}_n, \tag{11}$$

where $\mathbf{e}_n^{\mathbf{R}}$ represents the rotation error, $\mathbf{e}_n^{\mathbf{p}}$ represents the position error, $(\hat{\mathbf{R}}_n, \hat{\mathbf{p}}_n)$ represents the estimated orientation and position at time n, and $(\mathbf{R}_n, \mathbf{p}_n)$ represents the corresponding ground-truth pose. Then the estimation error squared (EES) can be expressed as $\varepsilon_n = \|\mathbf{e}_n\|^2$, and the RMSE at time n can be expressed as:

$$\text{RMSE}_n = \sqrt{\frac{1}{N_{mc}} \sum_{i=1}^{N_{mc}} \varepsilon_n^{(i)}}, \tag{12}$$

where $\varepsilon_n^{(i)}$ represents the EES computed at time n of the i-th Monte Carlo run. N_{mc} represents the total number of Monte Carlo runs.

Another indicator we used in this paper is *average* normalised estimation error squared (NEES), which shows the consistency of the estimator. As shown in [18], the NEES at time n is expressed as

$$\eta_n = \|\mathbf{e}_n\|^2_{\hat{\mathbf{\Sigma}}_n}, \tag{13}$$

where $\hat{\mathbf{\Sigma}}_n$ is the estimated covariance at time n. Then the *average* NEES of a Monte Carlo simulation with N_{mc} runs is

$$\bar{\eta}_n = \frac{1}{N_{mc}} \sum_{i=1}^{N_{mc}} \eta_n^{(i)}, \tag{14}$$

where $\eta_n^{(i)}$ represents the NEES computed at time n of the i-th Monte Carlo run. As it has been shown in [19] (pp. 234-238), under the hypothesis that the estimator is consistent and approximately linear-Gaussian, $N_{mc} \cdot \bar{\eta}_n$ is χ^2 distributed with $N_{mc} \cdot dim(\bar{\eta}_n)$ degree of freedom, where $dim(\bar{\eta}_n)$ represents the dimension of $\bar{\eta}_n$. For Monte Carlo simulations with $N_{mc} = 50$, the two-sided 95% probability regions are $[0.65, 1.43]$, $[1.48, 2.49]$, and $[2.36, 3.59]$ for $\bar{\eta}_n$ with dimension of 1, 2, and 3, respectively. As it has been described in [18], if $\bar{\eta}_n$ is significantly larger than the upper bound, then the estimator is overconfident.

[4]Let $\hat{\mathbf{T}}_n$ and $\mathbf{T}_n$ denote the estimated robot pose at time n and that of ground truth in $\mathbb{SE}(3)$, respectively, namely,

$$\hat{\mathbf{T}}_n = \begin{bmatrix} \hat{\mathbf{R}}_n & \hat{\mathbf{p}}_n \\ \mathbf{0}^T & 1 \end{bmatrix}, \quad \mathbf{T}_n = \begin{bmatrix} \mathbf{R}_n & \mathbf{p}_n \\ \mathbf{0}^T & 1 \end{bmatrix}.$$

Then the error between $\hat{\mathbf{T}}_n$ and $\mathbf{T}_n$ can be calculated as:

$$\hat{\mathbf{T}}_n \mathbf{T}_n^{-1} = \begin{bmatrix} \hat{\mathbf{R}}_n & \hat{\mathbf{p}}_n \\ \mathbf{0}^T & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{R}_n^T & -\mathbf{R}_n^T \mathbf{p}_n \\ \mathbf{0}^T & 1 \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{R}}_n \mathbf{R}_n^T & \hat{\mathbf{p}}_n - \hat{\mathbf{R}}_n \mathbf{R}_n^T \mathbf{p}_n \\ \mathbf{0}^T & 1 \end{bmatrix}.$$

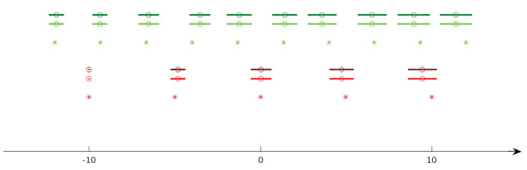


Fig. 4. The estimation result of the first small 1D SLAM simulation with 5 poses and 10 landmarks. Robot starts at position -10. Noise level: $\sigma_{od} = 0.20$, $\sigma_{ob} = 0.20$. Red dots: robot pose ground truth (represented by '$*$'), pose estimation by KF (dark color) and pose estimation by LLS (bright color). Green dots: landmarks of ground truth and estimation by KF and LLS. The small interval around each estimate represents its 2σ bound. Notice that for showing the estimation results clearly, we have put robot positions and landmarks in different lines, but they are actually on the same line.

C. Results and analysis for 1D case

In 1D SLAM, robot's motion model and observation model are linear functions, therefore the results of KF and LLS are exactly the same under the condition that they are using the same amount of information. Fig. 4 shows the estimation of the first simulation with 5 poses, where both the estimate and variances are the same for KF and LLS. Fig. 5 shows the estimation error with 2σ bound of the second simulation with 100 poses. The noise level is $\sigma_{od} = 0.07$ and $\sigma_{ob} = 0.07$. We can see that the estimation error and its 2σ bound of robot positions are exactly the same using KF and LLS in 1D SLAM.

D. Results and analysis for 2D case

For evaluating the performance of RI-EKF and NLLS in 2D SLAM, a Monte Carlo simulation with 50 runs is performed.

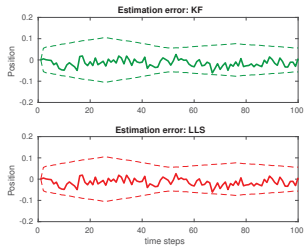


Fig. 5. The robot position estimation error with 2σ bound of the second 1D SLAM simulation (100 steps) using KF and LLS method, $\sigma_{od} = 0.07$, $\sigma_{ob} = 0.07$, unit: meter.

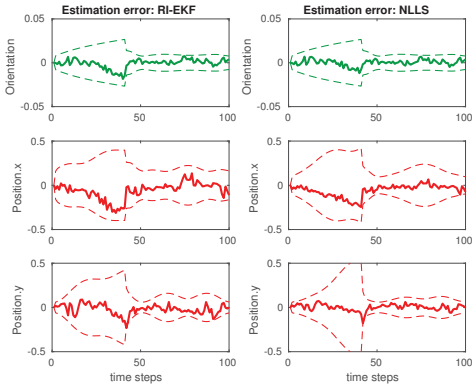


Fig. 6. The position and orientation estimation error with 2σ bound of a single 2D SLAM Monte Carlo run using RI-EKF and NLLS method, $\sigma_{od} = 0.07$, $\sigma_{ob} = 0.07$. Error is calculated using (11), unit: radian and meter.

For a single Monte Carlo run with noise level $\sigma_{od} = 0.07$ and $\sigma_{ob} = 0.07$, the robot position and orientation estimation error with 2σ bound is reported in Fig. 6, where both RI-EKF and NLLS achieve good estimation result. The error is calculated using (11). Further, for evaluating the general performance of RI-EKF and NLLS method at this noise level, we calculate the RMSE and *average* NEES of robot's orientation and position over the 50 Monte Carlo runs. The result is reported in Fig. 7, where blue lines represent the result of RI-EKF and red lines represent that of NLLS. From Fig. 6 and Fig. 7, we see that the estimation errors and the RMSE of position and orientation reduce obviously around the 41-th step where loop closure happens. Moreover, although the RMSE of NLLS is a bit lower than that of RI-EKF before the loop closure, they are very close when time step gets larger. Overall, position and orientation NEES of both methods do not exceed the upper bound of their 95% probability regions as described in Section IV-B.

Table I shows the average RMSE_n and average $\bar{\eta}_n$ of Monte Carlo simulations with different noise levels from 0.01 to 0.15. We can see that the averaged RMSE of both RI-EKF and NLLS increases slightly along with the increase of the noise level. An interesting point is that the averaged $\bar{\eta}_n$ of RI-EKF also increases along with the noise level, but that of NLLS changes little.

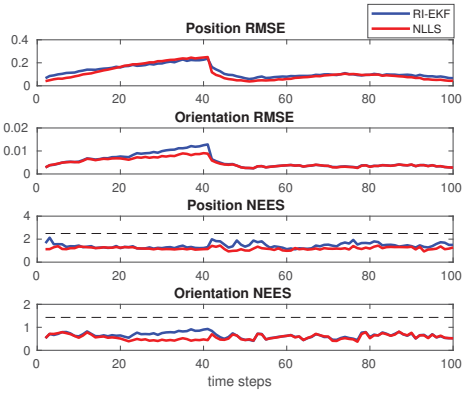


Fig. 7. RMSE and average NEES of 2D SLAM Monte Carlo simulation, $\sigma_{od} = 0.07$, $\sigma_{ob} = 0.07$. Blue lines: result by RI-EKF; red lines: result by NLLS; dashed lines: the upper bound of 95% confidence region.

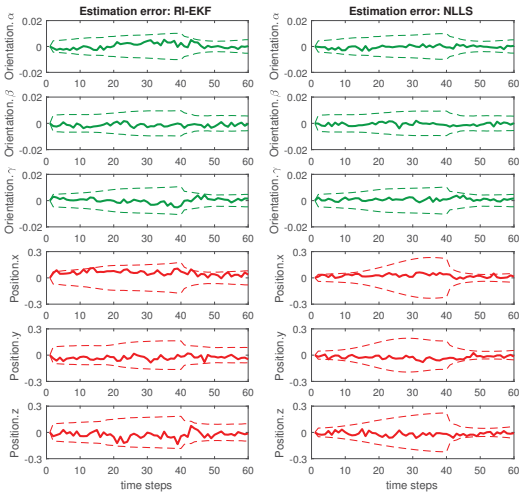


Fig. 8. The position and orientation estimation error with 3σ bound of a 3D SLAM Monte Carlo simulation, $\sigma_{od} = 0.07$, $\sigma_{ob} = 0.07$. Error is calculated by (11), unit: radian and meter.

E. Results and analysis for 3D case

Fig. 8 shows the estimation error with 3σ bound of a 3D simulation with noise level $\sigma_{od} = 0.07$ and $\sigma_{ob} = 0.07$. We see that in 3D SLAM, RI-EKF also achieves good performance w.r.t. estimation error. The general performance of RI-EKF and NLLS of the 3D Monte Carlo simulation is reported in Fig. 9, where it shows that both the average estimation error (RMSE) and the normalized average estimation error (NEES) of RI-EKF are close to that of NLLS in this 3D case. When looking at the RMSE, the result by RI-EKF and that by NLLS are similar after around the 41th step where loop closure happens.

V. Conclusion and Future Work

This paper evaluates the performance of invariant EKF based SLAM with respect to the optimization based SLAM. For 1D case, since all the models involved are linear, simple

TABLE I
PERFORMANCE OF 2D SLAM SIMULATION WITH DIFFERENT NOISE LEVELS

RI-EKF	$\sigma_{od}=0.01$ $\sigma_{ob}=0.01$	$\sigma_{od}=0.03$ $\sigma_{ob}=0.03$	$\sigma_{od}=0.05$ $\sigma_{ob}=0.05$	$\sigma_{od}=0.07$ $\sigma_{ob}=0.07$	$\sigma_{od}=0.09$ $\sigma_{ob}=0.09$	$\sigma_{od}=0.11$ $\sigma_{ob}=0.11$	$\sigma_{od}=0.13$ $\sigma_{ob}=0.13$	$\sigma_{od}=0.15$ $\sigma_{ob}=0.15$
RMSE of position	0.016	0.051	0.084	0.121	0.158	0.205	0.253	0.303
RMSE of orientation	0.001	0.002	0.004	0.005	0.007	0.010	0.013	0.016
NEES of position	1.202	1.274	1.331	1.432	1.574	1.743	2.111	2.213
NEES of orientation	0.602	0.607	0.620	0.652	0.681	0.886	0.992	1.126
NLLS	$\sigma_{od}=0.01$ $\sigma_{ob}=0.01$	$\sigma_{od}=0.03$ $\sigma_{ob}=0.03$	$\sigma_{od}=0.05$ $\sigma_{ob}=0.05$	$\sigma_{od}=0.07$ $\sigma_{ob}=0.07$	$\sigma_{od}=0.09$ $\sigma_{ob}=0.09$	$\sigma_{od}=0.11$ $\sigma_{ob}=0.11$	$\sigma_{od}=0.13$ $\sigma_{ob}=0.13$	$\sigma_{od}=0.15$ $\sigma_{ob}=0.15$
RMSE of position	0.015	0.047	0.076	0.108	0.144	0.172	0.203	0.235
RMSE of orientation	0.001	0.002	0.003	0.005	0.006	0.008	0.009	0.010
NEES of position	1.171	1.239	1.146	1.193	1.174	1.240	1.210	1.240
NEES of orientation	0.580	0.573	0.551	0.567	0.533	0.574	0.581	0.574

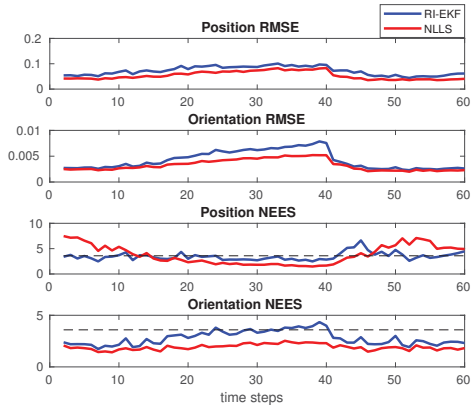


Fig. 9. RMSE and average NEES of a 3D SLAM Monte Carlo simulation, $\sigma_{od}=0.07$, $\sigma_{ob}=0.07$. Blue lines: result by RI-EKF; red lines: result by NLLS; dashed lines: the upper bound of 95% confidence region.

KF and LLS could be used to generate identical optimal results. For 2D/3D cases, since invariant EKF SLAM make uses of Lie group based error representation, the partial-observability does not have any impact on the EKF estimation consistency, resulting the EKF SLAM behaves similarly to optimization based SLAM in most of the practical scenarios. Although the invariant EKF can still result in poor quality eatimation when the noise level is set to be very high, the comparison in this paper makes researchers to rethink the value of EKF based SLAM algorithms in practice.

This paper has focused on the accuracy and consistency comparison between invariant EKF and optimization based SLAM. The computational complexity is not compared. Although the optimization based algorithms took much longer time than EKF based SLAM in our current implementation in MATLAB, we still cannot draw any conclusion on the computational cost since our code is not optimized yet. Future research is necessary for more thorough comparison especially in 3D, more comparisons using experimental datasets, and the improvement on the current invariant EKF SLAM algorithm.

REFERENCES

[1] G. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, 2001.

[2] F. Dellaert and M. Kaess, "Square root SAM: Simultaneous localization and mapping via square root information smoothing," *International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181-1203, 2006.

[3] S. J. Julier and J. K. Uhlmann, "A counter example to the theory of simultaneous localization and map building," *2001 IEEE International Conference on Robotics and Automation*, vol. 4, pp. 4238–4243, 2001.

[4] J. A. Castellanos, J. Neira, and J. D. Tardos, "Limits to the consistency of EKF-based SLAM," *Symposium on Intelligent Autonomous Vehicles*, Vol. 37, no. 8, pp. 716–721, 2004.

[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] A. Barrau and S. Bonnabel,"An EKF-SLAM algorithm with consistency properties," *CoRR*, arXiv:1510.06263, 2015.

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

[8] 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.

[9] G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," *2011 IEEE International Conference on Robotics and Automation*, pp. 3607–3613, 2011.

[10] T. Bailey, J. Nieto, J. Guivant, M. Stevens, and E. Nebot, "Consistency of the EKF-SLAM algorithm," In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3562–3568, 2006.

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

[12] J. Andrade-Cetto and A. Sanfeliu, "The effects of partial observability in SLAM," *2004 IEEE International Conference on Robotics and Automation*, vol. 1, pp. 397–402, 2004.

[13] K. W. Lee, W. S. Wijesoma, and J. I. Guzman, "On the observability and observability analysis of SLAM," *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3569–3574, 2006.

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

[15] K. Lenac, J. Cesic, I. Markovic, I. Cviic, and I. Petrovic, "Revival of filtering based SLAM? Exactly sparse delayed state filter on Lie groups," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1012–1018, 2017.

[16] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Real-time monocular SLAM: Why filter?," *2010 IEEE International Conference on Robotics and Automation*, pp. 2657–2664, 2010.

[17] S. Huang, Z. Wang, G. Dissanayake, and U. Frese, "Iterated D-SLAM map joining: Evaluating its performance in terms of consistency, accuracy and efficiency," *Autonomous Robots*, vol. 27, no. 4, pp. 409–429, 2009.

[18] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-Manifold Preintegration for Real-Time Visual-Inertial Odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1-21, 2017.

[19] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, "Estimation with Applications To Tracking and Navigation," John Wiley and Sons, 2001.