Implementation and Simulation of EKF-SLAM with Known and Unknown Correspondences in MATLAB

Sifan Jiang
sifanj@kth.se

Hongsheng Chang
changh@kth.se

***Abstract*—This project is to implement 2-D extended Kalman filter Simultaneous Localization and Mapping (EKF-SLAM) and simulate the algorithm in the environment of MATLAB. The implementation would include both of the situations of known landmarks correspondences and unknown correspondences.**

***Index Terms*—Data association, EKF-SLAM, MATLAB, SLAM**

I. Introduction

For autonomous vehicles exploring unknown environments, the ability to perform Simultaneous Localization and Mapping is essential. The Simultaneous Localization and Mapping solves the problem of the incrementally building of the pose of a robot and the position of the landmarks when the robot is placed at an unknown location in an unknown environment which in another word is that the robot has to build a consistent map of the environment and simultaneously determine its pose based on this map [1]. The common process of SLAM requires a recursive procedure of prediction of the robot or vehicle's state based on the interior sensor or input control to move the target and correction of the state and map based on the observation data. The prediction step is straight forward while the correction step with unknown landmark correspondences includes data association problem and determining if an observation corresponds to a new landmark.

So far, there has been three main paradigms of the solution to the SLAM problem, which are SLAM based on Kalman filter and Particle filter, and Graph-based SLAM [3]. And in this project, EKF-SLAM would be implemented, which is a solution of the family of Kalman filter. Among various algorithms developed for solving SLAM problem, the EKF-SLAM remains on of the most popular ones and has been used for several practical applications. EKF-SLAM is a estimate method used for solving nonlinear system. Extended Kalman filter uses the first terms of Taylor expansion for linearize the motion model and observation model for making sure the noises at next timestamp is are Gaussian distributions as well. The main resource of the algorithm of the EKF-SLAM is from the book of *Probabilistic Robotics* [5]. Based on the algorithm and the code from *Lab 1 of EL2320 Applied Estimation*, the EKF-SLAM in implemented and simulated in MATLAB.

Since the textbook of *Probabilistic Robotics* only gives a pseudocode for the algorithm of EKF-SLAM, some part of the code is not so natural and even contradictory when coming to the MATLAB code which could cause the wrong size of matrix or even negative size of matrix [5]. In this project, we implement an EKF-SLAM algorithm with unknown data association to avoid this situation from happening.

In all, the main contributions of this work are the following:

- Implement 2-D EKF-SLAM algorithm in MATLAB and simulate the algorithm with map and sensor data from *Lab files of EL2320 Applied Estimation*.
- Solve the data association problem of EKF-SLAM which is contradictory in the algorithm in the book of *Probabilistic Robotics* [5].

II. Simultaneous Localization and Mapping

In probabilistic form, the Simultaneous Localization and Mapping problem could be described in (1).

$$P(\mathbf{x}_k, \mathbf{m} \mid \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) \tag{1}$$

Equation (1) describe the joint posterior density of the robot poses and landmark positions at timestamp k. To calculate the result of (1), the Bayes filter is used. This computation requires a motion model and an observation model describing the effect of the control input and observation from the landmarks respectively [3].

The motion model for the robot can be described as (2).

$$P(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) \tag{2}$$

The observation model is to obtain the probability of making an observation $\mathbf{z}_k$ when the robot pose and landmark positions are known and can be described as (3).

$$P(\mathbf{z}_k \mid \mathbf{x}_k, \mathbf{m}) \tag{3}$$

Based on the motion model and the observation model, the SLAM algorithm is now implemented in Bayes filter, which is a recursive filter with prediction and correction steps:

- Prediction:

$$\begin{aligned} & P(\mathbf{x}_k, \mathbf{m} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0) \\ = & \int P(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) \\ & \times P(\mathbf{x}_{k-1}, \mathbf{m} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}, \mathbf{x}_0) \mathrm{d}\mathbf{x}_{k-1} \end{aligned} \tag{4}$$

- Correction:

$$\begin{aligned} & P(\mathbf{x}_k, \mathbf{m} \mid \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) \\ = & \frac{P(\mathbf{z}_k \mid \mathbf{x}_k, \mathbf{m}) P(\mathbf{x}_k, \mathbf{m} \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0)}{P(\mathbf{z}_k \mid \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})} \end{aligned} \tag{5}$$

The whole procedure of SLAM is illustrated in Fig. 1.

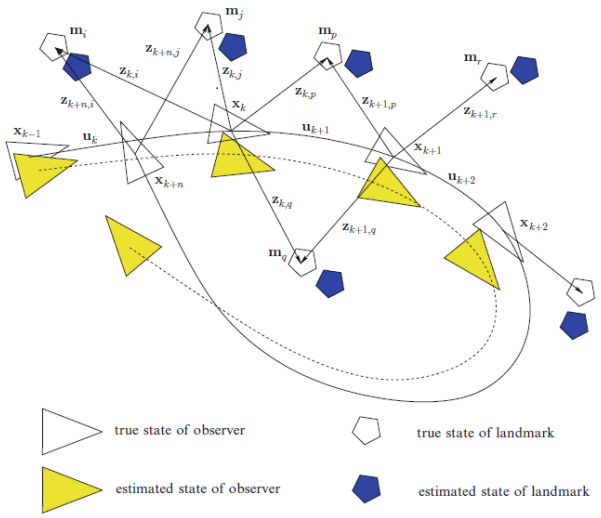

Fig. 1. SLAM Problem

III. EKF-SLAM Algorithms in *Probabilistic Robotics*

In EKF-SLAM, the motion model is changed from (2) to (6) and the observation model is changed from (3) to (7).

$$P(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) \Longleftrightarrow \mathbf{x}_k = \mathbf{g}(\mathbf{x}_{k-1}, \mathbf{u}_t) + \mathbf{w}_k \tag{6}$$

$$P(\mathbf{z}_k \mid \mathbf{x}_k, \mathbf{m}) \Longleftrightarrow \mathbf{z}(k) = \mathbf{h}(\mathbf{x}_k, \mathbf{m}) + \mathbf{v}_k \tag{7}$$

where $\mathbf{g}(\cdot)$ describes the vehicle kinematics and $\mathbf{h}(\cdot)$ describes the geometry of the observation. Also $\mathbf{w}_k$ and $\mathbf{v}_k$ are additive, zero mean uncorrelated Gaussian motion disturbances with covariance $\mathbf{Q}_k$ and observation errors with covariance $\mathbf{R}_k$ respectively [2].

The vector space of state in 2-D condition at timestamp k is $\boldsymbol{\mu}_k = [x\ y\ \theta\ m_{1,x}\ m_{1,y}\ \cdots\ m_{N,x}\ m_{N,y}]^T$ where x, y, and θ refer to x-coordinate, y-coordinate, and orientation (heading angle) of the robot and $m_{j,x}$, and $m_{j,y}$ are the position of j-th landmark. In abbreviation, $\boldsymbol{\mu}_k = [\mathbf{X}\ \mathbf{M}]^T$.

The covariance matrix of the state vector at timestamp k in abbreviation would be

$$\boldsymbol{\Sigma}_k = \begin{pmatrix} \boldsymbol{\Sigma}_{\mathbf{XX}} & \boldsymbol{\Sigma}_{\mathbf{XM}} \\ \boldsymbol{\Sigma}_{\mathbf{MX}} & \boldsymbol{\Sigma}_{\mathbf{MM}} \end{pmatrix}$$

The initial state and covariance are set to be $\boldsymbol{\mu}_0 = [0\ 0\ 0\ 0\ \cdots\ 0]^T$ and

$$\boldsymbol{\Sigma}_0 = \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \infty & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \infty \end{pmatrix}$$

When data association is unknown, we also need to initialize the number of landmarks to $N_0 = 0$. In both cases (known and unknown data association), a newly observed landmark need to be initialize (set position based on the observation and robot pose).

The EKF-SLAM Algorithm with known data association in [5] is shown in Alg. 1. The EKF-SLAM Algorithm with unknown data association in [5] is shown in Alg. 2. The algorithm is mainly follow the book, however, at some minor change is applied for convenient implementation in MATLAB. The lines 17 and 18 in known data association case and lines 21 and 22 in unknown data assocation case is added for reducing the memory consumption.

In both of the algorithms for known or unknown data association, the procedure before the first **For** loop is the prediction step and the procedure in the **For** loop is correction step. The algorithm in the textbook gives a detailed way to calculate the odometry of the robot which would be simplified in the implementation. Matrix F_x is used for matching the dimension difference between two matrix. In known correspondences case, since we assume the number of landmarks is also known, the main problem and difference from EKF localization is the initialization of newly observed landmark. To determine if the landmark observed has never been seen before is simple, a state vector for all the landmarks could be created to store such information which would be illustrated in following part. To initialize the landmark, the position of the landmark can be calculated by pose of the robot and the observation to such landmark which is a simple problem of geometry.

In the case of unknown data association, since the number and signature of the landmarks remains unknown, the main problem would also be determining if an observed landmark is a new landmark or not. In the textbook, the observed landmark would always be assumed as a new landmark at first. If the Mahalanobis distance of all existing landmarks in the map exceeds the threshold (value α), the landmark then would be considered as a new one. Since the signature of landmarks is unknown, for each newly observed landmarks, the

Algorithm 1 EKF_SLAM_Known_Correspondences

Input: μ_{t-1}, Σ_{t-1}, u_t, z_t, c_t

1: $F_x = \begin{pmatrix} 1 & 0 & 0 & \underbrace{0 \cdots 0}_{2N} \\ 0 & 1 & 0 & \\ 0 & 0 & 1 & \end{pmatrix}$

2: $\bar{\mu}_t = \mu_{t-1} + F_x^T \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\mu_{t-1,\theta} + \frac{v_t}{\omega_t}\sin(\mu_{t-1,\theta} + \omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\cos(\mu_{t-1,\theta} + \omega_t\Delta t) \\ \omega_t\Delta t \end{pmatrix}$

3: $G_t = I + F_x^T \begin{pmatrix} 0 & 0 & \frac{v_t}{\omega_t}\cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\cos(\mu_{t-1,\theta} + \omega_t\Delta t) \\ 0 & 0 & \frac{v_t}{\omega_t}\sin\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\sin(\mu_{t-1,\theta} + \omega_t\Delta t) \\ 0 & 0 & 0 \end{pmatrix} F_x$

4: $\bar{\Sigma}_t = G_t\Sigma_{t-1}G_t^T + F_x^T R_t F_x$

5: $Q_t = \begin{pmatrix} \sigma_r & 0 \\ 0 & \sigma_\phi \end{pmatrix}$

6: **for** All Observed Features $z_t^i = (r_t^i \ \phi_t^i)$ **do**

7: $j = c_t^i$ ▷ c_t^i is the known correspondence

8: **if** Landmark j never seen before **then**

9: $\begin{pmatrix} \bar{\mu}_{j,x} \\ \bar{\mu}_{j,y} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \end{pmatrix} + r_t^i \begin{pmatrix} \cos(\phi_t^i + \mu_{t,\theta}) \\ \sin(\phi_t^i + \mu_{t,\theta}) \end{pmatrix}$

10: **end if**

11: $\delta = \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{j,x} - \bar{\mu}_{t,x} \\ \bar{\mu}_{j,y} - \bar{\mu}_{t,y} \end{pmatrix}$

12: $q = \delta^T\delta$

13: $\hat{z}_t^i = \begin{pmatrix} \sqrt{q} \\ \text{atan2}(\delta_y, \delta_x) - \bar{\mu}_{t,\theta} \end{pmatrix}$

14: $F_{x,j} = \begin{pmatrix} 1 & 0 & 0 & \underbrace{0 \cdots 0}_{2j-2} & 0 & 0 & \underbrace{0 \cdots 0}_{2N-2j} \\ 0 & 1 & 0 & & 0 & 0 & \\ 0 & 0 & 1 & & 0 & 0 & \\ 0 & 0 & 0 & & 1 & 0 & \\ 0 & 0 & 0 & & 0 & 1 & \end{pmatrix}$

15: $H_t^i = \frac{1}{q}\begin{pmatrix} \sqrt{q}\delta_x & -\sqrt{q}\delta_y & 0 & -\sqrt{q}\delta_x & \sqrt{q}\delta_y \\ \delta_y & \delta_x & -1 & -\delta_y & -\delta_x \end{pmatrix} F_{x,j}$

16: $K_t^i = \bar{\Sigma}_t H_t^{iT}(H_t^i\bar{\Sigma}_t H_t^{iT} + Q_t)^{-1}$

17: $\bar{\mu}_t = \bar{\mu}_t + K_t^i(z_t^i - \hat{z}_t^i)$

18: $\bar{\Sigma}_t = (I - K_t^i H_t^i)\bar{\Sigma}_t$

19: **end for**

20: $\mu_t = \bar{\mu}_t$

21: $\Sigma_t = \bar{\Sigma}_t$

Output: μ_t, Σ_t

Algorithm 2 EKF_SLAM_Unknown_Correspondences

Input: μ_{t-1}, Σ_{t-1}, u_t, z_t, N_{t-1}

1: $F_x = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 \end{pmatrix}$

2: $\bar{\mu}_t = \mu_{t-1} + F_x^T \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\mu_{t-1,\theta} + \frac{v_t}{\omega_t}\sin(\mu_{t-1,\theta} + \omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\cos(\mu_{t-1,\theta} + \omega_t\Delta t) \\ \omega_t\Delta t \end{pmatrix}$

3: $G_t = I + F_x^T \begin{pmatrix} 0 & 0 & \frac{v_t}{\omega_t}\cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\cos(\mu_{t-1,\theta} + \omega_t\Delta t) \\ 0 & 0 & \frac{v_t}{\omega_t}\sin\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\sin(\mu_{t-1,\theta} + \omega_t\Delta t) \\ 0 & 0 & 0 \end{pmatrix} F_x$

4: $\bar{\Sigma}_t = G_t\Sigma_{t-1}G_t^T + F_x^T R_t F_x$

5: $Q_t = \begin{pmatrix} \sigma_r & 0 \\ 0 & \sigma_\phi \end{pmatrix}$

6: **for** All Observed Features $z_t^i = (r_t^i \ \phi_t^i)$ **do**

7: $\begin{pmatrix} \bar{\mu}_{N_t+1,x} \\ \bar{\mu}_{N_t+1,y} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \end{pmatrix} + r_t^i \begin{pmatrix} \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \\ \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \end{pmatrix}$

8: **for** $k = 1$ **to** $N_t + 1$ **do**

9: $\delta_k = \begin{pmatrix} \delta_{k,x} \\ \delta_{k,y} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{k,x} - \bar{\mu}_{t,x} \\ \bar{\mu}_{k,y} - \bar{\mu}_{t,y} \end{pmatrix}$

10: $q_k = \delta_k^T\delta_k$

11: $\hat{z}_t^k = \begin{pmatrix} \sqrt{q_k} \\ \text{atan2}(\delta_{k,y}, \delta_{k,x}) - \bar{\mu}_{t,\theta} \end{pmatrix}$

12: $F_{x,k} = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 0 & 1 & 0 \cdots 0 \end{pmatrix}$

13: $H_t^k = \frac{1}{q_k}\begin{pmatrix} \sqrt{q_k}\delta_{k,x} & -\sqrt{q_k}\delta_{k,y} & 0 & -\sqrt{q_k}\delta_{k,x} & \sqrt{q_k}\delta_{k,y} \\ \delta_{k,y} & \delta_{k,x} & -1 & -\delta_{k,y} & -\delta_{k,x} \end{pmatrix} F_{x,k}$

14: $\Psi_k = H_t^k\bar{\Sigma}_t H_t^{kT} + Q_t$

15: $\pi_k = (z_t^i - \hat{z}_t^k)^T\Psi_k^{-1}(z_t^i - \hat{z}_t^k)$

16: **end for**

17: $\pi_{N_t+1} = \alpha$

18: $j(i) = \text{argmin}_k \ \pi_k$

19: $N_t = \max\{N_t, j(i)\}$

20: $K_t^i = \bar{\Sigma}_t[H_t^{j(i)}]^T\Psi_{j(i)}^{-1}$

21: $\bar{\mu}_t = \bar{\mu}_t + K_t^i(z_t^i - \hat{z}_t^{j(i)})$

22: $\bar{\Sigma}_t = (I - K_t^i H_t^{j(i)})\bar{\Sigma}_t$

23: **end for**

24: $\mu_t = \bar{\mu}_t$

25: $\Sigma_t = \bar{\Sigma}_t$

Output: μ_t, Σ_t, N_t

signature of it would be determined sequentially. Also the matrix size in the unknown correspondences case is tricky.

IV. Implementation and Simulation of 2-D EKF-SLAM in MATLAB

A. Implementation

The main structure of our implementation is based on the programme from the *Lab files of EL2320 Applied Estimation* to give realize the reading and segmenting data from text files into different MATLAB variables. Also, we use some tool functions which were created by *Kai Arras* from *CAS-KTH* to plot the state of robot with covariance based ellipse to create gif videos. For the algorithm part of the code, in both cases it is divided into two parts: `predict.m` and `correct.m`.

As mentioned above, the motion model of the robot is simplied in the implementation, thus the prediction part would be changed into (8) and (9).

$$\bar{\mu}_t = \mu_{t-1} + F_x^T \begin{pmatrix} v_t\Delta t\cos\mu_{t-1,\theta} \\ v_t\Delta t\sin\mu_{t-1,\theta} \\ \omega_t\Delta t \end{pmatrix} \tag{8}$$

$$\begin{aligned} G_t &= I + F_x^T \begin{pmatrix} 0 & 0 & -v_t\Delta t\sin\mu_{t-1,\theta} \\ 0 & 0 & v_t\Delta t\cos\mu_{t-1,\theta} \\ 0 & 0 & 0 \end{pmatrix} F_x \\ &= I + F_x^T \begin{pmatrix} 0 & 0 & -\bar{\mu}_{t,y} \\ 0 & 0 & \bar{\mu}_{t,x} \\ 0 & 0 & 0 \end{pmatrix} F_x \end{aligned} \tag{9}$$

1) Known Data Association: As mentioned above, one of the problem of SLAM is to determine whether an observed landmark has never been seen before. In known data association case, a matrix `landmark_obs_flag` is used for recording the observation state of the landmarks.

2) Unknown Data Association: The main problem of the during the implementaion of EKF-SLAM with unknown data association is in the **For** loop of line 8 in Alg. 2. When $k = N_t + 1$, $2N_t - 2k = -2$ which is the width of zeros in matrix $F_{x,k}$ and should never be negative. And the reason why the size of the matrix is in such number

is to match the size of $\bar{\Sigma}_t$ obtained from the predict step which would be naturally be of size $(2N_t+3)\times(2N_t+3)$.

The solution in the implementation is to expand the size of $\bar{\Sigma}_t$ before the **For** loop of line 8 in Alg. 2, such that

$$\bar{\Sigma}_t = \begin{pmatrix} \bar{\Sigma}_t & \mathbf{0}^T \\ \mathbf{0} & \infty \end{pmatrix}$$

where

$$\mathbf{0} = \begin{pmatrix} 0 & \cdots & 0 \\ \underbrace{0 & \cdots & 0}_{2N_t+3} \end{pmatrix}$$
$$\infty = \begin{pmatrix} \infty & 0 \\ 0 & \infty \end{pmatrix}$$

The size of Σ at this moment is $[2(N_t+1)+3]\times[2(N_t+1)+3]$. If the observed landmark is not a new landmark, then

$$\bar{\Sigma}_t = \bar{\Sigma}_t\,(1:(2N_t+3), 1:(2N_t+3))$$

Futhermore, the number of zeros in matrix $F_{x,k}$ would be changed to $2(N_t+1)-2k$, thus the value would always be bigger or equal to zero.

B. Experimental Results

The simulation of EKF-SLAM is shown in videos and can be found under Video folder.

1) Known Data Association:

- `map_o3.txt` + `so_o3_ie.txt`:
  According to the description of *Lab 1 of EL2320 Applied Estimation*, the laser scanner has an accuracy of (1 cm, 1 degree), the odometry information has an un-modeled noise of approximately 1 cm and 1 degree per time step. So the motion distrubance and observation noise are set to be

$$R = \begin{pmatrix} 0.01^2 & 0 & 0 \\ 0 & 0.01^2 & 0 \\ 0 & 0 & (2\pi/360)^2 \end{pmatrix}$$
$$Q = \begin{pmatrix} 0.01^2 & 0 \\ 0 & (2\pi/360)^2 \end{pmatrix}$$

  The simulation result is shown in Fig. 2, 3, and 4, and video `map_o3+Video.gif`.
  As shown in the figures, the estimation of robot pose and landmark position is relatively close to the true pose of robot and landmarks. The error or the robot pose and average position of landmarks are small. Also, the covariance of the robot pose is small. Overall, the EKF-SLAM conducts well performance on this data sets.
- `map_pent_big_40.txt` + `so_pb_40_no.txt`:
  According to the description, the measurement noise is modeled with a standard deviation of 0.1 (m, rad) and the process noise is modeled with a standard deviation of 1 (m, m, rad). Also, the shape of the map should be noticed that the landmarks

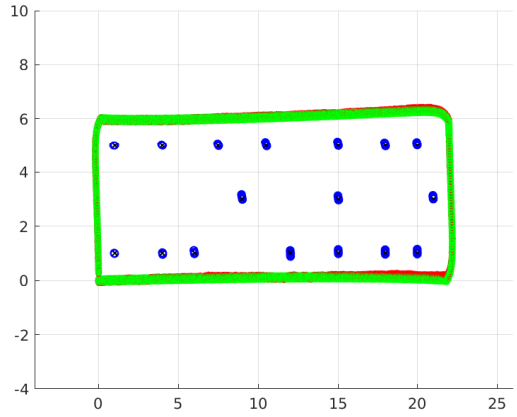

Fig. 2. Trace of Robot Pose and Map Location

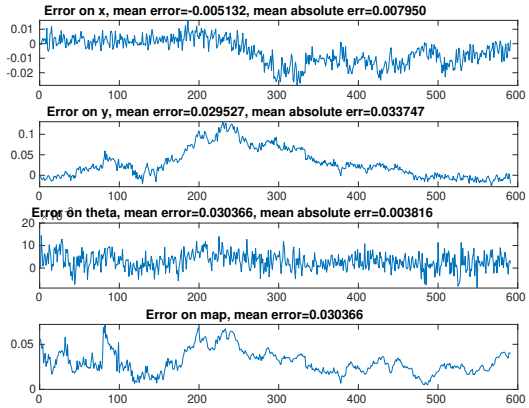

Fig. 3. Error of Robot Pose and Average Map Location

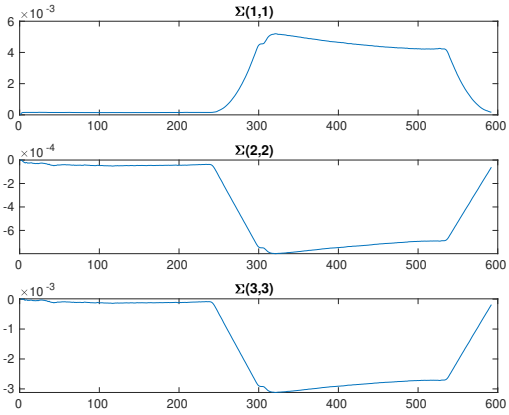

Fig. 4. Variance of Robot Pose

form a circle with is highly symmetric. So that the distrubance and noise are

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$
$$Q = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix}$$

The result is shown in Fig. 5, 6, and 7.

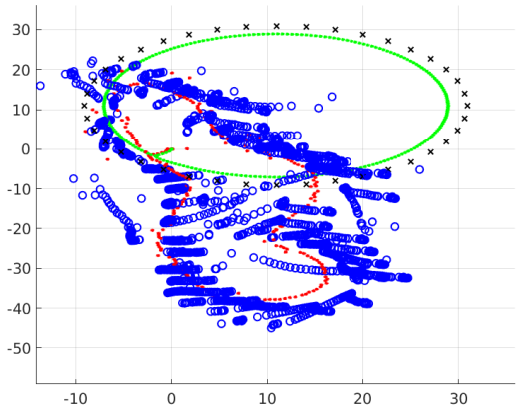

Fig. 5. Trace of Robot Pose and Map Location

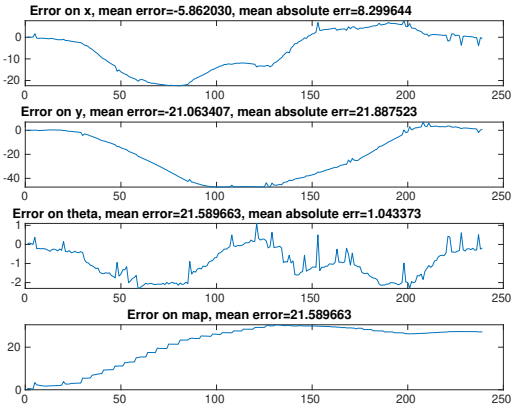

Fig. 6. Error of Robot Pose and Average Map Location

As shown in the figures, the EKF-SLAM has bad performance on this data sets with totally wrong estimation of robot poses and landmarks positions. The reason could be:

- The observation data of this data set could contain outliers. Unlike localization problems which can detect outliers based on the position of the landmarks to calculate Mahalanobis distances and use threshold to determine whether this observation is a outlier or not, the SLAM problem can not determine outlier since the observation may find a new landmark.
- Since the SLAM problem often includes highly non-linear motion for the robot, thus extended Kalman filter could give wrong and inconsistent prediction for the pose and position.

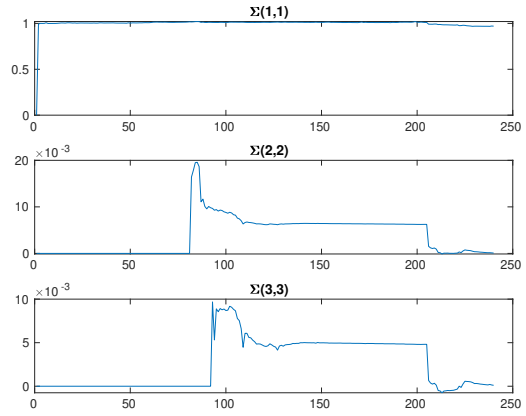

Fig. 7. Error of Robot Pose and Average Map Location

2) Unknown Data Association: Still, test on two data sets which are same to the case of known data association, and the setting of the motion disturbance and observation noise are same to the above case.

- `map_o3.txt + so_o3_ie.txt`:
  The result is illustrated in Fig. 8, 9, and 10.

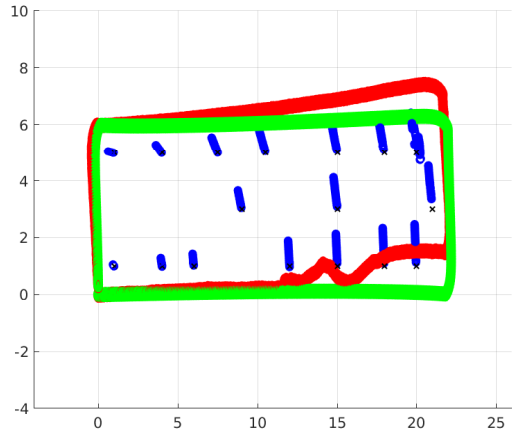

Fig. 8. Trace of Robot Pose and Map Location

Compared the figures to result of the case of known data association, we could see the unknown data association case could lead to larger error. That might be cause by the affect of landmarks that are not associated with the observation. However, they still need to be considered as a "weight". The far away of the landmark from the initial state of the

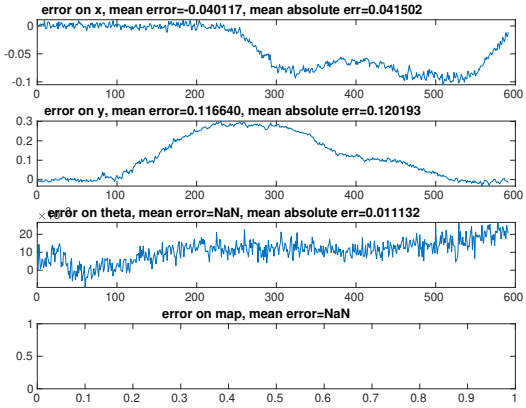

Fig. 9. Error of Robot Pose and Average Map Location

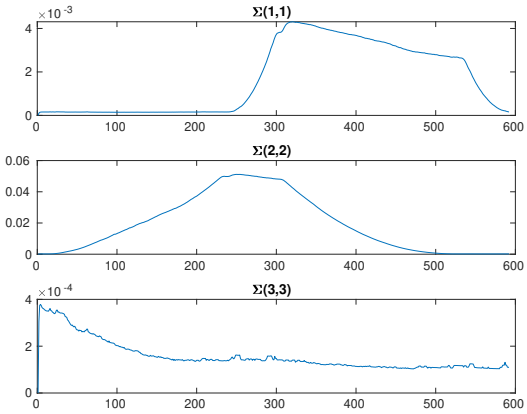

Fig. 10. Variance of Robot Pose

robot, the higher possibility it could deviate from its true position during the operating of the robot.

- `map_pent_big_40.txt` + `so_pb_40_no.txt`: The result is illustrated in Fig. 11, 12, and 13. As shown in the figures, the initial part of the EKF-SLAM process acceptable result (which could be seen much clearly on video). However, there appears a sharp changing in orientation of the robot which probably caused by the outlier observation.

V. Summary and Conclusions

Extended Kalman filter based Simultaneous Localization and Mapping is one of the most common solution to SLAM issue and have many applications. In this project, we use MATLAB to implement the algorithm of EKF-SLAM based on the textbook [5] and simulate the EKF-SLAM with the data sets from *Lab 1*. When implementing the 2-D EKF-SLAM algorithm with unknown data association, we resolve the problem that the algorithm on the textbook have some unclear definition of the size of the matrix.

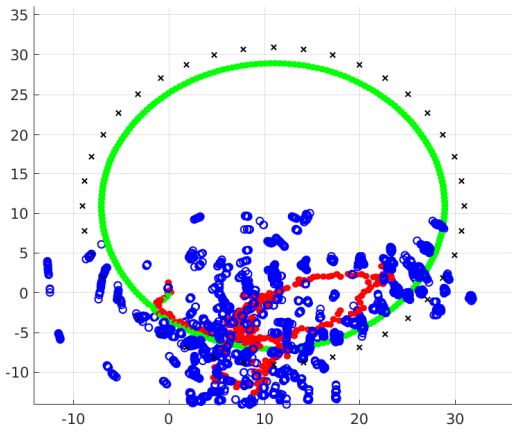

Fig. 11. Trace of Robot Pose and Map Location

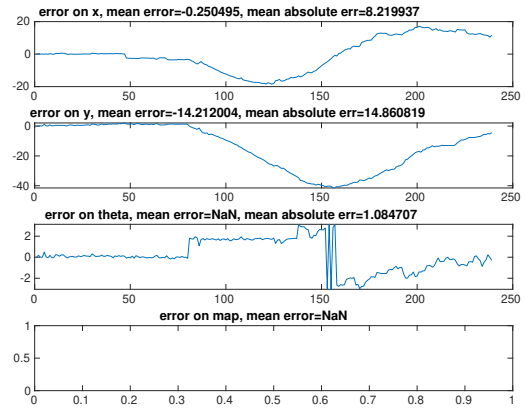

Fig. 12. Error of Robot Pose and Average Map Location

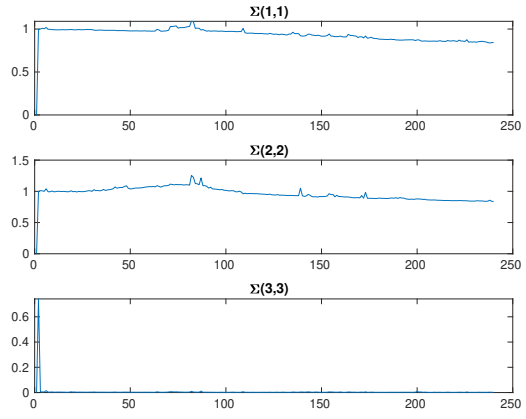

Fig. 13. Error of Robot Pose and Average Map Location

Through the experiment and simulation, we found that the EKF-SLAM can have well performance on relatively well measured data sets which have no or little outlier observation. However, when the data is not precise, the EKF-SLAM would play bad on the estimation of robot pose. Also, wrong data could lead to the wrong new landmarks which means the number of landmarks created in the whole process could larger than the real number of landmarks, and can not be effectively influenced by the change of threshold. Moreover, the EKF-SLAM with unknown data association would have accumulated error because of the inconsistency of the algorithm itself, so after robot has traveled a long journey and comes back to its initial point, it would be difficult for EKF-SLAM to recognize whether the landmark observed is new or not. Based on this point, the EKF-SLAM could have large difficulty on solving the the loop-closure problem.

REFERENCES

[1] G. Huang, A. Mourikis, and S. Roumeliotis. "Analysis and Improvement of the Consistency of Extended Kalman Filter Based SLAM," *IEEE*, May 2008.

[2] J. Castellanos, R. Martinez-Cantin, J. Tardos, and J. Neira, "Robocentric map joining: Improving the consistency of EKF-SLAM," *Robotics and Autonomous Systems*, vol. 55, pp. 21-29, 2007.

[3] H. Durrant-Whyte, and T. Bailey, "Simultaneous Localisation and Mapping (SLAM): Part I," Jun. 2006.

[4] T. Bailey, and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): Part II," *IEEE Robotics & Automation Magazine* vol. 13, no. 3, pp. 108-117, Aug. 2006.

[5] S. Thrun, W. Burgard, and D. Fox, "Probabilistic Robotics,"pp. 309–336, 2005.