Lab4 - EKF Map Based Localization

Probabilistic Robotics

Songyou Peng
psy920710@gmail.com

April 23rd , 2016

1 Introduction

Kalman filter is a recursion that tries to provide the best estimate of the state vector [1]. Extended Kalman filter (EKF) is the nonlinear version of Kalman filter which linearized about an estimate of the current mean and covariance [2]. For most of the cases of robotics localization in the real world, the prediction and measurement functions are nonlinear, so we need to use EKF.

In this report, we will discuss the implementation details of EKF map-based localization. Throughout the discussion, some observations about the EKF and problems we faced will also be mentioned.

2 Implementation

The EKF algorithm is composed of 3 main steps: prediction, data association, and updating. We will respectively discuss each step in details.

2.1 Prediction

First of all, we know the initial position of robotics in the world frame in advance. After getting the odometry, we want to estimate the next state of robotics based on the previous state and the odometry. There always exists some noise during the period of prediction. We can see the estimated state as a zero-mean Gaussian distribution, whose mean and covariance can be written as:

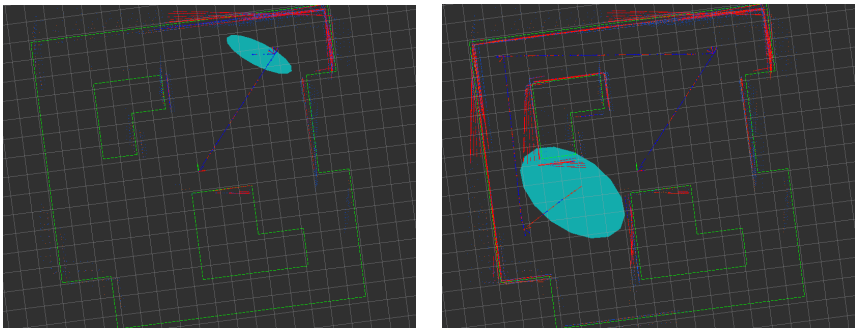
$$\hat{x}_k^- = f(\hat{x}_{k-1}^-, u_k, 0) \tag{1}$$

$$P_k^- = A_k P_{k-1} A_k^T + W_k Q_k W_k^T \tag{2}$$

where A_k and W_k are the Jacobian matrices with respect to previous state x_{k-1} and noise w_k respectively. The Q_k is the uncertainty matrix, which initially contains the σ of noise in its diagonal direction.

Here, the pose (state) of our robot with respect to the robot frame is compounded with the odometry with noise in each iteration, which makes f in (1)

a nonlinear model. After predicting in each iteration, the uncertainty keeps increasing. This can be seen in Fig. 1.



(a) Starting (b) Middle

Figure 1: Illustrations for the results of prediction (the size of green ellipse indicates the uncertainty)

2.2 Data Association

Predicting is not enough for sure so we need to utilize the measurement to localize our robot more accurately. Before that, given an observation (here the observation is the polar information of a line from the current pose), we need to know which map line best corresponds to this observation. This is data association.

First, we have to know the expected measurement model $z^- = h(\hat{x}_k^-)$, which can be written as:

$$h(\hat{x}_k^-) = \begin{cases} \rho_r = \rho_w - \sqrt{x^2+y^2}cos(arctan\frac{y}{x} - \psi_w), \\ \psi_r = psi_w - \theta \end{cases} \tag{3}$$

where $[x, y, \theta]$ is the predicted pose $\hat{x}_k^-$.

After acquiring the expected measurement, we can easily get the innovation v and its uncertainty S:

$$v_k = z - h(\hat{x}_k^-) \tag{4}$$

$$S_k = HP_k^-H^T + R \tag{5}$$

H is the Jocobian matrix of $h(\hat{x}_k^-)$.

It should be noted that, for the sake of getting the$H$, the map line is in the world frame instead of robot frame. When we compute the innovation in the formulation (4), the $h(\hat{x}_k^-)$ should be transformed to robot frame. This is one of the problems I encountered during the implementation.

And then we compute the Mahalanobis distance $vS^{-1}v^T$ for all the map lines. The reason of using Mahalanobis distance instead of Euclidean distance is because the predicted state is not just a point but represented by a Guassian distribution. We are not able to use Euclidean distance for calculating the distance between a point and a distribution. Finally, only if the smallest Mahalanobis distance among all the lines is smaller than the chi-square threshold, we trust this observation and use it for updating the state vector.

2.3 Updating

Once we ensure which observations are useful, we can update the state vector. As we have already got the "best" innovation v_k and uncertainty S_k in the data association part, we can easily get the key K_k, updated $\hat{x}_k$ and P_k:

$$K_k = P_k^- H_k^T S_k^{-1} \tag{6}$$

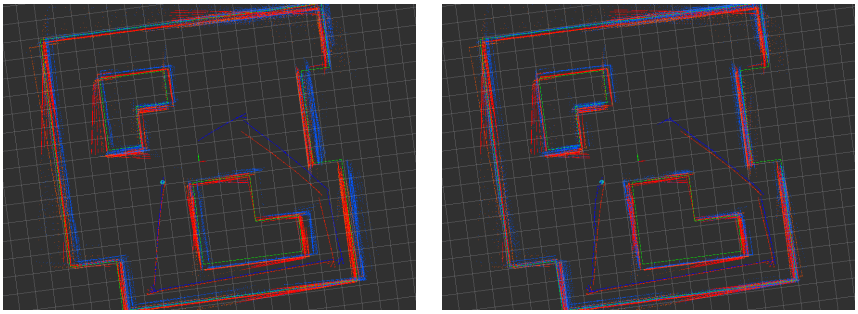
$$\hat{x}_k = \hat{x}_k^- + K_k v_k \tag{7}$$

$$P_k = (I - K_k H) P_k^- (I - K_k H)^T + K_k R K_k^T \tag{8}$$

The whole EKF process for one iteration finishes here. The new updated state vector and uncertainty matrix will be used as the input for the next iteration.

Here, we also tune the chi-square threshold. That the threshold is large means the algorithm accepts more observations and updates the state and uncertainty more fluently. If the threshold is small, only very a few accurate observation can be accepted and the state will not be updated often.

Because our observations are accurate, the results would be better if more observations can be used for updating. As we can see in Fig. 2, when the threshold is large, the blue lines, which are the expected measurements, are closer to the real map lines compared with the ones acquired from a smaller threshold.



(a) Chi-square = 0.1 (b) Chi-square = 0.3

Figure 2: Illustrations for the localization with different chi-square thresholds

We have also done the optional part by checking if the current observed line is longer than any map lines. If so, we know this observation is definitely wrong, then we simply discard this sensed line. After adding this constraint, the performance of robot localization slightly improves but not quite obvious. We think this is still due to the fact that the map is not ambiguous and our observations are quite good.

3 Conclusions

In this lab we have implemented the Extended Kalman filter to localize the robotics by iteratively predicting states, associating the observations with the map lines and finally updating states. We also compare the length of measured lines with lines in the given map in order to deal with some wrong data association situations.

Meanwhile, we find out that the EKF algorithm is more stable than the particle filter, but we still need to choose the chi-square threshold wisely.

References

[1] Choset, Howie. "Localization, Mapping, SLAM And The Kalman Filter According To George". www.cs.cmu.edu. N.p., 2016. Web. 23 Apr. 2016.

[2] Wikipedia, The Free Encyclopedia, s.v. "Extended Kalman filter," (accessed 23 April 2016, 2004), `https://en.wikipedia.org/wiki/Extended_Kalman_filter`