Investigation in state estimation for indoor robot navigation

1st Yihui Chen
National University of Singapore
e1010473@u.nus.edu

***Abstract*—This document is a model and instructions for LaTeX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

In recent years, mobile robot technology is constantly growing and developing, among which indoor mobile robots have been widely applied in industry and daily life, including AGV robots in factories, home sweeping robots and food delivery robots in the service industry. Navigations is one of the key tasks for mobile robots and can be summarized as three different questions: "where am I?", "where am I going?" and "how should I get there?" [1].

The "where am I?" problem is important since robots should first percept the unknown environment for further executions. Localization is aimed at solving it and can be divided into two main classes [2]. The relative methods [3] [4] rely on the odometry to figure out the robot state from the initial pose that is already known. However, they are sensitive to the initial value and limited by cumulative error.

Instead, absolute methods also combines the external observation from different sensors. Based on Bayes estimation theory, Kalman filter is one of the most used method for state estimation and proved to be robust. To tackle the strong nonlinearity in actual situations, Extended Kalman Filter was proposed and has been widely used in robot navigation [5], simultaneous localization and mapping (SLAM) [6] and multi-sensor fusion [7]. Some variants such as Error State Kalman Filter (ESKF) [8], Unscented Kalman Filter (UKF) [9] and Cubature Kalman Filter (CKF) [10] further improve the performance.

Particle filter is another direction to handle nonlinear problems. By sampling particles to approximate the probability density function and update them iteratively, the state of robot can be estimated. However, the updation of abundant particles can be computationally inefficient and affect the filter's real-time performance. Therefore, an approach that combines particle filter and particle swarm optimization algorithm [11] was proposed to achieve a robust and accurate estimation with less particles.

The rest of the paper will be organized as follows. Section II introduces the Bayesian filter and its application for state estimation in continuous cases. On the basis, Kalman Filter and Extended Kalman Filter are mentioned in section III and an example is tested. Section IV explains the theory of Particle Filter and gives a comparison with EKF. In section V, an EKF-based DWA planner is designed for real-time state estimation for robot navigation, and it is proved to be accurate and efficient. VI concludes the work.

II. Bayesian filter

A. Bayes theorem in discrete and continuous cases

Bayes theorem describes the probability of an event based on prior knowledge. Suppose that random variable X denote the event to be estimated while Y represent the observation result, x and y are their specific values respectively. Consdier Bayes rule in discrete cases and get

$$P(X = x \mid Y = y) = \frac{P(Y = y \mid X = x)P(X = x)}{P(Y = y)} \quad (1)$$

where $P(Y = y \mid X = x)$ is likelyhood that often describes the measurement accuracy of sensor, and $P(X = x)$ denotes the prior probability based on current knowledge. Using the law of total probability, $P(Y = y)$ on the denominator can be expanded as $P(Y = y) = \sum_{i=1}^{n} P(Y = y \mid X = x_i)P(X = x_i)$, which shows that its value is not related to the value of Y and thus it can be replaced by a constant η. Then Bayes theorem combines them and gives the posterior probability.

Now consider the Bayes fomula for continuous random variables

$$P(X < x \mid Y = y) = \frac{P(Y = y \mid X < x)P(X < x)}{P(Y = y)} \quad (2)$$

Since the fomula can not be directly used, the right side is written as a sum of discrete probabilities and turned into a limit form.

$$RHS = \sum_{u=-\infty}^{x} \frac{P(Y = y \mid X = u)P(X = u)}{P(Y = y)}$$
$$= \lim_{\varepsilon \to 0} \sum_{u=-\infty}^{x} \frac{P(y < Y < y + \varepsilon \mid X = u)P(u < X < u + \varepsilon)}{P(y < Y < y + \varepsilon)}$$

Let $f(\cdot)$ denotes the probability density function and apply mean value theorem, we derive

$$
\begin{aligned}
RHS &= \lim_{\varepsilon_1,\varepsilon_2,\varepsilon_3\to 0} \sum_{u=-\infty}^{x} \frac{(f_{Y|X}(\xi_1 \mid u))(f_X(\xi_2)) \cdot \varepsilon_1\varepsilon_2}{f_Y(\xi_3)\cdot \varepsilon_3} \\
&= \lim_{\varepsilon\to 0} \sum_{u=-\infty}^{x} \frac{f_{Y|X}(y \mid u) f_X(u)}{f_Y(y)} \cdot \varepsilon \\
&= \int_{-\infty}^{x} \frac{f_{Y|X}(y \mid x) f_X(x)}{f_Y(y)} dx
\end{aligned}
$$

where $\xi_1 \in (y, y+\varepsilon_1)$, $\xi_2 \in (u, u+\varepsilon_2)$ and $\xi_3 \in (y, y+\varepsilon_3)$. Also write the left side of Eq. 2 in a form of probability density function, i.e. $LHS = \int_{-\infty}^{x} f_{X|Y}(x \mid y)dx$, we finally get the Bayes formula for pdf in continuous cases.

$$
f_{X|Y}(x \mid y) = \frac{f_{Y|X}(y \mid x) f_X(x)}{f_Y(y)} \tag{3}
$$

Similarily, the denominator can be replace by a constant η, where $\eta = [\int_{-\infty}^{+\infty} f_{Y|X}(y \mid x) f_X(x)dx]^{-1}$.

B. Bayesian filtering algorithm

Suppose we have established the prediction and observation equation as

$$
\begin{aligned}
X_k &= F(X_{k-1}) + Q_k \\
Y_k &= H(X_k) + R_k
\end{aligned} \tag{4}
$$

where X_k, Y_k, Q_k, R_k are all random variables, and X_0, Q_k and R_k are mutually independent. In the prediction step, pdf of prior estimation $f_k^-(x)$ is firstly calculated.

$$
f_k^-(x) = \frac{d}{dx}(P(X_k < x)) = \frac{d}{dx}(\sum_{u=-\infty}^{x} P(X_k = u)) \tag{5}
$$

Expand $P(X_k = u)$ by total probability law and substitute the prediction function into it, we get

$$
\begin{aligned}
P(X_k = u) &= \sum_{v=-\infty}^{+\infty} P(X_k = u \mid X_{k-1} = v) P(X_{k-1} = v) \\
&= \sum_{v=-\infty}^{+\infty} P(X_k - F(X_{k-1}) = u - F(v) \mid X_{k-1} = v) P(X_{k-1} = v) \\
&= \sum_{v=-\infty}^{+\infty} P(Q_k = u - F(v)) P(X_{k-1} = v)
\end{aligned}
$$

Repeat what we do on RHS of Eq. 2, it becomes

$$
\begin{aligned}
P(X_k = u) &= \lim_{\varepsilon\to 0} \sum_{v=-\infty}^{+\infty} f_{Q_k}[u - F(v)] f_{k-1}(v) \cdot \varepsilon^2 \\
&= \lim_{\varepsilon\to 0} \int_{-\infty}^{+\infty} f_{Q_k}[u - F(v)] f_{k-1}(v) dv \cdot \varepsilon
\end{aligned}
$$

Substitute it into Eq. 5 and get the pdf of prior estimation

$$
\begin{aligned}
f_k^-(x) &= \frac{d}{dx}\left\{\sum_{-\infty}^{x} \lim_{\varepsilon\to 0} \int_{-\infty}^{+\infty} f_{Q_k}[u - F(v)] f_{k-1}(v) dv \cdot \varepsilon\right\} \\
&= \frac{d}{dx}\left\{\int_{-\infty}^{x} \int_{-\infty}^{+\infty} f_{Q_k}[u - F(v)] f_{k-1}(v) dv du)\right\} \\
&= \int_{-\infty}^{+\infty} f_{Q_k}[x - F(v)] f_{k-1}(v) dv
\end{aligned} \tag{6}
$$

After the observed value $Y_k = y_k$ is acquired by the sensor, the pdf of likelyhood is derived as

$$
\begin{aligned}
f_{Y_k|X_k}(y_k \mid x) &= \lim_{\varepsilon\to 0} \frac{P(y_k < Y_k < y_k + \varepsilon \mid X_k = x)}{\varepsilon} \\
&= \lim_{\varepsilon\to 0} \frac{P(y_k - H(x) < R_k < y_k - H(x) + \varepsilon)}{\varepsilon} \\
&= f_{R_k}[y_k - H(x)]
\end{aligned} \tag{7}
$$

Substitute Eq. 6 and Eq. 7 into Eq. 3, we get the pdf of posterior estimation:

$$
f_k^+(x) = \eta_k \cdot f_{R_k}[y_k - H(x)] \cdot f_k^-(x) \tag{8}
$$

where $\eta_k = (\int_{-\infty}^{+\infty} f_{R_k}[y_k - H(x)] \cdot f_k^-(x)dx)^{-1}$. With the prediction and update steps, the current state of robot can be estimated realtime by Bayes formula. The algorithm flow is given below.

Algorithm 1 Bayesian filter

INITIALIZE $f_0^+(x), Q, R$
for i=1,...,r **do**
  Predict Step $f_i^-(x) = \int_{-\infty}^{+\infty} f_{Q_i}[x - F(v)] f_{i-1}^+(v)dv$
  Update Step $f_i^+(x) = \eta_i \cdot f_{R_i}[y_i - H(x)] f_i^-(x)$
  Estimate State $\hat{x}_i^+ = \int_{-\infty}^{+\infty} x f_i^+(x)dx$
end for

III. Kalman filter

A. Kalman filter

Bayes filter estimates unknown probability density function recursively using a mathematical process model and external measurements, laying the foundation for realtime state estimation. However, it is usually hard to get analytical solutions when calculating improper integrals in the algorithm steps. Therefore, Kalman filter made some assumptions to simplify the process. Consider the prediction and observation equations in Eq. 4, suppose that both $F(\cdot)$ and $H(\cdot)$ are linear functions, and Q_k, R_k follow the normal distribution, i.e.

$$
\begin{aligned}
X_k &= FX_{k-1} + Q_k \\
Y_k &= HX_k + R_k \\
Q_k \sim N(0, Q), R_k &\sim N(0, R)
\end{aligned} \tag{9}
$$

Then if the random variable at the last time also follows a normal distribution, i.e. $X_{k-1}^+ \sim N(\mu_{k-1}^+, \sigma_{k-1}^+)$, we can

easily infer the pdf of prior probability is a normal distribution by convolution theorem.

$$\begin{aligned} &FX_{k-1}^{+} \sim N(F\mu_{k-1}^{+}, F^2\sigma_{k-1}^{+}), Q_k \sim N(0, Q) \\ &\Rightarrow X_k^{-} \sim N(F\mu_{k-1}^{+}, F^2\sigma_{k-1}^{+} + Q) \end{aligned} \tag{10}$$

Denote the mean and variance of X_k^- as μ_k^- and σ_k^-, the pdf of posterior probability $f_k^+(x)$ and estimated state $\hat{x}_k^+$ can be derived following the update rule. Then the random variable X_k^+ at the current time also follows the normal distribution, i.e. $X_k^+ \sim N(\mu_k^+, \sigma_k^+)$, where $\hat{x}_k^+ = \mu_k^+$ is the estimated state and they are given by

$$\begin{aligned} \mu_k^+ &= \mu_k^- + K(y_k - H\mu_k^-) \\ \sigma_k^+ &= (1 - KH)\sigma_k^- \\ K &= \frac{H\sigma_k^-}{H^2\sigma_k^- + R} \end{aligned} \tag{11}$$

Since the state variables are usually vectors, kalman filter algorithm can be generalized to a matrix form as below, where $\boldsymbol{F}$ and $\boldsymbol{H}$ are matrixes in prediction and observation equatons, and $\boldsymbol{\Sigma_k}$ is the covariance matrix.

Algorithm 2 Kalman filter

Prediction equation $\boldsymbol{X_k = FX_{k-1} + Q_k}$
Observation equation $\boldsymbol{Y_k = HX_k + R_k}$
$\boldsymbol{Q_k} \sim N(\boldsymbol{0}, \boldsymbol{Q})$, $\boldsymbol{R_k} \sim N(\boldsymbol{0}, \boldsymbol{R})$, $\boldsymbol{X_0} \sim N(\boldsymbol{\mu_0^+}, \boldsymbol{\Sigma_0^+})$
Initialize $\boldsymbol{\mu_0^+}, \boldsymbol{\Sigma_0^+}, \boldsymbol{Q}, \boldsymbol{R}$
for k=1,...,r **do**
  $\boldsymbol{\mu_k^- = F\mu_{k-1}^+}$
  $\boldsymbol{\Sigma_k^- = F\Sigma_{k-1}^+ F^T + Q}$ {prediction end}
  $\boldsymbol{K = \Sigma_k^- H^T (H\Sigma_k^- H^T + R)^{-1}}$ {the kalman gain}
  Obtain an observation value $\boldsymbol{y_k}$
  $\boldsymbol{\mu_k^+ = \mu_k^- + K(y_k - H\mu_k^-)^{-1}}$
  $\boldsymbol{\Sigma_k^+ = (I - KH)\Sigma_k^-}$ {update end}
  $\boldsymbol{\hat{x}_k^+ = \mu_k^+}$
end for

B. Extended Kalman filter

Prediction and observation functions are usually nonlinear in real cases, but they are simply supposed to be linear in Kalman filter, which leads to bad performance. Extended Kalman filter improve it by linearizing them by Taylor series. We still suppose that $X_{k-1}^+ \sim N(\mu_{k-1}^+, \sigma_{k-1}^+)$, and expand $F(X_{k-1}^+)$ by its first-order Taylor series about μ_{k-1}^+ as

$$\begin{aligned} F(X_{k-1}^+) &\approx F(\mu_{k-1}^+) + F'(\mu_{k-1}^+)(X_{k-1}^+ - \mu_{k-1}^+) \\ &\approx AX_{k-1}^+ + B \end{aligned} \tag{12}$$

where $A = F'(\mu_{k-1}^+)$ and $B = F(\mu_{k-1}^+) - F'(\mu_{k-1}^+)\mu_{k-1}^+$. Similarly, we follow the prediction step in Bayes filter algorithm and find X_k^- also follows the normal distribution, i.e. $X_k^- \sim N(A\mu_{k-1}^+ + B, A^2\sigma_{k-1}^+ + Q)$. Substitute A and B into it and it becomes

$$X_k^- \sim N(F(\mu_{k-1}^+), A^2\sigma_{k-1}^+ + Q) \tag{13}$$

Denote the mean and variance in Eq. 13 are μ_k^- and σ_k^- respectively, we calculate the pdf of posterior probability $f_k^+(x)$ by the update step in Bayes filter. Similarily, we expand the nonlinear observation function $H(X_k)$ by Taylor series about μ_k^- as

$$\begin{aligned} H(X_k) &\approx H(\mu_k^-) + H'(\mu_k^-)(X_k - \mu_k^-) \\ &\approx CX_k + D \end{aligned} \tag{14}$$

where $C = H'(\mu_k^-)$ and $D = H(\mu_k^-) - H'(\mu_k^-)\mu_k^-$. Following the update step we find the pdf of posterior probability is also in shape of normal distribution. Therefore, the random variable after update follows

$$\begin{aligned} X_k^+ &\sim N(\frac{R\mu_k^- + C\sigma_k^-(y_k - D)}{R + C^2\sigma_k^-}, (1 - \frac{C^2\sigma_k^-}{R + C^2\sigma_k^-})\sigma_k^-) \\ &\sim N(\mu_k^- + K(y_k - H(\mu_k^-)), (1 - KC)\sigma_k^-) \end{aligned} \tag{15}$$

where $K = \frac{C\sigma_k^-}{R + C^2\sigma_k^-}$. Then we get our estimated state $x_k^+ = \mu_k^+$. We can also generate the extended Kalman filter into matrix case. Different from the scalar case, here $\boldsymbol{Q}$, $\boldsymbol{R}$, $\boldsymbol{\Sigma_k}$ are all covariance matrixes, and $\boldsymbol{A}_{n\times n}$, $\boldsymbol{C}_{m\times n}$ (m observers and n state variables) need to be calculated in each loop:

$$\boldsymbol{A} = \begin{pmatrix} \frac{\partial F_1}{\partial X_{k-1}^1} & \frac{\partial F_1}{\partial X_{k-1}^2} & \cdots & \frac{\partial F_1}{\partial X_{k-1}^n} \\ \frac{\partial F_2}{\partial X_{k-1}^1} & \frac{\partial F_2}{\partial X_{k-1}^2} & \cdots & \frac{\partial F_2}{\partial X_{k-1}^n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial F_n}{\partial X_{k-1}^1} & \frac{\partial F_n}{\partial X_{k-1}^2} & \cdots & \frac{\partial F_n}{\partial X_{k-1}^n} \end{pmatrix}_{|X_{k-1}=\hat{x}_{k-1}^+=\mu_{k-1}^+} \tag{16}$$

$$\boldsymbol{C} = \begin{pmatrix} \frac{\partial H_1}{\partial X_k^1} & \frac{\partial H_1}{\partial X_k^2} & \cdots & \frac{\partial H_1}{\partial X_k^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial H_m}{\partial X_k^1} & \frac{\partial H_m}{\partial X_k^2} & \cdots & \frac{\partial H_m}{\partial X_k^n} \end{pmatrix}_{|X_k=\hat{x}_k^-=\mu_k^-} \tag{17}$$

The full flow of EKF algorithm is shown below.

C. Test results

To test the performance of these filters, we create a signal $x_1(t) = t^2$ and suppose the sensor noise follows the gaussian distribution, i.e. $\omega_1 \sim N(0, 0.1)$. First we use Kalman filter and establish prediction and observation equations. Since X_k can be expanded about X_{k-1} as $X_k = X_{k-1} + X_{k-1}'dt + \frac{1}{2}X_{k-1}''(dt)^2$, we choose $\begin{pmatrix} X_{k-1} & X_{k-1}' & X_{k-1}'' \end{pmatrix}^T$ and the two equations becomes

$$\begin{pmatrix} X_k \\ X_k' \\ X_k'' \end{pmatrix} = \begin{pmatrix} 1 & dt & \frac{1}{2}(dt)^2 \\ 0 & 1 & dt \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_{k-1} \\ X_{k-1}' \\ X_{k-1}'' \end{pmatrix} + Q_k \tag{18}$$

Algorithm 3 Extended Kalman filter

Prediction equation $\boldsymbol{X_k} = \boldsymbol{F}(\boldsymbol{X_{k-1}}) + \boldsymbol{Q_k}$
Observation equation $\boldsymbol{Y_k} = \boldsymbol{H}(\boldsymbol{X_k}) + \boldsymbol{R_k}$
$\boldsymbol{Q_k} \sim N(\boldsymbol{0}, \boldsymbol{Q})$, $\boldsymbol{R_k} \sim N(\boldsymbol{0}, \boldsymbol{R})$, $\boldsymbol{X_0} \sim N(\boldsymbol{\mu_0^+}, \boldsymbol{\Sigma_0^+})$
Initialize $\boldsymbol{\mu_0^+}, \boldsymbol{\Sigma_0^+}, \boldsymbol{Q}, \boldsymbol{R}$
for k=1,...,r **do**
  Calculate $\boldsymbol{A}$ using Eq. 16
  $\boldsymbol{\mu_k^-} = \boldsymbol{F}(\boldsymbol{\mu_{k-1}^+})$
  $\boldsymbol{\Sigma_k^-} = \boldsymbol{A}\boldsymbol{\Sigma_{k-1}^+}\boldsymbol{A}^T + \boldsymbol{Q}$ {prediction end}
  Calculate $\boldsymbol{C}$ using Eq. 17
  $\boldsymbol{K} = \boldsymbol{\Sigma_k^-}\boldsymbol{C}^T(\boldsymbol{C}\boldsymbol{\Sigma_k^-}\boldsymbol{C}^T + \boldsymbol{R})^{-1}$ {the kalman gain}
  Obtain an observation value $\boldsymbol{y_k}$
  $\boldsymbol{\mu_k^+} = \boldsymbol{\mu_k^-} + \boldsymbol{K}[\boldsymbol{y_k} - \boldsymbol{H}(\boldsymbol{\mu_k^-})]$
  $\boldsymbol{\Sigma_k^+} = (\boldsymbol{I} - \boldsymbol{K}\boldsymbol{C})\boldsymbol{\Sigma_k^-}$ {update end}
  $\hat{\boldsymbol{x}}_{\boldsymbol{k}}^+ = \boldsymbol{\mu_k^+}$
end for

$$Y_k = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} X_k \\ X_k{}' \\ X_k{}'' \end{pmatrix} + R_k \tag{19}$$

where $Q_k \sim N(0, Q)$, $R_k \sim N(0, R)$. The prediction for the higher derivative is considered more accurate and sensor noise is a little bit large, so here let $Q = diag\{1 \quad 0.01 \quad 0.0001\}$ and $R = 20$. Also, $\boldsymbol{\mu_0^+}$ and $\boldsymbol{\Sigma_0^+}$ are initialized as $\begin{pmatrix} 0.01 & 0 & 0 \end{pmatrix}^T$ and $diag\{0.01 \quad 0.01 \quad 0.0001\}$ respectively. As shown in Fig. 1, Kalman filter cut down the error between observation and real signals.

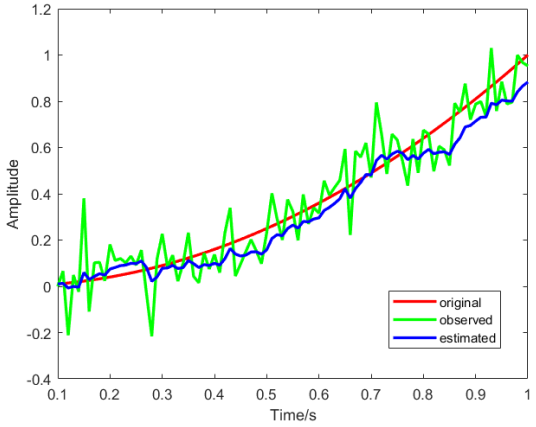

Fig. 1. Estimation for $x_1(t) = t^2$ using Kalman filter with one sensor

Now we add another sensor with higher measurement accuracy and modify the observation equation (Eq. 20), where we tend to trust more on the second sensor and let $R = diag\{3 \quad 5\}$. The filtering result after sensor fusion is illustrated in Fig. 2, where the estimation error is further reduced.

$$\begin{pmatrix} Y_{k_1} \\ Y_{k_2} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} X_k \\ X_k{}' \\ X_k{}'' \end{pmatrix} + R_k \tag{20}$$

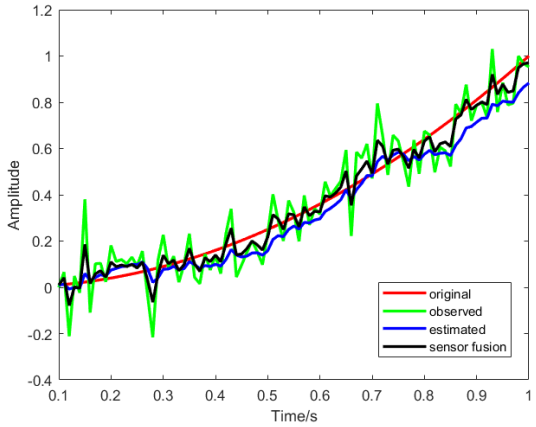

Fig. 2. Estimation for $x_1(t) = t^2$ using Kalman filter after sensor fusion

Then we choose another signal with stronger nonlinearity. The real signal $x_2(t)$ follows that $x_2(t+1) = sin(3x_2(t))$ and $x_2(0) = 0.1$. We still use the model in Eq. 18 and Eq. 19, and the simulation result is shown in Fig. 3. The limitation of Kalman filter is shown that it cannot track the nonlinear signal and result in a large estimation error.

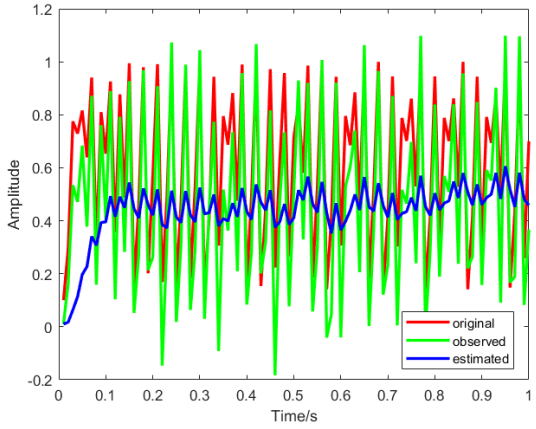

Fig. 3. Estimation for $x_2(t)$ using Kalman filter

Instead, we use the extended Kalman filter to process the same signal. The prediction and observation equation is given in Eq. 21, where $Q_k \sim N(0, 0.0001)$ and $R_k \sim N(0, 1)$. Suppose the initial state $X_0 \sim N(0.1, 0.1)$, the estimation performance is plot in Fig. 4, where tracking error is much smaller than that using Kalman filter.

$$\begin{aligned} X_k &= sin(3X_{k-1}) + Q_k \\ Y_k &= X_k^2 + R_k \end{aligned} \tag{21}$$

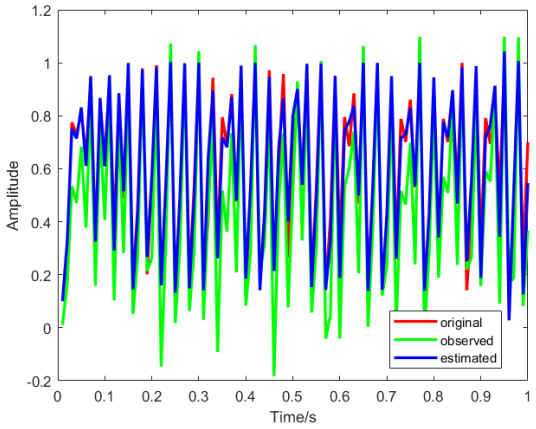

Fig. 4. Estimation for $x_2(t)$ using extended Kalman filter

In conclusion, Kalman filter assume that the prediction and observation function are both linear, and also suppose the error Q_k, R_k follows the normal distribution. Then states can be quickly estimated following predict and update rules in Bayes formula. Due to their bad performance on nonlinear cases, extended Kalman filter expand the nonlinear functions by Taylor series, which greatly improves the filtering performancing. However, EKF algorithm need to calculate the Jacobian matrix iteratively, which may be computationally expensive for complex functions. Moreover, the Jacobian matrix even not exist for discontinuous system. Some functions with strong nonlinearity can also lead to unsatisfactory results.

IV. Particle filters

A. Particle filtering theory

Particle filters have been widely used applied in robot localization and navigation. Unlike Kalman filters that make assumptions on prediction and observation equations, particle filters focuses on dealing with the improper integrals in Bayesian filering algorithm which are hard to acquired analytically. For a probability density function $f(x)$, we can use a bunch of particles with different weights to approach it,

$$f(x) \approx \sum_{i=1}^{n} \omega^{(i)} \delta(x - x^{(i)}) \tag{22}$$

where $\omega^{(i)}$ is the weight for the i^{th} particle and the weights satisfy $\sum_i \omega^{(i)} = 1$. To simplify the problem, here we suppose the random variable of last state X_{k-1} follows the normal distribution. We use n particles $x_{k-1}^{(i)}$ to approach the pdf, i.e. $f_{k-1}(x) = \sum_{i=1}^{n} \omega_k^{(i)} \delta(x - x_{k-1}^{(i)})$. Following the prediction step in Bayesian filtering algorithm, we get the pdf of prior probability.

$$\begin{aligned} f_k^-(x) &= \int_{-\infty}^{+\infty} f_Q[x - F(v)] f_{k-1}(v) dv \\ &= \sum_{i=1}^{n} \omega_{k-1}^{(i)} f_Q[x - F(x_{k-1}^{(i)})] \end{aligned} \tag{23}$$

To avoid sampling in each loop, we hope that the prior pdf is in the particle form like Eq. 22. Denote the pdf of a random variable Z is $f_Z = f_Q[x - F(x_{k-1}^{(i)})]$, and suppose Q follows normal distribution. Then by Fourier convolution theorem we know that Z can be seen as a sum of two independent random variables X and Y (shown in Eq. 24).

$$\begin{aligned} f_Z &= (2\pi Q)^{-\frac{1}{2}} e^{-\frac{[x - F(x_{k-1}^{(i)})]^2}{2Q}} \overset{F.T}{\rightarrow} e^{i \cdot F(x_{k-1}^{(i)})t} \cdot e^{-\frac{Q}{2}t^2} \\ f_X &= \delta[x - F(x_{k-1}^{(i)})] \overset{F.T}{\rightarrow} e^{i \cdot F(x_{k-1}^{(i)})t} \\ f_Y &= (2\pi Q)^{-\frac{1}{2}} e^{-\frac{x^2}{2Q}} \overset{F.T}{\rightarrow} e^{-\frac{Q}{2}t^2} \\ f_Z &= f_X * f_Y \Rightarrow Z = X + Y \end{aligned} \tag{24}$$

Therefore, each Z can be seen as a sum of certain event $X = F(x_{k-1}^{(i)})$ and a random event $Y \sim N(0, Q)$. Then $f_k^-(x)$ can be approximated by n particles $x_k^{(i)}$,

$$\begin{aligned} f_k^-(x) &= \sum_{i=1}^{n} \omega_{k-1}^{(i)} \delta(x - x_k^{(i)}) \\ x_k^{(i)} &= F(x_{k-1}^{(i)}) + v, v \sim N(0, Q) \end{aligned} \tag{25}$$

In this prediction step, only the positions of particles on the real line are changed. After an observation value y_k is acquired by the sensor, the pdf of posterior probability is derived following the update step in Bayesian filtering algorithm.

$$\begin{aligned} f_k^+(x) &= \eta f_R[y_k - H(x)] f_k^-(x) \\ &= \sum_{i=1}^{n} \eta f_R[y_k - H(x_k^{(i)})] \omega_{k-1}^{(i)} \delta(x - x_k^{(i)}) \\ &= \eta \sum_{i=1}^{n} \omega_k^{(i)} \delta(x - x_k^{(i)}) \end{aligned} \tag{26}$$

In this step, it can be seen as the update of particles' weights while their positions remain unchanged.

$$\omega_k^{(i)} = f_R[y_k - H(x_k^{(i)})] \omega_{k-1}^{(i)} \tag{27}$$

After the two steps, position and weight of particles are both changed, and the posterior probability density function can be approximated by these particles. Then the estimated state can be given by

$$\hat{x}_k^+ = \int_{-\infty}^{+\infty} x \sum_{i=1}^{n} \omega_k^{(i)} \delta(x - x_k^{(i)}) = \sum_{i=1}^{n} \omega_k^{(i)} x_k^{(i)} \tag{28}$$

B. Resampling

In practice, after normalizing the weights in each update step, few particles may take extremely high weights and lead to particle degeneracy, which will cause the failure of next update step. This is due to the limited number of particles and the normal distribution form of the pdf f_R. Therefore, resampling procedure is need in each iteration. In resampling algorithm, particles are randomly duplicated and killed but the

sum remains unchanged. Usually, we assume that particles with higher weights are more likely to be duplicated while those with lower weights tend to be eliminated. After that, all weights of particles are set to be $\frac{1}{n}$ so that particle degeneracy problem is solved. The whole algorithm pseudocode of particle filtering with resampling procedure is given below.

Algorithm 4 Particle filter

Prediction equation $X_k = F(X_{k-1}) + Q_k$
Observation equation $Y_k = H(X_k) + R_k$
$Q_k \sim N(0, Q)$, $R_k \sim N(0, R)$, $X_0 \sim N(\mu_0^+, \sigma_0^+)$
Initialize μ_0^+, σ_0^+, Q, R
Produce n particles $x_0^{(i)}$, set all weights $\omega_0^{(i)} = \frac{1}{n}$
for k=1,...,r **do**
  $x_k^{(i)} = F(x_{k-1}^{(i)}) + v, v \sim N(0, Q)$ {prediction step}
  Obtain an observation value y_k
  $\omega_k^{(i)} = f_R[y_k - H(x_k^{(i)})] \cdot \omega_{k-1}^{(i)}$ {update step}
  $\omega_k^{(i)} = \frac{\omega_k^{(i)}}{\sum \omega_k^{(i)}}$ {normalization}
  Resampling and set all weights to be $\frac{1}{n}$
  $\hat{x}_k^+ = \sum_{i=1}^n \omega_k^{(i)} x_k^{(i)}$
end for

C. Test results

Here we use the same nonlinear signal when testing EKF in section 3, and select 100 particles all with initial value 0.1 and weights $\frac{1}{100}$. We choose to believe more our observation function and set $Q = 0.01$ and $R = 0.001$. The simulation result is shown in Fig. 5, where the nonlinear signal can be recovered with relatively small estimation error.

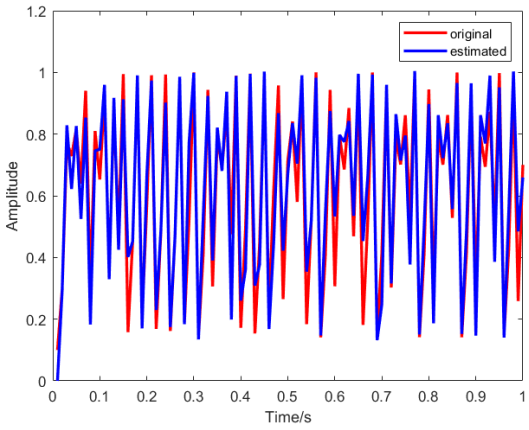

Fig. 5. Estimation for $x_2(t)$ using particle filter

Actually particle filter tends to have better performance than EKF on nonlinear functions. Here we simply assume that f_Q and f_R are both in shape of Gaussian probability-density function, and simply set the initial position and weights for particles, which may be result in error. Some numerical methods for sampling from an arbitrary pdf can be applied to improve the estimation performance, such as inverse-cumulative-distribution-funtion transform method (ITM) and the sample and reject method.

V. Design of an EKF-based DWA planner

Dynamic window approach (DWA) is a local path planner that widely used in robot navigation. In each control period, a dynamic window is created as $(v_{min}, v_{max}, \omega_{min}, \omega_{max})^T$ according to the kinematic limitation of robot and the outside environment. By sampling the velocities in the window, we suppose they remain unchanged in next time period and generate all possible trajectories. Those trajectories are scored by an evaluation function (Eq. 29) that considers the heading cost, velocity and distance towards obstacles.

$$G(v, \omega) = \sigma(\alpha \cdot heading(v, \omega) + \beta \cdot dist(v, \omega) + \gamma \cdot velocity(v, \omega)) \tag{29}$$

Then we select the best trajectory and choose the corresponding velocity $(v, \omega)^T$ as input for the controller. The best trajectory is selected iteratively until the robot reaches the goal (Fig. 6).

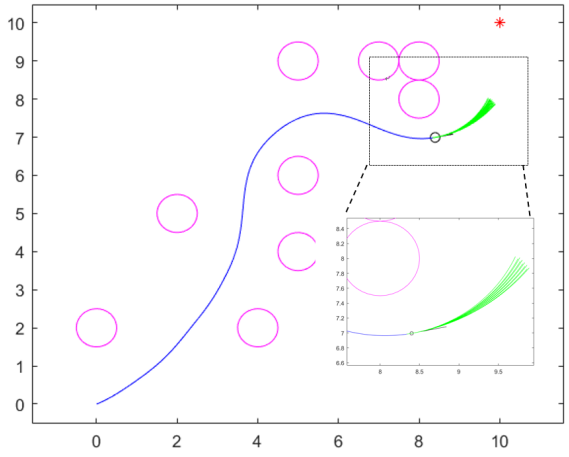

Fig. 6. Trajectories generated by DWA planner

In real cases, the position of robot can not be acquired accurately, and moving error also exists with the given input. Therefore, we apply the EKF for real-time state estimation. We observe the robot's position by a GPS sensor with noise $\omega_1 \sim N(0, 0.1)$ and the moving error by motor $\omega_2 \sim N(0, 0.01)$. For state estimation, the dynamic model of differential robot can be established as

$$
\begin{aligned}
X_k &= \begin{pmatrix} 1&0&0&0&0 \\ 0&1&0&0&0 \\ 0&0&1&0&0 \\ 0&0&0&0&0 \\ 0&0&0&0&0 \end{pmatrix} \begin{pmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \\ v_{k-1} \\ \omega_{k-1} \end{pmatrix} + \begin{pmatrix} (dt)cos\theta_{k-1} & 0 \\ (dt)sin\theta_{k-1} & 0 \\ 0 & dt \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} v_{k-1} \\ \omega_{k-1} \end{pmatrix} + Q_k \\
Y_k &= \begin{pmatrix} 1&0&0&0&0 \\ 0&1&0&0&0 \end{pmatrix} \begin{pmatrix} x_k \\ y_k \\ \theta_k \\ v_k \\ \omega_k \end{pmatrix} + R_k
\end{aligned} \tag{30}
$$

We assume that $Q_k \sim N(0, Q)$ and $R_k \sim N(0, R)$, where $Q = diag\{0.01, 0.01, 0, 0, 0\}$ and $R = diag\{0.1, 0.1\}$. The Jacobian matrixes is calculated as

$$
\begin{aligned}
A &= \begin{pmatrix} 1&0&-v_{k-1}(dt)sin\theta_{k-1} & (dt)cos\theta_{k-1} & 0 \\ 0&1&v_{k-1}(dt)cos\theta_{k-1} & (dt)sin\theta_{k-1} & 0 \\ 0&0&1 & 0 & dt \\ 0&0&0 & 1 & 0 \\ 0&0&0 & 0 & 1 \end{pmatrix} \\
C &= \begin{pmatrix} 1&0&0&0&0 \\ 0&1&0&0&0 \end{pmatrix}
\end{aligned} \tag{31}
$$

Each time we follow the prediction step to get the prior estimation $\hat{x}_k^-$, and update it with the observation value y_k from GPS. The posterior estimation $\hat{x}_k^+$ is then used as the state for selecting the next best trajectory. Simulation results for this EKF-based DWA planner are shown in Fig. 7.

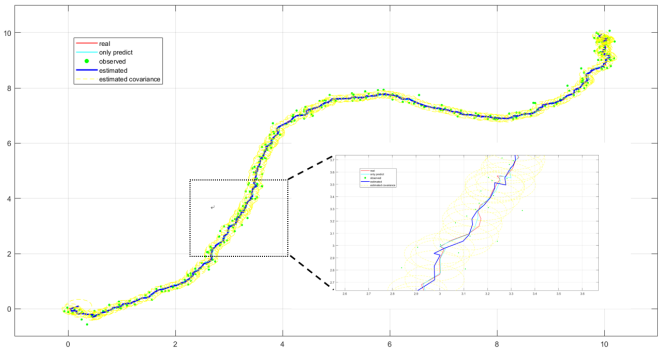

Fig. 7. Simulation results for the EKF-based DWA planner

The red line represent the real trajectory of robot while the blue line is the estimated one. With the relatively unaccurate observation values (green points), the EKF-based DWA planner realized real-time localization with low estimation error and path planning for navigation. Here we suppose that all noise follows the normal distribution to simplify the problem, but in real cases the noise distribution may have different probability density functions, which limits the performance of EKF.

VI. Conclusion

Acknowledgment

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression “one of us (R. B. G.) thanks ...”. Instead, try “R. B. G. thanks...”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

References

[1] J. J. Leonard and H. F. Durrant-Whyte, “Mobile robot localization by tracking geometric beacons,” *IEEE Transactions on robotics and Automation*, vol. 7, no. 3, pp. 376–382, 1991.

[2] J. Borenstein, H. R. Everett, L. Feng, and D. Wehe, “Mobile robot positioning: Sensors and techniques,” *Journal of robotic systems*, vol. 14, no. 4, pp. 231–249, 1997.

[3] A. A. Ahmed, H. Shi, and Y. Shang, “Sharp: A new approach to relative localization in wireless sensor networks,” in *25th IEEE International Conference on Distributed Computing Systems Workshops*. IEEE, 2005, pp. 892–898.

[4] S. Wang, Y. Li, S. Zhang, B. Wang, and H. Yang, “Relative localization of swarm robotics based on the polar method,” *International Journal of Advanced Robotic Systems*, vol. 19, no. 1, p. 17298806221080634, 2022.

[5] L. Chen, H. Hu, and K. McDonald-Maier, “Ekf based mobile robot localization,” in *2012 Third International Conference on Emerging Security Technologies*. IEEE, 2012, pp. 149–154.

[6] A. Barrau and S. Bonnabel, “An ekf-slam algorithm with consistency properties,” *arXiv preprint arXiv:1510.06263*, 2015.

[7] Y.-S. Park, W.-S. Choi, S.-I. Han, and J.-M. Lee, “Navigation system of uuv using multi-sensor fusion-based ekf,” *Journal of institute of control, robotics and systems*, vol. 22, no. 7, pp. 562–569, 2016.

[8] W. Xu, J. Yang, H. Wei, J. Mao, H. Lu, and X. Li, “Error-state kalman filter-based localization algorithm with velocity estimation for deep-sea mining vehicle,” *Ocean Engineering*, vol. 264, p. 112331, 2022.

[9] J. H. Han and N. Y. Ko, “Ukf localization of a mobile robot in an indoor environment and performance evaluation,” *Journal of the Korean Institute of Intelligent Systems*, vol. 25, no. 4, pp. 361–368, 2015.

[10] C. Xu, M. Ji, Y. Qi, and X. Zhou, “Mcc-ckf: A distance constrained kalman filter method for indoor toa localization applications,” *Electronics*, vol. 8, no. 5, p. 478, 2019.

[11] Q.-b. Zhang, P. Wang, and Z.-h. Chen, “An improved particle filter for mobile robot localization based on particle swarm optimization,” *Expert Systems with Applications*, vol. 135, pp. 181–193, 2019.