ME8135 Assignment 2

Glenn Shimoda

October 2020

1 Introduction

1.1 Overview

The purpose of this assignment is to simulate a robot moving in a circle around a landmark using the Extended Kalman Filter algorithm. Two measurement models are used. First, the same linear measurement model from assignment 1 is used. Then, the bearing/distance measurement to the landmark is used as the measurement model. The system was simulated using pygame in Python.

1.2 Nonlinear, Non-Gaussian Estimation

In chapter 3 of the text by Barfoot [1], the motion and measurement models are linear and the noise variables are Gaussian, which greatly simplifies the problem. Solving the peak of the posterior density via the *Maximum A Posteriori(MAP)* method results in the same solution as using the *Full Bayesian Posterior* method. This is because the MAP method gets the peak of the density, which is the mode, and the Bayesian method gets the mean of the entire distribution. When the distribution is Gaussian, the mean and mode are the same. Also, the motion and measurement models are linear since the transition and observation matrices only involve linear operations and thus the distribution of the variables are not changed, and thus remain Gaussian.

In chapter 4 of the text, we are introduced to nonlinear and/or non-Gaussian models. This means that the noises may no longer be Gaussian, and that the motion/measurement models are nonlinear, which changes the distribution of the variables they use. The models are now given by the functions f() and g():

$$Motion : x_k = f(x_{k-1}, v_k, w_k) \tag{1}$$

$$Measurement : y_k = g(x_k, n_k) \tag{2}$$

1.3 Extended Kalman Filter

The Extended Kalman Filter passes the inputs to f() and g() through a linear approximation about an operating point, which will be the posterior of the previous state $\hat{x}_{k-1}$ for the motion model f(), and the prior of the current state $\check{x}_k$ for the measurement model g(). The models are approximated by:

$$f(x_{k-1}, v_k, w_k) \approx \check{x}_k + F_{k-1}(x_{k-1} - \hat{x}_{k-1}) + w'_k \tag{3}$$

$$\check{x}_k = f(\hat{x}_{k-1}, v_K, 0) \tag{4}$$

$$F_{k-1} = \frac{\partial f(x_{k-1}, v_k, w_k)}{\partial x_{k-1}}\Big|_{\hat{x}_{k-1}, v_k, 0} \tag{5}$$

$$w'_k = \frac{\partial f(x_{k-1}, v_k, w_k)}{\partial w_k}\Big|_{\hat{x}_{k-1}, v_k, 0} w_k \tag{6}$$

$$g(x_k, n_k) \approx \check{y}_k + G_k(x_k - \check{x}_k) + n'_k \tag{7}$$

$$\check{y}_k = g(\check{x}_k, 0) \tag{8}$$

$$G_k = \frac{\partial g(x_k, n_k)}{\partial x_k}\Big|_{\check{x}_k, 0} \tag{9}$$

$$n'_k = \frac{\partial g(x_k, n_k)}{\partial n_k}\Big|_{\check{x}_k, 0} n_k \tag{10}$$

We note that equations (5), (6), (9), and (10) are Jacobians. (6) and (10) multiply the Jacobians by the difference of the operating point and expected noise value, which is zero in the case of white noise(zero-mean Gaussian). After taking the first two moments of these linear approximations (assuming they are Gaussian), and some tedious algebra with the Bayes filter, we arrive at the canonical Extended Kalman Filter equations:

$$Predictor\ Covariance : \check{P}_k = F_{k-1}\hat{P}_{k-1}F_{k-1}^T + Q'_k \tag{11}$$

$$Predictor\ Mean : \check{x}_k = f(\hat{x}_{k-1}, v_K, 0) \tag{12}$$

$$Kalman\ Gain : K_k = \check{P}_k G_k^T (G_k \check{P}_k G_k^T + R'_k)^{-1} \tag{13}$$

$$Corrector\ Covariance : \hat{P}_k = (1 - K_k G_k)\check{P}_k \tag{14}$$

$$Corrector\ Mean : \hat{x}_k = \check{x}_k + K_k(y_k - g(\check{x}_k, 0)) \tag{15}$$

We note this is very similar to the linear version of the Kalman Filter, with a few adjustments. First, as shown in equations (11), (13), and (14), the linear transition and observation matrices A_k and C_k are replaced by the Jacobians F_k and G_k which are the linear approximations of the nonlinear f() and g(), at zero noise. We also note that the covariance matrices Q'_k and R'_k have Jacobians incorporated into them. Finally, the predictor and corrector steps for the mean replace the linear f() and g() zero noise functions with the nonlinear f() and g() functions from (1) and (2) with zero noise.

2 Problem

2.1 Setup

The problem is similar to homework 1(same robot parameters) except a landmark L is placed at the coordinates (10, 10). In order to turn, angle is incorporated into the linear speed equation with bearing noise:

$$\dot{\theta}_k = \frac{r}{L}u_\psi + w_\psi \tag{16}$$

$$\theta_k = T\dot{\theta}_k + \theta_{k-1} \tag{17}$$

$$\dot{X}_k = \begin{bmatrix} \dot{x}_k \\ \dot{y}_k \end{bmatrix} = \begin{bmatrix} ru_\omega cos(\theta_k) + w_\omega \\ ru_\omega sin(\theta_k) + w_\omega \end{bmatrix} \tag{18}$$

$$X_k = T\dot{X}_{k-1} + X_{k-1} = f(x_{k-1}, v_K, w_k) \tag{19}$$

$$w_k = \begin{bmatrix} w_\omega & 0 \\ 0 & w_\psi \end{bmatrix} \tag{20}$$

The bearing noise w_ψ is incorporated into cos and sin functions which are nonlinear, while the input noise w_ω is added which is a linear transformation. The linear approximations by equations (3) to (6) are:

$$\check{X}_k = f(x_{k-1}, v_K, 0) = \begin{bmatrix} \hat{x}_{k-1} \\ \hat{y}_{k-1} \end{bmatrix} + T \begin{bmatrix} ru_\omega cos(T\frac{r}{L}u_\psi + \theta_{k-1}) \\ ru_\omega sin(T\frac{r}{L}u_\psi + \theta_{k-1}) \end{bmatrix} \tag{21}$$

$$F_{k-1} = \frac{\partial f(X_{k-1}, v_k, w_k)}{\partial X_{k-1}}\Big|_{\hat{X}_{k-1}, v_k, 0} = \frac{\partial (T\dot{X}_{k-1} + X_{k-1})}{\partial X_{k-1}}\Big|_{\hat{X}_{k-1}, v_k, 0} = 1 = I_2 \tag{22}$$

$$w'_k = \frac{\partial f(x_{k-1}, v_k, w_k)}{\partial w_k}\Big|_{\hat{x}_{k-1}, v_k, 0} w_k = \begin{bmatrix} \frac{\partial f_x}{\partial w_\omega} & \frac{\partial f_x}{\partial w_\psi} \\ \frac{\partial f_y}{\partial w_\omega} & \frac{\partial f_y}{\partial w_\psi} \end{bmatrix} \begin{bmatrix} w_\omega & 0 \\ 0 & w_\psi \end{bmatrix} = \begin{bmatrix} \frac{\partial f_x}{\partial w_\omega} w_\omega & \frac{\partial f_x}{\partial w_\psi} w_\psi \\ \frac{\partial f_y}{\partial w_\omega} w_\omega & \frac{\partial f_y}{\partial w_\psi} w_\psi \end{bmatrix} \tag{23}$$

$$\frac{\partial f}{\partial_\omega} = T\begin{bmatrix}1\\1\end{bmatrix} \tag{24}$$

$$\frac{\partial f}{\partial_\psi} = T^2 r u_\omega \begin{bmatrix}-sin(T\frac{r}{L}u_\psi + \theta_{k-1})\\ cos(T\frac{r}{L}u_\psi + \theta_{k-1})\end{bmatrix} \tag{25}$$

$$Q_k' = \begin{bmatrix}Var(w_x) & Cov(w_x, w_y)\\ Cov(w_x, w_y) & Var(w_y)\end{bmatrix} \tag{26}$$

$$Var(w_x) = \frac{1}{2}[(w_k'(1,1) - \bar{w}_x)^2 + (w_k'(1,2) - \bar{w}_x)^2] \tag{27}$$

$$Var(w_y) = \frac{1}{2}[(w_k'(2,1) - \bar{w}_x)^2 + (w_k'(2,2) - \bar{w}_x)^2] \tag{28}$$

$$\bar{w}_x = \frac{1}{2}[w_k(1,1) + w_k(1,2)] \tag{29}$$

$$\bar{w}_y = \frac{1}{2}[w_k(2,1) + w_k(2,2)] \tag{30}$$

$$Cov(w_x, w_y) = \frac{1}{2}[(w_k'(1,1) - \bar{w}_x)(w_k'(2,1) - \bar{w}_y) + (w_k'(1,2) - \bar{w}_x)(w_k'(2,2) - \bar{w}_y)] \tag{31}$$

In the case of the same linear measurement model, nothing changes, and the observation matrix takes the place of the Jacobian, giving the exact same corrector step in the Kalman Filter.

In the case of the new measurement model using bearing and distance measurements from the landmark(see the drawing in the handwritten derivation):

$$\phi_k(\theta_k, n_\phi) = \frac{\pi}{2} + \theta_k + n_\phi \tag{32}$$

$$d(X_k, n_d) = d(x_k, y_k, n_d) = \sqrt{(x_k - 10)^2 + (y_k - 10)^2} + n_d \tag{33}$$

$$z_k = \begin{bmatrix}10 + dcos(\phi_k)\\ 10 + dsin(\phi_k)\end{bmatrix} = g(X_k, n_k) \tag{34}$$

$$n_k = \begin{bmatrix}n_d & 0\\ 0 & n_\phi\end{bmatrix} \tag{35}$$

Where the noise variables are $n_d \sim N(0, 0.1)$ and $n_\phi \sim N(0, 0.01)$. Now, linearizing these with equations (7) to (10):

$$g(x_k, n_k) \approx \check{z}_k + G_k(X_k - \check{X}_k) + n_k' \tag{36}$$

$$\check{z}_k = g(X_k, 0) \tag{37}$$

$$G_k = \frac{\partial g(X_k, n_k)}{\partial X_k}\Big|_{\check{X}_k,0} = \frac{1}{\sqrt{(\check{x}_k-10)^2+(\check{y}_k-10)^2}}\begin{bmatrix}(\check{x}_k-10)^2cos(\phi_k) & (\check{y}_k-10)^2cos(\phi_k)\\(\check{x}_k-10)^2sin(\phi_k) & (\check{y}_k-10)^2sin(\phi_k)\end{bmatrix} \tag{38}$$

$$n'_k = \frac{\partial g(x_k, n_k)}{\partial n_k}\Big|_{\check{x}_k,0} n_k = \begin{bmatrix}n_d cos(\phi_k) & -dn_\phi sin(\phi_k)\\ n_d sin(\phi_k) & dn_\phi sin(\phi_k)\end{bmatrix}\Big|_{d(\check{X_k},0),\phi_k(\theta_k,0)} \tag{39}$$

$$R'_k = \begin{bmatrix}Var(n_x) & Cov(n_x,n_y)\\ Cov(n_x,n_y) & Var(n_y)\end{bmatrix} \tag{40}$$

$$Var(n_x) = \frac{1}{2}[(n'_k(1,1)-\bar{n}_x)^2 + (n'_k(1,2)-\bar{n}_x)^2] \tag{41}$$

$$Var(n_y) = \frac{1}{2}[(n'_k(2,1)-\bar{n}_x)^2 + (n'_k(2,2)-\bar{n}_x)^2] \tag{42}$$

$$\bar{n}_x = \frac{1}{2}[n_k(1,1)+n_k(1,2)] \tag{43}$$

$$\bar{n}_y = \frac{1}{2}[n_k(2,1)+n_k(2,2)] \tag{44}$$

$$Cov(n_x,n_y) = \frac{1}{2}[(n'_k(1,1)-\bar{n}_x)(n'_k(2,1)-\bar{n}_y) + (n'_k(1,2)-\bar{n}_x)(n'_k(2,2)-\bar{n}_y)] \tag{45}$$

2.2 Algorithm

We want to move the robot in a circle of radius 4m around the landmark. The steps of the algorithm are as follows:

1. **Initialize the State and Variables -** with (6,10) for both the coordinate X_0 and zero for the covariance matrix of the initial state $\check{P}_0$. We start at 6, 10, which is 4 meters to the left of the landmark, while being at the same y coordinate. The initial angle $\theta_0 = \frac{\pi}{2}$ which indicates the robot is facing upwards in the positive y direction. Set the constant control variables as $u_r = 4.75, u_l = 5.25$. These were developed from the zero noise model to circle around the landmark with a distance around 4m at all times.

2. **Predict the State -** by using equations (16) to (31) in equations (11) and (12) for the prior coordinate and covariance. Repeat this step 8 times to reflect 8 predictions per second.

3. **Correct the State -** use equations (32) to (45) in equations (13) to (15) to get the posterior coordinate and covariance estimates in the case of the bearing/distance measurement model. For a linear measurement model, use equations (13) to (15) by replacing R'_k with R_k (no Jacobian), G_k with $C_k = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$, and $n_k = \begin{bmatrix} r_x & 0 \\ 0 & r_y \end{bmatrix}$, with $r_x \sim N(0, 0.05)$, and $r_y \sim N(0, 0.075)$, and replace $g(\check{x}_k, 0)$ with $C_k \check{x}_k$.

4. **Repeat steps 2 and 3**

3 Results

Figure 1 shows the result of the linear measurement model until the robot approximately traced out a full circle, with the red dots being the motion updates, the blue line being the measurement correction, and figure 2 showing green ellipses as the covariance ellipse. Figures 3 and 4 show the same results for the bearing/distance measurement model. Both models achieve fairly accurate results, but there are imperfections in the circular motions. Both implementations approximate the motion model as linear, but it is a minor adjustment as only a small bearing noise variable, w_ψ is passed through the nonlinear sine and cosine functions. The distributions of sine and cosine are much different than the Gaussian distribution that the noise comes from, as the peaks of distributions of sine and cosine functions don't match their mean, which is the operating point of the linear approximation of the EKF, and it works under the assumption that the mean of the input passed through the non-linearity is the mean of the output of the non-linearity, which is not the case as pointed out by Barfoot (pg.108). Noise is also linearly added with the w_ω variable, but this does not undergo any non-linearities, so its effect on the motion model can be expected to be Gaussian.

The way the two implementations differ is in their covariance. In both implementations, covariances were very small, and needed to be scaled up to be shown in the pygame implementation. The linear measurement model however had results consistent with the goal of the Kalman Filter, which is reducing covariance through each measurement. In the range/distance measurement model, however, the covariance kept increasing even after implementing the measurement model, showing the shortcomings of the EKF as another linear approximation starts to show an increase in uncertainty. The EKF is only guaranteed to work with slightly nonlinear systems, and assumptions of the operating points being the true mean, as well as approximating the Jacobians as the coefficient matrices is a a source of error. The reason the linear measurement model may work better for the correction is because its distribution is exactly Gaussian since it's linear, so the certainty in the final prediction of the posterior mean is higher.

Finally, both systems have inherent bias because of the assumptions of linearity and the correct value as the operating point. These biases make it easier to estimate the robot's state, but further refinement to the EKF can achieve more accurate results. For example, the Iterative Extended Kalman Filter(IEKF) iterates multiple times until it converges to an accurate operating point. Approaches like Monte Carlo method and Sigmapoint Transformation transform the points without assuming they're linear, which can capture the first moments of the posterior distribution more accurately than from the linearization from the EKF. A recursive filter implementing the Monte Carlo method is the particle filter, which will be explored next.

References

[1] T. D. Barfoot, *State Estimation for Robotics.* Cambridge University Press, 2020.

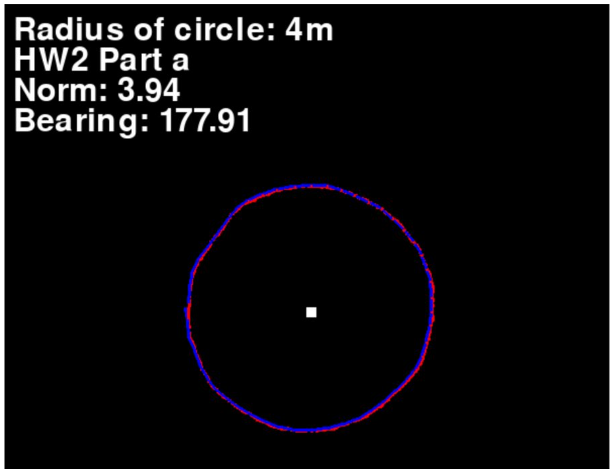

Figure 1: EKF with Linear Measurement Model

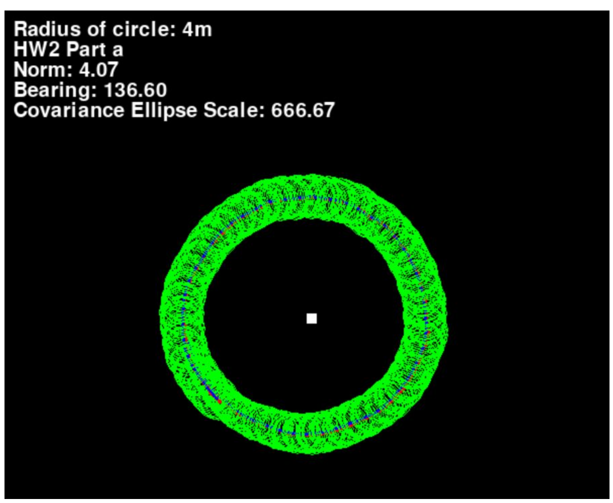

Figure 2: EKF with Linear Measurement Model with Covariance Ellipse

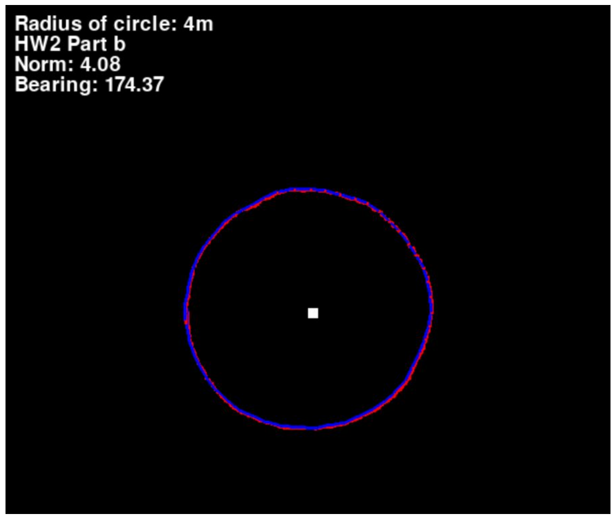

Figure 3: EKF with Linear Bearing/Distance Measurement Model

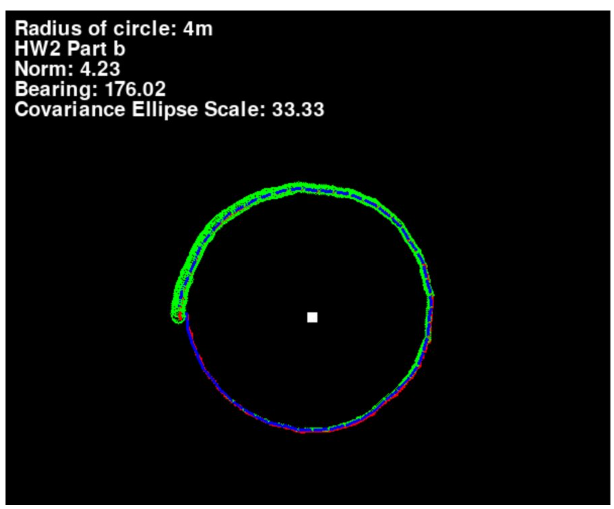

Figure 4: EKF with Linear Bearing/Distance Measurement Model with Covariance Ellipse