ME8135 Assignment 1

Glenn Shimoda

October 2020

1 Introduction

1.1 Overview

The purpose of this assignment is to simulate a robot moving in a straight line using a simple Kalman Filter algorithm. The system was simulated using pygame in python.

1.2 Linear Gaussian Estimation

The equations and concepts are provided by Barfoot [1]. In Linear Gaussian estimation, both the motion model of the robot and the observation model are linear when estimating its position:

$$Motion\ Model : x_k = A_{k-1}x_{k-1} + v_k + w_k,\ k = 1, ...K \tag{1}$$

$$Observation\ Model : y_k = c_k x_k + n_k,\ k = 1, ...K \tag{2}$$

Where $x_k \in R^N$ represents the coordinates of the robot at time k, $v_k \in R^N$ is the control input at time k, and $w_k \in R^N \sim N(0, Q_k)$ is the zero-mean process noise at time k. $n_k \in R^M \sim N(0, R_k)$ is the zero mean measurement noise. $A_k \in R^{N\times N}$ and $C_k \in R^{M\times N}$ are the transition and observation matrix, which are the main transformation equations to derive the motion and observation model, and will change depending on the problem as shown in the assignment problems.

As shown in the simple linear Gaussian case, the coefficient matrices are simple linear transformations, and the noise variables are all normally distributed. Thus, starting with an initial guess of the state $x_0 \in R^N \sim N(\check{x}_0, \check{P}_0)$ that is normally distributed, the distribution of the state remains the same throughout the process and is known to be Gaussian. Thus, the estimate can be accurate, and its accuracy assessed as the mean and mode of a Gaussian distribution are the same.

1.3 Kalman Filter

The Kalman Filter is a classic recursive algorithm that first predicts the state using the past state and current input (prior), which is then corrected using the latest measurement (posterior). In the linear case, the Kalman Filter is BLUE(best linear unbiased estimator), since it performs right at the Cramer-Rao Lower Bound (can't perform better than this.)

In canonical form, the five main equations for the Kalman Filter are:

$$Predictor\,Covariance : \check{P}_k = A_{k-1}\hat{P}_{k-1}A_{k-1}^T + Q_k \tag{3}$$

$$Predictor\,Mean : \check{x}_k = A_{k-1}\hat{x}_{k-1} + v_k \tag{4}$$

$$Kalman\,Gain : K_k = \check{P}_k C_k^T (C_k \check{P}_k C_k^T + R_k)^{-1} \tag{5}$$

$$Corrector\,Covariance : K_k = (1 - K_k C_k)\check{P}_k \tag{6}$$

$$Corrector\,Mean : \hat{x_k} = \check{x}_k + K_k(y_k - C_k\check{x}_k) \tag{7}$$

An important term in equation (7) is the innovation term, $y_k - C_k\check{x}_k$ as its the difference between the actual measurement model with noise and the expected value of the measurement(no noise), thus in the corrector mean equation, all the uncertainty is linearly added to the prior to get the posterior.

2 Problem

2.1 Setup

In this problem, a 2D simple robot has the parameters r=0.1m for the radius of the wheel, ur=ul=0.1m/s, and motion prediction occurs every 0.125 second, giving a period of T = 1/8 of a second. Noise is incorporated in both the x and y dimensions as $w_x \sim N(0, 0.1)$ and $w_y \sim N(0, 0.15)$. Due to 50% higher variance for noise in the y, the motion models are expected to deviate more along the y-coordinates. The velocity model is given by:

$$\dot{X} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \frac{r}{2} \begin{bmatrix} u_r + u_l + w_x \\ u_r + u_l + w_y \end{bmatrix} \tag{8}$$

This equation is then updated by using noting that $\dot{X} = \frac{X_k - X_{k-1}}{T}$:

$$\check{X}_k = \begin{bmatrix} \check{x} \\ \check{y} \end{bmatrix} = X_{k-1} + T\dot{X}_k \tag{9}$$

Note: capital X is used to denote the general coordinates, while lower case x is used for the horizontal coordinate in the regular x-y Cartesian coordinate

system. We will also use z to represent measurement instead of y that was mentioned earlier. We note that if we equate this corrector form to equations (1) or (4), then the transition matrix A_k is simply the identity matrix, while the product $T\dot{X}_k$ is the input v_k with the noise already incorporated. Due to the simplicity of the transition matrix, the covariance of the prior can easily be calculated(using equation 6) as:

$$\check{P}_k = A_{k-1}\hat{P}_{k-1}A_{k-1}^T + Q_k = \hat{P}_{k-1} + Q_k \tag{10}$$

where the covariance matrix of the process noise, Q_k, is given by:

$$Q_k = \begin{bmatrix} w_x & w_x w_y \\ w_x w_y & w_y \end{bmatrix} \tag{11}$$

The measurement model is given by the simple equation:

$$z_k = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} r_x & 0 \\ 0 & r_y \end{bmatrix} \tag{12}$$

Where we can relate the above to equation 2 to see that the first matrix is the observation matrix and the second matrix added at the end is the noise matrix, with the white noises being $r_x \sim N(0, 0.05)$ and $r_y \sim N(0, 0.075)$. We observe that once again, the variance in the y is 50% larger than in the x, and we note that the measurement noise variance is much smaller than the input noise variance, reflecting real life where sensor readings tend to be more certain that inputs, such as rotating a shaft to move the wheels of a robot. Given the measurement model, we can use the Kalman Gain and corrector steps of the Kalman filter equations (5)-(7) by noting the covariance matrix of the measurement noise is given by:

$$R_k = \begin{bmatrix} r_x & r_x r_y \\ r_x r_y & r_y \end{bmatrix} \tag{13}$$

The final information in the problem is that 8 motion predictions are calculated before 1 measurement is made (thus, 8 predictions and 1 measurement per second). The motion prediction will drift with progressively larger covariance(visualized by a covariance ellipse) before an accurate measurement is incorporated to give a more certain coordinate.

2.2 Algorithm

The steps of the algorithm are as follows:

1. **Initialize the State** - with zeros for both the coordinate X_0 and the covariance matrix of the initial state $\check{P}_0$
2. **Predict the State** - by using equations (8) and (9) for the prior coordinate, and equations (3), (10), and (11) for the covariance of the prior. Repeat this step 8 times to reflect 8 predictions per second.

3. **Correct the State** - use equations (12), (13), and (5)-(7) to get the posterior coordinate and covariance estimates.

4. **Repeat steps 2 and 3**

3 Results

Figure 1 shows the result of the algorithm through 8 measurements, with the red dots being the motion updates, the blue line being the measurement correction, and the green ellipses being the covariance ellipse. As expected, the model drifts more in the vertical direction due to higher variances among both the input and measurement noise. Also, the path laid out by the measurement steps are far more linear(and thus certain and closer to ground-truth) than simply the motion steps. This confirms the strength of the Kalman filter in using both a theoretical motion model along with a measurement model to create the best estimate of a system's current state.

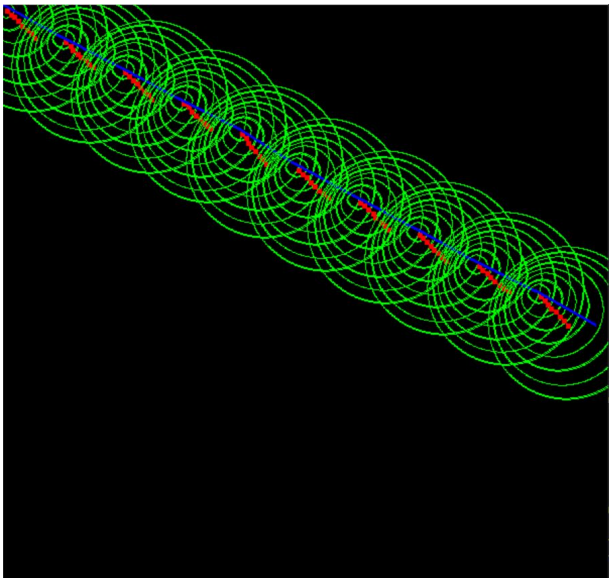

Figure 1: Kalman Filter Result

References

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