Directed Studies: Assignment 2

Navid Zarrabi
navid.zarrabi@torontomu.ca

Toronto Metropolitan University — June 3, 2023

1 Kalman Filter

Kalman Filter Use pyGame, or any other similar libraries, to simulate a simplified 2D robot and perform state estimation using a Kalman Filter. Motion Model:

$$\dot{x} = \frac{r}{2}(u_r + u_l) + w_x \tag{1}$$

$$\dot{y} = \frac{r}{2}(u_r + u_l) + w_y \tag{2}$$

r = 0.1 m, is the radius of the wheel, ur and ul are control signals applied to the right and left wheels. wx = N(0, 0.1) and wy = N(0, 0.15) Simulate the system such that the robot is driven 1 m to the right. Assume the speed of each wheel is fixed and is 0.1 m/s Use these initial values:

$$x_0 = 0 \tag{3}$$

$$y_0 = 0 \tag{4}$$

$$P_0 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \tag{5}$$

Assume the motion model is computed 8 times a second. Assume every second a measurement is given:

$$z = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} r_x & 0 \\ 0 & r_y \end{bmatrix} \tag{6}$$

where $r_x = N(0, 0.05)$ and $r_y = N(0, 0.075)$

First, I have written $\dot{x}$ in discrete format:

$$\dot{x} = \frac{x_t - x_{t-1}}{T} \tag{7}$$

and same goes for y. I have used notation from reference [?] where next state probability is expressed in the following form:

$$X_t = A_t X_{t-1} + B_t U_t + \epsilon_t \tag{8}$$

In this problem, $A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $B = \frac{rT}{2}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$, $\epsilon_t = \begin{bmatrix} Tw_x \\ Tw_t \end{bmatrix}$, $X = \begin{bmatrix} x_t \\ y_t \end{bmatrix}$, and $U_t = \begin{bmatrix} u_r \\ u_l \end{bmatrix}$. These values are obtained from rewriting equations 1 and 2 in matrix format. First and second moments of equation 8 results in a belief with following mean and covariance:

$$\begin{aligned} \bar{\mu}_t &= A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t &= A_t \Sigma_{t-1} A_t^T + R_t \end{aligned} \tag{9}$$

These results are merely from motion model. The measurements should also be considered for better accuracy. Measurement probability is as follows:

$$z_t = C_t X_t + \delta_t \tag{10}$$

where $c_t = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$, and $\delta_t = \begin{bmatrix} r_x \\ r_y \end{bmatrix}$. Hence, there are two normal distributions from two sources: Motion Model and Measurements. To combine these information, Kalman Gain is defined as:

$$K_t = \Sigma_t C_t^T \left(C_t \Sigma_t C_t^T + Q_t \right)^{-1} \tag{11}$$

Using K_t, a new normal distribution with the following mean and covariance is obtained:

$$\begin{aligned} \mu_t &= \bar{\mu}_t + K_t \left(z_t - C_t \bar{\mu}_t \right) \\ \Sigma_t &= \left(I - K_t C_t \right) \bar{\Sigma}_t \end{aligned} \tag{12}$$

Simulating algorithm explained above in PyGame yields the following results.

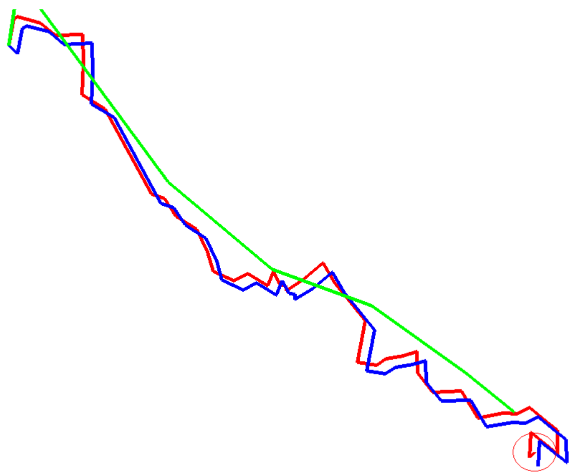

Figure 1: Simulation results for Kalman Filter

In Figure 1, green line shows the measurements (Equation 10), red line shows the ground truth pose (Equation 8), and blue line shows the output of Kalman filter (Equation 12). Measurements have a lower frequency than state according to the questions and that's why the green line is always behind the others. Whenever a new measurement arrives, the uncertainty ellipse becomes smaller. The updated state estimate and covariance matrix reflect the new information provided by the measurement, and are more accurate than the predicted state estimate and covariance matrix. The fusion of the prediction and measurement information reduces the uncertainty in the estimated state and improves the tracking accuracy.

2 Extended Kalman Filter

Repeat the previous assignment, this time with a classic motion model and range observations made from a landmark located at $M = [10, 10]$. L is the distance between the wheel, known as wheelbase, and is 0.3m.

$$\dot{x} = \frac{r}{2} \left(u_r + u_l \right) \cos(\theta) + w_x \tag{13}$$

$$\dot{y} = \frac{r}{2} \left(u_r + u_l \right) \sin(\theta) + w_y \tag{14}$$

$$\dot{\theta} = \frac{r}{L} \left(u_r - u_l \right). \tag{15}$$

Assume:

$$u_\omega = \frac{1}{2} \left(u_r + u_l \right), u_\psi = \left(u_r - u_l \right)$$

Then the equations become:

$$\dot{x} = r u_\omega \cos(\theta) + w_\omega, \quad \dot{y} = r u_\omega \sin(\theta) + w_\omega, \quad \dot{\theta} = \frac{r}{L} u_\psi + w_\psi$$

Program the robot such that it loops around point M.
(a) Compute the EKF with the linear measurement model in the previous assignment.

In EKF, linear predictions of Kalman filter are replaced by their non-linear generalizations [?]. In other words, non-linear Equation (16) is used instead of the linear Equation (8).

$$x_t = g(u_t, x_{t-1}) + \epsilon \tag{16}$$

Formulating the problem like first question in matrix form:

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_n \\ y_n \\ \theta_n \end{bmatrix} + \begin{bmatrix} rT\cos(\theta) & 0 \\ rT\sin(\theta) & 0 \\ 0 & Tr/l \end{bmatrix} \begin{bmatrix} u_r \\ u_l \end{bmatrix} + w_n T \tag{17}$$

where $G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ and $B = \begin{bmatrix} rT\cos(\theta) & 0 \\ rT\sin(\theta) & 0 \\ 0 & Tr/l \end{bmatrix}$. The measurement probability is the same as previous question according to the question.

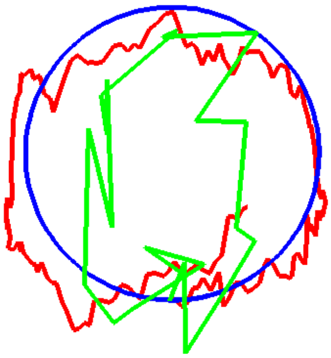

Figure 2: Simulation for EKF and linear measurements

Also, differential drive control is applied on the robot using if-then statements to keep the robot at a specified distance. In Figure 2, green lines are measurements, blue line is the ground truth, and the red line is the output of EKF.

(b) Compute the EKF with range/bearing measurements of point M. Assume range noise is $N(0, 0.1)$ and bearing noise is $N(0, 0.01)$. Range is in meters, and bearing is in radians. Visualize the measurements as well.

At this part, the measurement probability changes in a non-linear manner. Distance and Angle of the robot relative to the landmark gives us the polar cordinate of the robot:

$$X = \begin{bmatrix} \sqrt{(c_x - x)^2 + (c_y - y)^2} \\ \arctan(c_y - y)\,/\,(c_x - x) - \theta \end{bmatrix} \tag{18}$$

Jacobian of vector X is:

$$H = \begin{bmatrix} \frac{-c_x + x}{\sqrt{x^2+y^2}} & \frac{-c_y + y}{\sqrt{x^2+y^2}} & 0 \\ \frac{y - c_y}{\sqrt{(c_x-x)^2+(c_y-y)^2}} & \frac{x - c_x}{\sqrt{(c_x-x)^2+(c_y-y)^2}} & -1 \end{bmatrix}. \tag{19}$$

In the context of EKF, H substitutes C in Equations 11 and 12. The whole process is the same as part a, but we use Jacobian H instead of C and update the Kalman gain and new position using polar coordinates from the landmark.

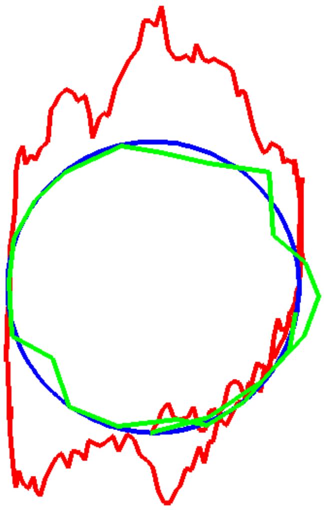

Figure 3: Simulation for EKF and non-linear measurements for u=0.5

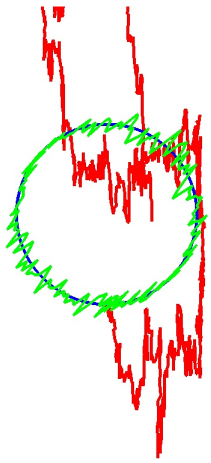

Figure 4: Simulation for EKF and non-linear measurements for u=0.1

R is also defined as:

$$H = \begin{bmatrix} \sigma^2_{range} & 0 \\ 0 & \sigma^2_{bearing} \end{bmatrix}. \tag{20}$$

As shown in Figure 3, the measurement probability looks closer to ground truth. However, the predicted pose using Kalman filter gets away from the reference from time to time. The result heavily depends on the amount of noise and robot speed configurations. For the test in Figure 3, $u_r = u_l = 0.5$ have been chosen. Figure 4 shows the results for $u_r = u_l = 0.1$. In some cases, the predicted pose of the system may deviate from the actual reference pose over time. This can happen if the noise in the measurements is too high, or if the system is moving too quickly for the filter to accurately track its state.