Directed Studies: Assignment 3

Navid Zarrabi
navid.zarrabi@torontomu.ca

Toronto Metropolitan University — June 11, 2023

1 Particle Filter

Use pyGame, or any other similar libraries, to simulate a simplified 2D robot and perform state estimation using a Particle 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 [1] 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. The measurements should also be considered for better accuracy. Measurement probability is as follows:

$$z_t = C_t X_t + \delta_t \tag{9}$$

where $c_t = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$, and $\delta_t = \begin{bmatrix} r_x \\ r_y \end{bmatrix}$. Particle filter is among the non-parametric approaches and the idea of this approach is to represent the posterior by a set of random state samples (particles). The pseudo-code of this algorithm is shown in Figure 1.

First, M particles should be generated using a distribution. According to references [1] and [2], it is possible to use different distributions but the most common distribution is Gaussian. Therefore, bi-variate

1: **Algorithm Particle_filter**($\mathcal{X}_{t-1}, u_t, z_t$)**:**
2: $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
3: for $m = 1$ to M do
4: sample $x_t^{[m]} \sim p(x_t \mid u_t, x_{t-1}^{[m]})$
5: $w_t^{[m]} = p(z_t \mid x_t^{[m]})$
6: $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
7: endfor
8: for $m = 1$ to M do
9: draw i with probability $\propto w_t^{[i]}$
10: add $x_t^{[i]}$ to $\mathcal{X}_t$
11: endfor
12: return $\mathcal{X}_t$

Figure 1: Particle Filter pseudo-code [1]

Gaussian distribution is used to generate M random samples (particles). These values are initially loaded to X_{t-1} and the particles are shifted using state Equation 8 as mentioned in line 4 of the algorithm. Some weights or important factors (w) is defined for each particle. These values are proportional to the probability of current observation given the current state as shown in Equation 9. Figure 2 illustrates these weights as the radius of circles. This Figure shows the prediction of particle filter after a few loops from last measurement and the random noise has moved some of the more probable particles far from the current position. However, there are still several circles with large importance factors around current position.

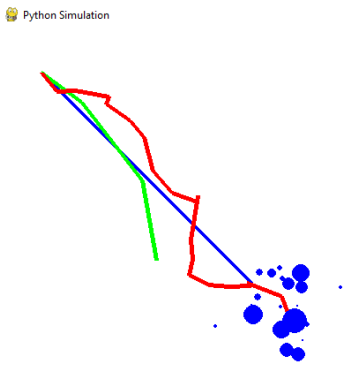

Figure 2: Particle Filter weights

Each particle is weighted with its corresponding importance factor and added to the current belief according to line 6 of the algorithm. Next, resampling is needed to modify the importance factors and particles. According to lines 8 to 10, M random particles should be drawn from the existing particle set. The selection process is according to expectation of particles and particles with higher importance factor have more chance to be selected at this step. Figure 3 shows the implementation of particle filters. First sub-figure shows the initial distribution of particles after a new measurement. The particles are dense since a new measurement decreases uncertainty of prediction. As we see in Figures 2 to 4, the particles are getting sparser, which is a result of more uncertainty after a few prediction steps from last measurement. The blue line shows the ground truth, the green line is the measurement, and the red line is the predicted position using particle filter.

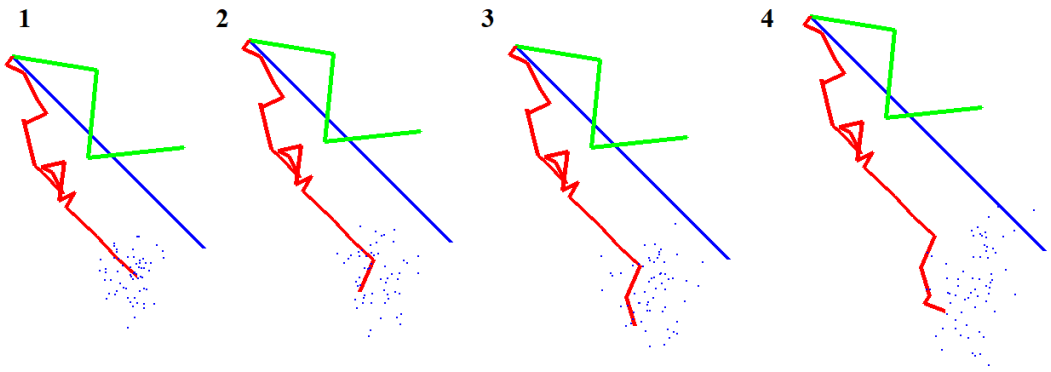

Figure 3: Particle Filter implementation

2 Particle Filter on circular path

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.

$$\dot{x} = \frac{r}{2}(u_r + u_l)\cos(\theta) + w_x \tag{10}$$

$$\dot{y} = \frac{r}{2}(u_r + u_l)\sin(\theta) + w_y \tag{11}$$

$$\dot{\theta} = \frac{r}{L}(u_r - u_l). \tag{12}$$

Assume:

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

Then the equations become:

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

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

In this question, non-linear Equation (13) is used instead of the linear state Equation (8).

$$x_t = g(u_t, x_{t-1}) + \epsilon \tag{13}$$

Formulating the problem like first question in matrix form:

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_n \\ y_n \\ \theta_n \end{bmatrix} + \begin{bmatrix} rT\cos(\theta) & 0 \\ rT\sin(\theta) & 0 \\ 0 & Tr/l \end{bmatrix} \begin{bmatrix} u_r \\ u_l \end{bmatrix} + w_n T \tag{14}$$

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.

Also, differential drive control is applied on the robot using if-then statements to keep the robot at a specified distance. In Figure 4, green lines are measurements, blue line is the ground truth, and the red line is the output of particle filter.

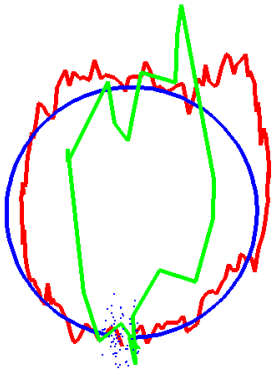

Figure 4: Comparing Predicted path (red) with ground truth (blue)

(b) Compute the Particle Filter 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 coordinate of the robot:

$$X = \begin{bmatrix} \sqrt{(c_x - x)^2 + (c_y - y)^2} \\ \arctan\left(c_y - y\right) / \left(c_x - x\right) - \theta \end{bmatrix} \tag{15}$$

Jacobian of vector X is:

$$H = \begin{bmatrix} \frac{-c_x + x}{\sqrt{x^2+y^2}} & \frac{-c_y + y}{\sqrt{x^2+y^2}} & 0 \\ \frac{y - c_y}{\sqrt{(c_x-x)^2+(c_y-y)^2}} & \frac{x - c_x}{\sqrt{(c_x-x)^2+(c_y-y)^2}} & -1 \end{bmatrix}. \tag{16}$$

R is also defined as:

$$H = \begin{bmatrix} \sigma^2_{range} & 0 \\ 0 & \sigma^2_{bearing} \end{bmatrix}. \tag{17}$$

Figure 5 shows the Predicted position for this part. Prediction results for this part seem closer to the measurements comparing to part a. Sometimes the simulations fail to produce desirable results and the predicted path deviates from ground truth but this rarely happens in part b simulations.

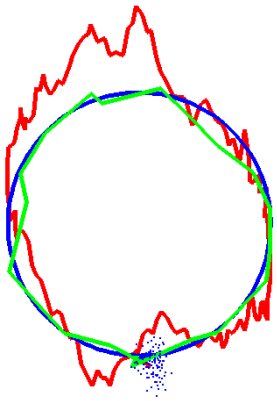

Figure 5: Comparing Predicted path (red) with ground truth (blue)

References

[1] Thrun, Sebastian. "Probabilistic robotics." Communications of the ACM 45, no. 3 (2002): 52-57.

[2] Barfoot, Timothy D. State estimation for robotics. Cambridge University Press, 2017.