State Estimation - Assignment 3

Haniyeh Altafi

June 12, 2023

1 Question 1 - Particle Filter

Use Particle Filter, to simulate a simplified 2D robot and perform state estimation using a Kalman Filter. Motion Model:

$$\dot{x} = \frac{r}{2}(u_r + u_l) + w_x \quad , \quad \dot{y} = \frac{r}{2}(u_r + u_l) + w_y$$

r = 0.1 m, is the radius of the wheel, u_r and u_l are control signals applied to the right and left wheels. $w_x = N(0, 0.1)$ and $w_y = N(0, 0.15)$

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

$$x_0 = 0, y_0 = 0, P_0 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \text{ (initial covariance matrix)}$$

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

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

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

Answer :

Below is my explanation of the Particle Filter and its associated algorithm:

The particle filter algorithm, also known as sequential Monte Carlo (SMC) or the bootstrap filter, is a powerful recursive filtering technique used for state estimation in sequential or time-varying systems. It approximates the posterior distribution of the system's state by representing it with a set of random samples called particles.

Here is an explanation of the particle filter algorithm:

1. Initialization:

Generate an initial set of particles, each representing a possible state of the system. These particles are typically drawn from a prior distribution. We assign equal weights to all particles. Here, I used Gaussian Distribution for generating sample particles.

2. Prediction:

Propagate each particle forward in time using a state transition model, typically represented by a stochastic equation. This step accounts for the system's dynamics and predicts the next state of each particle. Incorporate any control inputs or external factors that influence the state transition.

3. Update (Correction):

Obtain measurements from the system that provide information about the true state. Evaluate the likelihood of each particle by comparing its predicted measurement with the actual measurement obtained from z. Adjust the weights of particles based on their likelihood or importance. Particles that better align with the measurements receive higher weights, while those that deviate significantly receive lower weights.

4. Resampling:

Resampling is performed to focus computational resources on particles that have higher weights, ensuring that the particle set represents the posterior distribution more accurately. Draw new particles from the existing set with replacement, according to their weights. Particles with higher weights have a higher chance of being selected, while particles with lower weights may be eliminated. The resampling step essentially concentrates the particle cloud around regions of high posterior probability, discarding less probable particles.

5. Repeat:

Repeat steps 2 to 4 for each subsequent time step, iteratively updating the particle set as new measurements become available. The algorithm evolves over time, adapting the particle set to track the changing state of the system. By iteratively predicting, updating, and resampling, the particle filter algorithm provides an approximation of the posterior distribution of the system's state. The final set of particles represents the estimated state of the system at each time step, allowing for tracking and inference in dynamic environments where the underlying system evolves over time.

In this problem, we have the initial conditions and values below :

$$\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \mu_0$$

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{r}{2}\delta t & \frac{r}{2}\delta t \\ \frac{r}{2}\delta t & \frac{r}{2}\delta t \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

$$\Sigma_0 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$r = 0.1$$

$$\delta_t = \frac{1}{8}$$

$$U = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$R = \begin{bmatrix} \omega_x \delta t & 0 \\ 0 & \omega_y \delta t \end{bmatrix} = \begin{bmatrix} 0.1\delta t & 0 \\ 0 & 0.15\delta t \end{bmatrix}$$

$$Q = \begin{bmatrix} 0.05 & 0 \\ 0 & 0.075 \end{bmatrix}$$

$$z = C\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} r_x \\ r_y \end{bmatrix}$$

$$X_t = A_t X_{t-1} + B_t u_t + R_t t$$

Here, is the plotted trajectory of the robot using the particle filter.

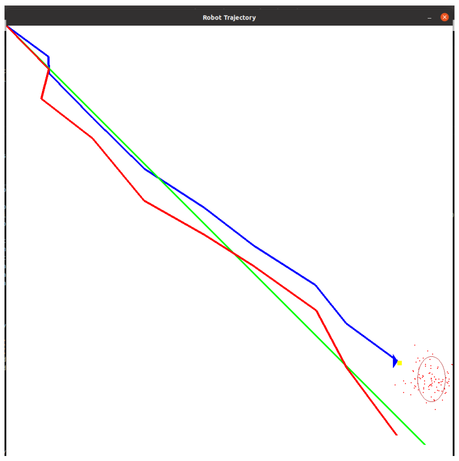

Figure 1: Trajectory of the Robot with Particle Filter. Green is the ground truth. Blue is the estimation. Red is the measurement

2 Question 2 - Particle 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 0.3 m.

$$\dot{x} = \frac{r}{2}(u_r + u_l)\cos(\theta) + w_x \quad , \quad \dot{y} = \frac{r}{2}(u_r + u_l)\sin(\theta) + w_y, \quad \dot{\theta} = \frac{r}{L}(u_r - u_l).$$

Assume

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

Then the equations become:

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

$w_\psi = N(0, 0.01)$ and $w_\omega = N(0, 0.1)$. Program the robot such that it loops around point M.

(a) Compute the Particle Filter with the linear measurement model in the previous assignment.

Answer: To rotate the car around a landmark, control signals must be applied to its wheels. For left rotations, only the left wheel signal should be applied. The rest of the formulation remains unchanged. By incorporating observations, the state covariance decreases, and the estimated positions move more smoothly.
First, we initialize the initial state estimate, error covariance matrix, process noise covariance, and measurement noise covariance.

$$\begin{bmatrix} \hat{x}_{k+1} \\ \hat{y}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_k \\ \hat{y}_k \\ \hat{\theta}_k \end{bmatrix} + \begin{bmatrix} r\Delta t\cos(\theta) & 0 \\ r\Delta t\sin(\theta) & 0 \\ 0 & \frac{r}{l}\Delta t \end{bmatrix} \begin{bmatrix} u_r \\ u_l \end{bmatrix} + w_n\Delta t$$

The covariance matrix will be:

$$P_0 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The procedure is exactly as explained in the previous question, the equations of the robot's trajectory are different.

Here, is the plotted trajectory of the robot.

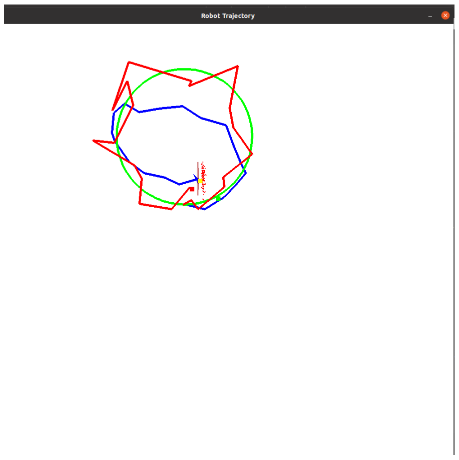

Figure 2: Trajectory of the Robot with Particle Filter. Green is the ground truth. Blue is the estimation. Red is the measurement.