Particle Filters

April 5, 2024

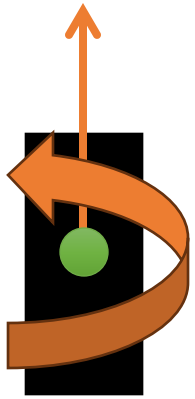
Aditya Penumarti

APRILab
ACTIVE PERCEPTION AND ROBOT INTELLIGENCE LAB

UF UNIVERSITY of FLORIDA

Particle Filter Preliminaries: Motion Model

- The motion model predicts the state into the future. The unicycle kinematics is best for robots with a differential drive system, like a Roomba!



$$\boldsymbol{u_t} = [V, \omega]^T$$

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} + \begin{bmatrix} V\cos\theta_t \\ V\sin\theta_t \\ \omega \end{bmatrix} \Delta t$$

Figure 1. Unicycle Kinematics

Particle Filter Preliminaries: Motion Model

- For a bicycle motion model (like that of a 4-wheeled car), the parameters here are: δ which is the steering angle, and L which is the wheelbase length.

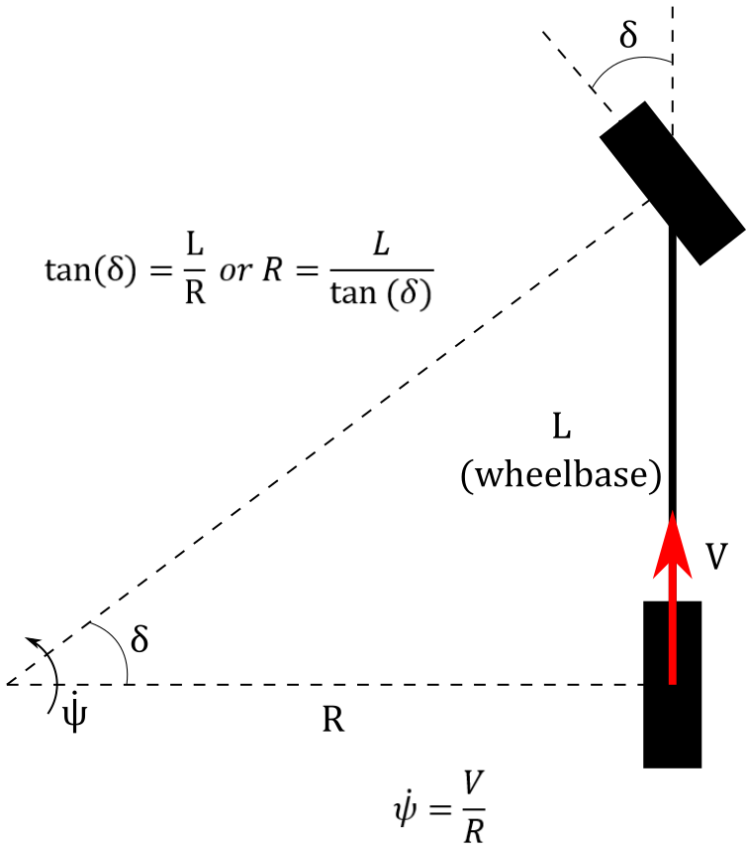


Figure 2. Bicycle Kinematics

$$\boldsymbol{u_t} = \left[V, \frac{V \tan \delta}{L}\right]^T$$

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} + V \begin{bmatrix} \cos \theta_t \\ \sin \theta_t \\ \frac{\tan \delta}{L} \end{bmatrix} \Delta t$$

Particle Filter Preliminaries: Sensor Model

- The sensor model for a particle filter is important to calculate the weights. Here in this toy example a distance to landmark sensor is used. This sensor provides a scalar distance between the robot and the landmark

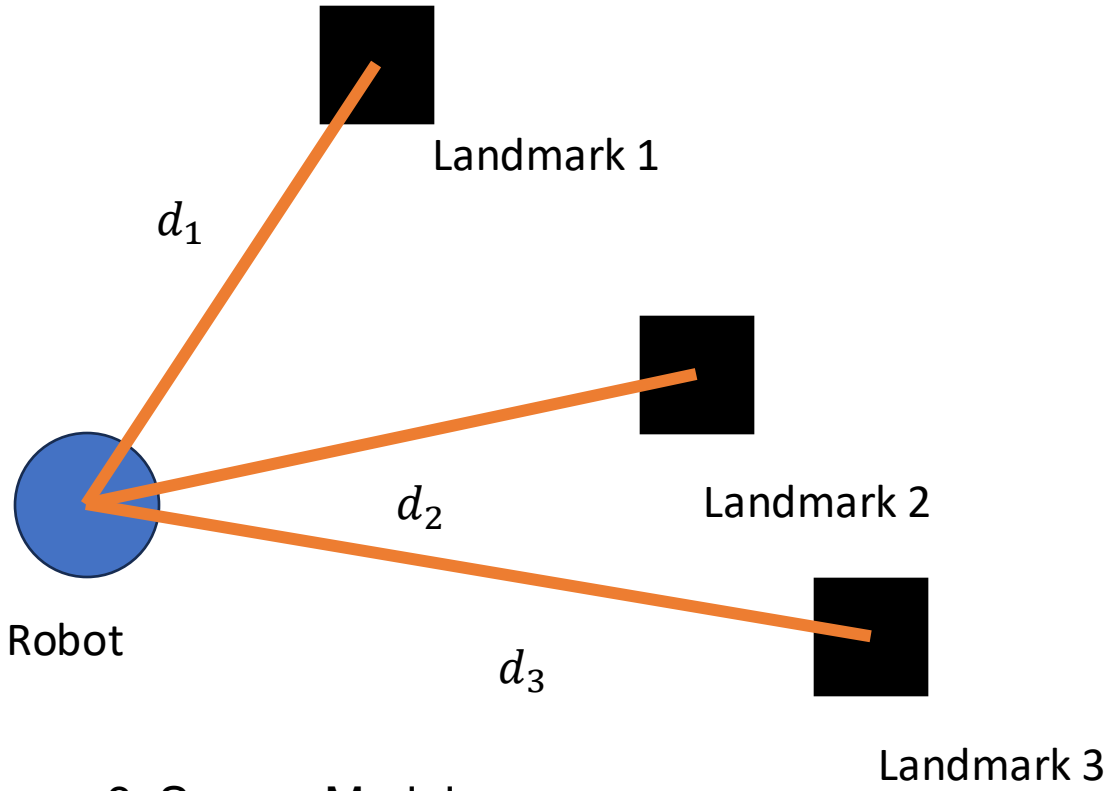


Figure 3. Sensor Model

Since we know the map the sensor model can be represented as such with $\mathbf{x_t}$ as the robot position, l_n as the n^{th} landmark position in xy plane:

$$z_t = || \mathbf{x_t} - l_n||_2$$

which is also d_n. This sensor model is used for the particles as well.



On to Filtering!

Particle Filter Initialization

- First the particles must be initialized. Here we are going to initialize a Gaussian around an initial guess of the robots pose (The particles can also be uniformly initialized). Effectively, we take the particles starting at the robots pose, and randomly distort the particles within a certain radius.

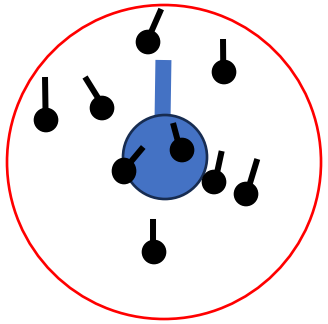


Figure 4. Particle Initialization

Then the weights are initialized to be all uniform:

$$w_t^{[i]} = \frac{1}{N_p}$$

Where N_p is the number of particles and $w_t^{[i]}$ is the i^{th} weight in the particle set at time t.

Particle Filter Prediction

- Then we can predict the particles. This prediction is performed by using the motion model.

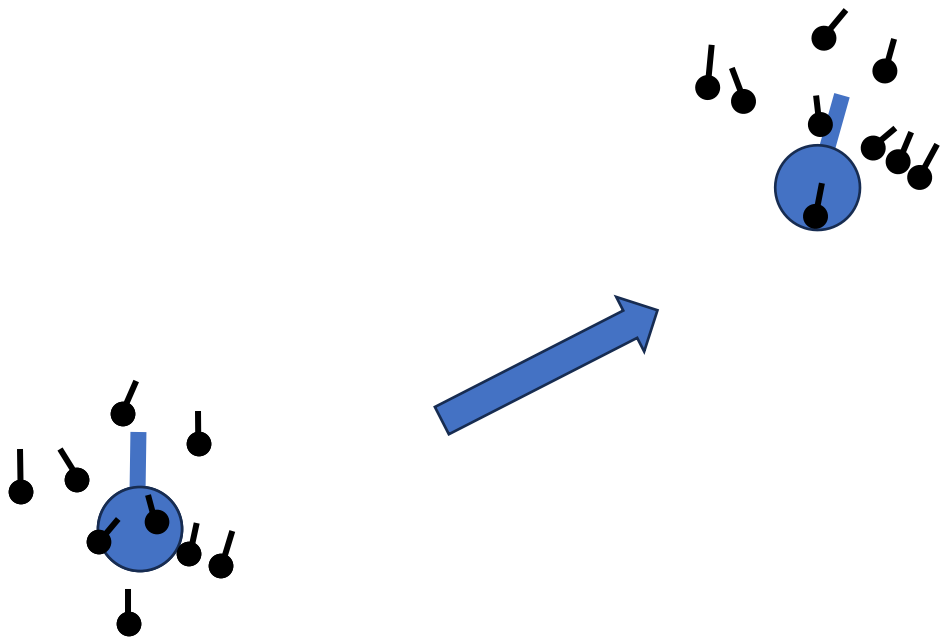


Figure 5. Predict Step

We give the robot an input command, say $\boldsymbol{u} = \left[1\frac{m}{s}, 1\frac{deg}{s}\right]^T$. Let the Δt be 1s as well.

This means, if we start at $\begin{bmatrix} 0\ m \\ 0\ m \\ 0\ deg \end{bmatrix}$ we will move :

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} 0\ m \\ 0\ m \\ 0\ deg \end{bmatrix} + \begin{bmatrix} 1\cos 0 \\ 1\sin 0 \\ 1 \end{bmatrix} 1 = \begin{bmatrix} 1\ m \\ 0\ m \\ 1\ deg \end{bmatrix}$$

If you do this with a small Δt you will start to move in an arcing path.

Particle Filter Update

- Remember now the weights are still uniform. We want to tell the filter that some of the state guesses are not good anymore, or some of them are much better.

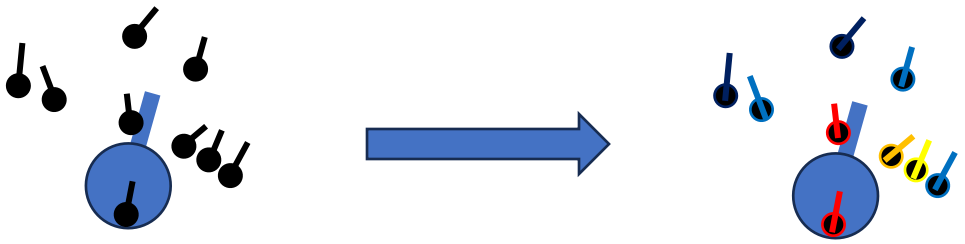


Figure 5. Update Step

The update step is calculated using the sensor model and measurement. We use the robot's measurement from the landmark l_n and compare it to that of the sensor model at each particle location, say $z^{[i]}$.

We then find the likelihood (think how much you would believe this particle to be the right one).

Intuition says the red particles are the most likely and the dark blue are the least. Using that we assign according weights to the particles.

Particle Filter Resample

- Now say the particles get really spread out, and you have mostly blue particles and one red particle. This is when the particles have been degenerated and need to be resampled

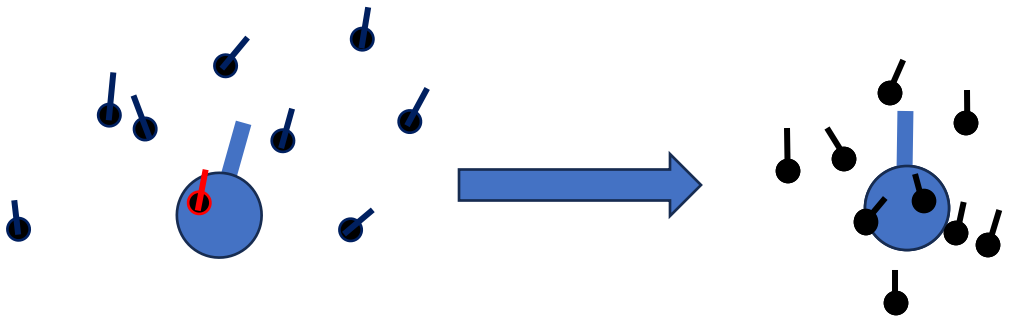


Figure 5. Resample Step

To resample, the highest weighted particles are drawn, and based on that the lowest weighted particles are replaced using these higher weighted particles. The weights are then reset to being $\frac{1}{N_p}$.

This gives you more particles affecting your estimate!

Particle Filter Estimate

- Using the weights and the particles an estimate can be formed. This is effectively just taking the weighted mean of all the guesses and getting an approximate estimated state of the robot

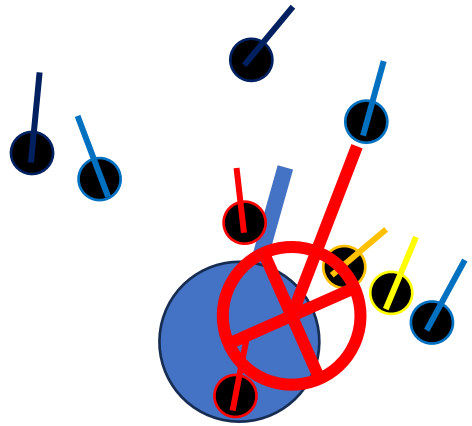
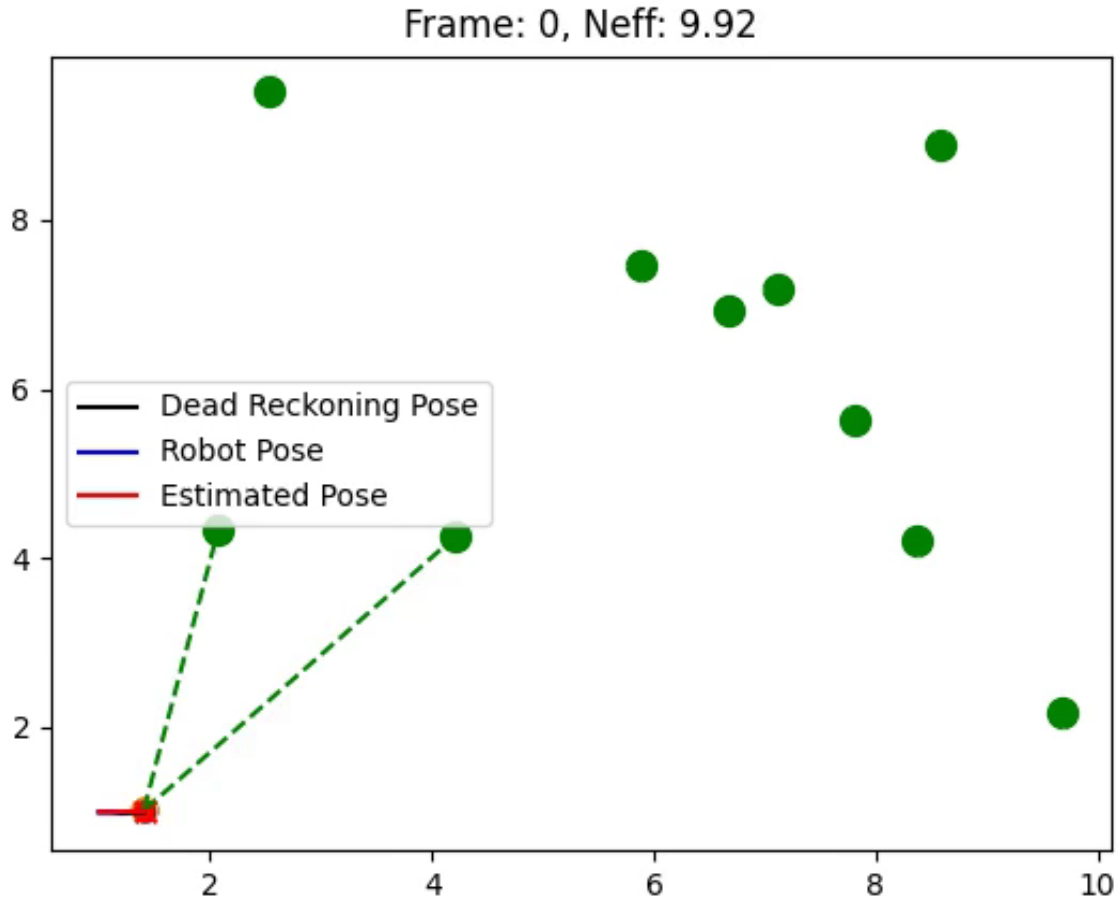


Figure 5. Estimate

Particle Filter Toy Example

- Now using all we have learned, let's try a toy example.



CoLab Link:
https://colab.research.google.com/drive/1cxYa0lfiL-9oqd-kkGMMXUZMvD2n1-0C?usp=sharing

TinyURL Link:
https://tinyurl.com/ycyd3kae