Particle filters or sequential Monte Carlo

Mahboubeh Asadi

I. Motion Model

: The following motion and observation equations are defined for a simplified 2D robot in discrete time.

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} ru_w cos(\theta_{k-1}) \\ ru_w sin(\theta_{k-1}) \\ r/Lu_\psi \end{bmatrix} + \begin{bmatrix} w_w \\ w_w \\ w_\psi \end{bmatrix} \quad (1)$$

$$y_k = \sqrt{(x_k - x_{landmark})^2 + (y_k - y_{landmark})^2} + n_k \quad (2)$$

$$u_w = (u_r + u_l)/2,\ u_\psi = u_r - u_l \quad (3)$$

$r = 0.1m$, is the radius of the wheel, u_r and u_l are control signal applied to the right and left wheels. $w_w = N(0, 0.1)$, $w_\psi = N(0, 0.15)$, $r_r = N(0, 0.05)$, and $r_\psi = N(0, 0.075)$ are considered. k is the discrete-time index and K its maximum. The variables have the following definitions:

- system state: x_k
- initial state: x_0
- input:v_k
- process noise: w_k
- measurement: y_k
- measurement noise: n_k

The goal is to circle around a landmark at $L = (10, 10)$.

II. Particle Filter

This report is mainly from [1]. The particle filter is one of the only practical techniques able to handle non-Gaussian noise and nonlinear observation and motion models. The approach taken here is based on sample importance resampling where the so-called proposal PDF is the prior PDF in the Bayes filter, propagated forward using the motion model and the latest motion measurement, v_k. The sample here refers to the particle, and when the number of samples $N \rightarrow \alpha$, it can approximate any form of probability density distribution. The main steps in the particle filter are as follows:

- Draw M samples from the joint density comprising the prior and the motion noise:

$$\begin{bmatrix} \hat{x}_{k-1,m} \\ W_k, m \end{bmatrix} \leftarrow p(x_{k-1}|\hat{x}_0, v_{1:k-1}, y_{1:k-1})p(w_k) \quad (4)$$

  where m is the unique particle index. In practice we can just draw from each factor of this joint density separately.
- Generate a prediction of the posterior PDF by using v_k. This is done by passing each prior particle/noise sample through the nonlinear motion model:

$$\hat{x}_{k,m} = f(\hat{x}_{k-1,m}, v_k, w_k) \quad (5)$$

  These new 'predicted particles' together approximate the density, $p(x_k|\hat{x}_0, v_{1:k}, y1 : k)$.
- Correct the posterior PDF by incorporating y_k. This is done indirectly in two steps:
  - First, assign a scalar weight, wk;m, to each predicted particle based on the divergence between the desired posterior and the predicted posterior for each particle:

$$w_{k,m} = \frac{p(\hat{x}_{k,m}|\hat{x}_0, v_{1:k}, y_{1:k})}{p(\hat{x}_{k,m}|\hat{x}_0, v_{1:k}, y_{1:k-1})} = \eta p(y_k|\hat{x}_{k,m}), \quad (6)$$

    where η is a normalization constant. This is typically accomplished in practice by simulating an expected sensor reading, $y_{k;m}$, using the nonlinear observation model:

$$\hat{y}_{k,m} = g(\hat{x}_{k,m}, 0) \quad (7)$$

    We then assume $p(y_k|x_{k;m}) = p(y_k|\hat{y}_{k;m})$, where the right-hand side is a known density (e.g., Gaussian).
  - Resample the posterior based on the weight assigned to each predicted posterior particle:

$$\hat{x}_{k,m} \leftarrow \{\hat{x}_{k,m}, w_{k,m}\} \quad (8)$$

    This can be done in several different ways. Madow provides a simple systematic technique to do resampling.

III. Result

If we consider a uniform random distribution for particles, and range as measurement, the result for the 2D robot when it moves around a landmark in a circle path, would be as "Figure 1" and 2".

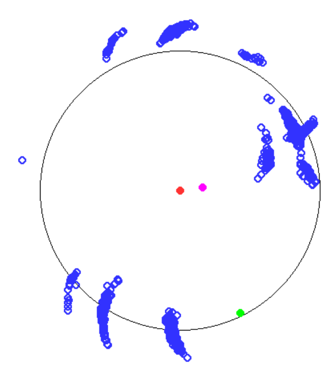

Fig. 1. 2D robot Estimated pose (pink) vs Ground Truth(Green)

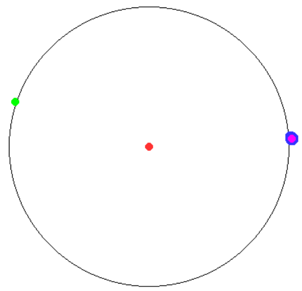

Fig. 2. 2D robot Estimated pose (pink) vs Ground Truth(Green)

As you can see in the "Figure 1" and 2" the estimation for range is accurate, but we have error in estimating the phase. To address this issue, we can add bearing as a measurement as follows:

$$y_k = \begin{bmatrix} \sqrt{(x_k - x_{landmark})^2 + (y_k - y_{landmark})^2} \\ atan2(y_k - y_{landmark}, x_k - x_{landmark}) - \theta_k \end{bmatrix} \quad (9)$$

The result after adding the measurement is as "Figure 3" and 4".

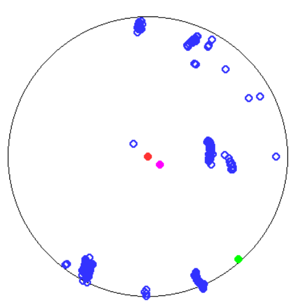

Fig. 3. 2D robot Estimated pose (pink) vs Ground Truth(Green) - measuring range and bearing

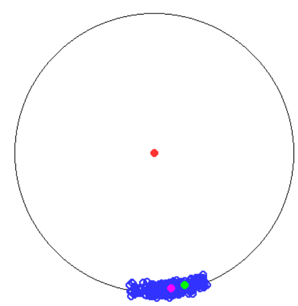

Fig. 4. 2D robot Estimated pose (pink) vs Ground Truth(Green) - measuring range and bearing

REFERENCES

[1] T. D. Barfoot, State estimation for robotics. Cambridge University Press, 2017.