Homework 3

Tyler Amos

26 May 2019

Q3

For code portions (a-c), please see the attached files. Note that I modified the code so it takes an additional option, whether to use period resampling based on a fixed number of iterations or a number of effective particles.

Q3 d - f

See attached files for visualizations.

I noticed in order to have a reasonable approximation of the ground truth, it was necessary to reserve some particles to be resampled according to a uniform distribution, not the likelihood-informed distribution. This helped to avoid particle depletion. In terms of number of particles, I found a relatively small number of particles was best as it allowed the filter to run quickly. Increasing the number of particles over 1,000 seemed to greatly slow down computation. I found a number of particles in the neighbourhood of 600-800 was usually sufficient to have a reasonable approximation. When resampling every time, it was necessary to have the most particles (~800), while resampling intermittently required less (~700), and resampling when the number of effective particles dropped below a threshold required the least (~600 - 700)

I also noticed the filter seemed able to identify the 'type' of location it was in more than the exact place. For example, the filter might indicate the robot's position as the southwest corner when the robot was in fact in the southeast corner. This would suggest the filter is identifying the presence of a corner, and even some elements of the robot's orientation. The filter often improved its estimate when the robot turned after following a straight path for a number of steps. I would hypothesize this is because the change in bearing and linear movement allow for more precise estimates of the robot's position after having narrowed the set of possibilities through straight-line movements.

I found not resampling at each step was helpful for both avoiding particle depletion and improving the estimate of the robot's pose. I hypothesize this is because the sucessive prediction update steps encodes more information about the movement model's likelihood that simply resamping each time. Even better than resampling at a fixed interval was using the number of effective particles to

trigger resampling. In general the estimate of the robot's pose was much better when using the number of effective particles and computation was much quicker.

With respect to the 'kidnapped' robot problem specifically, I found I was able to achieve convergence with a similar number of particles as before. A number of values allowed for convergence, ranging from 600 to just shy of 900. (my random un-weighted resampling ensures there is not complete convergence). A higher value (e.g., 2,000) did provide better performance in terms of accuracy, but would compute slower than smaller numbers.

Q4 a

Let's rewrite the equation as:

$$z = c[x_m - x_t] + w \tag{1}$$

Where z, x_m, x_t, w are vectors and c is a matrix. Now split $[x_m - x_t]$ into two vectors and multiply by the inverse of c, assuming it exists:

$$\begin{aligned} c^{-1}z &= c^{-1}cx_m - x_t + w \\ c^{-1}z + x_t - w &= x_m \end{aligned} \tag{2}$$

Which can be written out and the jacobian solved as:

$$\begin{bmatrix} cos(\theta_t)z_{t,x} - sin(\theta_t)z_{t,y} - x_{t,x} + w \\ sin(\theta_t)z_{t,x} - cos(\theta_t)z_{t,y} - x_{t,y} + w \end{bmatrix} = \begin{bmatrix} x_{m,x} \\ x_{m,y} \end{bmatrix} \tag{3}$$

Solving for the jacobian we obtain:

$$J_{f(x,y,\theta,z_x,z_y)} = \begin{bmatrix} -1 & 0 & -z_y cos(\theta) - z_x sin(\theta) & cos(t) & -sin(\theta) \\ 0 & -1 & -z_y sin(\theta) + z_x cos(\theta) & sin(\theta) & -cos(\theta) \end{bmatrix} \tag{4}$$

Q4 b

Q5

I worked with Tianchu Shu on some of the problems.

I spent approximately 20 hours on this problem set.