Solution to analysis in Home Assignment 4

Arshad Nowsath(nowsath)

May 18, 2020

Contents

1 Analysis

In this report I will present my independent analysis of the questions related to home assignment 1. I have discussed the solution with no-one, I swear that the analysis written here are my own.

2 Smoothing

2.1 a

As can be seen in Figure 1, the smoother estimate is much less noisy and approximates much better the true trajectory than the filter estimate. This can be confirmed in Figure 2, where the position error norm is plotted in the course of time. The smoother error is most of the time lower than the filter error. Reasonably, this make sense, since at the same time instant k, the smoother estimates the position based on all the measurements $p(x_k|y_{1:T})$, while the filter only estimate based on the measurements up to the current time $p(x_k|y_{1:k})$.

For similar reasons, we also expect the smoother error covariance to be smaller than the filter. Having access of previous and future measurements make the estimates much more precise. In other words, in the backward recursion, the smoother will decrease the error covariance. This can be confirmed by looking at the following equation:

$$P_{k|T} = P_{k|k} - G_k[P_{k+1|k} - P_{k+1|T}]G_k^T \tag{1}$$

where $G_K = P_{k,k+1|k}.P_{k+1|k}^{-1} >= 0=$. Recursively, one can easily prove that $P_{k+1|k} - P_{k+1|T} >= 0$ and then conclude that $P_{k|k} - P_{k|T} >= 0$ For the scalar case, we can say that $P_{k|T} < P_{k|k}$.

In addition, since both smoother and filter are based on a CT motion model, we also expect similarities in the trajectory errors. At t=20 and at t=40 when the vehicle changes from straight line to curve and vice-versa, the vehicle experiments high yaw accelerations, which are not likely to happen from the motion motion point of view. From these two time instants the error starts to increase for both estimates.

As a last remark, the smoother and filter error for the last samples are very similar. This is reasonable, since in the extreme case the last filter and smoother estimates will be the same: $p(x_k|y_{1:T})= p(x_k|y_{1:T})$ at k = T.

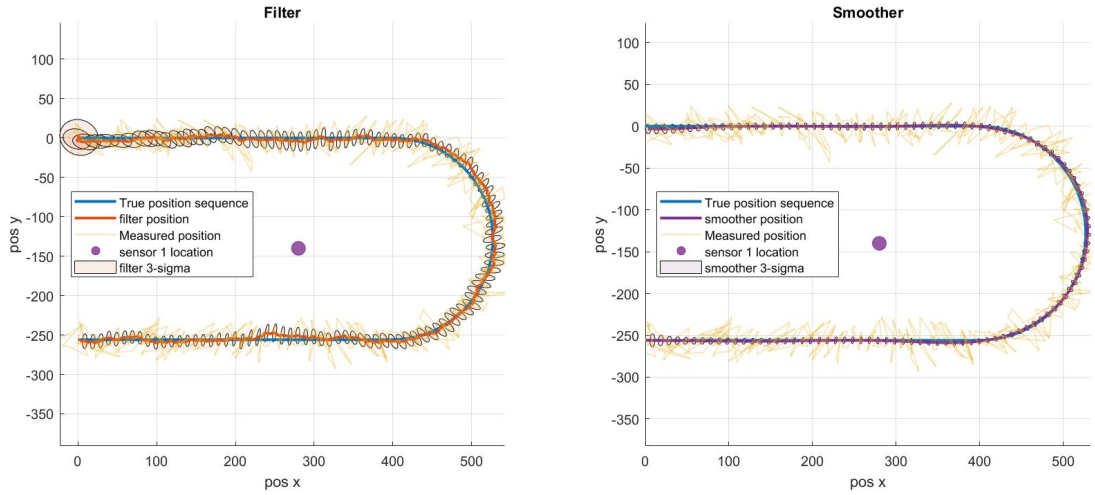


Figure 1: Filtered and smoothed positions

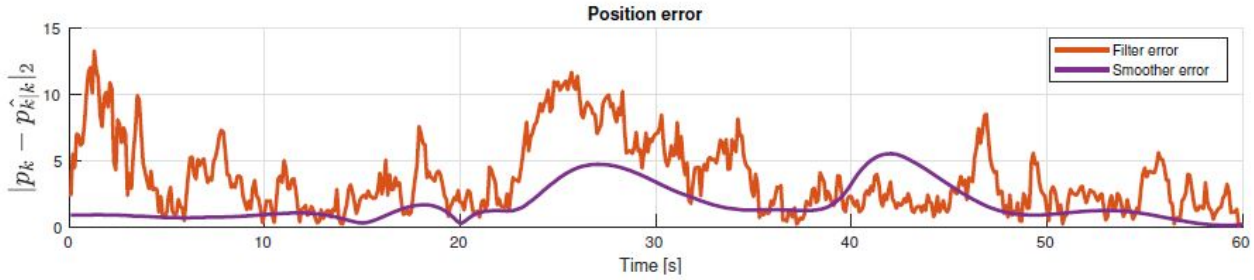


Figure 2: Filtered and smoothed position error

2.2 b

As can be verified in Figure 3, the filter outputs a considerable deviation in comparison to the true trajectory when close to the outlier, which happens at t=15s. This is the largest error in the entire sequence, see 4. However, this error is reasonably small, since the motion model "knows" that such motion detected by the sensor is not likely to happen, and therefore it is able to attenuate this outlier. Furthermore, the smoother is able to overcome the presence of the outlier even better than the filter. The error increases a bit, but this deviation in the trajectory is almost imperceptible. This is possible because at time t=15s the smoother has access to future measurements, and can therefore smooth this outlier based on the knowledge of all the measurements and the motion model. In other words, the backward recursion attenuate even more the filter output calculated in the forward step, based on the prediction of the last time step.

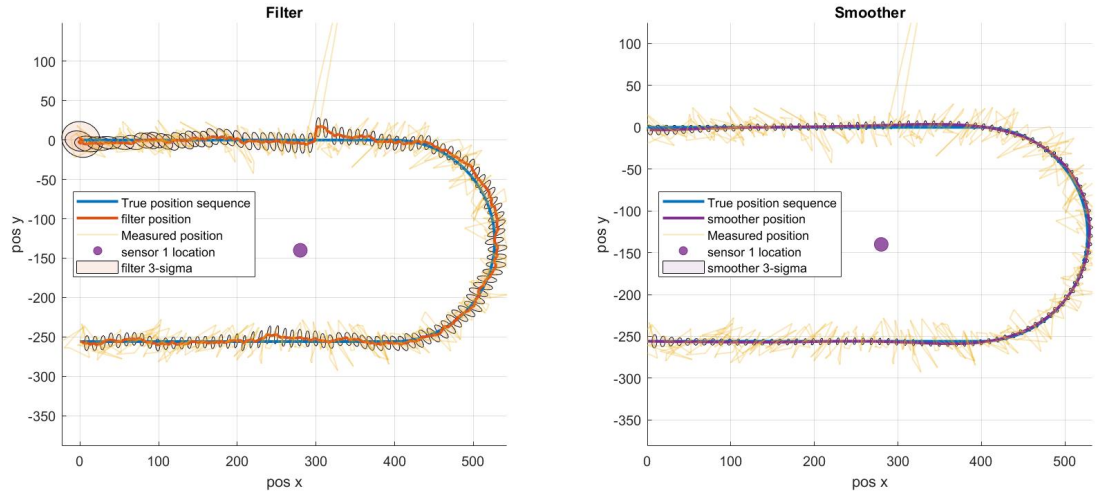


Figure 3: Filtered and smoothed positions with outlier at t=15s

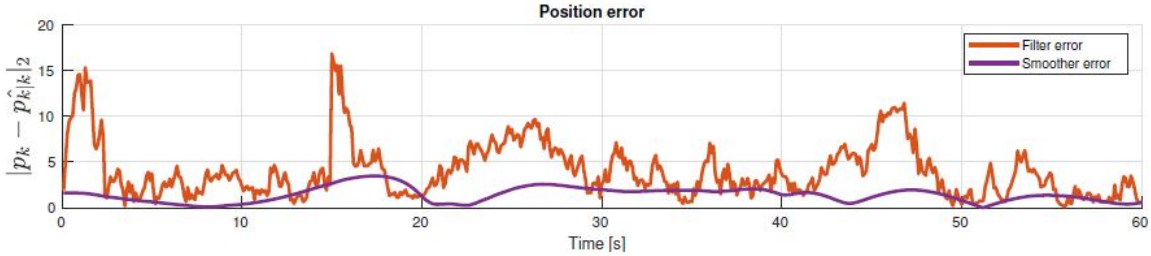


Figure 4: Filtered and smoothed positions with outlier at t=15s

3 Particle filters for linear and Gaussian systems

3.1 a

Figure 5 shows the true trajectory and the different filter estimates over time. Since it is known that the Kalman filter is the optimal filter for linear motion and measurement models, it will present theoretically the best performance among the filters. However, as we increase the number of particles, the particle filter will approximate even more the optimal Kalman filter estimates. In particular, the particle filter with N=100 particles and resampling approximated very well the Kalman filter estimates, see Figure 5.

Nevertheless, it is evident that the performance of the particle filter without resampling is much worse than the other two, even for high values of N. This happens because without resampling, the particles suffer from degeneracy, which degrades the filter performance in the course of time.

Follow bellow, the performance of the three filters in terms of MSE:

1.MSE KF = 0.9889 - kalman filter

2.MSE PF res = 0.9981 - particle filter with resampling N=100

3.MSE PF no res = 2.6229 particle filter without resampling N=100

As expected, the Kalman filter is optimal and presents the best performance, followed by the particle filter with resampling (close to Kalman but bit worse) and the particle filter without resampling (bad due to degeneracy problems). The posterior distributions for the three filters are

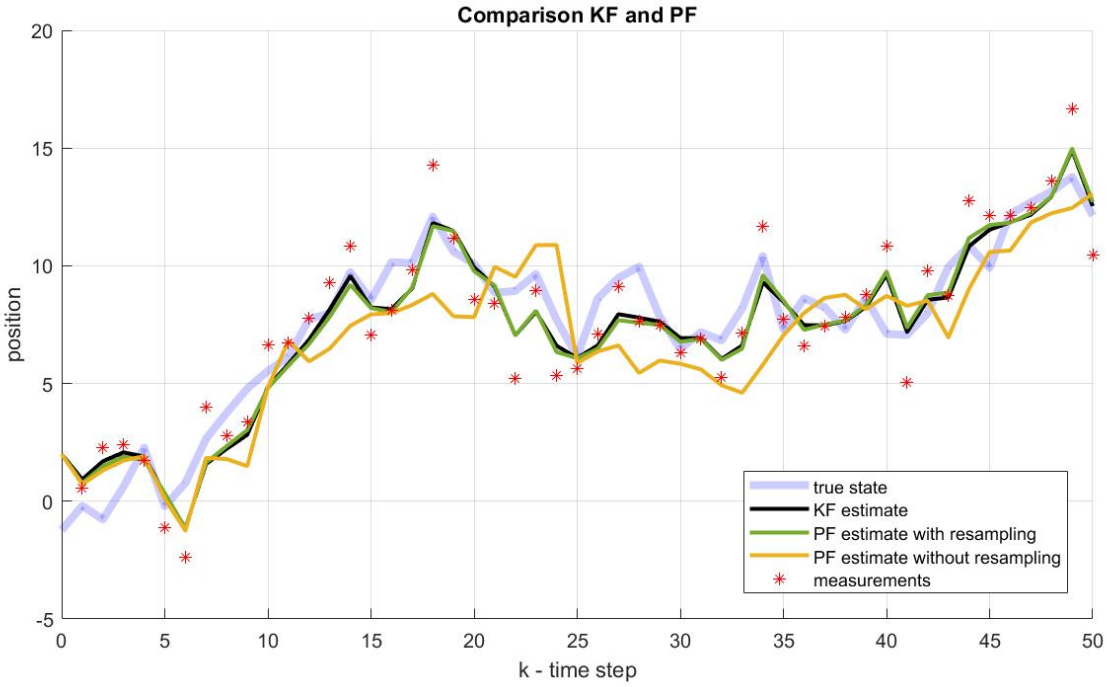
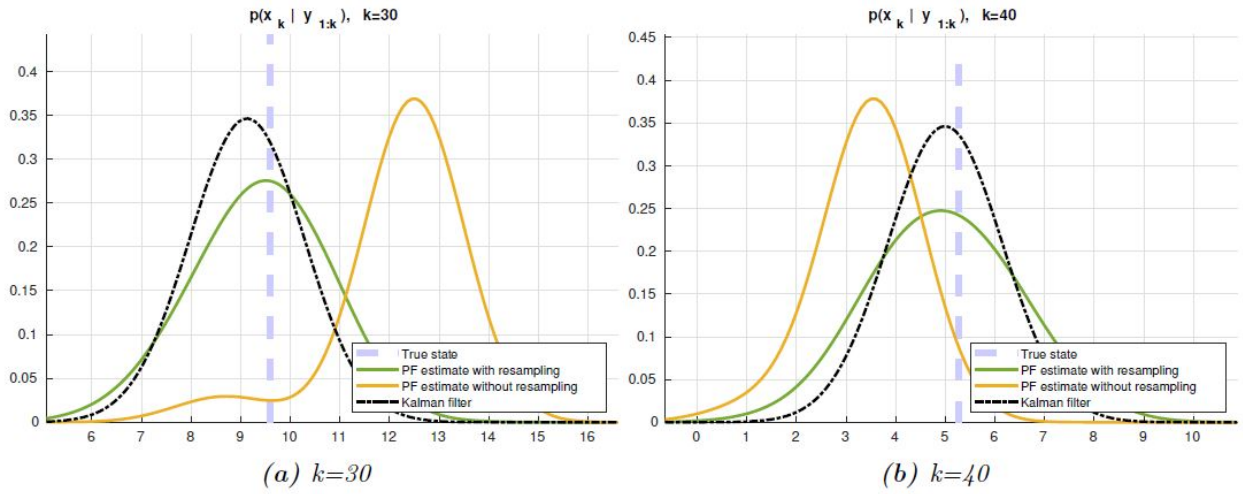


Figure 5: Comparison of particle filter with and without re sampling

plotted in Figure 6 for two time instants (particle filters densities were approximated as continuous). As expected the mean and covariance of the Kalman filter and PF with resampling are very similar and close to the true state, while the PF without resampling outputs estimates far from the true state.



(a) k=30 *(b) k=40*

Figure 6: Posterior density for kalman and particle filter at different time steps k

3.2 b and c

Figure 7 shows the true state and the particles trajectories for filters with and without resampling. To be able to analyze the weight of each particle, the same figure is plotted again, but now each particle is plotted with transparency proportional to its weight in comparison to other particles, see Figure 8.

Without resampling, the particles start to spread out in comparison to the true trajectory due to the uncertainty (covariance) in the motion model. After consecutive updates, the particle far from the true state and therefore likely to be far from the measurements are updated with a lower weight. After 20 time steps, the particles are far from the true state and only one single particle has a relevant weight. This degeneracy problem reduces a lot the performance of the filter.

On the other hand, one can see in both Figures 7 and 8, that resampling solves this degeneracy problem. The particles with low weight are more likely to be replaced by particles with high weight and at every time instant the weights are equally distributed. This result in particles that are always close to true state and represent well the true state probability.

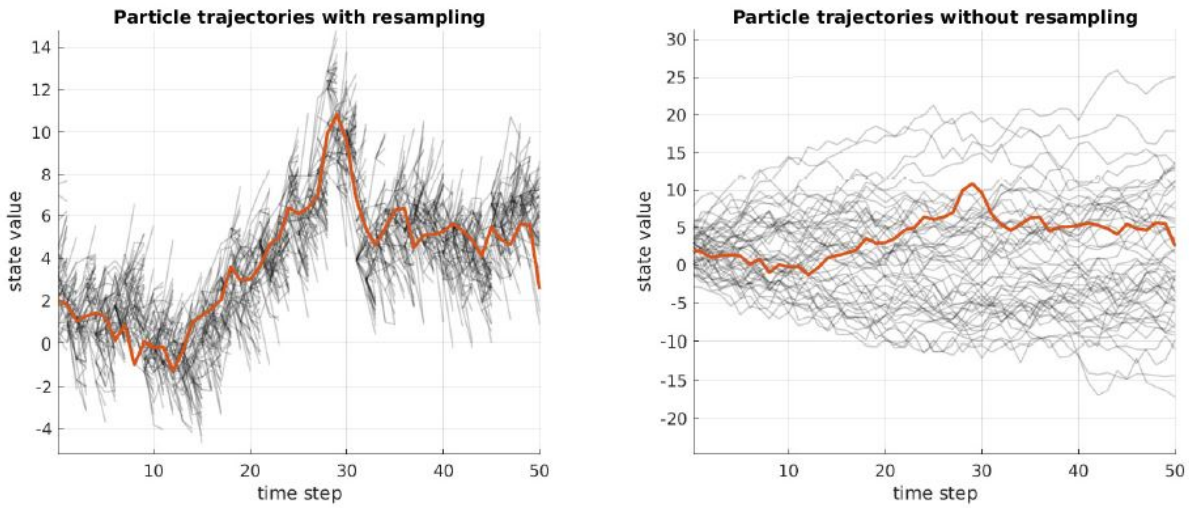


Figure 7: Comparison of particle filter with and without resampling (particles in black line) and true trajectory (orange line)

Comparison of particle filter with and without resampling (particles in black line) and true trajectory (orange line). Here, the the more opaque the color of the particle line, the larger its weight in comparison to other particles.

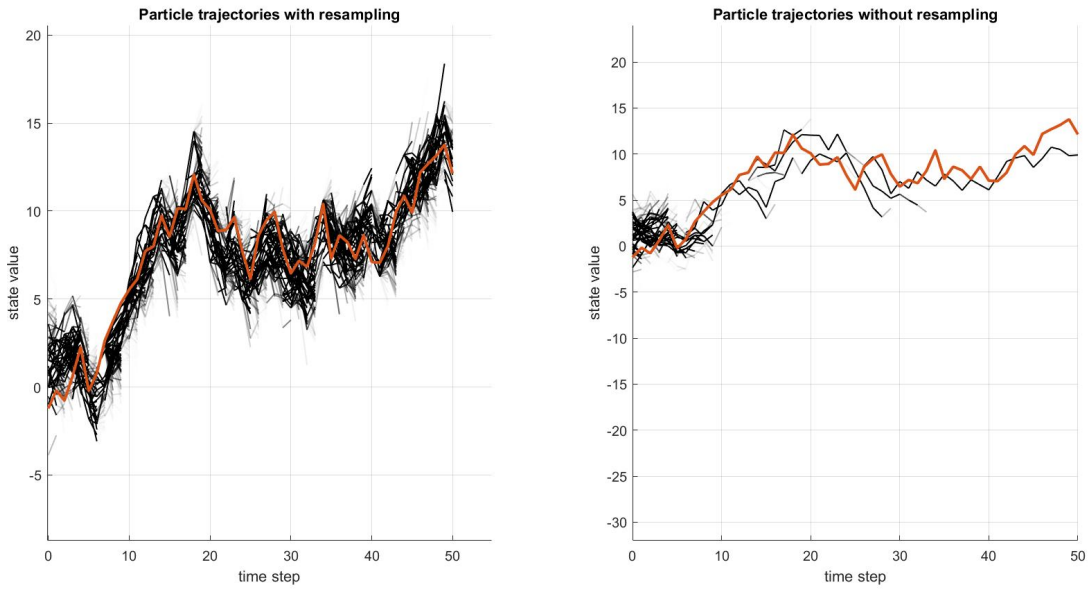


Figure 8: Comparison of particle filter with and without resampling (particles in black line) and true trajectory (orange line).

4 Bicycle tracking in a village

4.1 a

The true trajectory generated using the interactive GUI application is shown in Figure 9.

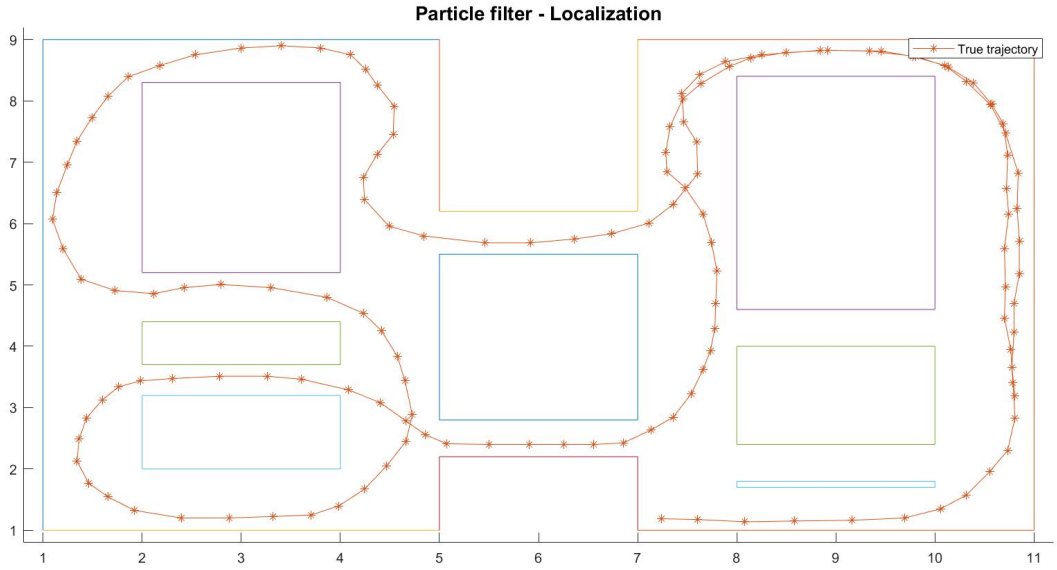


Figure 9: Generated true trajectory

4.2 b and c

First, it was decided to use a coordinated turn motion model because the true trajectory is smooth and the bicycle is most of the time turning with approximately the same yaw rate. Lets describe the usual CT motion model function as $p(x|x_{k-1}) = N(x_k; f(x_{k-1}), Q)$ which has the following states:

$$x = \begin{bmatrix} p_x \\ p_y \\ v \\ \psi \\ \omega \end{bmatrix} \tag{2}$$

It follows then that the measurement model is measuring a velocity vector:

$$y_k = h(x_k) + r_k = \begin{bmatrix} v_k cos(\phi_k \\ v_k sin(\phi_k) \end{bmatrix} + r_k \tag{3}$$

Further, in order to use the map information in the particle filter estimates, we can incorporate the map as a component of the motion model. Given the state at k-1, we know that there is zero probability of the vehicle to move and end-up inside a building. This information is very relevant and we can incorporate this in the motion model. Since we have knowledge of the map M, the new motion model can then be describe as $p(x_k|x_{k-1}, M)$, which can be decomposed using the Bayes rule:

$$p(x_k|x_{k-1}, M) \propto p(M|x_k).p(x_k|x_{k-1}) \tag{4}$$

where, $p(M|x_k)$ is the likelihood of the map being a road (not a building) given the prediction x_k and $p(x_k|x_{k-1})$ s the usual CT motion model. For a graphical representation the $p(x_k|x_{k-1}, M)$see Figure 10.

However, it follows that we can not use the motion model $p(x_k|x_{k-1}, M)$ as importance density because it is not Gaussian anymore and it would be very difficult to draw samples from it. The

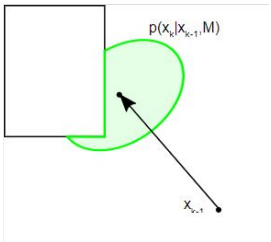


Figure 10: Non Gaussian motion model

solution is then to draw samples from $x_k^(i)$ $q(x_k|x_{k-1}, y_k) = p(x_k|x_{k-1})$which is Gaussian as before and compute the particle weights accordingly using previous equation

$$w_k^{(i)} \propto w_k^{(i-1)} \frac{p(y_k|x_k^{(i)}).p(x_k^{(i)}|x_{k-1}^{(i)}, M)}{q(x_k^{(i)}|x_{k-1}^{(i)}, y_k, M)} \propto w_k^{(i-1)} \frac{p(y_k|x_k^{(i)}).p(M|x_k^{(i)}.p(x_k^{(i)}|x_{k-1}^{(i)})}{p(x_k^{(i)}|x_{k-1}^{(i)})} \tag{5}$$

$$= w_k^{(i-1)}.p(y_k|x_k^{(i)}).p(M|x_k^{(i)}) \tag{6}$$

As a result, after sampling from $p(x_k|x_{k-1})$particles that are not on the road will have its weights set to zero because $p(M|x_k^(i)$ will be zero for those particles. After resampling, those particles will be replaced by another ones which are on the road.

4.3 d

Figure 11 shows the particles positions for some time step instants, assuming that the initial position is known.

4.4 e

Figure 11 shows the particles positions for some time step instants, assuming that the initial position is completely unknown. The trivial assumption to make is that there is equal probability of the bicycle being anywhere on the road. However, since the map is a non-linear distribution, we can use importance sampling, to sample from a uniform distribution that supports the map and then update the weights according to the map likelihood.

As can be seen in the Figure 11, in the first steps, many particles agglomerate, forming a multimodal distribution of the estimate. This is interesting and very reasonably. At time k=13, we can see that given the true trajectory traveled so far (orange line), all these agglomerates should have equal probability of representing the true position. This happens because we are measuring velocity and the velocity profile of the agglomerates centers are very similar to the true one.

With time, "fake" agglomerates will disappear. As can be seen at time k=46, there are two agglomerates with similar mass. In the next steps, the vehicle turns right, and the "fake" agglomerate will disappear after resampling due to the knowledge of the map that we have, see k=57. After around 60 time steps, the particles converge to good density and the estimate filter is able to estimate the true position with a good precision.

Orange line is the true trajectory; blue line is the estimated trajectory; purple dots represent the particles and the green ellipse represents the estimated covariance.

Figure 11: Posterior density of particle filter at different time steps k.