Solution to analysis in Home Assignment 4

Noa Lindén (noal)

Analysis

In this report I will present my independent analysis of the questions related to home assignment 1. I have discussed the solution with Carl Storckenfeldt, Gabriel Arslan Waltersson and Manish Suvarna but I swear that the analysis written here are my own.

Question 1

a

The true states, measured states, estimated states, smoothed states and their every fifth 3σ-contours are plotted in the figure below:

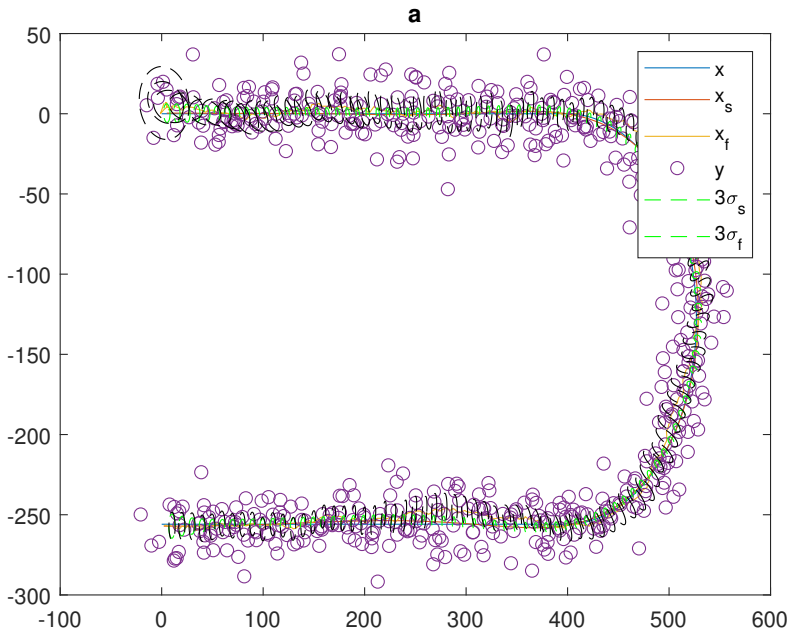

Figure 0.1: The true states, measured states, estimated states, smoothed states and their every fifth 3σ-contours.

Both the estimated and the smoothed states follow the true states fairly well. The smoothed values are closer to the true values though. This is expected since it uses more information. When estimating we only know past states, when smoothing we have knowledge about both past states and upcoming states. So when smoothing we know roughly were we are and where we're going to be. This allows for better "estimation", aka smoothing, of the current state.

The smoothed error covariances are smaller than their estimated counterparts. This is again due to having more information.

b

If one measurement deviates a lot this could be the result:

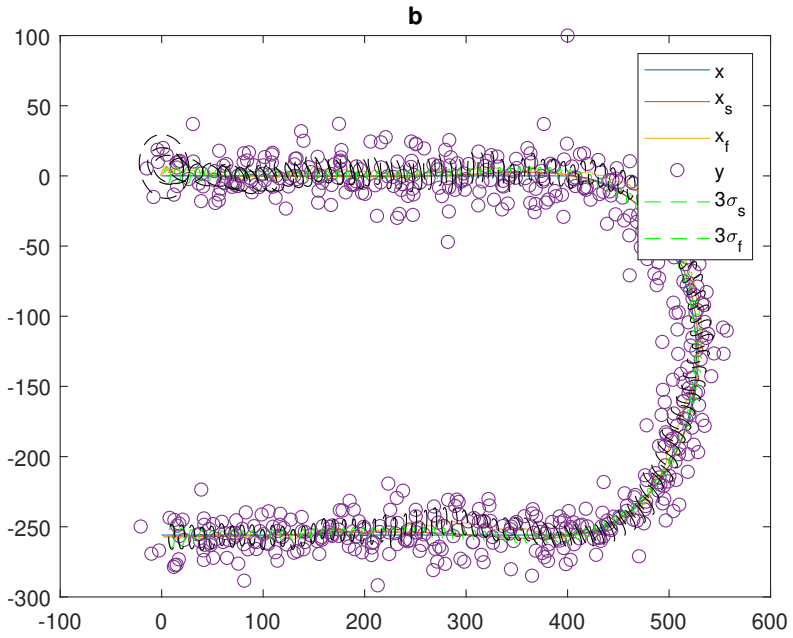

Figure 0.2: The true states, measured states, estimated states, smoothed states and their every fifth 3σ-contours if one measurement deviates a lot.

There is one measurement which deviates a lot. It is barely noticeable on the estimated and smoothed curves, however if one looks closely one can see that the estimated states deviates a bit more than usual from the true state. The smoothed state also deviates a bit more but is closer to the true state. The reason for the smoothed state being better is once again due to having more information. The smoother knows roughly where it's been and roughly where it's going. Given this and the motion it is highly unlikely that the state deviates as much as the measurement suggests. This leads to the smoother putting the smoothed state close to where it's been and where it's going. The estimated state follows roughly the same rules. It also knows roughly where it's been. So when it sees the measurement it also calculates this is as unlikely using the motion model and puts the estimated states closer to previous states.

Question 2

a

When resampling the pdfs produced but the particle filter are closer to the Kalman pdfs than when not resampling. This shows that resampling increases

the performance of the filter. With about 100 particles the resampled particle filter works approximately as well as the Kalman filter. This figure shows one of the worst cases for the particle filter:

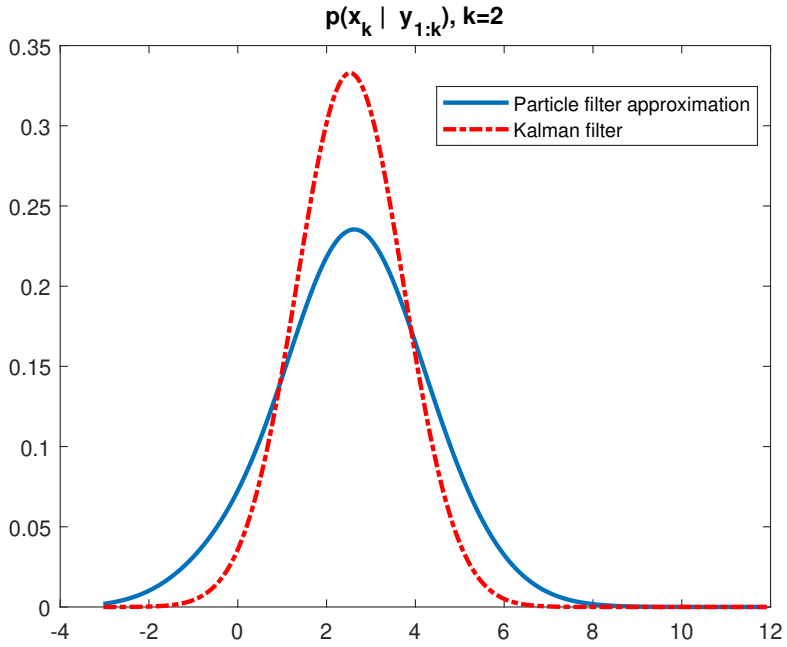

Figure 0.3: A bad resampled particle filter pdf.

The pdfs are sometimes closer, like this:

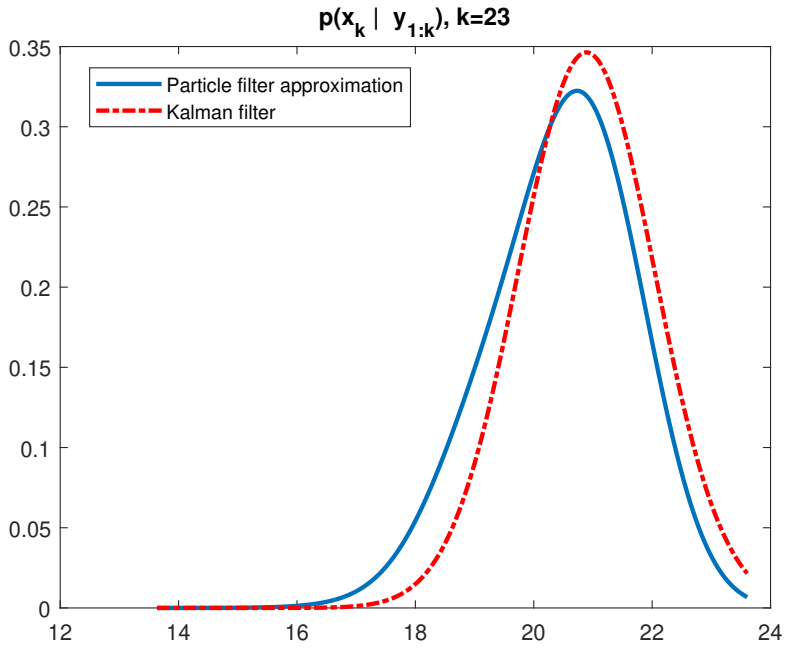

Figure 0.4: A good resampled particle filter pdf..

At the end of the 50 step simulation the particle filter pdf is still close to the Kalman pdf:

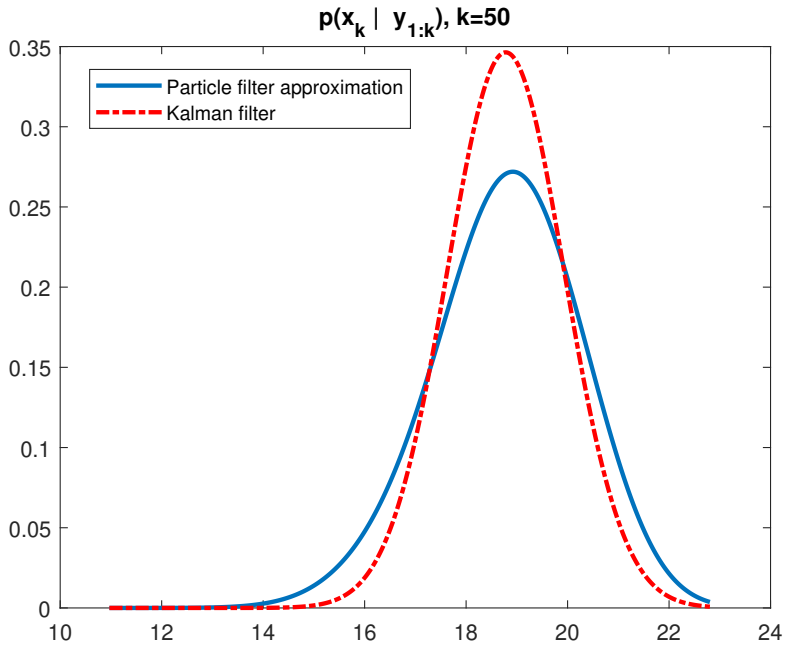

Figure 0.5: The last of the resampled particle filter pdf.

In the end the mean squared error for this 50 step simulation with 100 particles and motion model and measurement model as given in the problem is 2.2670.

The non resampled particle filter doesn't perform as well. Already after 5 steps in the simulation the pdf it produces deviates from the Kalman counterpart:

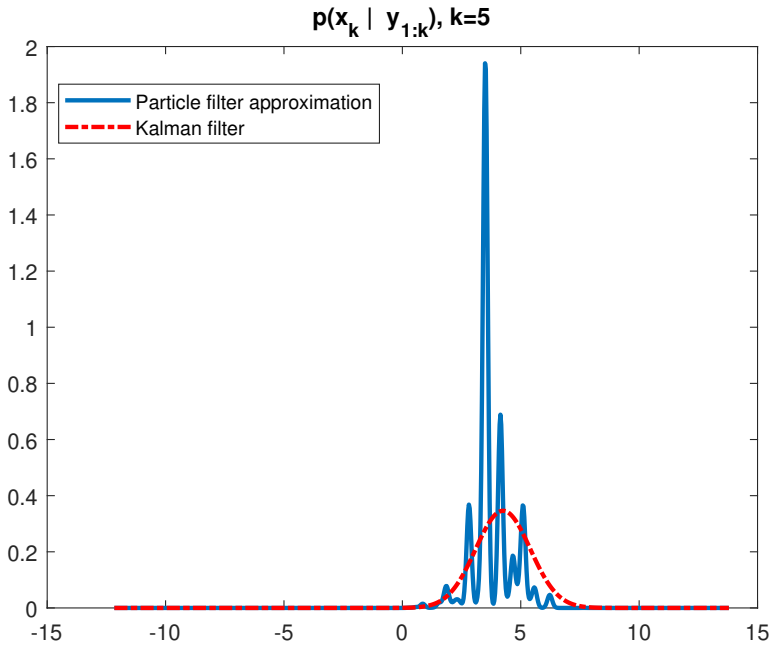

Figure 0.6: An early pdf produced by the non resampling particle filter.

Given enough time the non resampling particle filter will have only a few particles with weight making those values seem very plausible. Tendencies to this are seen here already in the spikes. The reason for this is that non of the particles or a few follow the true trajectory of the state. This makes it so that only those few have weights. Eventually non of the particles will follow the true trajectory but one will be close and as such get a higher weight. When normalizing the weights this weight will be close to 1.

A few more steps into the simulation the filter's pdfs are nonsense. This one shows the filter as being very sure of it's estimate despite it being wrong.

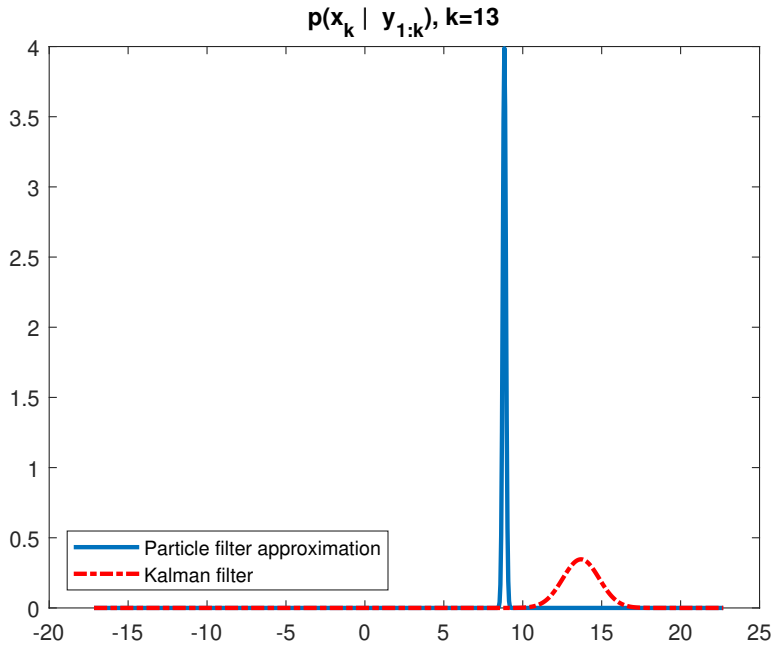

Figure 0.7: A later pdf produced by the non resampling particle filter which is off.

In the end the mean squared error for this 50 step simulation with 100 particles and motion model and measurement model as given in the problem is 15.7437.

b

This plot shows the evolution of the non resampled particles compared to the true state:

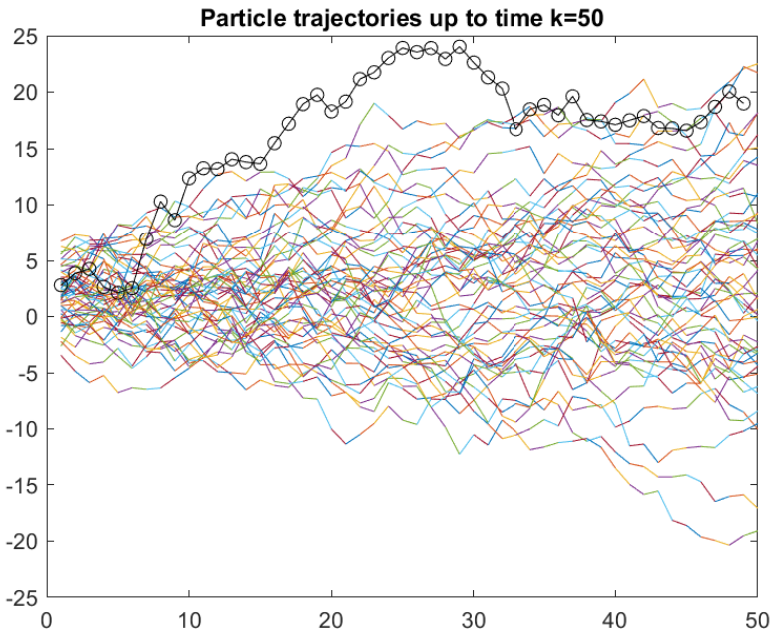

Figure 0.8: Evolution of the non resampled particles compared to the true state.

In the beginning many of the particles are fairly close to the true state. In the middle non are close and in end only a few are. This means that in the beginning the output of the filter will be good. In the middle however the output will not reflect reality as well. The value which gives the highest likelihood of the measurement, $\mathrm{p}(y_k \mid x_k^{(i)})$, will be the highest weighted. This likelihood might be small but it will be normalized to big. In the end the estimate will again be somewhat correct but the covaraince will be very small as there will only be a few particles with weight.

c

This plot shows the evolution of the resampled particles compared to the true state:

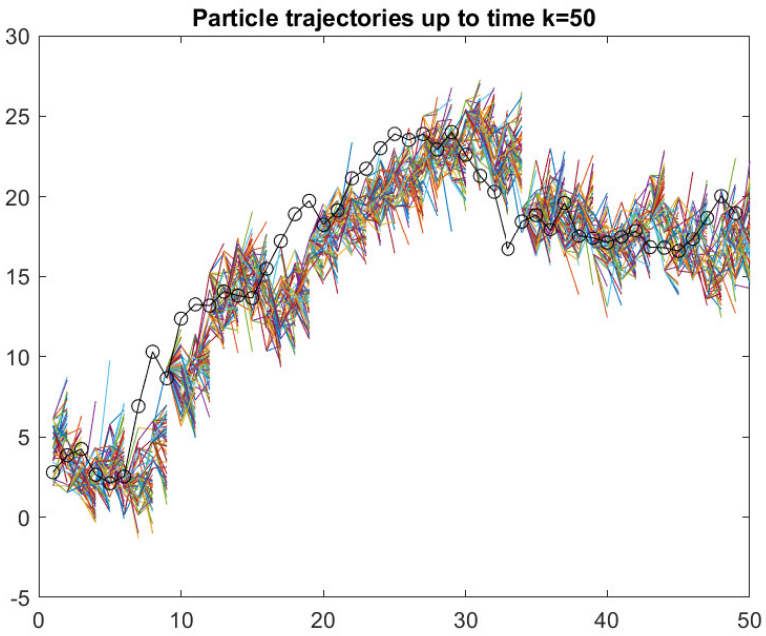

Figure 0.9: Evolution of the resampled particles compared to the true state.

Here the particles follow the true state all the way. This is because of the resampling. When performing an update step in the filter new particles are spawned out of the most likely old particles. Repeating this makes sure that improbable particles are discarded and only higher quality particles are used. This allows for a filter which works better over time. This both estimates the true state and the uncertainty better.

3

c

The map can be used when calculating weights. If a particle is inside a house or outside the map then it's corresponding weight is 0 since that's impossible. The equation used when updating the weights is this:

$$W_K^{(i)} \propto W_{K-1}^{(i)} p(y_k \mid x_k^{(i)}) \tag{1}$$

So by making p($y_k \mid x_k^{(i)}$) very low if $x_k^{(i)}$ is not on the road the map has been incorporated. This is easiest done by making y_k really big. The measurement model I used was h(x) = [x(3) + (1-isonroad(x(1),x(2))10^{10}, x(4) +

(1-isonroad(x(1),x(2))10^{10}]^T. This makes it so that if one predicted path is on a forbidden area the weight of that is close to 0. The same logic could be implemented in the motion model but that wouldn't be as easy in matlab as implementing it in the measurement model.

d

With this track:

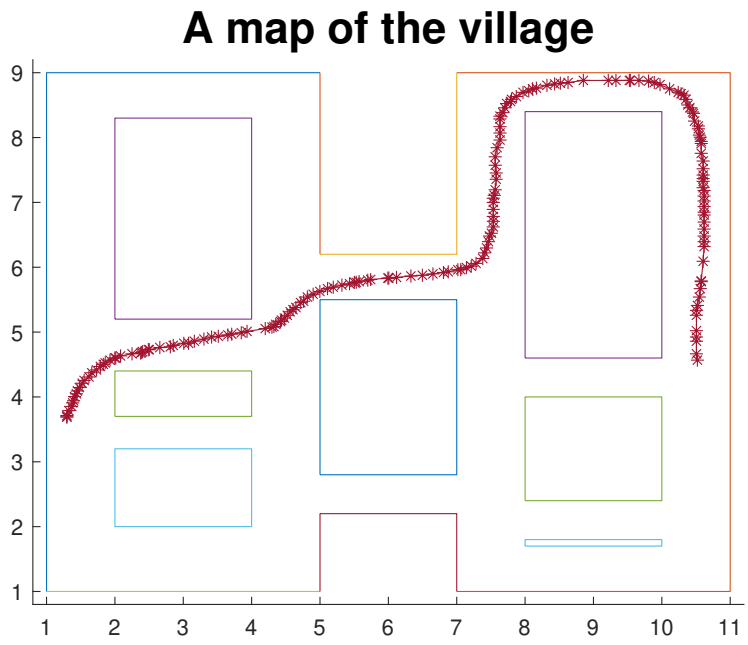

Figure 0.10: The track used.

And $Q = '\begin{pmatrix} 0.001 & 0 & 0 & 0 \\ 0 & 0.001 & 0 & 0 \\ 0 & 0 & 0.02 & 0 \\ 0 & 0 & 0 & 0.02 \end{pmatrix}'$, $R = '\begin{pmatrix} 5.0e-4 & 0 \\ 0 & 5.0e-4 \end{pmatrix}'$
and 2000 particles the following trajectory was estimated:

Figure 0.11: Estimated trajectory

The estimated trajectory follows the true trajectory well. This is could've been done even without the map but with the map some particles can be rejected and thus increasing the overall accuracy. As can be seen in this image some of the particles are in a forbidden area so their weight can be put to 0 which increases the accuracy:

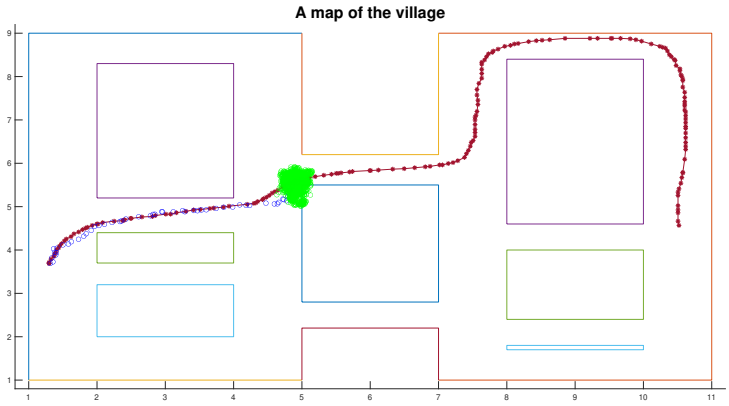

Figure 0.12: Example of when to use the map-information.

e

With this track:

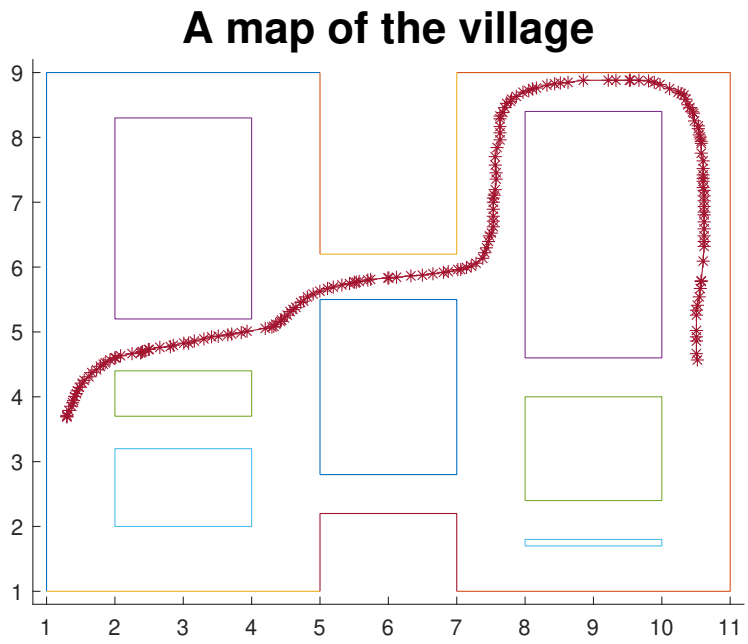

Figure 0.13: The track used.

And $Q = '\begin{pmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0.001 & 0 \\ 0 & 0 & 0 & 0.001 \end{pmatrix}'$, $R = '\begin{pmatrix} 5.0e-4 & 0 \\ 0 & 5.0e-4 \end{pmatrix}'$
and 5000 particles the following trajectory was estimated:

Figure 0.14: Estimated trajectory

In the beginning this is worse than when the start position was given but once the filter finds it's position the rest of the trajectory is as good. The information which allows it to locate it's position is the map, motion model and measurements. How the filter finds it's location I think is best described omitting the resampling. So if we start with 4 particles and a map:

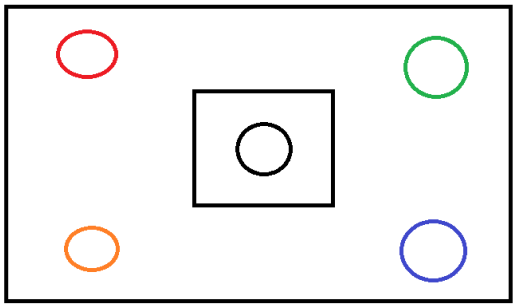

Figure 0.15: Estimated trajectory

Where the coloured circles are the particles, the black square in the middle a forbidden area, the black circle in the middle the estimated state and the black border the boundaries of the map. All the particles will be weighted the same and as such the estimated position is in the middle, even though it cannot be there. Once the velocities are measured and the particles are propagated through the motion model the following circles will be acquired:

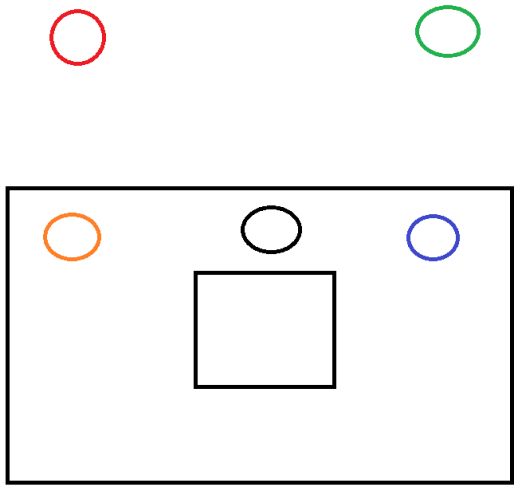

Figure 0.16: Estimated trajectory

Since two of the particles are outside of the boundaries their weight will be put to 0. Now only the two circles inside the boundaries have weight and the estimated state is the black circle. The velocities are again measured and propagated. Now the following circles are acquired:

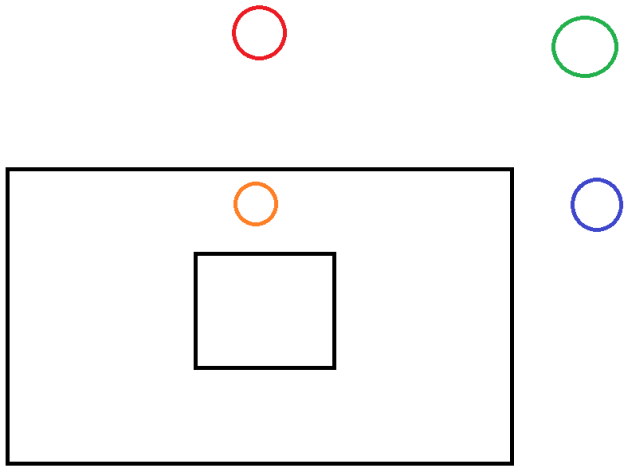

Figure 0.17: Estimated trajectory

Now only one particle is inside the boundaries so it will have all the weight.

And as such the estimated state overlaps with it. Now the filter has found the true state. Non if this would have been able to happen without the motion model, measurements or without the map. If resampling is introduced the filter finds it's way the same way but the particles with low weight will be discarded and new particles will be spawned out of the high probability particles.

This behaviour can be observed in the problem at hand as well. Here is an image of an early iteration when most of the particles are still spread around. Some of the particles are outside of the map and as such will be discarded:

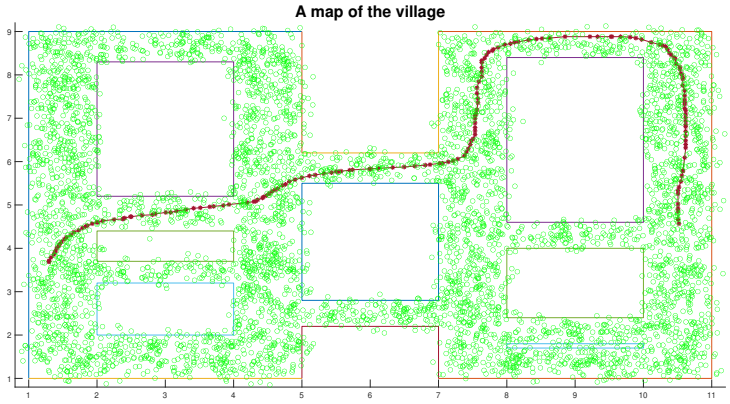

Figure 0.18: Particle filter has no idea where the true location is

Here the true location is still pretty much unknown.

This is at a later stage when most of the particles have been discarded and mainly two clusters remain:

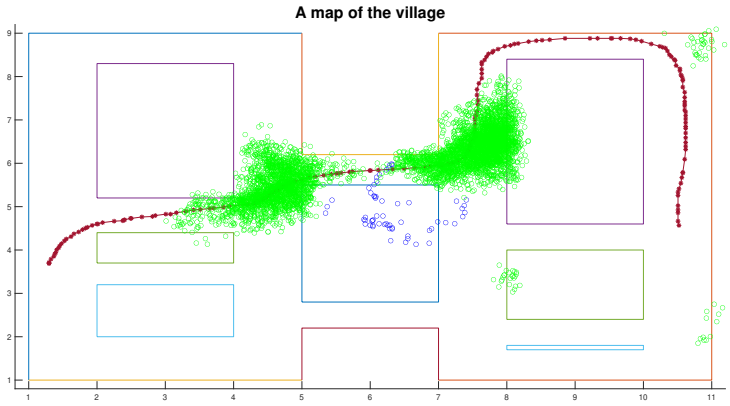

Figure 0.19: Particle filter has almost located the true position

Here the location is still unknown but basically only 2 contenders exist.