Lab 2
Applied Estimation EL2320

Addi Djikic – addi@kth.se – 941028-5473

December 2016

1 Part I, Preparatory Questions

1. It is the set of random state samples that are drawn of a posterior distribution and characterizes as: $\chi_t := x_t^{[1]}, x_t^{[2]}, ..., x_t^{[M]}$, where a particle is $x_t^{[m]}$. Vast number of particles are used to represent belief with a large group of particles.

2. From the particle filter algorithm, M number of particles are drawn from the set $\bar{\chi}_t$, and the probability of drawing such a particle is given as the importance weight, $w_t^{[m]}$, measurement on how likely it is. The weight represents the "power" and depends on the measurement update. The higher weight indicates a higher probability of our given system being in that state. Proposal distribution is the density distribution, $g(x^{[m]})$, represented by the particles. and target distribution is the probability distribution of a density function f, the true state of the particle. We estimate the target distribution by the proposal distribution. The relation between them is: $w_t^{[m]} = \frac{\text{target distribution}}{\text{proposal distribution}}$

3. It happens when random resampling occurs, and the danger could be that when series of random numbers occurs then they can "wipe out" the particles or all particles near its true state.

4. If we maintained the weight, then we would have particles in the area of interest where we do not need them to be represented due to the probability of the true state being in the area of interest is quite low. The resampling step is important because it forces particles back to the true posterior belief.

5. If the distributions has multiple peaks, then the problem with averaging will do that we are between the peaks and the state is not likely probable to exist there. The same occurrence would be if have several groups of particles.

6. We can use histograms, the idea of using histogram filters is to "split" the state space. We can ignore parts of the space where the is low probability. We can find the probability with the help of interpolation. We can also use Gaussian kernels and sum them up over all particles and look at the final distribution.

7. Sample variance occurs when random sampling happens at distributions, and if the variance is to large then the true belief will be bad. We can avoid this by for example increase the time between resamplings, or we can chose to use low variance sampling that does not sample independently but selects them using stochastic processes.

8. If the target distribution has a big spread then the pose uncertainty is large. This leads to many peaks varied throughout the distribution, and all need to be represented. To represent them all correctly we need large number of particles.

2 Part II

Q1. As one can see (6) is in the 2D state space model while (8) is the 3D while moving along a line. In (6) the angle is already defined and does not change. The 3D state in (8) will let us modify the angle θ so we can vary the angle when noise and errors are applied, it is more adaptive. But the 2D state is more simpler.

Q2. For this model, we use fixed velocity v_0 and fixed angular velocity ω_0. Thereof when we use this we model circular motions with a direction. We do not change the initial values even if noise comes along.

Q3. The purpose of the denominator constant is to normalize the expression with the Gaussian.

Q4. In multinomial-resample we generate numbers according to the number of particles, i.e. M random numbers generated. While in systematic-resample we only need to generate one.

Q5. We observe that the probability (in multinomial-resampling) of each particle being drawn is w, and the probability that the particle of not getting drawn from the total set M is $(1 - w)^M$, which then concludes that the probability that it survives is subtracted from one, giving that the survivability chance is $1-(1-w)^M$. The weight does not matter here it is the same for the interval $0 \leq w < \frac{1}{M}$.
In systematic-resample we break it down in steps. First case where $w = \frac{1}{M} + \epsilon$ the probability that it survives will actually be one. If we look at the algorithm, then as long as it is bigger than the CDF criteria it will always survive, which is the case because of the ϵ. But in the other case where $0 \leq w < \frac{1}{M}$ we can look at number of particles, the more we have the less chance of choosing particles will low weight, we can conclude that the survival chance is then wM.

Q6. As seen in the code, the measurement noise is modeled by **Sigma_Q** and the process noise by the variable **Sigma_R**.

Q7. We observe that after a while only one of the particles will survive. The one particle that survived was the one with the highest weight. We use fixed motion and after resampling all other particles converge to the last standing with highest weight.

Q8. Because we did not resample, the particles will be in the same random set that were set in the beginning. We will not have converges to the true state of the particles. They will behave according to the added process noise.

Q9. We observe that for small values set to the covariance matrix, the particles are being set as outliers because the particle clouds do not map to the true measurements, they do not converge towards the true state. When increasing the measurement noise one can observe that they will start to converge, once past a threshold then high numbers of the measurement noise will only contribute to the spread of the particles in the clouds.

Q10. For high values of the process noise we can observe that the spread of the particles becomes greater because of the increase in variance, and the convergence of particles will be fast towards true state. When applying low values to the process noise the diffusion is small, and we get a more narrow particle cloud. We do barely have any noise and that is why the particles are basically static and does not more around.

Q11. Our choice of process noise will depend highly on the motion model. For example, if we notice that the motion does not correspond to the true one, the simulated, then we have to increase the process noise to make up for the 'false' motions. If it is correct we can use lower values.

Q12. As mentioned in Q11 if the model is not correct then increasing of the process noise is needed, this will however lead to increase of number of particles so we can locate the true state in the particle clouds. If the motion is correct then it will be opposite, i.e. we need fewer particles.

Q13. One solution is that we can set some kind of boundary or 'threshold' for when to determine when an outlier is detected. We can look at the mean/average Likelihood and check if it is quite low, and then determine that it is an outlier.

Q14. Different values were tested when using the second circular model with the three different motions, *fixed*, *circular* and *linear*.

The best result I could achieve when using *fixed* model was an estimate error of 12.6 ± 5.3, the parameters for measurement noise was $diag([300\ 300])$ and $5 * diag([2\ 2\ 0.01])$ for process noise.

For the *linear* motion best result was 8.2 ± 4.3 with $Q = diag([300\ 300])$ and $R = 3 * diag([2\ 2\ 0.01])$.

For *circular* motion the results were 7.7 ± 4.0 and the parameters were $Q = diag([300\ 300])$ and $R = 2 * diag([2\ 2\ 0.01])$.

As mentioned before, we can conclude that when having good motion model, a lower process noise can be used. We see that the *fixed* motion demands a higher process noise, it is more sensitive in some case. While the *linear* and *circular* motion can use lower process noise.

Q15. As we can conclude, there are two parameters that affect the outlier detection, that is λ_Ψ and Q, where the λ_Ψ acts as a threshold. If we model a weak measurement noise, i.e. $Q \to 0$ then outliers will be detected. We will reject all the measurements that are not exactly as the true measurement.

Q16. It will result in 'bad' weights, the system will consider the measurements as valid, and later be included in the Likelihood calculation which will give worse state representation.

3 Datasets

3.1 1:st Dataset, Four Landmarks

The first map that we use, contains four landmarks places that are symmetric with each other. We need to increase the variable 'part_bound' because of the reason it corresponds to the spread of particles in order to cover the positions around the four landmarks and the four valid hypotheses that we need (that are spread out).

We start by running the simulation with only 1000 particles, we can observe after some test runs that it can hardly retain all the hypothesis reliable. One of the test runs can be seen in figure 1 (where it located a cloud), and in figure 2 after the finished run.

When we increase the particle number to 10000 we can obtain much better results, especially retaining the hypotheses in our runs. The reason is that when using a grater number of particles, we spread them out and have particles at more places. Which concludes that we have less chance of particle deviation ('wipe out') as we mentioned before.

The results with 10000 particles can be seen from figure 3 to 6, where in figure 3 and example of the four clouds that were created from the particles are visualized. In figure 4 we observ that our estimator found the best cloud to track. Figure 5 shows how it has converged to one cloud and it still tracks it. Figure 6 shows the finished run of the 10000 particles.

When using multinomial-resampling it does not preserve multiple hypotheses that good. The particles are chosen differently depending on the weights.
We also observe that when varying the measurement noise to higher values we have a bigger spread with the particles, but we seem to retain the hypothesis better to some amount. However when we use lower values we essentially see that the multiple hypotheses are not retained.

3.2 2:nd Dataset, Five Landmarks

For this dataset we used a map with 5 landmarks, we do not have a symmetric map anymore. We can follow the simulation with 10000 particles in figure 7 to 10. In figure 7 we see the five clouds that are spread out. In figure 8 we track the best cloud and we can see the other ones are starting to converge. In figure 9 we can see that the clouds after $t = 180$ have converged and we only track one now as expected. In figure 10 we show the complete run and display the mean errors.

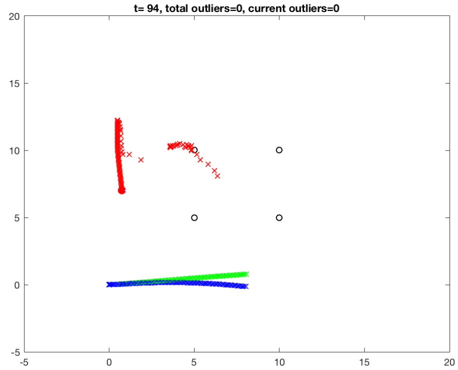

Figure 1: 1000 particles, tracked cloud in upper landmark.

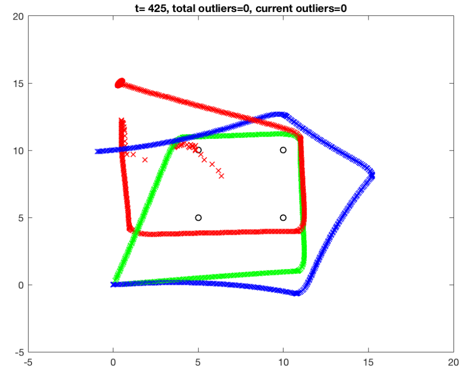

Figure 2: 1000 particles, finished process of tracked cloud.

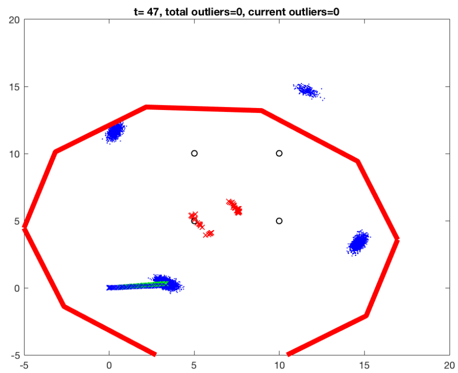

Figure 3: 10000 particles, to visualize clearly the four clouds that converged.

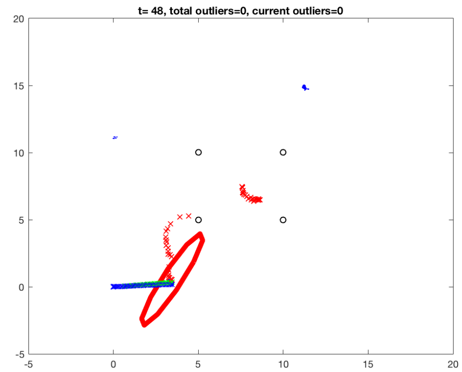

Figure 4: 10000 particles, found best cloud (red ring) to track.

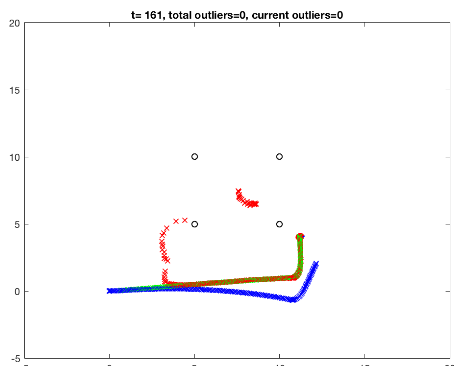

Figure 5: 10000 particles, still follows the best cloud.

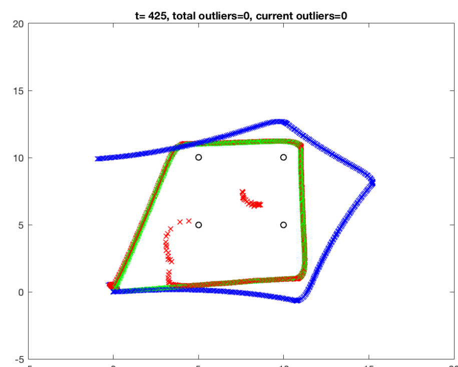

Figure 6: 10000 particles, finished process.

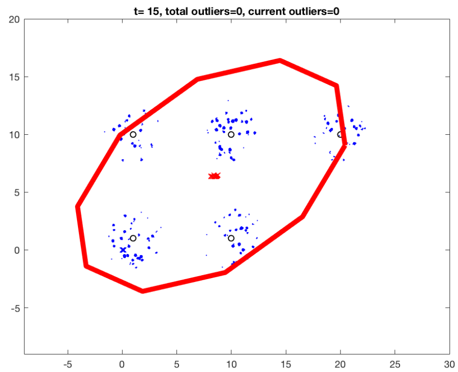

Figure 7: 10000 particles, we observe the five clouds created.

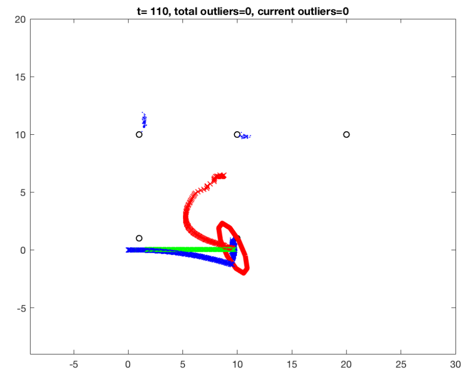

Figure 8: 10000 particles, our estimator chose the best cloud to track.

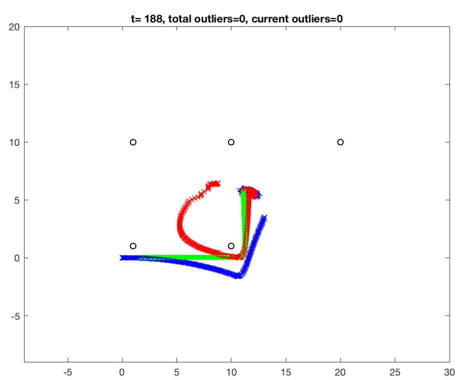

Figure 9: 10000 particles, after t = 180, we can observe that the other clouds converged as expected.

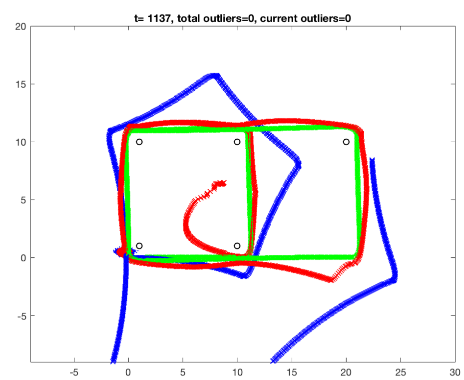

Figure 10: 10000 particles, the finished simulation with the five landmarks with mean absolute errors of [0.489 0.600 0.173]