Laboration 2

Part 1

1. The particles are random samples drawn from a distribution.
2. If we define f(x)=target distribution (the true posteriori) and g(x)=proposal distribution (our sampled distribution), then the importance weights w=f(x)/g(x). The importance weights are needed to adjust for the difference between our sampled distribution and the true posteriori.

   $$Importance\ weights\ w_t^{(m)} = p(z_t|\hat{x}_t^{(m)})$$
3. Even if sample variance is made small, repeated resampling will eventually cause this error to grow. Particle deprivation means that there are no particles in places that are important because of random or because that the number of particles is too small. Adding random particles might help, but not that well. In practice this occurs when the size of the particle set is small relative to the space of all states with high likelihood. The danger is that one might get the wrong estimation, e.g. the robot think it's in another spot than it actually is, or the temperature that are estimated shows the wrong value.
4. The system is dynamic and to represent the current state correctly we need to update the distribution. If we don't resample we need more particles, which leads to more computations. Also, the weights of some particles would get close to zero and other weights would get very large. We resample to reduce the particles where we probably not are.
5. When the posteriori can't be represented by estimation using maximum one peak, i.e. multi-modal situations.
6. By using density extraction. One method is to fit a Gaussian to the mean and variance of the particle, for example. To represent the whole space, we get a distribution by performing a density extraction.
7. The two sampling steps in the particle filter result in the particle distribution being close but not identical to the posteriori. The difference is called sample variance and we can reduce this error by using more particles or use the method of low variance sampling, for example.
8. Spread out distributions get more particles than tight ones.

Part 2

1. Advantages for 3D state-space is that $\bar{\boldsymbol{u}}_{\boldsymbol{t}}$ is modelled as it depends on ϑ, instead of just a constant as in 2D state-space. But in this case we don't measure ϑ, so 6 and 8 does the same

thing right know. But 6 is less dimension than 8 which means that we need less particles which means that the code will run faster for 6 than 8.

2. We can model circles, not ellipses. This is because v_0 and w_0 are constants and the limitations is that we need to know them.
3. It will affect how large/small the density gets. And the sum of all the possible probabilities should be one.
4. In the multinomial resampling method there is a need of as many random as there are particles. In the systematic resampling method there is only a need of one random.
5. The probability of a particle to survive the resampling is the same as 1-"the probability of not surviving". In vanilla resampling, the probability is 1-(1-w)^M which means that there is a chance that a particle with heavy weight won't be chosen. In the systematic resampling the probability is w*M which means that if the weight of the particle is 1/M or greater it will be chosen.
6. The variables modelling the measurement noise and the process noise are Sigma_Q and Sigma_R respectively (they are kept in the structor params).
7. The particles won't be spread out. All the particles will be on the same location.
8. The particles won't be able to focus around the point where we want to be estimating. This is because when we resample, we pick new particles depending on the weight from the particles in an earlier state and they will be placed where there was particles before, but it's more likely that they will end up on a spot where a particle with a higher weight was. When we don't resample, this won't happen.
9. When we decrease the measurement noise, we will decrease the area of where we want the particle to be. That means that the particles will be all over the room until one particle "accidently" is on the location where the area is, and because of the resample all particles will get there. But when we decrease even more we decrease that area, which means that the chances of a particle spot that area is smaller than before. If we increase the measurement noise, we increase the area where we want the particles to be. As a conclusion you can say that changing the measurement noise means changing the area of where the particles can be.
10. The smaller the process noise, the smaller spread the particles have from its modelled position. If increasing the process noise, the particles will deviate from its modelled position.
11. The more the motion model is similar to how the robot is moving, the lower value of the process noise we can choose.

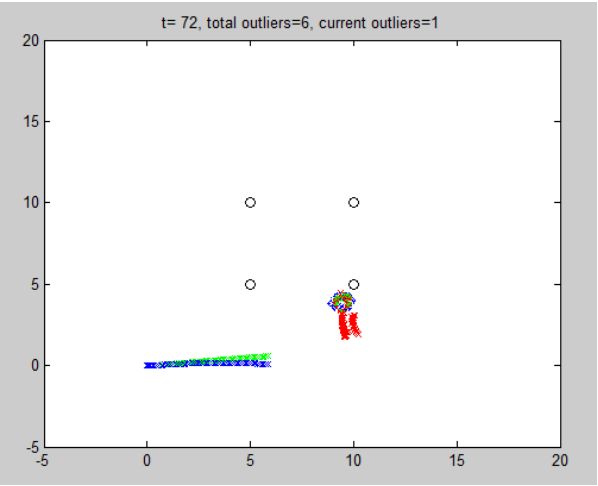
12. If the motion model is chosen in a way that doesn't represent the motion that the robot actually will have, the accuracy will get lower. So if the motion model is good, then the number of particles will be lower because the area of where the robot can be gets smaller and therefore we need less particles to cover that area.
13. A threshold can be used to define an outlier. For every measurement that has the average of all weights smaller than that threshold is an outlier, i.e. with the help of the weights, we can see if our measurement is an outlier. That is because if the weights are low, but the particles are in the correct area (because the first measurement wasn't incorrect), this must mean that just that measurement is an outlier.
14. When modelling a circular motion, the best precision is when multiplying Q with 3. Then the absolute error for the estimates gets 7.2 +- 3.6. When modelling a linear motion, the best precision is when multiplying Q with 5. Then the absolute error for the estimates gets 7.8 +- 3.7. When modelling a fixed motion, the best precision is when multiplying Q with 7 and R with 20. Then the absolute error for the estimates gets 11.2 +- 5.6. So the circular motion is the most sensitive, then linear and the least sensitive is the fixed motion.
15. The parameters in Psi are Q, R, delta_t and also the threshold affects the outlier detection. If |Q| goes to zero means that the area of where particles can be gets smaller (to zero) and then the outlier detection gets wrong because basically everything will be outliers then.
16. The weight gets a wrong value if there are undetected outliers.

MATLAB

Dataset 1

Part_bound says how large area the particles can be placed random out on. 10 is enough to random out over the whole map.

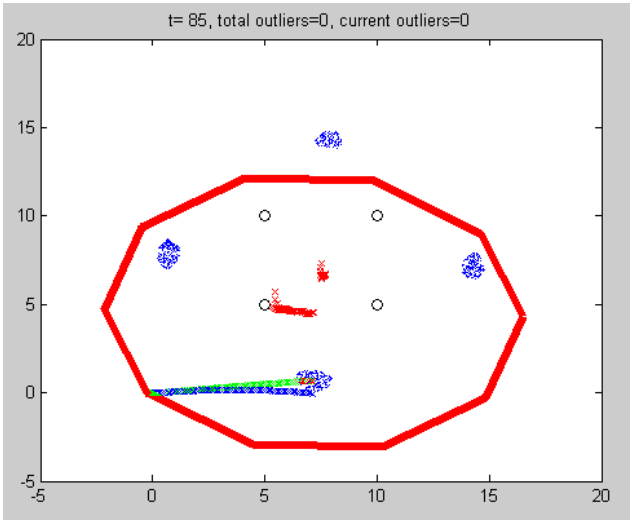
1000 particles – Global localization




```
mean error(x, y, theta)=(-2.236289, -0.322120, -1.286101)
mean absolute error=(4.837285, 3.610116, 1.286101)
total_time =66.501812
>>
```

Figure 1 With 1000 particles the hypothesis quickly becomes one

Global localization – Systematic resampling

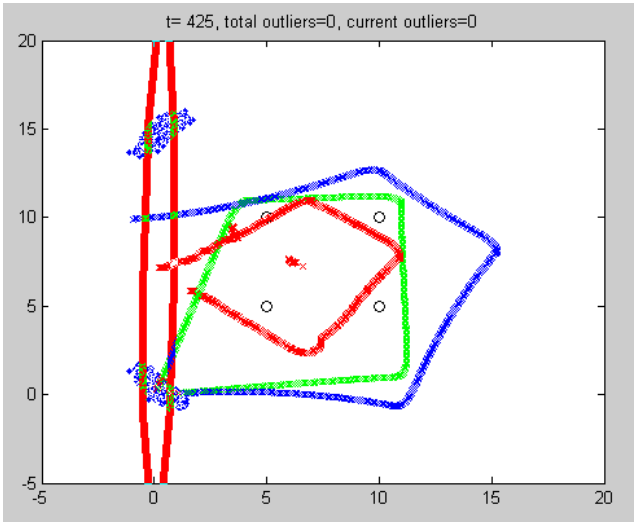



```
>> runlocalization_MCL('so_sym2_nk.txt','map_sym2.txt',1,1,1,[],2)

mean error(x, y, theta)=(-0.793345, -2.821583, 1.646409)
mean absolute error=(4.045065, 5.037550, 1.646409)
total_time =813.146435
```

Figure 2 With 10 000 particles and use of systematic resampling, there will be 4 hypothesis since we have 4 landmarks symmetricly

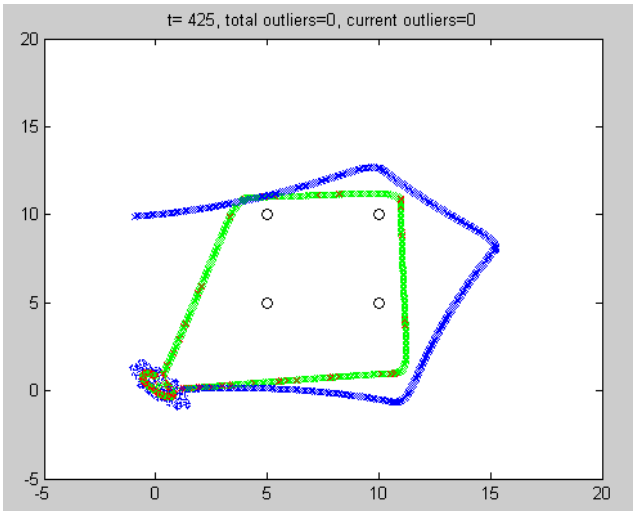
Global localization – Vanilla resampling




```
>> runlocalization_MCL('so_sym2_nk.txt','map_sym2.txt',1,1,1,[],2)
mean error(x, y, theta)=(0.040129, -1.831752, 0.688066)
mean absolute error=(2.265067, 2.972778, 0.688066)
total_time =497.846399
```

Figure 3 With vanilla resampling there won't be 4 hypothesis

Tracking – Systematic resampling




```
mean error(x, y, theta)=(0.001229, -0.001623, 0.003951)
mean absolute error=(0.003957, 0.004615, 0.004450)
total_time =596.080003
>>
```

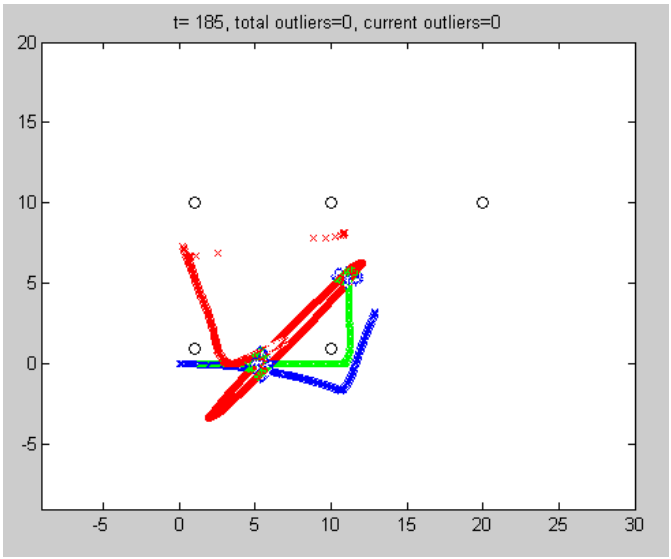
Figure 4 Tracking with systematic resampling works fine

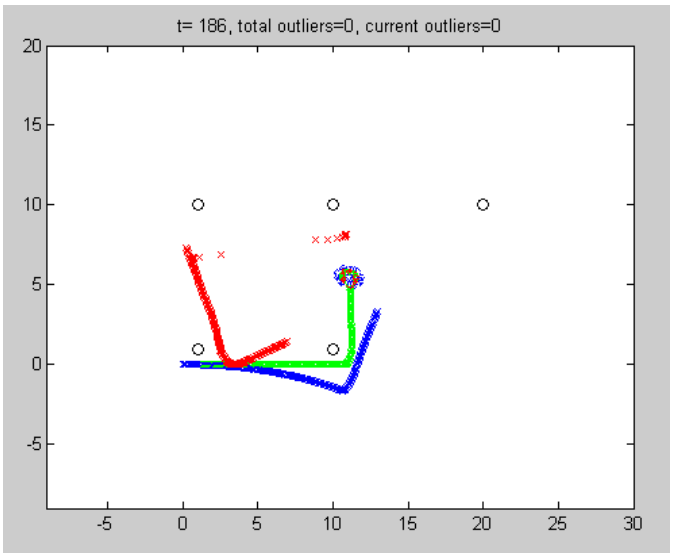
Filter sensitivity

To increase the noise doesn't change that much, but to decrease the noise makes the hypothesis disappear.

Dataset 2

Global localization – Systematic resampling

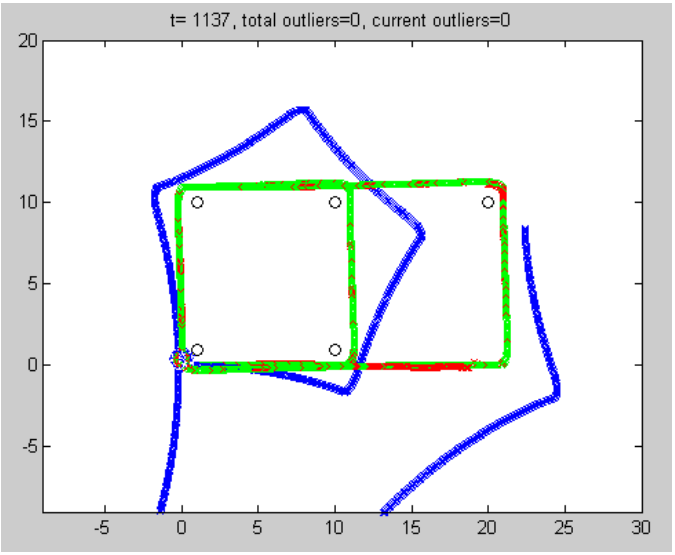





```
>> runlocalization_MCL('so_sym3_nk.txt','map_sym3.txt',1,1,1,[],2)
mean error(x, y, theta)=(0.027379, -0.808538, -0.170940)
mean absolute error=(0.586721, 0.835614, 0.189276)
total_time =1367.819840
```

Figure 5 Here can we see when the hypothesis converge to the correct one

Tracking – Systematic resampling




```
>> runlocalization_MCL('so_sym3_nk.txt','map_sym3.txt',1,1,1,[0;0;0],2)
mean error(x, y, theta)=(0.013489, 0.006506, 0.008340)
mean absolute error=(0.020948, 0.017285, 0.008764)
total_time =1346.363791
```

Figure 6 Tracking with systematic resampling works fine