EL2320 - Applied Estimation - Lab 1

Extended Kalman Filtering (EKF)

PART II

Alexandra Hotti

October 2019

1 Warm-up problem: Linear Kalman filter

• Question 1:

What are the dimensions of ε_k and δ_k? What parameters do you need to define in order to uniquely characterize a white Gaussian?

Answer: As ε_k is the state transition noise its dimensionality is equal to the dimensionality of the state x_t. In this assignment the state has two parameters, one for the cars position and on for its velocity. Therefore, the dimensionality of ε_k is: 2x1.

Next, δ_k is the measurement noise. Therefore, its dimensionality should correspond to that of z_k meaning the number of measurements. In this particular example we only measure the position of the car and thereby the dimensions of δ_k is 1x1.

A Gaussian in 1 dimension is defined by its standard deviation σ and mean μ. A white Gaussian has a mean value of $\mu = 0$. Thus, the white Gaussian in 1 dimension is defined as: $\mathcal{N}(0, \sigma^2)$. When the dimensionality increases the white Gaussian is instead defined by a vector with one mean value for each dimension $\bar{\mu}$, which again is equal to zero in each dimension in the case of a white Gaussian. The spread of the Gaussian in several dimensions is described by a covariance matrix Σ.

• Question 2:

Make a table showing the roles/usages of the variables (x, xhat, P, G, D, Q, R, wStdP, wStdV, vStd, u, PP). To do this one must go beyond simply reading the comments in the code. Hint: Some of these variables are part of our estimation model and some are for simulating the car motion.

Answer: See table 1.

Variable	Role / Usage
x	The cars true state of size 2x1 containing the position and velocity.
xhat	The estimated state of the car by the Kalman Filter with dimensions 2x1.
P	The covariance matrix used first to describe the uncertainty of the prediction phase $\bar{\Sigma}_t$ and later the uncertainty of the Kalman Filter estimate Σ_t after the measurement is included. Its dimensionality is 2x2.
G	A 2x2 Identity matrix used to get the correct matrix dimensions for the state transition process noise: ε_k.
D	Used to get the correct dimensions for the measurement noise. Here it is simply a 1x1 scalar.
Q	The variance of the measurement noise, with dimensions 1x1.
R	The process noise covariance matrix with dimensions 2x2.
wStdP	The actual σ^2 for the normally distributed zero mean noise of the simulated state position.
wStdV	The actual σ^2 for the normally distributed zero mean noise of the simulated state velocity.
vStd	The standard deviation of the actual measurement noise drawn from a normal distribution with mean zero.
u	The control action Incorporated during the prediction phase. Here it makes the car accelerate.
PP	Used to store the covariance matricies in vector forms over the interval: $t_0, .., t_n$.

Table 1: The roles of the variables in the matlab script $kf\_car.m$

• Question 3:

Please answer this question with one paragraph of text that summarizes broadly what you learn/deduce from changing the parameters in the code as described below. Choose two illustrative sets of plots to include as demonstration. What do you expect if you increase/decrease the covariance matrix of the modeled (not the actually simulated) process noise/measurement noise by a factor of 100 (only one change per run)? Characterize your expectations. Confirm your expectations using the code (save the corresponding figures so you can analyze them in your report). Do the same analysis for the case of increasing/decreasing both parameters by the same factor at the same time. Hint: It is the mean and covariance behavior over time that we are asking about.

Answer:

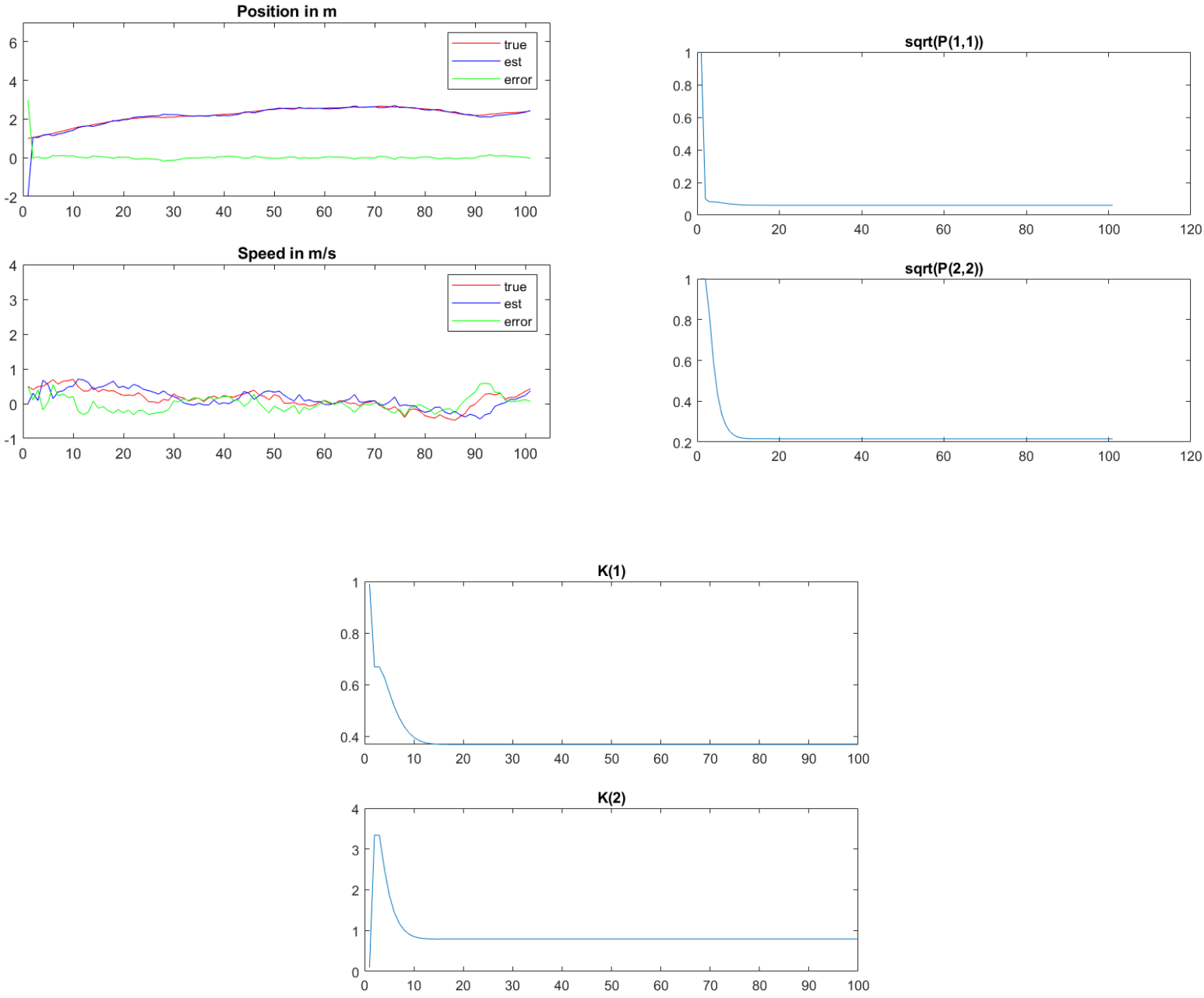


Figure 1: Default setting for the the transition R_t and measurement noise: Q_t

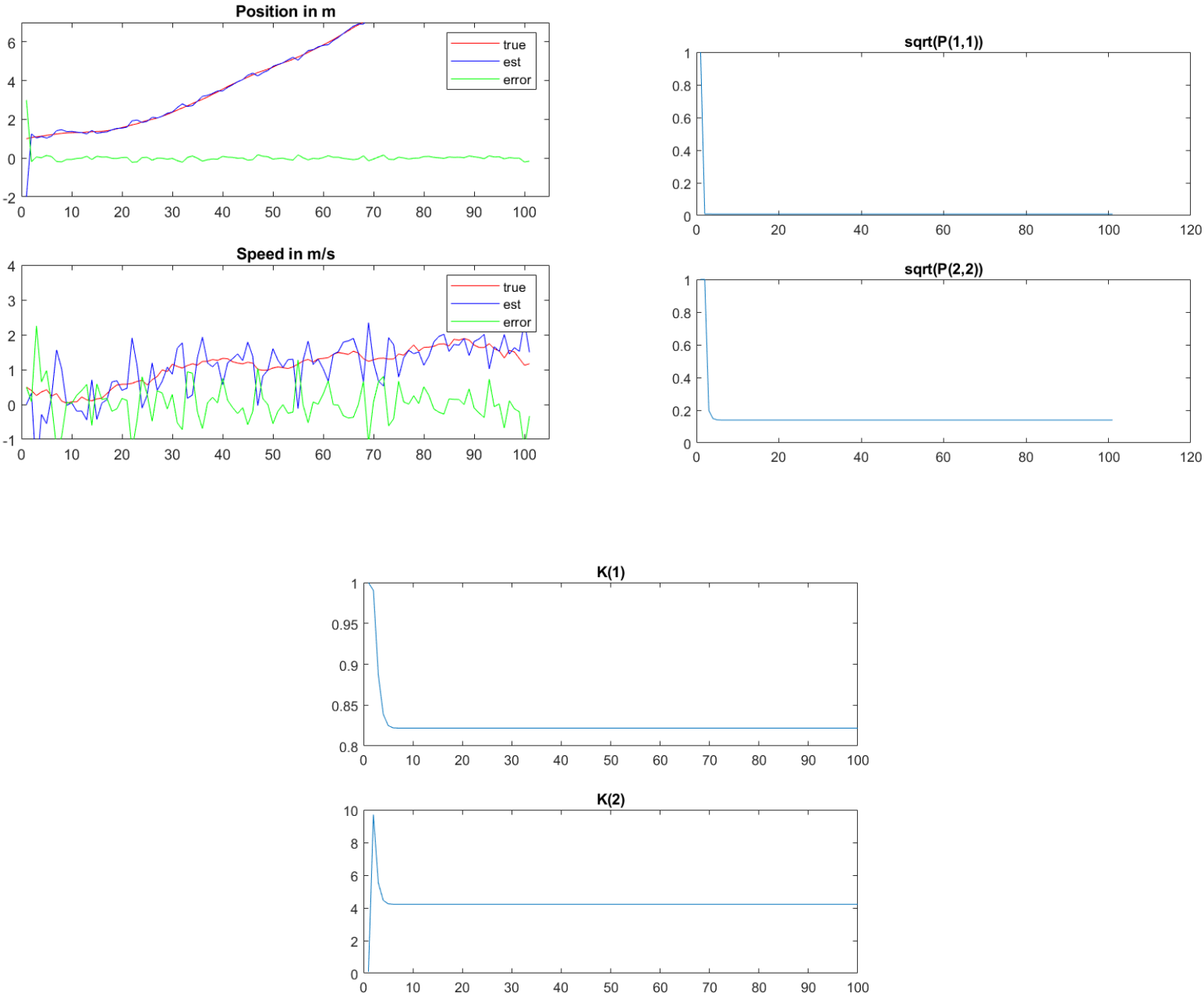


Figure 2: Decreasing the measurement noise: $Q_t \cdot \frac{1}{100}$

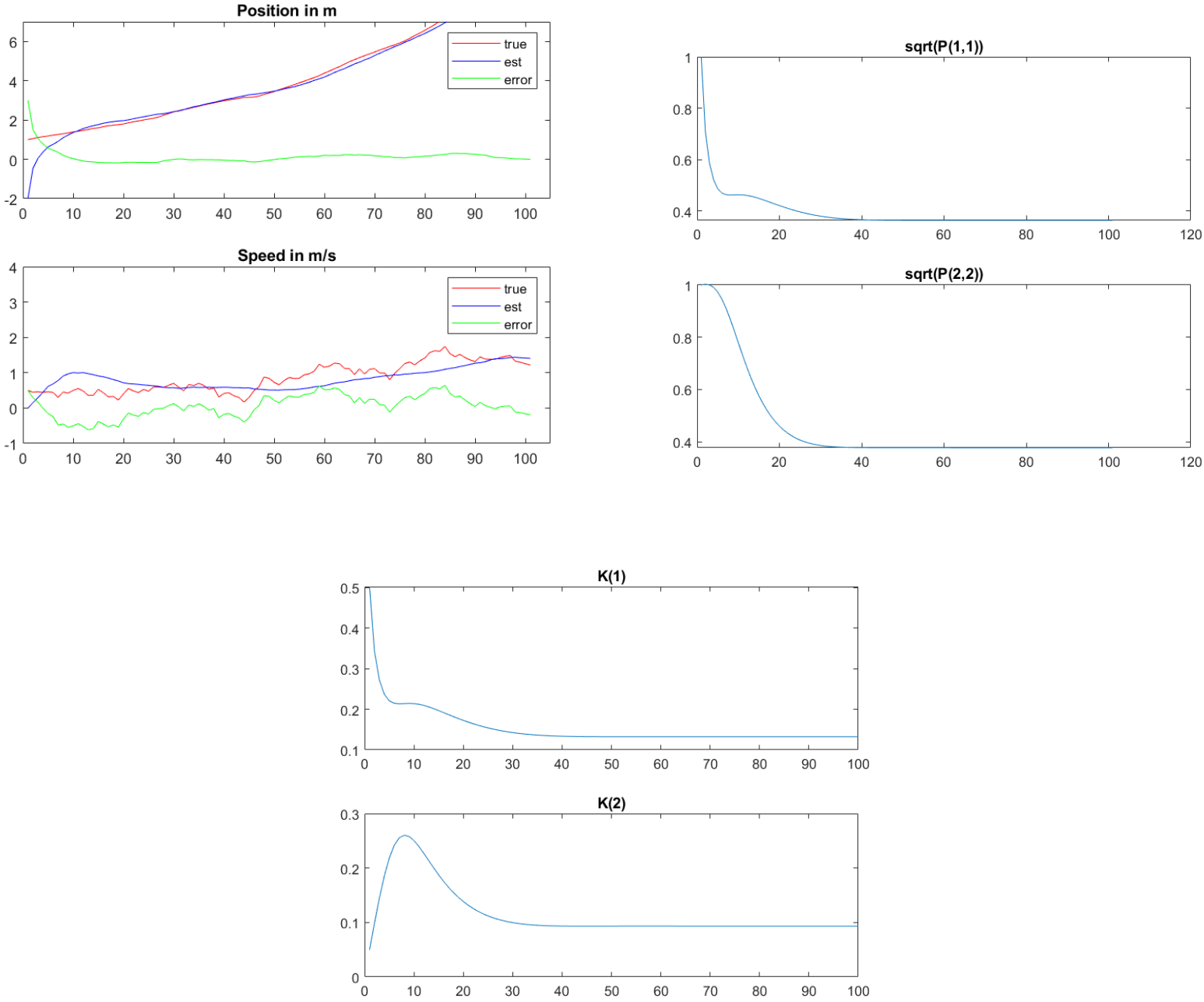


Figure 3: Increasing the measurement noise: $Q_t \cdot 100$

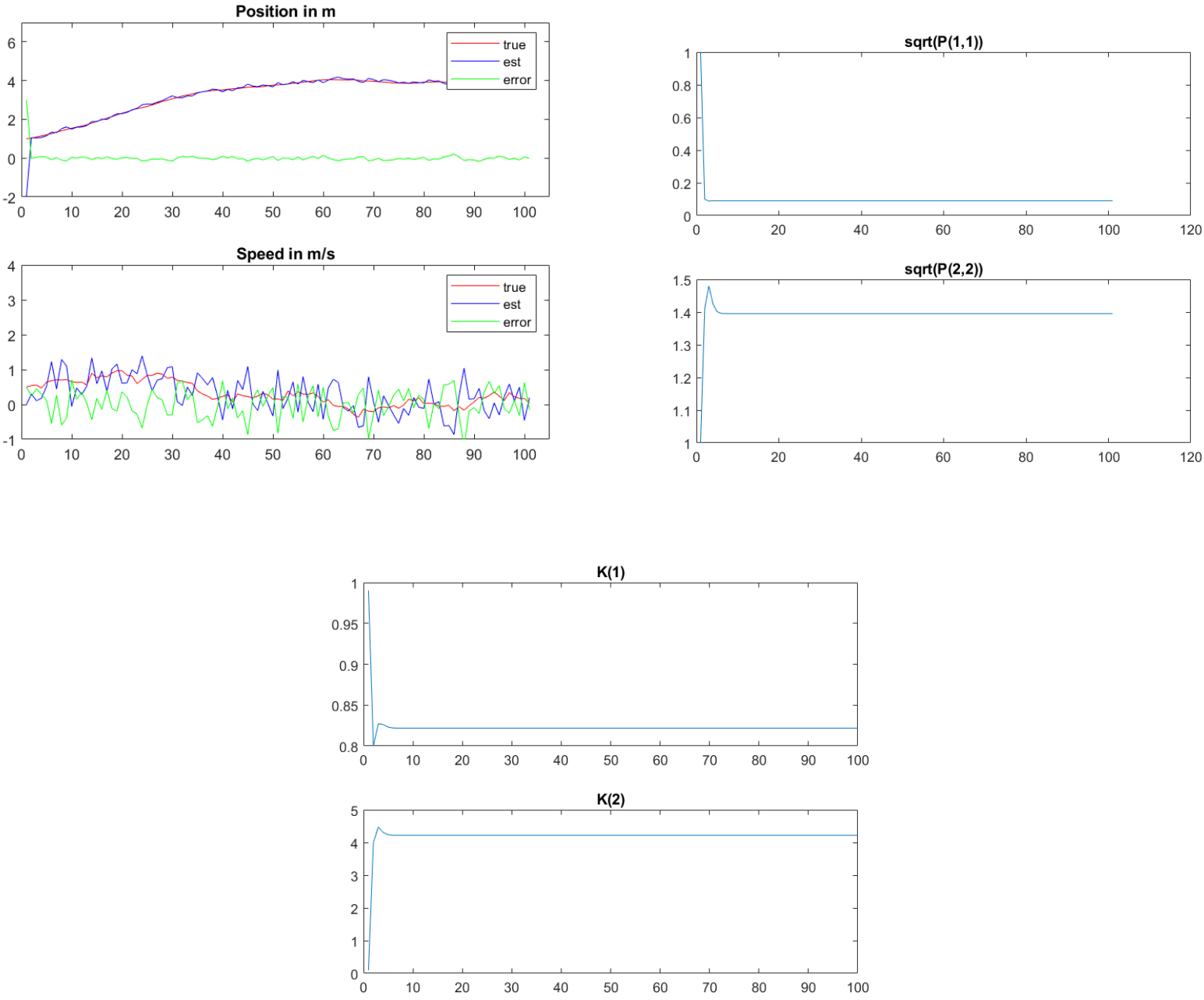


Figure 4: Increasing the transition noise: $R_t \cdot 100$

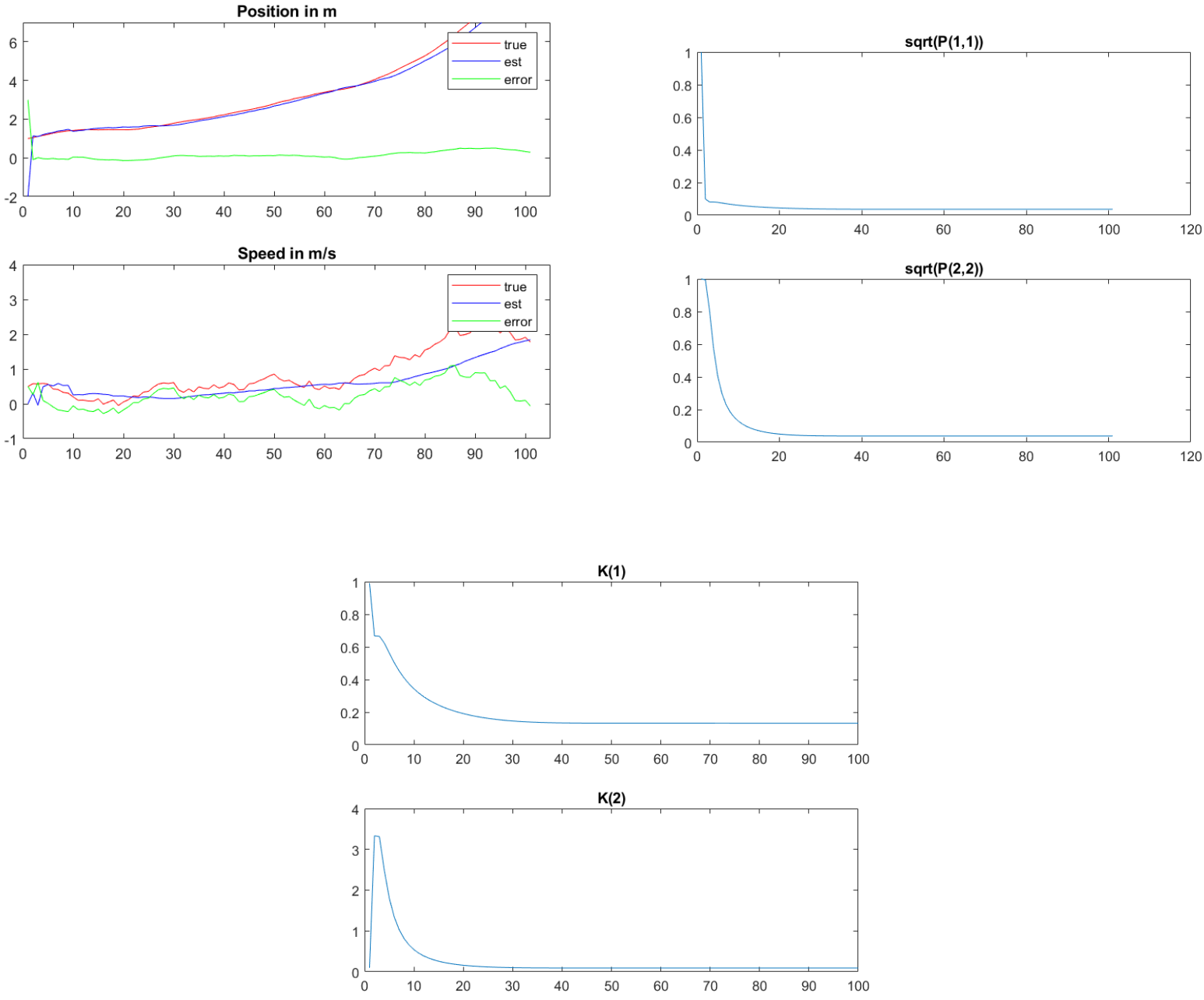


Figure 5: Decreasing the transition noise: $R_t \cdot \frac{1}{100}$

Findings from varying Q and R

In Figure 1 the results form using the default parameter settings for the measurement Q and transition noise R can be found. In Figure 2 the results from decreasing the measurement noise by a factor 100 can be found. When Q is decreased the Kalman Gain goes up significantly which decreases the uncertainty Σ and more weight is given to the new measurement compared to the default setting. Thus, the algorithm converges faster. When the measurement noise is instead increased by a factor 100 in Figure 3, one can see in the lower part of the Figure that the Kalman Gain goes down. Therefore, less weight is given to new measurements. Also, the plotted uncertainty Σ has increased compared to the default setting and the overall affect is that convergence is slower compared to Figure 1. Next, Q is set back to its default value and the transition noise R_t is increased by a factor 100 in Figure 4. Since we increase the uncertainty in the transitions we see how the Σ plot in the upper right of 4 increases. Now, we put more trust in our measurements compared to the predictions and thereby the Kalman Gain increases significantly in the lower half of 4. Now, the speed of the car is uncertain and thus convergence time increases. Lastly, the results for decreasing the transition noise R_t by a factor 100 can be found in Figure 4. The effects here are reverse to the case of increasing the

transition noise.

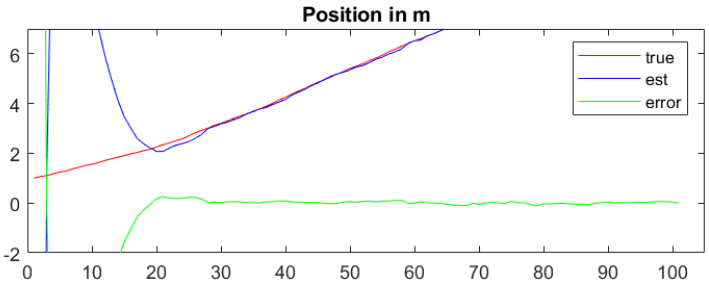
- **Question 4:**

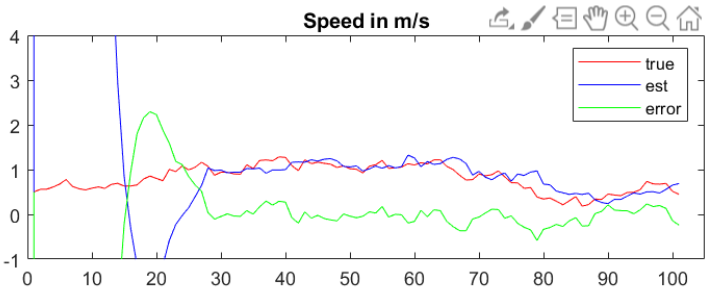
How do the initial values for P and $\hat{x}$ affect the rate of convergence and the error of the estimates (try both much bigger and much smaller)?

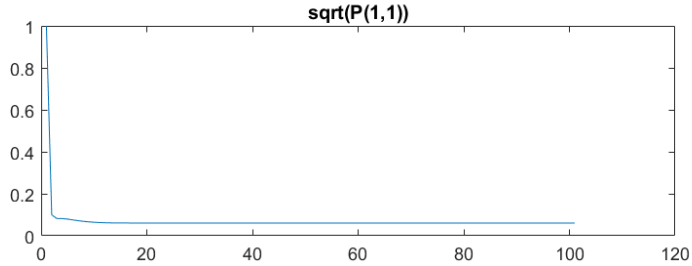
Answer:

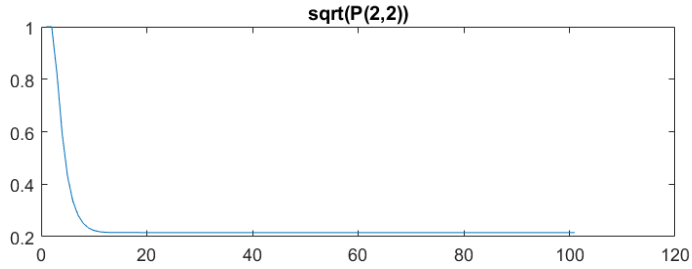
Varying the initial $\hat{x}$

The results from increasing the initial value for $\hat{x}$ by a factor 2000 can be found in Figure 6 and from decreasing $\hat{x}$ by a factor 2000 can be found in Figure 7. Decreasing $\hat{x}$ by a factor 2000 seems to give slightly faster convergence and smaller estimation error compared to when it is increased by a factor 2000. However, overall it looks as if the Kalman Filter is quite unaffected by how one chooses to initialise the state at the first time step.









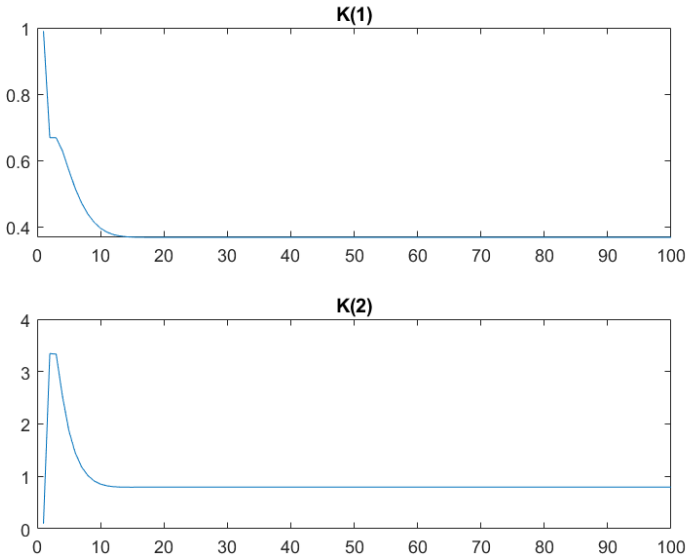
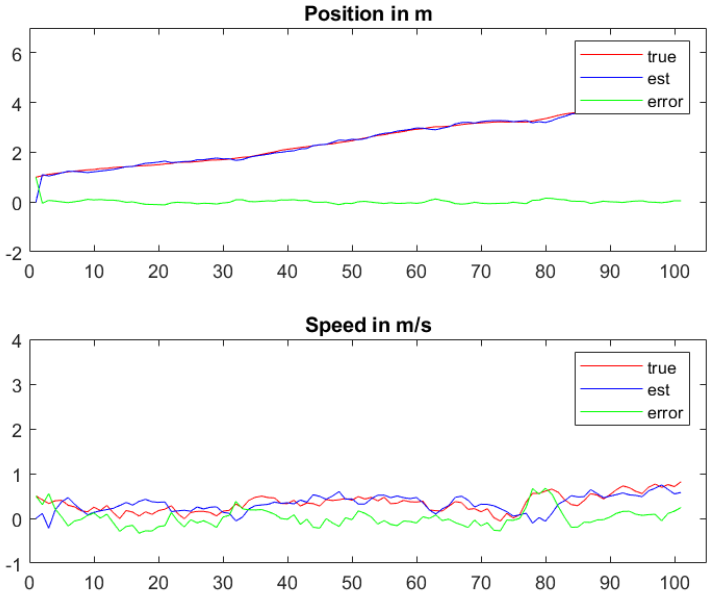
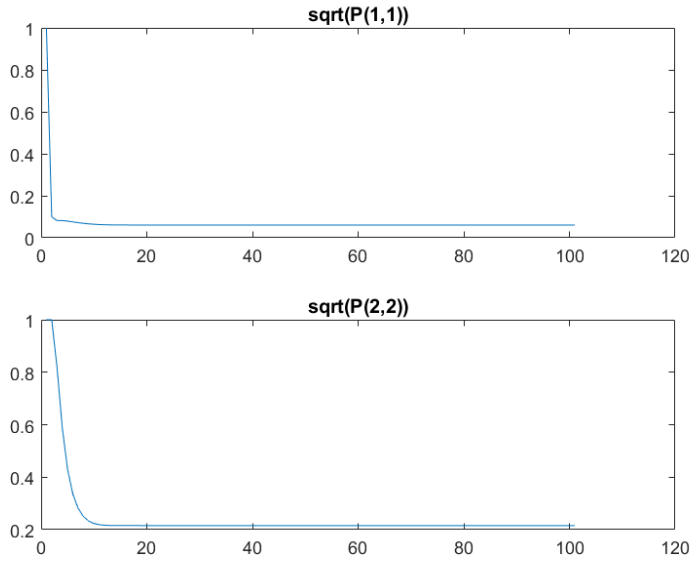


Figure 6: Increasing the initial values for the state: $\hat{x} \cdot 2000$





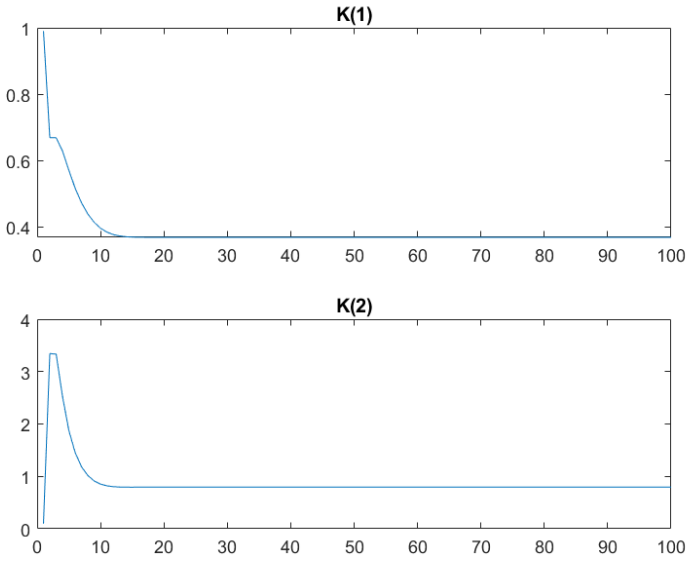


Figure 7: Decreasing the initial values for the state: $\hat{x} \cdot \frac{1}{2000}$

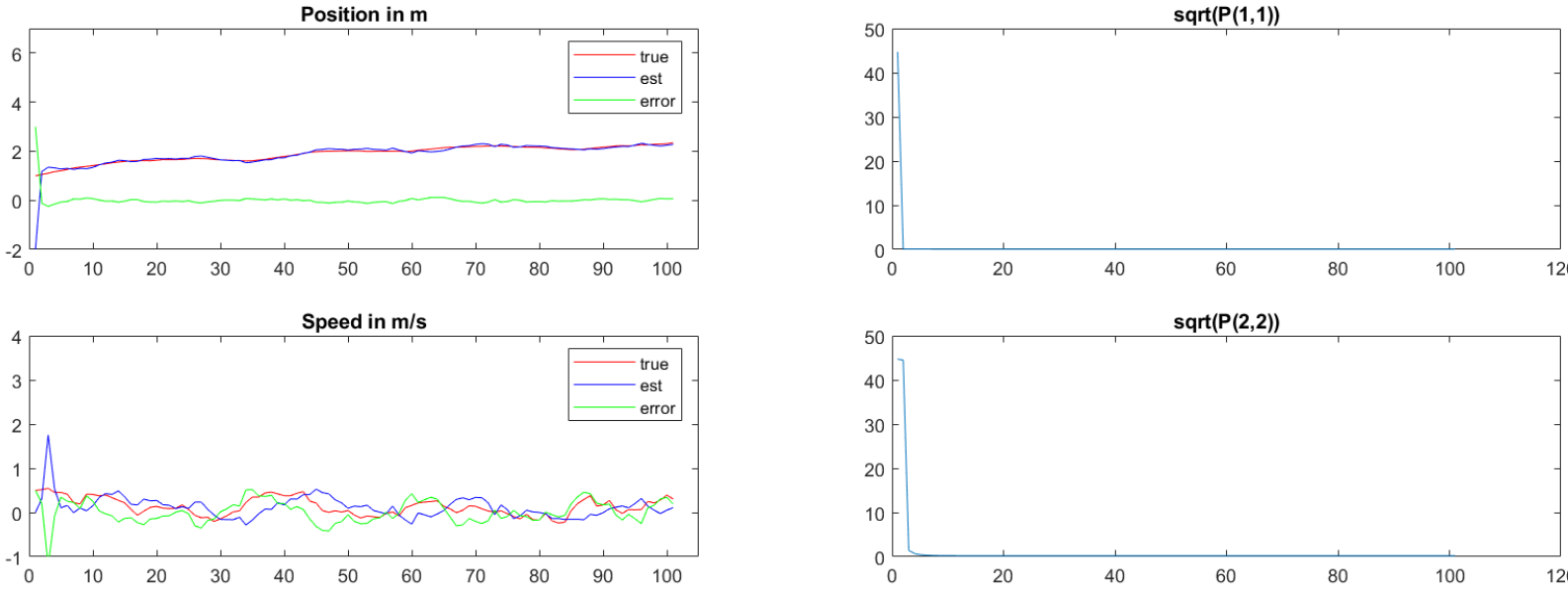
Varying the initial P

The results from increasing the initial value for P by a factor 2000 can be found in Figure 8 and from decreasing P by a factor 2000 can be found in Figure 9.

When the initial P value is very large one gets a large initial uncertainty in the state predictions and thus the initial values in the upper right in Figure 8 are very large. This then increase the Kalman Gain such that we give larger weights to measurement compared to the uncertain predictions. The reverse relationship is found in Figure 9. Since the predictions are very certain at the beginning the Kalman Gain becomes small initially and more weight is given to predictions.

When it comes to convergence using a large P value does not seem to have an affect. However, convergence slows down somewhat when the initial P value is decreased.

Lastly, the error in the state estimate seems to be unaffected by increasing P, but increases when P is decreased,



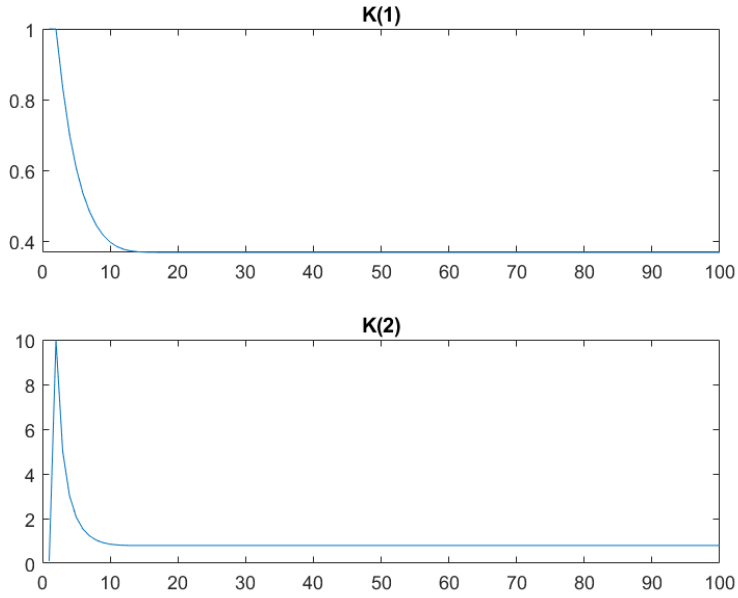
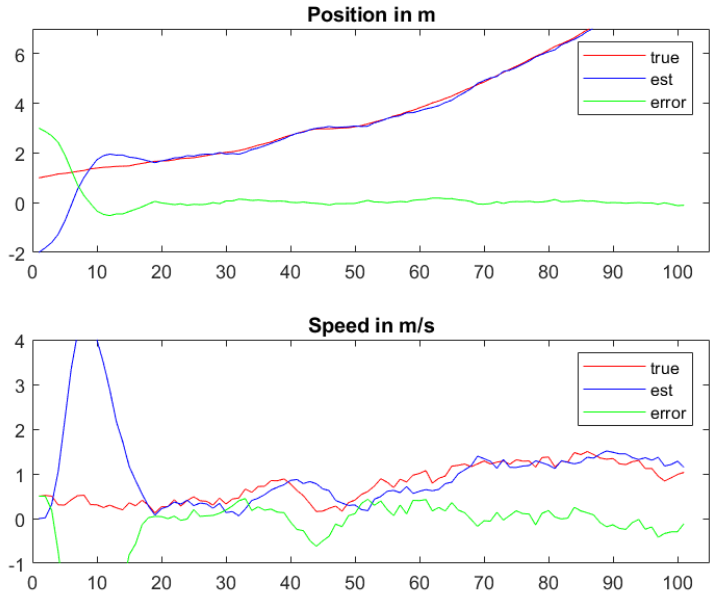
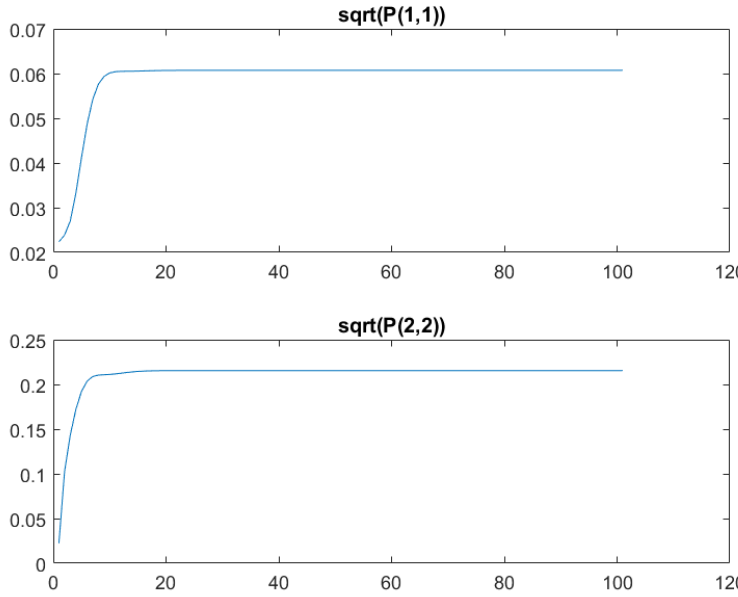


Figure 8: Increasing the initial values for P: $P \cdot 2000$





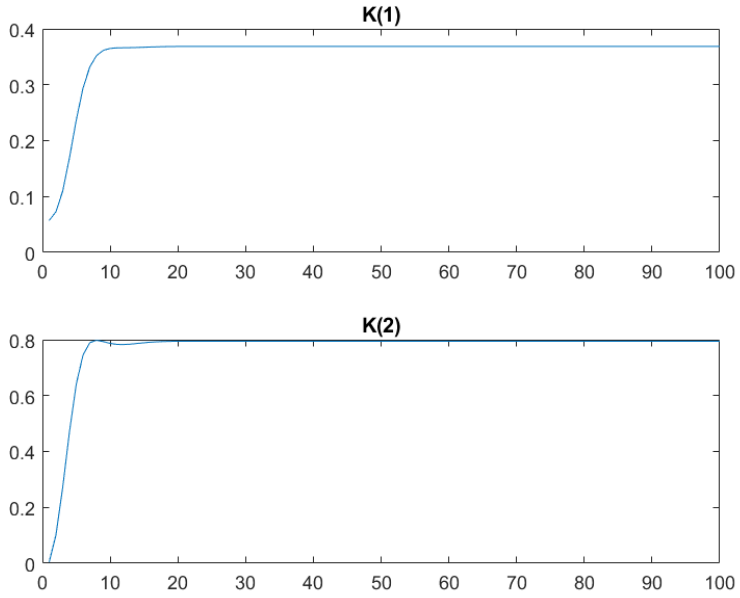


Figure 9: Increasing the initial values for P: $P \cdot \frac{1}{2000}$

2 Main Problem: EKF Localization

Question 5: Which parts of (1) and (2) are responsible for prediction and update steps?

$$p(x_t|u_{1:t}, z_{1:t}, \bar{x_0}, M) = \eta p(z_t|x_t, M) \int p(x_t|u_t, x_{t-1}) p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, \bar{x_0}, M) dx_{t-1} \tag{1}$$

Above, in eq.(1) the blue(cyan) part of the expression is the prediction step and the entire expression is the update.

$$\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1} \tag{2}$$

$$bel(x_t) = p(x_t|u_{1:t}, z_{1:t}, \bar{x_0}, M) = \eta p(z_t|x_t, M) \overline{bel}(x_t) \tag{3}$$

Here, the prediction is conducted in (2) and the the update is performed in (3).

Question 6: In the Maximum Likelihood data association, we assumed that the measurements are independent of each other. Is this a valid assumption? Explain why.

In the real world measurements are likely not always independent and thus making this assumption is not valid. However, in practice it turns out that even though we make this invalid assumption we can still get a model that works quite accurately. The benefit of making this assumption is that the total likelihood for the measurement becomes a product of disjoint individual likelihoods for each measurement. Thus, the maximum of this product is obtained by maximizing the individual likelihood of each measurement (Thrun et al. 2005, p. 218).

Question 7: What are the bounds for δ_M in (8)? How does the choice of δ_M affect the outlier rejection process? What value do you suggest for λ_M when we have reliable measurements all arising from features in our map, that is all our measurements come from features on our map? What about a scenario with unreliable measurements with many arising from so called clutter or spurious measurements?

- Since δ_M is a CDF, i.e. a valid probability distribution its bounds are in the range: $[0, 1]$.
- The choice of δ_M decides how many measurements we disregard. As δ_M increases we keep measurements with larger mahalonobis distances and thus fewer points are classified as outliers. Meaning that when $\delta_M = 1$, the probability of keep every measurement is 100%.

- If all measurements come from features in our map then we know that we have no outliers and thus we want to keep all measurements. Therefore, λ should be set as λ_{max}. Which means that the probability of keeping a measurement becomes 100%.
- If we have unreliable measurements in general we want to be careful about which measurements we accept and therefore we set a small λ value to filter out unreliable data.
- If feature spacing is small problems with data association could arise, such that we wrongly associate our measurements with another nearby feature due to uncertainty. To detect this source of error we could compare the mahalonobis distances of our measurements and not use measurements where for instance the two best features have a very similar distance. If the data instead is unreliable because feature spacing is very large other types of issues arises. If far away outliers should be rejected to hinder the algorithm from moving to a random mean. However, outliers close to the actual expectation will make us overconfident. This is not as damaging though as accepting far off outliers.

Question 8: Can you think of some down-sides of the sequential update approach (Algorithm 3)? Hint: How does the first (noisy) measurement affect the intermediate results?

When we use a sequential algorithm and our first measurement is noisy the algorithm will have no way of distinguish between noisy and non-noisy features. Thereby, its computed mean at this point will be located far away from the actual expectation. Since the association of the i^{th} data point is based on previous measurements subsequent non-noisy measurements for features far away from the noisy one could be regarded as outliers. This could ultimately result in divergence.

Question 9: How can you modify Algorithm 4 to avoid redundant recomputations?

There are recomputations in the beginning of the inner for loop. The batch update algorithm goes through every possible landmark for every observation in a batch. In this batch all computations will be based on the same $\bar{\mu}_t$ and $\bar{\Sigma}_t$ from the previous batch, meaning that the first three lines for a certain landmark will have the same result irrespective of the observation in that batch. Thus, these three parameters could be computed once for every batch and then be stored and accessed for every observation in that batch.

Question 10: What are the dimensions of $\bar{\nu}_t$ and $\bar{H}_t$ in Algorithm 4? What were the corresponding dimensions in the sequential update algorithm? What does this tell you?

- The dimensions of $\bar{\nu}_t^i$ for a certain observation i and landmark c_t^i corresponds to the dimensionality of $z_{t,i}$ which is: 2x1. Then, $\bar{\nu}_t$ is the concatenation of the $\bar{\nu}_t^i$ vectors for the number of inlier observations: n. Thus, the dimensionality of $\bar{\nu}_t$ is 2nx1.
- The dimensionality for $\bar{H}_{t,j}$ is 2x3. Thus, the concatenation of the inliner observation jacobian $\bar{H}_t$ gets the dimensionality 2nx3. Where n is the number of inliner observations.
- In the sequential update algorithm we only use one observation at the time, i.e. we have a batch size of 1. Therefore, the dimensionality of $\bar{\nu}_t^i$ is 2x1 and the dimensionality of $\bar{H}_t$ is 2x3.
- As only one observation at the time is used to update the parameter estimations in the sequential algorithm, this algorithm becomes considerably slower compared to the batch update algorithm. However, the batch update algorithm requires more memory during run time since it updates the parameters using matricies of observation batches.

3 EKF localization applied to three different Datasets

3.1 Dataset 1

Using the default standard deviations gives us small enough errors. These standard deviations gives us the following noise parameters in the motion and measurement models:

$$Q = \begin{pmatrix} 0.01^2 & 0 \\ 0 & (1^\circ \cdot \frac{\pi}{180^\circ})^2 \end{pmatrix} = \begin{pmatrix} 0.01^2 & 0 \\ 0 & (\frac{\pi}{180})^2 \end{pmatrix} \tag{4}$$

$$R = \begin{pmatrix} 0.01^2 & 0 & 0 \\ 0 & 0.01^2 & 0 \\ 0 & 0 & (\frac{\pi}{180})^2 \end{pmatrix} \tag{5}$$

The results can be found in Figure 10 and 11. In the right plot in Figure 11 one can see that the mean absolute error for the robot location in both x and y are $0.00m$. The absolute mean error for θ is $0.15° = 0.00261799387 \approx 0.0026$ radians.

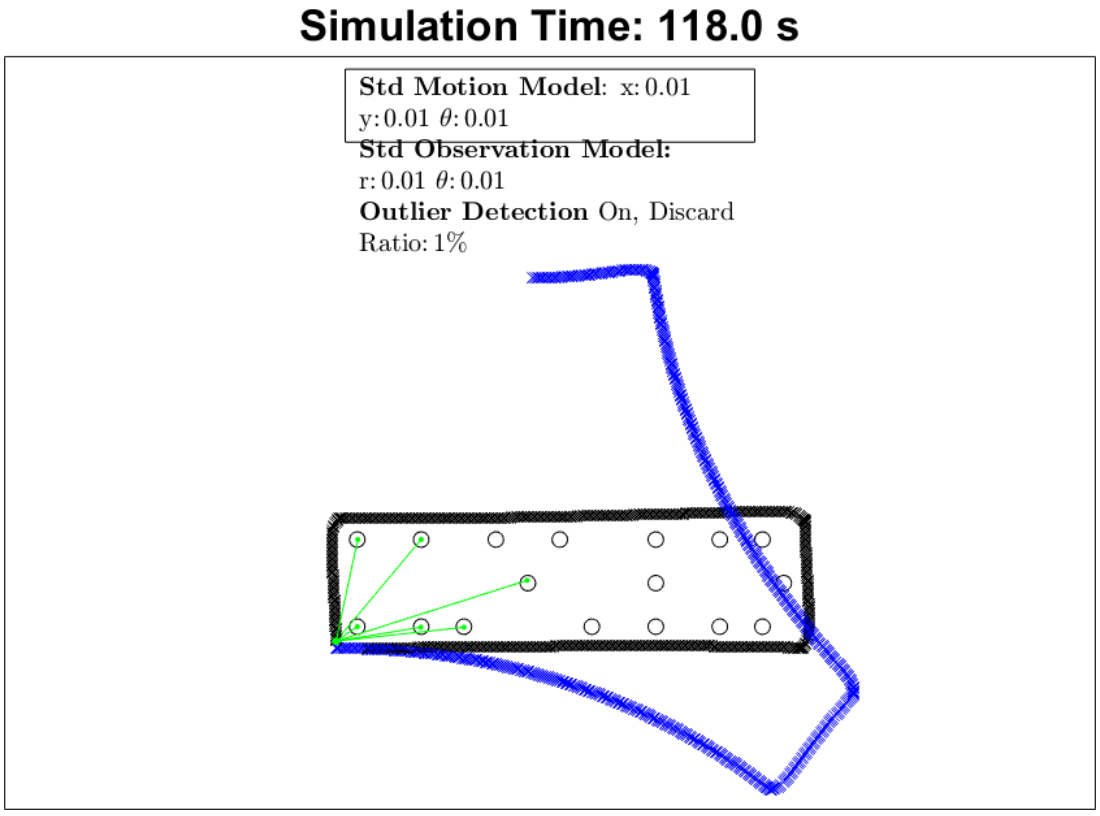


Figure 10: Simulation for Dataset 1.

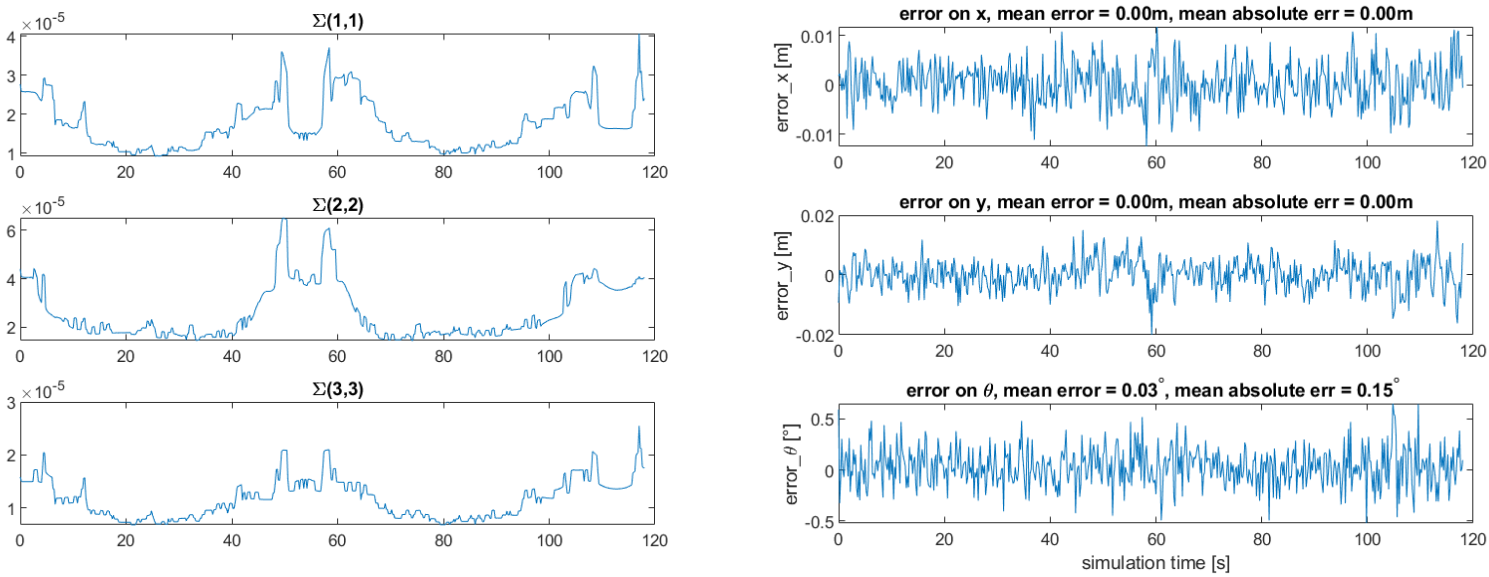


Figure 11: Evolution of state estimation covariance matrix for Dataset 1 to the left. Evolution of state estimation errors for x, y and θ for Dataset 1 to the right.

3.2 Dataset 2

Here, the default setting was used for the motion model noise, while I focused on setting the uncertainty in the measurement model and the threshold for the outlier detection. The following parameters were used:

$$Q = \begin{pmatrix} 0.023^2 & 0 \\ 0 & (13 \cdot \frac{\pi}{180})^2 \end{pmatrix} \tag{6}$$

$$R = \begin{pmatrix} 0.05^2 & 0 & 0 \\ 0 & 0.05^2 & 0 \\ 0 & 0 & (3 \cdot \frac{\pi}{180})^2 \end{pmatrix} \tag{7}$$

The results can be found in Figure 12 and 13. In the right plot in Figure 13 one can see that the mean absolute error for the robot location in both x and y are $0.05m$. The absolute mean error for θ is $2.54° = 0.044331363 \approx 0.044$ radians.

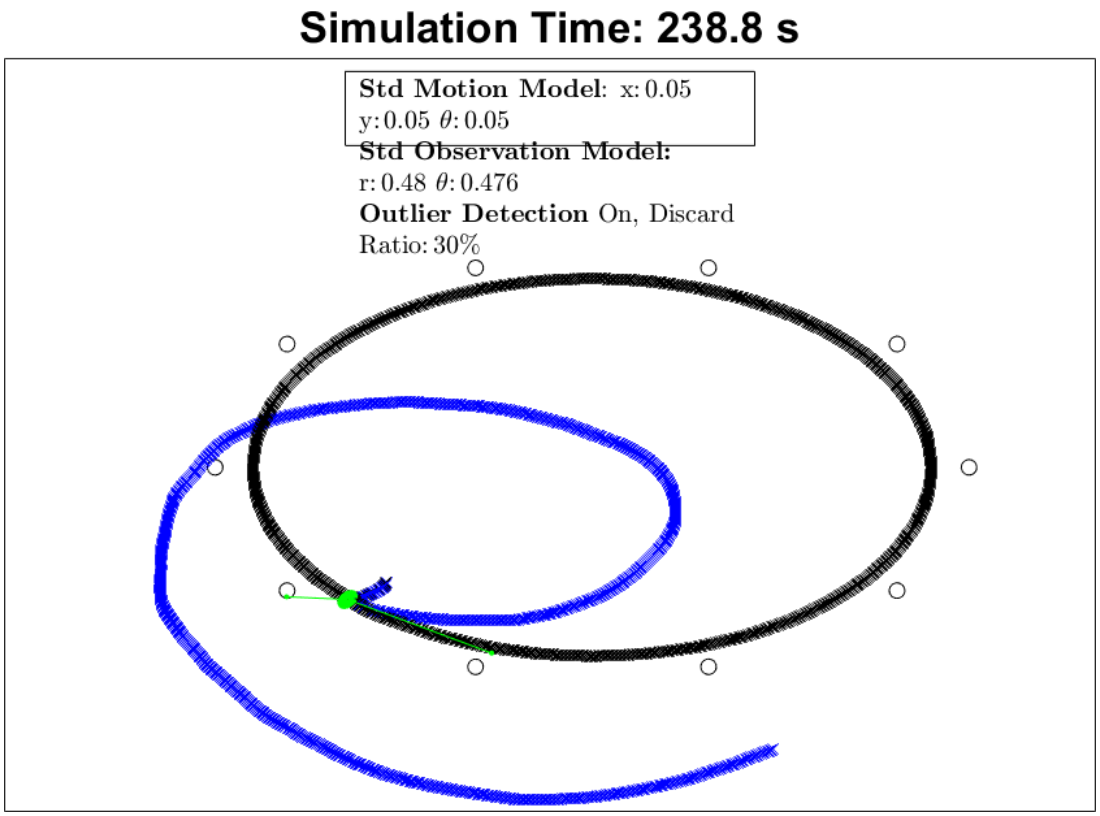


Figure 12: Simulation for Dataset 2.

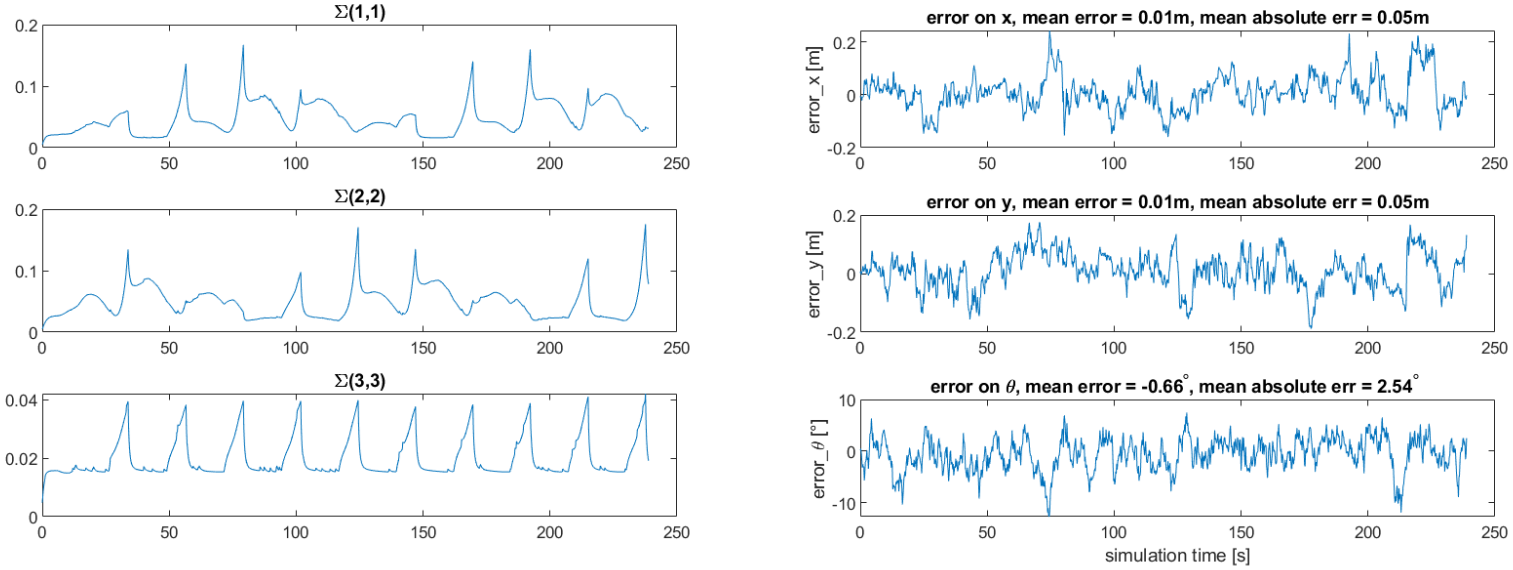


Figure 13: Evolution of state estimation covariance matrix for Dataset 2 to the left. Evolution of state estimation errors for x, y and θ for Dataset 2 to the right.

3.3 Dataset 3

The results can be found in Figure 14 and 15. Here the default parameter setting gave small enough errors. As can be seen in the Figure below, the standard deviation for the motion model was set to 1 and for the observation model

0.1. In the right plot in Figure 15 one can see that the mean absolute error for the robot location in x is $0.08m$ and in y is $0.09m$. The absolute mean error for θ is $2.74° = 0.0478220215 \approx 0.048$ radians.

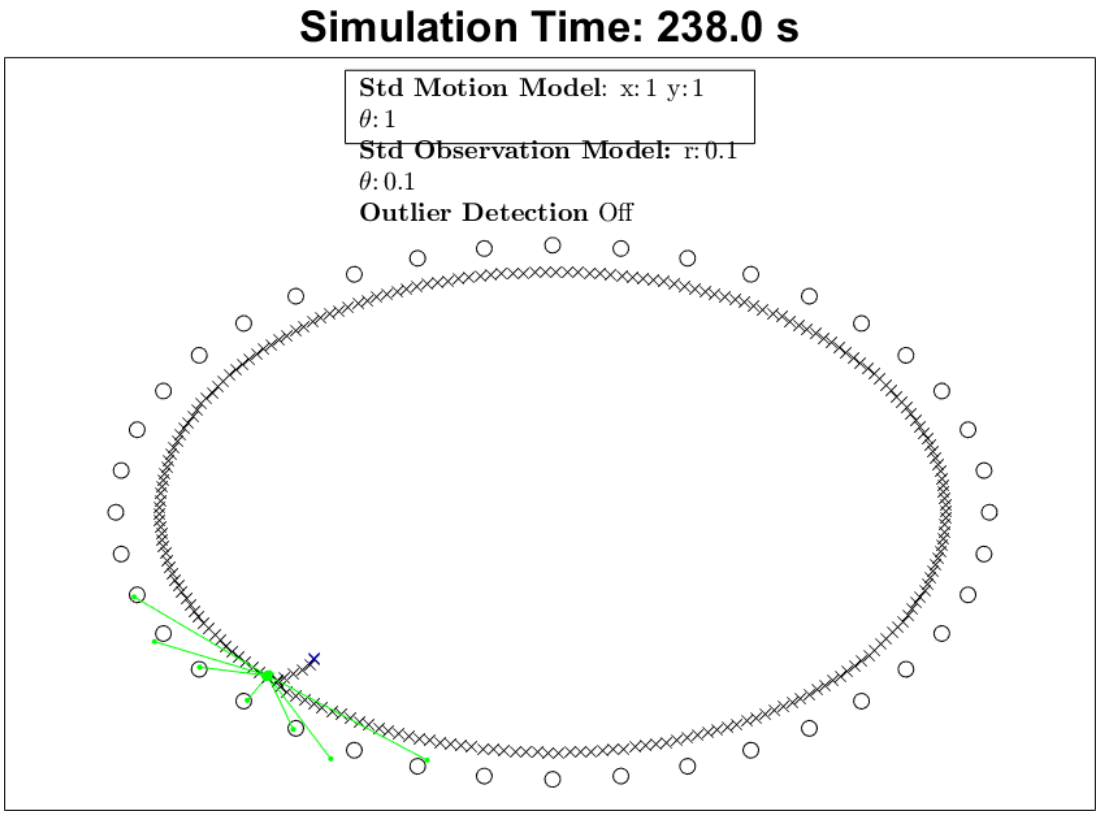


Figure 14: Simulation for Dataset 3.

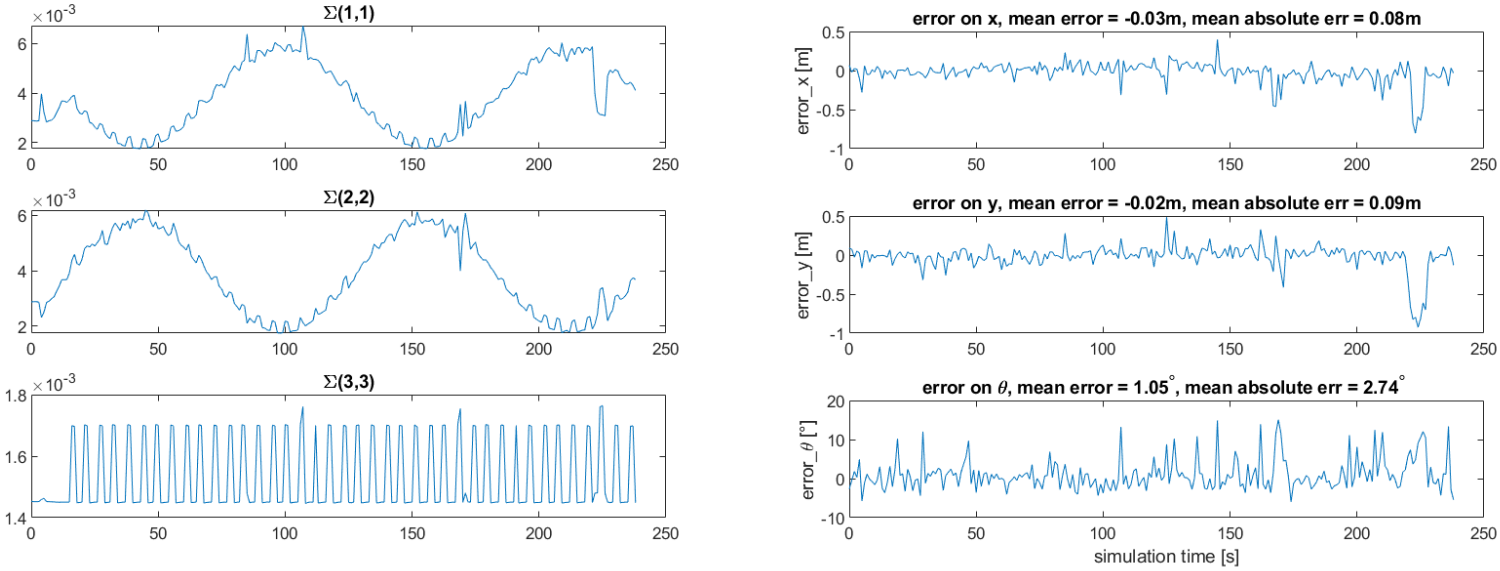


Figure 15: Evolution of state estimation covariance matrix for Dataset 3 to the left. Evolution of state estimation errors for x, y and θ for Dataset 3 to the right.

References

Thrun, S., Burgard, W. & Fox, D. (2005), *Probabilistic robotics*, MIT Press, Cambridge, Mass.