Assignment 2:

Task 2:

There are two stages of the Extended Kalman Filter:

Firstly, prediction of the mean and standard deviation of the location of the robot using the non-linear motion function Jacobian , the process noise (which is a matrix of expected motion model noise), control inputs , and the robots previous state and standard deviation . As outlined in equation 1 & 2 listed below.

Equation 2

Equation 1

Secondly the filter works by using the Kalman gain weighting , measurement uncertainty , measurement data (for our task aruco readings), to find the updated state and standard deviation of the robot using equation 3,4 and 5. These new state and standard deviation values are the positions where the robot is most likely to be given the trust in the motion model and the measurement readings.

Equation 4

Equation 5

Equation 3

The uncertainty matrix R within Figure 3 in task 1 is derived using the inverse kinematic matrix G which links the wheel distance and wheel diameter to the linear velocity and angular velocity as well as using the motion model (these equations are outlined just under table 2 within task 1). Using the uncertainties for these wheel parameters can propagate through this model to find the northing, eastings and angular uncertainties.

We achieved uncertainties for the wheel spacing and diameter using the resolution of our measurement device which we found to be 5% for our wheel diameter and 10% for our wheel spacing, as outlined within Table 2

Multiple linear test where taken with their timestamps interpolated to find an accurate value for aruco (measurement) noise values as shown within figure 6 of the task 1. This allowed us to accurately simulation our Kalman Filter for our different Q sweep values.

We used swept through a range of Q values to vary the ration between Q and R. This allowed us to change which parameter the robot trusted more and hence which parameter the Extended Kalman filters output would tend towards. This is shown from figure 8 within task 1 which outlines that there is a clear reduction in the distance between the Extended Kalman Filter output and the measurement reading when the measurement noise is reduced. There was a significant increase in error of the robots position when cornering. Due to this our choice of final parameters was partly based on the Q values with one of the lowest peaks within this graph.

We plotted the average error over the entire run for the each swept value seen within figure 7. There was a clear convergence minima from Q scaling values of 0.1 to 1.0 which coincided with some of our optimum cornering ranges. Due to this we chose to use our final parameter set seen within Table 2.

Overall we found that if the ration between Q and R was too low (small Q scaling factor), then the model ignored the motion model and only trusted the measurement reading which was had a constant error equal to the noise of the aruco measurement. We also found that if the Q value was too large then I the robot would ignore any measurement readings and solely trust the motion model as if there was no Extended Kalman Filter applied.

Task 3:

When used for extend periods of time, Extended Kalman Filters error from linearization accumulates according to [1]. Figure [1] outlines how the NEES (the value which is used to estimate whether the estimated states covariance matrix aligns with the actual estimation errors) increases. Because of this the system has an increasing level of state drift and needs to be re-localized after a certain period of time. EKF are also more effected by starting sensor noise and process uncertainties making them harder to calibrate for longer operational windows which could have an increase in changing environmental conditions which effect aspects such as wheel slip due to changing ground conditions, lidar permeability due to changing atmosphere condition, or wheel diameter due to tire wear. Particle filters are able to use adaptively resample the location of their particles over time which prevents this type of long term drift making them overall better with regards to accuracy in long usage situations.

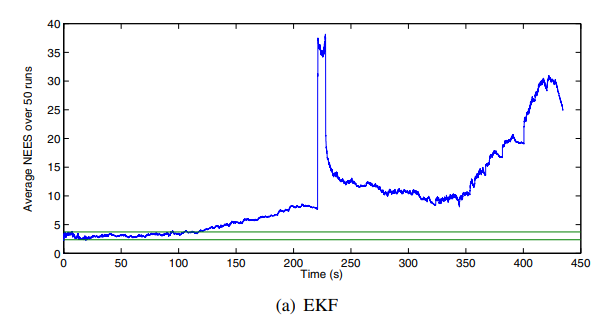


Figure : Outlines the Average NEES over 50 run for a vehicle configuration using an EKF

As described within [2] particle filters can be quite computationally demanding using up to 10MB of data. This can cause issues with regards to power usage for long operating times causing the robots battier to be drained quicker. Robots outfitted with an Extended Kalman Filters on the other hand require less computational power and therefore can stay operational for longer timeframes. Since particle filters require larger amounts of computational power, they also require better processing equipment which can make the robot overall more expensive and larger to not sacrifice accuracy within the model.

Since EKF’s linearizes the motion within a differential drive robot configuration there is a slight linearization error mapping its nonlinear function onto a gaussian distribution. This type of error is unbounded and can cause growing drift when left for a long period of time. On the other hand a systems such a as a particle filter works to consistently resample locations to prevent this type of drift.

Task 4:

As part of my contributions I wrote code to find the distance between the Kalman filter and the measurement data, as well as the overall error of the system for different Q:R ratios. I did this both by varying Q values (which was submitted within task 1 of the report) and varying the R matrix which (which was not submitted within task 1). Both methods where used to plot graphs such as figures 6 to 9 showing the overall error and the trends of the Kalman filter to ensure that it was working correctly.

The list of values iterated within my simulations can be found within the appendix located at the end of this report, as well as example of some of the code used to create the variable, append the data within the logs and extract the data.

I swept through both the Q and R values as I wanted to see if there would be a difference is overall for a set ration between Q and R, as within the Extended Kalman filter, this is the main functionality in which the robot decides which type of process (i.e measurement or motion model) the robot trusts. To model this functionality I used the variable d (as shown in figure 8 in task 1) which is used to asses if the Extended Kalman Filter was operating as expected.

Most of my team members throughout the assessment timeframe where working on setting up the Extended Kalman Filter and organizing the data logs to accurately ensure we could retrieve usable data, we could plot it, and we could create the right set of matrixes from our tests. In line with this I set out to ensure that we had all of the required data to make an informed decision on which parameters would be used for task 1 and we undertook the right tests to find aruco noise according to literature

# Bibliography

|  |  |
| --- | --- |
| [1] | J. N. J. G. M. S. a. E. N. Tim Bailey, “Consistency of the EKF-SLAM Algorithm,” *Australian Centre for Field Robotics.* |
| [2] | W. B. Dieter Fox, “Monte Carlo Localization: Efficient Position Estimation for Mobile Robots,” *School of Computer Science,* 1999. |

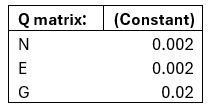


Table 1: Table of values used within R matrix sweep

Table 2: Table of Values kept constant for Q matrix

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **R matrix:** | **Base Value: (all variance are squared)** | **Scale Multiplier: 0.1** | **Scale Multiplier: 0.2** | **Scale Multiplier: 0.4** | **Scale Multiplier: 0.8** | **Scale Multiplier: 1.0** | **Scale Multiplier: 2.0** | **Scale Multiplier: 4.0** | **Scale Multiplier: 6.0** | **Scale Multiplier: 8.0** | **Scale Multiplier: 10.0** |
| N /m^2 | 0.002 | 0.0002 | 0.0004 | 0.0008 | 0.0016 | 0.002 | 0.004 | 0.008 | 0.012 | 0.016 | 0.02 |
| E /m^2 | 0.002 | 0.0002 | 0.0004 | 0.0008 | 0.0016 | 0.002 | 0.004 | 0.008 | 0.012 | 0.016 | 0.02 |
| G /Rad^2 | 0.02 | 0.002 | 0.004 | 0.008 | 0.016 | 0.02 | 0.04 | 0.08 | 0.12 | 0.16 | 0.2 |
| DOTX | 0.002 | 0.0002 | 0.0004 | 0.0008 | 0.0016 | 0.002 | 0.004 | 0.008 | 0.012 | 0.016 | 0.02 |
| DOTG | 0.02 | 0.002 | 0.004 | 0.008 | 0.016 | 0.02 | 0.04 | 0.08 | 0.12 | 0.16 | 0.2 |

A screen shot of a computer code

AI-generated content may be incorrect.

Appendix:

A screen shot of a computer program

AI-generated content may be incorrect.