**Homework #3 Report  
[EECN30168] Self-Driving Cars 2022**

**Student ID:** 311605004 **Name:** 劉子齊 **Date: 2022.10.19**

1. **Introduction**

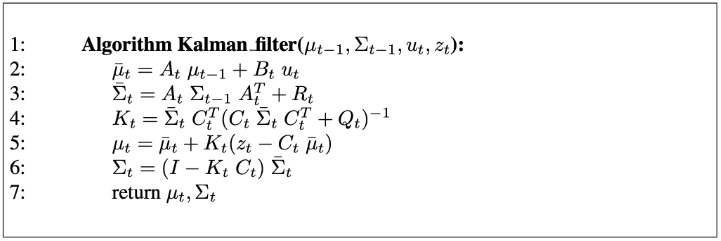
In this homework, our goal is to understand Kalman Filter, and implement the Kalman filter through our own understandings. This may sound a bit simple, since we could have done this by simply call the Kalman filter from the FilterPy library. However, in this homework, we could only use NumPy and standard libraries of python.

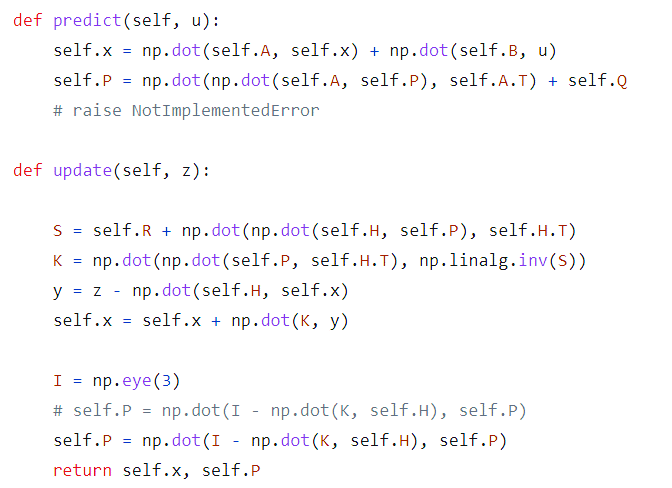
For the implementation requirements, we have to implement the Kalman filter with the robot state [x, y, yaw]. Then we need to plot the output through the plotting function provided by the Tas, and analysis the result. I’ll explain my implementation details in the following section.

1. **Code Explanation**

The following is the detail of the implementation of the predict and update function of my Kalman filter. Besides, I’ll explain the design of the covariance matrices in the forth session.

When implementing my Kalman filter, I followed the algorithm provided by the TAs, which was as the following figure:





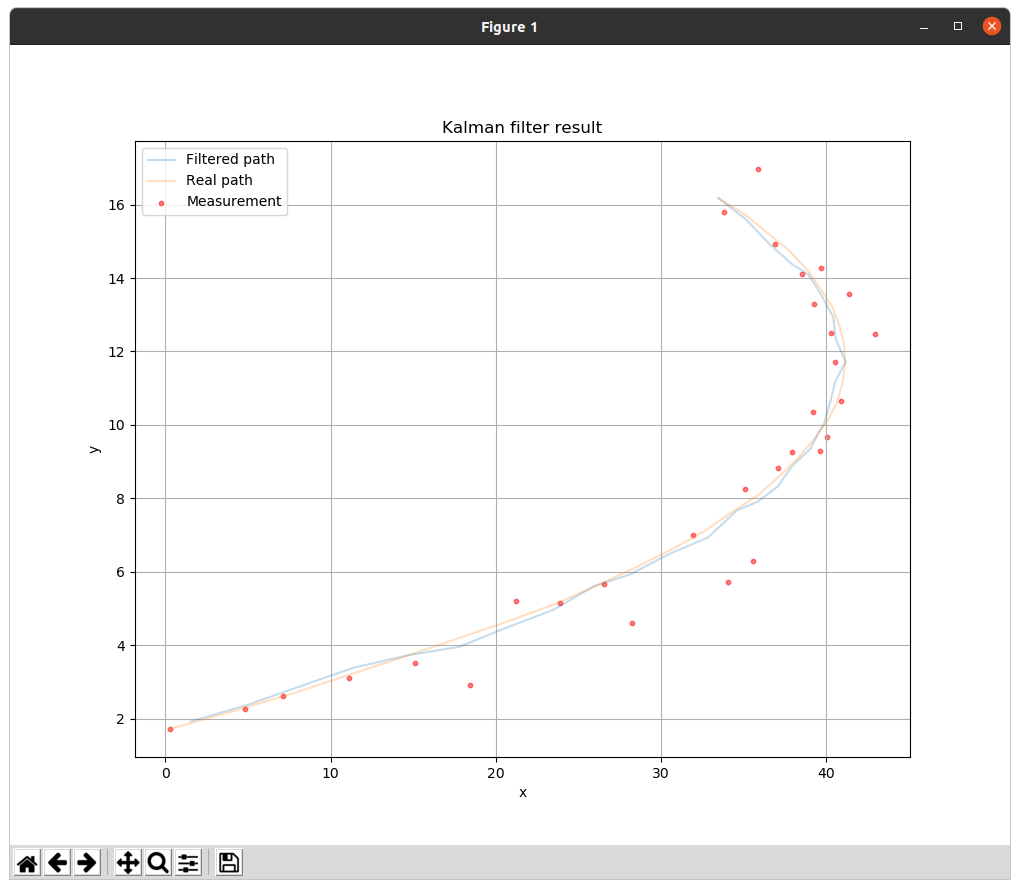
For the predict function, I first calculated the predicted state estimate x. I obtain the predicted state estimate x through adding the dot product of the transition matrix A and the state estimate x, and the dot product of the transition matrix B and the control vector u. Then I calculate the error matrix P through adding up the covariance of the state transition error Q with the dot product of the transition matrix A, the error matrix P and the transform of the transition matrix A.

For the update function, I first calculate the innovation covariance S, which can be found in the parentheses in the equation of the optimal Kalman gain K in the algorithm above. For the innovation covariance S, I added up the measurement error R with the dot product of the observation matrix H, the error matrix P, and the transform of the observation matrix H. Then I obtain the optimal Kalman gain K by calculating the dot product of the error matrix P, the transform of the observation matrix H, and the inverse of the innovation covariance S matrix.

Then I’ll calculate the measurement post-fit residual y, which is the subtraction of the dot product of the observation matrix H and the state estimate x to the matrix z. With the measurement post-fit residual y, we can get the final state estimate x by adding the state estimate x itself with the dot product of the optimal Kalman gain K and the measurement post-fit residual y.

Lastly, for the error matrix P, we first initialize a 3x3 matrix I with np.eye(), which with ones on the diagonal and zeros elsewhere. Then we calculate the error matrix P by calculating the dot product of the subtraction of the dot product of the optimal Kalman gain K and the observation matrix H from the 3x3 matrix we just created and the error matrix P itself. Finally, we get the final error matrix P, and the following figure is the plotting result of the Kalman filter I implemented.

1. **Result**



1. **Design of the Covariance Matrices (Q, R)**
2. **Affection of the value of Q & R**