# Position and Orientation Estimation of an RC Car Using Kalman Filter

Course Project Report
MAE 298
Estimation Theory and Application
Professor Xinfan Lin

**Mohammad Abtahi**
(sabtahi@ucdavis.edu)

**Mahdis Rabbani**
(mrabbani@ucdavis.edu)

University of California, Davis
Spring 2023

## Problem Statement

Autonomous driving has grown a reputation in scientific communities recently. The authors' initial research area focuses on autonomous racing and path-planning of a small racing car (RC) in highly dynamic environments. To achieve this purpose, the car's position and orientation (heading) should be known with appropriate accuracy. Currently, Marvelmind Indoor GPS System with a sampling rate of 25 Hz and differential precision of $\pm 2$ cm is used to localize the car. based on the desired specification for the RC during the race, it should maintain a velocity of at least 2.5 m/s, entailing the position updates every 10 cm due to the sampling frequency of the GPS with a potential 2 cm error. Since the RC is in a highly dynamic environment, neither the 10-cm interval for position update nor the 2-cm error is desirable.

In order to address this issue, the position and orientation of the RC are estimated based on the uncertain measured positions by designing a Kalman filter. As in any estimation problem, three areas are studied in this report: first, the continuous-time model of the RC is introduced and discretized. Second, the algorithm is provided, and two different approaches are investigated in this report. Third, the data is provided for the estimator using simulations. Since the RC in the authors' laboratory is not available now, the required data are generated in MATLAB Simulink and then given to the estimator. Finally, the results are provided followed by discussion and comparison between the two approaches.

## Kinematic Model of The Car

Under certain assumptions, a kinematic model for the lateral motion of a vehicle can be developed. Such a model provides a mathematical description of the vehicle motion without considering the forces that affect the motion. The equations of motion are based purely on geometric relationships governing the system.

**Assumptions:**

The major assumption used in the development of the kinematic model is that the velocity vectors at points A and B are in the direction of the orientation of the front and rear wheels respectively. In other words, the velocity vector at the front wheel makes an angle $\delta$ with the longitudinal axis of the vehicle. Also, the velocity vector at the rear wheel is in the direction of the longitudinal axis of the vehicle. This assumption can be simplified as assuming that slip angle is zero at both wheels. This is a reasonable assumption for low-speed motion of the vehicle like the case in this problem (speeds less than 5 m/s). The total lateral force from both tires can be denoted as $(mV^2/R)$ in which R is the radius of the road curvature. In this study the maximum velocity of the RC car is considered to be 2.5 m/s and the RC weight is about 6kg. Also in data generation simulations, the minimum radius of the turns is considered to be 2 m. Considering all these assumptions the lateral force generated by the tires is too small and can be neglected.

As previously mentioned in the proposal, the system is an RC car designed with a 10:1 ratio to a real racing car, and the 3-DOF bicycle model of the car is utilized as shown below.

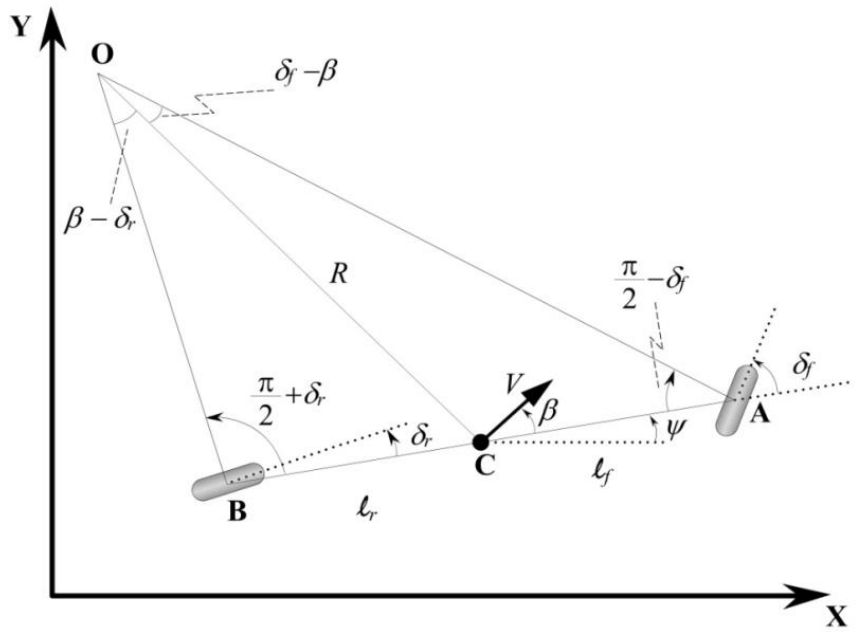


Figure 1. Kinematic diagram of the bicycle model [1].

In the bicycle model, the two left and right front wheels are represented by one single wheel at point A. It should be mentioned that only front-wheel steering is considered in this formulation and the steering on the rear wheels is not taken into account as a result $\delta_r = 0$ in this problem. The center of gravity (c.g.) of the vehicle is at point C. The distances of points A and B from the c.g. of the vehicle are $l_f$ and $l_r$ respectively. The wheelbase of the vehicle is $l_f + l_r$ .

This problem cares about the position $(X, Y)$ and the orientation $(\psi)$ of the car. So, the state vector can be written as $[X \quad Y \quad \psi]^T$. The control inputs consist of the steering angle $(\delta = \delta_f)$ and velocity $(V)$ which makes an angle $\beta$ with the longitudinal axis of the vehicle and usually is called slip angle. The imprecise measurement output is the position vector.

The ultimate goal is to estimate the state vector, implementing the imprecisely measured position and the orientation, for which we do not have a measurement system, using the steering angle and velocity as input and position as output.

Kinematic model equations in continuous form can be written as follows [1],

$$\hat{X} = \begin{bmatrix} X \\ Y \\ \psi \end{bmatrix}, \qquad \hat{Y} = \begin{bmatrix} X \\ Y \end{bmatrix}, \qquad u = \begin{bmatrix} V \\ \delta \end{bmatrix}$$

$$\dot{X} = V \times \cos(\psi + \beta)$$

$$\dot{Y} = V \times \sin(\psi + \beta)$$

$$\dot{\psi} = \frac{V \cos(\beta)}{l_f + l_r} \times \tan(\delta)$$

$$\beta = \tan^{-1}\left(\frac{l_r \times \tan(\delta)}{l_f + l_r}\right)$$

**Discrete-Time State Equations**

Continuous-time equations need to be discretized. Different methods are provided for discretizing continuous-time differential equations. In this project, Euler approximation method is used for this purpose. In this method, the first two terms of Taylor expansion of the function are used to approximate the derivative of that function. Discretization process is given below,

$$\dot{X} = \frac{X_{k+1} - X_k}{\Delta T} = V_k \times \cos(\psi_k + \beta_k) \Longrightarrow X_{k+1} = X_k + V_k \times \Delta T \times \cos(\psi_k + \beta_k)$$

$$\dot{Y} = \frac{Y_{k+1} - Y_k}{\Delta T} = V_k \times \sin(\psi_k + \beta_k) \Longrightarrow Y_{k+1} = Y_k + V_k \times \Delta T \times \sin(\psi_k + \beta_k)$$

$$\dot{\psi} = \frac{\psi_{k+1} - \psi_k}{\Delta T} = \frac{V_k \cos(\beta_k)}{l_f + l_r} \times \tan(\delta_k) \Longrightarrow \psi_{k+1} = \psi_k + \Delta T \times \frac{V_k \cos(\beta_k)}{l_f + l_r} \times \tan(\delta_k)$$

$$\beta_k = \tan^{-1}\left(\frac{l_r \times \tan(\delta_k)}{l_f + l_r}\right)$$

Considering a time step of 0.01 s, the state space equations of the system can be rewritten in a discrete-time form as below,

$$\hat{X}_{k+1} = \begin{bmatrix} f_X(\hat{X}_k, u_k) \\ f_Y(\hat{X}_k, u_k) \\ f_\psi(\hat{X}_k, u_k) \end{bmatrix}_k = f_k(\hat{X}_k, u_k)$$

$$f_X(\hat{X}_k, u_k) = X_k + V_k \times \Delta T \times \cos(\psi_k + \beta_k)$$

$$f_Y(\hat{X}_k, u_k) = Y_k + V_k \times \Delta T \times \sin(\psi_k + \beta_k)$$

$$f_\psi(\hat{X}_k, u_k) = \psi_k + \Delta T \times \frac{V_k \cos(\beta_k)}{l_f + l_r} \times \tan(\delta_k)$$

$$\hat{Y}_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ \psi \end{bmatrix}_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \hat{X}_k = C\hat{X}_k = h_k(\hat{X}_k, u_k)$$

$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

As can be seen, the model is nonlinear in two senses: first, there are sine and cosine terms of the state $\psi_k$ and input ($\delta_k$ and $\beta_k$, which is representing the input), and second, there are multiplication of input $V_k$ into the terms related to the state $\psi_k$ and input ($\delta_k$ and $\beta_k$). Although the measurement is linear, the system is generally nonlinear. Thus, Kalman filter is not applicable directly, and two other alternatives are considered:

1. Linearized Kalman Filter
2. Extended Kalman Filter

Both alternatives are implemented in this project, and the results are compared. The remainder of this report is as follows: first, linearized Kalman filter is designed, then the process is repeated for Extended Kalman filter. Finally, the results and discussion are provided.

## Linearized Kalman Filter

There are two different sources of uncertainty in the state space equations. One source is the uncertainty in the model which is mostly caused by simplifying the high-fidelity model or using inaccurate model parameters, also called process noise, and the other source is uncertainty and noises in the measurement system or sensors. These noises and uncertainties turn the state and measurement vectors to random vectors. In the context of Kalman filter, both process and measurement noises are assumed to be additive and from Gaussian distribution with zero mean and specific covariance matrices, $Q$ and $R$, as below,

$$[w]_{3\times1} \sim N(0\,, Q)$$
$$[v]_{2\times1} \sim N(0\,, R)$$

$$\hat{X}_{k+1} = f_k(\hat{X}_k, u_k, w_k)$$

$$\hat{Y}_k = h_k(\hat{X}_k, u_k, v_k)$$

At this step, the system needs to be linearized. For this purpose, variables are changed to the deviation from, and matrices $A'_k, B'_k, E'_k, C'_k, D'_k$, and $F'_k$ are calculated based on the Jacobian of the initial nonlinear functions at the operating point. The state operating point at step $k$ is assumed to be the final state estimation from step $k-1$. The operating point for the input at each step is assumed to be the previous step's input. The new state space equations can be written as follows,

$$\delta x_{k+1} = A'_k \delta x_k + B'_k \delta u_k + E'_k \delta w_k$$

$$\delta y_k = C'_k \delta x_k + D'_k \delta u_k + F'_k \delta v_k$$

where

$$\delta x_{k|k} = \hat{X}_{k|k} - \hat{X}_{k-1|k-1}$$

$$\delta u_k = u_k - u_{k-1}$$

and

$$A'_k = \left.\frac{\partial f_k}{\partial \hat{X}_k}\right|_{\hat{X}_{k-1|k-1}, u_{k-1}}, \qquad B'_k = \left.\frac{\partial f_k}{\partial u_k}\right|_{\hat{X}_{k-1|k-1}, u_{k-1}}, \qquad E'_k = \left.\frac{\partial f_k}{\partial w_k}\right|_{\hat{X}_{k-1|k-1}, u_{k-1}}$$

$$C'_k = \left.\frac{\partial h_k}{\partial \hat{X}_k}\right|_{\hat{X}_{k|k-1}, u_{k-1}}, \qquad D'_k = \left.\frac{\partial h_k}{\partial u_k}\right|_{\hat{X}_{k|k-1}, u_{k-1}}, \qquad F'_k = \left.\frac{\partial h_k}{\partial v_k}\right|_{\hat{X}_{k|k-1}, u_{k-1}}$$

The measurement update equation is linear, so there is no need for the Jacobian, and they are simply obtained as below,

$$C'_k = \left.\frac{\partial h_k}{\partial \hat{X}_k}\right|_{\hat{X}_{k|k-1}} = \left.\begin{bmatrix} \dfrac{\partial h_X(\hat{X}_k, u_k)}{\partial X_k} & \dfrac{\partial h_X(\hat{X}_{k+1}, u_{k+1})}{\partial Y_k} & \dfrac{\partial h_X(\hat{X}_k, u_k)}{\partial \psi_k} \\ \dfrac{\partial h_Y(\hat{X}_k, u_k)}{\partial X_k} & \dfrac{\partial h_Y(\hat{X}_k, u_k)}{\partial Y_k} & \dfrac{\partial h_Y(\hat{X}_k, u_k)}{\partial \psi_k} \end{bmatrix}\right|_{\hat{X}_{k|k-1}} = \left.\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}\right|_{\hat{X}_{k|k-1}}$$

$$D'_k = 0_{2\times2}$$

Noises are assumed to be additive. Thus, they are not parts of the nonlinearity, and their Jacobians are easily gained as below,

$$E'_k = \left.\frac{\partial f_k}{\partial w_k}\right|_{\hat{X}_{k|k}} = \left[\begin{matrix} \frac{\partial f_X(\hat{X}_k, u_k)}{\partial w_{X_k}} & \frac{\partial f_X(\hat{X}_k, u_k)}{\partial w_{Y_k}} & \frac{\partial f_X(\hat{X}_k, u_k)}{\partial w_{\psi_k}} \\ \frac{\partial f_Y(\hat{X}_k, u_k)}{\partial w_{X_k}} & \frac{\partial f_Y(\hat{X}_k, u_k)}{\partial w_{Y_k}} & \frac{\partial f_Y(\hat{X}_k, u_k)}{\partial w_{\psi_k}} \\ \frac{\partial f_\psi(\hat{X}_k, u_k)}{\partial w_{X_k}} & \frac{\partial f_\psi(\hat{X}_k, u_k)}{\partial w_{Y_k}} & \frac{\partial f_\psi(\hat{X}_k, u_k)}{\partial w_{\psi_k}} \end{matrix}\right]\Bigg|_{\hat{X}_{k|k}} = \left[\begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix}\right]\Bigg|_{\hat{X}_{k|k}}$$

$$F'_k = \left.\frac{\partial h_k}{\partial v_k}\right|_{\hat{X}_{k|k-1}} = \left[\begin{matrix} \frac{\partial h_X(\hat{X}_k, u_k)}{\partial v_{X_k}} & \frac{\partial h_X(\hat{X}_k, u_k)}{\partial v_{Y_k}} \\ \frac{\partial h_Y(\hat{X}_k, u_k)}{\partial v_{X_k}} & \frac{\partial h_Y(\hat{X}_k, u_k)}{\partial v_{Y_k}} \end{matrix}\right]\Bigg|_{\hat{X}_{k|k-1}} = \left[\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}\right]\Big|_{\hat{X}_{k|k-1}}$$

Since the matrices above are constant and time invariant, they do not need to be updated at each step. Matrices $A'_k$, and $B'_k$ are calculated as below,

$$A'_k = \left.\frac{\partial f_k}{\partial \hat{X}_k}\right|_{\hat{X}_{k|k}, u_k} = \left[\begin{matrix} \frac{\partial f_X(\hat{X}_k, u_k)}{\partial X_k} & \frac{\partial f_X(\hat{X}_k, u_k)}{\partial Y_k} & \frac{\partial f_X(\hat{X}_k, u_k)}{\partial \psi_k} \\ \frac{\partial f_Y(\hat{X}_k, u_k)}{\partial X_k} & \frac{\partial f_Y(\hat{X}_k, u_k)}{\partial Y_k} & \frac{\partial f_Y(\hat{X}_k, u_k)}{\partial \psi_k} \\ \frac{\partial f_\psi(\hat{X}_k, u_k)}{\partial X_k} & \frac{\partial f_\psi(\hat{X}_k, u_k)}{\partial Y_k} & \frac{\partial f_\psi(\hat{X}_k, u_k)}{\partial \psi_k} \end{matrix}\right]\Bigg|_{\hat{X}_{k|k}, u_k}$$

$$= \left[\begin{matrix} 1 & 0 & -V_k \times \Delta T \times \sin(\psi_k + \beta_k) \\ 0 & 1 & V_k \times \Delta T \times \cos(\psi_k + \beta_k) \\ 0 & 0 & 1 \end{matrix}\right]\Bigg|_{\hat{X}_{k|k}, u_k}$$

$$= \left[\begin{matrix} 1 & 0 & -V_k \times \Delta T \times \sin(\psi_{k|k} + \beta_k) \\ 0 & 1 & V_k \times \Delta T \times \cos(\psi_{k|k} + \beta_k) \\ 0 & 0 & 1 \end{matrix}\right]$$

$$B'_k = \left.\frac{\partial f_k}{\partial u_k}\right|_{\hat{X}_{k|k}, u_k} = \left[\begin{matrix} \frac{\partial f_X(\hat{X}_k, u_k)}{\partial V_k} & \frac{\partial f_X(\hat{X}_k, u_k)}{\partial \delta_k} \\ \frac{\partial f_Y(\hat{X}_k, u_k)}{\partial V_k} & \frac{\partial f_Y(\hat{X}_k, u_k)}{\partial \delta_k} \\ \frac{\partial f_\psi(\hat{X}_k, u_k)}{\partial V_k} & \frac{\partial f_\psi(\hat{X}_k, u_k)}{\partial \delta_k} \end{matrix}\right]\Bigg|_{\hat{X}_{k|k}, u_k}$$

$$= \left[\begin{matrix} \Delta T \times \cos(\psi_k + \beta_k) & \frac{\partial f_X(\hat{X}_k, u_k)}{\partial \delta_k} \\ \Delta T \times \sin(\psi_k + \beta_k) & \frac{\partial f_Y(\hat{X}_k, u_k)}{\partial \delta_k} \\ \Delta T \times \frac{\cos(\beta_k)}{l_f + l_r} \times \tan(\delta_k) & \frac{\partial f_\psi(\hat{X}_k, u_k)}{\partial \delta_k} \end{matrix}\right]\Bigg|_{\hat{X}_{k|k}, u_k}$$

where

$$\frac{\partial f_X(\hat{X}_k, u_k)}{\partial \delta_k} = \frac{\partial f_X(\hat{X}_k, u_k)}{\partial \cos(\psi_k + \beta_k)} \times \frac{\partial \cos(\psi_k + \beta_k)}{\partial \beta_k} \times \frac{\partial \beta_k}{\partial \delta_k}$$

$$= V_k \times \Delta T \times -\sin(\psi_k + \beta_k) \times \frac{\frac{l_r}{l_f + l_r}}{\left(1 + \left(\frac{l_r \times \tan(\delta_k)}{l_f + l_r}\right)^2\right) \times \cos^2(\delta_k)}$$

$$\frac{\partial f_Y(\hat{X}_k, u_k)}{\partial \delta_k} = \frac{\partial f_X(\hat{X}_k, u_k)}{\partial \sin(\psi_k + \beta_k)} \times \frac{\partial \sin(\psi_k + \beta_k)}{\partial \beta_k} \times \frac{\partial \beta_k}{\partial \delta_k}$$

$$= V_k \times \Delta T \times \cos(\psi_k + \beta_k) \times \frac{\frac{l_r}{l_f + l_r}}{\left(1 + \left(\frac{l_r \times \tan(\delta_k)}{l_f + l_r}\right)^2\right) \times \cos^2(\delta_k)}$$

$$\frac{\partial f_\psi(\hat{X}_k, u_k)}{\partial \delta_k} = \frac{1}{\cos^2(\delta_k)} \times \Delta T \times \frac{V_k \cos(\beta_k)}{l_f + l_r} + \Delta T \times \frac{V_k}{l_f + l_r} \times \tan(\delta_k) \times \frac{\partial \cos(\beta_k)}{\partial \delta_k}$$

$$= \frac{1}{\cos^2(\delta_k)} \times \Delta T \times \frac{V_k \cos(\beta_k)}{l_f + l_r}$$

$$+ \Delta T \times \frac{V_k}{l_f + l_r} \times \tan(\delta_k) \times -\sin(\beta_k) \times \frac{\frac{l_r}{l_f + l_r}}{\left(1 + \left(\frac{l_r \times \tan(\delta_k)}{l_f + l_r}\right)^2\right) \times \cos^2(\delta_k)}$$

Given the matrices above, it is possible to implement the Kalman filter algorithm now as below,

$$\delta x_{k|k-1} = A'_{k-1} \delta x_{k-1|k-1} + B'_{k-1} \delta u_{k-1}$$

$$P_{k|k-1} = A'_{k-1} P_{k-1|k-1} A'^{T}_{k-1} + E'_{k-1} Q_{k-1} E'^{T}_{k-1}$$

$$\delta x_{k|k} = \delta x_{k|k-1} + L_k\left(\delta y_{k_{meas}} - \delta y_{k|k-1}\right) = \hat{X}_{k|k} + L_k\left(\delta y_{k_{meas}} - C'_k \delta x_{k|k-1} - D'_k \delta u_k\right)$$

$$P_{k|k} = P_{k|k-1} - L_k C'_k P_{k|k-1}$$

where

$$L_k = P_{k|k-1} C'^{T}_k \left(C'_k P_{k|k-1} C'^{T}_k + F'_k R_k F'^{T}_k\right)^{-1}$$

The pseudocode of the implemented algorithm in MATLAB is given in Algorithm 1. It is notable that in this project, estimation is done offline after the measurement data is generated in Simulink.

***Algorithm 1. Linearized Kalman Filter***

1. Initialize the states
2. Calculate time-invariant Jacobian matrices $C_k^{'}$, $D_k^{'}$, $\mathrm{E}_{\mathrm{k}-1}^{'}$, and $\mathrm{F}_{\mathrm{k}}^{'}$
3. For all the measurement samples:
    a. Calculate Jacobian matrices $A_{k-1}^{'}$ and $B_{k-1}^{'}$
    b. Check the observability
    c. If observable:
        1. Conduct the model update
        2. Calculate Kalman gain $\mathrm{L_k}$
        3. Conduct the measurement update
    d. Else:
        1. Exit the code

***Algorithm 2. Extended Kalman Filter***

1. Initialize the states
2. Calculate time-invariant Jacobian matrices $\mathrm{C}_{\mathrm{k}}^{'}$, $\mathrm{E}_{\mathrm{k}-1}^{'}$, and $\mathrm{F}_{\mathrm{k}}^{'}$
3. For all the measurement samples:
    a. Calculate Jacobian matrix $A_{k-1}^{'}$
    b. Check the observability
    c. If observable:
        1. Conduct the model update
        2. Calculate Kalman gain $\mathrm{L_k}$
        3. Conduct the measurement update
    d. Else:
        1. Exit the code

## Extended Kalman Filter

The general concept of Extended Kalman Filter is the same as Linearize Kalman Filter except for the model and measurement update policies. For Extended Kalman Filter, the model and measurement updates are conducted based on the corresponding nonlinear function and linearization is only done for propagating the Gaussian noises. Thus, the matrices $A_k'$, $E_k'$, $C_k'$, and $F_k'$ are enough. These matrices are calculated exactly the same as before. The Extended Kalman filter algorithm can be written as follows,

$$\hat{X}_{k|k-1} = f_k\left(\hat{X}_{k-1|k-1}, u_{k-1}, 0\right)$$

$$P_{k|k-1} = A'_{k-1}P_{k-1|k-1}{A'}^{T}_{k-1} + E'_{k-1}Q_{k-1}{E'}^{T}_{k-1}$$

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + L_k\left(y_k - \hat{Y}_{k|k-1}\right) = \hat{X}_{k|k-1} + L_k\left(y_k - h_k\left(\hat{X}_{k|k-1}, u_k, 0\right)\right)$$

$$P_{k|k} = P_{k|k-1} - L_k C'_k P_{k|k-1}$$

$$L_k = P_{k|k-1}{C'_k}^{T}\left(C'_k P_{k|k-1}{C'_k}^{T} + F'_k R_k {F'_k}^{T}\right)^{-1}$$

Pseudocode of the Extended Kalman filter algorithm in MATLAB is given in Algorithm 2.

## Measurement Data Generation

As mentioned before, the RC car configuration is not ready to be implemented in this study. As a result, a higher fidelity Simulink model is created to generate the measurement data. RC car is modeled using the nonlinear kinematic equations introduced in previous part. Parameters $l_f \; and \; l_r$ are considered to be 16 cm and 14 cm respectively which differs from the corresponding values considered in both linearized and extended Kalman Filter models. To extract the measurement data, it is assumed that Marvelmind indoor GPS is able to evaluate the position X and Y of the RC car with a specific precision which would be discussed.

### Simulink Model

Within this Simulink model, the system incorporates two synthetic vectors representing velocity and steering angle as inputs. These control inputs are assumed to be derived from the control unit, which utilizes sensor feedback and operates within a closed-loop system. Subsequently, the control inputs are applied to the model, resulting in the RC car following a predetermined trajectory on the track. The trajectories of the system's states are saved in the MATLAB workspace, serves as an actual values for comparison with the results obtained from the Kalman Filters.

### Uncertainties

Model uncertainties are considered as follows:

1. Uncertainties due to model linearization of higher fidelity model which is a nonlinear kinematic model in linear Kalman filter case. 2. Model parameters mismatch between higher fidelity model and Kalman filters. 3. Uncertainties on initial condition values.

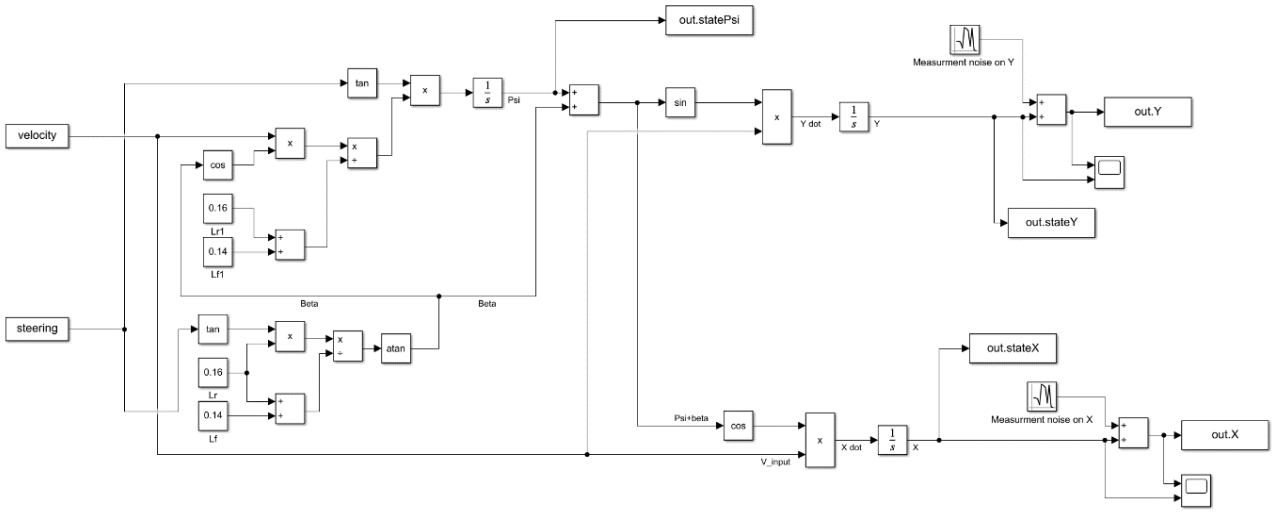


Figure 2. Simulink Model Implemented to Extract States Trajectory and Sensors Outputs.

### Velocity and Steering Profile

To simulate the RC car's trajectory along a predefined lap, synthetic input vectors of velocity and steering angle are generated. These vectors adhere to certain bounds: a maximum velocity of 2.5 m/s and a steering angle range of ±12°. The velocity limit is set to ensure that the low-speed assumption remains valid throughout the entire simulation. It is important to note that the steering angle bound is chosen to satisfy both the physical constraint imposed by the vehicle's maximum steering angle. Also the steering angle rate of change is simulated carefully so that the turning radius within the track is maintained at over 2 meters. Consequently, it is possible to assume that the impact of lateral forces on the tires can be considered negligible.

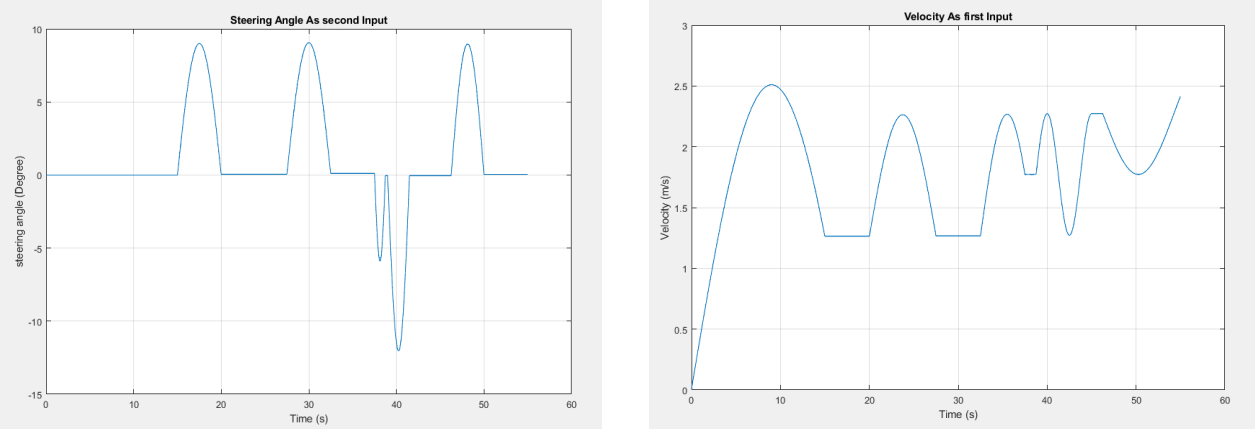


Figure 3. Steering Angle and Velocity Inputs Profile.

### Marvelmind Indoor GPS Specification

As can be seen from the sensor specification sheet, absolute precision is dependent on distance from the beacons. In the laboratory we aim to install 4 beacons on each wall. As a result roughly we can assume the beacons can read position data at maximum distance of 3 m. This will indicate that the error mean is in range of 1-9 cm which is the huge amount of error. Here it is assumed that error mean is 0.01. Also, differential precision of the sensor is $\pm 2$ cm which indicates that the variance of error is about 0.0004. Considering this information, a random normally distributed noise with the mean of 0.01m and variance of 0.0004 is added to the output signal of X and Y to simulate the sensor measurements.

| Parameter | Technical Specifications |
|---|---|
| Distance between beacons | - Reaches up to 50 meters and up to 100 meters with horn under laboratory conditions (Mini-RX or Super-Beacon to Super-Beacon with RX4 only)<br>- Recommended distance is 30 meters (Transducer4 on the first beacon is looking straight at the Transducer4 on the second beacon, other transducers are switched off) |
| Coverage area | - Reaches up to 1000m$^2$ with the Starter Set configurations<br>- Coverage for larger territories is provided using submap – like cells in cellular networks |
| Location precision | - Absolute: 1–3% of the distance to the beacons<br>- Differential precision: ±2cm |
| Location update rate | - 1/20Hz to 25Hz (Ultrasonic based only)<br>- 100Hz with ultrasonic + IMU fusion enabled (Only for Beacons HW v4.9-IMU-Discontinued)<br>- Can be set manually via Dashboard software<br>- Depends on the distance between mobile and stationary beacons (shorter distance—higher update rate)<br>- Depends on the number of mobile beacons (Non-Inverse Architecture; for Inverse Architecture no such dependency)<br>- Depends on the radio profile (500kbps, 153kbps, 38kbps)<br>- Slightly depends on the number of stationary beacons—dependence is not the same as for mobile beacons |

Figure 4. Marvelmind Indoor GPS Precision Specification.

### Measurement output for X and Y position

Finally, simulation results can lead to sensors output which also contains the noises considered with respect to Marvelmind indoor GPS spec sheet. It is further assumed that the sensors can determine the X and Y positions within a certain range of error, which is simulated using non-zero mean Gaussian noise. Figure 5 and Figure 6 illustrate the sensor output for X and Y measurements.

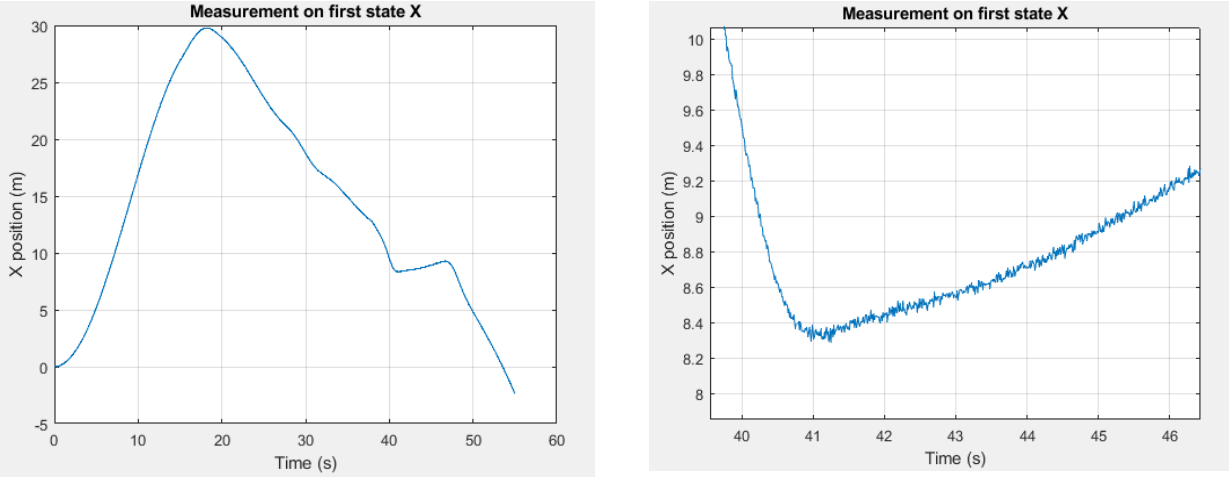


Figure 5. Sensor Output for X Measurement (Left). Detail View of Measurement Noise (Right).

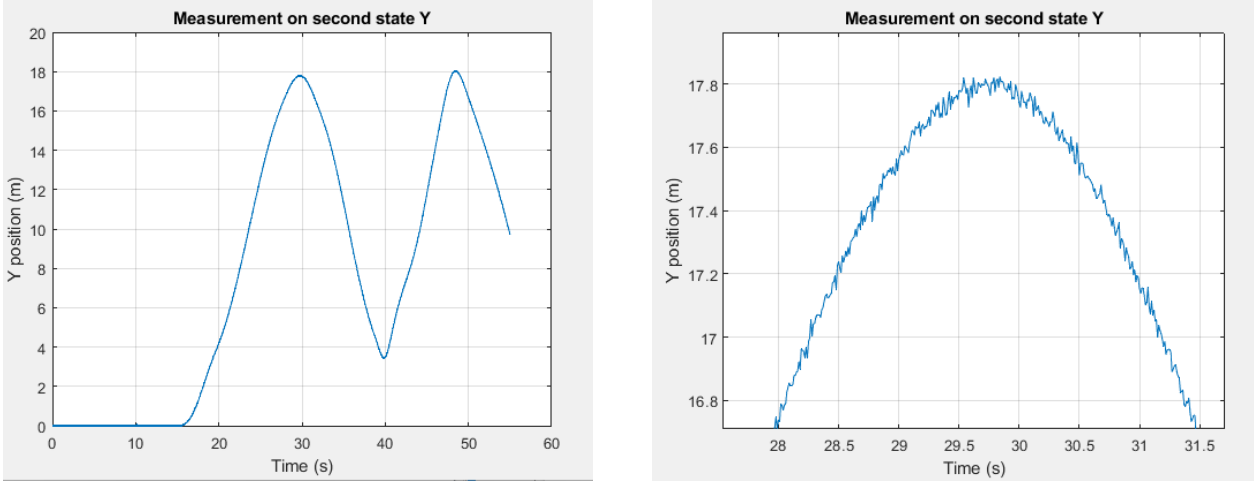


Figure 6. Sensor Output for Y Measurment (Left). Detail View of Measurement Noise (Right).

## RC Car Trajectory Extracted by Sensor Measurement

Here is the simulated track map and sensor measured position after running the simulation with synthetic input vectors.

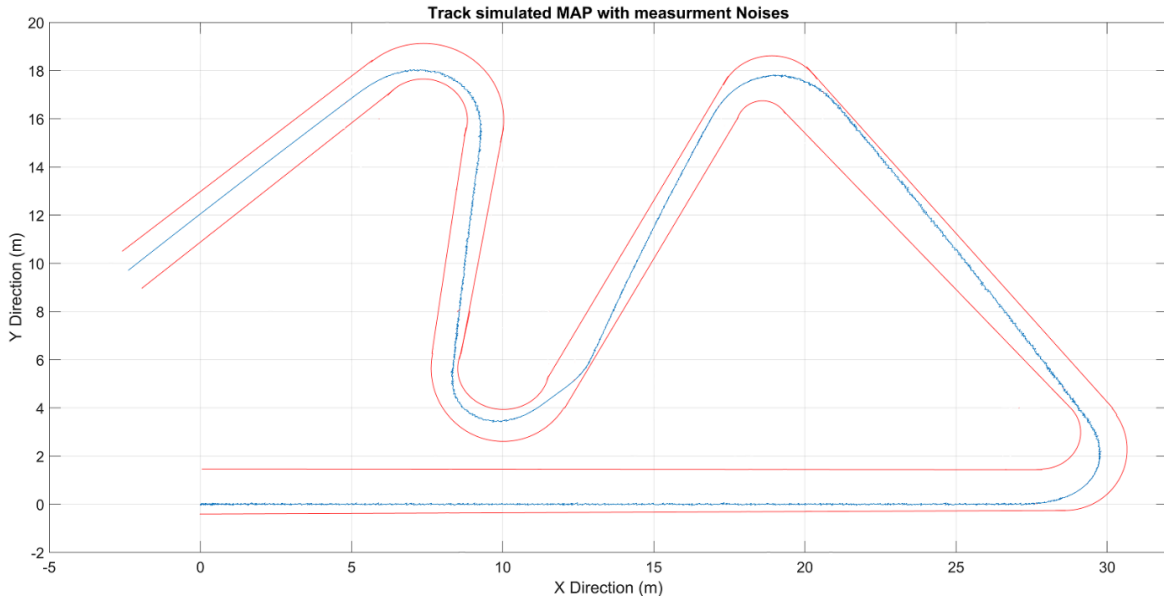


Figure 7. Simulated track and Measured position of RC car.

## Trajectory of Unmeasured state (Heading Angel)

Desired heading angle is plotted here which is the output of higher fidelity model simulation under influence of synthetic inputs. This profile can be considered as the actual value of third state when we want to calculate the Kalman filter error.

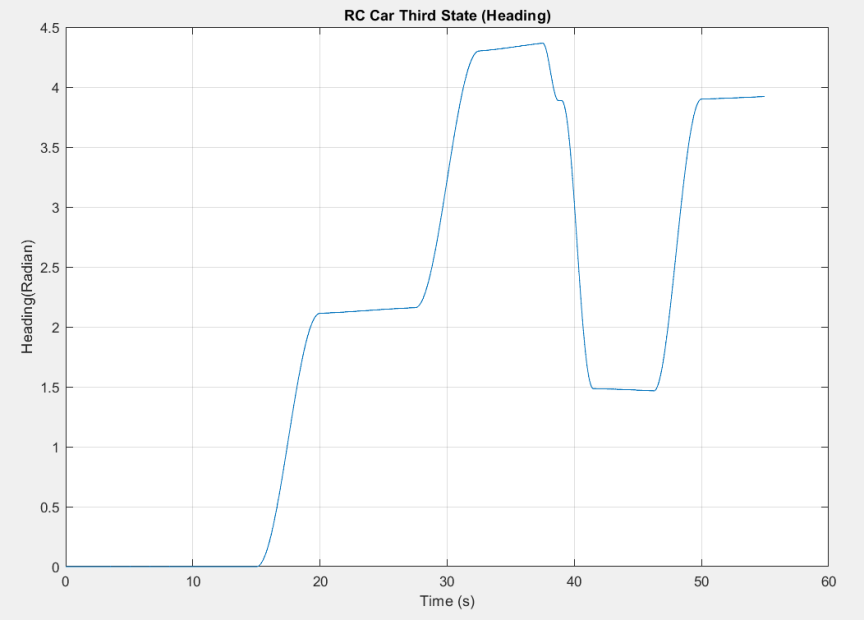


Figure 8. Unmeasured Heading Angel Profile Which should be Estimated by LKF and EKF.

### Low Frequency Simulation

The major problem with Marvelmind indoor GPS is its low sampling frequency. The desired sampling frequency for the purpose of this project is 100 Hz while the sampling frequency of the GPS is 25 Hz. To simulate this, the sampling frequency of the Simulink is set to 100 Hz, meaning that the states, inputs, and outputs are all calculated at 100 Hz. However, inside the MATLAB code, the measurement data is again sampled at 25 Hz, i.e., measurement data can be extracted after every 4 steps.

For measurement update in Kalman filter algorithm, measurement data is needed for every step. Since there is no data available for each step at this project, different approaches can be used. The approach used in this project is to use the last measured data for the steps with no data available. The algorithm used for implementing the explained procedure is given in Algorithm 3.

One very simple approach, however, is not to have measurement update at the steps lacking measurement data. So, the measurement correction happens every 4 steps once. Another approach is to extrapolate the input for the steps lacking measurement data based on the previous measurement data points. This approach probably can lead to more accurate results and is worth considering in future work.

***Algorithm 3. Measurement Data Sampling***

1. Load the measured data from Simulink at 100 Hz, named y_meas
2. Use y = y_meas(1)
3. If k is multiple of $\frac{100}{25} = 4$, i.e., every 4 steps
    a. Update y: y = y_meas(k)
4. Implement Kalman filter
5. Go to step 3

## Results and Discussion

To evaluate the performance of the designed Linearized and Extended Kalman filters, they are implemented in MATLAB using the data generated in Simulink as explained in previous section. The evaluation experiment is designed as shown in Figure 9. The input $u$ and measurement $y_{meas}$ provided by the Simulink are given to the Algorithm 1 and Algorithm 2, and the estimated states $\hat{X}$ are calculated. Then, these estimated states are compared to the actual states $X$ obtained from simulation.

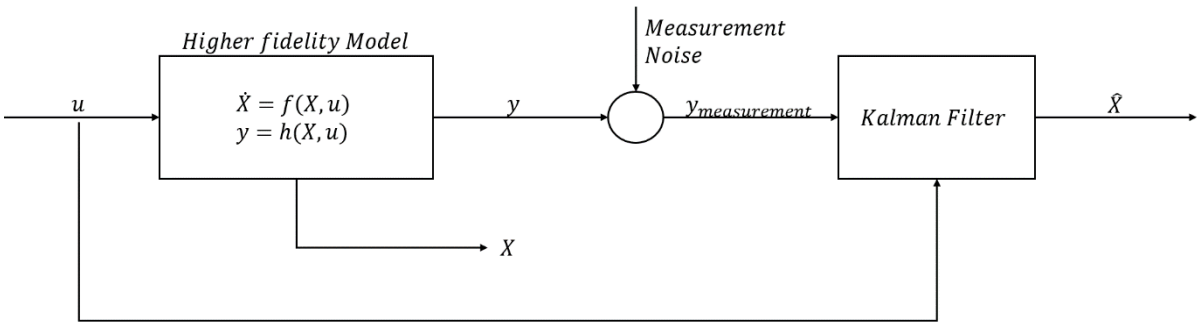


Figure 9. Schematic of Kalman filter evaluation experiment.

Note that the main problem of indoor GPS is its low sampling rate. The experiment should simulate this condition as well. For this purpose, the simulation is run at a frequency of 100 Hz. However, the measurement data provided by simulation is sampled at 25 Hz in MATLAB to provide the sensor measurement at 25 Hz, i.e., the measurement data is available every 0.04 s. The discretization time step in MATLAB is 0.01 s. Thus, there is no measurement data for the time steps in-between. In this paper, the measurement data for the intermediate steps are assumed to be constant and the same as the last available measurement data, i.e., measurement data is updated every 4 steps. However, the accuracy of the estimation can be enhanced by extrapolating the intermediate measurement data based on the last available datapoints.

The remainder of this section is organized as follows: first, the results for Linearized Kalman filter and Extended Kalman filter are provided, and they are compared to the actual states individually. Then, they are compared to each other and to the open-loop estimator. Finally, the errors of Extended Kalman filter are compared for measurement sampling rate of 25 Hz and 100 Hz.

**Linearized Kalman filter**

Since the statistics of the noise in unknown, the covariance matrices $R$ and $Q$ are chosen based on trial and error. A good initial guess for $R$ is the noise variance defined in sensor spec. Initially, since there are noises in the measurement, model can be more worthy than the measurement for $X$ and $Y$, i.e., the corresponding variances in $Q$ are smaller than those of in $R$. However, since there is no measurement for $\psi$, it is reasonable to assume that its variance which is the corresponding element in $Q$ is very smaller than that of $X$ and $Y$, i.e., the model is more reliable for estimating $\psi$. The noises are initialized as below,

$$[w]_{3\times1} \sim N\left(0\,,\begin{bmatrix} 3\times10^{-5} & 0 & 0 \\ 0 & 3\times10^{-5} & 0 \\ 0 & 0 & 1\times10^{-5} \end{bmatrix}\right)$$

$$[v]_{2\times1} \sim N\left(0\,,\begin{bmatrix} 4\times10^{-4} & 0 \\ 0 & 4\times10^{-4} \end{bmatrix}\right)$$

The estimated states of the Linearized Kalman filter for these specifications are given in Figure 10. As can be seen, the estimated $\psi$ has significant deviation from the actual one. This is reasonable based on the violations of Kalman filter assumptions. In Kalman filter, the system is assumed to be linear, and noises are assumed to be Gaussian with certain statistics.

Both assumptions are violated in this problem. However, Kalman filter still is the best linear observer, and its performance can be enhanced by tuning its parameters.

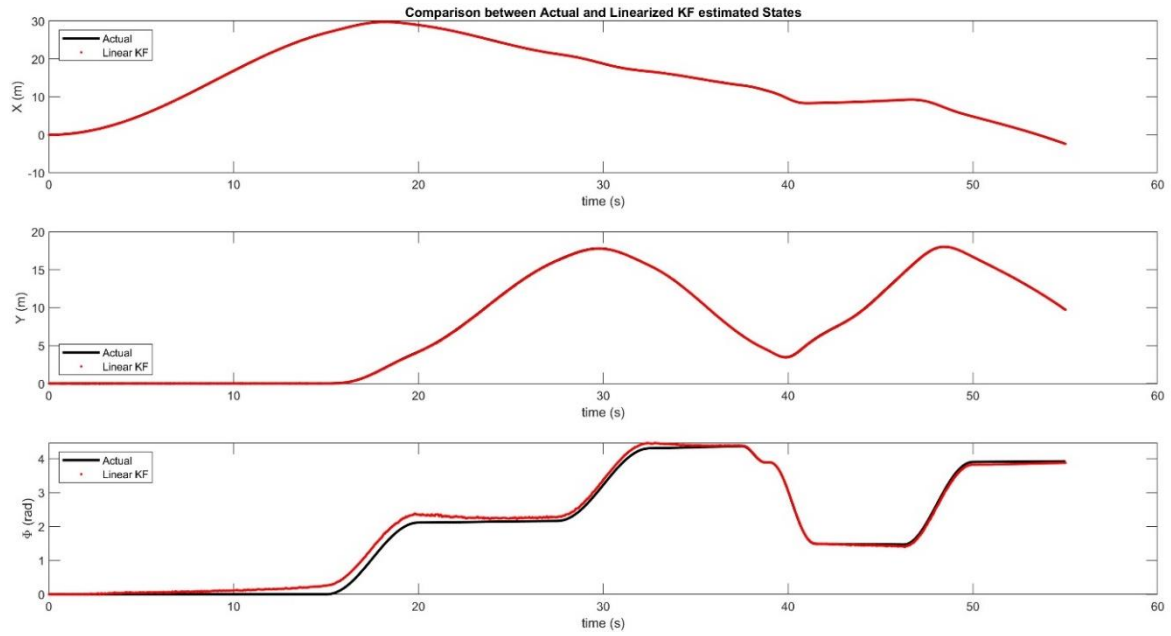


Figure 10. Linearized Kalman filter estimated states compared to the actual states.

The hyperparameters for tuning the Kalman filter are $R$ and $Q$. After the process of trial and error, the specifications below resulted in acceptable results,

$$[w]_{3\times1} \sim N\left(0\,,\ \begin{bmatrix} 3\times10^{-5} & 0 & 0 \\ 0 & 3\times10^{-5} & 0 \\ 0 & 0 & 1\times10^{-7} \end{bmatrix}\right)$$

$$[\mathrm{v}]_{2\times1} \sim \mathrm{N}\left(0\,,\ \begin{bmatrix} 4\times10^{-5} & 0 \\ 0 & 4\times10^{-5} \end{bmatrix}\right)$$

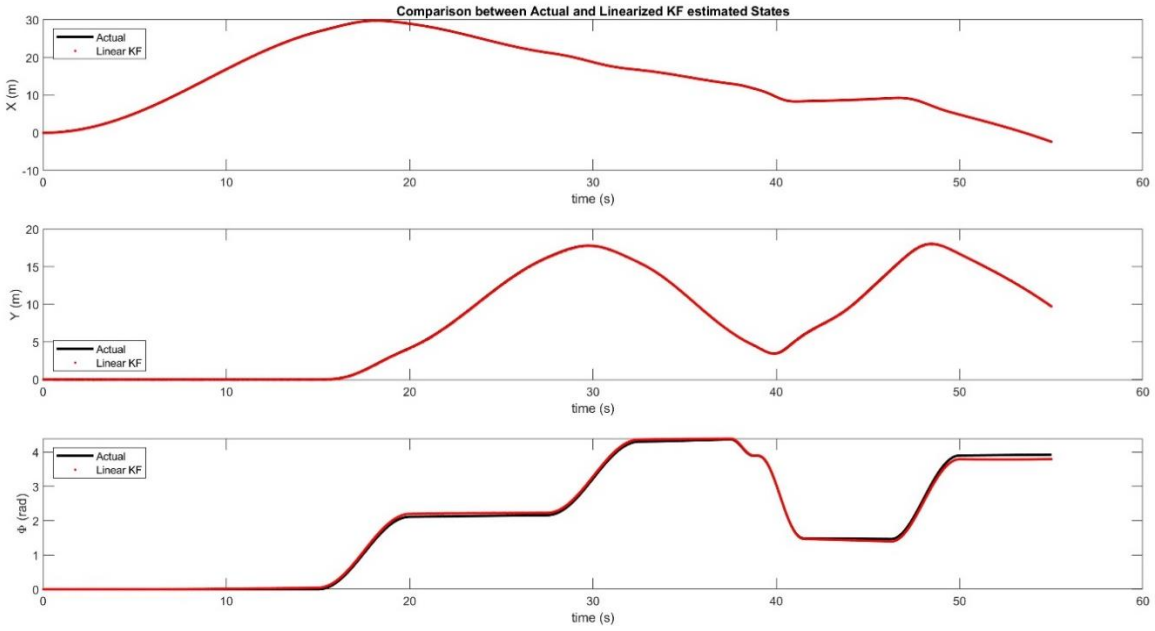


Figure 11. Linearized Kalman filter estimated states compared to the actual states after tuning the hyperparameters.

The results are shown in Figure 11. Obviously, there has been a significant improvement in performance.

Comparison of these $R$ and $Q$ with the previous ones demonstrates that measurement almost as important as the model for estimating $X$ and $Y$ while the model is even more reliable for estimating $\psi$ compared to its previous variance.

The estimation error and error variance are given in Figure 12 and Figure 13 respectively. As can be seen, the errors belonging to the states $X$ and $Y$ are small while that of $\psi$ is very large (about 7.4°). Also, the variance of estimation error for states $X$ and $Y$ quickly converges while that of $\psi$ is fluctuating. Thus, it is reasonable for the estimated $\psi$ to deviate from the actual one at the final steps as seen in Figure 11.

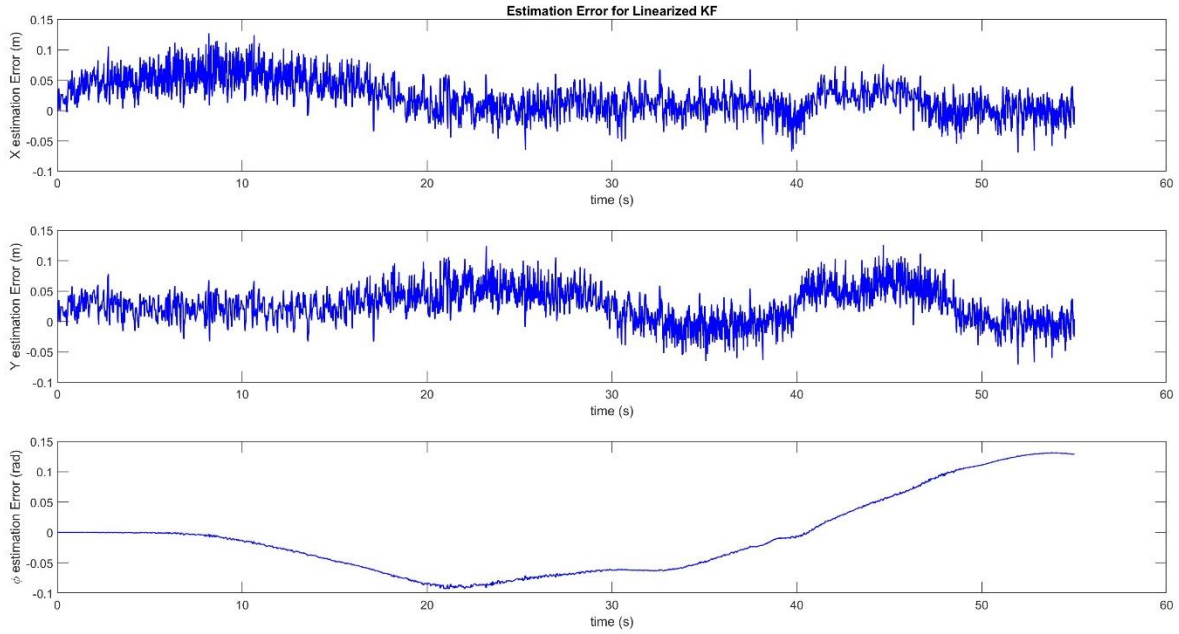


Figure 12. Linearized Kalman filter estimation error for each state.

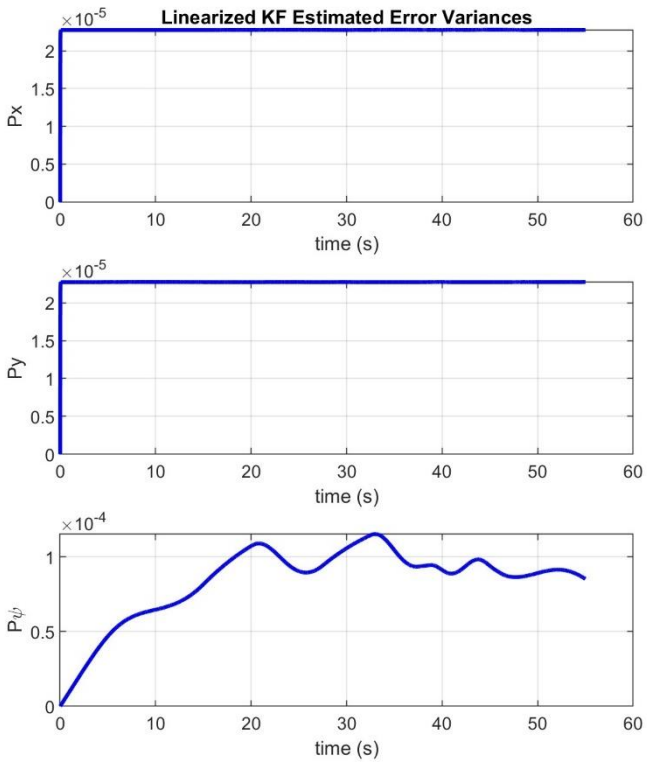


Figure 13. Linearized Kalman filter error variance for each state.

Although the results are satisfactory, it should be studied whether the Kalman filter is behaving as expected. The distribution of the estimation error is illustrated in Figure 14. As expected, the distributions do not match the Normal distribution. The reason is that the real noise in simulations is not zero-mean Gaussian. Also, as sensor measurements are considered to be constant in every 4 steps, measurement error is not independent any more. Furthermore, the distribution of error cannot be considered as Gaussian inside the simulation.

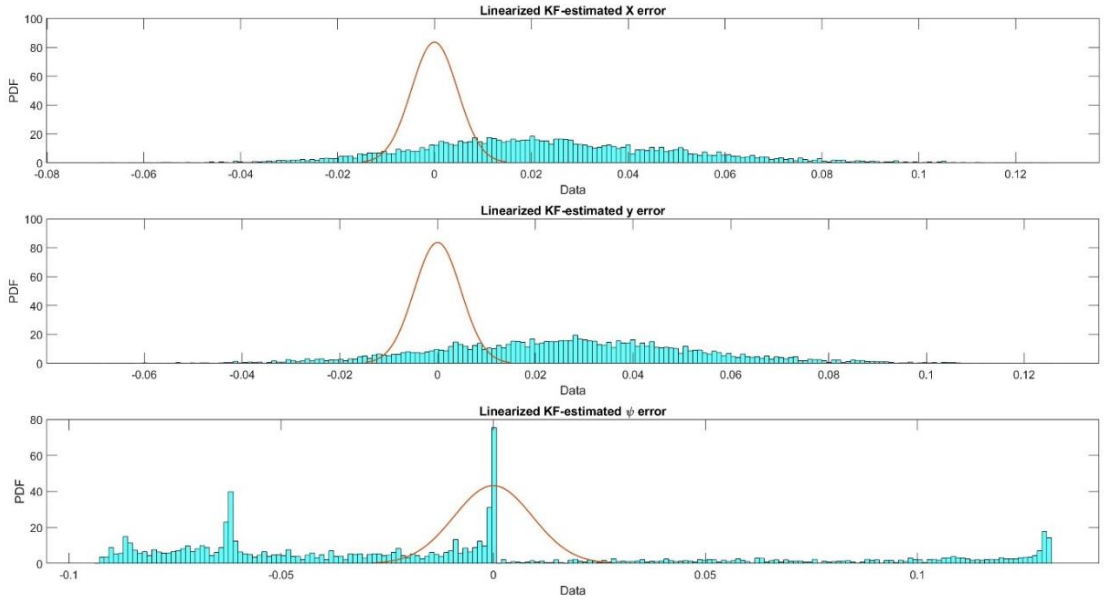


Figure 14. Distribution of the estimated error for Linearized Kalman filter. Red curve is the Normal distribution with zero mean and the variance of the last step's $P_{k|k}$. The histogram denotes the error distribution of the Linearized Kalman filter.

**Extended Kalman Filter**

The initialization for Extended Kalman filter is done the same as the Linearized Kalman filter. The result for the spec below is given in Figure 15. AS shown, the estimated states with Extended Kalman filter are properly matching the actual ones. This is reasonable since the nonlinearity of the system is taken into account in this algorithm. Also, it is noticeable that Extended Kalman filter is working appropriately with the covariances matrices which resulted in poor performance for Linearized Kalman filter at the initial guess step. This demonstrates that in Extended Kalman filter, one can trust on the model more (larger R), which is intuitively inferable since the model is not linearized in this algorithm.

$$[w]_{3\times1} \sim N\left(0\,,\ \begin{bmatrix} 3\times10^{-5} & 0 & 0 \\ 0 & 3\times10^{-5} & 0 \\ 0 & 0 & 1\times10^{-5} \end{bmatrix}\right)$$

$$[\mathrm{v}]_{2\times1} \sim \mathrm{N}\left(0\,,\ \begin{bmatrix} 4\times10^{-4} & 0 \\ 0 & 4\times10^{-4} \end{bmatrix}\right)$$

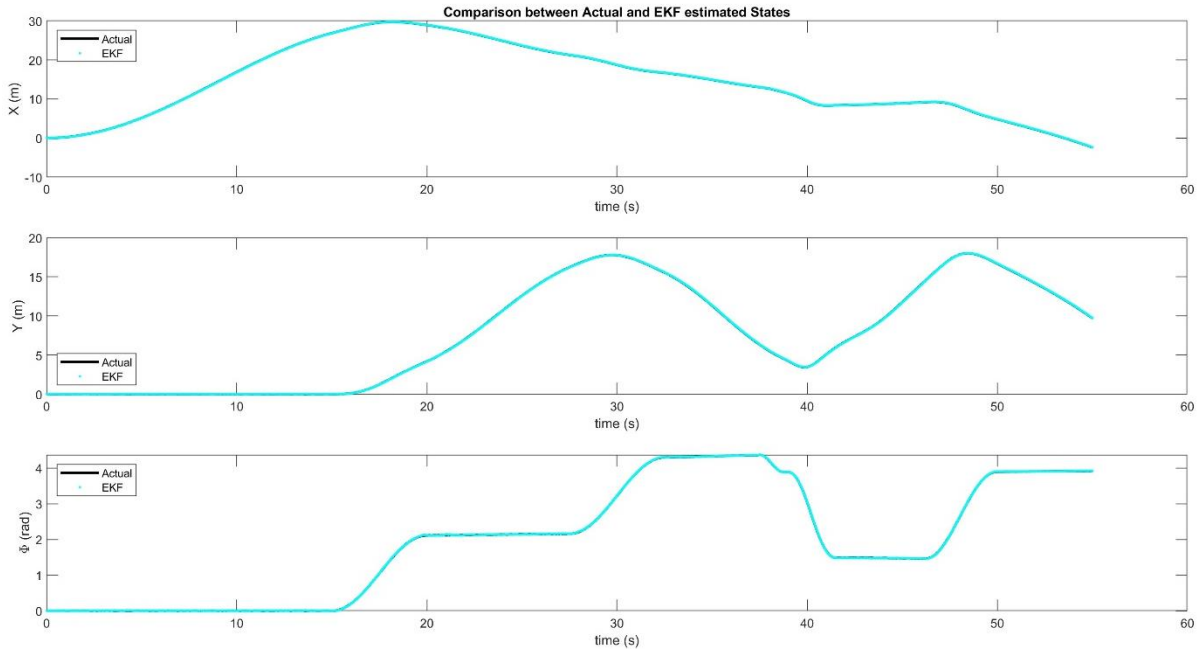


Figure 15. Extended Kalman filter estimated states compared to the actual states.

The estimation errors, their variances, and their distributions are shown in Figure 16, Figure 17, and Figure 18, respectively.

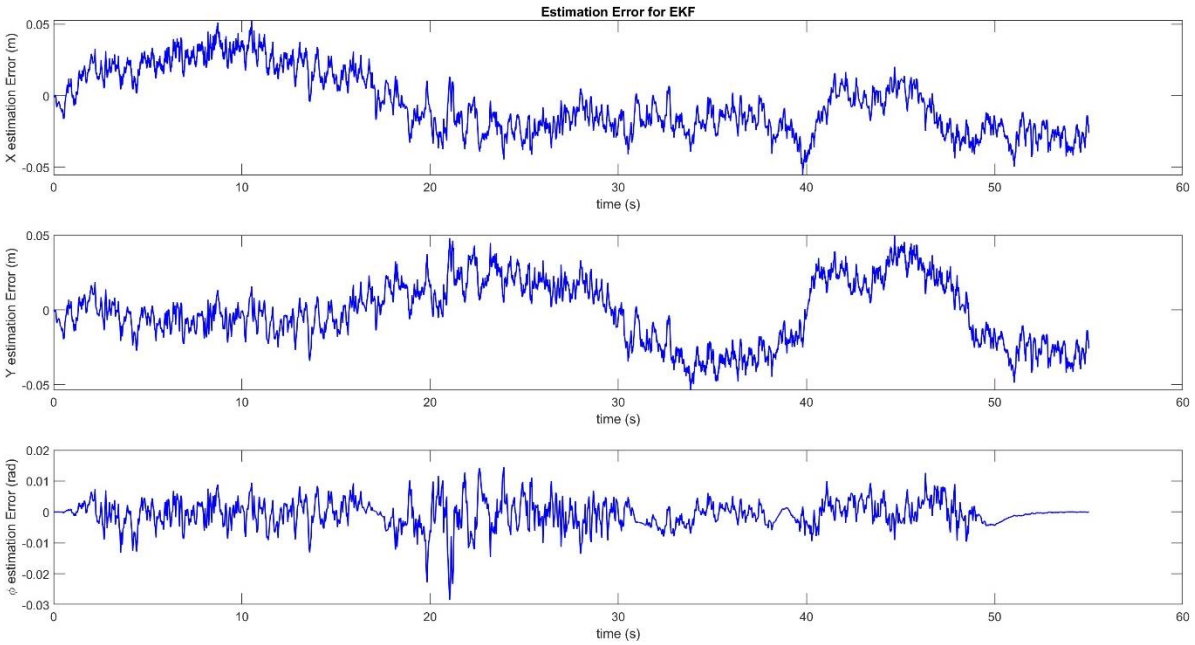


Figure 16. Extended Kalman filter estimation error for each state.

As can be seen, the estimation error is much smaller than that in Linearized Kalman filter, especially for $\psi$.

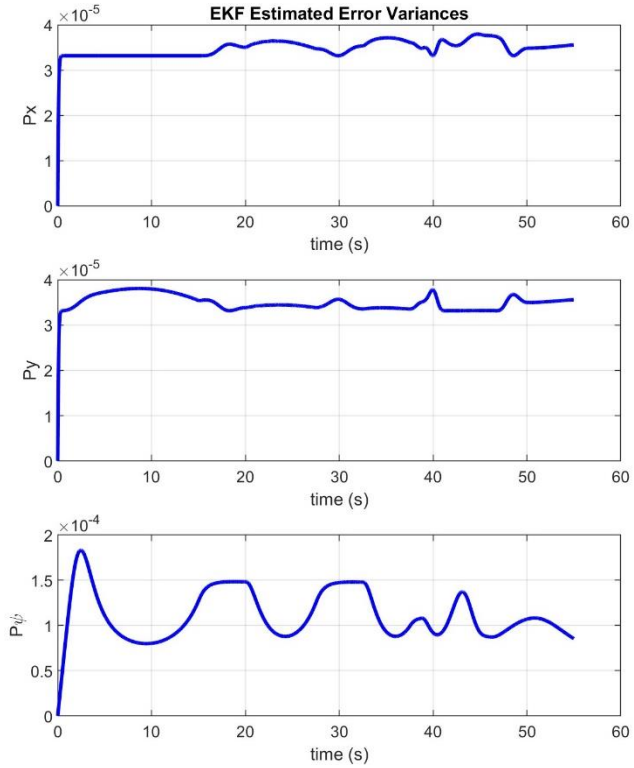


Figure 17. Extended Kalman filter error variance for each state.

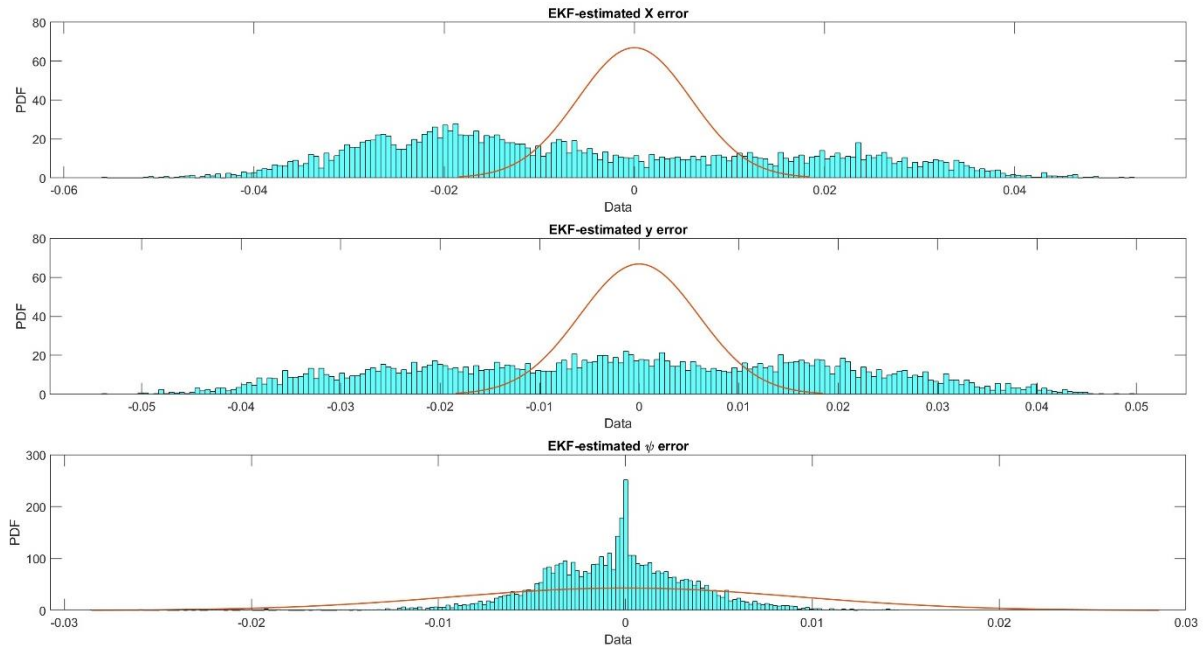


Figure 18. Distribution of the estimated error for Extended Kalman filter. Red curve is the Normal distribution with zero mean and the variance of the last step's $P_{k|k}$. The histogram denotes the error distribution of the Extended Kalman filter.

Figure 19 is a comparison between actual, Linearized Kalman filter, and Extended Kalman filter estimated $X$ and $Y$. It is the road map of the vehicle based on the estimated and actual values. As can be seen, both Linearized and Extended Kalman filter are operating properly, and the performance of the Extended Kalman filter is even better than that of the Linearized one. At some points, the deviation of the Linearized Kalman filter is significant. The reason might be the increase in process noise at those points which is majorly caused by linearization, i.e., the linear model is not that accurate at those points.

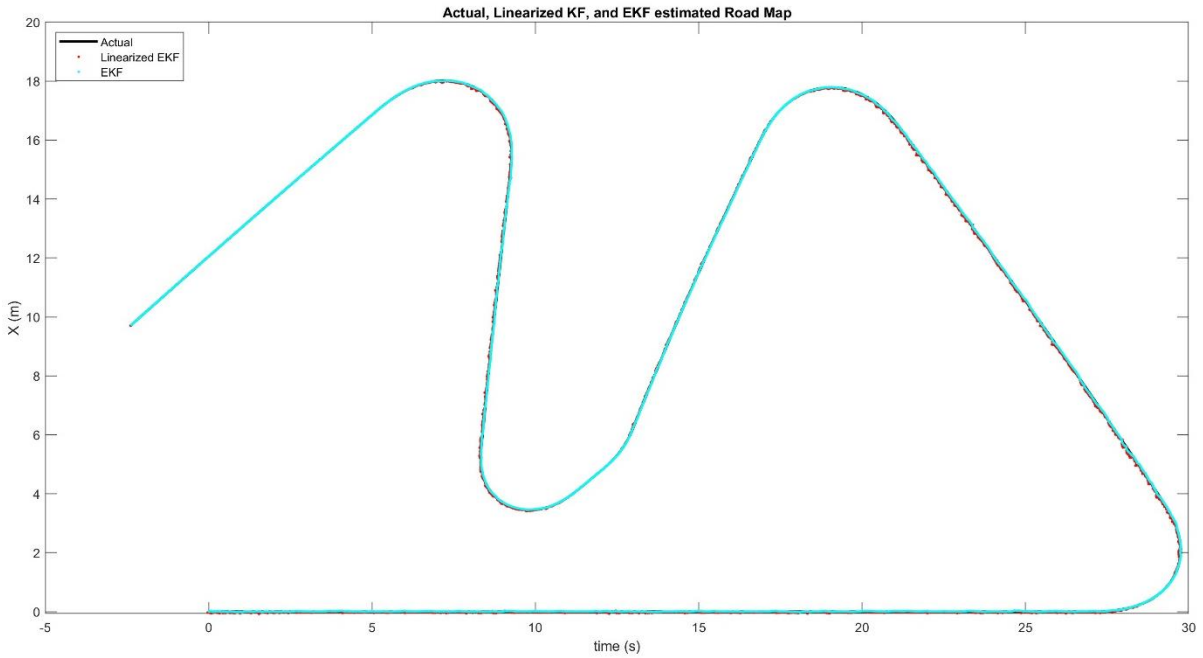


Figure 19. Road Map created from the states.

From the distributions it is inferable that Kalman filter is not behaving as expected. However, it is still a good observer, and it is preferred to open-loop system. Figure 20 shows a comparison between the actual states, open-loop, Linearized Kalman filter, and Extended Kalman filter estimations.

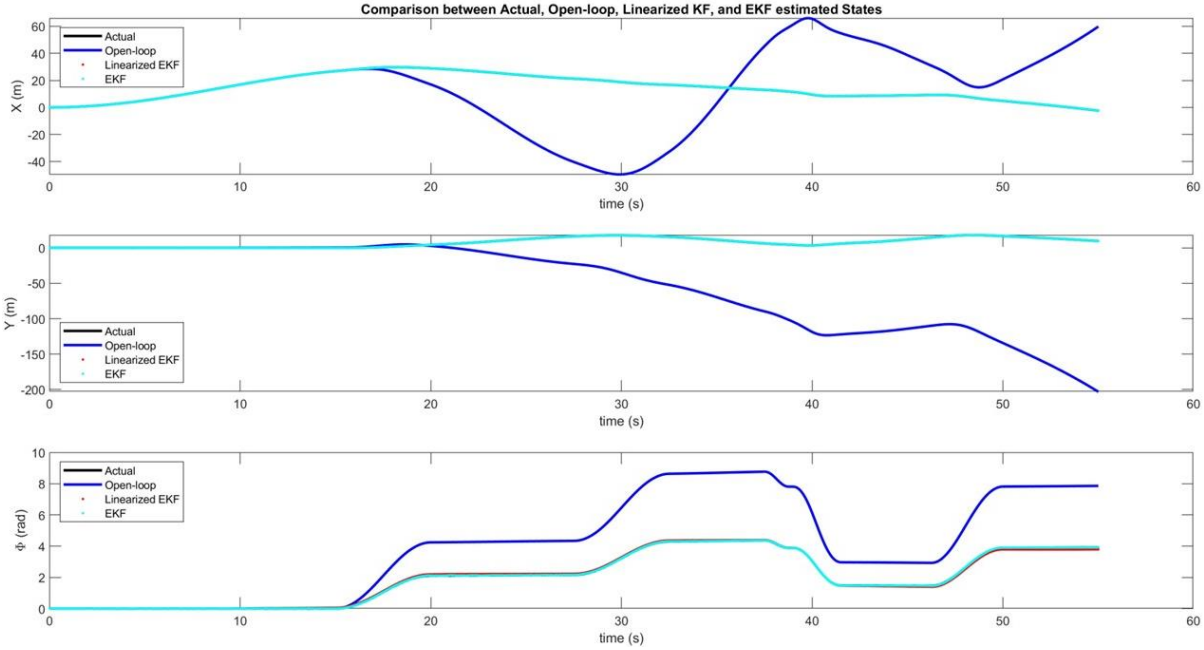


Figure 20. Comparison between the actual states and estimated states from open-loop, Linearized and Extended Kalman filter.

As shown, the open-loop states are way off the actual values. This confirms that even having Linearized Kalman filter is better than not having it at all.

Based on the results, it is fair to say both algorithms are working properly, but the Extended Kalman filter is preferrable because of its better performance and less computation effort.

Finally, it is crucial to check if the ultimate goal of this project has been reached. The final goal of the project is to use Kalman filter to compensate for the low frequency of the GPS. To evaluate this, the same Extended Kalman filter is used once again with the original measurement data from Simulink (data with frequency of 100 Hz), and the estimation errors are compared to those of the Extended Kalman filter with the measurement data at 25 Hz. The result is shown in *Figure 21*. As can be seen, for $X$ and $Y$ the error at 100 Hz is way better than those of at 25 Hz. But the estimation error for $\psi$ is almost the same for both cases. This is reasonable since there is no measurement for $\psi$. Thus, the Kalman filter is trusting on the model for estimating $\psi$ more, and the measurement frequency is not affecting the error significantly. However, the estimation error at both cases is much better than the open-loop case. Thus, using GPS even with poor performance, is better than not having it at all.

Other thing that can be inferred based on *Figure 21* is that at 100 Hz, the estimation error seems to have Gaussian distribution, unlike at 25 Hz. This can be caused by violating the independent noise assumption in Kalman filter theory. Since the measurement is assumed to be constant for the time steps lacking measurement, the noise at these time steps is the same and is not independent. This can affect the performance of Kalman filter.

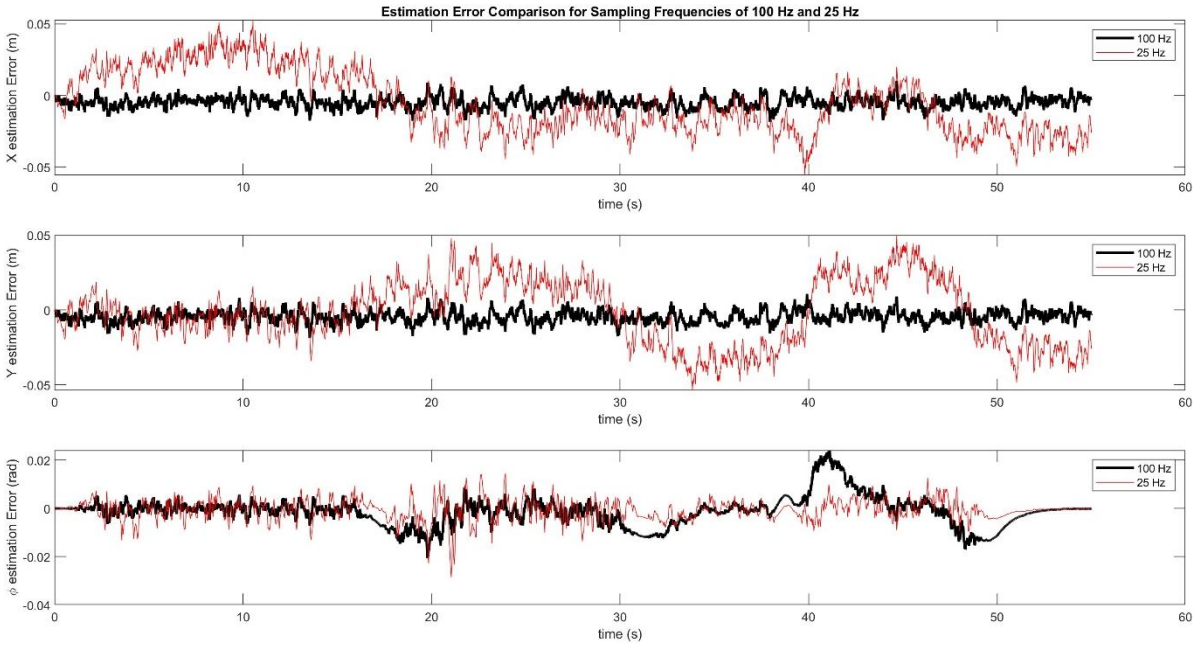


Figure 21. Estimation error of the Extended Kalman filter for measurement at 25 Hz compared to that of at 100 Hz.

Moreover, the performance of EKF can be compared with the case with no estimator, i.e., trusting on the sensor measurements. The error in this case is the deviation of the measurement from actual states. This error is compared with the EKF estimation error at Figure *22*. As shown, the estimation error of EKF is much smaller than the error of the sensor measurement. The measurement error demonstrates the error in the case no estimator is used. In that case, there is no measurement for $\psi$, and it must be either estimated or calculated open loop, which will result in a huge deviation from the actual $\psi$, as shown in *Figure 20*. These results prove that it is feasible to implement EKF for the purpose of this project.

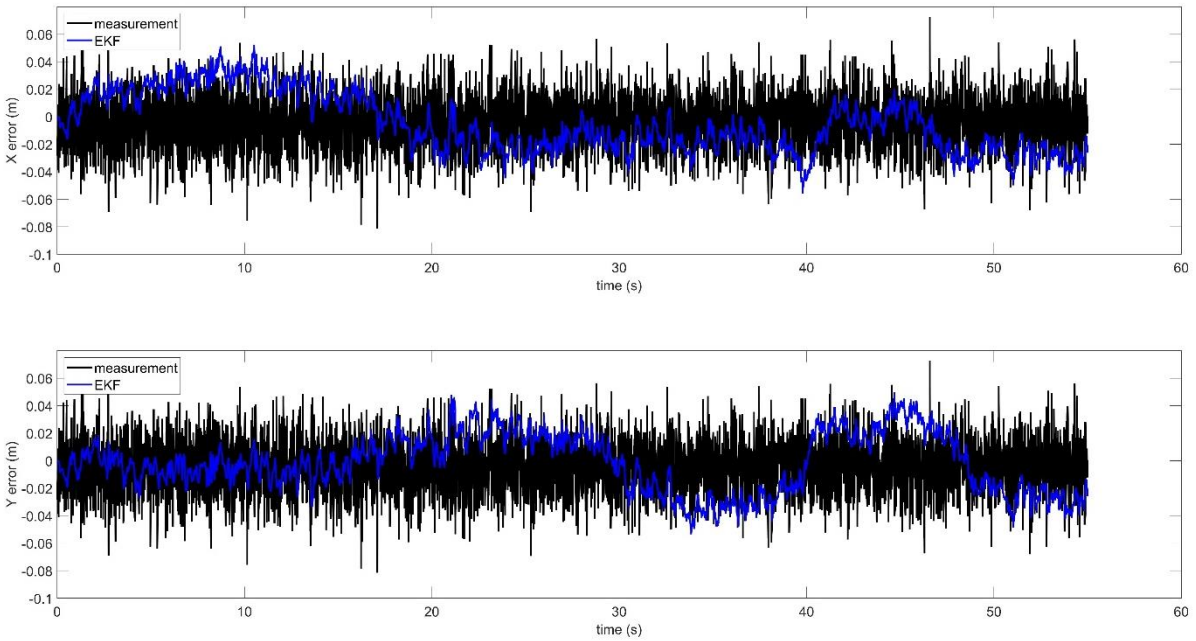


Figure 22. Comparison between EKF estimation error and the sensor measurement error.

## Conclusion

The focus of this project was on designing and implementing Kalman filter for the RC car with the purpose of enhancing the accuracy of the states and compensating for the low sampling frequency of the sensor. The model used for this purpose was kinematic bicycle model, where the speed is assumed to be small, and the tire slip is neglected. Nonlinearity of the system resulted in the need for designing Linearized or Extended Kalman filter. At this project, both algorithms were implemented. Data needed for the implementation was generated in Simulink using the Kinematic bicycle model as the higher fidelity model. The uncertainty sources were caused by linearizing the model, uncertainty in initial conditions and system parameters, and sensor noise. Because of the unknown statistics of noises, Kalman tuning was necessary for the Linearized Kalman filter. All in all, the results proved that EKF performs more accurately compared to Linearized KF as expected, and using EKF is beneficial compared to only trusting on the measurements.

References

[1] R. Rajamani, “Vehicle Dynamics and Control,” Springer US eBooks, Jan. 2012, doi: 10.1007/978-1-4614-1433-9.