BACHELOR

Path reconstruction using IMU and wheel sensors for parking maneuvers

Möhle, Levi S.

Award date:
2022

Department of Mechanical Engineering

4WC09 Bachelor Final Project

Path reconstruction using IMU and wheel sensors for parking maneuvers

DC 2022.056

L.S. Möhle 1418858

dr. ir. T.P.J. van der Sande Project coordinator

June 27, 2022

Abstract - *In this project, an extended Kalman filter (EKF) was implemented to a four-wheel vehicle to estimate the pose of the vehicle with respect to a certain reference frame, using only IMU and wheel sensors. The filter makes use of a kinematic model of a four-wheel vehicle, developed from the bicycle model extended into a two-track model. The EKF estimates the states x, y, ψ and $\dot{\psi}$ and is tuned on a circular trajectory, minimizing the root mean square error (RMSE) on the absolute distance from the reference frame r and on ψ. The tuned filter is tested on a circular trajectory and a parallel parking trajectory. These tests were both performed on an EKF estimating with the same car model as is used for the true pose and a simplified model, to test the filters capability of estimating based on a wrong model. The filter using a perfect model was able to estimate the states well with small RMSE scores on both trajectories. Using a simplified model resulted in big RMSE scores on the circular trajectory, but small again for the parallel parking trajectory. However, the second trajectory had a smaller time window, such that errors could not grow bigger over time. So both time window and quality of the prediction model influence the performance of the EKF.*

Contents

List of symbols

Symbol	Quantity	Unit	Abbreviation
β	Side slip angle	Radians	rad
δ_f	Input steering angle	Radians	rad
$\dot{\psi}$	Angular velocity	Radians per second	rad/s
ψ	Radial position	Radians	rad
A	State matrix	-	-
B	Input matrix	-	-
dt	Time step	Seconds	s
e	Error vector	-	-
f	Model function	-	-
H	Measurement matrix	-	-
h	Measurement function	-	-
L	Wheelbase	Meter	m
l_f	Distance between front axle and CoG	Meter	m
l_r	Distance between rear axle and CoG	Meter	m
R	Turning radius	Meter	m
r	Absolute distance from reference frame	Meter	m
r_t	Tire radius	Meter	m
t_w	Track width	Meter	m
v	Input velocity	Meter per second	m/s
v	Measurement noise vector	-	-
w	Process noise vector	-	-
X	State vector	-	-
x	Position in the x-direction	Meter	m
y	Position in the y-direction	Meter	m
z	Measurement vector	-	-

1 Introduction

Autonomous vehicles are the next big step in the automotive industry. They are considered as a safer and quicker way of transport, comfortable, more accessible, efficient and environmentally friendly. The development of autonomous vehicles has increased significantly in the last ten years, with big companies such as Waymo and Uber spending a lot of hours on developing safe autonomous cars. This development has resulted in systems such as Lane Keeping Assist or Adaptive Cruise Control, where a car keeps track of the surroundings and can perform small maneuvers. Parking has been made autonomous as well. For example, with Active Park Assist 2.0 from Ford [1] one can park with the simple click of a button or with Remote Smart Parking Assist from Hyundai [2] you can park the car from the outside. Besides being fully aware of its surroundings at all times, what is very important for all these techniques, is that is should accurately know its position with respect to a set path with the available sensors.

1.1 Related literature

A very popular sensor is the Global Positioning System (GPS) sensor, which regularly updates the position of an object into a known reference frame with the help of satellites orbiting the earth. However, GPS lacks consistency and accuracy compared to other sensors [3]. A more continuous way of determining position could be done by using an odometer or wheel encoder, which measures the rotation speed of a wheel. From this, angular and linear velocities can be calculated and, combined with time measurement, the distance travelled by integrating over time. However, systems using odometers suffer from accumulation of error over time, becoming less and less accurate if not updated with a correct position [4]. Therefore, the latter two methods are often combined, such that when the odometer error starts to accumulate, it is updated with GPS data, as proposed in [5]. Fusing different type of sensor data has been widely applied in position estimation. A common method for fusing sensor data is the Kalman filter: a set of equations that provides an efficient computational (recursive) solution of the least-squares method [6]. Also, inertial data from an Inertial Measurement Unit (IMU) in combination with GPS data can be applied to decrease the error. [7] shows how this method can directly be applied to avoid collisions for autonomous vehicles by using several Kalman filters to predict a car's position during lane switching. In [8] the Kalman filter is applied in similarly, but now the effects of GPS signal loss are tested. Their filter was still able to predict the position well with an accuracy of centimeters. However, the error became bigger when the GPS signal was lost for a longer time.
Lastly, a completely different method for sensor fusion is with the use of neural networks. Neural networks are often applied in areas where a system is not fully known, such that a network can learn a system's characteristics based on obtained data. An example is presented in [9], where different sensors are combined to estimate the wear of a certain tool. However, neural networks heavily depend on their training set, such as sensor data. A lack of data can lead to poor predictions. Also, training a network in general is computationally hard.

1.2 Problem definition

A problem that could arise when using GPS sensors lies in situations where there is no access to the GPS data for a longer period of time. Such a situation could be in a parking garage, where the GPS signal is blocked. Keeping track of a car's position is still very important to prevent collisions with other cars or persons. So a system is required that can determine its position using only local measurements of a car. So the main research objective can therefore be formulated as follows: ***developing a filter for path reconstruction of parking maneuvers, using IMU and wheel sensors***. Sensor data from an IMU and wheel sensors are fused using a Kalman filter, and its performance is evaluated on different trajectories.

1.3 Thesis outline

Firstly, in Chapter 2, the kinematic model of a four-wheel vehicle is described, which is required for the filter to work. Accordingly, a general overview of the filter used is given in chapter 3. Then in chapter 4, an explanation is given on how the filter was implemented with the vehicle model. In chapter 5 the results are presented and evaluated. Finally, the project concludes with a conclusion followed by a recommendation for future research.

2 Four-wheel vehicle model

To reconstruct the pose of a car, a model that describes the motion as a function of inputs is required. In this chapter, a kinematic model is introduced that describes the car's behaviour over time. The model is a kinematic model because the forces put on the car, such as tire or aerodynamic forces, are small at the near-zero velocities that are present during parking manoeuvres. Also, [10] shows that for low speeds, the difference between the two types of models is very small, but the required computational power is much lower for a kinematic model.

2.1 Assumptions

A car is a four-wheel vehicle, with a wheelbase L and track width t_w, schematically shown in Figure 2.1. All used car parameters are accessible in Table A.1 and Table A.1. The bicycle model presented in [11] is used as a basis. This model has been extended with two wheels on the front and rear, where both front wheels steer with a different steering angle due to Ackermann steering. However, for simplicity, the inner steering angle is assumed to be the total steering angle of the vehicle.

The vehicle moves with velocity $v(t)$ and steers with steering angle δ_f, and together, these quantities form the input u of the kinematic model. The state of the car consists of a longitudinal position $x(t)$, lateral position $y(t)$, and orientation $\psi(t)$ at the center of gravity of the car. Combined gives state vector X as:

$$X = \begin{bmatrix} x \\ y \\ \psi \end{bmatrix}. \tag{2.1}$$

Four wheel velocities ω_i for $i = 1, 2, 3, 4$ (where 1 denotes the front left tire, 2 front right, 3 rear left and 4 rear right) and the yaw rate $\dot{\psi}$ together make the output of the model. The positional output x, y and ψ are expressed in absolute coordinates with respect to a fixed coordinate frame O in the real world. In the case of Figure 2.1, that is the left bottom corner.

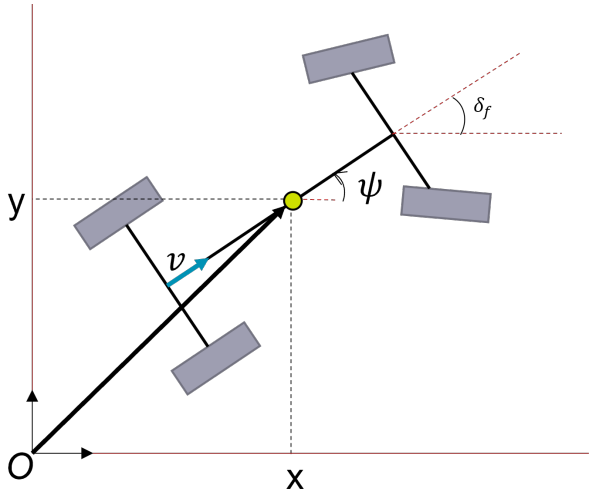

Figure 2.1: *Variable definitions of the four-wheel vehicle model.*

2.2 Kinematic constraints

The kinematic model is constrained as follows:

- All four wheels have a no-slip conditions, meaning their local lateral velocity is zero
- The heading directions of the front wheels are constrained by input δ_f

- The longitudinal velocity at the center of the rear axle is fixed by input v
- Relative motion between the four wheels is constrained by the wheelbase and track width of the car

The maximum velocity is based on typical parking speeds and taken as 1 m/s. The car manufacturer determines the range of possible steering angles, but typically a car can make a steering angle of 30°, or approximately 0.5 rad [12].

2.3 Input-output relations

The time derivatives of the pose relate to the input variables as follows:

$$\dot{X} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} v^* \cos(\psi + \beta) \\ v^* \sin(\psi + \beta) \\ v \frac{2\tan(\delta_f)}{2L + t_w \tan(\delta_f)} \end{bmatrix}. \tag{2.2}$$

The velocity v^* represents the velocity magnitude at the car's center of gravity, and is given by:

$$v^* = \sqrt{v^2 + \dot{\psi}^2 l_r^2}, \tag{2.3}$$

where regular v is the velocity at the rear axle of the car. The velocities $\dot{x}$ and $\dot{y}$ are derived from splitting the velocity vector with magnitude v^* up in the two planar directions of frame O. In 2.4 and 2.5, the Ackermann steering formula for the inner and outer front wheels are given [13].

$$\delta_{fin} = \tan^{-1}\left(\frac{L}{R - \frac{t_w}{2}}\right). \tag{2.4}$$

$$\delta_{fout} = \tan^{-1}\left(\frac{L}{R + \frac{t_w}{2}}\right). \tag{2.5}$$

To obtain $\dot{\psi}$, the inner wheel steering angle 2.4 is used, where $\frac{v}{\dot{\psi}}$ is substituted for the turning radius R:

$$\begin{aligned} \delta_{fin} &= \tan^{-1}\left(\frac{L}{\frac{v}{\dot{\psi}} - \frac{t_w}{2}}\right) \\ \tan(\delta_{fin}) &= \frac{L}{\frac{v}{\dot{\psi}} - \frac{t_w}{2}} \\ \dot{\psi} &= v\frac{2\tan(\delta_f)}{2L + t_w \tan(\delta_f)}. \end{aligned} \tag{2.6}$$

β is the side slip angle of the car, defined as follows from [11]:

$$\beta = \tan^{-1}\left(\frac{l_r}{L}\tan(\delta_f)\right), \tag{2.7}$$

such that β is also a function of the input δ_f. The side slip angle is the angle between the velocity vector of the vehicle and the heading direction. The variables l_r and L are car parameters defined as the distance from the center of gravity to the rear axle and the wheelbase, respectively.

2.4 Measurement function

For the Kalman filter, described in section 2, to work, a measurement function, predicting the measured quantities for a given input, is required. Therefore, rotational wheel velocities, as well as the yaw rate, will be described as a function of the inputs.

The four rotational wheel velocities are derived with vectors in coordinate frame O, which coincides with the center of rotation of the car, as shown in Figure 2.2

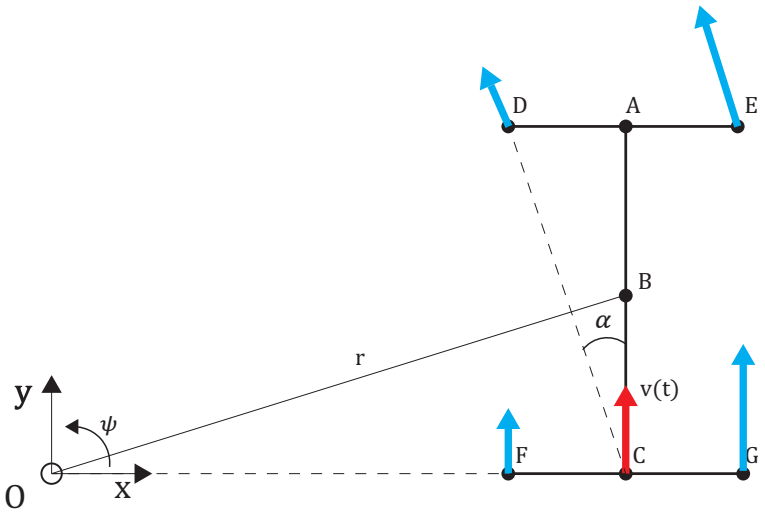

Figure 2.2: *The car with its input velocity vector $v(t)$ (red) and the resulting velocity vectors on the wheels (blue) in reference frame O.*

Now, points D, E, F and G are expressed with position vectors with respect to point C:

$$\begin{aligned}\vec{r}_D &= \vec{r}_C + \vec{r}_{D/C} \\ &= R\begin{bmatrix}\cos(\psi)\\ \sin(\psi)\end{bmatrix} + \sqrt{L^2 + \left(\frac{t_w}{2}\right)^2}\begin{bmatrix}\sin(\alpha+\psi)\\ \cos(\alpha+\psi)\end{bmatrix}.\end{aligned} \tag{2.8}$$

The velocity vector can then be derived as follows:

$$\begin{aligned}\dot{\vec{r}}_D &= \dot{\vec{r}}_C + \dot{\vec{r}}_{D/C} \\ &= (R\begin{bmatrix}-\sin(\psi)\\ \cos(\psi)\end{bmatrix} + \sqrt{L^2 + \left(\frac{t_w}{2}\right)^2}\begin{bmatrix}\cos(\alpha+\psi)\\ -\sin(\alpha+\psi)\end{bmatrix})\dot{\psi}.\end{aligned} \tag{2.9}$$

R can be eliminated with the relation $R = v/\dot{\psi}$ and combined with the tire radius r_t. Similarly this is done for all four wheels. Combined with the already found function of $\dot{\psi}$ in 2.2, measurement function h is given by:

$$h = \begin{bmatrix}\dot{\psi}\\ \omega_1\\ \omega_2\\ \omega_3\\ \omega_4\end{bmatrix} = \begin{bmatrix} v\frac{2\tan(\delta_f)}{2L + t_w\tan(\delta_f)} \\ \frac{1}{r_t}\sqrt{v^2 + \dot{\psi}^2 l_{CD}^2 - 2v\dot{\psi}l_{CD}\sin(\alpha)} \\ \frac{1}{r_t}\sqrt{v^2 + \dot{\psi}^2 l_{CD}^2 + 2v\dot{\psi}l_{CD}\sin(\alpha)} \\ \frac{1}{r_t}(v - \frac{1}{2}\dot{\psi}t_w) \\ \frac{1}{r_t}(v + \frac{1}{2}\dot{\psi}t_w)\end{bmatrix}. \tag{2.10}$$

Parameters l_{CD} and α can be derived from Figure 2.2. What should be noted here is that $\dot{\psi}$ has not been written out in the expressions for ω_i to keep it structured.

2.5 Results and verification

To verify whether the model works accordingly, simple functions for v and δ_f are entered for which the expected outputs are known beforehand. With this method, early errors can be eliminated from the model.

The first case only enters a constant velocity of 1 m/s into the model. Due to the absence of a steering angle, the vehicle should travel in a straight line, combined with a simulation time of 10 seconds, a total of 10 meters is travelled. In Figure 2.3 the trajectory is calculated, showing a straight path from 0 to 10 meters.

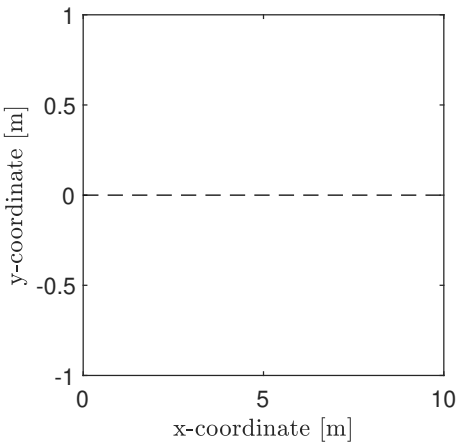

Figure 2.3: *Trajectory of the vehicle for a constant velocity v of 1 m/s after 10 seconds, with initial conditions $z = \begin{bmatrix}0 & 0 & 0\end{bmatrix}^T$.*

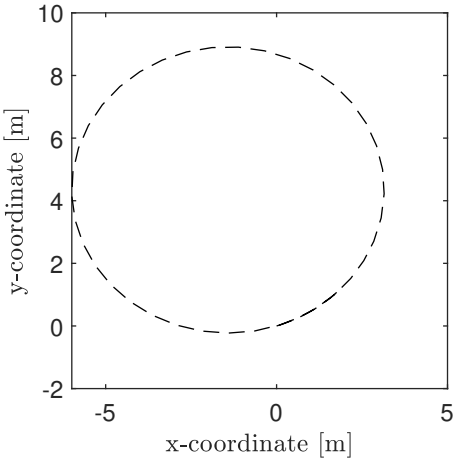

Figure 2.4: *Trajectory of the vehicle for a constant velocity v of 1 m/s and a constant steering angle δ_f of 0.5 rad after 30 seconds, with initial conditions $z = \begin{bmatrix}0 & 0 & 0\end{bmatrix}^T$.*

For the second case, a constant velocity is combined with a constant, non-zero steering angle. This should result in a circular path, with the radius of the path given by $R = v/\dot{\psi}$, where $\dot{\psi}$ is calculated based on the steering angle and the velocity.

Comparing the turning radius from the trajectory in Figure 2.4 to the calculated turning radius gives 4.57 m and 4.41 m, respectively. This is due to the fact that the position of the car is observed from the CoG and not the center of the rear axle. Therefore a bigger velocity is observed, v^*, resulting in a bigger turning radius.

The rotational wheel velocities are checked under similar conditions as in case two. One would expect that driving in a circle will yield a constant rotational wheel velocity for all four wheels. The model's output supports this, as shown in Figure 2.5. Also, driving in an anti-clockwise direction would mean for the right-sided wheels to spin faster, which is also correct in the model.

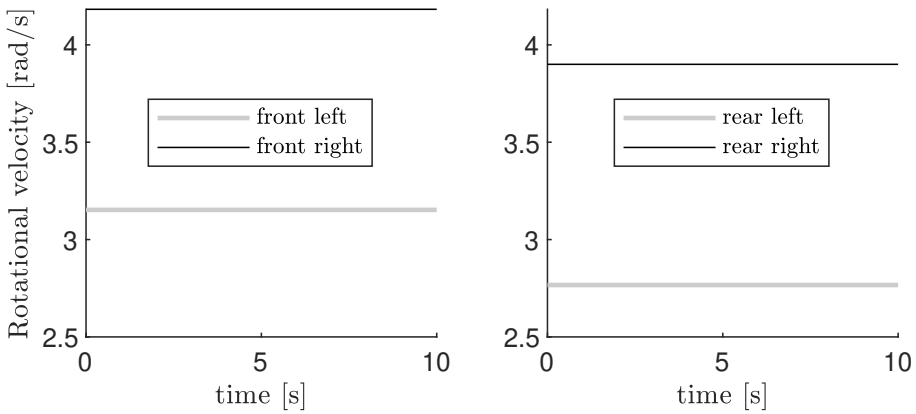

Figure 2.5: *Rotational wheel velocities of the front (right) and rear (left) tires of the vehicle for a constant velocity v of 1 m/s and a constant steering angle δ_f of 0.5 rad after 10 seconds, with initial conditions $z = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$.*

2.6 Performance metrics

The error in x and y are combined in r, also shown in Figure 2.2, which is defined as the distance from reference frame O, given by:

$$r = \sqrt{x^2 + y^2}. \tag{2.11}$$

The error in orientation angle ψ is observed as well. An error metric is needed to assess the quality of the estimations done in the next chapters. A choice was made to use the Root Mean Squared Error (RMSE), as it, in general, works well with zero values and has the same unit as the assessed values. RMSE is calculated as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(r - \hat{r})^2}, \tag{2.12}$$

with n the number of measurements, r the true absolute distance and $\hat{r}$ the estimated absolute distance to the reference frame. Similarly, this is done for ψ.

3 Filter

To combine both predictions from the model described in the previous chapter and the measurements obtained from a car to get to accurate state estimation, they need to be processed. A Kalman filter has been used for processing the two, which will be explained further in this chapter.

3.1 The Kalman filter

The Kalman filter is often applied in cases where you want to estimate the states of your system. In the mathematical sense, the Kalman filter is a set of equations that provides an efficient computational (recursive) solution of the least-squares method [6]. The latter is a more broad explanation. When it comes down to state estimation, the Kalman filter constantly predicts a state by inputting previously estimated states into a model of the linear equations, such as 3.1 and 3.2.

$$X_{k+1} = A_k X_k + B u_k + w_k \tag{3.1}$$

$$z_k = H_k X_k + v_k \tag{3.2}$$

Where subscript k denotes the time step, X_k, u_k and z_k the state, input and measurement vector (respectively) at time step k, and w_k and v_k the process and measurement noise (respectively), assumed to be white with a normal probability distribution:

$$p(w) \sim N(0, Q), \tag{3.3}$$

$$p(v) \sim N(0, R). \tag{3.4}$$

An overview of the operation of the Kalman filter is given in Figure 3.1. It will be used to explain the filter in more detail.
The filter is initiated with an initial condition $\hat{X}_0$ and error covariance P_0. Then the state for the next time step X_{k+1}^- is predicted with function f in (1), representing the model of the system, together with the error covariance P_{k+1}^- in (2). Together this makes the predicting step. Important here is the process noise matrix Q_k, which represents the uncertainty about the used model, i.e. larger values in Q_k lead to a significant error covariance and more uncertainty on the prediction of the state.
Subsequently follows the correcting step with computing the Kalman gain K_k. Here measurement noise matrix R_k plays an important role, as noisy measurements result in large values in R_k and therefore give a small gain, meaning that the state prediction is corrected a lot in (4), meaning the model is trusted more over the measurements. In (4) also, the measurement z_k and the prediction thereof h are compared: a big difference leads to a big correcting factor on the prediction X_k^- and vice versa. Finally, the error covariance P_k is updated and fed into the prediction step. This recursive process of predicting and correcting is essentially what the Kalman filter does.

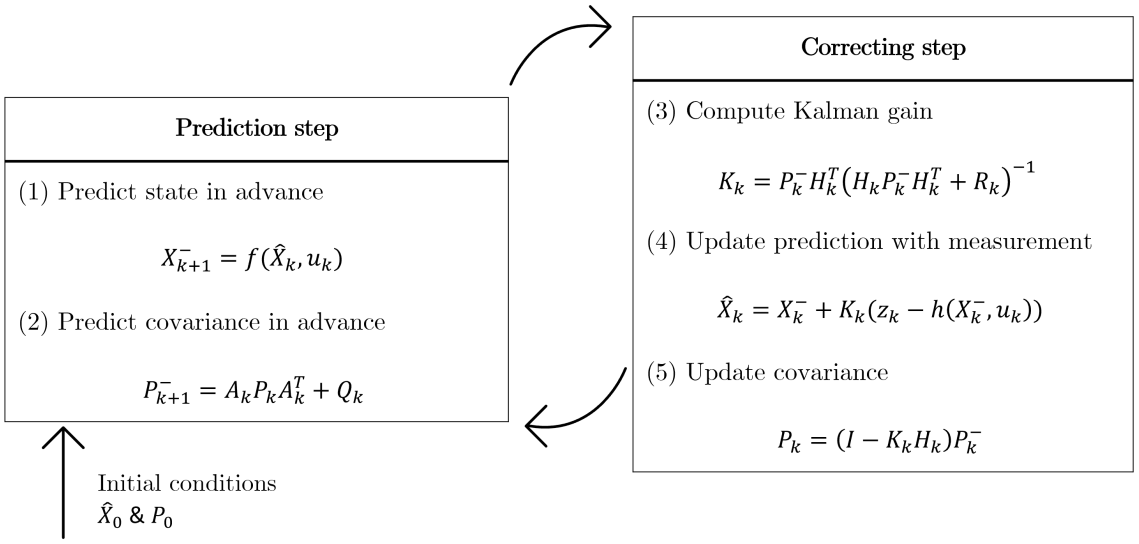

Figure 3.1: *A complete overview of the operation of the Kalman filter.*

3.1.1 Extended Kalman Filter (EKF)

Now, the 'regular' Kalman filter only works for systems described by linear equations. However, in reality, problems are usually of the non-linear type. Therefore, the Kalman filter has been extended for non-linear systems. The non-linear version that will be discussed is the Extended Kalman filter (EKF). The non-linear form of 3.1 and 3.2 then becomes:

$$X_{k+1}^- = f(\hat{X}_k, u_k) + w_k, \tag{3.5}$$

$$z_k = h(\hat{X}_k, u_k) + v_k. \tag{3.6}$$

The extended Kalman filter linearises the system around a current estimate, by taking the Jacobian of the non-linear state function f and h for each time step. Subsequently, this Jacobian is then evaluated with the current estimated state, which results in similar matrices as in 3.1 and 3.2:

$$A_k = \frac{\partial f(\hat{X}_k, u_k)}{\partial X}, \tag{3.7}$$

$$H_k = \frac{\partial h(\hat{X}_k, u_k)}{\partial X}. \tag{3.8}$$

Once these matrices are obtained, the exact same steps for prediction and correcting can be used as for a regular Kalman filter.

4 Vehicle implementations

To effectively apply the Kalman filter from section 3, certain assumptions and conditions were used, which are explained in this chapter.

4.1 Extra state

Now, yaw rate $\dot{\psi}$ has been added to the observed states to make sure that the H_k matrix is not all zero. The state vector in 2.1 is extended to

$$X = \begin{bmatrix} x \\ y \\ \psi \\ \dot{\psi} \end{bmatrix}. \tag{4.1}$$

By adding this state, there will be non-zero derivatives for the last column, as the wheel speed velocities depend on $\dot{\psi}$. With H_k as a non-zero matrix, the Kalman gain and estimated states are computed accordingly.

4.2 Initial inputs

The filter requires both initial conditions for the states $\hat{x}_k$ and the state estimate covariance P_k. Now, the assumption is made that the initial pose of the predicted states, x, y and ψ, are known. This can be justified by considering that the vehicle will calculate a path based on its current position, which can be set to any value, which for convenience is set to $\begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^T$. The covariance is a square matrix of size 4, with on the diagonal the covariance of the states themselves. These numbers come down to the certainty with which the initial conditions are known. Since it was just assumed that these conditions are known, these covariances can be set to 0. All other components, i.e. the covariance between two different states, are assumed to be zero, as the cross-covariance of two known states is also zero.

4.3 Jacobian matrices A_k & H_k

The matrices in 4.2 and 4.3 follow from 3.7 and 3.8. They are obtained by taking the derivative with respect to the state vector X.

$$A_k = \begin{bmatrix} 1 & 0 & -v^* \sin(\psi + \beta)dt & l_r^2 \frac{\dot{\psi}}{\sqrt{v^2 + \dot{\psi}^2 l_r^2}} \cos(\psi + \beta)dt \\ 0 & 1 & v^* \cos(\psi + \beta)dt & l_r^2 \frac{\dot{\psi}}{\sqrt{v^2 + \dot{\psi}^2 l_r^2}} \sin(\psi + \beta)dt \\ 0 & 0 & 1 & dt \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

$$H_k = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{-t_w}{2r_t} \\ 0 & 0 & 0 & 0 & \frac{-t_w}{2r_t} \\ 0 & 0 & 0 & 0 & \frac{l_{CD}(v \sin(\alpha) + l_{CD}\dot{\psi}}{r_t \sqrt{v^2 + 2v\dot{\psi} \sin(\alpha) + l_{CD}^2 \dot{\psi}^2}} \\ 0 & 0 & 0 & 0 & \frac{l_{CD}(-v \sin(\alpha) + l_{CD}\dot{\psi}}{r_t \sqrt{v^2 - 2v\dot{\psi} \sin(\alpha) + l_{CD}^2 \dot{\psi}^2}} \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

4.4 Process & Measurement noise

The matrices Q_k and R_k give information about the process and measurement noise respectively. Now, the measurement noise is given by the accuracy of the sensor being used, which is based on [14] for the wheel speed sensors and [15] for the yaw rate sensor. For this case, that would be four identical wheel speed sensors and a gyroscope (the relevant component of the IMU measuring the yaw rate of a car) for the yaw rate, which looks as follows:

$$R = \begin{bmatrix} 10^{-1} & 0 & 0 & 0 & 0 \\ 0 & 10^{-1} & 0 & 0 & 0 \\ 0 & 0 & 10^{-1} & 0 & 0 \\ 0 & 0 & 0 & 10^{-1} & 0 \\ 0 & 0 & 0 & 0 & 10^{-2} \end{bmatrix}. \tag{4.4}$$

The diagonal components are the noise of the sensors themselves. all other components off the diagonal are zero, as the sensors and thus their noise is not dependent of each other.

The entries for the process noise are tuned to find the values that result in the lowest RMSE scores. The entry on the x and y state are taken equal, as the tuning is performed on a circular trajectory, meaning both x and y contribute evenly, and therefore, the process noise should be the same. The entry on $\dot{\psi}$ is taken 100 times smaller than ψ, as the sample time is taken as $dt = 0.01\ s$ and integrating $\dot{\psi}$ over time gives ψ. This gives two parameters, p_1 and p_2, respectively, to tune. The matrix form is shown in 4.5.

$$Q = \begin{bmatrix} p_1 & 0 & 0 & 0 \\ 0 & p_1 & 0 & 0 \\ 0 & 0 & p_2 & 0 \\ 0 & 0 & 0 & p_2 dt \end{bmatrix} \tag{4.5}$$

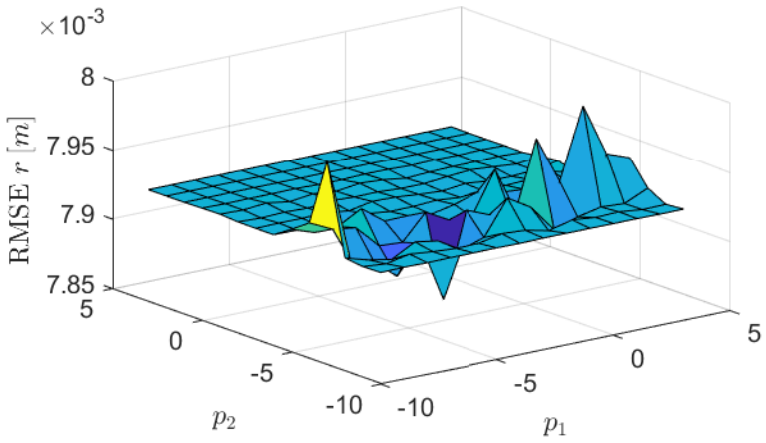

Figure 4.1: *The RMSE score on r for varying parameter values p_1 and p_2. A logarithmic scale is used, meaning $p_1 = 1 = 10^1$. Note that, even though there is a clear minimum, the scale of the RMSE score is very small (10^{-3} [m]).*

In Figure 4.1 and Figure 4.2 the variation of the RMSE scores for different p_1 and p_2 values. Both plots have a different optimal set of parameters, but the differences in RMSE scores are small. The p_1 and p_2 values of the minimum of the RMSE on r are chosen, meaning that $p_1 = -5$ and $p_2 = -7$.

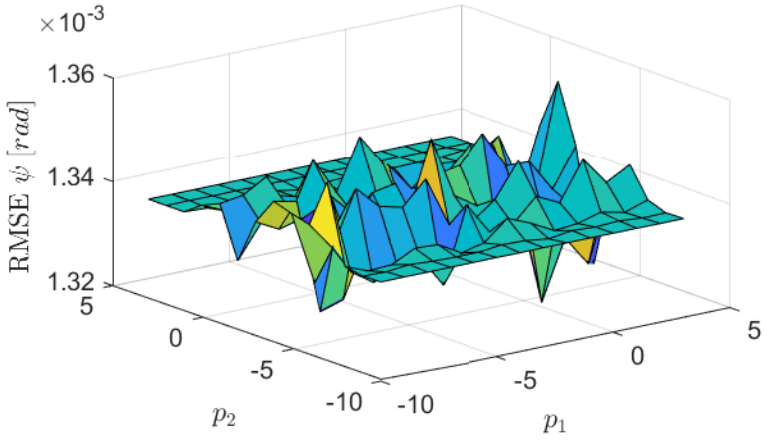

Figure 4.2: *The RMSE score on ψ for varying parameter values p_1 and p_2. A logarithmic scale is used, meaning $p_1 = 1 = 10^1$. Note here too, that even though there is a clear minimum, the scale of the RMSE score is very small (10^{-3} [rad]).*

That results in the following matrix:

$$Q = \begin{bmatrix} 10^{-5} & 0 & 0 & 0 \\ 0 & 10^{-5} & 0 & 0 \\ 0 & 0 & 10^{-7} & 0 \\ 0 & 0 & 0 & 10^{-9} \end{bmatrix}. \tag{4.6}$$

4.5 Model

The model that is initially used to predict the states in the filter is identical to the model that produces the true states, meaning the modelled vehicle is equal to the real vehicle in this project. However, a simplified version of this model is tested as well to find out whether the states can still be estimated with a low error. This simplification is done by assuming that δ_f is small and the turning radius R is big therefore the following holds:

$$\delta_{fin} = \arctan\left(\frac{L}{R - \frac{t_w}{2}}\right) \approx \frac{L}{R}. \tag{4.7}$$

And similarly for β:

$$\beta = \arctan\left(\frac{l_r}{L}\tan(\delta_f)\right) \approx \frac{l_r}{L}\tan(\delta_f). \tag{4.8}$$

Tuning the filter on the simplified model, as shown in Figure 4.3 and Figure 4.4, resulted in $p_1 = -10$ and $p_2 = -7$, shaping the Q matrix into:

$$Q = \begin{bmatrix} 10^{-10} & 0 & 0 & 0 \\ 0 & 10^{-10} & 0 & 0 \\ 0 & 0 & 10^{-7} & 0 \\ 0 & 0 & 0 & 10^{-9} \end{bmatrix}. \tag{4.9}$$

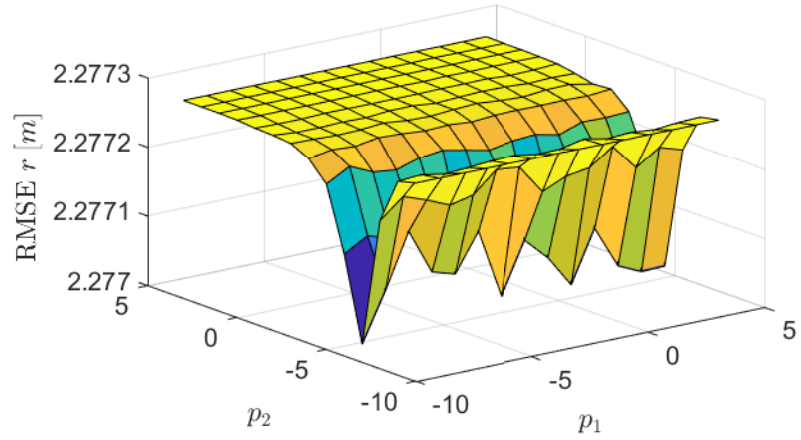

Figure 4.3: *The RMSE score on r for varying parameter values p_1 and p_2 for the filter using a simplified model. A logarithmic scale is used, meaning $p_1 = 1 = 10^1$.*

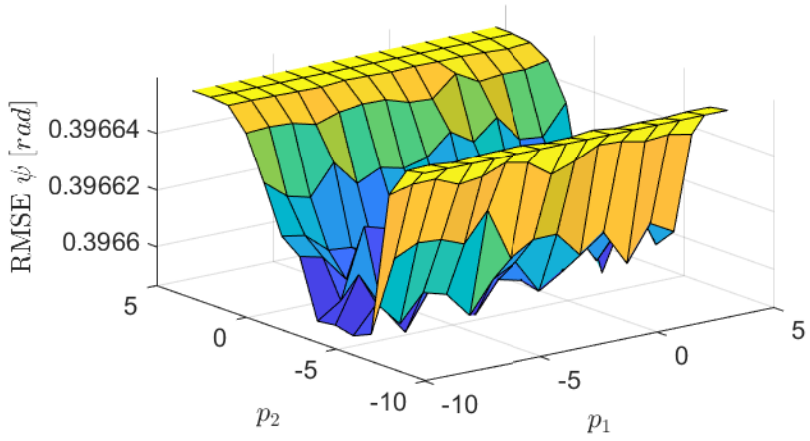

Figure 4.4: *The RMSE score on ψ for varying parameter values p_1 and p_2 for the filter using a simplified model. A logarithmic scale is used, meaning $p_1 = 1 = 10^1$.*

5 Filter Results

In this chapter the outcomes of different tests on the filter are presented. First, circular and parallel parking-like trajectory will be tested where the model the Kalman filter uses to predict the states is the same as the model used to generate the true signal. This will be referred to as using the perfect model. Second, the same trajectories are tested but on a Kalman filter that only has access to a simplified model as described in subsection 4.5.

5.1 Perfect prediction model

5.1.1 Circular trajectory

A constant velocity is combined with a constant steering angle, which results in a circular path like in section 2. The results from this input can be seen in Figure 5.1. The RMSE scores can be found in Table 5.1

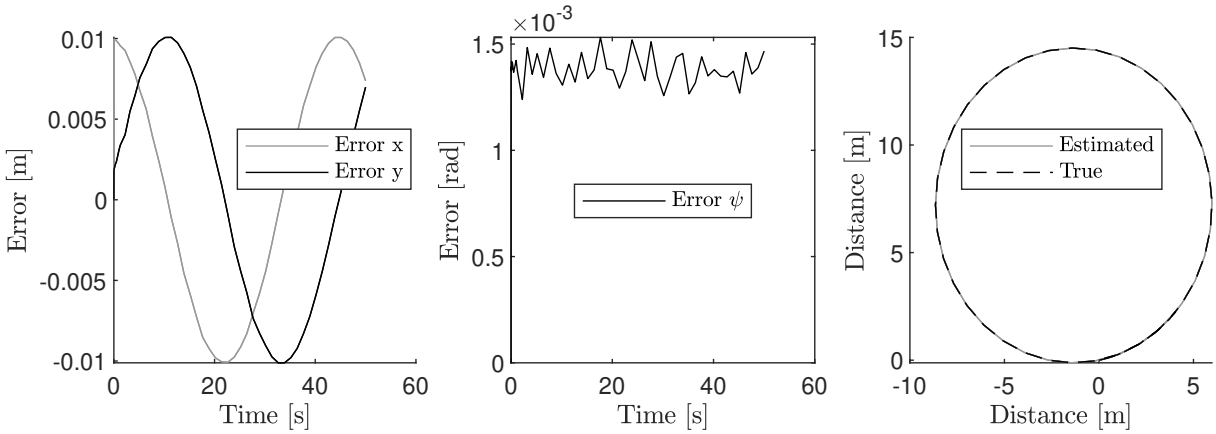

Figure 5.1: *The error plotted over time for the x, y and ψ coordinate in the left and middle plot respectively. On the right are plotted both the true and estimated trajectory for a constant velocity of 1 m/s and constant steering angle of 0.3 rad. The initial conditions $z = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^T$ and the perfect model were used.*

The true and estimated trajectory in Figure 5.1 lie almost perfectly on top of each other. Only in the error plots it becomes clear there is still a small error. The error on the x and y coordinates appears to have a sine-like character, while the error on ψ seems to converge to a certain value. In both cases, the RMSE scores are already small in the context of parking margins.

Now the estimation of the yaw rate is considered to review the filter's performance on making an estimation based on a noisy signal. This is shown in Figure 5.2. The estimated signal is still noisy, but its amplitude has been decreased drastically, from 0.01 to 0.0001 rad approximately.

Table 5.1: *The root mean square error on the distance from reference frame r and orientation ψ for different paths using the perfect model.*

path	RMSE R $[m]$	RMSE ψ $[rad]$
Circle	0.0079	1.3e-3
Parallel	0.0096	3.4e-4

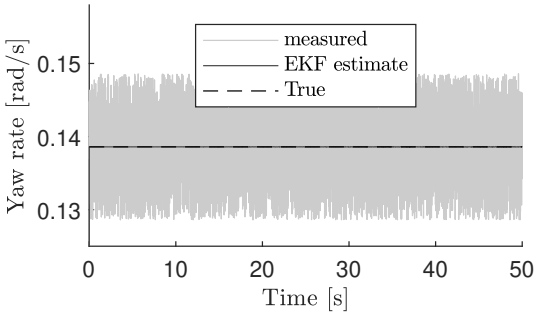

Figure 5.2: *The true, measured and estimated yaw rate $\dot{\psi}$ plotted over time for a constant velocity and constant steering angle.*

5.1.2 Parking maneuver

The next path is constructed by inputting a sine for the steering angle with an amplitude of 0.1 *rad*. By taking only half of the period of a sine wave, a parallel parking maneuver can be mimicked. The results of this input are shown in Figure 5.3.

Now, the error on the x-coordinate seems to be constant over time, whereas the y-coordinate is initially smaller but varies. Since only part of a sine has been inputted, one can expect the error on x and y to be periodic over a longer time window. What is different compared to the case of a constant steering angle is the periodic error on ψ. From the RMSE scores, it also appears to be very similar to the first case. Only the RMSE on ψ is an order smaller.

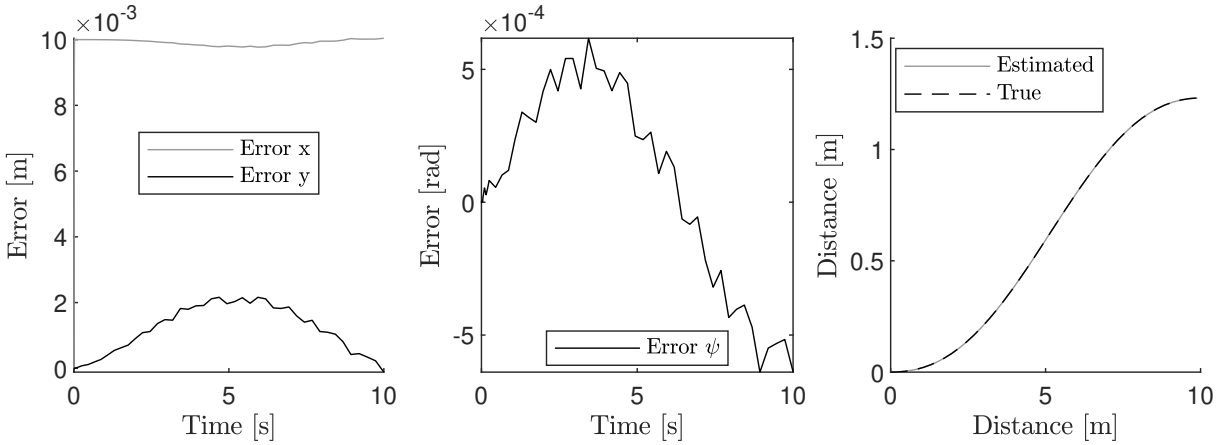

Figure 5.3: *The error plotted over time for the x, y and ψ coordinate in the left and middle plot respectively. On the right are plotted both the true and estimated trajectory for a constant velocity of 1 m/s and sinusoidal steering angle with an amplitude of 0.1 rad. The initial conditions $z = \begin{bmatrix}0 & 0 & 0 & 0\end{bmatrix}^T$ and the perfect model were used.*

Reviewing the quality of estimation of the yaw rate, the filter performs similar to the case of the circular trajectory in Figure 5.4: the noise on the measurement is heavily reduced.

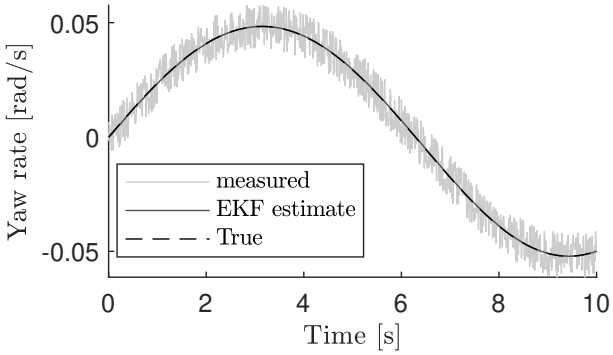

Figure 5.4: *The true, measured and estimated yaw rate $\dot{\psi}$ plotted over time for a constant velocity of 1 m/s and sinusoidal steering angle with an amplitude of 0.1 rad.*

5.2 Simplified prediction model

5.2.1 Circular trajectory

Using the simplified model, the error becomes almost three orders bigger than was the case when the perfect model was used on both the R and ψ, as can be seen both in the error plots in Figure 5.5 and the RMSE scores in Table 5.2.

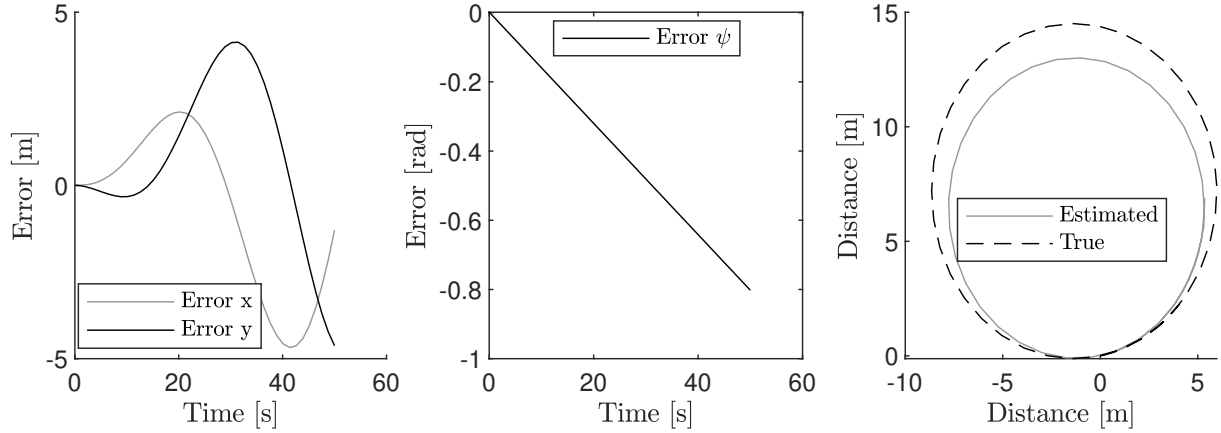

Figure 5.5: *The error plotted over time for the x, y and ψ coordinate in the left and middle plot respectively. On the right are plotted both the true and estimated trajectory for a constant velocity of 1 m/s and constant steering angle of 0.3 rad. The initial conditions $z = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^T$ and a simplified model were used.*

Table 5.2: *The root mean square error on the distance from the reference frame r and orientation ψ for different paths using a simplified model.*

path	RMSE R $[m]$	RMSE ψ $[rad]$
Circle	2.3	0.40
Parallel	0.0099	0.0046

The yaw rate estimation has an offset to the true signal in Figure 5.6, but again the noise on the signal is reduced.

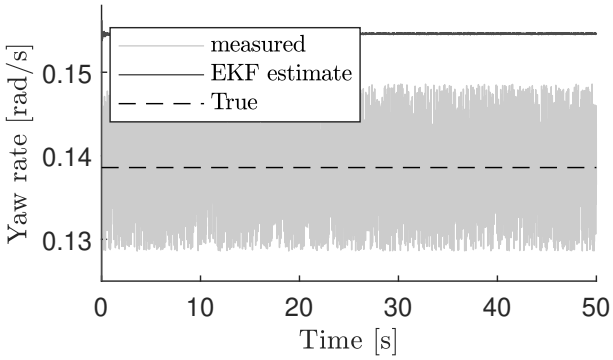

Figure 5.6: *The true, measured and estimated yaw rate $\dot{\psi}$ plotted over time for a constant velocity of 1 m/s and constant steering angle of 0.3 rad using a simplified model.*

5.2.2 Parking maneuver

For the parking maneuver in Figure 5.7, the error on x and y is of small scale initially, but grows over time. The error on ψ keeps increasing over time but on a small rate within the 10 seconds running time, such that the RMSE scores in Table 5.2 are of the same order as when the perfect model was used. Since this parking maneuver is only run for 10 seconds, the error does not continue to grow to bigger values, as is the case for the circular trajectory.

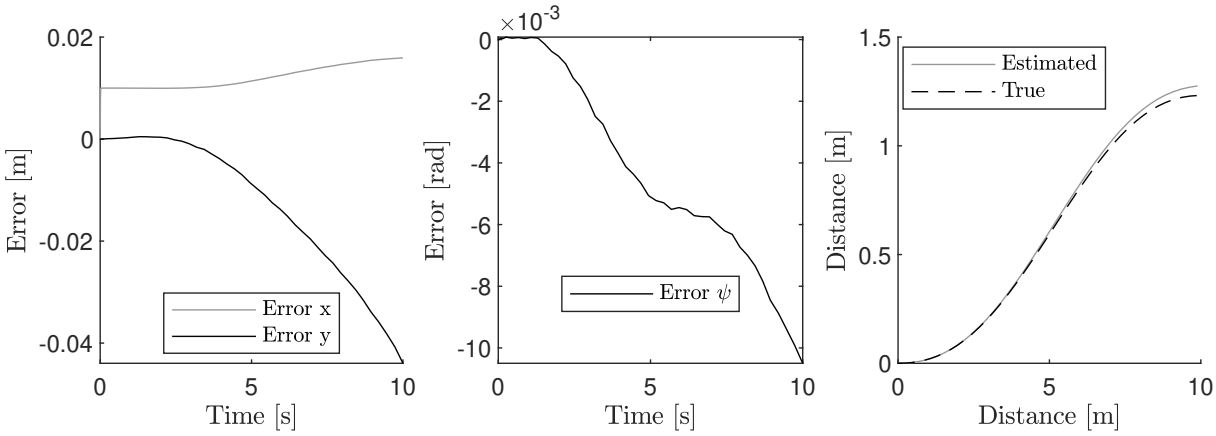

Figure 5.7: *The error plotted over time for the x, y and ψ coordinate in the left and middle plot respectively. On the right are plotted both the true trajectory as the estimated trajectory for a constant velocity of 1 m/s and sinusoidal steering angle with an amplitude of 0.1 rad.. The initial conditions $z = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^T$ and a simplified model were used. Note that x and y distances on the right plot are scaled differently.*

The yaw rate estimation in Figure 5.8 is close to the true signal, only here too, there is a slightly positive offset.

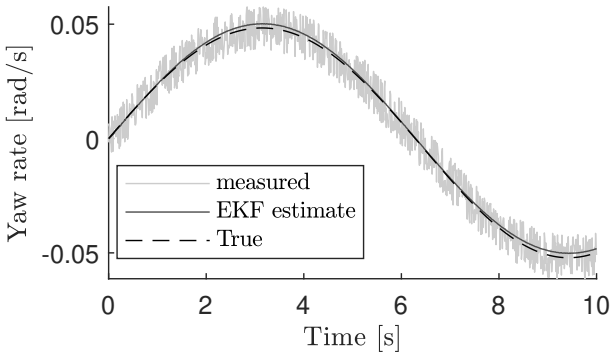

Figure 5.8: *The true, measured and estimated yaw rate $\dot{\psi}$ plotted over time for a constant velocity of 1 m/s and sinusoidal steering angle with an amplitude of 0.1 rad using a simplified model.*

5.3 Discussion

The EKF using a perfect model to predict the states can reconstruct the given trajectory with a small error. This is due to the good estimation of the yaw rate: if the yaw rate is predicted well, then so is ψ since there is only an integration step between the two. And from a well-predicted ψ follows an accurate x and y position since those are dependent on ψ.

However, the EKF cannot estimate the yaw rate well if it has to use a simplified model. This automatically results in poor x, y and ψ estimations, as those dependent on the yaw rate. For a circular trajectory, this becomes best visible: the EKF estimates a bigger yaw rate, which with $R = \frac{v}{\dot{\psi}}$ can be calculated back to a smaller turning radius and thus a smaller circle. That is indeed the case in Figure 5.5. This problem is less apparent for the parallel parking trajectory, but that can be explained by the smaller time window of 10 instead of 50 seconds. The longer the yaw rate estimation is off, the bigger the error on ψ; therefore, the error on x and y grows as well, which is already visible in Figure 5.7. But if the time scale is small, the EKF can still track the trajectory accurately.

6 Conclusion and recommendation

6.1 Conclusion

In this project, an extended Kalman filter (EKF) was implemented to a four-wheel vehicle to estimate its pose with respect to a certain reference frame, using only IMU and wheel speed sensors. The EKF was able to estimate the x and y coordinates with an RMSE score of order 10^{-3} m on r and 10^{-3} rad on ψ for a circular trajectory and with an order of 10^{-3} m and 10^{-3} rad on the parallel parking trajectory. These results were produced with the EKF estimating using the same vehicle model used to produce the true signal. For the simplified vehicle model, the EKF produces an RMSE of order of 10^{1} m on r and 10^{-1} rad on ψ on the circular trajectory. Similar scores are obtained on the parallel parking trajectory as for the case of using the same vehicle model, as this trajectory was performed in a smaller time window, where the error could not yet grow big. The error on the estimation thus increases when the vehicle model is not known well, although on the short times scales ($t < 10$ s), the error is kept small.

6.2 Recommendation

First, the vehicle model used in this project could be improved, especially on the steering angle input. Both inner and outer wheel steering angles should be included to get an accurate description of the total steering angle of the car, as currently, only the inner steering angle is used. A suggestion might be to include both the inner and outer front wheel steering angles as input to the model instead of just using one of the two. Improving the model to behave more like a real vehicle has shown better estimation capabilities of the EKF.

Another method of tuning the EKF parameters, especially the process noise matrix Q, could result in better estimation qualities. Currently, only the diagonal entries have been considered to be non-zero, and therefore, they were tuned. However, a better understanding in this matrix's cross-terms might show there are more non-zero components that can be tuned and used to improve the EKF.

Lastly, actualising the filter on a physical setup, such as an autonomous vehicle, could give interesting insights into the filter's capability to estimate the position in real-time. This also gives a chance to verify the measurement noise of the sensors. Problems such as bigger estimations errors due to delay in computations times or sensor drift could arise.

References

[1] Ford Motor Company, "What is Active Park Assist 2.0?," 2022.

[2] "Hyundai Remote Smart Parking Assist (RSPA) — Vern Eide Hyundai Sioux City," 2022.

[3] N. M. Drawil, H. M. Amar, and O. A. Basir, "GPS Localization Accuracy Classification: A Context-Based Approach," *IEEE Transactions on intelligent transportation systems*, vol. 14, no. 1, 2013.

[4] W. Rahiman and Z. Zainal, "An overview of development GPS navigation for autonomous car Autonomous Ground Vehicle View project FPGA Application View project An Overview of Development GPS Navigation for Autonomous Car," 2013.

[5] C. Vlcek, P. McLain, and M. Murphy, "GPS/dead reckoning for vehicle tracking in the 'urban canyon' environment," *Proceedings of the IEEE-IEE Vehicle Navigation and Informations Systems Conference*, 1993.

[6] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," *Department of Computer ScienceUniversity of North Carolina at Chapel Hill*, 1995.

[7] R. Toledo-Moreo and M. A. Zamora-Izquierdo, "Collision avoidance support in roads with lateral and longitudinal maneuver prediction by fusing GPS/IMU and digital maps," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 4, pp. 611–625, 2010.

[8] C. Huang, Y. Liu, Y. Jia, and H. Chen, "Position estimation for an unmanned ground car (UGC) by multi-sensor fusion under random loss of GPS signals; Position estimation for an unmanned ground car (UGC) by multi-sensor fusion under random loss of GPS signals," *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, 2015.

[9] D. A. Dornfeld and M. F. DeVries, "Neural Network Sensor Fusion for Tool Condition Monitoring," *CIRP Annals*, vol. 39, pp. 101–105, 1 1990.

[10] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," *IEEE Intelligent Vehicles Symposium, Proceedings*, vol. 2015-August, pp. 1094–1099, 8 2015.

[11] R. Rajamani, *Vehicle Dynamics and Control*. 2 ed., 2005.

[12] D. Berjoza, "Engineering for rural development research in kinematics of turn for vehicles and semitrailers," *Latvia University of Agriculture, Faculty of Engineering, Motor Vehicles Institute*, 2008.

[13] G. Klančar, A. Zdešar, S. Blažič, and I. Škrjanc, "Motion Modeling for Mobile Robots," *Wheeled Mobile Robotics*, pp. 13–59, 2017.

[14] L. Teschler, "How GMR wheel speed sensors help advance vehicle control and braking systems," 2019.

[15] Murata Manufacturing Co, "SCC1300-D02 Combined Gyroscope and 3-axis Accelerometer with digital SPI interfaces," *82113000 datasheet*, 2022.

A Appendix Chapter 2

A.1 Car parameters

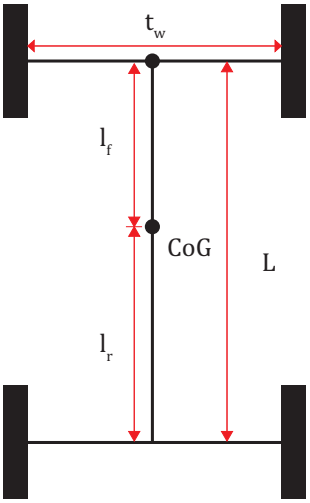

Figure A.1: *The car parameters from chapter 2 visualized*

Table A.1: *Values of the car parameters shown in Table A.1*

Parameter	Value $[m]$
t_w	1.5
L	2.0
l_r	1.2
l_f	0.8
r	0.3