Master's Thesis

Vehicle Speed Estimation Based on Artificial Intelligence Algorithms

Author:
Andrzej Skrodzki

Budapest University of Technology and Economics
Department of Control for Transportation and Vehicle Systems

Supervisor:
Donát Rácz

Master's Thesis,
October 8, 2025

Contents

Introduction

Accurate vehicle speed estimation plays a critical role in modern automotive systems, contributing to various functionalities such as tire slip calculation, advanced driver-assistance systems (ADAS), intelligent transportation systems (ITS), and autonomous driving technologies. Conventionally, the measurement of vehicle speed has been accomplished through two primary methods: direct measurement via wheel speed sensors or indirect derivation using GPS data. However, it must be acknowledged that these methodologies are not without their limitations. The functionality of wheel speed sensors can be compromised by factors such as wheel slip or defects in the sensor itself. Concurrently, the reliability of GPS signals is often compromised in specific environments, including tunnels, urban canyons, and dense forests.

In this context, the Controller Area Network (CAN) bus, which facilitates communication between electronic control units (ECUs) in vehicles, offers a promising alternative source of information for estimating vehicle speed. The CAN bus transmits various sensor readings and control signals continuously, with many of these signals being correlated with vehicle dynamics. The utilization of these signals for speed estimation has the potential to enhance system robustness, particularly in scenarios where traditional methods are rendered ineffective.

The advent of artificial intelligence (AI) has led to a surge in the utilization of data-driven methodologies for addressing intricate estimation challenges. Artificial intelligence (AI) models, particularly those designed for time-series data, such as recurrent neural networks (RNNs), are well-suited to capture the temporal dependencies and nonlinear relationships inherent in CAN signals. When trained effectively, these models can provide real-time, accurate estimations of vehicle speed based solely on CAN data.

This thesis explores the development and validation of AI-based models for real-time vehicle speed estimation using CAN signal data from a passenger car. The estimation performance is evaluated against GPS measurements, which serve as ground truth. Moreover, the study explores hybrid approaches that integrate artificial intelligence (AI) with conventional filtering methods, with the objective of combining the strengths of both paradigms. The enhancement of the reliability and accuracy of speed estimation contributes to the development of safer and more intelligent vehicle systems.

Chapter 1

Vehicle speed

1.1 Defining Vehicle Speed

The speed of an arbitrary point in space is defined as the magnitude of the velocity vector of that point, that is considered in the inertial reference frame ($\mathcal{R}_{[x_0,y_0,z_0]}$, later denoted as $\mathcal{R}_0$ for simplicity). The velocity vector can be written as the time derivative of the position vector, where the position vector points from the origin of the inertial reference frame to the chosen moving point in space. However, when considering a moving vehicle in the same space, a relatively constrained set of points has to be considered instead of one moving point. In order to use the previously mentioned formula for determining the speed of the vehicle, one point has to be chosen on the vehicle that will represent the whole. This will be the center of gravity (CoG) that is the average location of the weight of the vehicle. With this in mind the vehicle speed can be determined as follows:

$$v_{CoG} = \|[\mathbf{v}_{CoG}]_{\mathcal{R}_0}\| = \left\| \frac{d[\mathbf{r}_{CoG}]_{\mathcal{R}_0}}{dt} \right\|, \tag{1.1}$$

where v_{CoG} is the vehicle speed, $[\mathbf{v}_{CoG}]_{\mathcal{R}_v}$ is the velocity vector of the CoG in the inertial reference frame and $[\mathbf{r}_{CoG}]_{\mathcal{R}_0}$ is the position vector of the CoG in the inertial reference frame. This way the velocity and the position vectors can be expressed in a 3 dimensional space as:

- position vector: $[\mathbf{r}_{CoG}]_{\mathcal{R}_0} = \begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix}_{\mathcal{R}_0}$
- velocity vector: $[\mathbf{v}_{CoG}]_{\mathcal{R}_0} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}_{\mathcal{R}_0}$

As the moving vehicle in space is considered as a 3 dimensional body, the orientation of

it can be described with a body-fixed reference frame denoted by $\mathcal{R}_{[x_v,y_v,z_v]}$ (later denoted by $\mathcal{R}_v$ for simplicity), where x_v points forward along the vehicle frame's longitudinal axis, y_v points sideways to the left along the lateral axis and z_v points upwards to complete the right-handed coordinate system. Let this coordinate system be placed in the CoG of the vehicle. With the help of this body-fixed coordinate system the previously determined velocity vector ($[\mathbf{v}_{CoG}]_{\mathcal{R}_0}$) can be expressed in the body-fixed reference frame with the help of the rotational transformation matrix $T_{\mathcal{R}_0 \to \mathcal{R}_v}$ between the two reference frames. The rotational transformation matrix can be obtained from the rotational matrices around the three axes, namely x_0, y_0 and z_0. The angles of rotation around the axes are the following:

- angle of rotation around axis x_0: roll - ϕ
- angle of rotation around axis y_z: pitch - θ
- angle of rotation around axis z_0: yaw - ψ

From here the transformation matrix is as follows:

$$T_{\mathcal{R}_0 \to \mathcal{R}_v} = R_z(\psi) \cdot R_y(\theta) \cdot R_x(\phi) \tag{1.2}$$

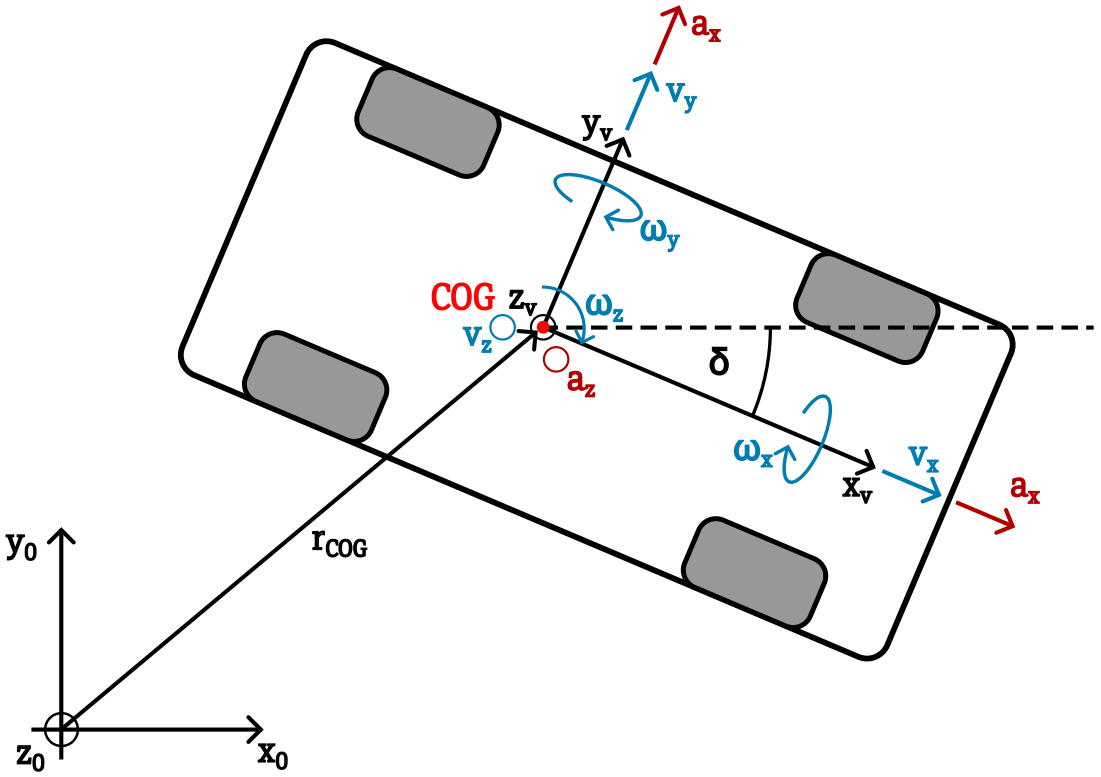


Figure 1.1: Vehicle speeds

1.2 Measurements for determining vehicle speed

There are various ways to determine the speed of the vehicle. In this section the focus is on exploring these different ways and analyzing them.

1.2.1 Engine speed

The first way is calculating the wheel and vehicle speed from the rotational speed of the engine [1]. It is a rather theoretical calculation but it gives a simplified overview how the engine is connected to the wheels through the gearbox and the differential. The setup can be seen on Figure 1.2.

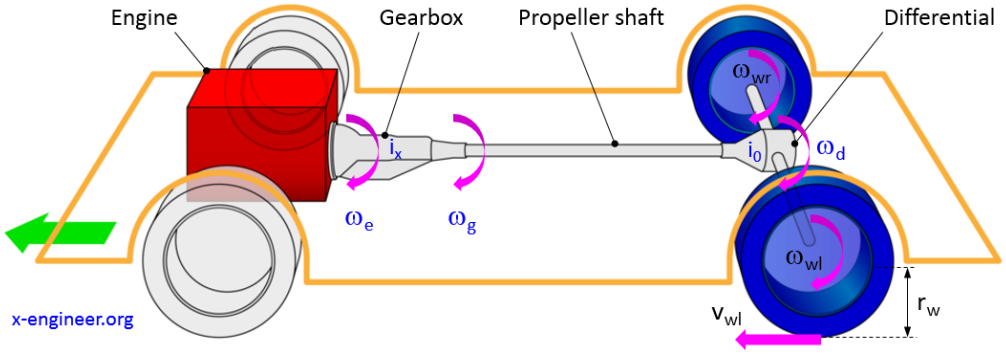


Figure 1.2: Calculating vehicle speed from engine speed [1]

The formula for determining the vehicle speed is the following:

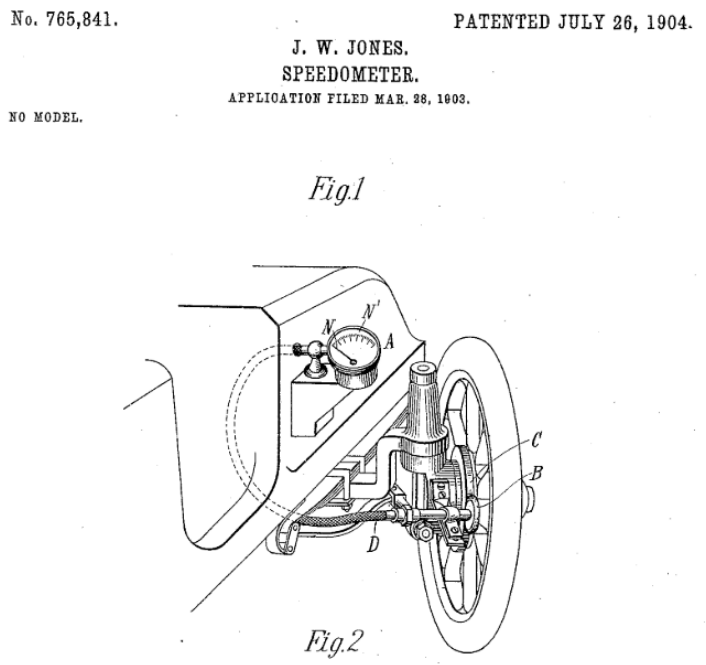
$$v_v = \frac{N_e \cdot \pi \cdot r_\omega}{30 \cdot i_x \cdot i_0} \left[\frac{m}{s}\right], \tag{1.3}$$

where N_e [rpm] is the engine speed, r_ω [m] is the wheel radius, i_x [-] is the gear ratio of the engaged gear and i_0 [-] is the gear ratio of the differential.

The main advantage of this method comes from its simplicity. Based on a one measured data, that is the engine shaft's rotational speed (N_e), the longitudinal vehicle speed can be determined. Contrary to this it can only give a estimate due to the simplifications it considers. It doesn't take in account the slip in the clutch between the gearbox and the engine shaft, the longitudinal and lateral slip of the tires, the difference in speed of the left and right wheels in case of traveling in a curve, the case when the gearbox and the engine are detached while the vehicle is being in neutral (gear 0) and when the wheels are locked but the vehicle is still motion. Considering the points mentioned above, this method could be used in highway driving where the vehicle travels with constant high speed on a straight path with little curvature.

1.2.2 Wheel speed

Following the previous method where the measured unit was the rotational speed of the engine's shaft, here the shift goes to measuring the rotational speed of the vehicle's wheel. This is done by a speedometer. The first speedometer is credited to Josip Belušić [2]. His invention in 1888 was based on the electromagnetic induction which worked the following way. A flexible rotating cable is connected to the vehicle wheel's rotating shaft via a friction disk or a pair of gears that translates the rotational motion to a disk shaped magnet. This rotating magnet creates a fluctuating magnetic field which induces a rotating movement in the speed cup that surrounds the magnet. However the speed cup's movement is limited by a spring. This spring will allow greater rotation when greater torque applies to the speed cup that is in the case of faster rotation from the magnet. The needle that showed the speed of the vehicle is linked to the speed cup. This way the needle moved proportionally to the speed of the vehicle [3]. Later this principle was patented by Otto Schulze in 1903 [4] and Joseph W Jones in 1904 [5]. Both of the patents share the same principle.


No. 765,841. PATENTED JULY 26, 1904.

J. W. JONES.

SPEEDOMETER.

APPLICATION FILED MAR. 28, 1903.

NO MODEL.

Fig.1

Fig.2

Figure 1.3: Speedometer patented by Joseph W Jones in 1904 [5]

The rather mechanical method described above was in use for several decades but it has multiple downsides: wear of the elements due to the mechanical connections, uncertainty coming from the mechanical measurement method and the fact of having the speed from only one wheel. To compensate the points mentioned before, electric wheel speed sensors were introduced to enhance the precision of the measurements. There are two types of electronic wheel speed sensors: passive and active.

The principle of the passive wheel speed sensor is the following. A toothed reluctant ring is mounted on the rotating wheel. Above this ring a variable reluctance (VR) sensor is placed. When the wheel starts to rotate each tooth changes the magnetic flux through the coil of the sensor which induces an alternating voltage signal. The frequency of this analog sine wave is proportional to the rotational speed. However, the active wheel speed sensor uses a semiconductor magnetic sensor (Hall or magneto-resistive sensor) and a magnetic encoder ring attached to the rotating wheel. As the ring rotates, the magnetic field changes are detected by the magnetic sensor. These changes are then converted to square wave digital signals [6]. The active sensors can overall provide a more precise measurement than the passive sensors.

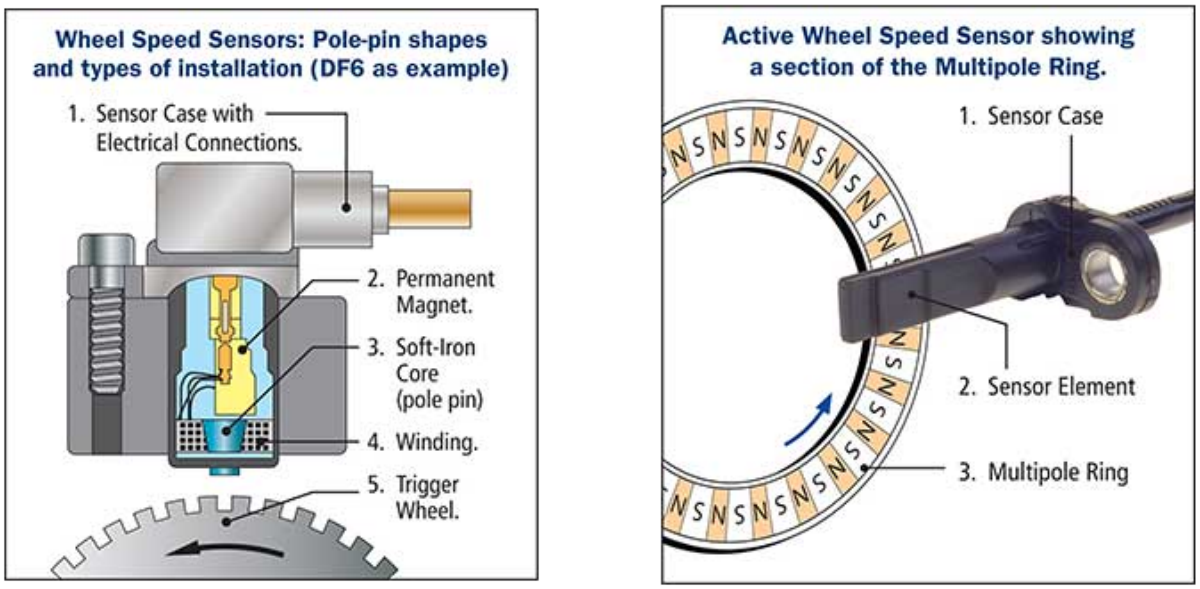


Figure 1.4: Passive (left) and active (right) wheel speed sensors from [6]

In the case of these wheel speed sensors, the output is a rotational speed. From this rotational speed the ground speed of the specific wheel can be determined with the help of the diameter of the wheel. This measurement can be done for all the wheels, which will serve as an input for later discussed estimator algorithms.

1.2.3 Inertial Measurement Unit

The inertial measurement unit (IMU) can also provide relevant measurement data to determine the vehicle speed. An IMU consist of a set of linear accelerometers and gyroscopes to determine the acceleration along and the rotational rate around the longitudinal, lateral and vertical axes. This gives two vectors: acceleration $\mathbf{a}$ and angular velocity vector

$\boldsymbol{\omega}$ in the body fixes coordinate system:

$$\mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \text{ and } \boldsymbol{\omega} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{1.4}$$

Before calculating the velocity, the gravitational vector has to be deducted from the measured acceleration vector that can be done with the help of the angular velocity vector. From here, as the velocity vector's time derivative equals the acceleration vector, the velocity can be derived by an integral:

$$\mathbf{v}(t) = \mathbf{v}(t_0) + \int_{t_0}^{t} \mathbf{a}(\tau) d(\tau) \tag{1.5}$$

It is important to mention here that this measurement method is not robust by itself. Apart from setting the initial velocity of the vehicle, the biases in the accelerometers lead to integration drift that accumulates into large errors over time. To mitigate this issue this method has to be combined with other velocity measurements and estimators.

1.2.4 Global Navigation Satellite System

Global Navigation Satellite System (GNSS) refers to any constellation that provides global positioning, navigation and timing services. The currently available GNSS systems in Europe are GPS, Galileo, Glonass and BeiDou [7].

The velocity of the vehicle can be obtained by exploiting the raw Doppler, the carrier phase or the pseudorange measurements with a GNSS receiver. These measurements are the following:

- Raw Doppler (RD) measurement:
  - The RD measurement captures how much does the frequency of the signal change between transmitting and receiving.

$$D = f_{transmitted} - f_{received} \tag{1.6}$$

  - The Doppler shift D is related to the change of the distance between the satellite and the receiver. From the Doppler shift the range rate $\dot{\rho}$ can be determined that is the radial velocity v_r along the range between the satellite and the receiver.

$$\dot{\rho} = v_r = \lambda \cdot D = \dot{\rho} + c \tag{1.7}$$

- Carrier phase measurement:

- Pseudorange differentiation:

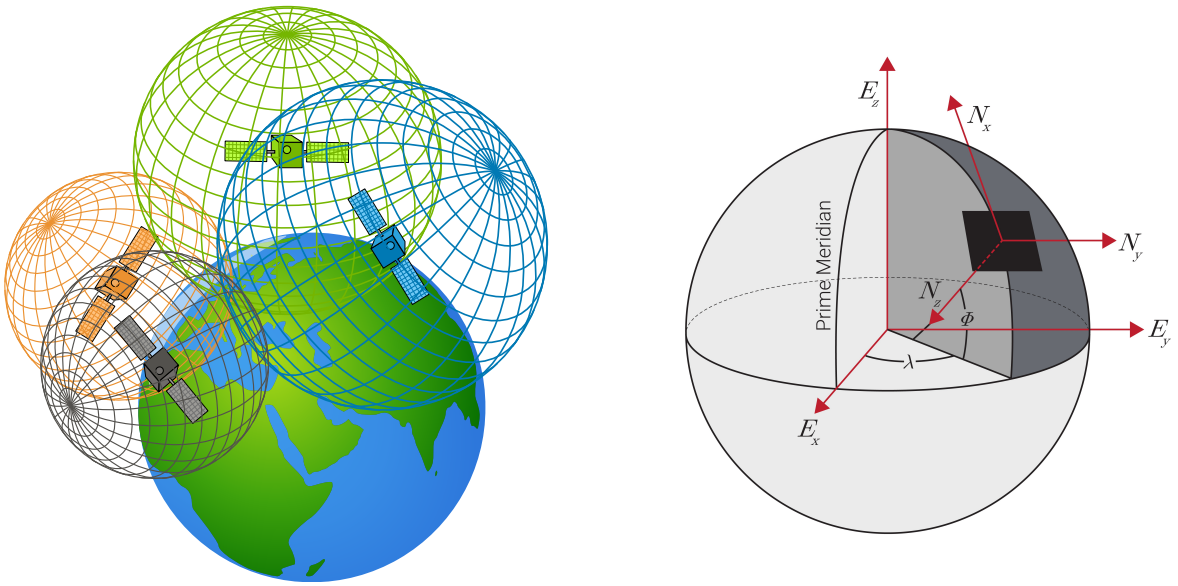


Figure 1.5: Trilateration (left) from [8] and ECEF coordinate system (right) from [9]

1.3 Estimating vehicle speed

1.3.1 Accelerometer and speedometer

The simplest method to estimate the vehicle speed is to take the measurements from the speedometers at all the wheels. The longitudinal speed of each wheel can be determined from the rotational speed data from the sensors. These values then can be transformed into the CoG of the vehicle. There the transformed longitudinal velocities can be compared and the largest one can be chosen.

$$v_{CoG} = max[v_{FR}; v_{FL}; v_{RR}; v_{RL}], \tag{1.8}$$

where v_{FR}, v_{FL}, $v_{RR}$and v_{RL} are the transformed velocity values from each wheel.

Another simple algorithm for vehicle speed estimation is presented in [10]. Here the vehicle speed is determined by the measurements from the IMU's longitudinal accelerometer and the wheel speed sensors. The formula that estimates the speed is described as follows:

$$v_{CoG}(k) = v_{simple}(k_{nobrake}) + \sum_{i=k_{nobrake}}^{k} a_{meas}(i) \cdot dt, \tag{1.9}$$

where the k is the time step counter, $k_{nobrake}$ is the last time index where there was no braking, a_{meas} is the acceleration value form the IMU's longitudinal accelerator and dt is the time sample. This method works in a way that estimation is taken apart into two

categories based on whether the vehicle is barking or not. The simplicity comes at the price of inaccuracies. The estimator does not perform well when accelerometer offset is set and the dynamic change of the wheel radius is not implemented. An other difficulty comes from dealing with the noisy measurements from the wheel speed sensors and the IMU.

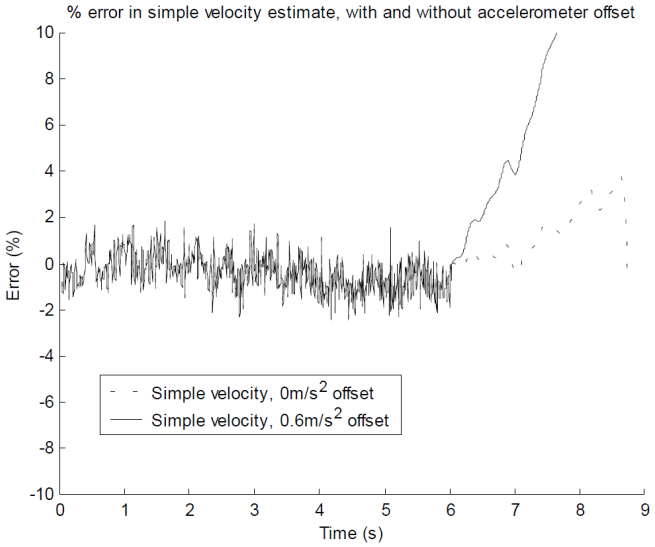


Figure 1.6: Simple algorithm for estimating vehicle speed form [10]

1.3.2 Kalman filter

To elevate the performance of the previously described methods, a Kalman filter can be added to the system. At a high level the Kalman filter estimates the vehicle speed from noisy sensor data by iterating between the following two steps:

1. Prediction
    - The filter takes the last estimate and predicts the next state's longitudinal or lateral velocity based on the simple dynamic motion model.
    - The filter provides an estimate and a uncertainty measure for the prediction.
2. Correction
    - Based on the predicted data and the new measurement data the filter calculates an optimal blend of the two values.
    - A posterior estimate is determined that is more accurate and less noisy than the pure measurement data.

Following this structure a Kalman filter is proposed in [11] to estimate the vehicle's speed. To enhance the accuracy of the estimator, 4 different driving situations are described with 4 different covariance matrices.

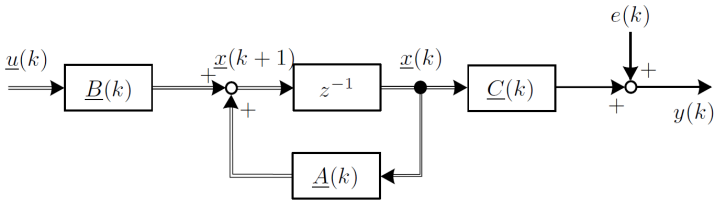


Figure 1.7: Vehicle speed estimation with Kalman filter from [11]

To elevate the usage of Kalman filter a more sophisticated and non-linear vehicle model can be used to predict the vehicle speed. In [12] an Extended Kalman filter is introduced with vehicle model that runs parallel with the system itself. The overview of the filtering approach can be seen on figure 1.8

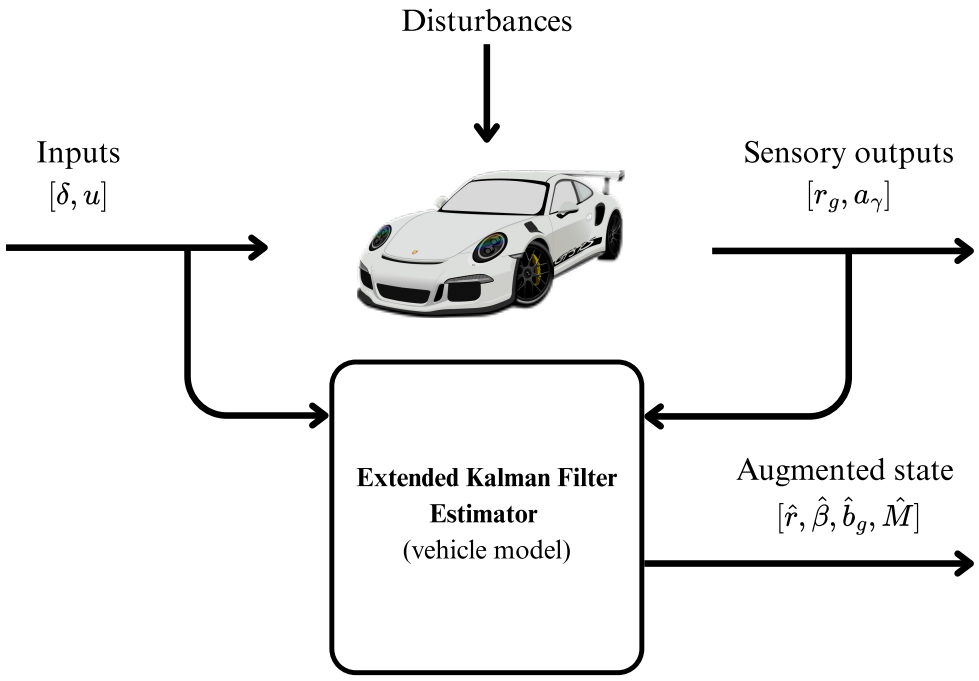


Figure 1.8: Extended Kalman filter for vehicle speed estimation from [12]

1.3.3 Fuzzy logic

An other approach to estimate the vehicle speed is using fuzzy logic. Fuzzy logic offers heuristic approach to handle uncertainty and imprecision of the sensors. In [11] a comprehensive method is described using the Fuzzy logic to determine the vehicle speed using different driving scenarios.

The main motive behind the proposed estimator is determining the weights of the sensor signal values. All the speed values coming from the senors receive a weighting

factor k_i. The weighted average is then formulated as follows:

$$\hat{v}_{CoG}(k) = \frac{\sum_{i=4}^{4} k_i \cdot v_{Ri,C}(k) + [\hat{v}_{CoG}(k-1) + T_S \cdot a_{X,C}(k)]}{\sum_{i=1}^{5} k_i}, \tag{1.10}$$

where the k_i weights are determined with the rules of the fuzzy logic. The overall scheme of the algorithm can be seen on figure 1.9.

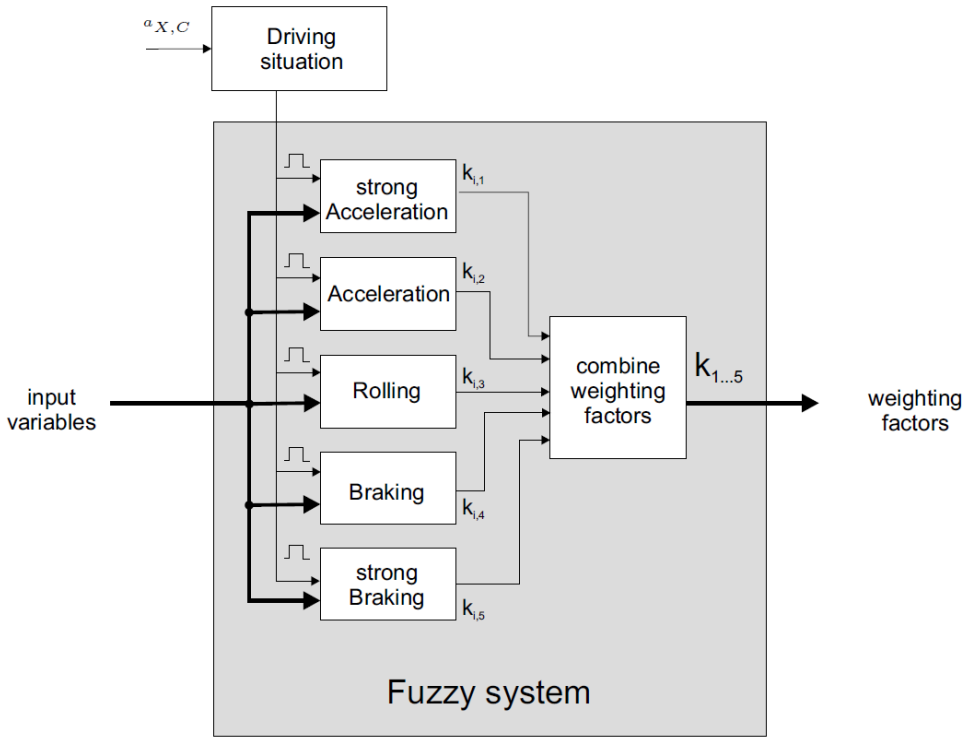


Figure 1.9: Vehicle speed estimation with Fuzzy logic from [11]

1.3.4 GNSS

There are multiple algorithms to determine the velocity of the vehicle using the GNSS system. In article [13] 4 algorithms are described and compared. These methods are the following: Raw Doppler (RD), Time-Differenced Pseudorange (TDPR), Time-Differenced Carrier Phase (TDCP) and Double-Differenced Carrier Phase (DDCP). These algorithms were both tested in static and dynamic environments and postprocessed after the tests.

The most accurate algorithm in dynamic environments is the RD method that utilizes the Doppler shift from the carrier phase difference of the transmitted and received signal. The measurement model can be written as:

$$\lambda D = \dot{\rho} + c(d\dot{t}_R - d\dot{t}_S) + \dot{I}_\rho + \dot{T}_\rho + \dot{\epsilon}_\rho, \tag{1.11}$$

where λ is the wavelength, D is the Doppler shift, $\dot{\rho}$ is the rate of the pseudorange, c is

the speed of light in vacuum, $d\dot{t}_R$ and $d\dot{t}_S$ are the satellite and receiver clock drifts, $\dot{I}_\rho$ and $\dot{T}_\rho$ are the rates of change in the ionospheric and tropospheric delays and $\dot{\epsilon}_\rho$ combines the unmodeled error and observation noise. From this model the velocity of the vehicle can be determined in the global ECEF frame. The tests with this method resulted a [cm/s] level accuracy that is precise and accurate enough for driver assistance features.

The significant advantage of this algorithm is definitely the precision that it provides and can be used to enhance the previously described estimators. The disadvantages are that the GNSS receivers are not standard in passenger and commercial vehicles leaving little space for wide-spread usage of the advantages, the 10 [Hz] sampling time is relatively low that requires some type of interpolation between the samples and that the determined velocity is not in the vehicle's body fixed frame. For determining the longitudinal and lateral velocities the heading angle is required that can be calculated if two receivers are applied to the vehicle. The last disadvantage comes from the situations when the signals from the satellites are not visible in case of driving in a tunnel or urban areas with tall buildings.

1.3.5 Artificial Intelligence

After discussing the more conventional estimators that rely on deterministic formulas for calculating the vehicle speed, let the focus be shifted to the main topic of this thesis: Artificial Intelligence (AI) for vehicle speed estimation.

Artificial intelligence has been a greatly investigated field in the recent years for a wide range of applications. Despite the robustness and highly safety critical approach of the vehicle industry, the investigation of possible applications has been started. There are two main approaches:

1. Integrating AI algorithms to previously developed and used algorithms and models. This way an AI algorithm can be seen as a way to enhance the already developed and tested estimators.

2. Replace the previously used algorithms with a trained AI algorithm.

In article [14] a physics-constrained neural network (PCNN) has been introduced. This approach merges the well established and developed physical models with the advantages of a neural network (NN). Here the created network is capable of learning the coefficients of a single track vehicle model that can predict the vehicle's movement. The main advantage of this application is that by using the known dynamic and physical properties of a vehicle, the coefficients can be determined faster and more precisely than if there were no established constrains.

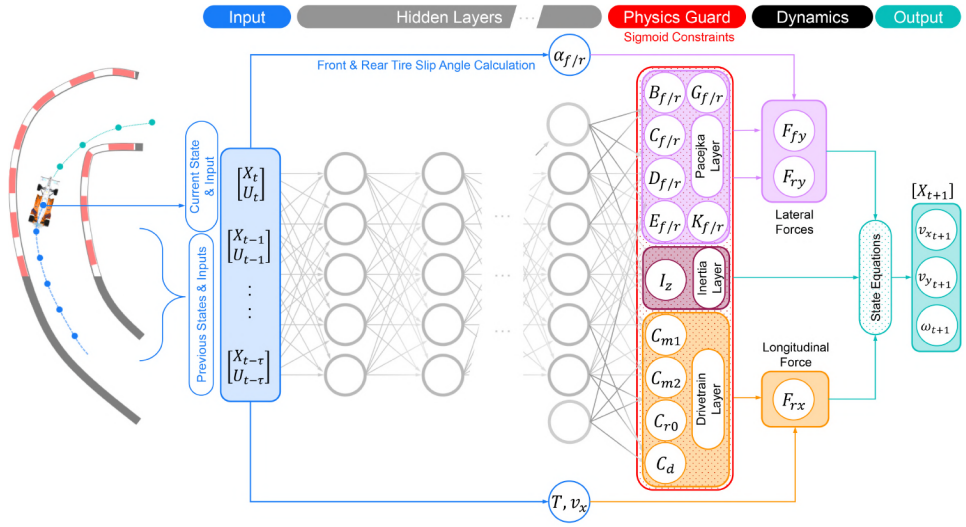


Figure 1.10: Physics constrained neural network from [14]

A similar approach was investigated in [15] where a physics embedded neural network in combination of a vehicle model was developed where the previously discussed advantages were exploited. The outcome of this investigation was the performance shift from the baseline neural network model showing faster learning speed, higher accuracy and better generalization.

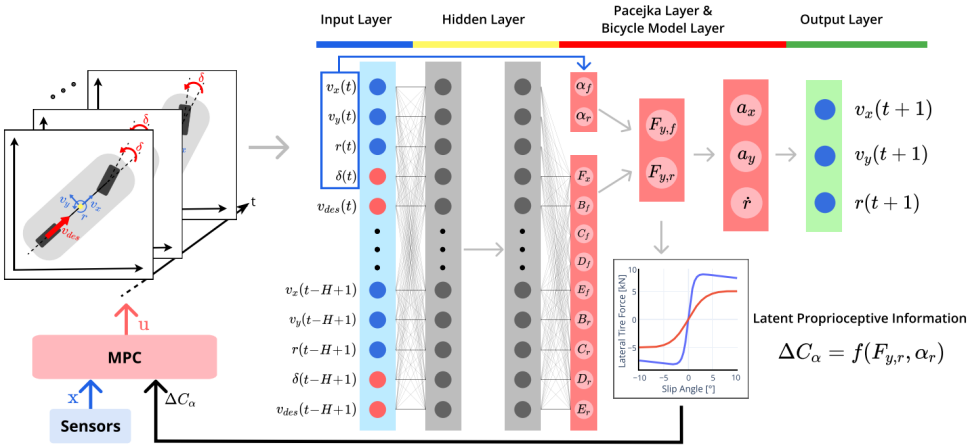


Figure 1.11: Physics embedded neural network for state estimation from [15]

An other approach is to combine AI algorithms with Kalman filtering to enhance state estimation [16]. Here a method for combining Long-Short Term Memory network (LSTM), Transformers and Expectation-Maximization - Kalman Filter (EM-KF). This combined approach of TL-KF (Transformers, LSTM - Kalman filter) is capable of capturing long term dependencies and model time-series data. The structure of the developed model can be seen on figure 1.12

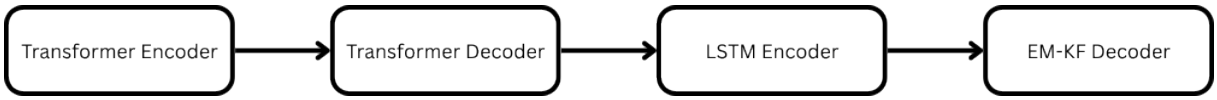


Figure 1.12: TS-KF model from [16]

1.4 Conclusions

During the literature review a comprehensive overview was presented about the used and developed vehicle speed estimators. It is visible that the well established estimators, like different model based estimators with Kalman filters have been there for longer time. In the meantime, the newly emerging field of AI has a promising future in state estimation aiding the already used estimator algortihms.

Chapter 2

Development environment

In this chapter the environment of the development will be described, where the structure of the vehicle and its sensors, and the requirements of the estimator will be discussed.

2.1 Available sensor data

A modern vehicle is equipped with multiple ECU-s, numerous sensors and actuators that support and enhance driving quality. When considering vehicle motion control applications the mostly used and available sensors are the following:

Sensor Type	**Variables**	**Unit of Measurement**
Wheel speed sensor	$\omega_{FR}, \omega_{FL}, \omega_{RR}, \omega_{RL}$	[°/s]
Wheel torque sensor	$M_{FR}, M_{FL}, M_{RR}, M_{RL}$	[Nm]
Acceleration from IMU	a_x, a_y, a_z	[m/s^2]
Angular rates from IMU	$\dot{\alpha}$, $\dot{\beta}$, $\dot{\gamma}$	[°/s]
Brake pressure sensor	$P_{FR}, P_{FL}, P_{RR}, P_{RL}$	[bar]
Road wheel angles	$\delta_{FR}, \delta_{FL}, \delta_{RR}, \delta_{RL}$	[°]
Steering wheel angle	φ	[°]

Table 2.1: Overview of sensor types, variables, and units

GNSS receivers can be also considered as precise sensors that are available during the development phase but not on the vehicles from the production line in order to keep the wider availability. Therefore the calculated longitudinal and lateral speeds from the GNSS measurements will only serve for reference, validation, testing and training purposes. The mentioned sensor units can be seen on figure 2.1.

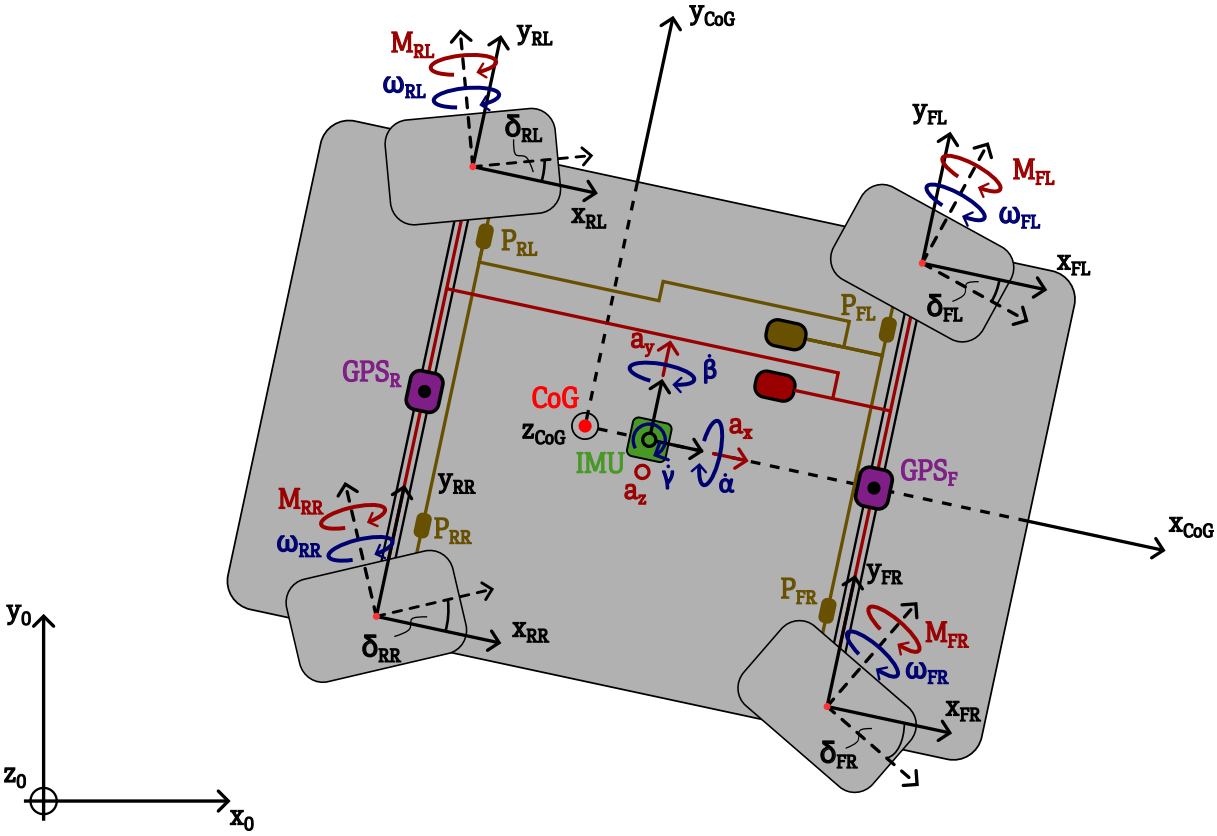


Figure 2.1: Available sensor data for speed estimation

The structure of the available sensor on the CAN bus data that was used in the further development is the following. Sampling time is 2 $[ms]$ for all the sensors except the speed values provided by the GNSS receivers, that is 100 $[ms]$. Above this, the sensor data is collected in a large enough buffer to provide all the values at the same time. This way at every 2 $[ms]$ all the inputs will arrive at the same time to the estimator. The speeds from the GNSS receivers will be handled separately from base structure. The structure can be seen in table 2.2:

time	a_x	ω_{FR}	ω_{FL}	ω_{RR}	ω_{RL}	M_{FR}	M_{FL}	M_{RR}	M_{RL}	a_y	a_z	$\dot{\alpha}$	$\dot{\beta}$	$\dot{\gamma}$	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...
6.220	0.2819	31.3643	31.3791	31.5755	31.5894	0	0	19.087	19.087	0	0	0	0	0.2329	...
6.222	0.2819	31.3643	31.3791	31.5755	31.5894	0	0	19.087	19.087	0	0	0	0	0.2349	...
6.224	0.2819	31.3643	31.3791	31.5755	31.5894	0	0	19.087	19.087	0	0	0	0	0.2369	...
6.226	0.2819	31.3643	31.3791	31.5755	31.5894	0	0	19.087	19.087	0	0	0	0	0.2389	...
6.228	0.2819	31.3643	31.3791	31.5755	31.5894	0	0	19.087	19.087	0	0	0	0	0.2409	...
6.230	0.2893	31.511	31.5304	31.7222	31.7406	0	0	19.087	19.087	0	0	0	0	0.2429	...
6.232	0.2893	31.511	31.5304	31.7222	31.7406	0	0	19.087	19.087	0	0	0	0	0.2449	...
6.234	0.2893	31.511	31.5304	31.7222	31.7406	0	0	19.087	19.087	0	0	0	0	0.2469	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	...

Table 2.2: Example of available data structure with 2$[ms]$ sampling time

2.2 Ways to use AI to estimate vehicle speed

Before advancing to the possible algorithms for vehicle speed estimation using AI, some high level requirements shall be set to be able to validate the results. The requirements are the following:

1. The developed estimator shall provide more accurate vehicle speed values than a model based vehicle speed estimator.
2. The developed estimator shall be tested and validated with data that was not used for training purposes.
3. The developed estimator shall be trained with data that covers the range of use of a vehicle.
4. The developed estimator shall be applicable in real time usage in a test vehicle.

With these requirements in mind the possible solutions can be discussed. The algorithms can be categorized into two groups. Solutions where the vehicle speeds are directly estimated by the AI algorithm from the sensor input data (1) and where the AI algorithm is combined with an already existing estimator (2). Both of these solutions offer great potential in estimating vehicle speed.

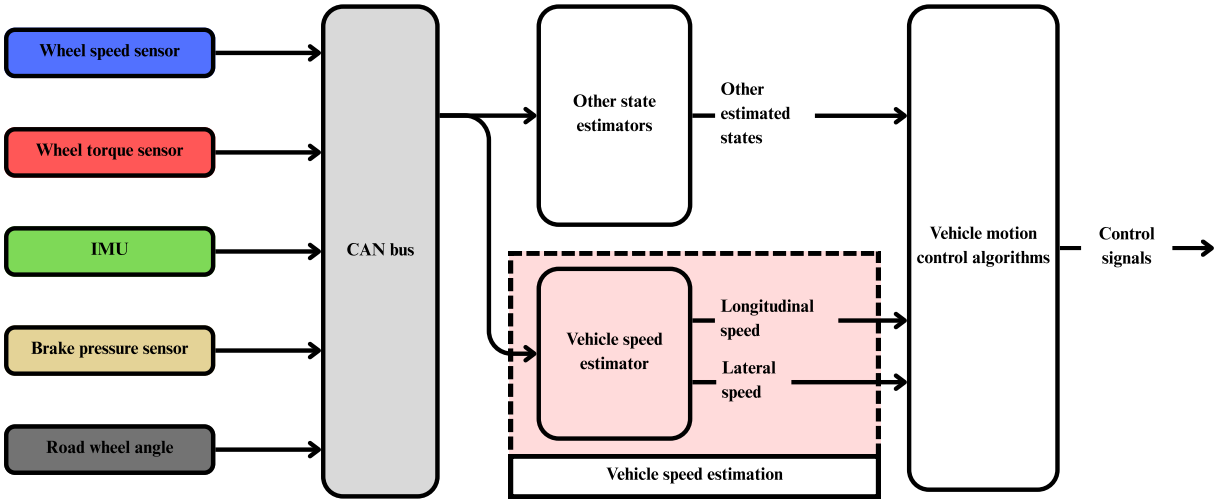


Figure 2.2: Vehicle speed estimation in the simplified control

2.2.1 Direct estimation

When directly estimating any state of the vehicle, the structure is the following. The trained AI model receives the sensor data as inputs and outputs the state for which it was trained. In this thesis the state to estimate is the vehicle's longitudinal and lateral speed.

The question now arises what models are suitable for this application? The possible models to use in this scenario are models that can handle time series data well and can estimate an output based on the sequence it was provided with. The possible models are the following:

- Basic/simple Recurrent Neural Network (RNN)
- Long-short Term Memory (LSTM)
- Gated Recurrent Unit (GRU)
- Transformer for time series
- Temporal Convolutinal Network (TCN)

The teaching phase of these models will be relatively similar. All models will be taught on previously recorded sensor data coming from vehicle simulations and real-life vehicle tests. These recorder datasets will be extended with a longitudinal and lateral speed values calculated from the GNSS measurements. This way every input will be labeled with the desired output.

2.2.2 Combination of existing estimators with AI model

As it can be seen in the literature review that there are already existing applications of combining an AI model with an already established state estimator. The key advantages of this approach could be improved robustness, adaptive parameter tuning of the applied filter with implementing AI and non-linear error compensation.

Some possible implementations can be the following:

- Residual learning:
  - A neural network predicts the residual error from a standard Extended Kalman Filter (EKF) applied for vehicle speed estimation. The final speed will result from the combination of the estimated and the neural network's value for the residual.
- Neural network aided Kalman filter:
  - The parts of the Kalman filter are replaced with trained modules
  - E.g when the predict module is replaced by a neural network that determines the next state.
- Adaptive gain scheduler:
  - The AI module dynamically tunes the noise covariance of the implemented Kalman filter based on the CAN signals.

2.2.3 Advantages and disadvantages of AI application

Prior to moving on to describing the developed models and estimators it is important to state the advantages and disadvantages when applying an AI model to a vehicle's control system. In case of a vehicle that is designed to be in contact (to transport or be driven by) with humans, the developed control system has to comply with specific international standards (e.g ISO 26262) that assures the safety of the system.

The main advantage when implementing AI to a vehicle control system is the possibility of capturing non-linear relationships between the CAN signals to estimate an arbitrary state. This property makes it possible to establish complex relationships between state variables without the knowledge of more complicated physical formulas.

On the other hand, there are some disadvantages that need to be considered. An implemented AI can be considered as a black box where the inner operation makes it difficult to validate from safety relevant perspective. This also creates a part of the system that lacks transparency and explainability. An other disadvantage may arise from missing out on data quality and availability. Resource constraints in embedded systems may also block usability if the created model uses more computational power or memory than the available amount.

Chapter 3

Direct estimation

In this chapter the steps of developing the chosen AI models for estimating vehicle speeds will be discussed. First, the simple recurrent neural network (RNN) will be presented. This will be followed by two more specific RNN models: long short-term memory (LSTM) and gated recurrent unit (GRU). All of these models will be trained and tuned to directly estimate the longitudinal and lateral speeds of the vehicle from the previously discussed CAN signals.

3.1 Recurrent Neural Network - RNN

3.1.1 Theoretical background

Contrary with regular feedforward neural networks where the output is determined only by the current values of the inputs, the RNN is designed in a way that after each step the information is fed back to the network. This way the output will be dependent on the previous inputs as well and the model will have a memory like property giving the possibility of handling temporal data.

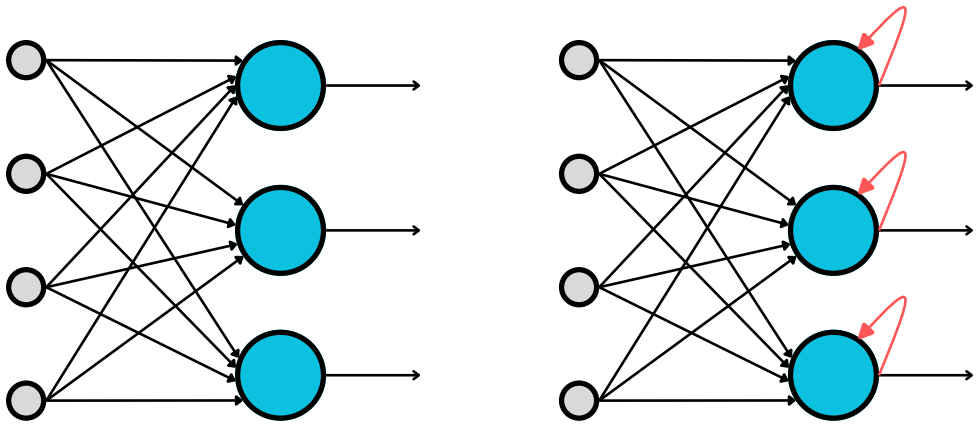


Figure 3.1: Feedforward nn (left) compared to recurrent nn (right)

The fundamental building blocks of a RNN are the recurrent units (RU). These units hold an inner state that connect the in- and outputs. The inner states maintain information about previous inputs in a sequence. To understand how does the network's architecture function, let it be unfolded in time. This can be seen on figure 3.2.

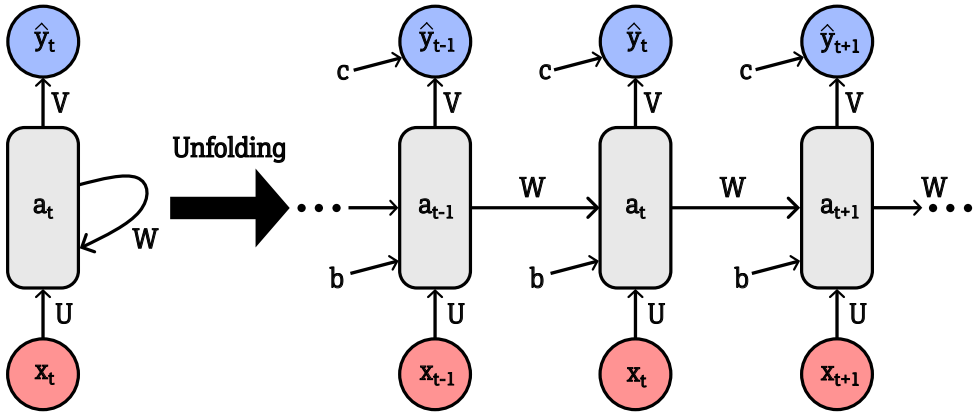


Figure 3.2: Unfolding the architecture of a RNN in time

It is visible that at time step t the output vector ($\hat{y}_t$) is determined by the state of the hidden layer (a_t). However the hidden layer is determined by the previous state of the hidden layer (a_{t-1}) and the input vector (x_t) of the current timestep t, that can be written as

$$a_t = f(a_{t-1}, x_t; \theta), \tag{3.1}$$

where θ represents the biases (b) and weights (U, W) and $f()$ is the activation function (e.g tanh()). This equation can be expanded as

$$a_t = f(U \cdot x_t + W \cdot a_{t-1} + b), \tag{3.2}$$

where U is the weight matrix connecting the input layer and the hidden layer, W is the weight matrix to connect the hidden layer's previous state and the current state and b is the bias vector for the hidden layer. From this the output vector can be determined as

$$\hat{y}_t = g(V \cdot a_t + c), \tag{3.3}$$

where V is the weight matrix connecting the hidden layer and the output vector, c is the bias vector for the output layer and $g()$ is the activation function (e.g softmax()).

To determine the weights and biases of the network backpropagation through time (BPTT) is used, that is an extended version of the backpropagation used in FNNs. This method is based on the error between the determined output of the network and the desired/labeled output at time step t. BPTT involves two major steps: forward pass and backward pass. These steps are detailed below:

1. Forward pass

- During the forward pass the sequence of inputs form $t = 1$ to $t = n$ passes through the network, where n is the length of the sequence.
- The output vector y_t is determined at every timestep with the use of the previously described formulas.
- The loss is calculated at every timestep between the desired (y_t) and the calculated ($\hat{y}_t$) output.
- The loss function is given by the mean squared error (MSELoss):

$$L(y, \hat{y}) = \frac{1}{t} \sum_{t=1}^{n} (y_t - \hat{y}_t)^2 \tag{3.4}$$

2. Backward pass

- During the backward pass the gradients of the loss function $L(y, \hat{y})$ are calculated with respect of the weights of the network (U, V, W).
- W.r.t the weight matrices the derivatives of the loss functions can be determined as:

$$\frac{\partial L}{\partial U} = \sum_{t=1}^{n} \frac{\partial L}{\partial a_t} \cdot \frac{\partial a_t}{\partial U} \tag{3.5}$$

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{n} \frac{\partial L}{\partial a_t} \cdot \frac{\partial a_t}{\partial W} \tag{3.6}$$

$$\frac{\partial L}{\partial V} = \sum_{t=1}^{n} \frac{\partial L}{\partial \hat{y}_t} \cdot \frac{\partial \hat{y}_t}{\partial V} \tag{3.7}$$

- The gradient values inform about the direction (increase or decrease) and the magnitude of the change in the weight matrices.
- These gradients can then be used for updating the values of each weight matrix with the help of the chosen optimizer (e.g. SGD, Adam).

Common issues when applying RNNs are exploding and vanishing gradients during the backward pass. Both of these refer to the magnitude of the calculated gradients: $\left\| \frac{\partial L}{\partial W} \right\|$. Exploding gradient problem stems from the gradient being too high which is a consequence of the learning rate being to big. On the other hand vanishing gradient refers to a gradient being near zero, which leads to slow learning.

3.1.2 Implementing a RNN

When implementing a RNN architecture for a specific application, there are some general hyperparameters to determine. These parameters can be found in table 3.1.

Hyperparameter	Description
Model Architecture:	
Hidden size	Dimensionality of the hidden (and cell) state vector at each time step.
Number of layers	Number of stacked RNN/LSTM/GRU layers (depth of the model).
Cell type	RNN variant (vanilla RNN, LSTM, GRU).
Bidirectionality	Whether to process the sequence in both forward and backward directions.
Dropout rate	Fraction of units dropped between layers to prevent overfitting.
Training Configuration:	
Learning rate	Step size for weight updates in gradient descent.
Optimizer	Algorithm for updating parameters (SGD, Adam, RMSProp, etc.).
Batch size	Number of sequences processed per gradient update.
Number of epochs	Full passes over the training dataset.
Gradient clipping	Maximum norm for gradients to prevent exploding gradients.
Sequence length	Number of time steps in each input subsequence (truncated BPTT window).
Regularization:	
Weight decay	L2 penalty on weights to discourage large values.
Early stopping	Halt training if validation loss stops improving.
Data-related:	
Normalization method	How inputs are scaled (min–max, z-score, etc.).
Input window size	How many past time steps are fed to the network.
Prediction horizon	How many future steps the network is trained to predict.

Table 3.1: Key hyperparameters for RNN-based sequence models

3.1.3 Results

The used data that was created from 4 simulations that covered 4 basic driving situations. These are: accelerate until a certain longitudinal acceleration (auax-drv), brake until a certain longitudinal acceleration (buax-brk), steer until a certain lateral acceleration (suay) and sine wave like steering (sin-steer). This way a range of driving situations can be covered. To create more datasets, random values were used within a certain range for the same simulations types.

The results can be seen on figures below.

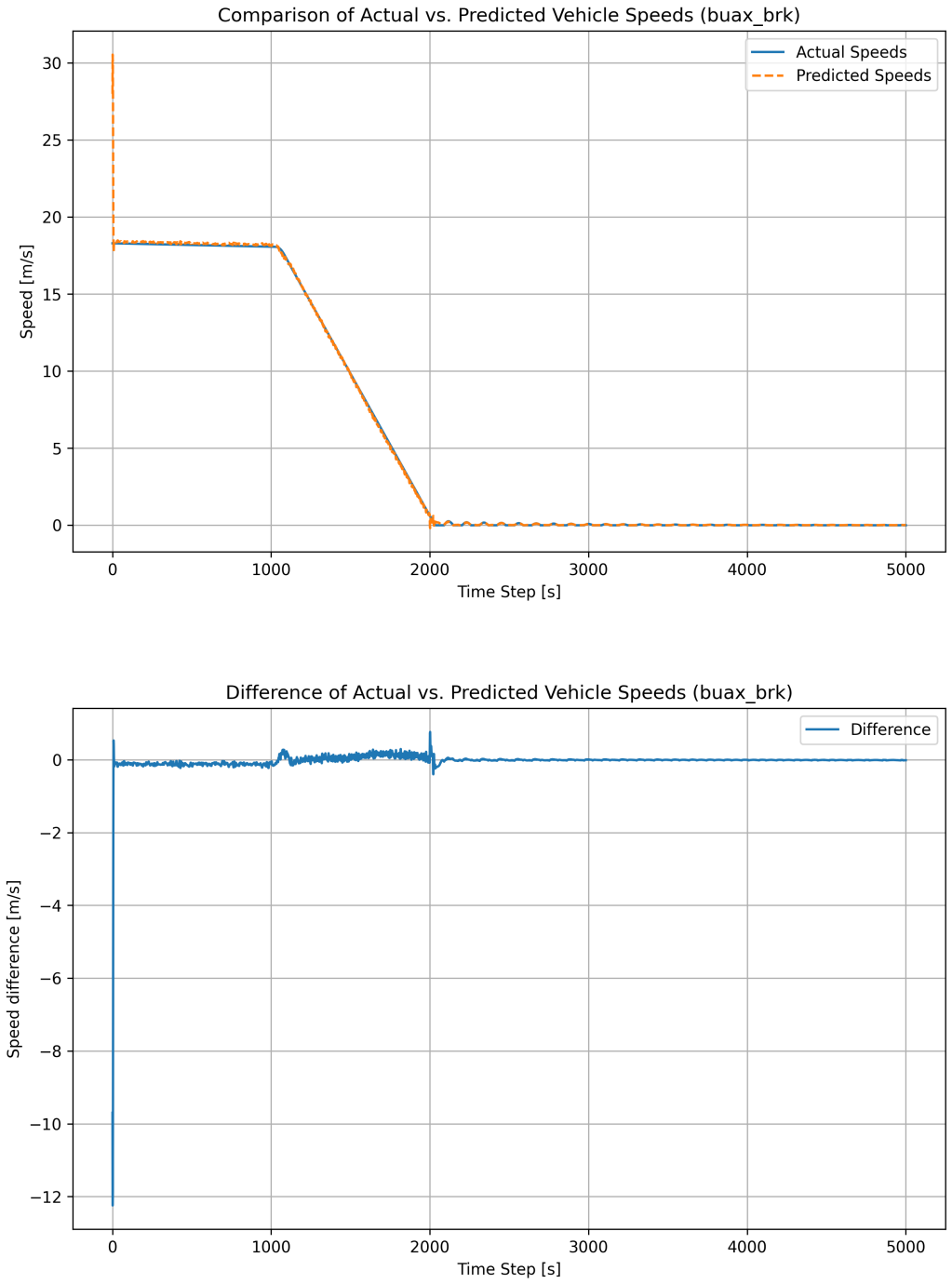


Figure 3.3: Testing speed estimation with a buay-brk validation dataset

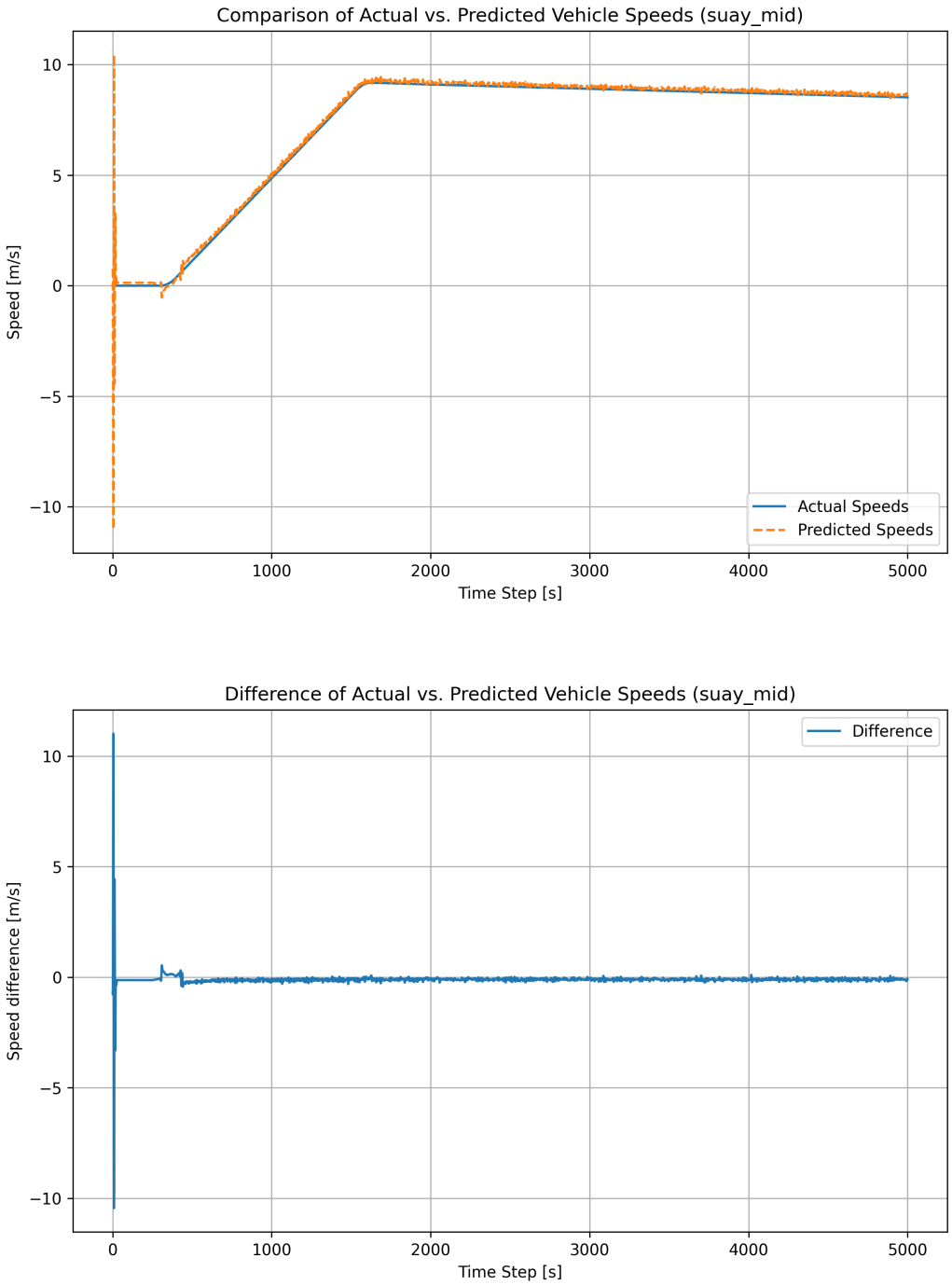


Figure 3.4: Testing speed estimation with a suay validation dataset

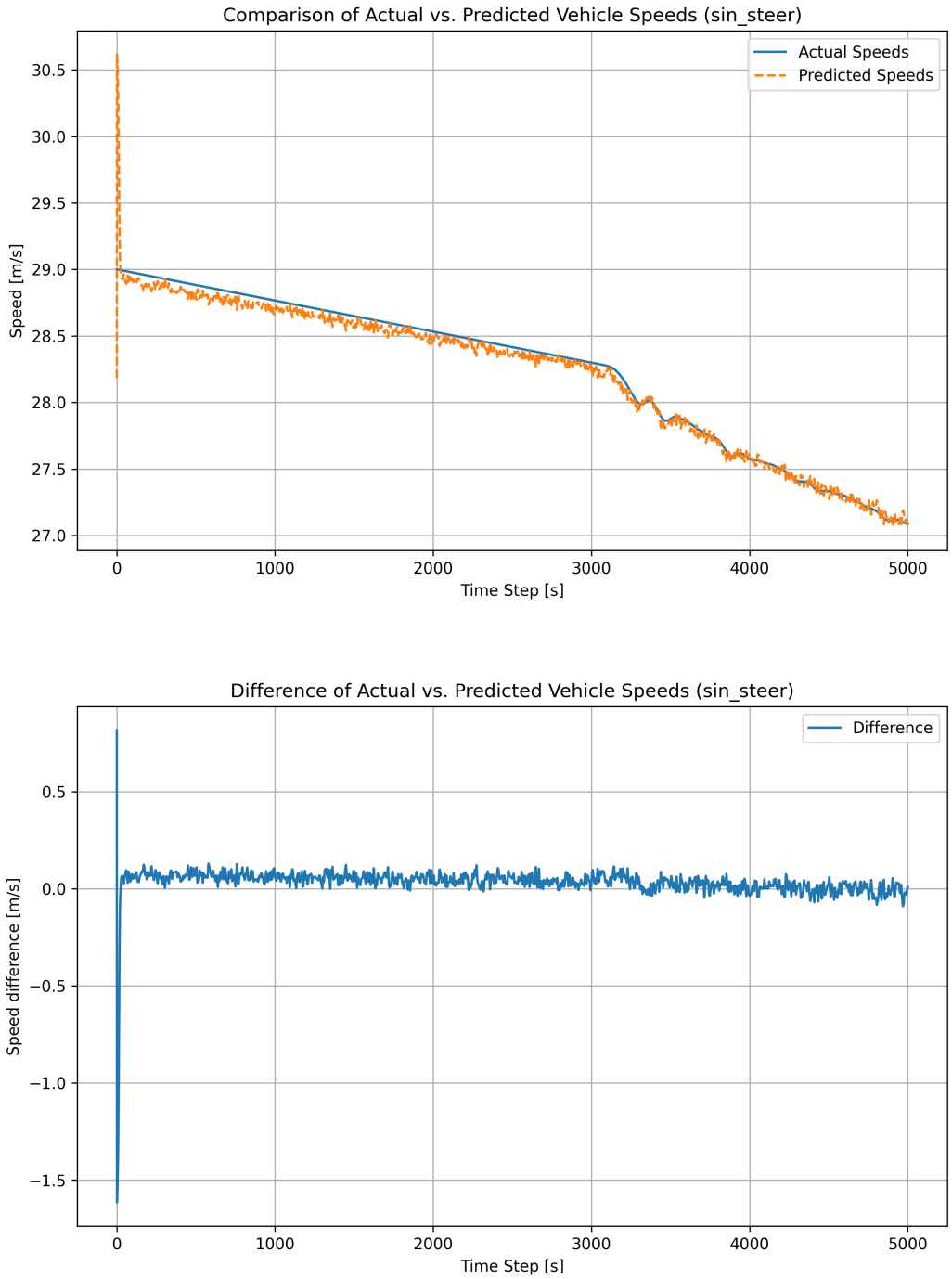


Figure 3.5: Testing speed estimation with a sin-steer validation dataset

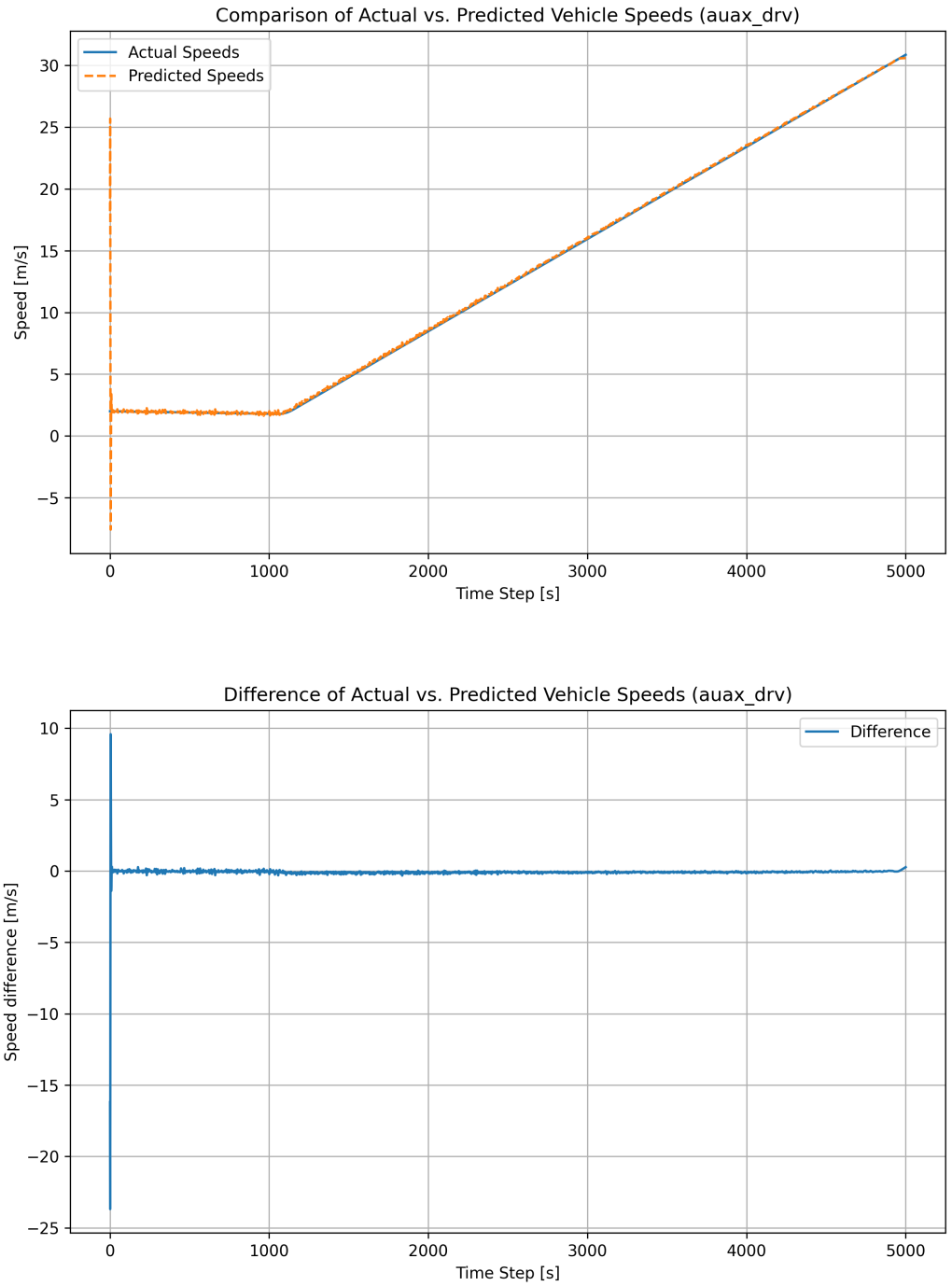


Figure 3.6: Testing speed estimation with a auax-drv validation dataset

It can be seen on the figures that the estimated values for are still not that accurate when replaying the whole simulation. In the beginning, before the number of inputs reach the sequence length, there is a large error due to the missing information as RNNs are trained to estimate based on a sequence. In addition a smoothening algorithm would also improve the quality of the estimation as it is relatively noisy. To further improve the estimator, lateral speed values can be added as an additional output however his will need more resources for training and a larger model to comprehend the new output.

3.2 Scope of further development

The method that was described above is just the first AI model that can be used. The further development will touch on longitudinal and lateral vehicle speed estimation with LSTM, GRU, Transformers and Temporal convolutional network. In addition, a wider range of test cases will be implemented for training purposes to cover more driving scenarios. To finish it off, some hybrid methods will also be expanded to see and compared with the direct AI estimators.

Bibliography

[1] *How to calculate wheel and vehicle speed from engine speed.* URL: https://x-engineer.org/calculate-wheel-vehicle-speed-engine-speed/ (visited on 05/05/2025).

[2] The Editors of Encyclopaedia Britannica. *speedometer.* URL: https://www.britannica.com/technology/speedometer (visited on 05/07/2025).

[3] Chris Woodford. *Speedometers.* URL: https://www.explainthatstuff.com/how-speedometer-works.html (visited on 05/05/2025).

[4] Otto Schulze. "Improvements in Speed Indicators". Pat. GB000190322622A (Germany). May 20, 1903.

[5] Joseph W Jones. "Speedometer". Pat. US765841A (United States). July 26, 1904.

[6] *Wheel Speed Sensors – More than just ABS.* URL: https://premierautotrade.com.au/news/wheel-speed-sensors%E2%80%93more-than-just-abs.php (visited on 05/07/2025).

[7] *What is GNSS.* URL: https://www.euspa.europa.eu/eu-space-programme/galileo/what-gnss (visited on 05/07/2025).

[8] *GPS satellites - trilateration.* URL: https://openclipart.org/detail/191659/gps-satellites-trilateration (visited on 05/08/2025).

[9] *Math frames.* URL: https://www.vectornav.com/resources/inertial-navigation-primer/math-fundamentals/math-refframes (visited on 05/08/2025).

[10] Chul Ki Song, Michael Uchanski, and J. Hedrick. "Vehicle Speed Estimation Using Accelerometer and Wheel Speed Measurements". In: July 2002. DOI: 10.4271/2002-01-2229.

[11] Lars Nielsen Uwe Kiencke. *Automotive Control Systems: For Engine, Driveline, and Vehicle.* Springer Berlin, Heidelberg, 2005.

[12] Giulio Reina, Matilde Paiano, and Jose-Luis Blanco-Claraco. "Vehicle parameter estimation using a model-based estimator". In: *Mechanical Systems and Signal Processing* 87 (2017). Signal Processing and Control challenges for Smart Vehicles, pp. 227–241. ISSN: 0888-3270. DOI: https://doi.org/10.1016/j.ymssp.2016.06.038. URL: https://www.sciencedirect.com/science/article/pii/S0888327016302205.

[13] Li Ji et al. "Evaluation of the performance of GNSS-based velocity estimation algorithms". In: *Satellite Navigation* (2022).

[14] John Chrosniak, Jingyun Ning, and Madhur Behl. "Deep Dynamics: Vehicle Dynamics Modeling With a Physics-Constrained Neural Network for Autonomous Racing". In: *IEEE Robotics and Automation Letters* 9.6 (June 2024), pp. 5292–5297. ISSN: 2377-3774. DOI: 10.1109/lra.2024.3388847. URL: http://dx.doi.org/10.1109/LRA.2024.3388847.

[15] Taekyung Kim, Hojin Lee, and Wonsuk Lee. *Physics Embedded Neural Network Vehicle Model and Applications in Risk-Aware Autonomous Driving Using Latent Features*. 2022. arXiv: 2207.07920 [cs.RO]. URL: https://arxiv.org/abs/2207.07920.

[16] Zhuangwei Shi. "Incorporating Transformer and LSTM to Kalman Filter with EM algorithm for state estimation". In: *CoRR* abs/2105.00250 (2021). arXiv: 2105.00250. URL: https://arxiv.org/abs/2105.00250.

List of Figures

List of Tables