Road friction estimation using an artificial neural network in a simulated environment

Jonas Karlsson

LUND UNIVERSITY

Department of Automatic Control

MSc Thesis
TFRT-6095
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Tryckeriet i E-huset
Lund 2020

Abstract

With the transition of responsibilities from the driver to the automated driving systems in vehicles, the systems need to have been tested for an extensive list of test scenarios as the passengers require high trustworthiness. The friction coefficient for the tyre-road friction is of high importance for the control of the vehicle but the coefficient is dependent on the physical complexity and nonlinear behaviour of tyres and is difficult to measure. Hence, testing is performed in controlled environments which limits the systems exposure to different testing scenarios.

The purpose of this thesis and the underlying work was to develop and evaluate a process for friction estimation using machine learning. The aim was to produce an estimation method using neural networks that are trained on data from a vehicle model implemented in a simulated environment using Unity 3D, which is a software platform for simulation and game development. The master thesis was produced at Combine Control Systems AB for Lund University in cooperation with National Electric Vehicle Sweden AB (NEVS).

Keywords: Friction estimation, vehicle dynamics, vehicle simulation, artificial neural networks

Acknowledgements

In before the start of this thesis I would like to take the time to thank those who helped this project reach its destination. First, I would like to express my gratitude towards my supervisor Simon Yngve at Combine for all help every step of the way. Additional thanks to Anton Holm at Combine for the guidance in the maze of machine learning. Special thanks to Yunlong Gao and Torbjörn Norlander at NEVS for the cooperation and the support throughout the project. I would also like to thank my supervisor Anton Cervin at the Department of Automatic Control at Lund University for the beneficial feedback during the project.

Lund, December 2019
Jonas Karlsson

Contents

List of Figures

Notation

Symbols

F_x	$\triangleq$	Longitudinal force
F_y	$\triangleq$	Lateral force
F_z	$\triangleq$	Normal force
F_{rr}	$\triangleq$	Rolling resistance force
F_{air}	$\triangleq$	Air resistance force
δ	$\triangleq$	Wheel angle from steering
M	$\triangleq$	Torque / Moment of force
I_{zz}	$\triangleq$	Moment of inertia
α_v	$\triangleq$	Angular acceleration
ω_v	$\triangleq$	Angular velocity
l_f	$\triangleq$	Distance from centre of gravity to front axle
l_r	$\triangleq$	Distance from centre of gravity to rear axle
T_w	$\triangleq$	Track width
a_x	$\triangleq$	Longitudinal acceleration
a_y	$\triangleq$	Lateral acceleration
v_x	$\triangleq$	Longitudinal velocity
v_y	$\triangleq$	Lateral velocity
φ	$\triangleq$	Road bank angle
Θ	$\triangleq$	Road inclination angle
CGH	$\triangleq$	Height to centre of gravity
m_s	$\triangleq$	Sprung mass
m_u	$\triangleq$	Unsprung mass
r_e	$\triangleq$	Effective radius
κ_x	$\triangleq$	Longitudinal slip
α	$\triangleq$	Lateral slip
μ	$\triangleq$	Friction coefficient/quotient

Acronyms

ANN	-	Artificial Neural Network
API	-	Application Programming Interface
CDP	-	Combined Data Points
DV	-	Digital Vehicle
MAE	-	Mean Absolute Error
MLP	-	Multilayer Perceptron
NEVS	-	National Electric Vehicle Sweden AB
NN	-	Neural Network
PV	-	Physical Vehicle
SDP	-	Single Data Points

1

Introduction

Active safety systems and advanced driver-assistance systems are becoming more prevalent in vehicles, and responsibilities are transferred from the driver to the control systems. With an increasing presence of algorithms, the importance of reliability in the underlying variables is increasing. To achieve trustworthy performance several issues need to be addressed including complexity, robustness, and uncertainty [**1**].

The introductory chapter will present background information to friction estimation, why it is important, and the impact that the driver's and vehicle's knowledge of external and internal factors have on the friction estimation problem. Difficulties related to the development and testing process are described, and background on the applicability of simulation and neural networks is given. Furthermore, the approach and purpose of this thesis is described, followed by the delimitations and the methodology.

1.1 Background

Problem formulation

Friction estimation As the contact patches of the vehicle's tyres are the only connections between the vehicle and the road, the interaction occurring at those points are of high importance to control the motion and stability of the vehicle [**2**]. Notably, tyre-road friction estimation has been found to be one of the most important factors in avoiding car crashes and is crucial to the performance of safety systems [**3**]. The physical complexity and nonlinear behaviour of the tyres induce difficulties in defining and measuring the friction coefficient [**2**; **4**], and has motivated a large number of papers and articles to improve friction estimation using a set of proposed methods.

Identifying the friction coefficient gives information about the maximum friction force that the tyre can generate on the surface of the road which affects the control and stability of the vehicle [**5**]. Several variables have been identified as having a crucial role in the identification of the friction force, however, some variables cannot be measured or are estimated due to issues of cost or difficulties in designing suitable sensors [**1**].

Driver and weather impact Factors that are indirectly related to the vehicle also influence the vehicle behaviour and two of the most prominent are the driver and the weather [**3**]. An investigation into 6.4 million vehicle crashes found that 24% of crashes occur in unfavourable weather conditions [**6**], and weather related issues have an impact on both crash frequency and severity [**3**]. It was found that drivers ignore the lower friction when driving on wet pavement instead of dry, as a statistically significant difference in speed could not be found between these scenarios. However, drivers did reduce the velocity because of limited sight [**7**]. Other research found that the frequency of crashes are substantially increased in snowy conditions but the severity is reduced as drivers recognise the increased crash risk and reduce the speed, but not enough for the crash frequency to be equivalent to dry driving conditions [**6**]. The increased occurrence of crashes during bad weather conditions has been linked to the friction effect on the vehicle's safety and stability systems, as well as the driver's assessment of the road condition [**3**]. The gathered data indicate that drivers and systems need to improve of their detection and handling in adverse scenarios.

Development and testing process While drivers are aware of the increased crash risk in adverse road conditions as well as in weather that is difficult to detect as having a negative impact on vehicle handling [**8**; **6**], their actions required are not sufficient to avoid an increased crash risk [**9**]. Safety systems, however, perform well in crash avoidance actions, but have issues with identifying and estimating the required variables, notably the friction coefficient [**1**]. This suggests a potential loss of information when the transportation industry is moving towards autonomous vehicles, thus increasing the impact and importance of estimation methods. With the transformation of a modern vehicle from a sensor platform that passes information to the driver to take intelligent action, to a fully autonomous platform, manual intervention can no longer be relied upon, further strengthening the case for robustness [**10**].

For the case of autonomous vehicles a high level of confidence is even more important as more problems exist in the testing process. Five major issues have been identified in research of the expanded testing and validation process for autonomous vehicles, including driver out of the loop and stochastic systems. Where a vehicle with a human driver requires the ability to identify a critical malfunction and then can expect the driver to take control and solve the issue, an autonomous vehicle is required to solve both the initial issue and the resulting problem hence increasing the number of issues that require testing [**12**].

Systems including non-deterministic algorithms are troublesome in connection to structured testing processes as the same scenario may produce different results every time due to the random nature of non-deterministic systems [**12**].

To achieve a higher confidence in the performance, the vehicle's stability and safety systems need to be evaluated for a large number of test scenarios. Comparing to fatality rates from the United States Department of Transportation (NHTSA), fatalities per mile travelled in 2017 amount to $1.16 \cdot 10^{-8}$ [**11**]. For the testing of a complete vehicle system, this amounts to 116 million miles travelled to validate the failure rate for a single incident on a fatal severity level. It is likely that the tests need to be longer, and be repeated several times over to attain statistical significance [**12**]. This can be reduced using accepted testing methods for the whole vehicle systems and individual sub-systems, but this assumes that the testing methods are highly representative of the real-world scenarios.

Testing for real-world scenarios has two major issues. In general, gathering data implies the collection of information from the occurrence of an event meaning that the event needs to happen. For situations with a low probability, the outcome may not be analysed as it might not have happened.

Secondly, testing for a large set of scenarios is infeasible as the true values of the variables need to be known in order to fully evaluate the system. For an environment where any variable is evaluated against another estimate, the testing is prone to errors and thus reduces the available testing sites to controlled scenarios where the friction is known, which limits the robustness of the results [**13**].

Motivation

Simulation In order to overcome some of the issues related to real-world testing, simulation has been proposed as a solution. For the first sub-problem, scenarios with a low probability of occurring can be targeted by having full control of internal and external factors to ensure that the situation has been evaluated. Furthermore, dangerous situations can safely be tested in a digital environment as the physical risk to the driver and other involved actors is removed [**13**].

For the second problem, using a digitally constructed environment, the estimation of a variable is compared to the true value of the variable in order to perform a valid verification [**13**]. In comparison, for real-word testing, when an estimation is compared to a different method of estimation, the result can achieve high precision but is not guaranteed to have a high accuracy as the true value is not known.

In general, the main advantages of simulation have been described as allowing controllability, reproducible results, and standardization as the environment can be manipulated to fit the aim of the research. Full control of the scenarios may be used to reduce the effect of randomness to produce the same results [**13**].

Neural networks Due to the complexity of friction estimation, particularly related to modelling of tyre dynamics, it is desirable to find a method that enables results on the same level as the tyre model presented in Section 2.4 while avoiding too intricate modelling. The purpose of the models in the case of friction estimation is to find a connection between a number of input variables and the resulting friction coefficient [**14**].

In the area of artificial neural networks, the "Universal approximation theorem" states that a multi-layered neural network with one hidden layer can arbitrary well approximate a continuous function of n variables with certain assumptions on the contained mathematics. A proof for this can be found using Debao Chen's theorem [**15**]. However, it is preferred to use a larger amount of hidden layers as this will add more activation functions, thus making the network less linear and reducing the risk of overfitting.

This motivates the attempt to use neural networks as an effort to move the problem from identifying a complex physically derived connection between the input and output variables, to a purely mathematical connection.

Related work

Several publications exist that use different machine learning methods to gather information about the interaction between the tyre and road.

Nolte *et al.* use two different deep convolutional neural network models to classify images from existing data sets of road surfaces, to provide additional information to existing friction estimation algorithms [**16**]. Google search had to be used to gather enough data for the training, indicating that a method to collect data would be beneficial to the algorithm. The network does not estimate the friction coefficient, meaning that it can not replace the network in this thesis, but can act as a complement to it.

Panahandeh *et al.* use and compare thee machine learning models to classify the road surface as slippery or non-slippery based on data from a fleet of connected vehicles and weather data from a national meteorological institute [**17**]. The estimation algorithm is evaluated against previously estimated values, implying that a true comparison is never made. However, as a binary classification is used, the error from not using true values is reduced.

Ribeiro *et al.* use a time delay neural network to estimate the friction coefficient from the estimation of the lateral tyre force and normal force using a Kalman filter [**18**]. The neural network is trained using a Matlab model and performs very well from a theoretical standpoint, but the performance on data from a real vehicle is not investigated.

All publications above focus on either simulation or real-world data. The machine learning models using the simulation approach achieve impressive results, but are never validated in a real-world scenario. The methods focusing on real-world data have trouble achieving a high accuracy, or estimates beneficial information to existing friction estimation algorithms. All alternatives have issues that could be circumvented using a combined of simulation and real-world testing, which is investigated in this thesis.

1.2 Purpose

The purpose of the underlying work to this thesis was to develop and investigate a process to estimate tyre-road friction using an artificial neural network trained in a simulated environment. The aim of the study was to verify if the artificial neural network can be used to improve the performance of the vehicle for unknown tyre-road friction.

The implementation presented in this thesis is only one way to apply the process, and the chosen technologies were decided based on preferences of involved actors. Hence, the purpose was not to evaluate the used software but to:

1. Develop a completely digital process for friction estimation
2. Evaluate the performance of the process.

1.3 Methodology

The process that is evaluated in this thesis can be divided into three major parts: vehicle dynamics, simulation, and the artificial neural networks.

Vehicle dynamics is the basis for friction estimation, and several vehicle- and tyre models have been developed to represent different physical properties of their counterpart. The initial work for the study was to decide which types of models that best fit the purpose and were going to be used. The chosen models were then to be implemented in the simulation environment and validated against real-world data logs provided by NEVS to confirm actuality.

A simulation environment was developed using the Unity 3D platform as Unity was predetermined to be used for building the simulation software. The purpose of the simulations was to generate data that was used to train, test and partially evaluate the neural networks. The requirements for the simulation software were set by the inputs and outputs of the system where the inputs are the parameters and scenarios that require testing, and the outputs are the generated data and results for evaluation. The inputs are based on requirements from theory of artificial neural networks and information from NEVS.

In order to determine accuracy and evaluate the networks, different architectures and designs of neural networks were used to estimate the friction coefficient for the validation scenarios. Both generated data and logged data were used to evaluate the networks in order to compare the performance of different setups.

1.4 Delimitations

The focus of the underlying work and thesis was to provide an understanding of and evaluate a process to estimate tyre-road friction using an artificial neural network. Certain parts of a complete vehicle system were not implemented in the simulations as they were deemed either out of scope for the purpose of the study or to have insignificant impact on the results. It was not the aim to develop a completely optimised ANN nor to implement a perfect digital twin of the vehicle, thus, neglecting some components were deemed as reasonable. One example would be the effects of air resistance on the vehicle, which however could be applied in a later effort for improved results.

1.5 Outline

In Chapter 2 the choice of vehicle model is motivated and the mechanics of the system is briefly explained. Then, the tyre model is described and equations are provided with focus on slip, forces, and friction. This gives an overview of the dynamics that were run in the simulations.

Chapter 3 provides an explanation to Unity 3D, and its role for simulation. Moreover, the final implemented structure of the vehicle model and tyre model is presented. Finally, the simulation process from input configuration and test scenarios to logged data is described.

Chapter 4 introduces the method of validation and presents the results from comparing the implemented simulation model to logged data received from NEVS.

Chapter 5 provides brief theory on neural networks and the requirements it sets on training. Furthermore, the chapter presents the types, designs, and architectures of the neural networks that were chosen for the evaluation. Finally, the two different types of outputs from the neural networks are covered.

Chapter 6 presents the results from the two approaches for the process evaluation and a sensitivity analysis on the process. Short discussions are held about the obtained results.

Chapter 7 concludes the underlying work and the thesis with comments about the results, and the strengths and weaknesses of the process.

2

Vehicle dynamics

The goal of modelling the kinematics and dynamics of a vehicle is to obtain a mathematical description of the functionality and characteristics of the vehicle. The model can then be applied in a variety of areas and used with an array of different analytical tools. Each area and analytical tool has distinct traits and focus areas, implying that the area of use decides which model to adopt and its limitations for the application [**19**]. Thus, focus should be on the areas of interest that align with the purpose of the thesis.

For a complete and accurate simulation of a vehicle, the number of parameters and states would be overly extensive and the relationships too complex for a functional model to be obtained [**14**]. The validation results of such a model would most likely prove to have a high accuracy, but have a negative impact on computational requirements and development costs in relation to what is appropriate for the expected results. Hence, it is beneficial to find a balance between accuracy and used resources, and a simpler model that still fulfil the requirements is preferred.

2.1 Vehicle model

As the thesis focus on the interaction between the wheel and the road, certain parts of the vehicle model is of little significance to the results. Thus, components with less influence on wheel behaviour received less attention and were not modelled, whereas components with significant impact were modelled based on previous works in the area of vehicle dynamics [**1**; **20**; **14**]. For this reason, the contact between the vehicle body and the wheels is considered to be stiff, which means that suspension is disregarded.

2.2 Single-track model

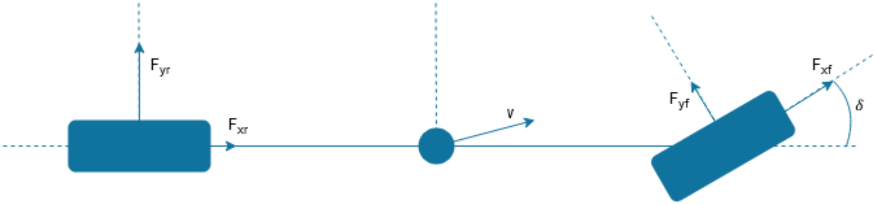


Figure 2.1 The single track vehicle model.

A common and simple option for modelling a vehicle is the linear single track model, also called the *bicycle model*. It allows for approximate modelling, but is based on a collection of simplifications, including [**19**]:

- The merging of the wheels on each axle translating the contact patches of the wheels to the longitudinal trajectory line.

- Distribution of forces in the lateral direction is constant.

The model has several simplifications that reduce the controllability of the testing to an extent where it limits the information available from the simulations.

The first point removes the possibility for individual control of all four tyres [**19**], therefore limiting the available data from the wheels to what is deemed as a too imprecise level.

The second point has a definite impact on the friction coefficient. As the denominator of the friction quotient, which is described later, is the normal force, not including the changes to the normal forces due to the transfer of load in the lateral direction would set a constant friction maximum and limit the available friction force. In total, the simplifications reduce the level of detail of the model too much for the purpose of the thesis.

2.3 Two-track model

A model which has more degrees of freedom is the two-track model. The major difference is the involvement of all four tyres. This allows the incorporation of all individual tyre forces, hence, increased accuracy in modelling roll dynamics and the friction coefficient for every wheel [**1**].

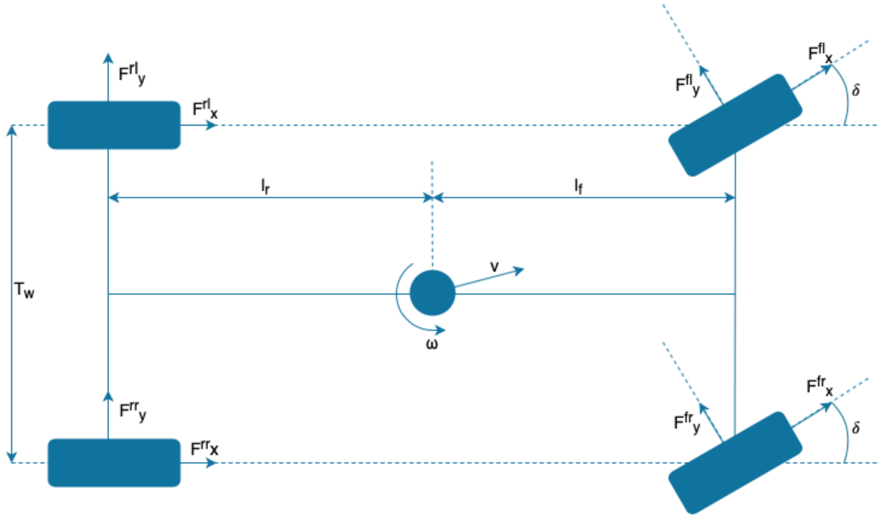


Figure 2.2 A two-track vehicle model

Vehicle body model

For the following equations the superscripts indicate the selection, i.e. f for front, and rl for rear-left, and the subscripts indicate the direction. From Figure 2.2 the generalised forces can be described from the individual tyre forces:

$$F_x^f = (F_x^{fl} + F_x^{fr})cos(\delta) - (F_y^{fl} + F_y^{fr})sin(\delta) \tag{2.1}$$

$$F_x^r = F_x^{rl} + F_x^{rr} \tag{2.2}$$

$$F_y^f = (F_y^{fl} + F_y^{fr})cos(\delta) + (F_x^{fl} + F_x^{fr})sin(\delta) \tag{2.3}$$

$$F_y^r = F_y^{rl} + F_y^{rr} \tag{2.4}$$

which is the summation of all forces for the front and rear axles in the x and y direction, where δ is the steered wheel angle, as in the difference in angle between the vehicle's longitudinal trajectory line and the wheel's longitudinal trajectory line. Furthermore, the vehicle torque can be derived:

$$\begin{aligned} M = l_f \cdot (&sin(\delta)(F_x^{fl} + F_x^{fr}) + cos(\delta)(F_y^{fl} + F_y^{fr})) + \\ &\tfrac{T_w}{2} \cdot (sin(\delta)(F_y^{fl} - F_y^{fr}) + cos(\delta)(F_x^{fr} - F_x^{fl})) + \\ &\tfrac{T_w}{2} \cdot (F_x^{rr} - F_x^{rl}) - \\ &l_r \cdot (F_y^{rr} + F_y^{rl}) \end{aligned} \tag{2.5}$$

where l_f and l_r is the distance to the front and rear axle from the centre of gravity respectively, and T_w is the track width, as in the distance between the contact points of the wheels in the lateral direction. The translational and angular motion can then be described by:

$$F_x = F_x^f + F_x^r - F_x^{air} - mg \cdot sin(\Theta) - \sum F_{rr} = m(\alpha_x - v_y \cdot \omega_v) \tag{2.6}$$

$$F_y = F_y^f + F_y^r - F_y^{air} - mg \cdot sin(\varphi) = m(\alpha_y + v_x \cdot \omega_v) \tag{2.7}$$

$$M = I_{zz} \cdot \alpha_v \tag{2.8}$$

where a_x and a_y is the acceleration in the longitudinal and lateral direction respectively, α_v the angular acceleration for the vehicle, F_{rr} the rolling resistance force for each wheel, φ the road bank angle, Θ the road inclination angle, and ω_v the angular velocity around the centre of rotation for the vehicle. Note that for a more accurate representation the aerodynamic drag force F^{air} needs to be included, but has been neglected and is set to zero in this thesis.

Vehicle load transfer

When the wheel is in contact with the road and generates longitudinal and lateral forces, the normal force that acts on the tyre limits the maximum force that can be generated [**21**]. When no acceleration takes place in either direction, the normal force on each tyre is constant and can be found using simple rigid body mechanics. However, when the vehicle is in motion the distribution of normal forces changes [**1**]. An example is braking, which transfers load to the front wheels of the vehicle [**14**]. To be able to capture the full range of the forces, the model needs to incorporate the dynamics of load transfer.

Longitudinal load transfer

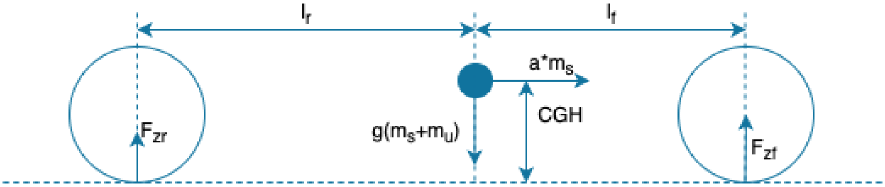


Figure 2.3 Longitudinal load transfer model

In the longitudinal direction, the normal force for the front and rear axle can be found using the equation of force in the z-direction and equation of torque.

$$F_z^{rear} = \frac{g \cdot (m_s + m_u) \cdot l_f - m_s \cdot a_x \cdot CGH}{l_f + l_r} \tag{2.9}$$

$$F_z^{front} = \frac{g \cdot (m_s + m_u) \cdot l_r + m_s \cdot a_x \cdot CGH}{l_f + l_r} \tag{2.10}$$

Lateral load transfer

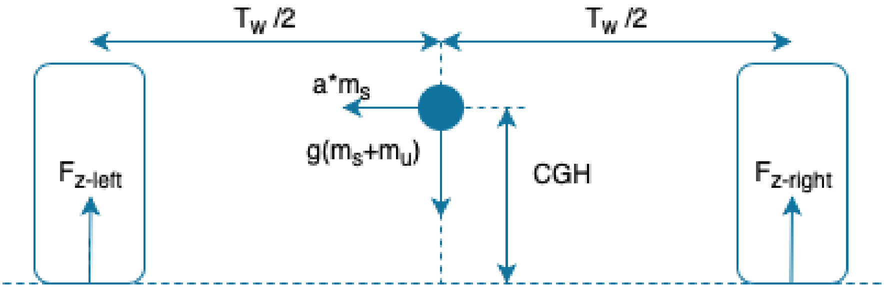


Figure 2.4 Lateral load transfer model

Using the same equations of force and torque applied in the lateral direction of the vehicle, the load transfer equations can be formulated as:

$$F_z^{left} = \frac{g \cdot (m_s + m_u)}{2} + \frac{m_s \cdot a_y \cdot CGH}{T_w} \tag{2.11}$$

$$F_z^{right} = \frac{g \cdot (m_s + m_u)}{2} - \frac{m_s \cdot a_y \cdot CGH}{T_w} \tag{2.12}$$

Combined load transfer To apply the longitudinal and lateral load transfer simultaneously, each equation from equation 2.9 to equation 2.12 can be divided by the sum of the total weight to obtain the percentage assigned to the relative section. The normal force for each wheel can then be found by multiplying the associated percentages with the weight. For the front-left wheel, the following is applied:

$$F_z^{front-left} = \text{percentageFront} \cdot \text{percentageLeft} \cdot \text{weight} \tag{2.13}$$

Equation 2.13 can be applied for every wheel using the correct percentages.

2.4 Tyre model

The interaction between the tyre and road is highly important in order to model the behaviour of a vehicle[**14**]. To obtain the friction force F, one simply needs to solve:

$$\mu = \frac{F}{F_z} \tag{2.14}$$

However, μ can be found difficult to compute as the friction coefficient μ is required to compute the force F, and the force is required for the quotient used to describe μ. However, the complexity depends on the chosen approach. If one neglects the deformation of the tyre, a simple but imprecise solution can be found [**14**]. The use of a more accurate approach requires the inclusion of the tyre deformation [**4**] and the problem becomes complex and can motive the number of papers on the subject.

The appropriateness of a tyre model is decided by the type of research that is to be carried out, hence, no universal model exists. As the result of the thesis is partially based on measurements from a physical vehicle, it is important that the digital and the physical version use the same model for them to be comparable. For the implementation of the tyre model, data was received from NEVS that describe the relationship between slip, normal force, and friction force for the tyres that were used during the collection of the received logged data. For reasons of confidentiality, neither tyre model data nor logged data will be presented in this thesis.

Slip

For the tyres to generate longitudinal and lateral forces from the contact with the road, the occurrence of slip is required. Slip is the difference in velocity for the wheel and the road, and exist in both the longitudinal and lateral direction [**1**]. Slip is a common choice of parameter to include in the modelling of a tyre, and can in combination with the normal force be considered to be one of the two variables with most impact on the friction quotient μ [**14**].

Longitudinal slip When a driving or braking torque is applied to a wheel and the wheel is accelerated in either direction, a momentary difference in velocity is introduced. The difference in velocity is the longitudinal slip and is described as the difference between the circumferential velocity of the tyre and the velocity of the vehicle at the point of the wheel, commonly known as the speed of the road [**4**]. The circumferential velocity is given by:

$$v_c = \omega \cdot r_e \tag{2.15}$$

where ω is the angular velocity of the wheel, and r_e is the effective radius of the tyre. The effective radius is defined as the ratio v_x/ω_0 where v_x is the road velocity, ω_0 is the angular velocity of a free-rolling wheel. The usage of effective radius is required due to the deformation of the tyre that arises when the wheel comes into contact with the ground, especially notable with a large vertical load [**4**]. The longitudinal component of slip can then be defined as:

$$\kappa_x = -\frac{v_x - v_c}{v_x} \tag{2.16}$$

where $v_x = v_c$ indicates no slip, and $v_c = 0$ implies that the wheel is completely locked [**14**]. A positive slip for this definition means that the vehicle is accelerating.

Lateral slip

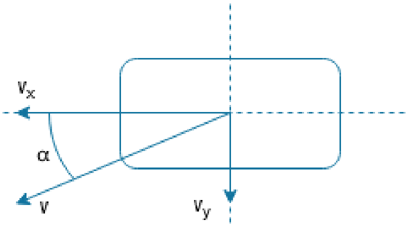


Figure 2.5 Slip angle for tyre

When the front wheels are turned, the heading of the wheel and the wheel travel velocity differ in direction. This gives rise to a velocity in the lateral direction that is used in combination with the longitudinal velocity to describe the lateral slip using:

$$tan(\alpha) = \frac{v_y}{v_x} \tag{2.17}$$

where α is the slip angle, and is used to define lateral slip [**4**].

Forces

A press on the acceleration pedal will produce a driving torque from the engine. The torque is transmitted throughout the components of the propulsion system experiencing loss, and a change in magnitude by a ratio decided by the design of the transmission, and finally reaches the driven wheels [**21**].

The torque generated from braking is instead acting directly on the wheel due to the mechanical design and placement of the brake system. The braking torque will produce a force opposite to the motion and reduce the angular velocity by decelerating, thus creating a slip in the opposite direction of the direction of rotation [**21**].

Longitudinal forces The torque applied to the wheels from the engine while accelerating, τ_{eng}, is reduced by the dissipation of power due to the deformation of the tyre, also known as rolling resistance. The rolling resistance is assumed to depend linearly on the vertical load and a factor f_{roll} is generally introduced to represent this effect. The propulsion force on the wheel can then be defined as:

$$F_x^{prop} = \frac{\tau_{eng} \cdot ratio}{radius} - f_{roll} \cdot F_z \tag{2.18}$$

However, the propulsion force is limited by the maximum available friction force which is given by $\mu \cdot F_z$. The traction force in the longitudinal direction when accelerating can thus be described by [**21**]:

$$F_x^{wheel} = min(F_x^{prop}, \mu \cdot F_z) \tag{2.19}$$

Lateral forces Lateral force is generated from wheels due to the existence of either slip or camber. The camber angle is the angle of the wheel around the longitudinal axis and is a major part of lateral forces when modelling motorcycles, but the effect for cars is much smaller in relation to the force generated by lateral slip [**4**]. Thus, the effect of camber is ignored in the underlying work and the thesis.

Friction

In the situation of both cornering and deceleration or acceleration, the tyres will experience both longitudinal and lateral slip. When the events occur simultaneously it is known as *combined slip*. This differs from the case where only one type of slip is occurring, called *pure slip*[**1**]. As the magnitude of the traction forces cannot exceed the available friction force, occurrence of combined slip will affect the maximum value of the longitudinal and lateral traction force. Combined slip is more complex than the case of *pure slip*, and a semi-empirical approach can be chosen as a solution where empirical data is transformed using a physical model to be applicable for other situations [**4**].

The relation between longitudinal and lateral force during combined slip can be described using the friction ellipse, where the assumption is that the resultant force is limited and lies in the ellipse described by:

$$\left(\frac{F_x}{F_{x,max}}\right)^2 + \left(\frac{F_y}{F_{y,max}}\right)^2 \leq 1 \tag{2.20}$$

where $F_{x,max}$ is given by $\mu \cdot F_z$ and $F_{y,max}$ can be found using the magic formula [**1**], described below.

Magic Formula A commonly used model to compute tyre forces in vehicle dynamics is the *Magic Formula*. The semi-empirical model has been developed over a long period of time by several partners and has produced a general formula that is defined by [**21**]:

$$y = D\sin\left[C \cdot \arctan\left\{Bx - E(Bx - \arctan(Bx))\right\}\right] \tag{2.21}$$

$$Y(X) = y(x) + S_V \tag{2.22}$$

$$x = X + S_H \tag{2.23}$$

where the uppercase variables *B, C, D, E* are factors that alter the shape of the magic formula curve and are named as follows:

B	stiffness factor
C	shape factor
D	peak value
E	curvature factor

and S_H and S_V are shifts of the curve in the horizontal and vertical direction respectively, which are used to match the empirical tests better. Y is the output force, and X is the input variable $tan(\alpha)$ for the lateral curve or κ_x for the longitudinal curve.

The curve applied in the longitudinal and lateral direction can be seen in the figures below.

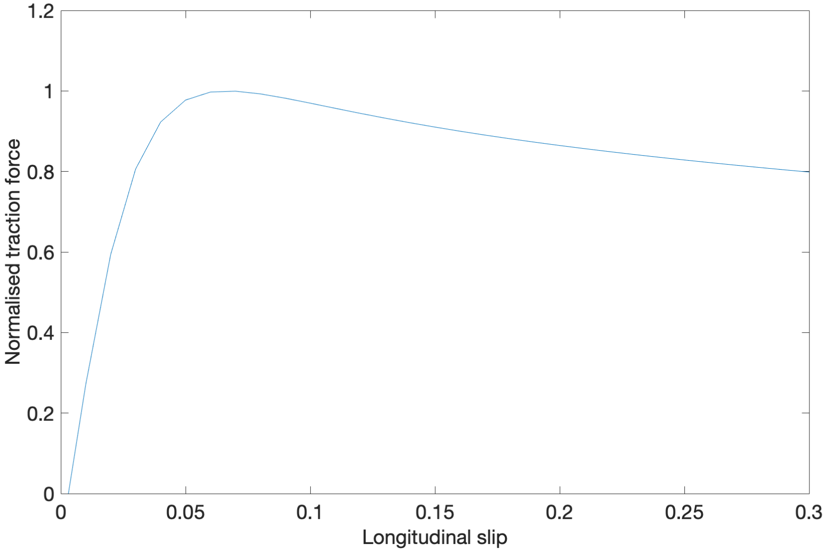


Figure 2.6 An example of a magic formula curve for longitudinal traction force. Note that this does not represent the curve used in the thesis.

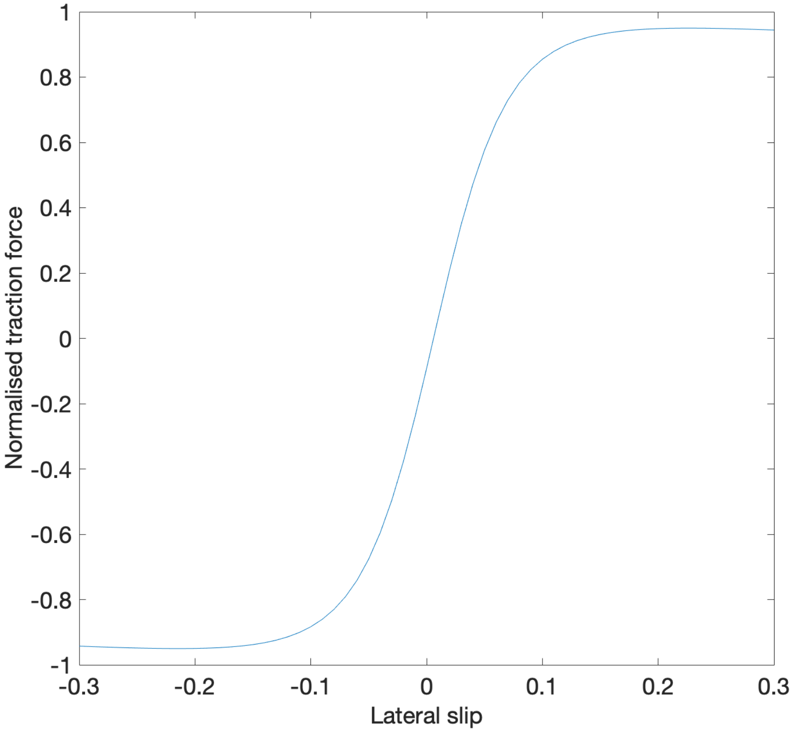


Figure 2.7 An example of a magic formula curve for lateral traction force. Note that this does not represent the curve used in the thesis.

3

Vehicle simulation

The ability to evaluate during the development stage is essential to understanding the performance of the product. For full evaluation it is important that tests are performed systematically under controlled conditions, which is especially important if repeatability is required. As the real-world environment is a large non-deterministic system, the use for controlled environments is understandable given the pre-conditions.

Simulation is a powerful solution to study variables of interest in scenarios that are repeatable and controlled to perform a scientific evaluation of a process. The goal of vehicle simulation is to achieve the functionality of the real-world vehicle. The concept is called fidelity and is defined as "faithfulness to the original" [**23**]. In order for the simulation to be applicable and provide value for the testing process, the models need to have a good-enough level of fidelity.

The effort to shorten product cycles and reduce the development time of new vehicle models pushes the automotive market to introduce new technologies that allow for faster testing and evaluation. This can be achieved through the implementation of simulation and virtual prototypes, which removes the need for a physical prototype to exist in certain areas of interest. However, the methods used on the virtual prototypes must be proven and trusted for the process to be valid. Another reasoning for the use of simulation is the price competition for vehicles where manufacturers and suppliers are forced to not only shorten development time but also cut costs in the development phase to stay competitive [**19**].

As mentioned in the introduction, testing can become a limitation as the number of tests can become extremely large. The reasoning for this is that options for testing scenarios are combinatorial in nature and thus the sum of scenarios greatly increases with the addition of a single option [**24**].

3.1 Unity 3D

Unity3D is a real-time development platform initially created as a game engine but has since included software to support realistic simulation and has been adopted by several industries including engineering and automotive. The platform uses the .NET platform and scripting is done using a C# API. To achieve high physics fidelity the platform relies on *Nvidia PhysX* [**25**] for physics computations.

A Unity program is based on a collection of objects called *GameObject*. The GameObjects work as containers for a set of components. Each GameObject has a *transform* component that manages the location, rotation, and physical scale of the object, and uses the *PhysX* engine to compute the new position and rotation during each update from the physics engine. Various physics components exist that handle different parts of the physics and have several changeable physical properties related to their task, including:

- Rigidbody - controls the kinematic and dynamic properties including mass, drag, and inertia.
- Mesh - component which controls the material, and the mesh, which sets the boundaries for the object and is used to compute collisions.
- Joint - used to connect and control how GameObjects behave in relation to each other and limit the degrees of freedom.

An example execution for a single refresh:

1. A force with a specific magnitude is applied on a point on a GameObject.
2. The rigidbody of the GameObject, the rigidbodies of the GameObject's children objects, and the GameObjects that are connected by joints, are collected to see which objects that are affected by the applied force.
3. Every affected objects then check for collision with other objects using the designated meshes used for each GameObject.
4. If a collision has taken place, the materials and other physical properties of the objects are used to compute the effects of the collisions.
5. The final translational and rotational motion is applied for every involved GameObject.

The platform allows for controlled testing as results for identical tests are reproducible unless a non-deterministic algorithm has been introduced to the simulation software. The software can be built for different target platforms, and can be built for both visual feedback and running on a server for optimised performance [**26**].

Furthermore the time scale and refresh rate for the physics engine can be adapted to fit the complexity of the simulation and the hardware that the simulation is running on, allowing for faster than real-time testing.

The option of running the simulations on servers is favourable for software testing because of the potentially large input space as multiple instances of the simulation software can be run on multiple servers, limiting the testing time to the computational power available and the cost of computations.

3.2 Model structure

The following sections present how the vehicle and tyre models were implemented in the simulation software in relation to the unity components and scripts. The implementation was made for the thesis and is based on the theory presented in Chapter 2 on vehicle dynamics, and focus on creating a program that supports large scale simulation usable for achieving the advantages discussed in the introductory chapter and the previous segments of the vehicle simulation chapter. The implemented model in Unity 3D can be seen in Figure 3.1 below.

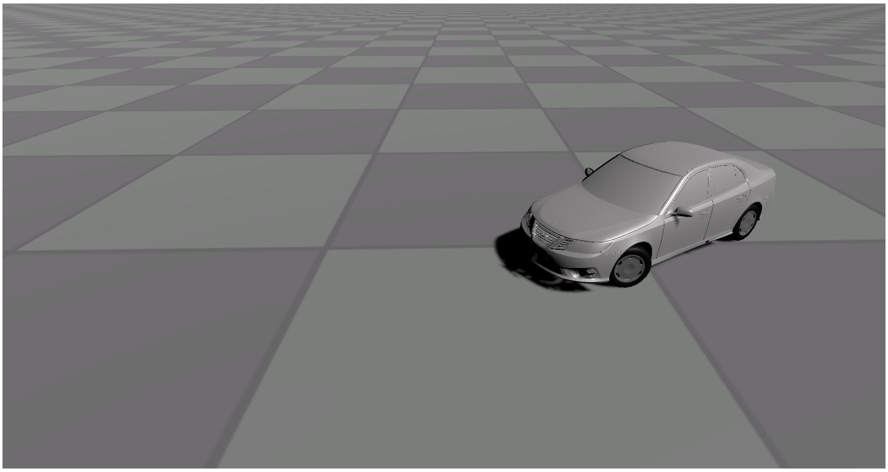


Figure 3.1 The vehicle model in Unity 3D.

Vehicle model structure

Figure 3.2 shows the components of the vehicle that were implemented in Unity 3D using theory on vehicle dynamics. The diagram excludes parts that are not affected by vehicle dynamics such as the log system and testing system, as well as the details of the wheel components which will be covered in the next section.

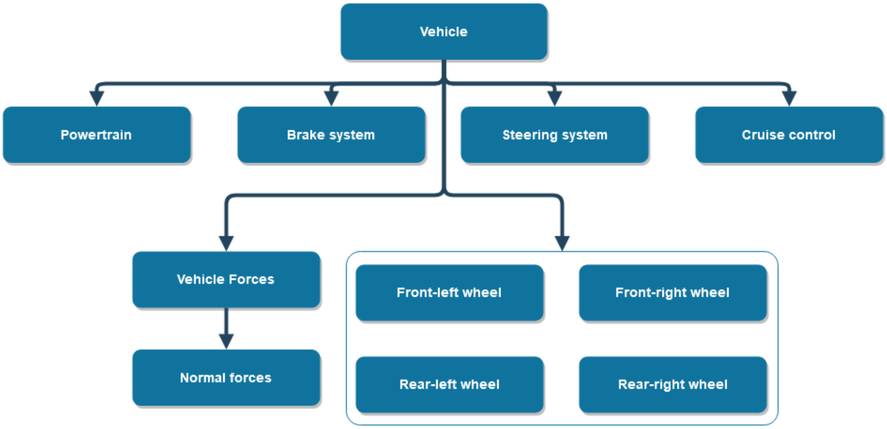


Figure 3.2 A diagram of the parts of the vehicle implemented in Unity 3D.

Vehicle The vehicle object is the container object of the vehicle system, and is implemented as a two-track model described in Chapter 2.3. However, since all wheels are assumed to be in contact with the road, vertical motion, roll, and pitch is set as constant for the vehicle relative to the road.

The vehicle body is implemented as a rigid body and contains both the sprung and unsprung mass in the manually set centre of gravity. The object contains information supplied by NEVS such as masses, moment of inertia, and centre of gravity to increase the fidelity of the model.

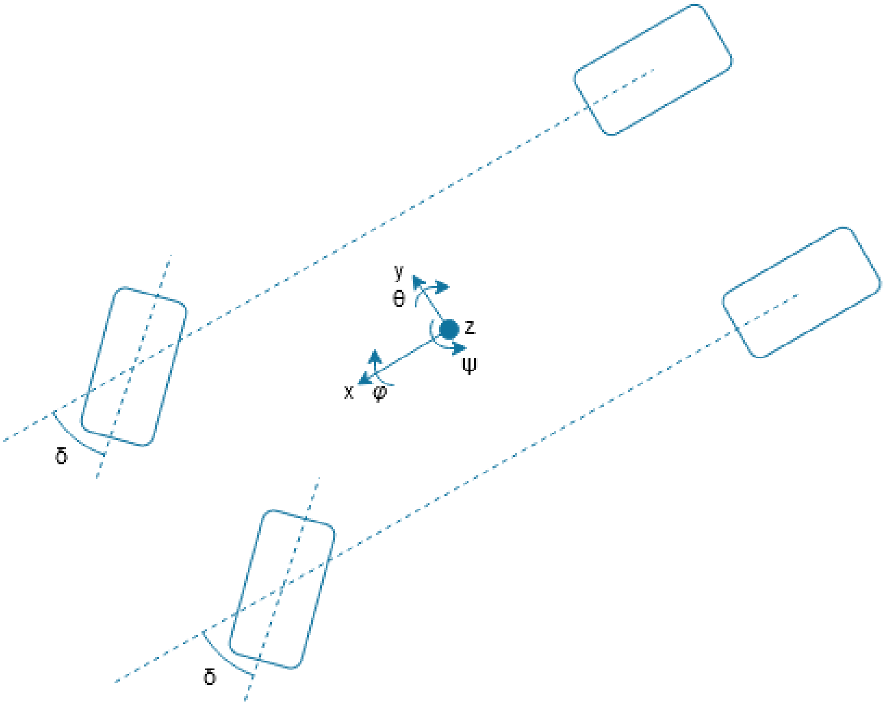


Figure 3.3 The vehicle object model for Unity 3D.

Powertrain The powertrain component is only implemented by scripts and clamps the maximum and minimum torque that can be produced by the electrical engine. The engine torque is directly applied to the driven wheels using an open differential, thus, the produced torque is split evenly between the driven wheels. Losses in the transmission is applied to match the maximum torque for the engine and wheels that were provided by NEVS.

Brake system The braking system is implemented by applying the same level of braking torque to all wheels, where the maximum braking torque was given by NEVS. In reality, the distribution of brake torque is often asymmetric but the difference is ignored for the digital vehicle.

Steering system The rotation of the wheels around the vertical axis is given by an input to the steering system by a turning of the steering wheel. The input angle is converted using a function provided by NEVS and then applied to the steered wheels, thus, the inner and outer wheel angles are assumed to be the same.

Cruise control Cruise control is implemented using a programmed script for scenarios where tests are being performed in a pre-determined velocity. The cruise control is designed using a PI-controller with back-calculation for anti-windup, and the output saturates between maximum engine torque and maximum braking torque as the negative limit.

Vehicle forces Forces in the vertical direction, i.e. normal forces, are computed as presented in Section 2.3 and assigned to each individual wheel, whereas forces in the longitudinal and lateral direction are computed in the tyre model.

Tyre model structure

Figure 3.4 presents the components of the tyre model that were implemented in Unity 3D. Every wheel is identical in the aspect of components but have different constraints based on their location, as a wheel placed on the rear axle is not rotated by the steering wheel and cannot rotate around the vertical axis.

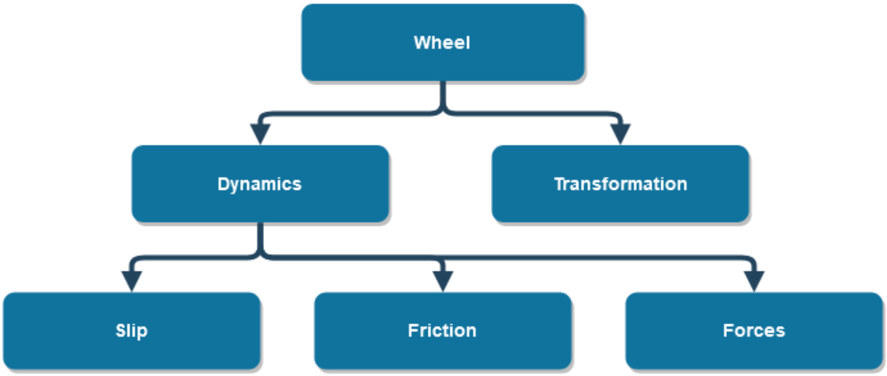


Figure 3.4 A diagram of the components of the tyre model implemented in Unity 3D.

Wheel The wheels are attached to the vehicle in their respective locations using joints which are identical except for the front axle joints which can rotate around the vertical axis due to steering.

The wheels are implemented as rigid bodies, which is inaccurate as the deformation of the tyres is a major factor in their dynamic behaviour. However, the problem is circumvented as the slip and friction components are implemented using data from NEVS produced from real tyres, thus, the deformation behaviour is implemented on a computational level, but not on a visual one.

Transformation The transformation component manages rotational motion around the vertical and horizontal axes, which is used to compute the slip in the longitudinal and lateral direction.

Slip The slip for each wheel is computed as presented in Section 2.4. The angular velocity for wheels that are not driven is the velocity in the longitudinal direction divided by the wheel radius. For driven wheels, the torque that is applied from either the engine or the brake system is converted to angular acceleration using:

$$\tau = I \cdot \alpha \tag{3.1}$$

where I is the moment of inertia, and α the angular acceleration. The resultant acceleration or deceleration is the source of the slip that arises due to the change of input.

Friction The friction in the longitudinal and lateral direction are computed using the magic formula described in Section 2.4. The factors for the curves in the different directions are chosen using data that was received from NEVS, and the actual values will not be presented due to confidentiality.

The curves that are generated in the simulations are based on the road surface that was chosen for the test scenario and new curves are generated using interpolation with the received data as a reference.

Tyre forces Using a powertrain that allows a high level of torque to be transmitted to the wheels or when driving on a road with adverse friction conditions, there is a limit to how much traction the vehicle can experience. The forces that can be generated are limited on the upper side by the friction force $\mu \cdot F_z$, and on the lower side by the resistance forces, including resistance from road slope angle, rolling resistance, and air resistance[**21**]. See Figure 3.5.

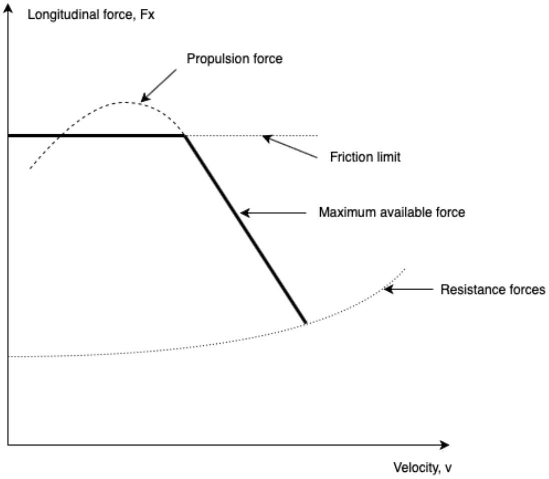


Figure 3.5 A traction diagram including rolling and air resistance.

3.3 Simulation process

Figure 3.6 represents the structure and the purpose at each stage of the simulation process, and the process is composed of three major parts, input to the simulations, running the simulations, and output from the simulations. These topics will be covered in the following sections, and the data flow for the simulation process can be seen in Figure 3.7.

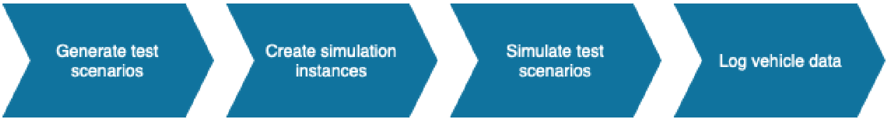


Figure 3.6 Overview of the simulation process.

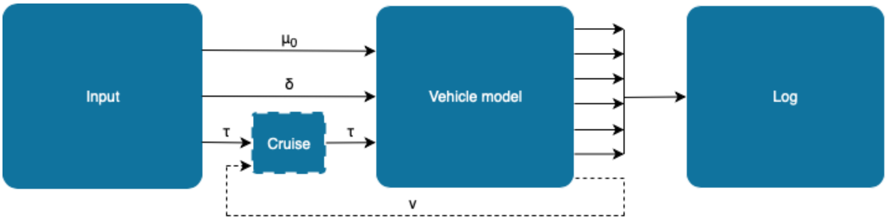


Figure 3.7 Data flow of the simulation process where μ_0 is the reference friction for the ground, δ is the steering wheel angle, v is the vehicle velocity and τ is the torque. Note that τ can be both positive (engine torque) or negative (braking torque).

Input configuration

In order to train a neural network to be applicable to a large collection of possible vehicle scenarios, big sets of data are required to achieve a sufficient accuracy. Thus, it would be irrational to manually create every possible combination of scenarios and an automatic combinatorial input process has been implemented. The inputs to the vehicle are designed to match the inputs controllable by a driver, meaning that the inputs to the vehicle model can represent turning of the steering wheel, and pressing on the gas or brake pedal, or using cruise control. Furthermore the input configuration allows each scenario to manipulate the environment in the form of selecting the road condition by changing the reference friction, which is the peak friction coefficient.

The input scenarios are based on two types of vehicle dynamics: straight line acceleration, and sinusoidal turning. Both scenarios contain a set of parameters which are required to completely define the test scenario. To cover a large collection of scenarios, each parameter is defined by a start value, an increment value, and an end value. This produces a range of possible values for a single parameter, and the ranges of values for every parameter for the scenario are combined to produce a large set of unique test scenarios.

Straight line acceleration is defined by three parameters: reference friction μ_0, test duration, and engine torque τ_{eng}. The test is performed from a stand-still and the vehicle will accelerate for the duration of the test applying a constant torque during the whole test. A simple example of straight line input configuration which yields 60 input scenarios can be seen in table 3.1 below.

Table 3.1 Example of straight line acceleration configuration

Parameter	Start	Increment	End
Reference friction	0.1	0.1	1.0
Test duration	20	0	20
Engine torque	50	50	300

Sinusoidal turning is performed at a selected velocity which the car will accelerate to using the cruise control. When the target velocity is reached, the test is started and the steering wheel will turn according to a sinus curve which is defined by an amplitude in degrees of steering wheel turning (not wheel angle), and a wavelength. The amplitude and wavelength are input parameters along with the target velocity and the reference friction.

Data generation

The input scenarios that are produced from the input configuration are split into groups based on the number of simulation instances to run. As it is possible to run a large set of input scenarios, the time required to run the simulations can become long. The short configuration example in table 3.1 produces 60 scenarios where each scenario run for 20 seconds plus the additional time for resetting between tests. For real-time simulation, this small configuration takes 20 minutes to run, hence it is desired to be able to run the scenarios in parallel, especially when configuring for thousands of scenarios.

An empirical test was run showing that a single core per simulation instance was sufficient, and for a computer using newer processors with 8 available cores, total simulation duration would be cut to two and a half minutes. This highlights the applicability of the process design where the computational power is the limit for testing a large set of scenarios.

Each instance of the simulation software follows the data-flow seen in Figure 3.7, where the group of scenarios are converted into one file for each simulation instance and a log file is created accordingly. For every update of the physics engine, the inputs to the vehicle model is read, the vehicle dynamics calculations are performed according to Figure 3.8, and finally, the requested vehicle data is logged to the log file. This is repeated at a frequency of 50 Hz, meaning that the 60 scenarios would produce 60 000 data points for each parameter.

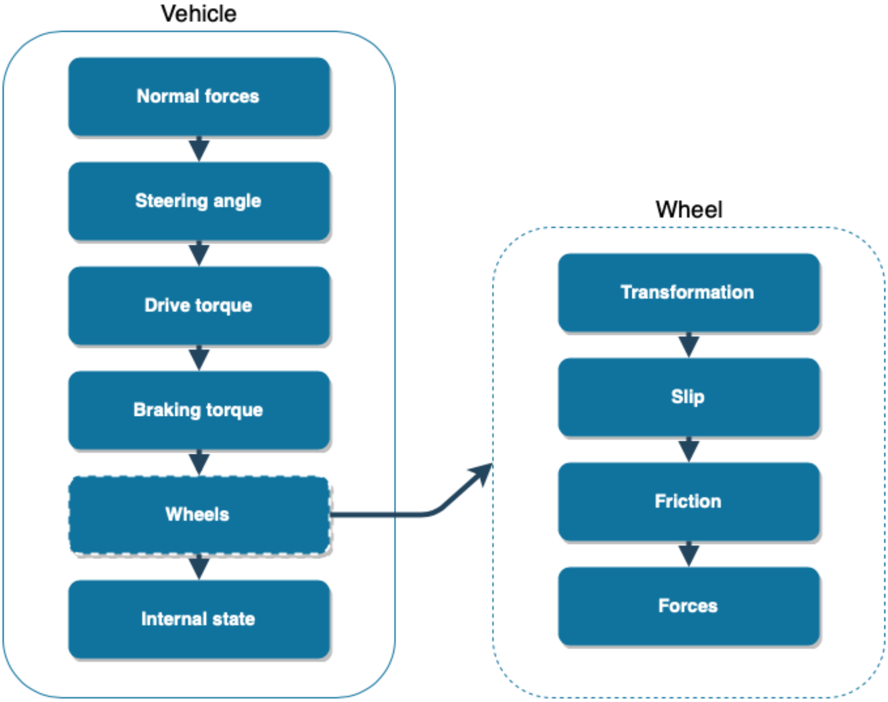


Figure 3.8 The order of vehicle script execution for every update.

The internal state component of the vehicle contains the computations that convert the forces applied at the wheels to translational and rotational acceleration.

4

Vehicle validation

For the process to be applicable for a specific real vehicle, the digital representation of the vehicle needs to behave similarly to the physical car. Since the inputs to the neural network are dependent on vehicle parameters and the vehicle state, too large inaccuracies would cause a mismatch, implying that the network tries to estimate the friction for a vehicle that is not a match with the intended one.

The usefulness of a simulation can be decided by the comparison of the digital vehicle (DV) and the physical vehicle (PV) in terms of their behaviour. If the intersection, $DV \cap PV$, is null, which means that there is no similarity between the DV and PV, no usefulness for the simulation can be found and the process is not applicable for the PV, otherwise it is useful to some level.

If $DV \subset PV$, the simulation model can be described as being a degree of accurate, but incomplete as the behaviour is comparable to the PV, but not all behaviour is included in the DV.

For $PV \subset DV$, the simulation is considered a degree of complete, but inaccurate as the behaviour of the PV is displayed by the DV, but additional behaviour was also encountered from the DV. The goal is to find a DV, which is equivalent to the physical vehicle PV, $DV \equiv PV$, thus, is both complete and accurate [**27**].

For the purpose of the validation it is desired for the DV to reach a high degree of accuracy. Incompleteness is expected for all behaviour of the PV as data was not provided to cover all behaviour of the vehicle, however, completeness for the provided data can be analysed.

Full validation of a DV imply a problem of catch-22. To reach the goal of $DV \equiv PV$, both full accuracy and full completeness is required. In practice, this requires a perfect model or digital twin to be evaluated for every possible test scenario that could be encountered, and be compared to the behaviour of the PV running the same scenarios. This is highly resource consuming, and evaluating for the same test scenarios using a PV negates the benefits from using a DV if performed in an inefficient manner as the purpose of the DV is to not having to run tests on the PV.

However, this can be avoided by generalising the scenarios and reducing the value range of the vehicle parameters for the PV.

4.1 Model comparison

The aim of the comparison is to find similar behaviour for similar situations, that is, the same inputs to the DV and PV should optimally produce two identical logs from both vehicles. As the simulator was implemented to run on the same inputs that a human driver input to a car, in other words, the steering wheel angle, engine torque, and brake torque, the simulations can mimic driving patterns in the PV.

To validate the DV, the starting point was the data logs received from NEVS. The test scenarios run on the physical vehicle were performed at a target velocity with turning of the steering wheel, implying the use of cruise control for the DV. As the tests were focused on lateral behaviour at a relatively constant velocity, lateral variables were used for the comparison. The following figures depict the steering angle, the yaw rate, and the lateral acceleration of the DV and the PV.

Validation scenario 1

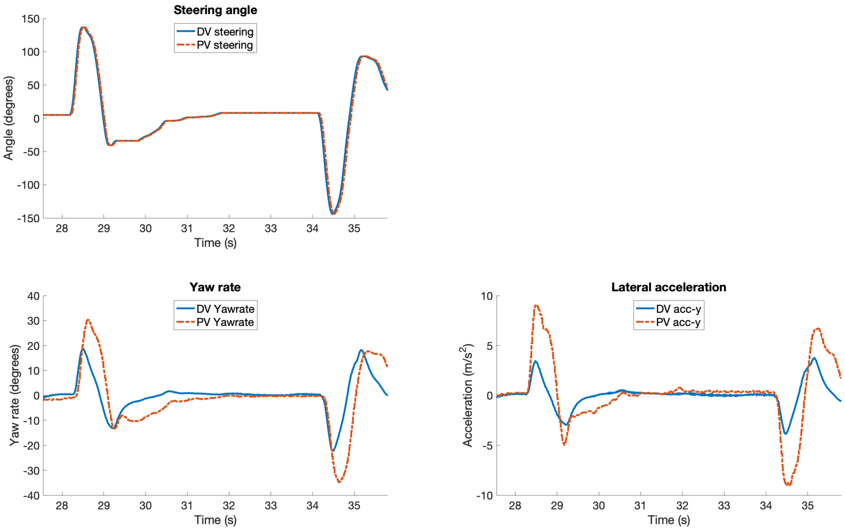


Figure 4.1 Validation scenario 1. Comparison between DV and PV for steering angle, yaw rate, and lateral acceleration for velocity 1.

From the first test scenario, it can be noted that the digital vehicle performs the major movements of the physical vehicle, but minor changes are not captured. Furthermore, the digital vehicle has trouble with magnitude, especially for the lateral acceleration.

Validation scenario 2

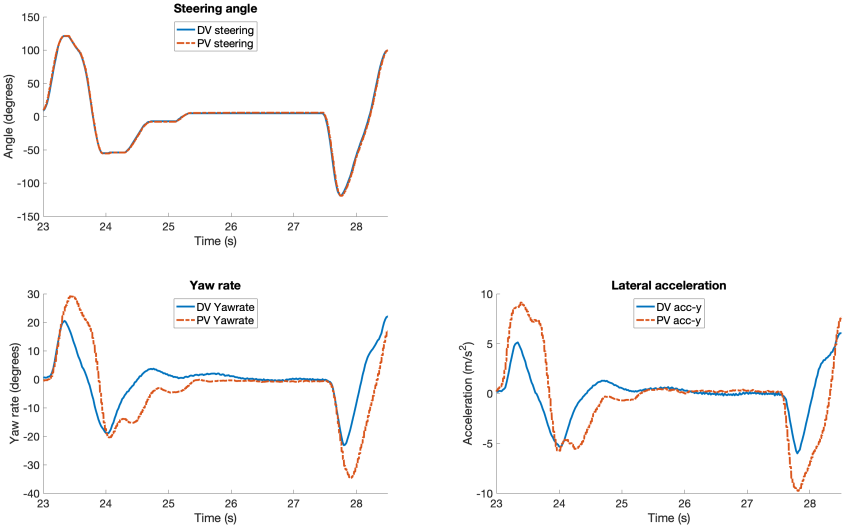


Figure 4.2 Validation scenario 2. Comparison between DV and PV for steering angle, yaw rate, and lateral acceleration for velocity 2.

Similar behaviour to the first test scenario is displayed for the second test. Once again, the DV experiences problems with magnitude but peak values are closer to the PV, than in scenario 1.

Validation scenario 3

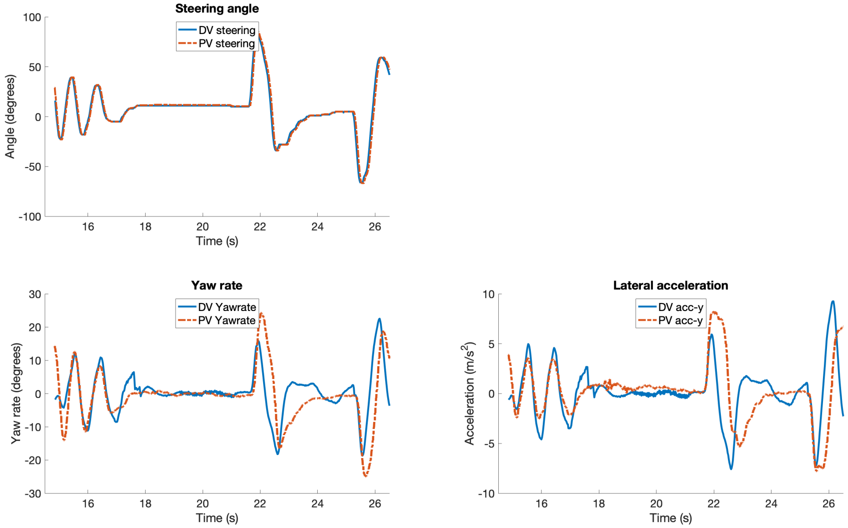


Figure 4.3 Validation scenario 3. Comparison between DV and PV for steering angle, yaw rate, and lateral acceleration for velocity 3.

In the initial stage of the test, the yaw rate and lateral acceleration are not aligned until the first peak, as is expected due to the values starting close to zero for the DV, which is not the case for the logged data. The yaw rate for the DV and PV behave at a previously unseen level of similarity in the beginning, and the lateral acceleration peaks for the DV are slightly larger than for the PV. For the turns with a larger steering wheel angle, the behaviour seen in Figure 4.1 and Figure 4.2 can once again be noted.

Validation scenario 4

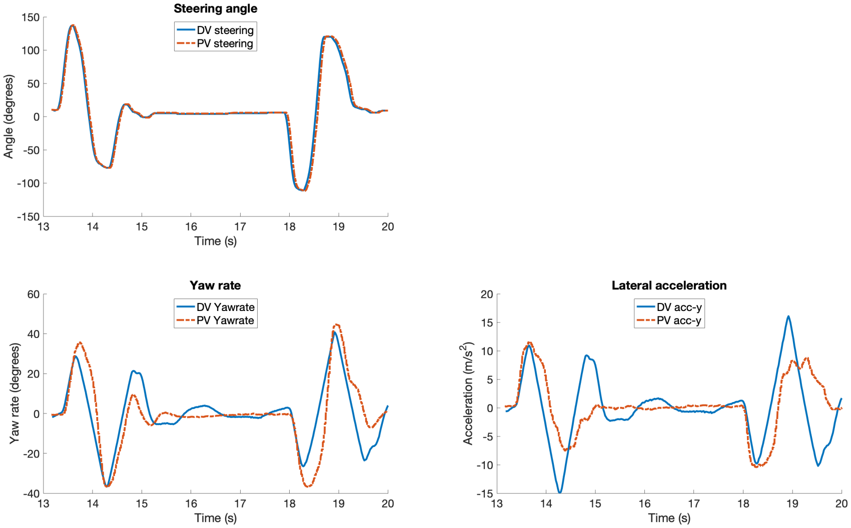


Figure 4.4 Validation scenario 4. Comparison between DV and PV for steering angle, yaw rate, and lateral acceleration for velocity 4.

For test scenario four, the DV displays similar behaviour as previously but with more disturbances in the mid segment. Moreover, the lateral acceleration peaks have a higher magnitude than the PV.

In general, the digital vehicle displays part of the behaviour of the physical vehicle, where the DV can be seen to behave with a level of similarity to the PV for the major actions. The magnitudes for the actions vary, and a higher level of accuracy would benefit the digital vehicle. As expected, the digital vehicle demonstrates $DV \subset PV$. However, it does so with a degree of accuracy and completeness that is moderate.

The usage of an imperfect vehicle model, where $DV \equiv PV$ is false, will have negative consequences on the accuracy results of the process evaluation. However, as briefly mentioned in the Sections 1.2 Purpose and 1.4 Delimitations, the purpose of the thesis is not to create a perfect model as the effort is highly resource demanding. The errors from the validation will extend when adapted from simulation to the logged data for the neural network as the errors displayed in the validation will result in input values that are not perfectly aligned with what would be the inputs from the PV. While the validation is not optimal, it is deemed adequate for the continued process evaluation. The results in the section on process evaluation will highlight and discuss differences in achieved performance for scenarios where the impact of the vehicle model varies.

5

Neural networks

The complex modelling of the tyre dynamics motivates the effort to use neural networks to find a link between sensor values and the friction quotient. However, the use of neural networks, or machine learning in general, in applications that can be deemed as life-critical is discussed, with one of the reasons being that neural networks can be labelled as "black boxes". While a network can perform very well for the task of approximating a function [**28**], the structure and variables of the network does not give much insight into the function that is being approximated, thus it can be difficult to analyse the network and understand the inner workings of it. However, the discussion whether neural networks, for the "black box" reasoning, should be allowed to be used in safety systems is neglected for this thesis as the performance of neural networks for friction estimation is evaluated.

In order to avoid having to re-invent the wheel and manually implement mathematical optimisation methods, back-propagation, and other parts, a neural network API was used for the project. Keras [**29**] is written in Python and was used for the machine learning tasks for the thesis.

5.1 A brief outline of neural networks

This section briefly covers theory on neural networks in order to increase the understanding of the mathematical basis to the work that has been done for the thesis.

Neurons

Figure 5.1 below represents the mathematical model of a single artificial neuron, which consists of inputs, weights, bias, an activation function, and an output.

The neuron has scalar inputs x, also called features, and scalar weights w. The inputs are multiplied with the weights to form weighted inputs, which are summed together with the bias b. The bias can also be modelled using a constant input of

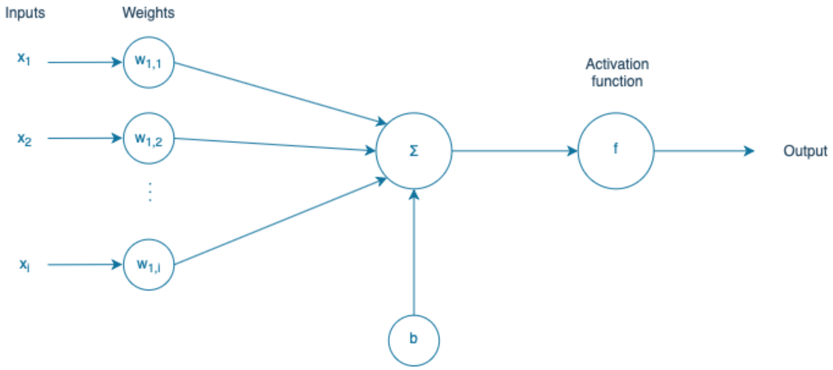


Figure 5.1 A single artificial neuron.

1 with the weight b. The summed value is input to an activation function which produce the output value. Hence, the output can be described by:

$$output = f\left(\sum(wx) + b\right) \tag{5.1}$$

Single artificial neurons can be combined such that the output from one neuron is the input to one or several other neurons, and can thus be stacked both horizontally and vertically. This forms layers of neurons and increases the computational capabilities and complexity compared to a single neuron.

Activation functions

An activation function is a modification used for the summed inputs and bias to alter the value towards a selected goal. Both linear and non-linear functions can be used, where the selected function should be chosen based on the specification of the problem [**30**]. Three common activation functions are the hard limit function, the linear function, and the sigmoid function.

The hard limit function outputs a zero if the inputs are less than zero, and one if the inputs are greater or equal to zero. The hard limit activation function is useful for cases where a binary result is expected.

A linear function provides more information to the output than the hard limit function, but two problems exist. First, the derivative is a constant which is not preferred for training. Secondly, using several linear functions in a chain will produce an output linear to the first input no matter the length of the chain, and cannot be used to approximate non-linear problems.

The sigmoid function is a non-linear function where the output is limited between zero and one. An advantage versus the hard limit function is that a value close to zero will not have the same effect on the output, for example, 0.49 and 0.51, compared to 0 and 1, depending on the sign of the input. Furthermore, the derivative is the steepest closest to zero, which implies that the function has a tendency to move towards one end of the curve. An alternative to the sigmoid function is the hyperbolic tangent function which is a scaled sigmoid function with the same properties, except that the range of values is between -1 and 1.

Backpropagation

When sufficient layers of neurons have been connected to form a neural network, the network requires training in order to be able to solve the associated problem. Each neuron has three variables that can be changed in order to alter the behaviour of the network, the weights, the bias, and the activation function. The activation function is a design choice and cannot be changed during training, leaving the weights and the bias.

To alter the weights and the bias, a solution has been presented using Bayesian back-propagation. The methodology of back-propagation is to design a cost function for the neural network, which can be used to alter the variables based on the cost that each input has [**30**]. The purpose is to find the minimum of the cost function, transforming the problem into an optimisation problem.

For a set of inputs **x**, a correct value **t** exists for every output, implying that the cost can be quantified. A common method is the *mean squared error*, where the cost (error) can be defined by $e = (t - output)$. The cost function $F(\mathbf{x}) = E(e^2)$ can then be minimised using stochastic gradient descent where the partial derivatives for the weights and biases are [**30**]:

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha \frac{\delta F}{\delta w_{i,j}^m} \tag{5.2}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha \frac{\delta F}{\delta b_i^m} \tag{5.3}$$

where α is learning rate of the network. Applying these derivatives to a situation where we have multiple inputs, a sensitivity can be found for every weight and bias for the received output value. Thus, the inputs are propagated through the network to receive a final output value, which is compared to the true value and an error can be computed. Finally, the error can be propagated backwards - hence the name *back-propagation* - to alter the value of each weight and bias based on the error that their respective neuron produced. By repeating the process numerous times, a minimum to the cost function can be found.

The advantage of using a Bayesian method is that the error approaches the lower bound, decided by the backpropagation method used, when the training set approaches infinity, and will on average have equal or lower error compared to any other method if certain constraints are followed [**31**].

Training requirements

In general, it is difficult to include previous knowledge when creating a neural network. This results in networks being dependent on the data they are given and can only perform on a level as good as the data is. Neural networks are not good at extrapolating results, meaning that if the network has not been trained on data for a certain situation, it will not perform well when evaluating data from that kind of situation [**30**]. For this reason, the network needs to cover the full range of input variables that it can be exposed to during evaluation.

It was agreed upon by all partners to the thesis that the ranges for the following variables were of the most relevant for the vehicle model:

- Longitudinal slip κ_x - [-0.25, 0.25]
- Lateral slip α - [-0.15, 0.15]

However, as the logged data from the physical vehicle did not include any scenarios with negative longitudinal slip, only non-negative values were included in the generated data. The data coverage for those variables can be seen in Figure 5.2 and 5.3 where it can be noted that the range are covered by the training set.

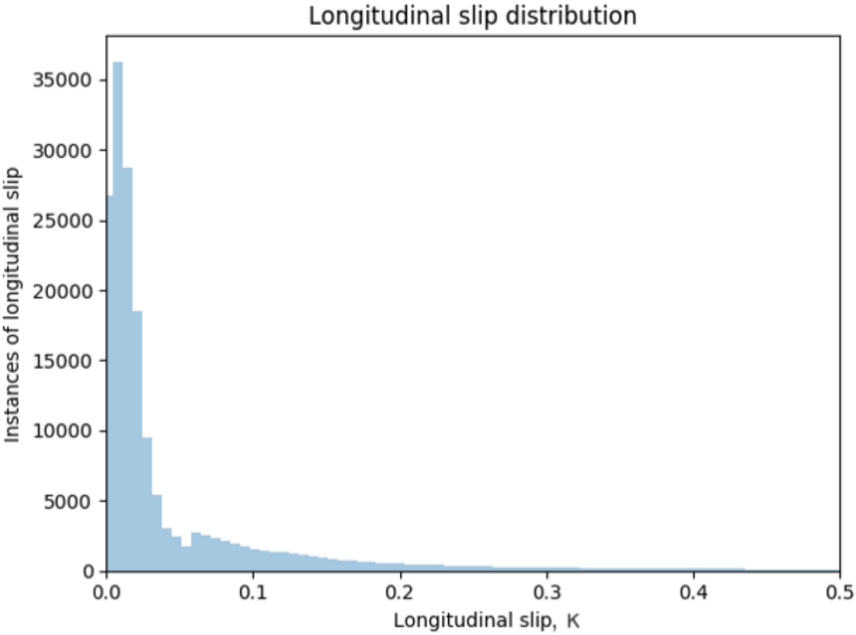


Figure 5.2 Longitudinal slip distribution.

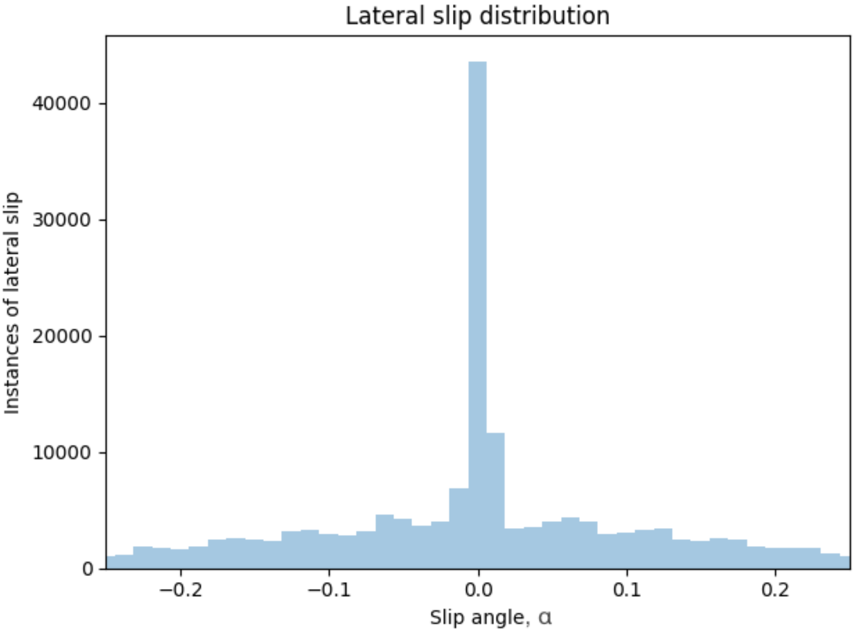


Figure 5.3 Lateral slip distribution.

5.2 Neural network architecture and design

One neural network type was used for the process, a *multilayer perceptron* (MLP). A MLP is a feedforward type of a neural network where the input is translated from the input layer to the output layer without loops. A MLP neural network consists of at least three layers, the input layer, a hidden layer, and an output layer, where each neuron in every layer is connected to all neurons in the previous and subsequent layer. The MLP type of network is known for being able to approximate universal functions [**33**; **34**; **28**], and can therefore be applied to the underlying work of the thesis.

Input selection

Two types of input selection were chosen for the thesis, a *theoretical* selection, and an *applicable* selection. All data that is generated from the simulations and passed to the neural network is noiseless to allow for a fair comparison, as the data received from the real-world testing vehicle is filtered.

The applicable selection contains features that can be measured from the physical vehicle, thus the network can be applied on the physical vehicle without requiring an addition of a physical component to the vehicle if the available computational power exists. Worth noting is that the required computational power is low as it only needs to perform the multiplications required to transverse the neural network once. The inputs for the applicable input selection are as follows,

- Angular velocity for the front-left wheel
- Vehicle forward velocity
- Vehicle longitudinal acceleration
- Vehicle lateral acceleration
- Vehicle yaw rate
- Acceleration pedal usage in percent, (0-100)
- Steering angle

The theoretical selection of features contains inputs that have been deemed as relevant to determine the friction quotient. The selection is named *theoretical* as it includes variables from the vehicle and tyre state of the digital vehicle that are not measured on the physical comparison vehicle. The inputs used for the theoretical input selection are the same as for the applicable selection, as well as,

- Angular velocity for the remaining wheels
- Normal force for the front-left wheel

- Longitudinal force for the front-left wheel
- Lateral force for the front-left wheel
- Vehicle yaw acceleration
- Vehicle lateral jerk
- Longitudinal slip for the front-left wheel
- Lateral slip for the front-left wheel

It is worth noting that several of these variables cannot be directly measured for a real vehicle, but have been included for the evaluation of the network. Furthermore, the inputs related to the front-left wheel are used as it is the friction coefficient for the front-left wheel that is estimated. By using inputs to the network that are closely related to the friction coefficient, a higher accuracy is expected and a reference for the optimal performance of this type of network model is achieved.

SDP input For the first approach, for each of the input selections, the features were fed to the network for every timestamp with logged data. This approach, where the set of features are made of *Single Data Points* (SDP), allows the network to estimate a friction every time new values are logged from the vehicle.

CDP input The second approach to the inputs were to use one set of features representing the sensor values, identical to the SDP approach, but also to include statistical parameters and short trends. To accomplish this, five additional inputs were added for every original input value. The value of the five additional parameters were the:

- mean
- median
- standard deviation
- maximum value
- minimum value

which were evaluated for previous SDP inputs. This yields a set of features where one part, SDP, is updated at a high frequency and one part, the aggregated statistical features, at a lower frequency, giving a feature set of *Combined Data Points* (CDP) that is the combination of the SDP and the aggregated features. The lower frequency is decided by the range of inputs to be included for the aggregation, where the value from the last five measures were used in the underlying work.

Network design

For both the theoretical and applicable selection of inputs, the same networks were used. The number of neurons, also called *units*, in all hidden layers was chosen arbitrarily, as an optimal network is difficult to design [**35**]. A collection of layer sizes were tested, ranging from 8 to 256, where the final number of units used was decided by finding a maximum of the accuracy. Note that the maximum could have been a local maximum, which was disregarded as it was out of scope to find an optimal network design.

For all designs of the network two hidden layers were used as it has been found adequate to approximate nonlinear functions [**33**].

The activation function used for all networks was the hyperbolic tangent function (tanh) as it has been shown to give the best accuracy, especially using a "Tanh-Tanh" network configuration [**36**]. The same was true during early evaluation where the tanh function gave the highest accuracy.

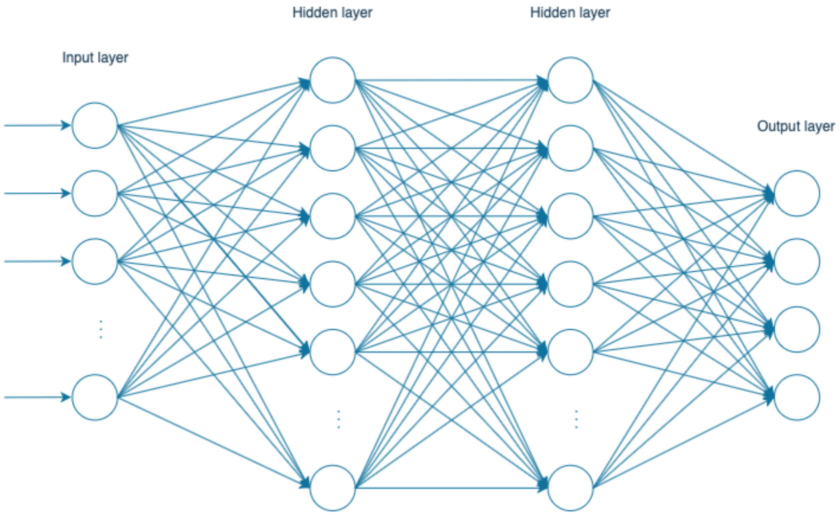


Figure 5.4 A model of a the MLP neural network that was used in the thesis with one input layer, two hidden layers, and one output layer. Compare to Figure 5.1 of a single neuron.

Normalisation

During the training of neural networks using backpropagation the error value used for the cost function should decrease over time as the model is trained. This implies that changes in the weights and biases are also reduced over time as the network is fine-tuning the parameters at the end of training. This can be troublesome in combination with sets of features where the value of a feature has a large distribution.

If the multiplication of the input and the weight, $x \cdot w$, is considerably more affected by the shift in value of the input than the shift in value of the weight, the training becomes more difficult. Thus, normalisation is used to reduce the shift in values, called "internal covariate shift". Normalisation transforms the inputs so that the inputs have a mean of zero and a variance of one, which helps reduce the training time and increases the generalisation performance of the network [**37**].

Overfitting

Overfitting is a phenomenon where the neural network performs unreasonably well on the training data in comparison to the evaluation data. The problem exists when the model is trained on data that it already has seen or data that is similar to previous data. When the network is exposed to previously unseen values of the features the network will perform worse as it has not been trained for anything close to the new combination of feature values. To avoid overfitting, the network needs to be trained for the full range of values for every feature that the network can be exposed to. Furthermore, a technique called *dropout* can be used [**38**].

Dropout is an idea based on the genetic optimisation algorithms [**38**] where the genes that are passed on to the child are changed by a small random mutation. This is applied for neural networks by adding a random chance for neurons to be removed or set to zero for a single set of features during training. While it has been found to slightly increase training time, it has also been found to increase the generalisation of the network, hence decreasing overfitting [**38**]. For the underlying work, a 20 % dropout rate was used for all network designs.

5.3 Neural network output

To evaluate and compare the neural network models, two types of results were used, *regression* and *classification*. From discussions with NEVS, it was decided that the thesis was to focus on classification, but regression was used to put the results from the classification into more context.

Using regression the neural network has one output neuron and tries to pinpoint the exact value of the output, that is, the friction.

For classification, the network has multiple output neurons equal to the amount of categories that the output can be mapped to, where a category is decided from the value at each output neuron.

Regression

For the regression neural network, the network had the same design for both the theoretical and applicable input selection as no increase in accuracy was received by increasing the number of neurons in the hidden layers nor from increasing the number of hidden layers.

The network was setup with either 7 or 17 input neurons, depending on the input selection. Furthermore, only SDP was used for the regression. The network had two hidden layers with 32 units in each layer using the hyperbolic tangent activation function. The input was normalised and a 20% dropout was used. For backpropagation, the optimisation algorithm *ADAM* [**39**] was used, and the loss was measured using the mean squared error.

Classification

For the classification of friction, four categories were decided to be adequate, where the range indicates which category that is correct based on the value of the friction quotient.

- Ice - $0 < \mu \leq 0.15$
- Snow - $0.15 < \mu \leq 0.4$
- Wet asphalt - $0.4 < \mu \leq 0.7$
- Dry asphalt - $0.7 < \mu$

Two neural network models were evaluated for both the theoretical and applicable input selections as well as for both SDP and CDP.

First, a network that is of similar design to the regression model is used to be able to compare the results. Hence, only the output layer differs as four output neurons are required for the classification. A *softmax* activation function was used for the output layer which transforms the value of the outputs to be distributed in a range between zero and one, where the sum of all outputs equal to one. Thus, the outputs can be interpreted as the percentages for the inputs to belong in a certain category. The loss function was changed to *categorical cross-entropy* [**40**] where the loss is determined by:

$$H(p,q) = -\sum p(x) log(q(x)) \tag{5.4}$$

where p is the true distribution, [0, 1, 0, 0] for the case of snow, and q is the distribution from the output layer. This network was used for inputs identical to the regression problem.

Second, an identical network to the first was also used but the number of neurons in the hidden layers were doubled to 64 to take the increased number of inputs using

CDP into account. For this approach the number of inputs were increased for the applicable inputs and theoretical inputs to 42 and 102 respectively.

6

Process evaluation

For the process to be applicable and provide value, the benefits gained from using simulation and neural networks, such as comparing to the true value for estimation, need to outweigh the loss from the sources of errors that have been added by the process. The sources of errors include the accuracy of the digital vehicle and the performance of the neural network.

There are three approaches for the process to be applied to friction estimation, and to be evaluated for:

1. Train neural network (NN) on generated data, evaluate on unseen generated data.
2. Train NN on generated data, evaluate on data from physical vehicle
3. Train and evaluate on data from physical vehicle.

The first approach removes sources of error from the process as the digital vehicle is used to generate data for both the training and the evaluation, hence temporarily removing the physical vehicle from the process. Even though the differences between the DV and PV do not affect the results, the choice of vehicle dynamics model for the DV has an impact on the results as details neglected in the vehicle model reduce the potential performance of the process due to loss of information. Since several sources of errors related to the vehicle dynamics are eliminated, the approach will give more accurate information regarding the usefulness of the neural networks, and was used for the evaluation for regression and classification, for all neural network designs, and for all types of input, namely SDP, CDP, applicable selection, and theoretical selection.

The second approach is most useful in combination with the completion of the first approach. As the second approach includes the physical vehicle in the evaluation process, the evaluation results can be compared to the evaluation results from the first approach to see how well the neural network performs on the physical vehicle. However, as previously mentioned, some difficulties exist as to deciding the root causes of the differences in results.

The third approach would be useful to evaluate for at least two possible reasons. One, if the neural network can predict better than the existing estimation method in the physical vehicle, or two, if the NN can continue to train after being installed in the vehicle. However, the amount of logged data required for the third approach was not available during the underlying work and the approach could not be evaluated. The lack of logged data from the PV strengthens the case for the problems of the testing process presented in the introduction to the thesis.

6.1 DV performance evaluation

The following section presents the performance of the neural networks for the regression and the classification problems for approach one, thus the networks are trained and evaluated using the digital vehicle. The same training data has been used for all networks, and the architecture and design follow the descriptions in Section 5.2.

Regression

The results for the neural networks using regression differ from the classification problems as a right or wrong cannot be presented as clearly. Thus, the performance was measured using *mean absolute error* (MAE), which can be described using:

$$MAE = \frac{\sum^{n} |y_i - t_i|}{n} \tag{6.1}$$

where y_i is the estimate from the neural network, and t_i is the actual friction value.

The neural network using the applicable input selection, achieved a MAE of 0.09. Whereas the same network using the theoretical input selection achieved a MAE of 0.06.

Classification

The results for the classifications are presented using the accuracy in terms of how many percentage of the data that was placed into the correct friction category.

More details for the accuracy are given by confusion matrices which display how the network tried to place the data for each category. Each row in a confusion matrix represents the estimations of the network for the category of that row. Thus, a number at location (x, y) in the confusion matrix represents the percentage of input data for the category y that the network predicted belong in category x. Hence, an optimal confusion matrix using normalised values, should contain ones in the diagonal direction, and zero for every location where $x \neq y$.

SDP using the applicable input selection The neural network using the SDP approach and the applicable input selection achieved an average accuracy of 85.1% with the distribution according to Figure 6.1 below.

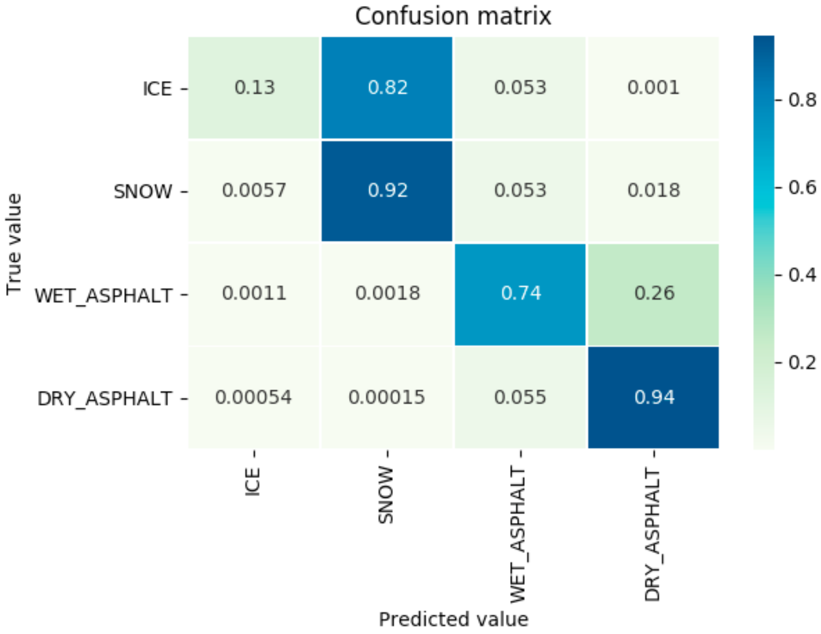


Figure 6.1 The confusion matrix for the classification using SDP and the applicable input selection.

From the confusion matrix it can be seen that the model mostly predicts scenarios with ice as snow. This result can be altered by for example changing the ranges for the categories, or improving the neural network model. However, changing the ranges post-result was deemed to not provide any value to the evaluation of the process. Neither is the purpose of the thesis to find an optimal NN, thus were the results kept.

SDP using the theoretical input selection Using the same network as in Figure 6.1, but increasing the inputs to include all theoretical inputs achieved an average accuracy of 91.8%. By using the theoretical input selection instead of the applicable, the accuracy for all categories improved but no large change can be noted.

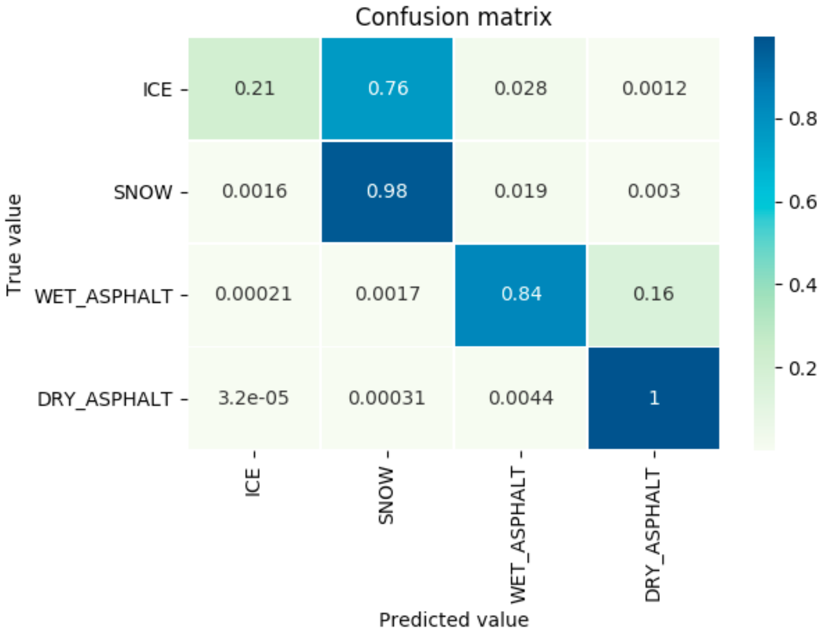


Figure 6.2 The confusion matrix for the classification using SDP and the theoretical input selection.

The neural network model using the theoretical input selection does also mostly predict ice as snow, and the reasoning for the previous result apply.

CDP using the applicable input selection The neural network for CDP using the applicable input selection uses 42 input neurons, and the hidden layers have 64 neurons each. This yields an average accuracy of 89.2%. By providing the network with the additional statistical inputs, the accuracy has increased by 4.1% in comparison to using SDP for the applicable input selection.

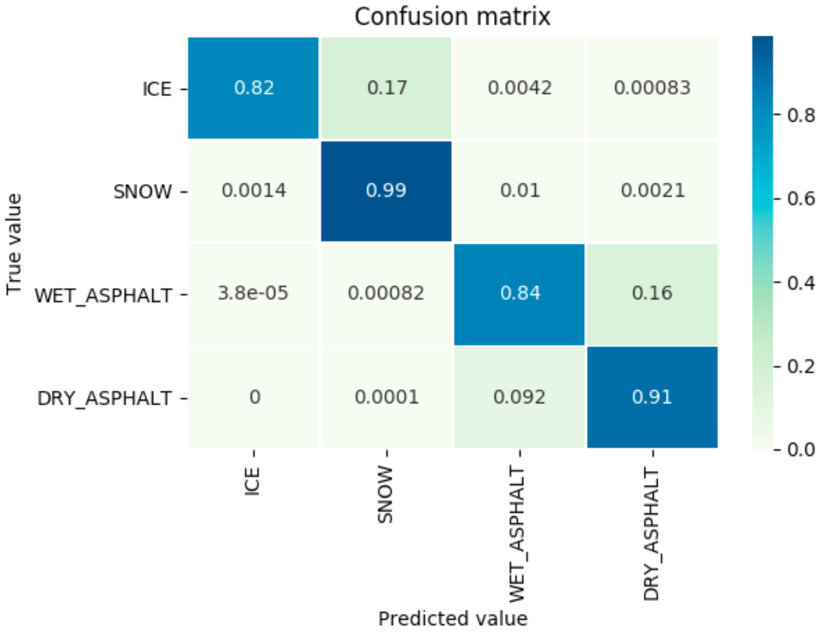


Figure 6.3 The confusion matrix for the classification using CDP and the applicable input selection.

From the confusion matrix, it can be seen that the additional inputs made the network better at classifying the ice category and changed from an accuracy of 13% to 82% which provided a substantial amount of the overall accuracy gain. Worth noting is that the accuracy for the dry asphalt category worsened. As the vehicle log data used for all DV performance evaluations is the same, and SDP is a subset of CDP, the result should not be seen if the two neural network models were trained and evaluated a large amount of times.

CDP using the theoretical input selection The neural network for CDP using theoretical input selection, uses 102 input neurons, and the hidden layers have 64 neurons each. This yields an average accuracy of 94.3%.

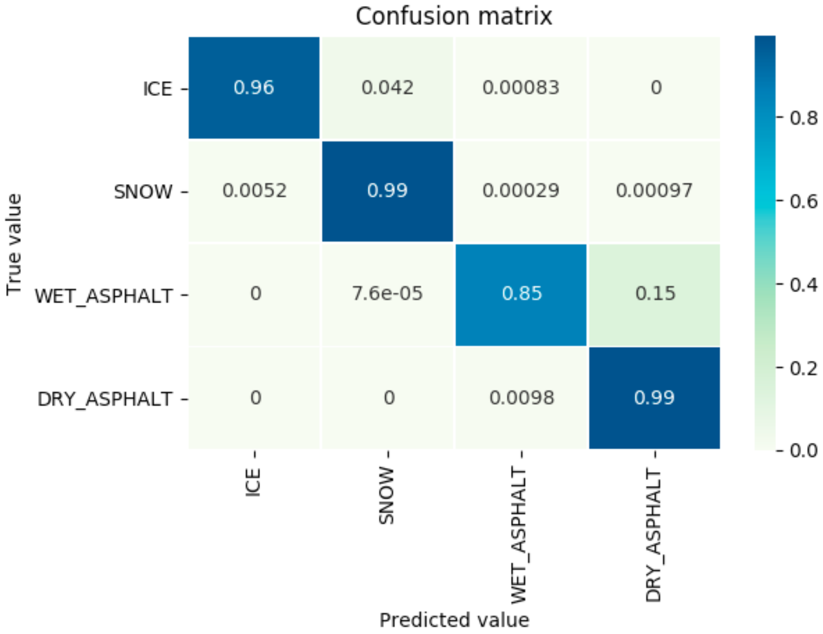
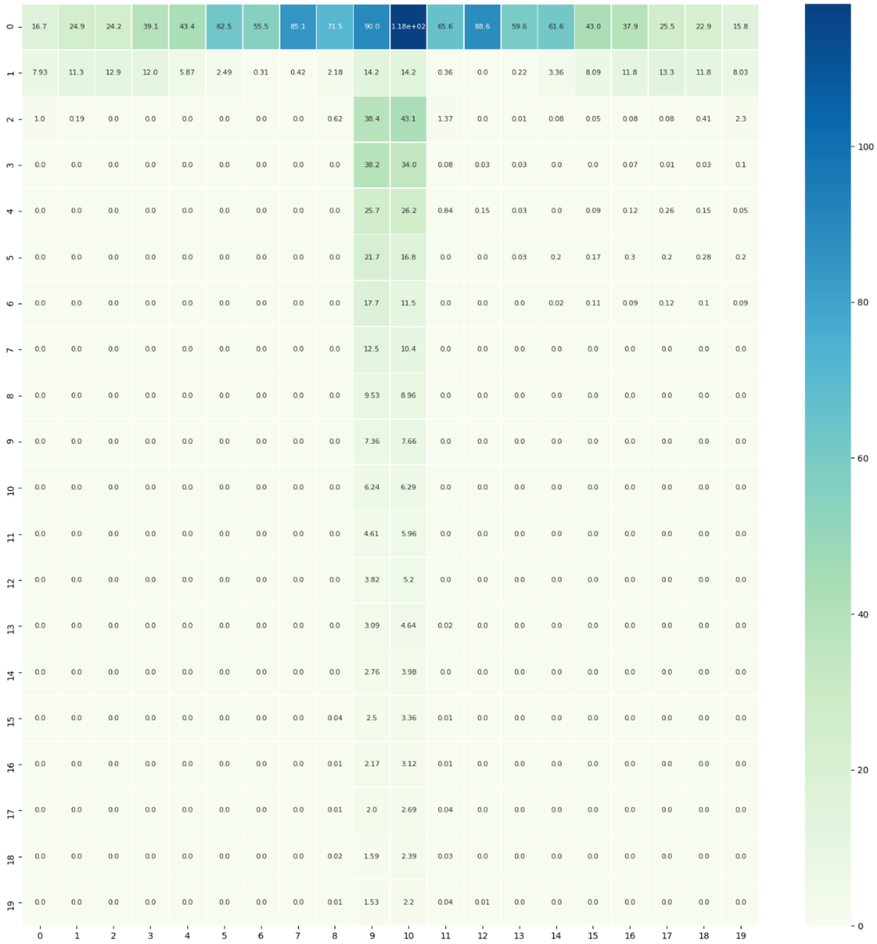


Figure 6.4 The confusion matrix for the classification using CDP and the theoretical input selection.

In comparison to using SDP for the theoretical input selection, the neural network model now predicts the ice 96% of the time instead of 21%, and 14% better than using CDP for the applicable input selection. For the other categories, the network predicts with an approximately equal accuracy in comparison to SDP for the theoretical input selection. However, the snow and dry asphalt categories were hard to improve due to the already high accuracies.



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	16.7	24.9	24.2	39.1	43.4	62.5	55.5	85.1	71.5	90.0	1.18e+02	65.6	88.6	59.6	61.6	43.0	37.9	25.5	22.9	15.8
1	7.93	11.3	12.9	12.0	5.87	2.49	0.31	0.42	2.18	14.2	14.2	0.36	0.0	0.22	3.36	8.09	11.8	13.3	11.8	8.03
2	1.0	0.19	0.0	0.0	0.0	0.0	0.0	0.0	0.62	38.4	43.1	1.37	0.0	0.01	0.08	0.05	0.08	0.08	0.41	2.3
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	38.2	34.0	0.08	0.03	0.03	0.0	0.0	0.07	0.01	0.03	0.1
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	25.7	26.2	0.84	0.15	0.03	0.0	0.09	0.12	0.26	0.15	0.05
5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	21.7	16.8	0.0	0.0	0.03	0.2	0.17	0.3	0.2	0.28	0.2
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	17.7	11.5	0.0	0.0	0.0	0.02	0.11	0.09	0.12	0.1	0.09
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	12.5	10.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	9.53	8.96	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	7.36	7.66	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	6.24	6.29	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.61	5.96	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.82	5.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.09	4.64	0.02	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.76	3.98	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.04	2.5	3.36	0.01	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
16	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.01	2.17	3.12	0.01	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
17	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.01	2.0	2.69	0.04	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
18	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.02	1.59	2.39	0.03	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
19	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.01	1.53	2.2	0.04	0.01	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 6.5 Heatmap showing all data points over lateral slip in the x-direction, and longitudinal slip in the y-direction, where the slip data is divided into a matrix combining longitudinal slip from Figure 5.2 and lateral slip data from Figure 5.3.

Figure 6.5 presents all data points for the lateral slip in the x-direction and longitudinal slip in y-direction. The value in each square represents the number of occurrences divided by a factor of 100 for that combination of longitudinal and lateral slip. The longitudinal slip values are in the range from 0 for index 0, and 0.5 for index 19. The lateral slips range from -0.25 to 0.25 for 0 and 19 respectively. It can be noted that few of the data points contain both a high lateral and high longitudinal slip, implying that the network is not trained for such occurrences and cannot be correctly evaluated for such conditions.

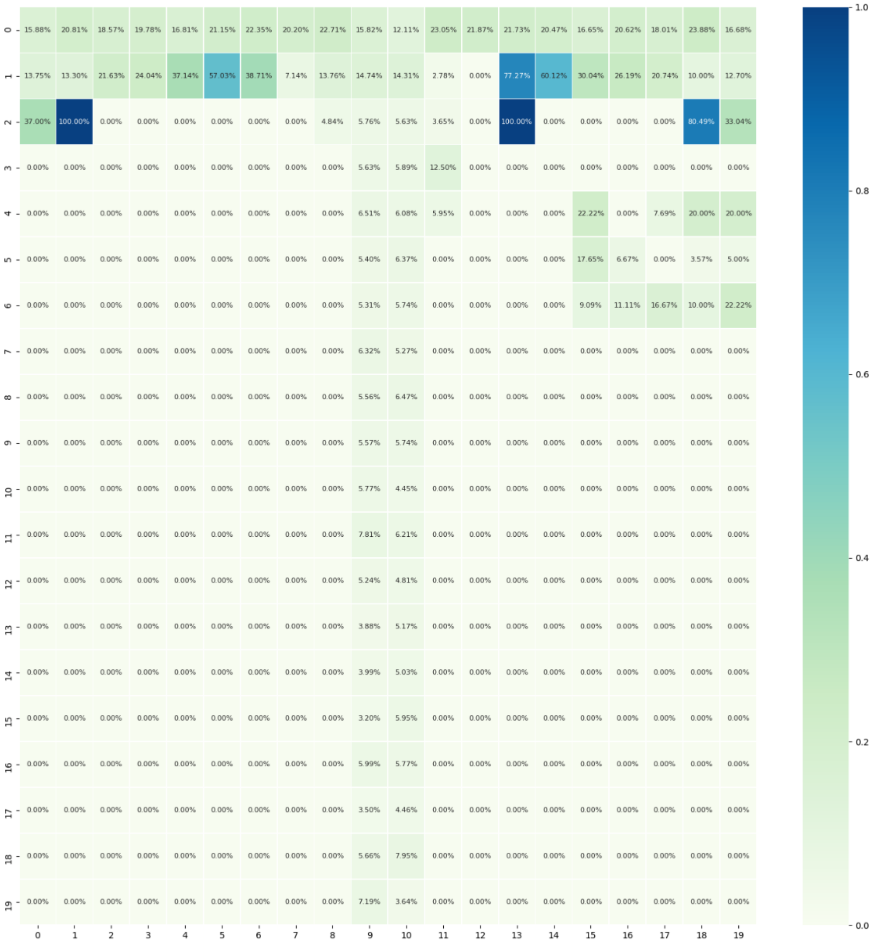


Figure 6.6 The heatmap is the same as for Figure 6.5 but shows the distribution of miss-classified data for the same longitudinal and lateral distribution instead.

The heat map in Figure 6.6 presents the percentage of data points for each group of longitudinal and lateral slip that have been classified incorrectly. The axes correspond to and use the same values as in Figure 6.5.

From the results of the regression and classification, it can be noted that the theoretical input selection performs better than the applicable input selection for each configuration of the network. This is expected behaviour as the inputs in the applicable input selection are a subset of the inputs in the theoretical input selection, that is, $applicable \subset theoretical$. However, some of the inputs used for the theoretical input selection can currently not be measured on the physical vehicle and the results are thus not usable for the PV.

Using the same mathematical reasoning, the results using CDP are expected to always be better than using SDP, as $SDP \subset CDP$ and all value contributed from SDP is included in CDP.

6.2 PV performance evaluation

For the second approach, the neural networks are trained on the generated data and evaluated on the data logs from the physical vehicle, and the same data logs used for the vehicle validation are used for the evaluation. The neural network model is trained using the CDP approach with the applicable input selection, and has been used for all evaluations on data from the physical vehicle.

Evaluation scenario 1 For evaluation scenario 1, the neural network had an accuracy of 60.5%.

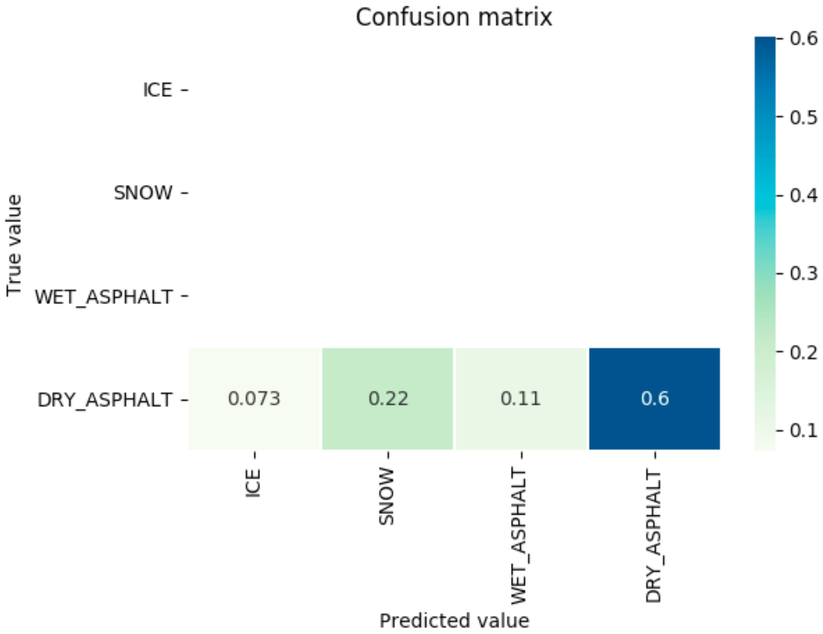


Figure 6.7 The confusion matrix for the PV evaluation for scenario 1.

Evaluation scenario 2 For evaluation scenario 2, the neural network had an accuracy of 43.1%.

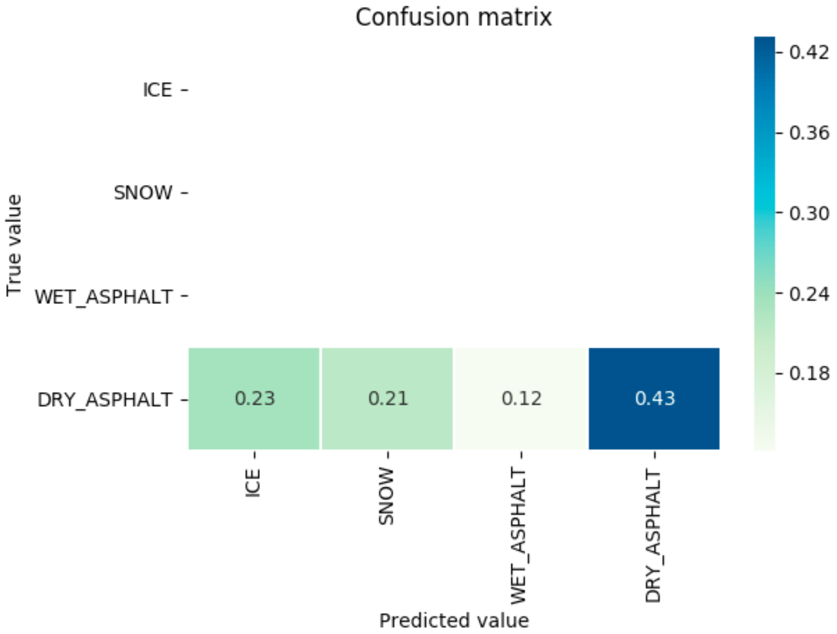


Figure 6.8 The confusion matrix for the PV evaluation for scenario 2.

Evaluation scenario 3 For evaluation scenario 3, the neural network had an accuracy of 37.4%.

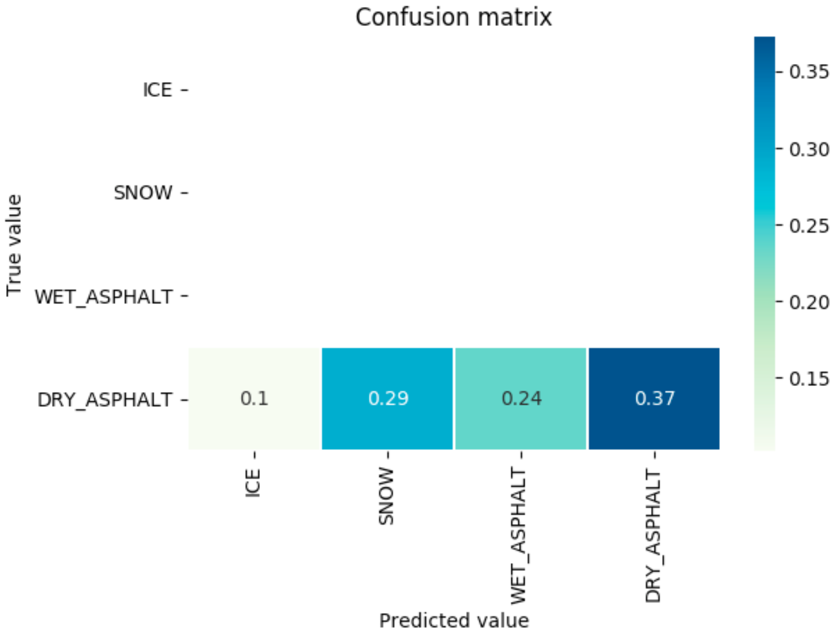


Figure 6.9 The confusion matrix for the PV evaluation for scenario 3.

Evaluation scenario 4 For evaluation scenario 4, the neural network had an accuracy of 73.7%.

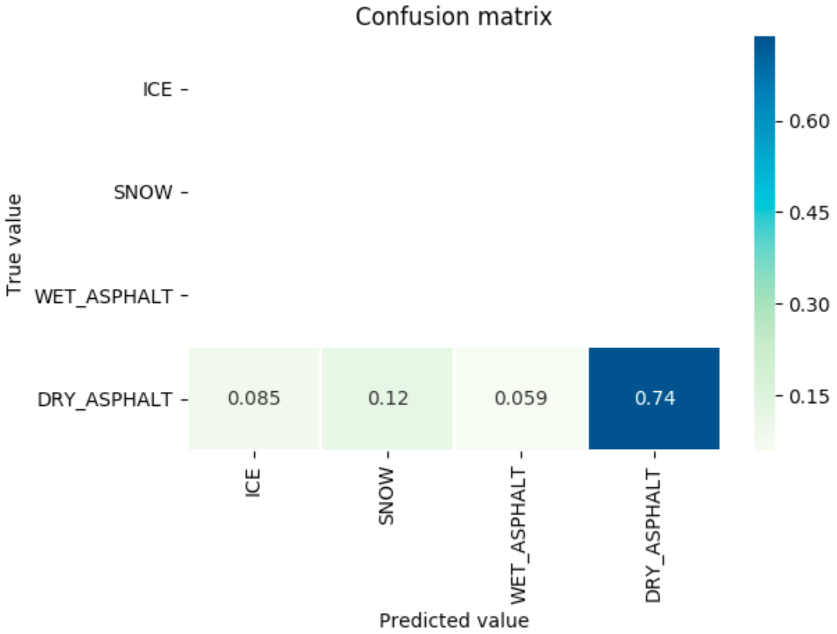


Figure 6.10 The confusion matrix for the PV evaluation for scenario 4.

It can be noted that all evaluations performed on logs from the physical vehicle perform worse than the DV evaluation at 89.2%. A worse performance was expected as at least two major sources of error have been introduced:

First, the physical vehicle which is not equivalent to the digital vehicle as evident from Section 4.

Second, differences in neural network input data. For the DV evaluation, the network was evaluated on different test scenarios than it was trained on, but it was still evaluated on straight line acceleration and sinusoidal turning. The logged data from the physical vehicle include large turning manoeuvres, but those scenarios differ from the scenarios used for training.

It should be noted that the logged data did not specify the friction quotient, meaning that a regression evaluation for the physical vehicle could not be performed, and the road surface used for the tests are assumed to fit into the range of the dry asphalt category.

6.3 Sensitivity analysis

In an effort to analyse the effect of changes to the vehicle model, a sensitivity analysis was performed on the digital vehicle. It was presumed that the vehicle parameter that will differ the most for the vehicles is the sprung mass as it is dependent on the driver and eventual passengers.

Using similar test scenarios as previously but with a smaller data set, a baseline performance was acquired with the same vehicle model that was used throughout the underlying work. The weight of the vehicle was adjusted in six steps of 10% from -30% to 30%, where each iteration ran the same test scenarios as the baseline model.

The performance for each model was as follows:

Table 6.1 Sensitivity analysis for sprung mass.

Weight difference	Absolute percentage point difference
-30%	20.0
-20%	5.0
-10%	16.4
0	0
10%	10.6
20%	8.2
30%	4.3

As the sensitivity analysis uses a smaller data set than previous evaluations, the neural network model cannot achieve the same level of generalisation and is more sensitive to inputs that it has not previously seen. Since all tests were run using the same test scenario, the effect from using a smaller data set was expected to be reduced. Still, from the results it can be seen that the performance of the neural network is unreliable for input that it has not been trained on. The expected behaviour was for the difference to the baseline performance to increase with a larger difference in weight, but the achieved performance is unexpected, which highlights the need for extensive training.

7

Conclusion

The process has shown that a setup which induces low static errors, that is, the vehicle model is accurate, and the training data is similar to the evaluation data, performs well. The optimal setup using the CDP approach with the theoretical input selection, reaches an accuracy of 94.3% which is the highest accuracy that was reached for the process. For an applicable process, the highest possible result with the current neural network setup and input data is around 89.2%.

For the evaluation using approach two where the neural networks were evaluated against the logged data from the physical vehicle, the network performed in the range between 37.4% and 73.7%. This is a large span and is clearly worse than for the first approach. However, two major sources of error have been introduced and the results highlight the importance of having an accurate vehicle model, and using training data that matches the environments and scenarios that the physical vehicle could be exposed to.

Even though 37.4% is far from a desired accuracy, it shows that the process with obvious inaccuracies in the models, produced results almost 50% better than a process with no knowledge of the friction quotient while using a simplified vehicle dynamics model, relatively simple neural network models, and a low amount of data pre-processing.

The process evaluation in this thesis is a small insight into the expected benefits from changing physical difficulties of the testing and development process into programming and data science obstacles. The technologies used in the underlying work can be expanded upon, where the vehicle model needs improvement, and the possibilities of machine learning and data science have barely been utilised.

The process has been shown to be useful and provide value when it is used according to the assigned task. Using simulation, the process removes sources of error that are difficult to eliminate in the physical world, including the comparison to the true value of the variables, and the accessible way to test for a large combinatorial set of environments and scenarios. However, the process has also introduced sources of error to the friction estimation problem, including the digital vehicle model and the neural network. Thus, the benefits of the process needs to be taken advantage of for the process to show its strengths. If done correctly, a process that can reduce development time and costs, remove physical risk, and increase reliability has been presented. If the process is applied in an adverse way, the achieved result is the addition of a set of errors to the development process.

Bibliography

[1] Schofield, B. 2008. *Model-Based Vehicle Dynamics Control for Active Safety*. PhD Thesis TFRT-1083.
Department of Automatic Control. Lund University.

[2] Ahn, C.S. 2011. *Robust Estimation of Road Friction Coefficient for Vehicle Active Safety Systems*. PhD Thesis.
University of Michigan.

[3] Khaleghian, S., Emami, A. Taheri, S. 2017. *A technical survey on tire-road friction estimation*.
Friction 5:123

[4] Svendenius, J. 2007. *Tire Modeling and Friction Estimation*. PhD Thesis TFRT-1077.
Department of Automatic Control. Lund University.

[5] Boyuan, L., Haiping, D., Weihua, L. 2014. *Comparative study of vehicle tyre-road friction coefficient estimation with a novel cost-effective method*.
Vehicle System Dynamics: international journal of vehicle mechanics and mobility, vol. 52.

[6] Pisano, P., Goodwin, L., Rossetti, M. 2008. *U.S. highway crashes in adverse road weather conditions*.
24th Conference on International Interactive Information and Processing Systems for Meteorology, Oceanography and Hydrology.

[7] Lamm, R., Choueiri, E., Mailaender, T. 1990. *Comparison of Operating Speeds on Dry and Wet Pavements of Two-Lane Rural Highways*.
Transportation research record 1280.8

[8] Rämö, P., 1999. *Effects of weather-controlled variable speed limits and warning signs on driver behaviour*.
Transportation research record 1689.1

[9] Andrey, J. 2010. *Long-term trends in weather-related crash risks*.
Journal of Transport Geography, vol. 18.

[10] Gerla, M., Lee, E., Giovanni, P., Lee, U. 2014. *Internet of Vehicles: From Intelligent Grid to Autonomous Cars and Vehicular Clouds*.
2014 IEEE World Forum on Internet of Things (WF-IoT).

[11] NHTSA. 2019. *Motor Vehicle Traffic Fatalities and Fatality Rates, 1899-2017*.
`https://cdan.nhtsa.gov/tsftables/Fatalities\%20and\%20Fatality\%20Rates.pdf`

[12] Koopman, P., Wagner, M. 2016. *Challenges in Autonomous Vehicle Testing and Validation*.
SAE International Journal of Transportation Safety 4.1.

[13] De Winter, J., Van Leeuwen, P.M., Happee, R. 2012 *Advantages and disadvantages of driving simulators: A discussion*.
Proceedings of measuring behaviour. Vol. 2012.

[14] Wallén, L. 2001. *Dynamic Tyre Models in Adaptive Slip Control*. MSc Thesis TFRT-5673.
Department of Automatic Control. Lund University.

[15] Csáji, B.C. 2001. *Approximation with Artificial Neural Networks*. MSc Thesis. Faculty of Sciences, Etvs Lornd University.

[16] Nolte, M., Kister, N., Maurer, M. 2018. *Assessment of Deep Convolutional Neural Networks for Road Surface Classification*.
arXiv:1804.08872.

[17] Panahandeh, G., Ek, E., Mohammadiha, N. 2017. *Road Friction Estimation for Connected Vehicles using Supervised Machine Learning*.
arXiv:1709.05379.

[18] Ribeiro, A., Moutinho, A., Fioravanti, A., de Paiva, E. 2019. *Estimation of Tire-Road Friction for Road Vehicles: A Time Delay Neural Network Approach*.
arXiv:1908.00452.

[19] Schramm, D., Hiller, M., Bardini. R. 2017. *Vehicle Dynamics: Modeling and Simulation. Second edition*.
Springer-Verlag Berlin Heidelberg.

[20] Gillespie, T.D. 1992. *Fundamentals of Vehicle Dynamics*.
Society of automotive engineers.

[21] Jacobson, B., et al. 2017. *Vehicle Dynamics Compendium*.
21 University of Technology.

[22] Pacejka, H. 2005. *Tire and vehicle dynamics*.
Elsevier.

[23] Lee. A.T. 2018 *Vehicle simulation : perceptual fidelity in the design of virtual environments.*
Taylor & Francis, CRC Press.

[24] Satoshi, M. 2017. *Software Testing Design Techniques Used in Automated Vehicle Simulations.*
2017 IEEE International Conference on Software Testing, Verification, and Validation Workshops (ICSTW).

[25] Nvidia.
`https://www.nvidia.com/`.

[26] Unity 3D.
`https://unity.com`.

[27] Marks, R.E. 2007. *Validating Simulation Models: A General Framework and Four Applied Examples.*
Computational economics 30.3.

[28] Lawrence, S., Tsoi, AC., Back, A. 1996. *Function Approximation with Neural Networks and Local Methods: Bias, Variance and Smoothness.*
Australian Conference on Neural Networks. Vol. 1621.

[29] Keras API.
`https://keras.io`.

[30] Demuth, H.B. et al. 2014 *Neural network design.*
Martin Hagan.

[31] Buntine, W.L., Weigend, A.S. 1991. *Bayesian Back-Propagation.*
Complex systems 5.6.

[32] Pinkus, A. 1999. *Approximation theory of the MLP model in neural networks.*
Acta numerica 8.

[33] Hornik, K., Stinchcombe, M., White. H. 1989. *Multilayer feedforward networks are universal approximators.*
Neural networks 2.5.

[34] Cybenko, G. 1989. *Approximation by superposition of a sigmoidal function.*
Mathematics of control, signals and systems. 2.4.

[35] Benardos, P.G., Vosniakos, G-C. 2007. *Optimizing feedforward artificial neural network architecture.*
Engineering Applications of Artificial Intelligence 20.3.

[36] Karlik, B., Olgac, A.V. 2011. *Performance Analysis of Various Activation Functions in Generalized MLP Architectures of Neural Networks.*
International Journal of Artificial Intelligence and Expert Systems 1.4.

[37] Ioffe, S., Szegedy, C. 2015. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.*
arXiv:1502.03167.

[38] Srivastava, N. et al. 2014. *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*.
The journal of machine learning research 15.1.

[39] Kingma, D.P., Ba, J. 2014. *ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION*.
arXiv:1412.6980.

[40] Keras cross-entropy.
`https://keras.io/losses/`

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER'S THESIS

Date of issue
January 2020

Document Number
TFRT-6095

Author(s)
Jonas Karlsson

Supervisor
Simon Yngve, Combine
Anton Cervin, Dept. of Automatic Control, Lund University, Sweden
Bo Bernhardsson, Dept. of Automatic Control, Lund University, Sweden (examiner)

Title and subtitle
Road friction estimation using an artificial neural network in a simulated environment

Abstract

With the transition of responsibilities from the driver to the automated driving systems in vehicles, the systems need to have been tested for an extensive list of test scenarios as the passengers require high trustworthiness. The friction coefficient for the tyre-road friction is of high importance for the control of the vehicle but the coefficient is dependent on the physical complexity and nonlinear behaviour of tyres and is difficult to measure. Hence, testing is performed in controlled environments which limits the systems exposure to different testing scenarios.

The purpose of this thesis and the underlying work was to develop and evaluate a process for friction estimation using machine learning. The aim was to produce an estimation method using neural networks that are trained on data from a vehicle model implemented in a simulated environment using Unity 3D, which is a software platform for simulation and game development. The master thesis was produced at Combine Control Systems AB for Lund University in cooperation with National Electric Vehicle Sweden AB (NEVS).

Keywords
Friction estimation, vehicle dynamics, vehicle simulation, artificial neural networks

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-80

Recipient's notes

Security classification

http://www.control.lth.se/publications/