

Learning comfort objective from lane change demonstrations for optimal control

Stijn Staring

Thesis voorgedragen tot het behalen
van de graad van Master of Science
in de ingenieurswetenschappen:
werktuigkunde

Promotor:
Prof. dr. ir. Jan Swevers

Assessoren:
Prof. dr. ir. Herman Bruyninckx
Dr. ir. Bert Pluymers

Begeleider:
Dr. ir. Son Tong

© Copyright KU Leuven

Without written permission of the thesis supervisor and the author it is forbidden to reproduce or adapt in any form or by any means any part of this publication. Requests for obtaining the right to reproduce or utilize parts of this publication should be addressed to Faculteit Ingenieurswetenschappen, Kasteelpark Arenberg 1 bus 2200, B-3001 Heverlee, +32-16-321350.

A written permission of the thesis supervisor is also required to use the methods, products, schematics and programmes described in this work for industrial or commercial use, and for submitting this publication in scientific contests.

Zonder voorafgaande schriftelijke toestemming van zowel de promotor als de auteur is overnemen, kopiëren, gebruiken of realiseren van deze uitgave of gedeelten ervan verboden. Voor aanvragen tot of informatie i.v.m. het overnemen en/of gebruik en/of realisatie van gedeelten uit deze publicatie, wend u tot Faculteit Ingenieurswetenschappen, Kasteelpark Arenberg 1 bus 2200, B-3001 Heverlee, +32-16-321350.

Voorafgaande schriftelijke toestemming van de promotor is eveneens vereist voor het aanwenden van de in deze masterproef beschreven (originele) methoden, producten, schakelingen en programma's voor industrieel of commercieel nut en voor de inzending van deze publicatie ter deelname aan wetenschappelijke prijzen of wedstrijden.

Preface

I am very pleased to present my master thesis to complete my Master of Science in Mechanical Engineering. Conducting this research was an informative process in which I was able to apply the knowledge and skills that I gained during my studies. Writing my thesis and thus completing my studies would not have been possible without the support of many people for which I am grateful. First of all, I would like to thank my promoter Professor Swevers from the KU Leuven and my mentor, Dr. Tong from Siemens for the professional discussions and tips in order to improve my results. I also want to thank Flavia Acerbo, employee at Siemens, who answered my questions on several occasions. Last but not least, I would like to thank my family for their ongoing support during all phases of my studies. They have always been my biggest fans and I could not have done this without the opportunities they have given me.

Stijn Staring

Contents

Preface	i
Abstract	iv
Samenvatting	v
List of Figures and Tables	vi
List of Abbreviations and Symbols	ix
1 Introduction	1
1.1 Importance of topic	1
1.2 Problem formulation and link with previous studies	2
1.3 Thesis objective and structure	2
2 Background in optimal control problems	5
2.1 Optimal control problem (OCP)	5
2.2 Model predictive control	8
2.3 Conclusion	9
3 State of the art modelling of comfort	11
3.1 What is comfort during driving?	11
3.2 Inverse optimal control	13
3.3 Conclusion	16
4 Learning from ideal data	17
4.1 Non-linear bicycle model	18
4.2 Learning algorithm	20
4.3 Ideal data	27
4.4 Ideal data learning results	32
4.5 Conclusion	40
5 Learning with complex vehicle model	41
5.1 Flow of the algorithm	42
5.2 Tracking MPC	43
5.3 Learning results with the Amesim model	48
5.4 Conclusion	53
6 Enhanced weighting factor update	55
6.1 Formulation	55
6.2 Discussion	57

6.3	Conclusion	57
7	Conclusion	59
A	Jerk equations of the non-linear bicycle model	63
A.1	Equations	63
B	Results of the ideal data validation	65
B.1	Time limit	65
B.2	Amount of optimization points	70
C	Results of the averaging learning on 3 ideal datasets	75
D	Accuracy results of the tracking MPC	85
E	Results of learning with the 15 dof Amesim model	91
	Bibliography	101

Abstract

The human driver transforms from being an active traffic participant into a passive agent due to the developments in autonomous driving. Consequently a motorist involvement becomes limited to perceiving and rating decisions taken by the vehicle. In order to enhance user acceptance and to increase competitiveness, it is important that a notion of driver specific comfort is taken into account i.e. trade-offs made between different comfort contributors.

Therefore the objective of this research is to develop an algorithm making use of feature-based inverse optimal control which is able to learn driver specific experiences of comfort during a lane change, captured in weighting factors of a comfort objective function. Learning is done by looking at observations performed by the human driver because these are carried out based on an internal unknown objective function to generate comfortable paths. The retrieved objective function after learning, will be an approximation of the real one used by the human agent.

Comfort features, which are integrated kinematic vehicle signals in a formulation so that they determine a notion of comfort, are used in order to quantify similarity between observed and learned vehicle paths. The contribution of this thesis is to implement the theoretical idea of feature-based learning with practical relevant vehicle models e.g. a 15 degrees of freedom Amesim model. When the comfort objective is identified, it can be embedded in a motion planning formulation for autonomous driving.

In order to validate the developed algorithms for learning driver specific weighting factors, data is generated with a known objective function from which the weighting factors are tried to find back. Implementations are attained with a non-linear bicycle model and a more complex 15 degrees of freedom Amesim model. Learning is done making use of gradient descent, wherefore an alternative method is proposed based on an optimized weighting factor update.

Results were retrieved that show that the lateral weighting factors, that determine a lane change, were accurately learned. Due to a good match between learned and observed feature values, the observed kinematic vehicle signals were also adequately learned. This research was performed with the help of Siemens Digital Industries Software.

Samenvatting

De menselijke bestuurder transformeert van een actieve naar een passieve deelnemer van het verkeer door ontwikkelingen die worden verwezenlijkt in onderzoek naar zelfrijdende auto's. Dit heeft als consequentie dat de rol van de automobilist beperkt wordt tot het waarnemen en beoordelen van handelingen, uitgevoerd door een zelfrijdend voertuig. Om consumentenacceptatie te bevorderen en de competitiviteit van het product te verhogen, is het belangrijk dat specifiek bestuurderscomfort gemonitoreerd kan worden. Dit wil concreet zeggen, de afwegingen die gemaakt worden tussen de verschillende variabelen die bijdragen tot comfort.

Daarom is onderzoek uitgevoerd met als doel een algoritme te ontwikkelen dat gebruikt maakt van 'feature-based inverse optimal control' dat in staat is om specifieke comfort bevindingen, bevat in de gewichten van een comfortobjectieffunctie, te leren. Het manuever gekozen om dit te doen is een rijbaanwisseling. Het algoritme moet leren door te kijken naar observaties van het manuever uitgevoerd door een menselijke bestuurder, omdat deze gebaseerd zijn op een interne, onbekende objectieffunctie. De objectieffunctie verkregen na het leerproces, benadert de werkelijke interne functie die gebruikt werd door de menselijke bestuurder.

Comfortfeatures, wat geïntegreerde voertuigsignalen zijn in een formulering zodat ze een notie van comfort bevatten, worden gebruikt om de overeenkomst uit te drukken tussen het geobserveerde en geleerde voertuigtraject. De bijdrage van deze thesis is het theoretische idee van feature gebaseerd leren te gebruiken met praktisch relevant voertuigmodellen, zoals het 15 vrijheidsgraden bevattende Amesim model. Wanneer het comfortobjectief is geïdentificeerd, kan het gebruikt worden om trajecten te plannen voor een zelfrijdend voertuig.

Om de ontwikkelde algoritmes te valideren, is data gegenereerd met een op voorhand gekend objectief dat zal geprobeerd worden teruggevonden. Implementaties zijn verwezenlijkt die gebruik maken van een 'non-linear bicycle model' en een complex Amesim model. Tijdens het leerproces is de gradiënt afdaling methode gebruikt, waarvoor een alternatief wordt voorgesteld dat gebaseerd is op een optimalisatie van de wegingsfactorupdate. Resultaten werden bekomen die aantonen dat de laterale wegingsfactoren die bepalen hoe een rijbaanwisseling wordt uitgevoerd, accuraat geleerd kunnen worden. Door een goede match tussen de geleerde en geobserveerde featurewaarden, werden de geobserveerde voertuigsignalen accuraat geleerd. Dit onderzoek is uitgevoerd met de hulp van Siemens Digital Industries Software.

List of Figures and Tables

List of Figures

1.1	Concept visualization of autonomous driving. (source: [7])	1
1.2	Example of a lane change which is the investigated maneuver in this thesis.	3
2.1	Overview of different discretization methods.	6
2.2	Schematic view of the time shooting approaches (left: multiple shooting, right: single shooting).	8
2.3	Visualization of the optimal control problem solved in one iteration of the MPC (Source: [15]).	9
2.4	Visualization of the application of the first step of the calculated control signal during one iteration of the MPC (Source:[15]).	9
3.1	Overview of comfort parameters in autonomous vehicle with old parameters (blue) and new ones (red).(Source: [8])	12
4.1	Non-linear bicycle model (Source: [20]).	17
4.2	Basic flow of the inverse optimal control algorithm.	21
4.3	A comparison between the numerical jerk (left) based on Eq. (4.23) and Eq. (4.1) and the analytical jerk (right) based on appendix A and Eq. (4.2).	28
4.4	Lateral acceleration during a lane change $V_0 : 25.00 \frac{m}{s}$ and $L : 6.94m$ generated with the 15 dof Amesim model.	31
4.5	slip angle during lane change $V_0 : 25.00 \frac{m}{s}$ and $L : 6.94m$ generated with the 15 dof Amesim model.(blue:front,red:rear)	31
4.6	Observed, initial and learned paths for 3 different observed lane changes generated with common underlying objective function.	35
4.7	The evolvement of the average f_{rel} over the learning iterations.	35
4.8	The different outputted weighting factors over the learning iterations.	36
4.9	Flow of the conflict method as part of the basic flow diagram of Figure 4.2	37
4.10	The evolvement of f_{rel} of dataset $V_0 : 22.22 \frac{m}{s}$, $L : 3.47 m$ over the learning iterations.	38

4.11 This figure shows the averaged, relative error given by $\frac{100 \cdot \sum_{n=1}^{ND} 1 - f_{rel,i} }{ND}$ of the matching of the feature values for the three lateral features. (3 datasets: blue, 7 datasets: orange)	39
4.12 This figure shows the averaged, relative error given by $\frac{100 \cdot \sum_{n=1}^{ND} 1 - f_{rel,i} }{ND}$ of the matching of the feature values of the three lateral features. (Averaging method: blue, conflict method: orange)	40
5.1 The 15 dof amesim model with as inputs the amount of throttle, braking and steerwheelangle.	41
5.2 The flow of learning with the Amesim model.	42
5.3 This figure shows hows the tracking MPC is connected with the Amesim model using different sampling times. (red: $T_s : 0.01\text{ s}$ and green: $T_{MPC} : 0.1\text{ s}$)	46
5.4 This figure shows the tracking result of the reference (red) by the Amesim model (blue).	47
5.5 This figure shows the obtained oscillating signals for the jerk when a piecewise signal of δ_s and t_r is applied.	48
5.6 This figure shows the observed paths, the initial solution retrieved with as weighting factors an all one vector and the learned solution.	51
5.7 This figure shows the error made between the observed and learned paths.	51
5.8 This figure shows the averaged f_{rel} during learning of five datasets.	52
5.9 This figure shows the observed jerk signals, the initial solution retrieved with as weighting factors an all-one vector and the learned solution.	52
C.1 This figure shows the amount of throttle and the angle of the front wheel of the bicycle model during the lane change maneuvres.	78
C.2 This figure shows the amount of first derivative of throttle and the first derivative of the angle of the front wheel of the bicycle model is given as input during the lane change maneuvres.	80
C.3 The convergence of f_{rel} towards one during the learning iterations.	81
C.4 The gradient $\frac{\partial F_{diff}}{\partial \theta}$ estimated by $\mathbf{f}_{obs} - \mathbf{f}_{learned}$ towards zero during the different learning iterations.	81
C.5 The learned weights during the different learning iterations.	82
C.6 The difference of θ with respect to one used in the previous iteration.	82
C.7 This figure shows which case of the three available in the RPROP algorithm is chosen during the learning iterations.	83
E.1 This figure shows the amount of throttle and the angle of the front wheel of the bicycle model during the lane change maneuvres.	96
E.2 This figure shows the amount of first derivative of throttle and the first derivative of the angle of the front wheel of the bicycle model is given as input during the lane change maneuvres.	97
E.3 The convergence of f_{rel} towards one during the learning iterations.	97
E.4 The gradient $\frac{\partial F}{\partial \theta}$ estimated by $\mathbf{F}_{obs} - \mathbf{F}(r_{expected})$ towards zero during the different learning iterations.	98

LIST OF FIGURES AND TABLES

E.5	The learned weighting factors during the different learning iterations.	98
E.6	The difference of θ with respect to one used in the previous iteration.	99
E.7	This figure shows which case of the three available in the RPROP algorithm is chosen during the learning iterations.	99

List of Tables

4.1	Used vehicle model parameters.	20
4.2	Overview of normalization factors.	26
4.3	This table shows the retrieved feature values using two different initial guesses in Eq. (4.8).	29
4.4	This table shows the retrieved feature values using different time limits in Eq. (4.8) and using as vehicle model Eq. (4.2).	29
4.5	This table shows the retrieved feature values using different amount of control point N in Eq. (4.8) and using as vehicle model Eq. (4.2).	30
4.6	This table shows the weighting factors learned from the dataset $V_0 : 22.22 \frac{m}{s} - L : 3.47 m$ and the associated f_{rel} at a two different amount of iterations.	33
4.7	This table shows the f_{rel} for each individual dataset at convergence using the average method.	34
4.8	This table shows the f_{rel} for each individual dataset at convergence using the conflict method.	38
5.1	This table shows the feature values of a lane change $V_0 : 22.22 \frac{m}{s}$, $L : 3.47 m$ for respectively the Bicycle and Amesim model.	43
5.2	Overview of the tuning parameters used in the objective of Eq. (5.1).	44
5.3	This table shows which lane changes were used during the four different simulations.	49
5.4	This table shows the different features for the individual datasets used.	49
5.5	This table shows the weighting factors learned for the different simulations that start from an all-one vector and uses as chosen weighting factors $[4.0, 5.0, 1.0, 6.0, 1.0, 2.0]$	50
5.6	This table shows the averaged f_{rel} at convergence.	50

List of Abbreviations and Symbols

Abbreviations

MS	Multiple shooting
SS	Single shooting
Gs	Gsteeringfactor
OCP	Optimal control problem
MPC	Model predictive control
ODE	Ordinary differential equation
ND	Number of datasets
dof	Degrees of freedom

Symbols

	Vectors are printed in bold.
N	The number of control actions.
N_{MPC}	Amount of control points in the control horizon of Eq. (5.1).
m	Amount of observed lane changes.
Z	Amount of features.
\mathbf{r}	The 2D path followed during a lane change maneuver.
$F(\mathbf{r})$	Feature vector $\in \mathbb{R}^{Z \times 1}$ with on its entries the different feature values for the learned path \mathbf{r} .
$F_{obs}(\mathbf{r}_{obs})$	Observed feature vector $\in \mathbb{R}^{Z \times 1}$ with on its entries the different feature values for the observed path \mathbf{r}_{obs} .
F_{diff}	The difference between the observed and expected feature vector: $F_{obs}(\mathbf{r}_{obs}) - F(\mathbf{r})$.
V_0	Longitudinal speed at the start of the lane change.
V_{des}	Desired longitudinal speed during a lane change. Taken equal to V_0 during this thesis.
L	Desired lateral distance travelled at the end of the lane change.

LIST OF ABBREVIATIONS AND SYMBOLS

θ	Weights that are learned in the objective function Eq. (4.12).
f_{obs}	Feature vector calculated from the observed lane change path according to Eq. (4.2.2).
$f_{learned}$	Feature vector as calculated according to Eq. (4.2.2) from the lane change path, following out of solving Eq. (4.8).
f_{rel}	Relative feature vector defined as the element wise division of the learned feature vector by the observed one i.e. $\frac{f_{learned}}{f_{obs}}$.
t_r	Amount of normalized throttle given.
δ	Angle of the front wheel of the bicycle model.
δ_s	The steer wheel angle related to δ by G_s according to $\delta_s = G_s \cdot \delta$.
Δu	Update value used in the RPROP algorithm in section 3.2.2.
q	State vector of the OCP defined in section 2.1.
u	Control vector of the OCP defined in section 2.1.

Chapter 1

Introduction

1.1 Importance of topic

”Society expects autonomous vehicles to be held to a higher standard than human drivers.” [18] This quote sets the tone of the technology in autonomous driving. In order to be accepted by the public, autonomous vehicles should perform at least as reliable as conventional human drivers on parameters such as safety. Despite widespread research on self-driving vehicles, the acceptance by users stays only limited [3]. Research shows that purchase behaviour of customers can be directly linked with comfort. To gain more trust by the public it is clear that the challenge of making autonomous vehicles as comfortable as possible, should be tackled. This immediately leads to question: what is exactly comfort during driving?

Driving comfort is a personal experience and also depends on the emotional state of the driver. Therefore more than one driving style for an autonomous vehicle should be identified [23]. The state of the driver can be communicated to the vehicle at the start of each ride and different driving styles can be obtained by changing the parameters in a motion planning algorithm.



Figure 1.1: Concept visualization of autonomous driving. (source: [7])

1.2 Problem formulation and link with previous studies

In order to identify specific comfort preferences of the driver that are quantified by parameters, the vehicle should be able to learn by demonstration [12]. Despite that each driver has its own preferences, they are based on common comfort criteria where different trade-offs are made. For example, some drivers prefer a more aggressive driving style than others. This will manifest in a different set of parameters than the ones attained for a defensive driver for the same comfort criteria. The comfort criteria will later in this thesis be translated into an objective function where different weighting factors are used in order to quantify different comfort trade-offs made. In literature, this approach is called inverse optimal control or inverse reinforcement learning because it is learning the objective function for an optimal control problem.

In order to learn the weighting factors which can be used to distinguish different drivers, research about the common notion of comfort is necessary. Passenger surveys in public road transport about carsickness as described in paper [21], have identified lateral acceleration as the primarily responsible for motion sickness. It is explained that drive style is a main factor to influence the amount of sickness and it was found that sickness is higher when drivers drove with a higher average magnitude of fore-and-aft and lateral motion. These effects were far more significant than the effect of vertical vibrations. There is also a consensus reached about the contribution of continuous trajectories to the prevention of motion sickness and the natural feel of paths [8]. This means that higher order kinematic variables like accelerations and jerks also should be considered when measuring comfort. Chapter 3 will give a more detailed description on the literature study conducted in order to investigate the state of the art modelling of comfort. It will also discuss the theory of inverse optimal control.

1.3 Thesis objective and structure

The goal of this thesis is to build further on research of learning by demonstration as explained in paper [12] and to refine this idea by making use of more realistic vehicle models. Concretely, the thesis is focussed on the implementation and validation of an algorithm that learns weighting factors in a comfort objective function which can explain observed driver data.

The learning process is done offline. After the different weighting factors are identified, the learned objective can be used to plan paths which can be followed by an autonomous vehicle, making use of an online tracking MPC.

The data used as observations is generated by simulations where it is assumed that the vehicle is driving on a straight road at an initial speed between $80 \frac{\text{km}}{\text{hours}}$ and $100 \frac{\text{km}}{\text{hours}}$. The maneuver investigated is a lane change maneuver as can be seen in

Figure 1.2. In order to generate data, the non-linear bicycle and a more realistic, 15 dof of freedom Amesim model were used.

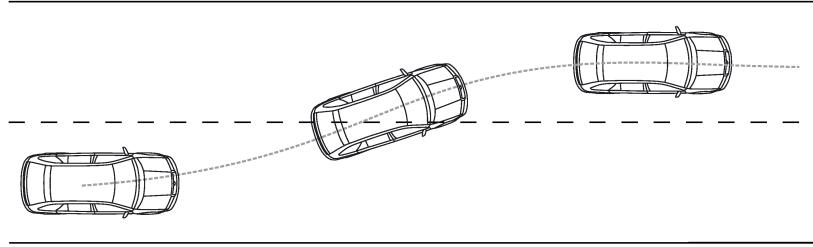


Figure 1.2: Example of a lane change which is the investigated maneuver in this thesis.

The execution of this research was supported by "Siemens Digital Industries Software" located in Leuven, which enabled a direct link with reality. Software was made available i.e. Simcenter Amesim.

The structure of this thesis is as follows. First, the reader is made acquainted with the optimal control concept in chapter 2. The meaning is explained in section 2.1 and section 2.2 gives an introduction of MPC. Next, chapter 3 discusses the state of the art modelling of comfort. This chapter is made out of two parts, section 3.1 answers the question posed in section 1.1, explaining the meaning of comfort during driving. Section 3.2 concerns a discussion on the inverse optimal control concept. Chapter 4 gives insight in learning from ideal data. The non-linear bicycle model is discussed in section 4.1, whereafter the formulation of the learning algorithm follows in section 4.2. Next, ideal data generation is analysed and validated in section 4.3. Subsequently, different methods for learning from multiple datasets are discussed and results were analysed in section 4.4. In chapter 5, the non-linear bicycle model will be substituted by a more complex 15 degrees of freedom Amesim model. First the flow of the learning algorithm is discussed in section 5.1. Next the developed tracking MPC needed, is presented in section 5.2. Section 5.3 gives an overview of the learning results. In chapter 6 the theoretical concept of an enhanced weight factor update method is proposed. First the formulation is analysed in section 6.1, whereafter a discussion follows in section 6.2. Finally, chapter 7 gives a general conclusion of this thesis.

Chapter 2

Background in optimal control problems

This chapter gives some background information on the theory behind optimal control problems (OCP) in section 2.1. After a global introduction and a discussion about discretization and time shooting methods, this chapter introduces the concept of model predictive control (MPC) in section 2.2.

2.1 Optimal control problem (OCP)

An optimal control problem determines the desired inputs and corresponding state trajectories to change the system from an initial state to a desired final state in an optimal way while satisfying input and state constraints [13]

$$\begin{aligned} \min_{\mathbf{q}, \mathbf{u}} \quad & \int_0^T l(\mathbf{q}(t), \mathbf{u}(t)) dt + E(\mathbf{q}(T)) \\ \text{s.t.} \quad & \dot{\mathbf{q}}(t) = \mathbf{f}(\mathbf{q}(t), \mathbf{u}(t)) \\ & \mathbf{q}(0) = \mathbf{q}_0, \quad \mathbf{q}(T) = \mathbf{q}_T \\ & \mathbf{q}(t) \in Q, \quad \mathbf{u}(t) \in U, \quad t \in [0, T] \end{aligned} \quad (2.1)$$

\mathbf{q} is called the state vector and contains all the states of the system. \mathbf{u} is the control vector containing the controls. In theory the amount of states of a system can be infinite, but in order to sufficiently model a system, only a well chosen set of states is needed. Typical controls for a vehicle are steer wheel angle and the amount of throttle, which can be directly linked to the amount of propulsion force. Higher order controls are also possible.

The objective function l of the optimization problem Eq. (2.1) is integrating over the desired control horizon T . In the objective function it is indicated what should be minimized and this is in function of the different states and controls. $E(\mathbf{q}(T))$ describes the terminal cost and the objective is reduced when the system comes closer to a predefined final state at the end of the control horizon. The dynamics of the system are modelled by an explicit ordinary differential equation $\dot{\mathbf{q}}(t) = \mathbf{f}(\mathbf{q}(t), \mathbf{u}(t))$

and the evolution of the vehicle states are retrieved by integrating this ODE over the control horizon. Q and U represent the feasible space wherein solutions for the OCP can be sought. It is worth noting that the control horizon time T itself can be an optimization variable. What comes out of an OCP indicates what states will be visited by the system and which controls have to be applied in order to optimize the associated objective function with respect to predefined constraints [16].

There is a difference between soft and hard constraints. A hard constraint directly demarcates the feasible solution spaces $\mathbf{q}(t) \in Q$, $\mathbf{u}(t) \in U$. A soft constraint is added in the objective function l and will contribute to a more optimal solution when it is better fulfilled. Also when a soft constraint is not met, this can be an optimal solution which is not the case for a hard constraint [22].

2.1.1 Time discretization

The optimal control problem Eq. (2.1) is continuous in time. This means that it has infinite dimensions, but to be able to run the optimization problem on digital systems, there is need for discretization. Several ways to do this are summarized in Figure Eq. (2.1) [10].

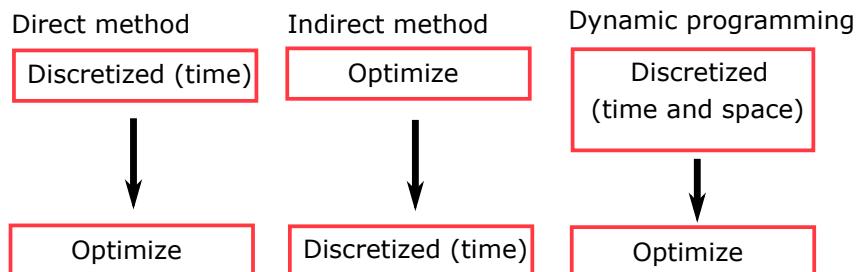


Figure 2.1: Overview of different discretization methods.

Since direct methods are best suited to solve practically relevant OCPs [13], this thesis has chosen for to use this method. To implement the discretization of time when using the Direct method, a time shooting approach is used.

2.1.2 Time shooting

A shooting approach makes use of a time grid. Time will be sampled and on every time instant the optimal control problem is assessed. On these discrete points constraints will not be violated, but there are no limits on the amount of violation between the different time samples. In order to reduce the amount of constraint violation the sample rate should be taken high enough, while bearing in mind the extra optimization variables introduces a higher calculation load. Another approach to take constraint violation into account, is using spline based optimization formulations [13]. Control changes are only allowed on the different discrete time instances

2.1. Optimal control problem (OCP)

which leads to a piecewise control signal.

Two different shooting approaches exist:

1. Multiple shooting (MS)

During multiple shooting, every new time sample new optimization variables are defined for the states and controls. The blue bars in Figure 2.2 (left) indicate the control signal applied to the system. The red dots are system states that are defined as optimization variables and are not constant during a time interval ΔT . In order to make the connection between states at time t_i and t_{i+1} , time integration is embedded in \mathbf{F} according to $\mathbf{q}^{k+1} = \mathbf{F}(\mathbf{q}^k, \mathbf{u}^k)$. These connections are put as constraints in the optimization and are called in literature 'path closing constraints' [10]. Eq. (2.2) presents the discretized version of Eq. (2.1) when using the multiple shooting approach

$$\begin{aligned} \min_{\mathbf{q}(.), \mathbf{u}(.)} \quad & \sum_{k=0}^{N-1} \frac{l(\mathbf{q}^k, \mathbf{u}^k) + l(\mathbf{q}^{k+1}, \mathbf{u}^{k+1})}{2} \cdot \Delta T + E(\mathbf{q}^N) \\ \text{s.t.} \quad & \mathbf{q}^{k+1} = \mathbf{F}(\mathbf{q}^k, \mathbf{u}^k) \quad k = [0, \dots, N-1] \\ & \mathbf{q}^0 = \mathbf{q}_{measured} \\ & \mathbf{q}^k \in Q, \quad k = [0, \dots, N]. \\ & \mathbf{u}^k \in U \quad k = [0, \dots, N-1] \\ & \mathbf{q}^N \in Q_f, \quad N \in \mathbb{N} \end{aligned} \quad (2.2)$$

2. Single shooting (SS)

In the single shooting or sequential approach which is visualized in Figure 2.2 (right), only the first state and the controls are taken as optimization variables. Other states during the time horizon are derived from the initial state and the applied control. This is achieved by substituting the states during the control horizon by the integration results from the initial state. The mathematical formulation of this is shown in Eq. (2.3). In Figure 2.2 (right) are the states that result from integration indicated in green [10]

$$\begin{aligned} \mathbf{q}^1 &= \mathbf{F}(\mathbf{q}^0, \mathbf{u}^0) \\ \mathbf{q}^2 &= \mathbf{F}(\mathbf{F}(\mathbf{q}^0, \mathbf{u}^0), \mathbf{u}^1) \\ \mathbf{q}^3 &= \mathbf{F}(\mathbf{F}(\mathbf{F}(\mathbf{q}^0, \mathbf{u}^0), \mathbf{u}^1), \mathbf{u}^2) \\ &\dots \end{aligned} \quad (2.3)$$

This thesis has opted for the multiple shooting approach and a Runge-Kutta numerical integration scheme is implemented in order to define the path closing constraints. Runge-Kutta is an explicit integration scheme which has a higher calculation cost

2. BACKGROUND IN OPTIMAL CONTROL PROBLEMS

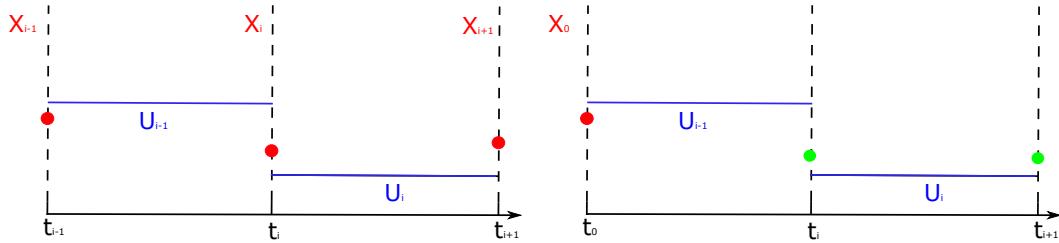


Figure 2.2: Schematic view of the time shooting approaches (left: multiple shooting, right: single shooting).

than a standard Euler scheme but is more reliable for non-linear systems and has a higher stability with respect to the chosen time step [13].

It can be noted that the difference between the two shooting approaches is that 'Multiple shooting'(MS) will lead to a larger Hessian of the objective and a larger Jacobian of the constraints. The reason for this is the introduction of more optimization variables. The advantage of MS is then again that these matrices are more sparse because a certain state is only dependent on the previous one and a certain control. Therefore these matrices can be used more efficiently with respect to calculation load. In single shooting every state depends on the begin state and different controls which give smaller, more densely populated matrices which are more calculation expensive [10].

2.2 Model predictive control

In slow changing environments as for example a chemical plant, MPC is already a mature control technique. More recently this technology also made its introduction in controlling systems with higher dynamics e.g. vehicle control, due to an increase of computational power and the use of more efficient algorithms [13]. In the next section a short introduction of the formulation is given.

During MPC, optimizations are solved in a loop to be able to notice disturbances, changing environments and model-plant mismatch. Therefore it makes use of a moving control horizon¹. Every iteration an OCP is solved, starting from the current state and using a certain control horizon. From the resulting control signals only the first control value is applied for one time interval of the finite control horizon $N \cdot T_s$ as is visualized in Figures 2.3 and 2.4.

The decision on the amount of samples N that are used in the control horizon is based on a trade-off between higher accuracy and calculation effort [20, 13]. In Figure 2.3 the solving of an OCP during one MPC iteration on time sample $t + 1$ is depicted. Figure 2.4 shows that only the first control of the computed control signal will be applied. This induces the system to change its current state. Next a

¹There are also other implementations e.g. shrinking time horizon, where the control horizon gets smaller after each iteration.

new OCP can be solved. A single OCP-iteration only takes the model of the system into account to calculate the control signals. The iterative way of solving makes the MPC able to deal with unexpected acquired states through feedback that is given by each new current state. As stated earlier, the downside of this approach is a bigger computational load, which makes efficiently written software a necessity [15].

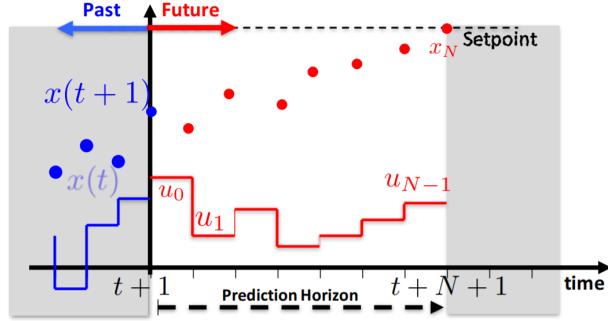


Figure 2.3: Visualization of the optimal control problem solved in one iteration of the MPC (Source: [15]).

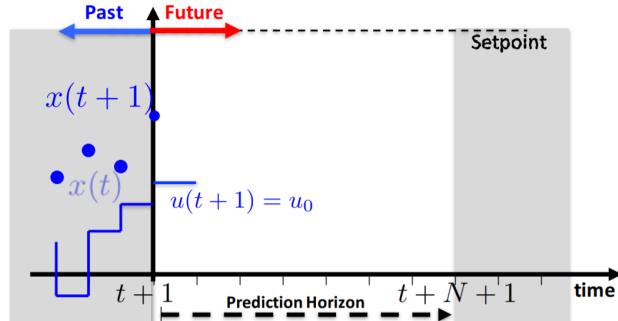


Figure 2.4: Visualization of the application of the first step of the calculated control signal during one iteration of the MPC (Source:[15]).

2.3 Conclusion

In this chapter the reader was made acquainted with concepts of optimal control and model predictive control, which will be used in this thesis. First, a continuous OCP was considered and options how to discretize it. The option to discretize time that fits this thesis best, is the "Direct method" combined with a multiple shooting approach. It is ensured that the time step is chosen small enough to avoid constraint violation.

Chapter 3

State of the art modelling of comfort

As discussed in the introduction, the goal of this thesis is to implement a method to capture personal experience of comfort during driving. This will be done by using inverse optimal control where the weighting factors in the objective function are learned from demonstration. A literature study was needed to understand how comfort is defined and how the inverse optimal control method can be implemented.

This chapter is structured as follows. First, the definition of comfortable driving is looked into in section 3.1. Section 3.2 concerns a discussion of the inverse optimal control concept used.

3.1 What is comfort during driving?

In the US patent of L. Daniel [6], the idea is to assess the amount of comfort by calculating a value for carsickness. This value is calculated by a weighted sum of sway motion, surge motion and heave motion of the vehicle. These motions are being directly calculated from the lateral acceleration, fore-aft acceleration and the vertical acceleration of the vehicle.

In the paper 'Investigating ride comfort measures in autonomous cars' [8], it is explained that due to the introduction of autonomous vehicles there will be an other perception of comfort. Figure 3.1 indicates in blue the claimed traditional comfort factors and in red the new ones that additionally have to be taken into account when driving in autonomous vehicles. Concretely this can be translated into the preference of smooth trajectories and low lateral motions when the roads are assumed to be sufficiently smooth. A hypothesis taken, is that motion sickness will be more prominent in autonomous driving due to the loss of control. It is also argued that the distance to an obstacle is contributing to a comfortable feeling.

3. STATE OF THE ART MODELLING OF COMFORT



Figure 3.1: Overview of comfort parameters in autonomous vehicle with old parameters (blue) and new ones (red).(Source: [8])

In 'Analysis of Driving Style Preference in Automated Driving' [4], three studies were completed to capture the definition of comfortable driving in autonomous vehicles. During the first study, human drivers drove manually with their own driving style a set of maneuvers and in this data, there was sought after relevant metrics that could be used to define comfortable driving. The results of this research was that accelerations play a key role in comfort but are not the only factor [4].

The different comfort metrics found were:

- lateral and longitudinal acceleration;
- lateral and longitudinal jerk;
- quickness of the maneuver;
- headway distance;

Quickness of the maneuver indicates that it is received as comfortable when the vehicle gives a good response and reaches the desired lateral displacement faster. Additionally comfortable driving as assessed in [4], can be summarized as sufficiently smooth trajectories and keeping enough headway distance in order to have a feeling of control and safety. These results suggest that when an algorithm is evaluating comfort, a notion of the surrounding traffic should be present. It was found when the traffic density on the road is higher, the driver is more tolerant towards less smooth driving behaviour e.g. to be able to insert in a busy lane. In this case higher comfort can be attained if the driver has the feeling of a fast response of the vehicle, translating in early peaks of acceleration. Vertical vibrations are outside the scope when roads are assumed sufficiently smooth.

In a second study the main metrics found from study one are rated making use of a questionnaire, asking about the amount of comfort retrieved. "Out of this it followed that accelerations are again playing a key factor", according to paper [4]. For lane changes it could be concluded that maneuvers with a small lateral acceleration and early perceivable onset were more comfortable.

That also jerk plays an important role in the attained amount of comfort, is confirmed by paper [9] where it is stated that: "Jerk has been shown to elicit a stronger influence on comfort than acceleration".

3.2 Inverse optimal control

Because every human has its own driving style, it is a cumbersome task to tune the weighting factors of an objective function for each individual in order to model a personal perception of comfort. In paper [17] it is showed that manual tuned weighting factors will besides lead to suboptimal solutions in comparison with learned ones from data. Inverse optimal control is the activity of learning an agent's objective function from expert demonstrations [2].

The goal of the learning algorithm explained in this thesis is to derive the weighting factors θ_j of different linearly combined comfort criteria f_j , embedded in the cost function J . As the observed data is explained better during learning, it is assumed that J is getting closer to the inner comfort objective of the individual driver. However, it should be noted that the driver takes unknowingly a lot of comfort criteria into account and they are not all linearly related as is suggested by Eq. (3.1). Therefore the comfort cost function in Eq. (3.1), will always be an approximation of reality. Although as discussed in paper [12], it is possible to capture the magnitude of the main contributors of comfort experienced by the users

$$J = \sum_{j=1}^Z \theta_j \cdot f_j. \quad (3.1)$$

In order to create a generative model that creates vehicle paths \mathbf{r}_i with equivalent kinematic characteristics as the path that was observed $\mathbf{r}_{obs,i}$, a feature-based inverse optimal control is applied, with $i \in [1...m]$ and m the amount of observed trajectories [12, 1]. A feature is encoding relevant kinematic properties onto a scalar value and the difference between the demonstrated and calculated feature values, give an indication about the similarity of the kinematic signals i.e. displacement, velocity, acceleration and jerk.

3.2.1 Feature-based inverse optimal control

A feature value maps a signal onto a scalar and encapsulates a comfort criteria. The higher the scalar value the more discomfort is experienced by the driver. An example is given by Eq. (3.2) that measures the amount of accelerations in a maneuver

$$f_j : \mathbf{r} \rightarrow f_j(\mathbf{r}) = \int_0^T a_x(t)^2 + a_y(t)^2 dt. \quad (3.2)$$

The different feature values are collected in a feature vector $\mathbf{F} \in \mathbb{R}^Z$ with on its entries the different feature values f_j . The path that is followed by the vehicle is represented by \mathbf{r} .

The human driver is not a deterministic agent and is modelled by a stochastic distribution $p(\mathbf{r}|\boldsymbol{\theta})$. For certain weighting factors $\boldsymbol{\theta} \in \mathbb{R}^Z$ a path \mathbf{r} is produced as being a sample of a stochastic distribution. The distribution chosen, is the distribution of maximum entropy Eq. (3.3) as suggested in paper [5]. When using the maximum entropy distribution the amount of prior information embedded in the distribution is minimized. It is also stated that many physical systems will move towards a maximal entropy configuration over time

$$p(\mathbf{r}|\boldsymbol{\theta}) = \exp(-\boldsymbol{\theta}^T \cdot \mathbf{F}(\mathbf{r})). \quad (3.3)$$

Eq. (3.3) can be interpreted as a cost function $\boldsymbol{\theta}^T \mathbf{F}(\mathbf{r})$ where agents are exponentially more likely to select trajectories with a lower cost which is felt as more comfortable [12].

In order to explain the observed feature vector \mathbf{F}_{obs} , the weighting factors $\boldsymbol{\theta}$ need to be found that match the expected feature vector $\mathbf{F}(\mathbf{r}_{expected})$ with \mathbf{F}_{obs} . The observed feature vector $\mathbf{F}_{obs} \in \mathbb{R}^Z$ has on its entries the different feature values $f_{obs,j}$. $\mathbf{r}_{expected}$ is defined as $E(p(\mathbf{r}|\boldsymbol{\theta}))$. To go towards a match with the observed feature vector, a gradient descent method can be used with as estimation of the gradient $\frac{\partial \mathbf{F}_{diff}}{\partial \boldsymbol{\theta}}$ by $\mathbf{F}_{obs} - \mathbf{F}(\mathbf{r}_{expected})$ and with \mathbf{F}_{diff} the difference between the feature vectors [5, 11]. Only the sign of the gradient is used and there are two directions for every weighting factor update: increase or decrease. Therefore an intuitive explanation exist for using $\mathbf{F}_{obs} - \mathbf{F}(\mathbf{r}_{expected})$ as estimation for the gradient. The weighting factor is increased when the expected feature is higher than the observed one, which means that the associated comfort feature will be punished more severely in Eq. (3.3) and paths with a higher value of that comfort feature become less likely. The weighting factor will be decreased when the expected feature is lower than the observed one [12]. Eq. (3.4) summarizes the gradient descent method with α the step size taken in the direction of descent

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \alpha \frac{\partial \mathbf{F}_{diff}}{\partial \boldsymbol{\theta}}^k. \quad (3.4)$$

When $\boldsymbol{\theta}_{optimal}$ is found the gradient is minimized and the feature vectors will match as closely as possible. In order to achieve this, the problem remains how to retrieve $\mathbf{F}(\mathbf{r}_{expected})$. In order to calculate $\mathbf{F}(\mathbf{r}_{expected})$ a Hamiltonian Markov chain Monte

Carlo stochastic distribution sampling method was used in paper [11]. In paper [12] a simplified and less calculation demanding approach is proposed, where it is assumed that the expected path is also the one that is been assessed as the most comfortable by the human driver. The path that is perceived as the most comfortable can be estimated by minimizing Eq. (3.1) for certain weighting factors. Equation 3.5 summarizes the method proposed by paper [12]

$$\mathbf{r}_{expected} = \underset{\mathbf{r}}{\operatorname{argmax}} p(\mathbf{r}|\boldsymbol{\theta}) = \underset{\mathbf{r}}{\operatorname{argmin}} \boldsymbol{\theta}^T \cdot \mathbf{F}(\mathbf{r}), \quad (3.5a)$$

$$\frac{\partial \mathbf{F}_{diff}}{\partial \boldsymbol{\theta}} = \mathbf{F}_{obs} - \mathbf{F}(\mathbf{r}_{expected}). \quad (3.5b)$$

3.2.2 RPROP algorithm

From reference [16] it is known that not every step size of the gradient is leading to convergence towards a minimum. When the step size is too small, it will take a long time to converge. However when it is chosen too big, cycling behaviour between limit points occurs. In order to avoid this unwanted behaviour, a method is needed to change the step size taken during the course of the algorithm. For this the resilient backpropagation algorithm (RPROP) [19] is used as it was first proposed by Martin Riedmiller and Heinrich Braun in 1993.

The main advantage when using RPROP is that the size of the gradient is not blurring the update value of the weighting factors. The update value Δu is solely dependent on the sign of the current gradient, the sign of the gradient in the previous iteration and the update value in the previous iteration. The three possible cases for the update value during iteration t are given by:

$$\Delta u_j^t = \begin{cases} \eta^+ \cdot \Delta u_j^{t-1}, & \text{if } \frac{\partial f_j}{\partial \theta_j}^t \cdot \frac{\partial f_j}{\partial \theta_j}^{t-1} > 0 \\ \eta^- \cdot \Delta u_j^{t-1}, & \text{if } \frac{\partial f_j}{\partial \theta_j}^t \cdot \frac{\partial f_j}{\partial \theta_j}^{t-1} < 0 \\ \Delta u_j^{t-1}, & \text{if } \frac{\partial f_j}{\partial \theta_j}^t \cdot \frac{\partial f_j}{\partial \theta_j}^{t-1} = 0 \end{cases}. \quad (3.6)$$

$$0 < \eta^- < 1 < \eta^+ \quad (3.7)$$

When the update value of the weighting factor is determined, it is applied in the direction of steepest descent which equals the opposite direction of the current gradient

$$\Delta \theta_j^t = \begin{cases} -\Delta u_j^t, & \text{if } \frac{\partial f_j}{\partial \theta_j}^t > 0 \\ +\Delta u_j^t, & \text{if } \frac{\partial f_j}{\partial \theta_j}^t < 0 \\ 0, & \text{else} \end{cases}. \quad (3.8)$$

Exception on Eq. (3.8):

$$\Delta \theta_j^t = -\Delta \theta_j^{t-1}, \text{ if } \frac{\partial f_j}{\partial \theta_j}^t \cdot \frac{\partial f_j}{\partial \theta_j}^{t-1} < 0. \quad (3.9)$$

With $\theta_j^{t+1} = \theta_j^t + \Delta\theta_j^t$, $j \in \mathbb{N}_{[1\dots Z]}$ and $t \in \mathbb{N}_{[1\dots \tau]}$ the amount of iterations. Every time the partial derivative $\frac{\partial f_j}{\partial \theta_j}^t$ of the corresponding weighting factor θ_j^t changes its sign, it is assumed that the last update was too big and the local minimum was passed. In Eq. (3.6) the step size is then reduced and in order to go back to the previous situation the update of the weighting factor is done as indicated by Eq. (3.9). To not again decrease the update value when going back to the previous situation where the gradient will again change it's sign, $\frac{\partial f_j}{\partial \theta_j}^t$ is set to zero.

If the derivative retains its sign with respect to the previous iteration, the update value is slightly increased in order to accelerate convergence in shallow regions [19]. Parameters set by the user are: Δu_j^0 , Δu_j^{max} , Δu_j^{min} , η^+ and η^- . In this thesis following values were chosen: $\Delta u_j^0 = 0.1$, $\Delta u_j^{max} = 1.0$, $\Delta u_j^{min} = 10^{-7}$, $\eta^+ = 1.2$ and $\eta^- = 0.5$.

3.3 Conclusion

In order to identify comfort criteria during driving, a literature study was conducted which was mainly questionnaire based. The results were that to be able to evaluate comfort, higher order kinematic variables like accelerations and jerks must be considered. These variables are important to acquire a smooth vehicle path and to give a continuous and natural feeling when driving. The quickness of the maneuver and the feeling of save driving should also be taken into account. Therefore comfort is naturally influenced by the environment of the vehicle. It was found that when more traffic is present, a higher tolerance for less smooth trajectories is granted. Finally, it was explained that if the goal of the maneuver is more rapidly completed, this influences the amount of comfort in a positive way.

In the second part of the literature study the concept of inverse optimal control was clarified and it was explained that it is the process of identifying the unknown objective of the human driver. Next, feature based learning has been looked into and a practical method was found in order to retrieve θ_{opti} making use of gradient descent. After this the RPROP algorithm was introduced that will be used to update the weighting factors while making use of the gradient descent method in order to match expected feature values with the observed ones.

Chapter 4

Learning from ideal data

This chapter focusses on implementing the learning of the different weighting factors in the comfort cost function $\theta^T \mathbf{F}(\mathbf{r})$, following the methods explained in 3.2. The use of ideal data is assumed which means that vehicle model mismatch is avoided by using the same model for learning the weighting factors and generating data. Thereby is the approximation that comfort of a driver is modelled by a linear relation of features, exactly fulfilled. Data is generated by choosing a set of weighting factors in order to produce kinematic vehicle signals by minimizing Eq. (3.5a). A validation of the developed learning algorithm is done when the chosen weighting factors used in generating the data are retrieved.

First, the non-linear bicycle model and the used parameters are explained in section 4.1. A non-linear bicycle model has been chosen because this can sufficiently model the dynamics during a comfortable lane change maneuver and the calculation cost stays affordable. The model makes abstraction of suspension dynamics i.e. neglects rolling and pitching and the position of the centre of gravity is fixed on the vehicle during driving [22]. Further on, the developed algorithm to learn the weight from demonstration is discussed in section 4.2. Afterwards it is shown how the ideal data is generated and validated in section 4.3 and the learning results are analysed in section 4.4.

As noted before the learning process discussed, concerns an offline optimization which allows for higher calculation loads because of the absence of pushing real time constraints.

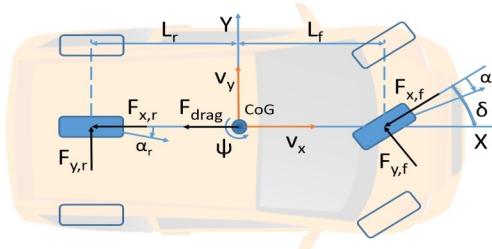


Figure 4.1: Non-linear bicycle model (Source: [20]).

4.1 Non-linear bicycle model

The free body diagram of the non-linear bicycle model can be seen in figure 4.1. Two variants of this model are discussed which are differentiated by the smoothness of the controls that are used.

- Model has 6 states and 2 controls:

$$\mathbf{X} = [x \ y \ v_x \ v_y \ \psi \ \dot{\psi}]^T \text{ and } \mathbf{U} = [t_r \ \delta]^T. \quad (4.1)$$

- Model has 10 states and 2 controls:

$$\mathbf{X} = [x \ y \ v_x \ v_y \ \psi \ \dot{\psi} \ t_r \ \delta \ a_x \ a_y]^T \text{ and } \mathbf{U} = [\dot{t}_r \ \dot{\delta}]^T. \quad (4.2)$$

x and y in the above formulation are the position of the centre of gravity of the vehicle in the global coordinate system. v_x and v_y are the vehicle velocities in the local vehicle frame. ψ is the vehicle yaw angle and $\dot{\psi}$ the yaw rate. The control vector of Eq. (4.1) consists of the throttle t_r and the angle of the front wheel δ . In the more extended bicycle model Eq. (4.2), throttle and front wheel angle serve as states. a_x and a_y are the total accelerations of the centre of gravity in the local vehicle frame. The inputs in this second formulation are the first order derivatives of throttle and front wheel angle.

The equations of motion derived and checked in literature [20] are¹ :

$$\begin{aligned} \dot{x} &= v_x \cos(\psi) - v_y \sin(\psi), \\ \dot{y} &= v_x \sin(\psi) + v_y \cos(\psi), \\ m\dot{v}_x &= F_{x,f} \cos(\delta) - F_{y,f} \sin(\delta) + F_{x,r} - F_{drag} + mv_y \dot{\psi}, \\ m\dot{v}_y &= F_{x,f} \sin(\delta) + F_{y,f} \cos(\delta) + F_{y,r} - mv_x \dot{\psi}, \\ \dot{\psi} &= \dot{\psi}, \\ I_z \ddot{\psi} &= L_f(F_{y,f} \cos(\delta) + F_{x,f} \sin(\delta)) - L_r F_{y,r}, \\ \dot{t}_r &= \dot{t}_r, \\ \dot{\delta} &= \dot{\delta}, \\ a_{tx} &= \dot{v}_x, \\ a_{nx} &= -v_y \dot{\psi}, \\ a_{ty} &= \dot{v}_y, \\ a_{ny} &= v_x \dot{\psi}, \\ j_x &= \dot{a}_{tx} + \dot{a}_{nx}, \\ j_y &= \dot{a}_{ty} + \dot{a}_{ny}. \end{aligned} \quad (4.3)$$

¹ Appendix A shows the complete jerk equations j_x and j_y .

The drag force is calculated as:

$$F_{drag} = C_{r0} + C_{r1}v_x^2, \quad (4.4)$$

with C_{r0} the roll resistance and C_{r1} the air drag contributions.

To calculate the tyre forces, a linear tyre model is used instead of a more complex non-linear model e.g. Pacejka tyre model. The longitudinal tyre forces are calculated as:

$$\begin{aligned} F_{x,f} &= \frac{t_r T_{max}}{2R_w}, \\ F_{x,r} &= F_{x,f}. \end{aligned} \quad (4.5)$$

R_w is the wheel radius and T_{max} the maximum torque the engine is able to supply. Because $F_{x,r} = F_{x,f}$ the longitudinal forces that are induced by the engine are equally distributed between front and rear axle (division by 2 in above equations). The coefficient t_r is the normalised amount of throttle that can be applied and has a value between -1 and 1 (negative for braking). In the bicycle model it is assumed that braking behaves the same as giving a negative amount of throttle. The lateral tyre forces are calculated based on the tyre slip angles α_f and α_r :

$$\begin{aligned} \alpha_f &= -\text{atan}\left(\frac{\dot{\psi}L_f + v_y}{v_x}\right) + \delta, \\ \alpha_r &= \text{atan}\left(\frac{\dot{\psi}L_r - v_y}{v_x}\right), \end{aligned} \quad (4.6)$$

resulting in:

$$\begin{aligned} F_{y,f} &= 2K_f\alpha_f, \\ F_{y,r} &= 2K_r\alpha_r. \end{aligned} \quad (4.7)$$

The use of this linearised lateral tyre model is valid for small lateral accelerations ($a_y \leq 4m/s^2$) and slip angles ($\alpha \leq 5^\circ$) [20]. It is acceptable to use this approximate model in this thesis as the goal is to learn a comfortable and thus smooth lane change manoeuvre. However, these constraints will be checked during section 4.3.2.

The vehicle model parameters used are described in table 4.1. These values were provided by Siemens Digital Industries Software. The *Gsteerfactor* is used to approximate the relation between the front wheel angle and the steer wheel angle turned by the driver, linearly according to: $\delta = \frac{\delta_s}{Gs}$.

Parameter	Value
Vehicle mass m [kg]	1430
Moment of inertia I_z [kgm^2]	1300
Front axle distance L_f [m]	1.056
Rear axle distance L_r [m]	1.344
Roll resistance coefficient C_{r0} [N]	0.6
Air drag coefficient C_{r1} [$\frac{Ns^2}{m^2}$]	0.1
Engine torque limit T_{max} [Nm]	584
Wheel radius R_w [m]	0.292
Lateral front tyre stiffness K_f [N]	41850.85
Lateral rear tyre stiffness K_r [N]	51175.78
Gsteerfactor G_s [-]	16.96

Table 4.1: Used vehicle model parameters.

4.2 Learning algorithm

The goal of the learning algorithm is to learn the weighting factors $\boldsymbol{\theta}$ in the comfort objective function: $\boldsymbol{\theta}^T \mathbf{F}(\mathbf{r})$. Its formulation is presented in section 4.2.1. The features that are the entries of the feature vector $\mathbf{F}(\mathbf{r})$ capture a notion of comfort felt by the driver. Based on the literature study displayed in Chapter 3 and on paper [12], the comfort criteria embedded in the different feature values are presented in section 4.2.2. The scenario of a lane change on a straight road is modelled with the time horizon of the maneuver itself taken as an optimization variable T .

4.2.1 Formulation of the algorithm

As has been discussed in chapter 3, $\boldsymbol{\theta}_{opti}$ gives the best possible fit between $\mathbf{F}(\mathbf{r}_{expected})$ and $\mathbf{F}_{obs}(\mathbf{r}_{obs})$. Without vehicle mismatch, a match of feature values will induce a good match of the kinematic vehicle signals as will be further discussed in section 4.4. The flow of a single dataset learning algorithm can be seen in Figure 4.2.

The learning is started by proposing a set of weighting factors e.g. all equal to one. Eq. (3.5a) is minimized in order to generate an expected path. From this $\mathbf{F}(\mathbf{r}_{expected})$ can be retrieved by using the definition discussed in section 4.2.2. Afterwards, the relative features $f_{rel,i}$ are calculated by element-wise divide the expected feature vector by the observed one \mathbf{F}_{obs} . A perfect match is acquired when the division equals one and the learning algorithm is terminated. The tolerance on convergence towards one, is chosen during this chapter equal to 10^{-3} .

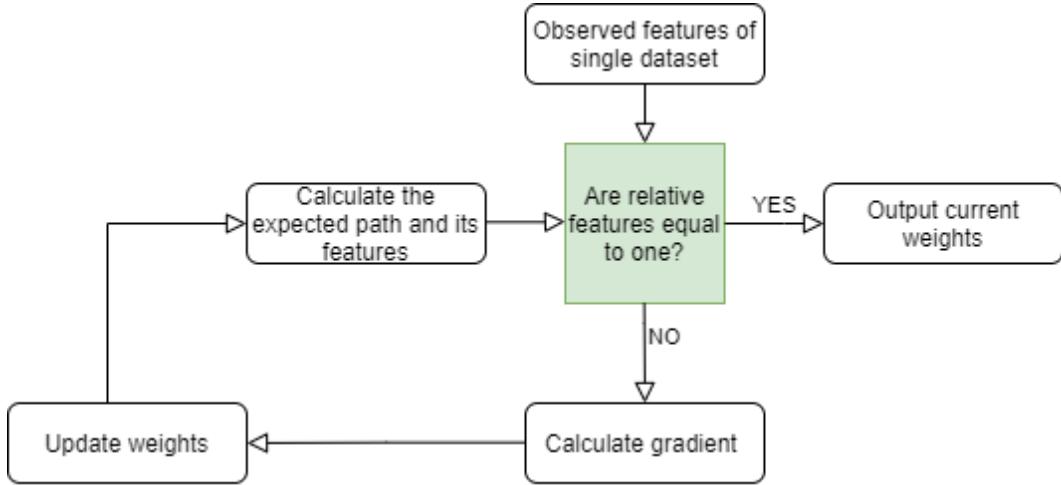


Figure 4.2: Basic flow of the inverse optimal control algorithm.

While no convergence takes place the weighting factors are updated, making use of the estimation of the approximated gradient and the RPROP algorithm explained in Chapter 3. The new weighting factors are used to calculate a new expected path which will result in different expected feature values $\mathbf{F}(\mathbf{r}_{expected})$. Next, a more detailed description of the calculation of the expected path is given by Eq. (4.8) which uses as vehicle model Eq. (4.2)

$$\begin{aligned}
 & \min_{\mathbf{X}(\cdot), \mathbf{U}(\cdot), T} \quad \theta^T \mathbf{F}(\mathbf{X}, \mathbf{U}, T) \\
 \text{s.t.} \quad & \mathbf{X}^{k+1} = I(\mathbf{X}^k, \mathbf{U}^k) \quad k = [0, \dots, N-1] \\
 & \mathbf{X}^0[1 : 8] = \mathbf{X}_{initial} \\
 & T \leq T_{limit} \\
 & \mathbf{F}(\mathbf{X}^k) \geq 0 \quad k = [0, \dots, N] \\
 & \mathbf{H}(\mathbf{X}^k) = 0 \quad k = [0, \dots, N] \\
 & \mathbf{X}^k \in \mathbb{R}^{10 \times 1} \quad k = [0, \dots, N] \\
 & \mathbf{U}^k \in \mathbb{R}^{2 \times 1} \quad k = [0, \dots, N-1] \\
 & T \in \mathbb{R}, \quad N \in \mathbb{N}
 \end{aligned} \tag{4.8}$$

Where $\mathbf{X} \in \mathbb{R}^{10 \times N+1}$ and $\mathbf{U} \in \mathbb{R}^{2 \times N}$ contain respectively the states and controls of Eq.(4.2) during the maneuver. The time of the maneuver is represented as T . In order to discretize time, a multiple shooting approach is adopted as explained in section 2.1.1. The amount of integration intervals N is chosen equal to 1000 and determines the amount of controls applied to go from the initial state towards the end state. Inside the function I the Runge-Kutta time integration method is embedded in order to connect different states when a certain control is applied for ΔT . Here the equations of motion Eq. (4.3) are inputted in the optimization because derivatives of

the vehicle states are needed. The time discretization used can be categorized as a direct method and because a multiple shooting approach is conducted at every time instance, a new set of state optimization variables is introduced. The path constraint vector \mathbf{F} demarcates together with the equality constraint vector \mathbf{H} , the feasible space of the solutions for \mathbf{X} and \mathbf{U} . An overview of the constraints used, is given by Eq. (4.9) and Eq. (4.10)

$$\mathbf{F} = \left\{ \begin{array}{ll} -\frac{\text{Width Lane}}{2} \leq y^k \leq \frac{3 \cdot \text{Width Lane}}{2}, & k = [0, \dots, N] \\ 0 \leq x^k, & k = [0, \dots, N] \\ -\frac{\pi \cdot 150}{180Gs} \leq \delta^k \leq \frac{\pi \cdot 150}{180Gs}, & k = [0, \dots, N] \\ -1 \leq t_r^k \leq 1, & k = [0, \dots, N] \end{array} \right\}, \quad (4.9)$$

$$\mathbf{H} = \left\{ \begin{array}{l} y^N = \text{Width Lane} \\ vy^N = 0 \\ \psi^N = 0 \\ \dot{\psi}^N = 0 \\ \delta^N = 0 \end{array} \right\}. \quad (4.10)$$

The constraints displayed in \mathbf{H} make sure that at the end of the lane change, the slip angles in the tires and the steer wheel angle are zero. From Eq. (4.3) this induces that the lateral velocity, acceleration and yaw acceleration also become zero and this marks the end of the lane change. y^N ensures that the wanted lateral distance is covered. This distance can be calculated from the start position of the vehicle and the width of the lane in order to end up at the centre line of the desired lane.

At the start of the lane change, straight driving at constant longitudinal speed is assumed. To achieve this, the constraints of Eq. (4.11) are used. No constraints for acceleration is needed. This would give a redundancy due to the other initial states in combination with the equations of motion 4.3. In Eq. (4.11) all initial states are zero except the initial speed $v_{x,start}$ and $t_{r,start}$. The amount of throttle at the start of the lane change is chosen to overcome the aerodynamic drag without accelerating. This is given by $t_r^0 = \frac{(C_{r0} + C_{r1}v_{start}^2)r_w}{T_{max}}$. Therefore, it can be concluded that the parameters that distinguish different lane changes are $v_{x,start}$ and Width Lane . This is exploited when generating different ideal lane change datasets for chosen weighting factors

$$\mathbf{X}_{initial} = \begin{bmatrix} x_{start} \\ y_{start} \\ v_{x,start} \\ v_{y,start} \\ \dot{\psi}_{start} \\ \ddot{\psi}_{start} \\ t_{r,start} \\ \delta_{start} \end{bmatrix}. \quad (4.11)$$

The time limit constraint in Eq. (4.8) is needed in order to demarcate the optimization solution space. When set, it has to take two conflicting criteria into account. It has to be chosen large enough in order to have a minor influence on the lane change behaviour introduced by this constraint. Secondly, it has to be taken small enough to preserve good conditions for the numerical integration performed inside $\mathbf{F}(\mathbf{X}, \mathbf{U}, T)$, as will be seen in in 4.2.2. This is because the number of optimization points $N + 1$ of the states is fixed, which means that a larger time limit will give a coarser time discretization. In this thesis the time limits used are 25 s and 30 s, which is a compromise between the two criteria. This choice is validated in section 4.3.2. It is worth noting that with the removal of the time limit constraint the solution of Eq. (4.8) gives a lane change that takes around 160 s. This is not a realistic results because the objective will, as previously explained, not have good numerical properties.

In order to solve Eq. (4.8) an initial guess is needed for the longitudinal velocity in order to avoid the emergence of an invalid number. The default initial guess used in the CasADi software is an all zero vector. As can be seen in Eq. (4.6) this would give a division by zero in the calculation of the slip angles.

To further enhance the solving speed of Eq. (4.8) also initial guesses are given for the other vehicle states and additionally the controls. To do this, a feasible solution of the non-linear bicycle model for a lane change is needed because IPOPT, which is used to calculate the solution of Eq. (4.8), is an interior point method. Therefore, the initial guesses for \mathbf{X} , \mathbf{U} and T are taken from the observed ideal data.

Another way to speed up the solving time of the IPOPT solver, is setting the initial guess of the lambda multipliers internally used, equal to the ones found during the previous call of Eq. (4.8) during the loop visualized by figure 4.2.

The time needed for the CPU to calculate the expected path for a certain set of weighting factors and thus solving Eq. (4.8), takes around 5 s (python implementation) when a time limit of 30 seconds and N equal to 1000 is used.

As discussed above, the solver used to calculate the states and control signals in Eq. (4.8), is IPOPT which is an open source solver. The idea behind it is to smoothing the KKT conditions and transform the optimization problem into a smooth root finding problem [16].

4.2.2 Objective function

The objective function used in Eq. (4.8) has to represent comfort felt by the driver and is based on the literature study displayed in section 3.1. The choice made in how to define the different features is important because it sets the fixed framework where the weighting factors will be learned in. It can be expected that the linear relation of features given by $\theta^T \mathbf{F}(\mathbf{r})$ will serve as an approximation for the real, more complex comfort objective of a human drive. The feature framework that is further discussed in this section, can be validated and adjusted based on an user study

$$\text{discomfort} = \theta_1 \cdot f_1 + \theta_2 \cdot f_2 + \theta_3 \cdot f_3 + \theta_4 \cdot f_4 + \theta_5 \cdot f_5 + \theta_6 \cdot f_6. \quad (4.12)$$

$$f_i, \theta_i \in \mathbb{R} \quad i \in \mathbb{N}$$

Feature 1: longitudinal acceleration

$$f_1 : \mathbf{r} \rightarrow f_1(\mathbf{r}) = \int_0^T a_{x,\text{total}}^2(t) dt. \quad (4.13)$$

Feature one assesses the amount of discomfort by integrating the total longitudinal acceleration in the local axis. The local axis is fixed to the centre of gravity of the vehicle, as can be seen in Figure 4.1. The total longitudinal acceleration $a_{x,\text{total}}$ is the sum of $a_{x,\text{tangential}}$ and $a_{x,\text{normal}}$ as described in Eq. (4.3).

Feature 2: lateral acceleration

$$f_2 : \mathbf{r} \rightarrow f_2(\mathbf{r}) = \int_0^T a_{y,\text{total}}^2(t) dt. \quad (4.14)$$

Feature two assesses the amount of discomfort by integrating the total lateral acceleration in the local axis. The total lateral acceleration $a_{y,\text{total}}$ is the sum of $a_{y,\text{tangential}}$ and $a_{y,\text{normal}}$ as described in Eq. (4.3).

Feature 3: longitudinal jerk

$$f_3 : \mathbf{r} \rightarrow f_3(\mathbf{r}) = \int_0^T j_x^2(t) dt. \quad (4.15)$$

Feature three gives the amount of comfort by integrating the total change of longitudinal acceleration during the followed path.

Feature 4: lateral jerk

$$f_4 : \mathbf{r} \rightarrow f_4(\mathbf{r}) = \int_0^T j_y^2(t) dt. \quad (4.16)$$

Feature four gives the amount of comfort by integrating the total change of lateral acceleration during the followed path.

Feature 5: desired speed

$$f_5 : \mathbf{r} \rightarrow f_5(\mathbf{r}) = \int_0^T (v_{des} - v_x)^2 dt. \quad (4.17)$$

v_{des} is assumed to be a constant value and set equal to the start velocity just before the lane change.

Feature 6: desired lane change

$$f_6 : \mathbf{r} \rightarrow f_6(\mathbf{r}) = \int_0^T (L - y)^2 dt. \quad (4.18)$$

L is a constant and set equal to the desired lateral distance. If the vehicle reaches its desired lateral displacement faster, this is perceived as a good response and is interpreted as comfort, as discussed in section 3.1.

In order to implement the integrals defined above in the objective function of Eq. (4.8), discretization is needed. For this the Crank-Nicolson numerical integration is used as is shown in Eq. (4.19) for feature f_j

$$\int_{f_j^n}^{f_j^{n+1}} df_j = \int_{t^n}^{t^{n+1}} P(t) \cdot dt, \quad (4.19a)$$

$$f_j^{n+1} - f_j^n = \frac{1}{2} \frac{P(t^{n+1}) + P(t^n)}{\Delta T}. \quad (4.19b)$$

To summarize, the objective described by Eq. (4.12) consists of a set of comfort features that model the amount of discomfort experienced during a maneuver. This is achieved by mapping kinematic signals onto scalar feature values through integration. By finding the driver specific weighting factors θ in Eq. (4.12), it is possible to model driver preferences between different comfort features. With the information of the identified objective function an autonomous vehicle can perform path planning of the most comfortable path to do a lane change for a specific driver.

As is shortly discussed in Chapter 3, the perception of driving safe contributes to the amount of comfort that is experienced. Safe driving comprises next to smooth behaviour also distances between other road agents. However features that consider the environment are not taken into account in Eq. (4.12). This can be achieved if data of the position of other vehicles during the maneuver is available. The paper 'Learning driving styles for autonomous vehicles from demonstration' [12], gives some suggestions showed by Eq. (4.20) and Eq. (4.21)

$$f_d = \sum_{k=1}^{NA} \int_0^T \frac{1}{(x_{o,k}(t) - x)^2 + (y_{o,k}(t) - y)^2} \cdot dt. \quad (4.20)$$

$$NA \in \mathbb{N}$$

With $[x_o, y_o]_k$ the position of the closest point of a different agent and NA the total amount of road agents in the nearby area.

Not only the bird's eye view distance between two vehicles plays a role, also the following distance of a vehicle in the same lane is important. This can be modelled as:

$$f_d = \int_0^T \max(0, \hat{d} - d(t)) \cdot dt. \quad (4.21)$$

$$\hat{d} \in \mathbb{R}$$

The minimum desired following distance \hat{d} , can be calculated based on the distance needed to avoid collision during unexpected events when driving at a certain longitudinal velocity.

An other assumption, which is not discussed in this thesis, is the time limit to finish a maneuver. In a real life application however, this limit often influences the maneuver. After the weighting factors are identified, the most comfortable path with a set time limit can be planned for a specific driver.

4.2.3 Normalization factors

To reduce the effect of order of magnitude given by the units in the objective, a normalization of the features is done. The kinematic signals of an example lane change are produced and the same features that are used in the objective function are calculated from it. These will be the normalization factors as can be seen in Eq. (4.22). Table 4.2 gives an overview of the lane change normalization factors used.

Normalization factor	Value
Nr.1	0.0073
Nr.2	2.64
Nr.3	0.0073
Nr.4	11.28
Nr.5	0.047
Nr.6	17.14

Table 4.2: Overview of normalization factors.

Because of the normalization the relative weighting factors defined as $\theta_{r,i} = \frac{\theta_{abs,i}}{norm_i}$, will quantify the trade-offs between different comfort features without disturbance of

units used. Aside of this, it allows to learn weighting factors faster, because of the absence of large size differences induced by unit differences that are present in the absolute weighting factors. During the learning loop (Figure 4.2) the max weighting factor update is set to 1 in this thesis. When the absolute weighting factors are learned instead of the relative ones, it would give a substantially higher amount of iterations when started from an all-one vector as initial guess. When the algorithm is used in practise, these normalization factors can be recalculated from an observed lane change making use of the definition of the features as defined in section 4.2.2.

4.3 Ideal data

As mentioned at the begin of this chapter the term 'ideal data' concerns data that is generated with a non-linear bicycle model and with an objective function Eq. (4.12) known beforehand which consists of a linear combination of features and weighting factors θ . This means that the algorithm that learns the weighting factors can be validated by checking if it is able to find back the correct weighting factors. First the generation of ideal data is presented in section 4.3.1 whereafter it is validated in section 4.3.2.

4.3.1 Generation

The relative weighting factors chosen to generate the data are $[4, 5, 1, 6, 1, 2]$, which gives as absolute weighting factors $[549.75, 1.90, 137.44, 0.53, 21.43, 0.12]$ when the normalization discussed in section 4.2.3 is taken into account. The objective that is used in Eq. (4.8) is given by Eq. (4.22). The different feature values are the ones as defined in section 4.2.2

$$\frac{4}{0.0073} \cdot f_1 + \frac{5}{2.64} \cdot f_2 + \frac{1}{0.0073} \cdot f_3 + \frac{6}{11.28} \cdot f_4 + \frac{1}{0.047} \cdot f_5 + \frac{2}{17.14} \cdot f_6. \quad (4.22)$$

$$f_i \in \mathbb{R}, i \in \mathbb{N}$$

When multiple ideal datasets have to be generated in order to serve as observations, the initial speed V_0 and width of the lateral distance L are varied, while using the same set of weighting factors.

4.3.2 Validation

In this section the results of the generated ideal data are discussed. There is looked at a numerical vs analytical formulation, influence of the initial guess, choice of time limit, amount of control points, use of linear tire model and at discussion on the resulting feature values of the generated data.

Numerical vs Analytical formulation

In section 4.1 about the non-linear bicycle model, two different variants were described by Eq. (4.1) and Eq. (4.2). Variant one has only 6 states and the accelerations and

jerks in the objective function Eq. (4.12), are calculated by making use of numerical differentiation described by Eq. (4.23) from the other vehicle states. Variant two on the other hand has the total accelerations as direct states in the vehicle model and uses an analytical formulation of the jerk as defined in appendix A.

$$\frac{\partial \phi}{\partial t} = \frac{\phi(i+1) - \phi(i-1)}{2\Delta t} \quad (4.23a)$$

$$\frac{\partial^2 \phi}{\partial t^2} = \frac{\phi(i+1) - 2\phi(i) + \phi(i-1)}{\Delta t^2} \quad (4.23b)$$

In order to validate the two approaches the generated lateral jerk signals are compared in Figure 4.3.

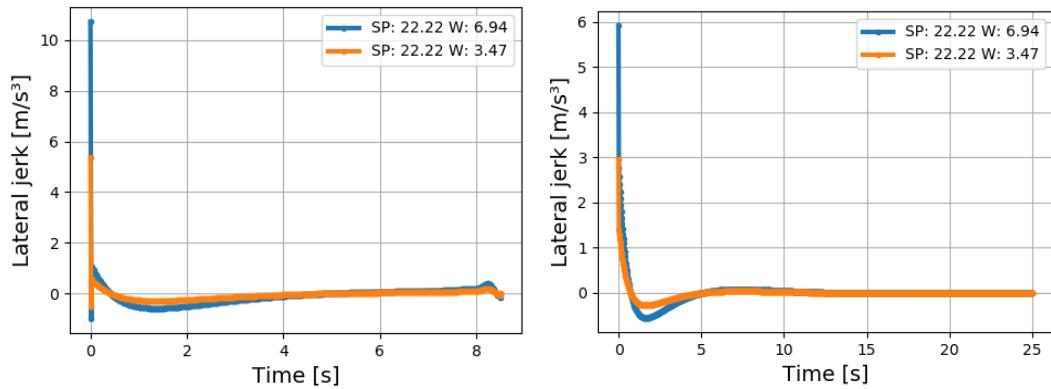


Figure 4.3: A comparison between the numerical jerk (left) based on Eq. (4.23) and Eq. (4.1) and the analytical jerk (right) based on appendix A and Eq. (4.2).

It is clear that the analytical formulation (right figure) of the jerk gives smoother results and less high peaks which is in line with observations of the more complex 15 dof Amesim vehicle that is discussed in chapter 5. Therefore the analytical formulation is a better approach of reality. The analytical formulation contains more information about the vehicle i.e. the equations resulting from numerical differentiation are approximations of the analytical ones.

Initial guess

To determine if the generated data is a local solution, two different initial guesses $V_0 : 22.22 - L3.47$ and $V_0 : 25.00 - L6.94$ are used whereof the retrieved data feature values are summarized in table 4.3. V_0 is the initial speed and L the desired lateral distance of the lane change used as initial guess. These two initial guesses are generated using the numerical formulation as described in section 4.3.2. In order to calculate the data features displayed in table 4.3, time limit is set on 30 s, N on 1000, $V_0 = 22.22 \frac{m}{s}$ and $L = 3.47 m$ as parameters in Eq. (4.8) and the analytical formulation is used. The results suggest that the generated data feature values are not a local solution because they are found back from different start points of the

optimization. During the learning of the weighting factors as described in Figure 4.2, the initial guess is set equal to the observed data.

Feature Value	V0:22.22 - L3.47	V0:25.00 - L6.94
Nr.1	6.83e-8	6.83e-8
Nr.2	0.37	0.37
Nr.3	1.77e-7	1.77e-7
Nr.4	0.57	0.57
Nr.5	1.98e-6	1.98e-6
Nr.6	30.94	30.94

Table 4.3: This table shows the retrieved feature values using two different initial guesses in Eq. (4.8).

Time limit

In order to check the dependency of the generated data on the chosen T_{limit} constraint in Eq. (4.8), data is generated for a lane change with N, the amount of control points equal to 1000, initial velocity equal to $80 \frac{km}{h}$ (V_0), a desired lateral displacement of $3.47 m$ (L) and a varying T_{limit} constraint as indicated in table 4.4. Appendix B shows what the difference in feature values actually means for the different kinematic signals of the vehicle.

Feature Value	20 s	50 s	100 s
Nr.1	3.66e-8	1.13e-7	2.04e-7
Nr.2	0.37	0.38	0.38
Nr.3	1.13e-7	3.98e-7	1.56e-6
Nr.4	0.58	0.57	0.54
Nr.5	1.72e-6	2.27e-6	3.16e-6
Nr.6	31.05	30.74	30.42

Table 4.4: This table shows the retrieved feature values using different time limits in Eq. (4.8) and using as vehicle model Eq. (4.2).

Looking at even greater time limits is not desirable because beyond a time limit of $100 s$ the time discretization gets larger than $0.1 s$ for N equal to 1000, which will result in a less reliable discretization.

From the result of table 4.4 it can be concluded that the influence of manually setting the time limit in Eq. (4.8), can be neglected. The lateral features that are the most important ones, are very similar and also the kinematic signals for the lateral variables found in Appendix B show the same behaviour.

Amount of control points

The test carried out to investigate the dependence of the resulting feature values on the amount of control points N , uses the same parameters as described in the previous section but fixes the time limit on 30 s and varies N over 500, 1000 and 1500 points. The results are shown in table 4.5 whereof it follows that a choice of N equal to 1000 is justified considering the small difference of the obtained features when N is chosen equal to 1500. For a full overview of the different kinematic signals, reference is made to Appendix B. Again the kinematic vehicle signals confirm the same conclusion as taken from table 4.5.

Feature Value	500	1000	1500
Nr.1	9.42e-8	6.63e-8	6.45e-8
Nr.2	0.38	0.37	0.37
Nr.3	5.61e-7	1.77e-7	1.11e-7
Nr.4	0.56	0.57	0.58
Nr.5	2.50e-6	1.98e-6	1.85e-6
Nr.6	30.66	30.94	31.05

Table 4.5: This table shows the retrieved feature values using different amount of control point N in Eq. (4.8) and using as vehicle model Eq. (4.2).

Linear tire model

This section checks if the conditions to use a linearised lateral tire model is valid. In paper 'Safety-Critical Control for Non-affine Non-linear Systems with Application on Autonomous Vehicle' [20], it was found that this is the case when there are small lateral accelerations ($a_y \leq 4\frac{m}{s^2}$) and slip angles ($\alpha \leq 5^\circ$) during the maneuver. Figure 4.4 gives the total lateral acceleration during a lane change maneuver that moves two lanes or an estimated lateral distance of 6.94 m . Figure 4.5 shows the slip angle during this maneuver. From the graphs it can be concluded that the linearisation of the lateral tire forces is valid and there is no need for a more complex tyre model. Both figures are generated with the complex vehicle model discussed in chapter 5 and make use of Eq. 4.6 in order to estimate the slip angle.

Feature values

In the tables above, it can be seen that the first, third and fifth feature values concerning longitudinal behaviour of the vehicle are very small. This often means that during a lane change these features have a low influence. It can be expected that for small observed feature values, it becomes hard to accurately learn weighting factors that generate feature values that match with these small values. The reason for this is because the feature value is negligible which means that any weight factor times zero stays zero in the objective function of Eq. 4.8. For this reason almost any weighting factor can be used for the longitudinal features without observing



Figure 4.4: Lateral acceleration during a lane change $V_0 : 25.00 \frac{m}{s}$ and $L : 6.94m$ generated with the 15 dof Amesim model.

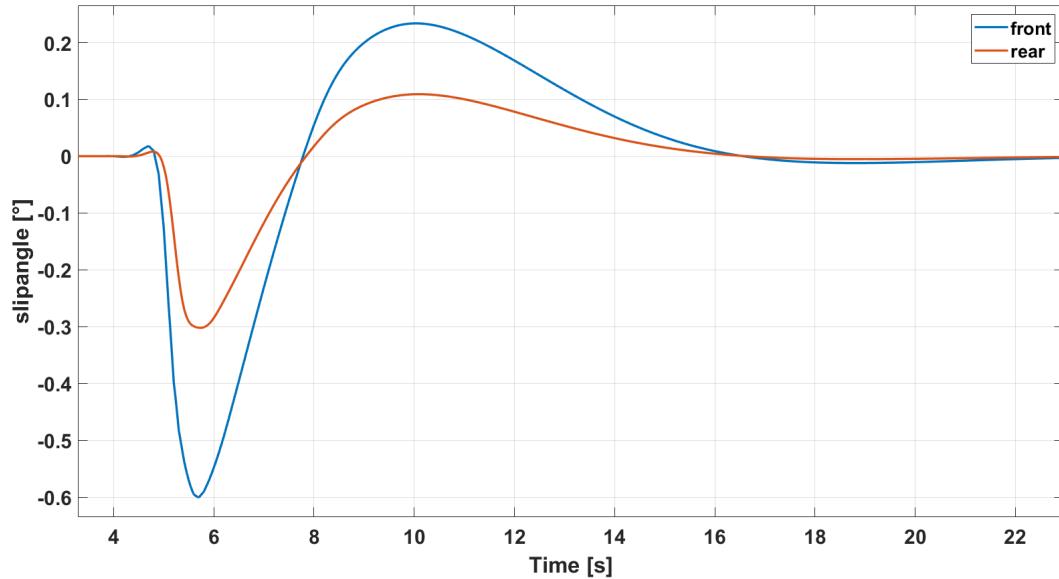


Figure 4.5: slip angle during lane change $V_0 : 25.00 \frac{m}{s}$ and $L : 6.94m$ generated with the 15 dof Amesim model.(blue:front,red:rear)

a different lane change outputted by the optimization done in Eq. 4.8. Consequently, when the initial speed is varied and the lateral displacement stays the same when generating ideal data, because of the small longitudinal feature values the data features found are almost identical. This will be further discussed in section 4.4.1.

Conclusion

In this section there is looked into the correctness of the generated data making use of Eq. (4.2), that further will be used to learn from. The influence of different choices i.e. initial guess, the time limit and the amount of control points has been checked. Out of this it followed that the same data features were found for different initial guesses and that it is allowed to take the time limit equal to 30 s and the amount of control points to 1000 as is done further on. Next, it was concluded that it is justified to use a linearised tire model. Finally, the resulting feature values of the ideal data were shortly discussed. It was concluded that a worse learning is expected for the retrieved longitudinal features due to their small values.

4.4 Ideal data learning results

In this section the results of learning weighting factors from ideal data is discussed. Section 4.4.1 concerns the learning of a single dataset $V_0 : 22.22 \frac{m}{s}$, $L : 3.47 m$. In reality one demonstration doesn't capture human preferences perfectly. Therefore, learning of multiple datasets is needed. The averaging method is presented in section 4.4.2 and the conflict method in section 4.4.3. Afterwards a comparison is made in section 4.4.4.

It is known beforehand that the weighting factors used to generate the data feature values are $[4, 5, 1, 6, 1, 2]^2$ and the initial guess of the weighting factors is chosen as an all-one vector. The convergence criteria to stop the learning loop as displayed in Figure 4.2, is when the maximum amount of iterations set to 300 is reached or if the data feature values are accurately regenerated with learned weighting factors.

This is quantified by $f_{rel,i} = \frac{f_{learned,i}}{f_{obs,i}} \leq 10^{-3}$. f_{obs} equals the generated data feature vector retrieved according to section 4.3 and is constant during learning. $f_{learned}$ is the feature vector calculated from the learned path and changes during every learning loop. Convergence is reached when the three lateral feature values 2, 4 and 6, that dominantly define the lane change, are accurately fitted for a certain set of weighting factors. The simulations done in this thesis are performed on a notebook provided by Siemens with Intel Core i7-7920HQ CPU @ 3.10GHz and 32 GB of RAM memory.

4.4.1 Single dataset learning

The generated data features that have to be matched, can be seen in table 4.3. The resulting weighting factor vector $\boldsymbol{\theta}$ and f_{rel} outputted at convergence on iteration 28 is displayed in table 4.6. The weighting factor concerning the lateral acceleration

²The corresponding features of the chosen weighting factors can be seen in section 4.2.2 and are summarized by $[f_{ax}, f_{ay}, f_{jx}, f_{jy}, f_{diffvx}, f_{diffy}]$.

(Nr.2), is taken as reference in order to compare the learned weighting factors with the chosen ones. The results show that the lateral weighting factors are found accurately back. Table 4.6 also shows the results when the algorithm is manually set to run for 121 iterations. A difference is noted for the lateral features, that have an increased matching of $f_{rel,i}$ with a new accuracy of 10^{-6} . However, the convergence of the longitudinal features $f_{rel,i}$ towards one, didn't increase much. This means that the longitudinal features are constraint in their ability to converge towards one.

	$It.28 - \theta$	$It.28 - f_{rel}$	$It.121 - \theta$	$It.121 - f_{rel}$
Nr.1	14.469	0.9907	10.021	1.0004
Nr.2	5.000	0.9995	5.000	1.0000
Nr.3	3.045	1.0037	2.179	0.9976
Nr.4	5.977	1.0006	5.998	1.0000
Nr.5	3.689	0.9941	2.538	0.9892
Nr.6	1.998	1.0001	2.002	1.0000

Table 4.6: This table shows the weighting factors learned from the dataset $V_0 : 22.22\frac{m}{s} - L : 3.47m$ and the associated f_{rel} for two different amount of iterations.

As already suggested in section 4.3.2 this is because of the small size of the feature values of the longitudinal direction. Less accurate learned longitudinal feature values will have a negligible influence on the overall behaviour of the planned path and as can be seen, a range of weighting factors will give an acceptable match quantified by $f_{rel,i}$.

In order to learn the longitudinal weighting factors of the human driver accurately, a maneuver should be considered where these features will be more prominent e.g. an acceleration maneuver or different features can be chosen e.g. $a_{tot}^2 = a_x^2 + a_y^2$. Because the interest in the longitudinal weighting factors during a lane change is marginal this is not further discussed during this thesis.

It could be argued that if the longitudinal feature weighting factors are not so important in defining the lane change, they have to be removed from the objective Eq. (4.12). This is not a correct assumption. The longitudinal features are less dominant during a lane change and therefore different longitudinal weighting factors give rise to the same lane changes. However, they still play a roll in comfortable path planning in order to avoid nervous throttle and consequently longitudinal jerk and acceleration behaviour.

4.4.2 Averaging method

In order to simultaneously learn from multiple datasets the averaging method is proposed by paper [12]. The flow of the algorithm is similar as shown in Figure 4.2 and the convergence criteria stays the same $f_{rel,i} = \frac{f_{learned,i}}{f_{obs,i}} \leq 10^{-3}$ for the lateral features. However, instead of inputting a data feature vector \mathbf{f}_{obs} based on a single

dataset, an averaged one over multiple datasets is used.

To calculate the gradient used to update the weighting factors θ , the difference between the averaged data feature vector and the averaged learned feature vector is taken. In order to obtain the averaged learned feature vector, m times the optimization Eq. (4.8) is called with m the amount of observed maneuvers. After solving for m resulting learned feature vectors, they are averaged. Each distinct maneuver has a different initial longitudinal speed and desired lateral displacement.

The datasets chosen to perform the learning on are: $V_0 : 22.22 \frac{m}{s} - L : 3.47 m$, $V_0 : 25.00 \frac{m}{s} - L : 3.47 m$ and $V_0 : 22.22 \frac{m}{s} - L : 6.94 m$. The resulting weighting factors and average f_{rel} found after 25 iterations are respectively $[14.459, 5.000, 3.204, 5.984, 3.720, 2.000]$ and $[0.9981, 0.9996, 0.9829, 1.0000, 0.9946, 1.0001]$. This means that the lateral chosen weighting factors are accurately found back. The f_{rel} of the individual datasets can be seen in table 4.7. The individual f_{rel} at convergence shows how good the match of feature values is when applying the learned weighting factors at Eq. (4.8) with the same parameters for V_0 and L . It is shown that there is a good match between the learned feature vectors and the individual data feature vectors.

Feature	V022.22 - L3.47	V022.22 - L6.94	V025.00 - L3.47
Nr.1	1.0000	0.9973	1.0064
Nr.2	1.0002	0.9992	1.0001
Nr.3	0.9860	0.9816	0.9983
Nr.4	1.0012	0.9995	1.0010
Nr.5	0.9972	0.9948	0.9902
Nr.6	0.9999	1.0002	0.9999

Table 4.7: This table shows the f_{rel} for each individual dataset at convergence using the average method.

Figure 4.6 shows the three initial guesses of the paths that make use of an all-one weighting factor vector in red, purple and brown. The finally learned paths can be seen in pink, grey and yellow. These paths lay exactly on the observed ones. From this it is concluded that matching of feature values which are scalars, give a good representation for the matching performance of the 2D kinematic vehicle signals. The rest of the kinematic signals originating from the non-linear bicycle model can be seen in Appendix C. The progress towards convergence over the iterations is shown in Figure 4.7. Convergence is reached in 25 iterations.

Figure 4.8 gives a view of how the learned weighting factors change over the iterations for the different features. The weighting factors in these graphs differ with the ones shown at the beginning of this section by a scaling factor $\frac{5.0}{\theta_2}$. Both these weighting factors will generate the exact same lane change when used in the objective of Eq. (4.8). During the learning process this degree of freedom can be removed by fixing the second weighting factor equal to 5.000. Convergence is reached after 59 iterations and

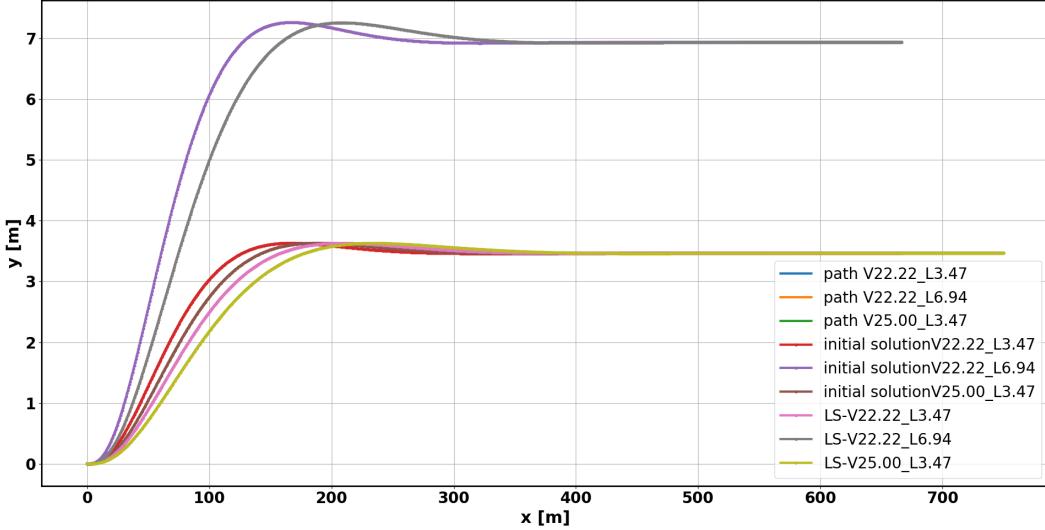


Figure 4.6: Observed, initial and learned paths for 3 different observed lane changes generated with common underlying objective function.

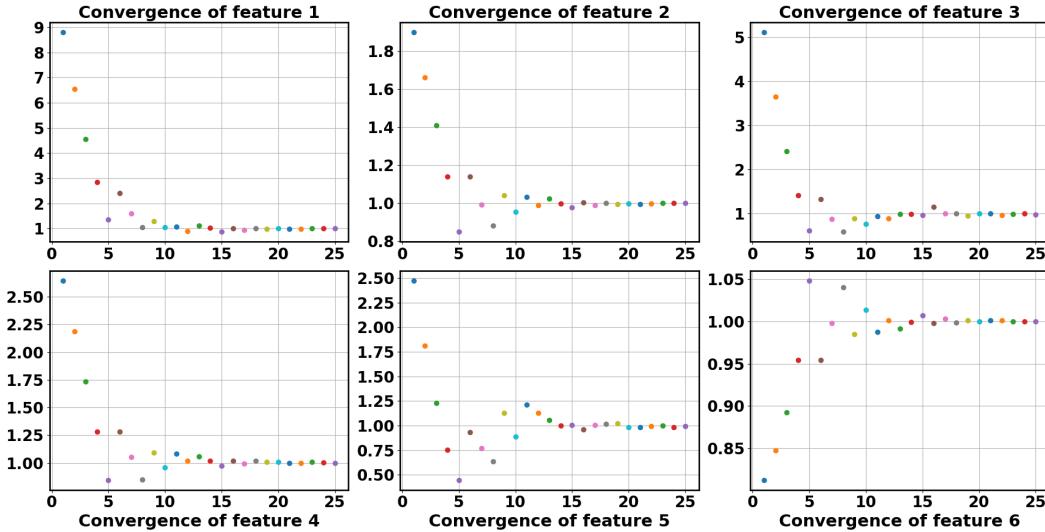


Figure 4.7: The evolvement of the average f_{rel} over the learning iterations.

following weighting factors are outputted $[15.143, 5.000, 3.222, 6.016, 3.880, 2.007]$ where the lateral weighting factors again clearly are found back. It takes the algorithm longer to find equivalent weighting factors because of the lost in the ability of the RPROP algorithm to update all the weighting factors. Therefore the scaling degree of freedom is retained during the rest of this thesis.

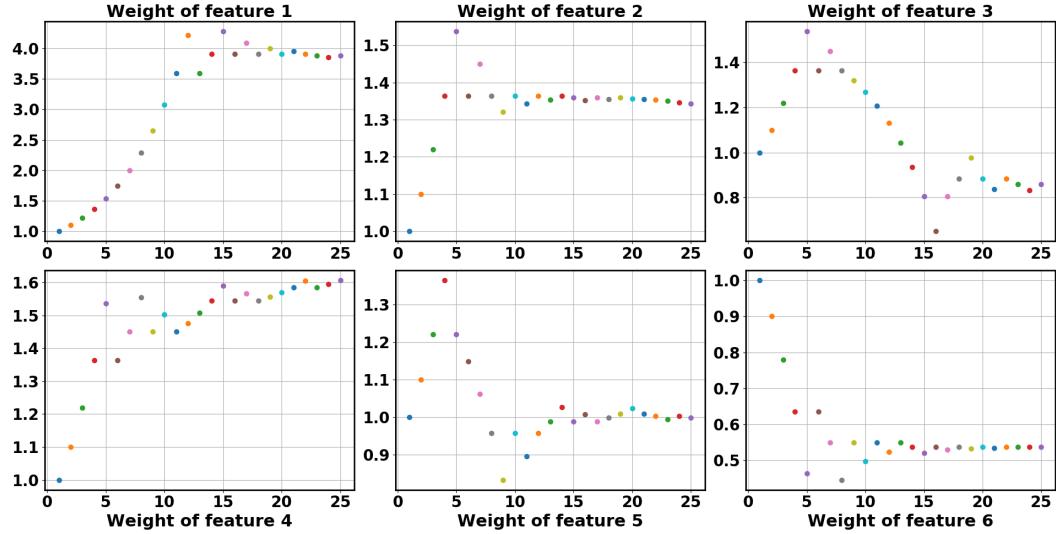


Figure 4.8: The different outputted weighting factors over the learning iterations.

4.4.3 Conflict method

The second method proposed to learn simultaneous of multiple datasets, is the conflict method. Its main idea is to only update a weighting factor if the gradient, as given by Eq. (3.5b), of the different individual datasets point all in the same direction. If there are individual gradients with conflicting signs, the update direction is ambiguous and the concerning weighting factor is not updated. The updating will be resumed when the conflict is solved by updating the other weighting factors. Solving conflicts is possible because the features are not independent of each other. Figure 4.9 shows how the conflict check is integrated together with the RPROP algorithm in the 'update weighting factors' box in the basic learning flow of Figure 4.2. The blue boxes serve as output ports.

Depending if the previous case was 'RPROP case 2' and the sign difference between the current and previous gradient, three distinct RPROP cases can be identified which outputs the new update value, delta weighting factors (dw) and the exception value as discussed in section 3.2.2. Inside the conflict block, it is checked if all the signs of the current gradients are consistent. This boils down to verifying on which entry in the m different gradients there is a sign difference. Here is m the amount of observations and an individual gradient is calculated as $\mathbf{F}_{obs,i} - \mathbf{F}(\mathbf{r}_{expected,i})$ with $m \in \mathbb{N}_{[1 \dots ND]}$.

If there is a sign difference, this means that for one dataset the learned feature value is higher than the observed one and the corresponding weighting factor should be increased (negative gradient) and for at least one other dataset, the corresponding weighting factor should be decreased (positive gradient). For such a case the conflict test will give rise to a positive conflict value. If there is no conflict there will be

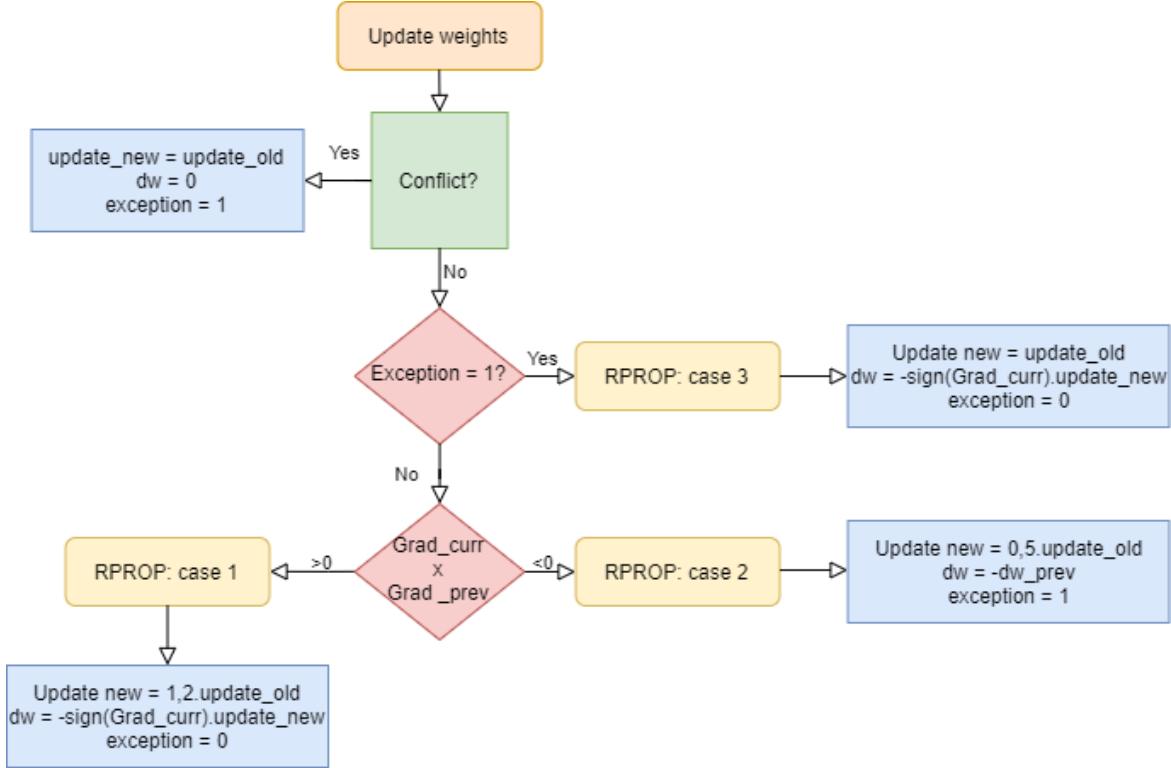


Figure 4.9: Flow of the conflict method as part of the basic flow diagram of Figure 4.2

unity in the decision of the RPROP case, because also no conflict was spotted in the gradients during the previous iteration. When a conflict is resolved, the next case will be automatically equal to RPROP case 3 because no decision can be made to either increase or decrease the update value. The convergence criteria used stays the same as discussed in 4.4.2 but now the additional criteria is added that there should still be improvement possible. This means that it should be possible to update a weighting factor in a direction that benefits every dataset.

The learning algorithm is applied on the same three datasets as discussed in section 4.4.2 and uses as initial guess of the weighting factors an all-one vector. The resulting weights found after 32 iterations are $[14.283, 5.000, 3.114, 5.979, 3.645, 2.000]$. The learning algorithm stopped because no update of the weighting factors could be done without ambiguity. It can be concluded that the lateral weighting factors are found back. The f_{rel} of the individual datasets can be seen in table 4.8. It is shown in the table that there is a good match between the learned feature vectors and the individual data feature vectors. Figure 4.10 presents the convergence of the f_{rel} vector of dataset $V_0 : 22.22 \frac{m}{s}$, $L : 3.47 m$. The individual convergence plots for the other datasets are very similar.

4. LEARNING FROM IDEAL DATA

f_{rel}	V022.22 - L3.47	V022.22 - L6.94	V025.00 - L3.47
Nr.1	0.9939	0.9913	1.0003
Nr.2	1.0003	0.9993	1.0002
Nr.3	0.9911	0.9868	1.0032
Nr.4	1.0016	0.9999	1.0014
Nr.5	1.0009	0.9985	0.9941
Nr.6	0.9998	1.0001	0.9998

Table 4.8: This table shows the f_{rel} for each individual dataset at convergence using the conflict method.

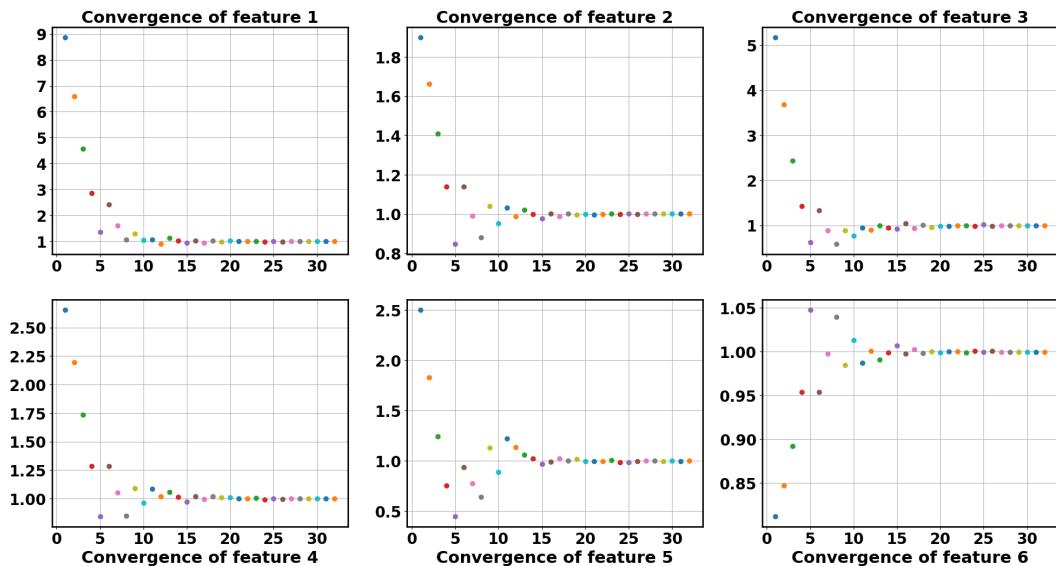


Figure 4.10: The evolvement of f_{rel} of dataset $V_0 : 22.22 \frac{m}{s}$, $L : 3.47 m$ over the learning iterations.

As can be seen in table 4.8 and Figure 4.10 the conflict method is capable of giving an adequate convergence towards the observed features. As demonstrated in section 4.4.2, when the feature values match there is also a good match between the learned and demonstrated kinematic signals.

The algorithm was stopped because no improvement could be realized anymore. It could be hypothesized that when real driver data is used where the human driver is less consistent in producing observations that resemble its underlying objective, the algorithm is stopped earlier. This will naturally constrict the algorithm in its feature matching accuracy due to no available direction of improvement. This can already be seen when more datasets are used in the conflict method e.g. 7 datasets. The algorithm stops then at 25 iterations because it reached earlier a point of no improvement. The learned weighting factors for 7 datasets are

$[14.785, 5.000, 3.223, 5.968, 3.770, 2.000]$.

Figure 4.11 gives the averaged error that each individual $f_{rel,i}$ vector makes with respect to convergence to one for the three important lateral features. The three important lateral features shown in Figure 4.11 are respectively the remaining lateral distance, lateral acceleration and lateral jerk. ND stands for number of datasets used.

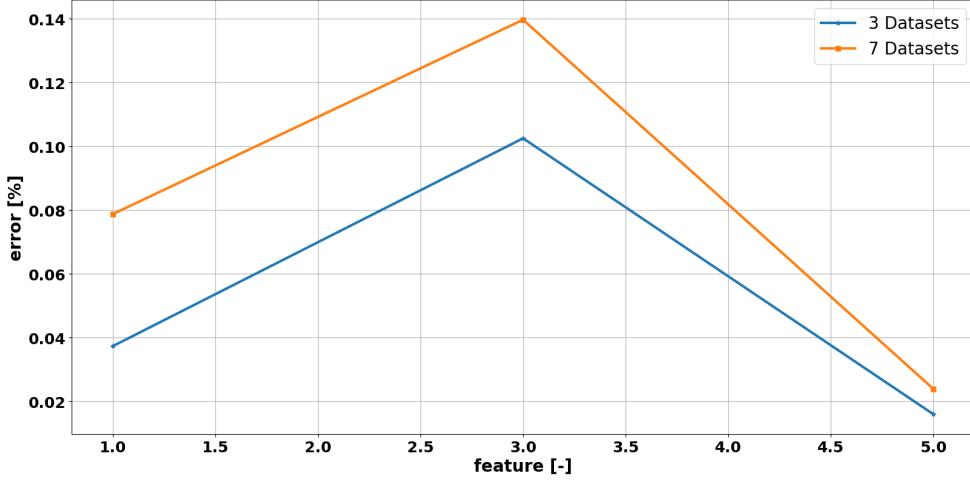


Figure 4.11: This figure shows the averaged, relative error given by $\frac{100 \cdot \sum_{n=1}^{ND} |1 - f_{rel,i}|}{ND}$ of the matching of the feature values for the three lateral features. (3 datasets: blue, 7 datasets: orange)

Figure 4.11 shows that the prematurely stop of learning with 7 datasets in comparison with 3 datasets gives an slightly higher error in the match of feature values of the individual datasets.

4.4.4 Comparison of methods

In this section a comparison is made between the averaging and conflict method of combining multiple datasets. Figure 4.12 shows that the averaging method learns weighting factors that can better explain the individual data feature vectors of the different datasets. Therefore this will be the method further used during this thesis.

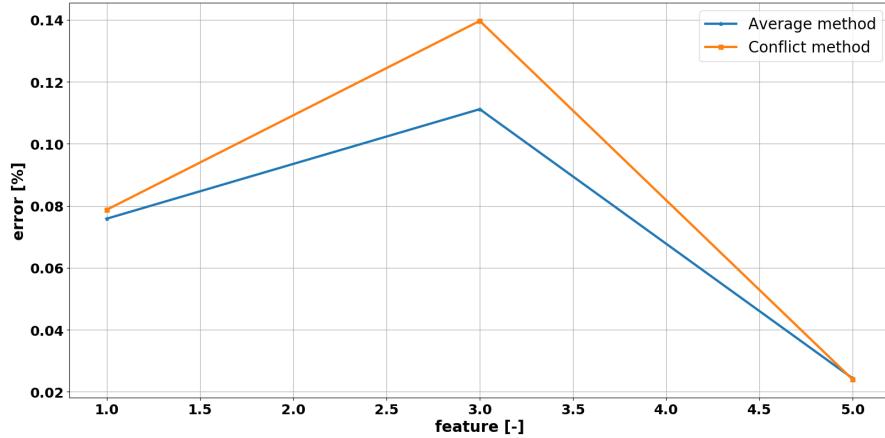


Figure 4.12: This figure shows the averaged, relative error given by $\frac{100 \cdot \sum_{n=1}^{ND} |1 - f_{rel,i}|}{ND}$ of the matching of the feature values of the three lateral features. (Averaging method: blue, conflict method: orange)

4.5 Conclusion

In this chapter the implementation of learning from ideal data was analysed. Data is considered ideal when it is produced with the same vehicle model as is used during the learning loop which calls the optimization Eq. (4.8). At the start of this chapter, the non-linear bicycle model was discussed and its parameters presented in table 4.1. Next, a step by step build up of the learning algorithm was shown. It was displayed how the ideal data is generated and validated. Out of this it followed that the choice of the number of control points is set on 1000 and the time limit on 30 s. Afterwards, the learning results were discussed and the conflict and averaging method were compared. It was concluded that for a lane change maneuver it is only important to learn the lateral weighting factors accurately in order to explain the observed data. Feature matching was also accomplished for the longitudinal features but there was a boundary on the accuracy due to the small longitudinal feature values attained during a lane change. Contrary to the lateral weighting factors, the longitudinal chosen weighting factors were not accurately found back. Because the conflict method gives for the same amount of datasets a slightly larger error on the ability to explain the feature values of the individual datasets and because of the hypothesis that this method will behave worse when used with real driver data, the averaging method is chosen to be further used in this thesis. The estimate of the gradient $\frac{\partial F_{diff}}{\partial \theta}$ by $F_{obs} - F(r_{expected})$ was found adequate in order to match the learned and observed feature values.

Chapter 5

Learning with complex vehicle model

Because the non-linear bicycle model makes abstraction of dynamics that are applicable in a real vehicle, a more complex model is introduced to improve the reality factor of the simulations. In order to achieve this the 15 degrees of freedom Amesim model as can be seen in Figure 5.1, is provided by Siemens. The parameters of this model are tuned by the company in order to behave similar to a test car they are currently using. The model has as inputs the amount of throttle, braking and steerwheelangle.

First, section 5.1 explains how the Amesim model will be integrated into learning the different weighting factors. Next, a tracking MPC needed for the implementation will be examined in section 5.2. Afterwards, the learning results will be discussed in section 5.3.

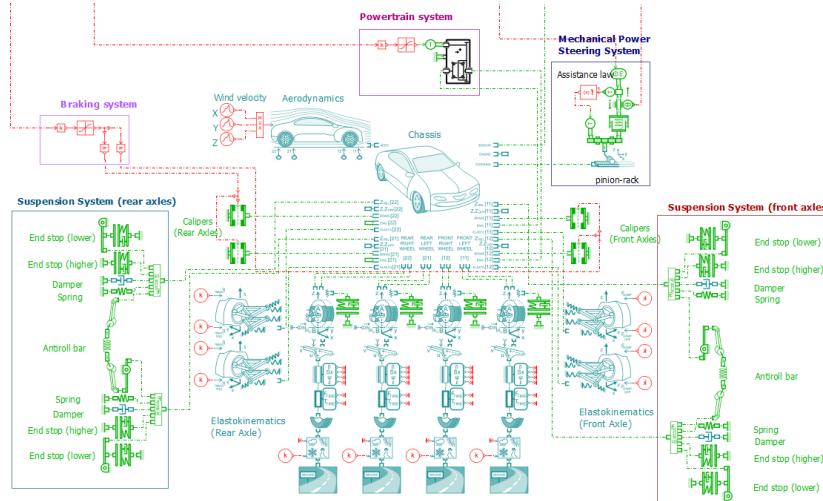


Figure 5.1: The 15 dof amesim model with as inputs the amount of throttle, braking and steerwheelangle.

5.1 Flow of the algorithm

The Amesim model serves as a black box and no explicit dynamic equations are available to include in Eq. (4.8) as was done with the non-linear bicycle model Eq. (4.3). Therefore a path is planned with the simpler non-linear bicycle model and afterwards a model predictive control tracking algorithm is applied with the Amesim model in order to follow the calculated reference. The working principles of a MPC is discussed in 2.2. Diagram 5.2 shows the flow of handlings that is done during the learning of the weighting factors with the Amesim model.



Figure 5.2: The flow of learning with the Amesim model.

As is described in Diagram 5.2 the feature vector $\mathbf{F}(\mathbf{r})$ is calculated based on a path tracked by the Amesim model and it is used in the convergence block to estimate the gradient $\frac{\partial \mathbf{F}_{diff}}{\partial \theta}$. The estimation is calculated by $\mathbf{F}^* - \mathbf{F}(\mathbf{r}_{expected})$. The reason why the path planned by the bicycle model has to be tracked is to avoid including vehicle mismatch error in the learning process. It is otherwise possible that even when the algorithm is able to match the feature vectors $\mathbf{F}(\mathbf{r})$ and $\mathbf{F}^*(\mathbf{r})$, coming from two different vehicle models, the learned kinematic signals of the bicycle model will not represent the observations that were produced by the Amesim model accurate enough. In order to check this, table 5.1¹ shows the different feature values for a lane change $V_0 : 22.22 \frac{m}{s}$, $L : 3.47 m$ generated with both the models.

Out of table 5.1 it can be concluded that the feature values are reasonably preserved when a trajectory is tracked by the Amesim model. However, although the feature values are not that far apart, not all kinematic signals of the bicycle and Amesim model will match accurately as can be seen in Appendix D. This will further be discussed in section 5.2.

¹Note that the feature values of the bicycle model are slightly different than the ones in table 4.3. This is because in the previous table the time limit was set on 30 s and here time limit is 25 s. This is done to have a better useable time discretization of 0.025 s to input as reference signal to the Amesim model.

Feature Value	Bicycle model	Amesim model
Nr.1	5.28e-8	3.33e-7
Nr.2	0.37	0.38
Nr.3	1.41e-7	1.20e-4
Nr.4	0.57	0.49
Nr.5	1.89e-6	9.40e-6
Nr.6	30.99	31.05

Table 5.1: This table shows the feature values of a lane change $V_0 : 22.22 \frac{m}{s}$, $L : 3.47 m$ for respectively the Bicycle and Amesim model.

This chapter also gives an answer on the question if the estimate of the gradient $\frac{\partial \mathbf{F}_{diff}}{\partial \theta}$ by $\mathbf{F}^*(\mathbf{r}) - \mathbf{F}(\mathbf{r}_{expected})$ is still sufficient in order to match the learned and observed feature values in section 5.3.

5.2 Tracking MPC

To be able to integrate the Amesim model in the learning process in an adequate manner, good tracking is desirable for the important lateral features that determine the lane change. Good tracking is therefore wanted for: $y(t)$, $a_y(t)$ and $j_y(t)$. The structure of this section is set up as follows: first the MPC formulation is discussed in section 5.2.1 and afterwards the tracking results are discussed in section 5.2.2.

5.2.1 MPC formulation

First the OCP that will be called during the MPC has to be defined. This is done by using the non-linear bicycle model defined with 10 states as seen in Eq. (4.2), inside the OCP formulation given by Eq. (5.1). It uses as objective an error function with respect to the reference what will assure following behaviour. The control horizon of the MPC is N_{MPC} and equal to 50 points which means a control horizon of 1.25 s because the reference is sampled with T_{pl} equal to 0.025 s. The parameters of the bicycle model stay the same as the ones listed in table 4.1. In order to formulate the gap closing constraints, which connect the previous states to the next in the chosen multiple shooting formulation, Runge-Kutta integration is embedded in function I . The first time that the OCP is solved and the current state is not yet given by the virtual sensors of the Amesim model, the current state is set equal to the first reference point of the bicycle model. The path constraints can be seen in Eq. (5.2). These are not necessary at first sight, but contribute by decreasing the feasible solution space of the solver. It is checked that these constraints are not binding, which means that the found solution is reachable even if the constraints were removed.

$$\begin{aligned}
 & \min_{\mathbf{X}(\cdot), \mathbf{U}(\cdot)} E(\mathbf{X}(\cdot), \mathbf{U}(\cdot)) \\
 \text{s.t. } & \mathbf{X}^{k+1} = I(\mathbf{X}^k, \mathbf{U}^k) \quad k = [0, \dots, N_{MPC} - 1] \\
 & \mathbf{X}^0 = \mathbf{X}_{current} \\
 & \mathbf{F}(\mathbf{X}^k) \geq 0 \quad k = [0, \dots, N_{MPC}] \\
 & \mathbf{X}^k \in \mathbb{R}^{10 \times 1} \quad k = [0, \dots, N_{MPC}] \\
 & \mathbf{U}^k \in \mathbb{R}^{2 \times 1} \quad k = [0, \dots, N_{MPC} - 1] \\
 & N_{MPC} \in \mathbb{N}
 \end{aligned} \tag{5.1}$$

$$\mathbf{F} = \left\{
 \begin{array}{ll}
 -\frac{\text{Width Lane}}{2} \leq y^k \leq \frac{3 \cdot \text{Width Lane}}{2}, & k = [0, \dots, N_{MPC}] \\
 0 \leq x^k, & k = [0, \dots, N_{MPC}] \\
 -\frac{\pi \cdot 5}{180} \leq \psi^k \leq \frac{\pi \cdot 5}{180}, & k = [0, \dots, N_{MPC}] \\
 v_{x,start} - 1 \leq v_x^k \leq v_{x,start} + 1, & k = [0, \dots, N_{MPC}]
 \end{array}
 \right\} \tag{5.2}$$

The error function that serves as the objective of the OCP is given by the following equation. The tuning parameters that gave the best tracking results are shown in table 5.2.

$$\begin{aligned}
 \text{objective} = & W_1(\mathbf{x}[2:end] - \mathbf{ref}_x)^T (\mathbf{x}[2:end] - \mathbf{ref}_x) + W_2(\mathbf{y}[2:end] - \mathbf{ref}_y)^T (\mathbf{y}[2:end] - \mathbf{ref}_y) \\
 & + W_3(\mathbf{v}_x[2:end] - \mathbf{ref}_{v_x})^T (\mathbf{v}_x[2:end] - \mathbf{ref}_{v_x}) + W_4(\mathbf{v}_y[2:end] - \mathbf{ref}_{v_y})^T (\mathbf{v}_y[2:end] - \mathbf{ref}_{v_y}) \\
 & + W_5(\dot{\psi}[2:end] - \mathbf{ref}_\psi)^T (\dot{\psi}[2:end] - \mathbf{ref}_\psi) + W_6(\ddot{\psi}[2:end] - \mathbf{ref}_{\dot{\psi}})^T (\ddot{\psi}[2:end] - \mathbf{ref}_{\dot{\psi}}) \\
 & + W_7 \dot{\mathbf{t}}_r^T \dot{\mathbf{t}}_r + W_8 \dot{\boldsymbol{\delta}}_s^T \dot{\boldsymbol{\delta}}_s + W_9 \mathbf{a}_x^T \mathbf{a}_x
 \end{aligned}$$

$$\mathbf{x}, \mathbf{y}, \mathbf{v}_x, \mathbf{v}_y, \boldsymbol{\psi}, \dot{\boldsymbol{\psi}}, \mathbf{a}_x \in \mathbb{R}^{(N_{MPC}+1) \times 1} \quad \mathbf{ref}_i, \dot{\mathbf{t}}_r, \dot{\boldsymbol{\delta}}_s \in \mathbb{R}^{N_{MPC} \times 1}$$

Parameter	Value
W1	10
W2	10
W3	30
W4	1.0
W5	100
W6	1.0
W7	5.0
W8	0.01
W9	0.01

Table 5.2: Overview of the tuning parameters used in the objective of Eq. (5.1).

The tuning parameters displayed were attained by trial and error but there is an intuitive explanation for why these states were used and which order of magnitude they were given.

It was first tried to focus on path tracking in order to see how the other states would differ when the Amesim model drove exactly the same path as the non-linear vehicle and therefore $x(t), y(t)$ and $\psi(t)$ were included. They define the orientation of the vehicle. Nervous inputs² behaviour was noticed and in order to smooth the input, they were given a small weighting factor. Only a small weighting factor was given to assure good tracking behaviour. It was observed that this strategy resulted in a good tracking of the important signals for the learning algorithm: $y(t), a_y(t)$ and $j_y(t)$. As will be explained in 5.2.2 the Amesim model made a deceleration in the longitudinal direction at the begin of the maneuver. In order to give the model time to stabilize before the start of the maneuver, a straight driving part of 15 s and 2.5 s was respectively added before and after the reference lane change maneuver outputted by Eq. (4.8). To remove the oscillation faster of the longitudinal signals, v_x and a_x were included in the error function.

The initial guess when the OCP is solved for the first time, is only needed for v_x . This is because otherwise an invalid value would emerge in the calculation of the slip angle according to Eq. (4.6). When the OCP is defined and implemented in CasADi, the opti environment is saved as a function that is called during the MPC. To improve the solving time of Eq. (5.1), the solver is switched to a SQP method using an active-set QP solver instead of IPOPT as was used in Eq. (4.8). "Interior point methods (like IPOPT) are generally robust at finding a minimizer, but the barrier parameter will make the algorithm walk away from a perfect initial guess, only to come back after a while." [10] A hot starting was implemented by feeding the solution of a previous MPC iteration as initial guess for the optimization variables and the lambda multipliers which made it logical to switch the solver and solving method. Together with the straight driving parts, a simulation of 40 s is performed and one iteration of the MPC where one OCP is solved, takes around 0.15 s. The main time consumed to calculate the solution is mainly the loading of the different simulation components, before the actual running of the MPC.

With the OCP defined, it can be included in a MPC formulation. Every MPC iteration uses the current state of the Amesim model and a part of the reference that pertains to the current control horizon. The Amesim model is evaluated at a sampling rate of $T_s = 0.01$ s and the optimization Eq. (5.1) is called at a rate of $T_{MPC} = 0.1$ s. The reference has a sampling rate of $T_{pl} = 0.025$ s. In order to connect the tracking MPC with the Amesim model using different sampling times, the Simulink model of Figure 5.3 was build.

The three inputs to the Amesim model (largest block) in Figure 5.3 are δ_s , t_r and

²The meaning of nervous inputs is very fast changing and therefore oscillating input behaviour.

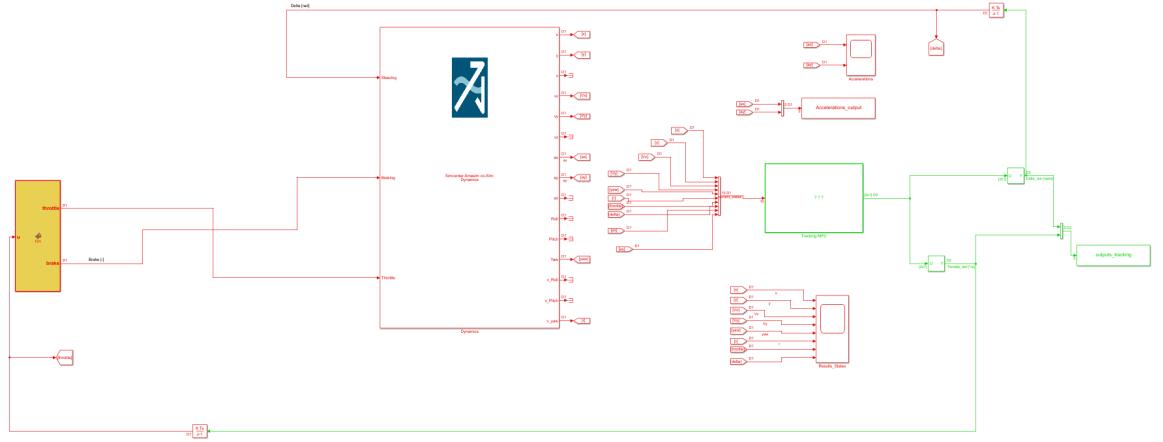


Figure 5.3: This figure shows hows the tracking MPC is connected with the Amesim model using different sampling times. (red: $T_s : 0.01 \text{ s}$ and green: $T_{MPC} : 0.1 \text{ s}$)

braking but the control outputs of Eq. (5.1) give \dot{t}_r and $\dot{\delta}_s$. Therefore 'Forward Euler integration' blocks are introduced using for 0.1 s the control outputs of Eq. (5.1) in order to calculate every 0.01 s the inputs for the Amesim model. The throttle of the bicycle model can theoretically become positive and negative. These two states were separated and feeded to the Amesim model as throttling (positive) or braking (negative).

5.2.2 Tracking results

In this section the results of the tracking MPC are presented. For a full overview of the results, reference is made to Appendix D. Figure 5.4 shows that the Amesim model (blue) can follow the planned reference path (red) with an accuracy of 10^{-3} m . Analysing the results showed that the important signals were followed sufficiently well to be able to use the tracking MPC in the learning configuration of Figure 5.2. This could also be seen in table 5.1 that gave similar feature values for the Amesim model and its reference. However there are also differences between the reference and the followed trajectory. This justifies the reasoning that not all states have to be included in the error function of Eq. (5.1). Tracking all states accurately will give worse results due to inherent vehicle model mismatch.

The first difference is that the Amesim model needs a bigger steerwheelangle than was predicted by the reference. Also the amount of throttle needed to drive forward at a constant speed is slightly different because of a more complete way of aerodynamic resistance modelled by the Amesim model in comparison to Eq. (4.4) used by the bicycle model. Another observation is that the lateral jerk and first derivative of the steerwheelangle display less high peaks. Also, when zooming in closely on the longitudinal jerk around 15 s , which coincides with the start of the lane change and 22 s which is more at the end, nervous behaviour of the longitudinal jerk is witnessed.

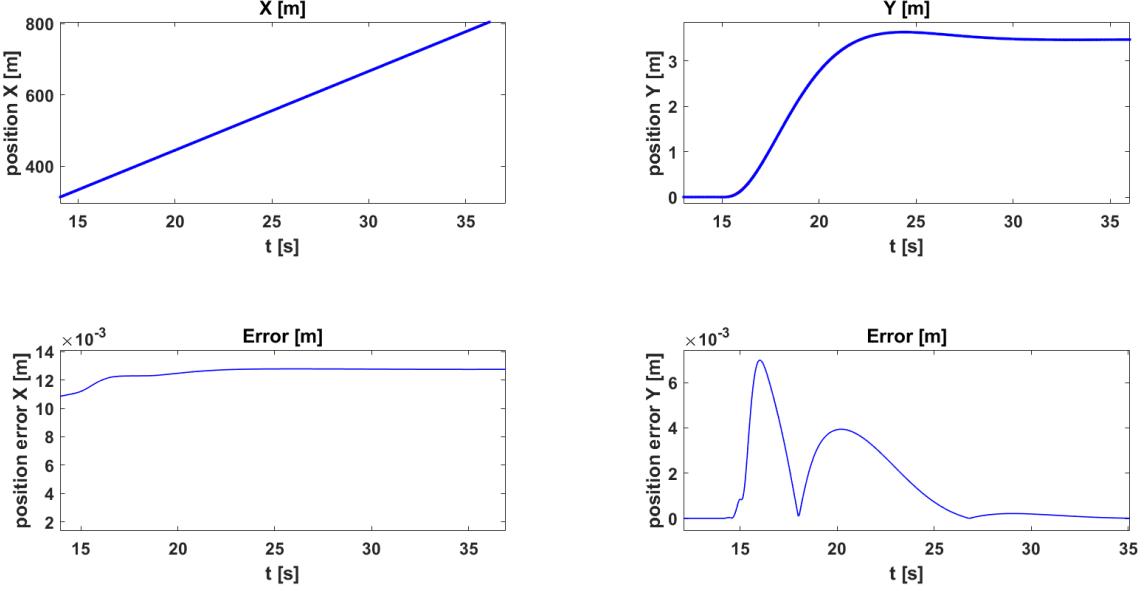


Figure 5.4: This figure shows the tracking result of the reference (red) by the Amesim model (blue).

Around 15 s a bump in the reference of the throttle can be noticed because that is the point where straight driving at constant speed ends and the reference is from that point onwards produced by a solution of Eq. (4.8), which will differ from a constant value. The influence on v_x and a_x is however marginal. Further it is worth to note that the jerks don't come directly from the Amesim model but the signal is post-processed by numerical differentiation.

To be able to generate adequate jerk signals, it was necessary to include smooth inputs of throttle and steerwheelangle. Figure 5.5 shows oscillating behaviour of the lateral jerk when not the first derivative of throttle and steerwheelangle are used as control outputs of Eq. (5.1). That the jerk signal is dependent on the first derivative of throttle and steeringwheelangle is a logical result, because these variables were also embedded in the analytical jerk equations of the non-linear bicycle model in Appendix A.³

³For the results of Figure 5.5, no straight driving reference path is added. The lane change directly starts at *time* = 0.0.

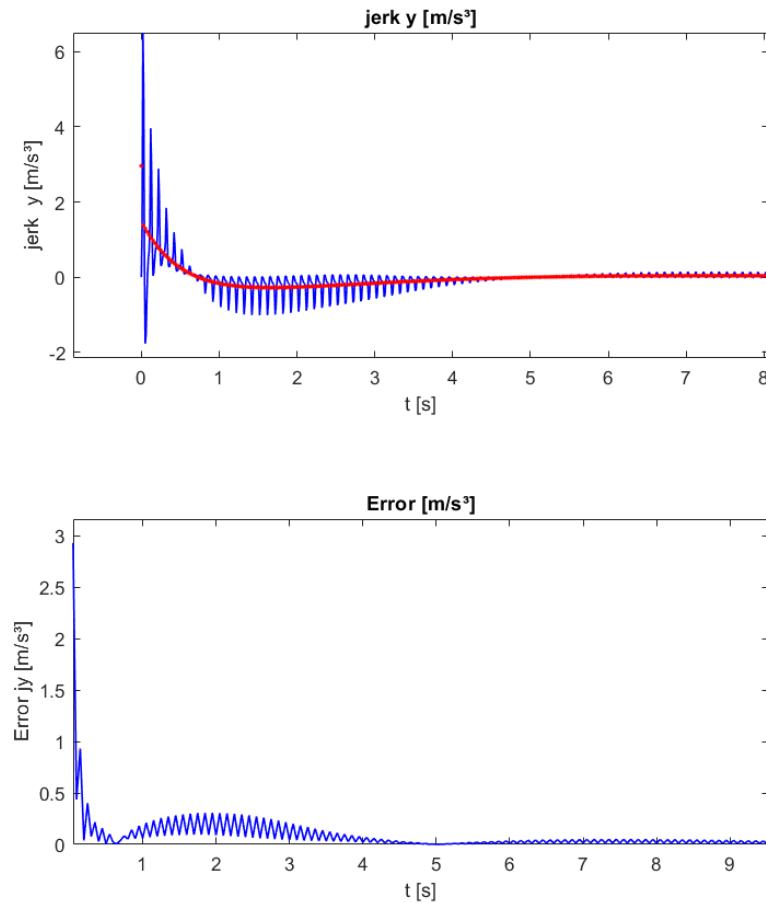


Figure 5.5: This figure shows the obtained oscillating signals for the jerk when a piecewise signal of δ_s and t_r is applied.

5.3 Learning results with the Amesim model

Diagram 5.2 shows in blue the observed path that is generated out of Eq. (4.8) with known weighting factors whereafter the path based on a bicycle model is tracked by a tracking MPC to get the kinematic signals of the 15 dof Amesim model. From this $\mathbf{F}^*(\mathbf{r})$ is calculated which stays constant during the whole learning process.

The learning loop shown in red in diagram 5.2 starts at the lane change planner, where Eq. (4.8) is called with an initial guess of the weighting factors an all-one vector. After the planned path is outputted, tracked⁴ and the associated feature vector $\mathbf{F}(\mathbf{r})$ is calculated, it is checked if the lateral feature values match accurate enough in the convergence block. If this is not the case, the difference of the features is taken as an estimate for $\frac{\partial \mathbf{F}_{diff}}{\partial \theta}$ and used in the RPROP algorithm in order to

⁴The first 10 s of the tracked Amesim signal is removed. The remaining maneuver consists of 5 s straight driving before doing the actual lane change.

5.3. Learning results with the Amesim model

generate a new planned path by calling Eq. (4.8).

Four simulations are conducted with respectively one, two, three and five datasets. Table 5.3 shows which longitudinal speed is driven at the begin of the lane change (V_0) and the desired lateral displacement (L). As is noted before in Chapter 4, if only the start speed is varied, the lateral feature values stay the same and the longitudinal features stay very small. This means that these datasets are seen as almost the same when learning. When the lateral desired displacement is changed, this gives a clear difference in the lateral feature values. An overview of the different features of the individual datasets is given in table 5.4.⁵

1 Dataset	2 Datasets	3 Datasets	5 Datasets
$V_0 : 22.22 - L : 3.47$	$V_0 : 22.22 - L : 3.47$ $V_0 : 25.00 - L : 6.94$	$V_0 : 22.22 - L : 3.47$ $V_0 : 25.00 - L : 6.94$ $V_0 : 27.78 - L : 3.47$	$V_0 : 22.22 - L : 3.47$ $V_0 : 25.00 - L : 6.94$ $V_0 : 27.78 - L : 3.47$ $V_0 : 22.22 - L : 6.94$ $V_0 : 25.00 - L : 3.47$

Table 5.3: This table shows which lane changes were used during the four different simulations.

Feature	V022.22 - L3.47	V022.22 - L6.94	V025.00 - L3.47	V025.00 - L6.94	V027.78 - L3.47
Nr.1	5.18e-07	7.20e-06	3.88e-07	5.97e-06	4.01e-07
Nr.2	0.38	1.50	0.38	1.51	0.38
Nr.3	12.29e-05	14.53e-05	1.67e-05	7.23e-05	5.37e-06
Nr.4	0.51	2.06	0.52	2.09	0.52
Nr.5	9.50e-06	4.42e-05	1.66e-05	7.08e-05	3.71e-05
Nr.6	91.25	364.99	91.25	364.97	91.25

Table 5.4: This table shows the different features for the individual datasets used.

The weighting factors learned for the four different simulations is presented in table 5.5⁶ and f_{rel} at convergence is given in table 5.6. f_{rel} displays the matching of the averaged learned feature values with the averaged data feature vector. Convergence for the first three simulations is reached after 28 iterations and for 5 datasets after 26 iterations. The convergence criteria used, expects a match of the lateral features (2,4,6) with a tolerance of 10^{-3} .

⁵In the calculation of the feature values the 5 s straight driving before the lane change is included. This contributes to a higher value for feature number six than seen earlier in table 5.1

⁶The weighting factors that come from the simulations are scaled so that the second weighting factor equals 5.0.

Weight	1 Dataset	2 Datasets	3 Datasets	5 Datasets
Nr.1	33.4230	5.2294	3.9036	4.2995
Nr.2	5.0000	5.0000	5.0000	5.0000
Nr.3	2.8662e-06	1.0636	1.9803	3.4930
Nr.4	6.0018	6.0018	6.0018	6.0278
Nr.5	4.4445	1.2705	1.1869	1.5694
Nr.6	2.0011	2.0011	2.0011	2.0044

Table 5.5: This table shows the weighting factors learned for the different simulations that start from an all-one vector and uses as chosen weighting factors [4.0, 5.0, 1.0, 6.0, 1.0, 2.0].

f_{rel}	1 Dataset	2 Datasets	3 Datasets	5 Datasets
Nr.1	1.1315	1.0097	0.9643	0.9677
Nr.2	1.0000	1.0002	1.0003	1.0005
Nr.3	0.2865	0.4496	0.4867	0.9957
Nr.4	1.0000	1.0002	1.0002	0.9993
Nr.5	1.0718	1.0030	0.9937	0.9772
Nr.6	1.0000	1.0000	1.0000	1.0000

Table 5.6: This table shows the averaged f_{rel} at convergence.

Out of table 5.5 and 5.6 it is concluded that the lateral chosen weighting factors are found back and an accurate match of the lateral features is achieved. It is also observed that the longitudinal weighting factors will come closer to their chosen value and give a better match for the longitudinal feature values, when more observed lane changes is included in the learning. Figure 5.6 presents the observed paths, the initial solution when the weighting factor vector is chosen equal to an all-one vector and the learned solution when 2 datasets are used. Figure 5.7 shows the error made between the observed and learned path which is in order of magnitude of $10^{-4}m$. All the resulting kinematic signals are displayed in Appendix E.

Figure 5.8 shows the convergence of the averaged f_{rel} towards one during the learning iterations when 5 datasets are used. Except for the longitudinal jerk (feature 3) all the learned feature values converge towards the observed ones. From the retrieved averaged f_{rel} in table 5.6 and in Figure 5.8, the feature value of the longitudinal jerk (Nr. 3), stands out for its bad feature matching behaviour. Table 5.5 gives the impression that there is a good feature matching when 5 datasets are used, but as can be seen in Figure 5.8, the relative feature value 3 still cycles around one. A reason for this is, as discussed in 5.2.2, bad quality of the jerk data which contributes to a deteriorated learning. The jerk during the lane change can be seen in Figure 5.9. It is good to note that the longitudinal jerk signal has a small order of magnitude, what the marginal effect on the lane change behaviour explains. At time zero the

5.3. Learning results with the Amesim model

longitudinal behaviour is not fully stabilized and at second 5 the lane change starts.

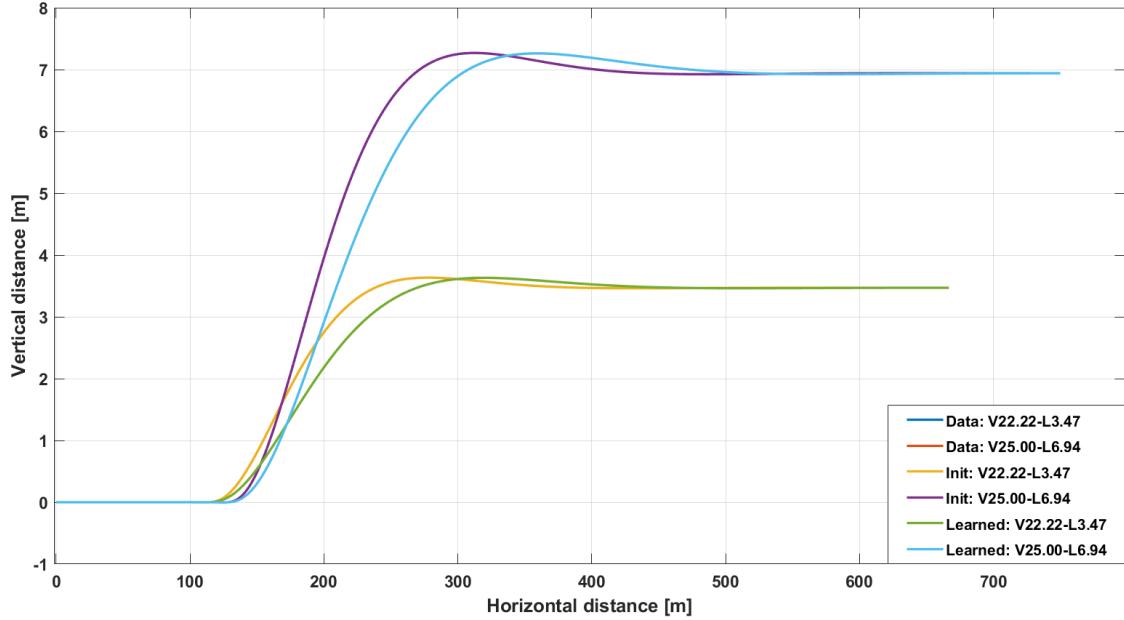


Figure 5.6: This figure shows the observed paths, the initial solution retrieved with as weighting factors an all one vector and the learned solution.

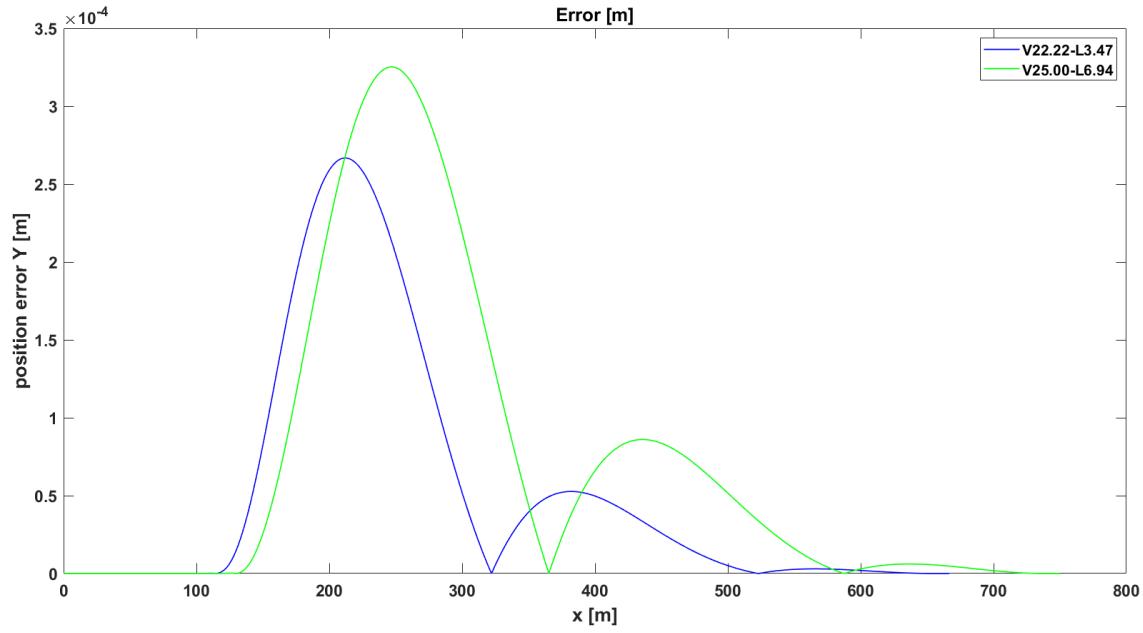


Figure 5.7: This figure shows the error made between the observed and learned paths.

5. LEARNING WITH COMPLEX VEHICLE MODEL

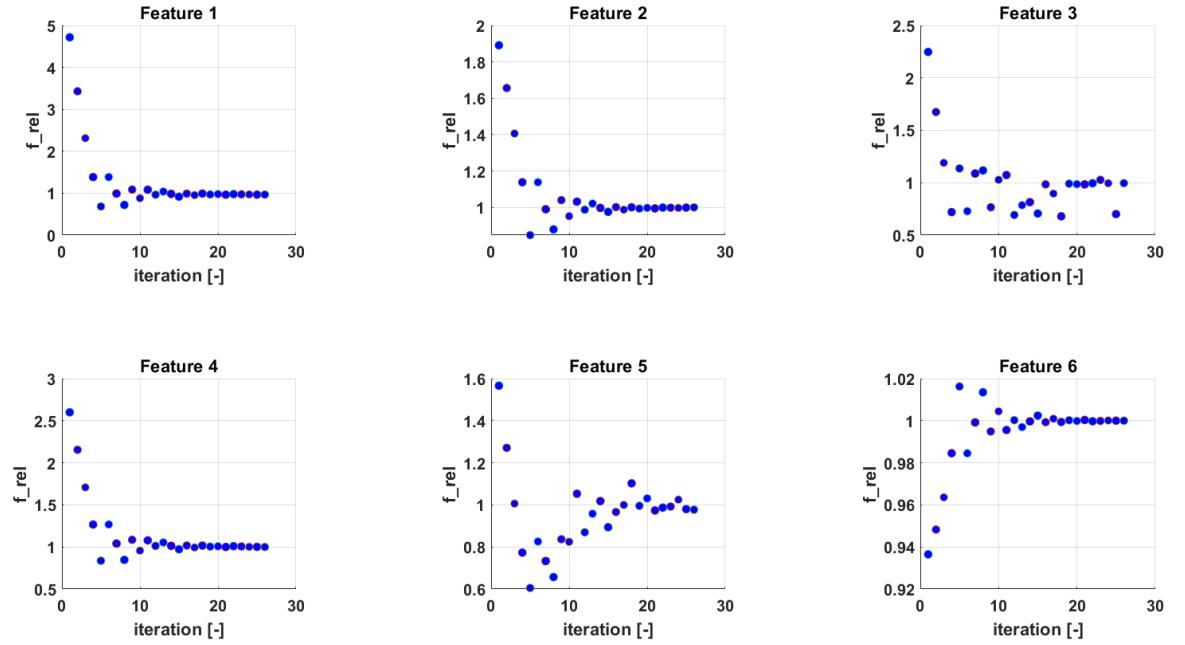


Figure 5.8: This figure shows the averaged f_{rel} during learning of five datasets.

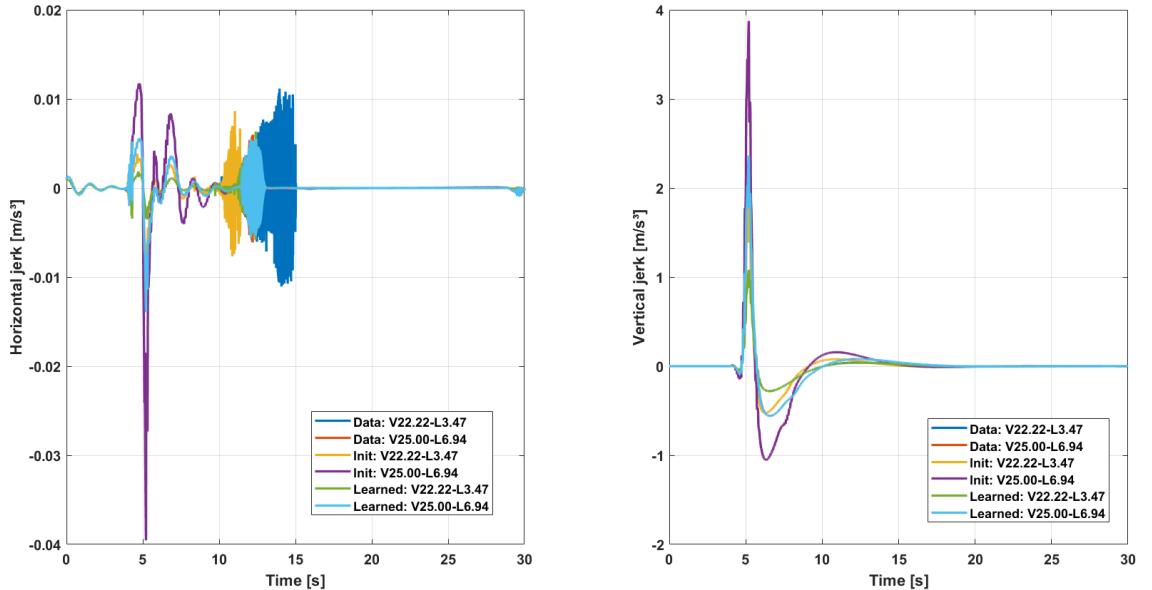


Figure 5.9: This figure shows the observed jerk signals, the initial solution retrieved with as weighting factors an all-one vector and the learned solution.

5.4 Conclusion

In this chapter the learning of driver specific comfort parameters with a realistic 15 dof Amesim model was discussed. The flow of the different steps of the learning process and how the Amesim model is embedded, was looked into. The question posed at the beginning was: "Is it possible to find back the underlying comfort weighting factors when the gradient $\frac{\partial F_{diff}}{\partial \theta}$ is estimated by $\mathbf{F}^* - \mathbf{F}(\mathbf{r}_{expected})$?" It was shown that this estimation is sufficient in order to learn weighting factors that allow matching between an averaged learned feature vector and an averaged data feature vector.

First the formulation of the used tracking MPC was presented and its tracking performance evaluated. The results are that tracking of the planned path was possible to an accuracy of $10^{-3}m$ and also the other important features a_y and j_y could be tracked to a satisfying accuracy as can be seen in Appendix D. However δ_s , t_r in steady-state and the peaks of δ_s and j_y , gave a difference with the reference when the path generated by the non-linear bicycle model is being accurately tracked. Next a notion was made about the bad quality of the obtained longitudinal jerk signal. Eventually it was stated that only a smooth signal for the lateral jerk can be retrieved when the input signals are at least of first order.

After the tracking MPC was validated, the learning results with the Amesim model were discussed. Five simulations were conducted with a different amount of observed lane changes. Out of the results it could be concluded that for the lateral features there is a good convergence found for $f_{rel,i}$ and the chosen lateral weighting factors are found back every time. When more observed lane changes are included during the learning, the results for the longitudinal weighting factors and the feature matching of the longitudinal feature values improves. Only the convergence of the longitudinal jerk stayed deficient, which is due to bad quality of the longitudinal jerk data that was used to learn the weighting factors from.

Chapter 6

Enhanced weighting factor update

This chapter theoretically describes an alternative for updating $\boldsymbol{\theta}^k$ during the learning iterations. The previous method made use of the gradient approximation $\mathbf{F}_{obs} - \mathbf{F}(\mathbf{r}_{expected})$ embedded in a gradient descent formulation and the step size was determined by the RPROP algorithm. The method proposed here, calculates $\boldsymbol{\theta}^{k+1}$ based on an unconstrained optimization.

The chapter is structured as follows. First, the formulation is analysed in section 6.1, whereafter a discussion follows in section 6.2. Finally a conclusion is given in section 6.3.

6.1 Formulation

The goal of the learning algorithm is to find a set of weight factors $\boldsymbol{\theta}$ that will match the feature vector \mathbf{F} as closely as possible to the observed feature vector \mathbf{F}_{obs} . From this the unconstrained minimization of the error function of the feature vectors match, follows as defined by Eq. (6.1)

$$E(\boldsymbol{\theta}) = |f_{obs,1} - f_1(\boldsymbol{\theta})| + |f_{obs,2} - f_2(\boldsymbol{\theta})| + |f_{obs,3} - f_3(\boldsymbol{\theta})| + |f_{obs,4} - f_4(\boldsymbol{\theta})| + |f_{obs,5} - f_5(\boldsymbol{\theta})| + |f_{obs,6} - f_6(\boldsymbol{\theta})|. \quad (6.1)$$

However $\mathbf{F}(\boldsymbol{\theta})$ is not known analytically and therefore should be approximated by a second order multivariate Taylor expansion as described by Eq. (6.2). The solution of the resulting minimization of Eq. (6.1) is taken as $\boldsymbol{\theta}^{k+1}$. The gradient and the hessian are respectively given by Eq. (6.3) and Eq. (6.4),

$$f_i(\boldsymbol{\theta}) \approx f_i(\boldsymbol{\theta}^k) + \nabla f_i(\boldsymbol{\theta})^T|_{\boldsymbol{\theta}=\boldsymbol{\theta}^k} \cdot (\boldsymbol{\theta} - \boldsymbol{\theta}^k) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}^k)^T \cdot \mathbf{H}_i(\boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}^k} \cdot (\boldsymbol{\theta} - \boldsymbol{\theta}^k), \quad (6.2)$$
$$\forall i, k \in \mathbb{N}$$

$$\nabla f_i(\boldsymbol{\theta})^T = \left[\frac{\partial f_i}{\partial \theta_1} \quad \frac{\partial f_i}{\partial \theta_2} \quad \frac{\partial f_i}{\partial \theta_3} \quad \frac{\partial f_i}{\partial \theta_4} \quad \frac{\partial f_i}{\partial \theta_5} \quad \frac{\partial f_i}{\partial \theta_6} \right] \quad (6.3)$$

$$\mathbf{H}_i(\boldsymbol{\theta}) = \begin{bmatrix} \frac{\partial^2 f_i}{\partial \theta_1^2} & \frac{\partial^2 f_i}{\partial \theta_1 \partial \theta_2} & \frac{\partial^2 f_i}{\partial \theta_1 \partial \theta_3} & \frac{\partial^2 f_i}{\partial \theta_1 \partial \theta_4} & \frac{\partial^2 f_i}{\partial \theta_1 \partial \theta_5} & \frac{\partial^2 f_i}{\partial \theta_1 \partial \theta_6} \\ \frac{\partial^2 f_i}{\partial \theta_2 \partial \theta_1} & \frac{\partial^2 f_i}{\partial \theta_2^2} & \frac{\partial^2 f_i}{\partial \theta_2 \partial \theta_3} & \frac{\partial^2 f_i}{\partial \theta_2 \partial \theta_4} & \frac{\partial^2 f_i}{\partial \theta_2 \partial \theta_5} & \frac{\partial^2 f_i}{\partial \theta_2 \partial \theta_6} \\ \frac{\partial^2 f_i}{\partial \theta_3 \partial \theta_1} & \frac{\partial^2 f_i}{\partial \theta_3 \partial \theta_2} & \frac{\partial^2 f_i}{\partial \theta_3^2} & \frac{\partial^2 f_i}{\partial \theta_3 \partial \theta_4} & \frac{\partial^2 f_i}{\partial \theta_3 \partial \theta_5} & \frac{\partial^2 f_i}{\partial \theta_3 \partial \theta_6} \\ \frac{\partial^2 f_i}{\partial \theta_4 \partial \theta_1} & \frac{\partial^2 f_i}{\partial \theta_4 \partial \theta_2} & \frac{\partial^2 f_i}{\partial \theta_4 \partial \theta_3} & \frac{\partial^2 f_i}{\partial \theta_4^2} & \frac{\partial^2 f_i}{\partial \theta_4 \partial \theta_5} & \frac{\partial^2 f_i}{\partial \theta_4 \partial \theta_6} \\ \frac{\partial^2 f_i}{\partial \theta_5 \partial \theta_1} & \frac{\partial^2 f_i}{\partial \theta_5 \partial \theta_2} & \frac{\partial^2 f_i}{\partial \theta_5 \partial \theta_3} & \frac{\partial^2 f_i}{\partial \theta_5 \partial \theta_4} & \frac{\partial^2 f_i}{\partial \theta_5^2} & \frac{\partial^2 f_i}{\partial \theta_5 \partial \theta_6} \\ \frac{\partial^2 f_i}{\partial \theta_6 \partial \theta_1} & \frac{\partial^2 f_i}{\partial \theta_6 \partial \theta_2} & \frac{\partial^2 f_i}{\partial \theta_6 \partial \theta_3} & \frac{\partial^2 f_i}{\partial \theta_6 \partial \theta_4} & \frac{\partial^2 f_i}{\partial \theta_6 \partial \theta_5} & \frac{\partial^2 f_i}{\partial \theta_6^2} \end{bmatrix}. \quad (6.4)$$

In order to investigate the local behaviour of a certain feature value f_i in function of $\boldsymbol{\theta}$, the weight factors are slightly varied around $f_i(\boldsymbol{\theta}^k)$. This is done as described by Eq. (6.5) for weight θ_j

$$\theta_j^{k,+} = \theta_j^k + \Delta\theta, \quad (6.5a)$$

$$\theta_j^{k,-} = \theta_j^k - \Delta\theta, \quad (6.5b)$$

$$\frac{\partial f_i}{\partial \theta_j^k} = \frac{f_i|_{\theta_j^{k,+}} - f_i|_{\theta_j^{k,-}}}{2\Delta\theta}, \quad (6.6a)$$

$$\frac{\partial^2 f_i}{\partial \theta_{k,j}^2} = \frac{f_i|_{\theta_j^{k,+}} - 2f_i|_{\theta_j^k} + f_i|_{\theta_j^{k,-}}}{\Delta\theta^2}. \quad (6.6b)$$

For every evaluation of f_i , Eq. (4.8) is called and outputs six feature values. The solution of the optimization variables and lambda multipliers are known when Eq. (4.8) was solved with weight factors equal to $\boldsymbol{\theta}^k$. The solution of the previous iteration is used as initial guess which only differ with a very small value $\Delta\theta$ when Eq. (4.8) is called to solve $f_i|_{\theta_j^{k,+}}$. It is expected that when the SQP method is used, the good initial guess is fully exploited and convergence is rapidly reached.

Eq. (4.8) is called 12 times in order to estimate the 6 gradients of the different features and the diagonals of the six Hessians, when also making use of the during the previous iteration attained result for $f_i|_{\theta_j^k}$.

The hessian is assumed to be symmetrical, which means that only the cross derivates above the diagonal have to be calculated. The calculation of the cross derivatives can become quite complex, but an intuitive approach relies on subsequent use of one-dimensional finite-difference discretization as in the example given by Eq. (6.7).

This formulation is second order when the mesh is equidistant, orthogonal and with p and z the mesh-point numbering [14]

$$\frac{\partial^2 \phi}{\partial x \partial y} \approx \frac{\frac{\partial \phi}{\partial y}|_{p+1,z} - \frac{\partial \phi}{\partial y}|_{p-1,z}}{2\Delta x}. \quad (6.7)$$

$$p, z \in \mathbb{N}$$

This translates with the use of Eq. (6.6a) into Eq. (6.8)

$$\frac{\partial^2 f_i}{\partial \theta_p^k \partial \theta_z^k} \approx \frac{f_i|_{\theta_p^{k,+}, \theta_z^{k,+}} - f_i|_{\theta_p^{k,+}, \theta_z^{k,-}} - f_i|_{\theta_p^{k,-}, \theta_z^{k,+}} + f_i|_{\theta_p^{k,-}, \theta_z^{k,-}}}{4(\Delta \theta)^2}. \quad (6.8)$$

The conditions of an equidistant mesh is fulfilled when $\Delta \theta$ is taken the same for the different weights. The condition of an orthogonal mesh is automatically satisfied because the size of a weight factor can change independently of the other weight factors.

Because during the calculation of every cross derivative Eq. (4.8) is called 4 times, this means that in order to calculate the full Hessian a total of 72 evaluations of Eq. (4.8) are needed. After this, the error function of Eq. (6.1) can be minimized while making use of an approximated $\mathbf{F}(\boldsymbol{\theta})$.

6.2 Discussion

The method proposed is clearly more calculation expensive than the original gradient descent method with the approximated gradient. Therefore future steps are to validate if evaluations of Eq. (4.8) are indeed very fast retrieved when using a SQP method with a very good initial guess. In order to reduce calculation time, there can be looked at hessian approximations or first order Taylor expansion can be used.

6.3 Conclusion

In this chapter a short introduction was given about possible future improvements on updating the weight factors during the learning algorithm. It was proposed to use a multivariate Taylor expansion in order to approximate $\mathbf{F}(\boldsymbol{\theta})$. The next weight factors follow out of the best possible fit with the observed feature values.

Chapter 7

Conclusion

The contribution made with this thesis is the development of an algorithm based on inverse optimal control, that can learn driver specific experiences of comfort during a lane change captured in weighting factors of an objective function.

First a literature study was conducted in order to identify comfort during driving. Here it was found that smooth paths play an important role in contributing to the retrieved amount of comfort of a human driver. Next, it was looked at the theoretical idea of feature-based inverse optimal control. This was the starting point of this thesis and the goal was to implement this theoretical idea for in practise useable models e.g. 15 dof Amesim model.

In the following chapter, learning from ideal data was discussed. Here it was presented how ideal data was generated and validated and it was shown that the developed algorithm was able to accurately learn the chosen lateral weighting factors that define the lane change maneuver. Thereby it was displayed that the estimation of the gradient $\frac{\partial \mathbf{F}_{diff}}{\partial \boldsymbol{\theta}}$ by $\tilde{\mathbf{F}} - \mathbf{F}(\mathbf{r}_{expected})$ can be used in order to converge towards the data feature vectors used.

Furthermore, learning with a 15 degrees of freedom vehicle model was analysed. For this a tracking MPC formulation was developed and it was shown that also here the algorithm was able to accurately learn the chosen lateral weighting factors while the learned feature values converge towards the data feature values.

As last the theoretical idea of an improved method of updating the weighting factor vector $\boldsymbol{\theta}$ during the learning iterations, was discussed.

This work has its practical usage in the avoidance of manual tuning of a comfort objective which when identified, can be used for path planning. Because this research is conducted together with Siemens, it is in line with their research and this thesis can be used to build further on.

The maneuver discussed is a lane change maneuver. The only way that this is defined in Eq. 4.8 is by setting constraints on the end state Eq. (4.10), so that a lane change is performed. It is thereby possible to change these end constraints so that also other maneuvers are learned e.g. an acceleration maneuver.

During this thesis the environment is not assessed although this is a necessary

7. CONCLUSION

condition in order to take comfort into account as followed out of section 3.1. As discussed in section 4.2.2, it is possible to add features that encapsulate notions about the environment of the autonomous vehicle so that the feeling of driving safe can be assured. Therefore a suggestion for future work is to perform the developed algorithms on real driver data which also contains information about the vehicle its surroundings and to further validate the theoretical weighting factor update concept, proposed in chapter 6.

Appendices

Appendix A

Jerk equations of the non-linear bicycle model

In this section the jerk equations that were derived from the non-linear bicycle model and used in the analytical learning algorithm are displayed.

A.1 Equations

The equations of jerk in function of the non-linear vehicle states is the derivate of the total acceleration of the centre of gravity. The undermentioned equations were validated with 'MUPAD', a symbolic toolbox in Matlab.

$$\begin{aligned} j_{x,t} = & \frac{1}{M} \cdot (-\sin\delta t_r c \dot{\delta} + \cos\delta \dot{t}_r c + \cos\delta 2K_{y,f} \arctan\left(\frac{v_y + \omega a}{v_x}\right) \dot{\delta} \\ & + \sin\delta 2K_{y,f} \frac{v_x a_{t,y} + v_x \dot{\omega} a - a_{t,x} v_y - a_{t,x} \omega a}{v_x^2 + v_y^2 + 2v_y \omega a + \omega^2 a^2} - \cos\delta 2K_{y,f} \delta \dot{\delta} \\ & - \sin\delta 2K_{y,f} \dot{\delta} + \dot{t}_r c - c_{r1} 2v_x a_{t,x}) \end{aligned}$$

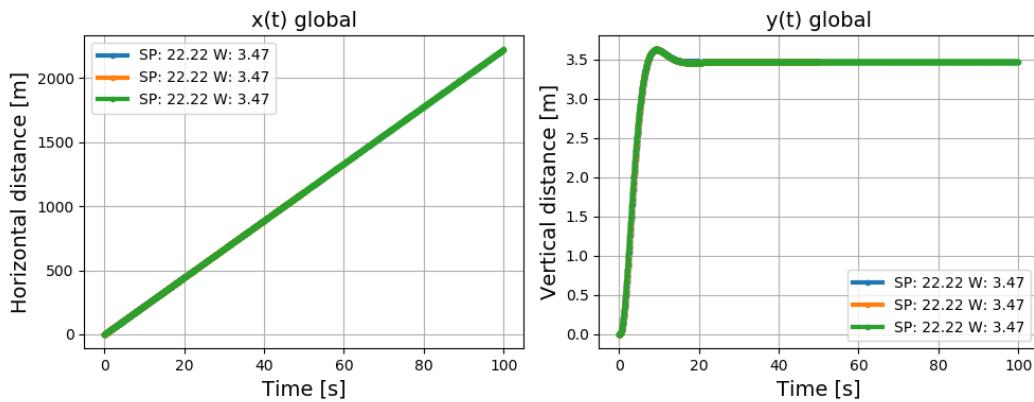
$$\begin{aligned} j_{y,t} = & \frac{1}{M} \cdot (\cos\delta t_r c \dot{\delta} + \sin\delta \dot{t}_r c + \sin\delta 2K_{y,f} \arctan\left(\frac{v_y + \omega a}{v_x}\right) \dot{\delta} \\ & - \cos\delta 2K_{y,f} \frac{v_x a_{t,y} + v_x \dot{\omega} a - a_{t,x} v_y - a_{t,x} \omega a}{v_x^2 + v_y^2 + 2v_y \omega a + \omega^2 a^2} - \sin\delta 2K_{y,f} \delta \dot{\delta} \\ & + \cos\delta 2K_{y,f} \dot{\delta} - 2K_{y,r} \frac{v_x a_{t,y} - v_x \dot{\omega} b - a_{t,x} v_y + a_{t,x} \omega b}{v_x^2 + v_y^2 - 2v_y \omega b + \omega^2 a^2}) \end{aligned}$$

Appendix B

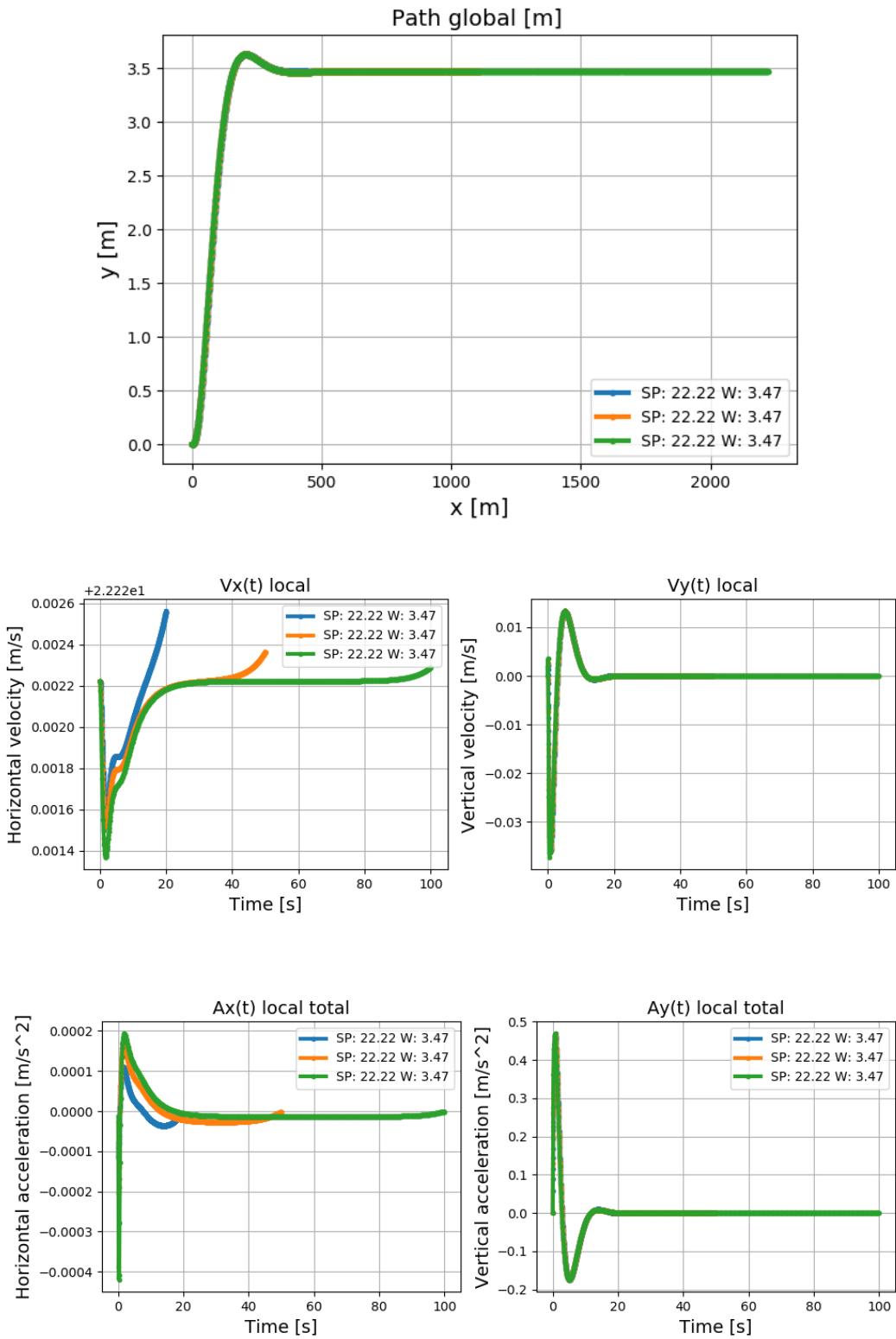
Results of the ideal data validation

In this appendix all the kinematic signals of the bicycle model that were checked in order to validate the ideal data, are presented. The lane change used for validation has as initial speed $22.22 \frac{m}{s}$ and uses a desired lateral distance of $3.47 m$. First the different results of varying the time limit are shown and next come the results with regard to varying the amount of control points N .

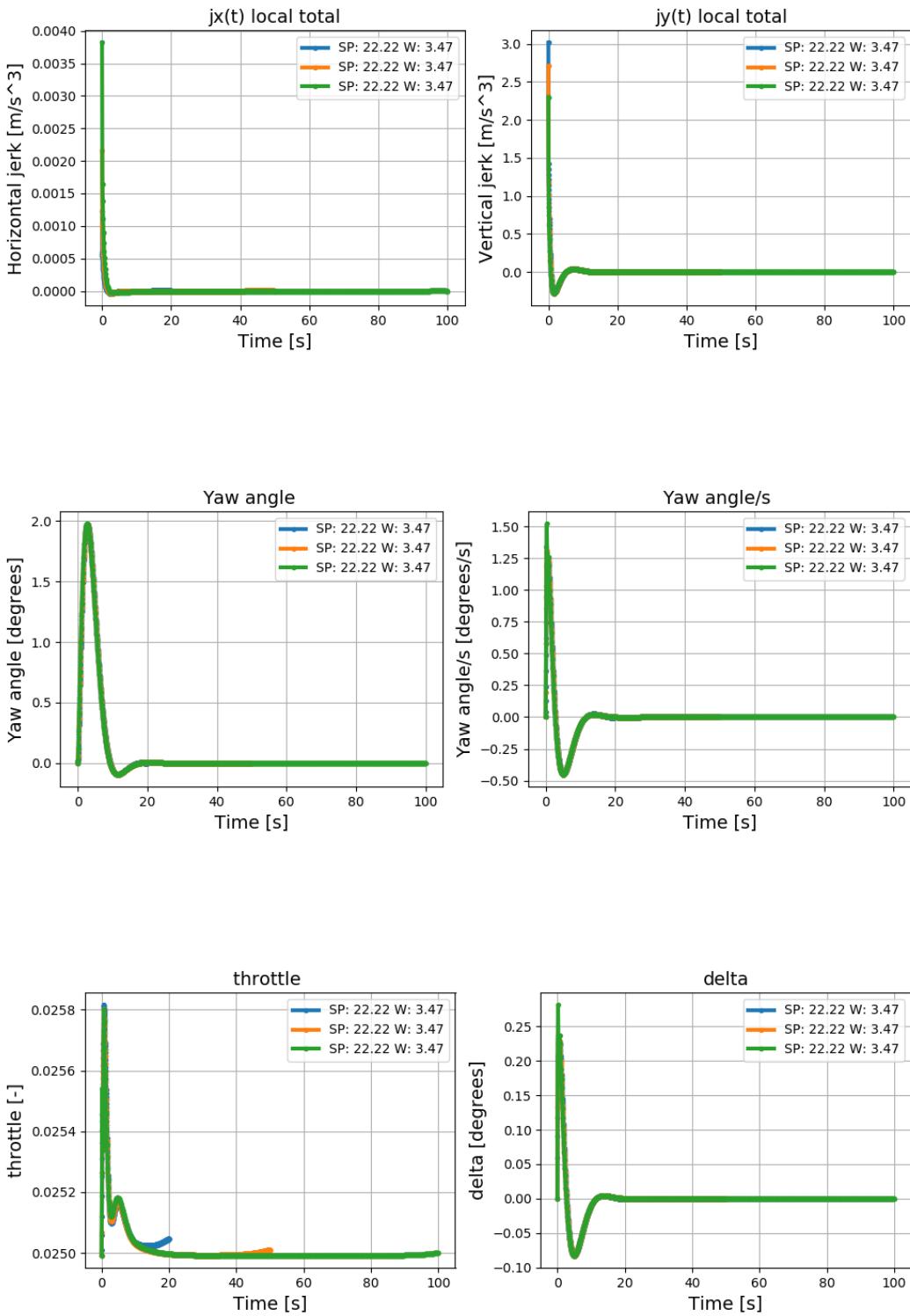
B.1 Time limit



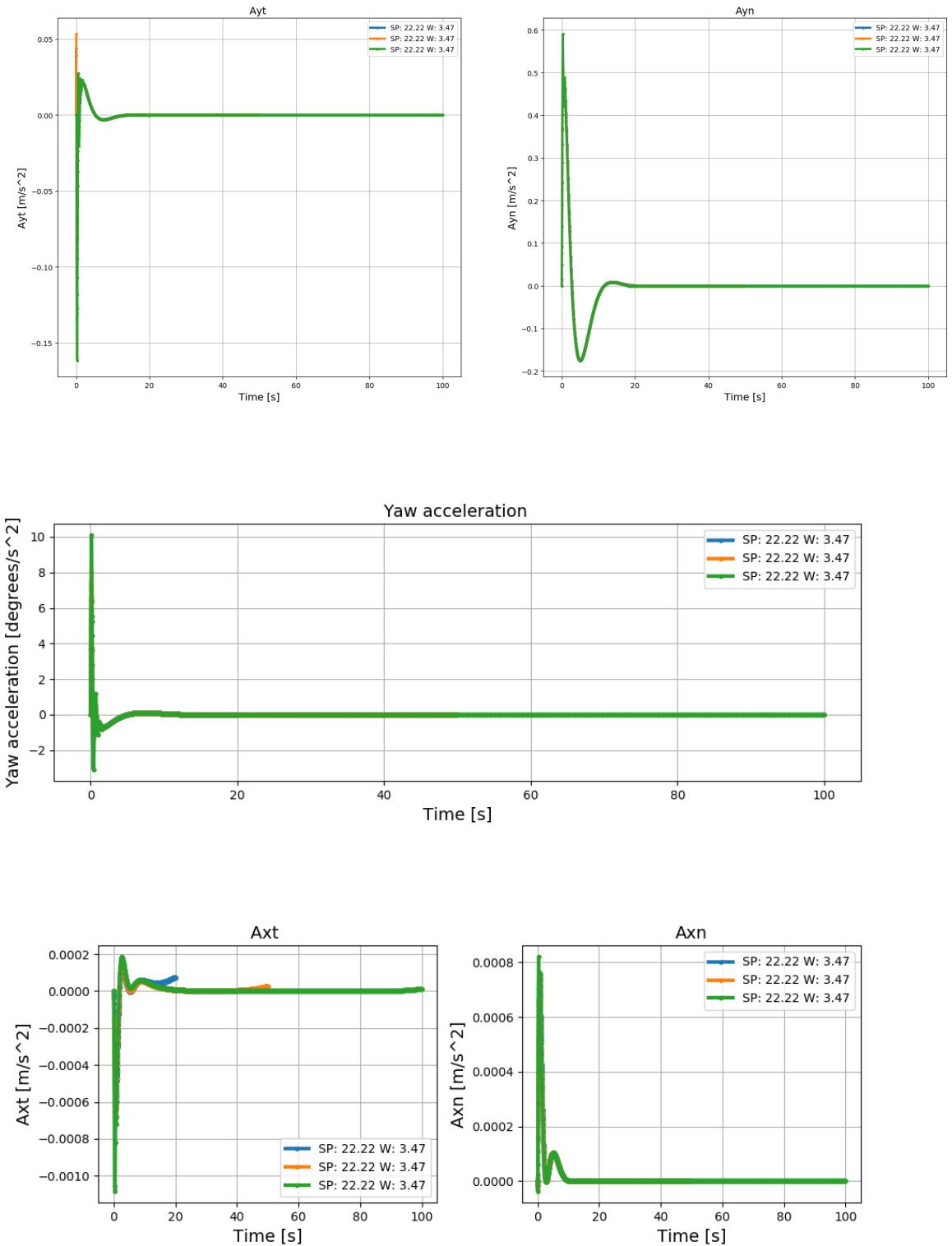
B. RESULTS OF THE IDEAL DATA VALIDATION



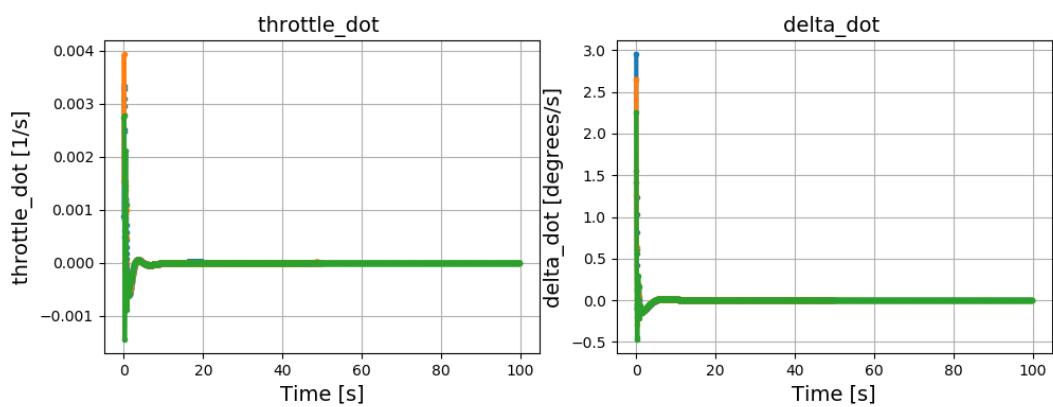
B.1. Time limit



B. RESULTS OF THE IDEAL DATA VALIDATION

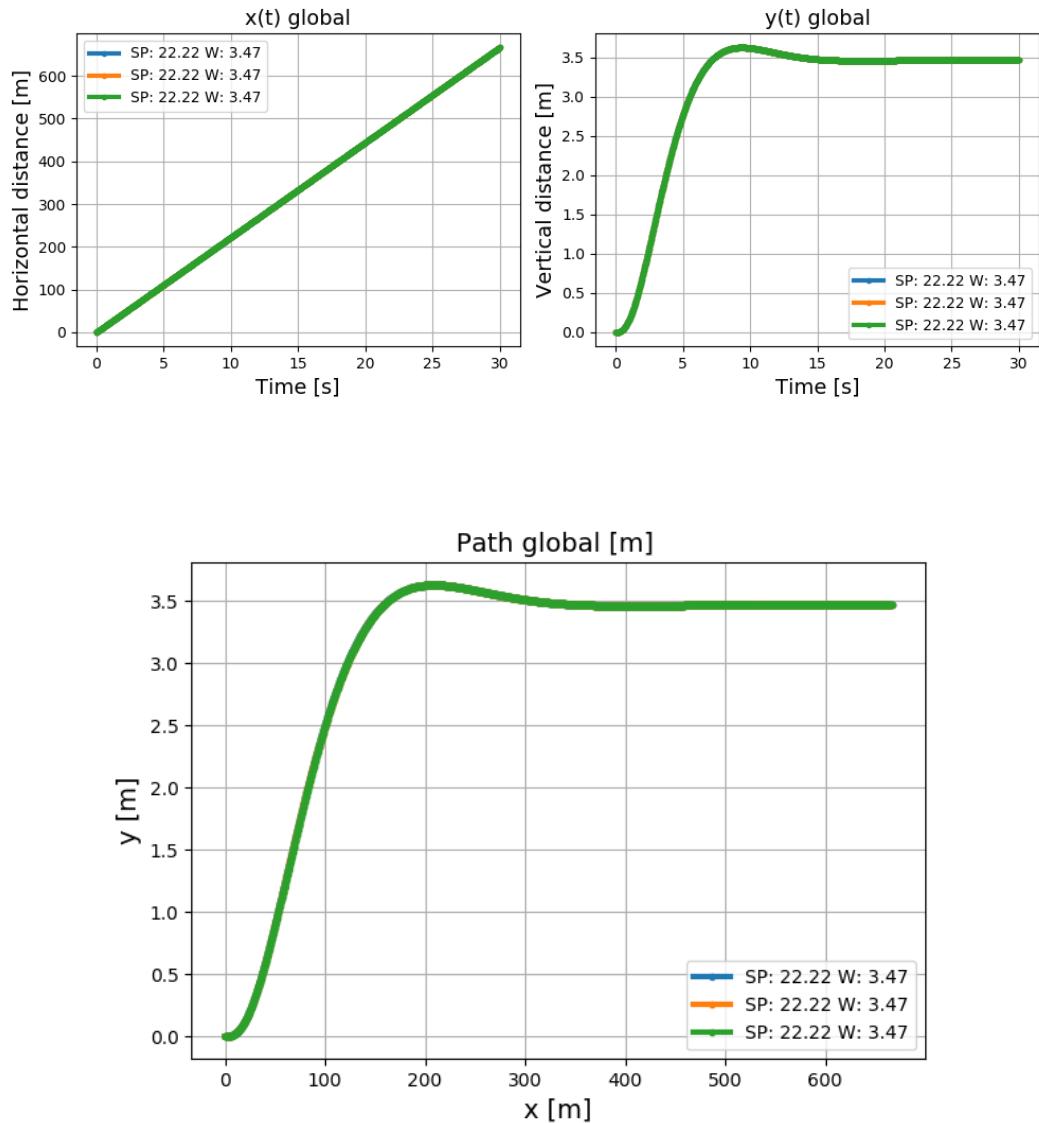


B.1. Time limit

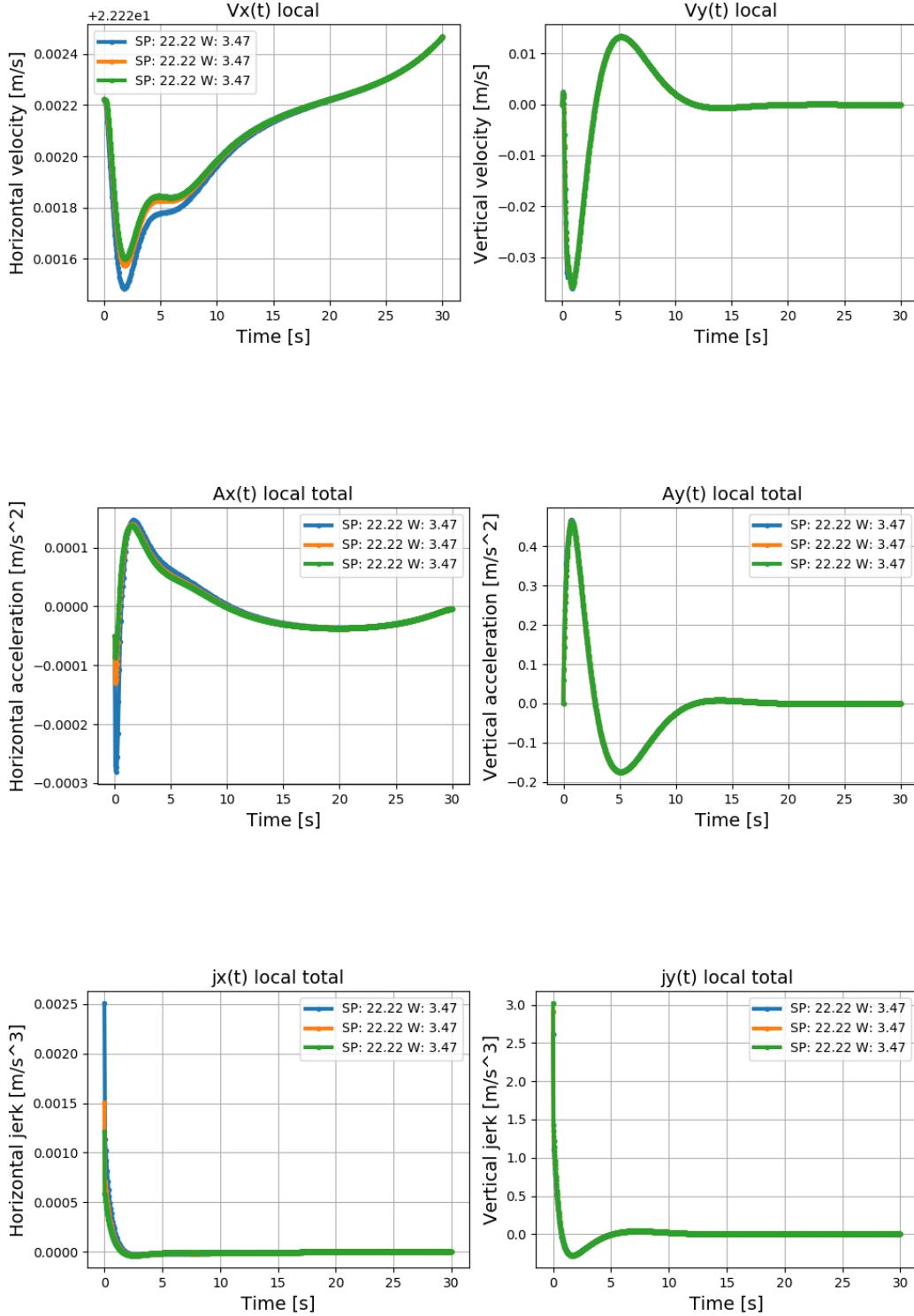


B. RESULTS OF THE IDEAL DATA VALIDATION

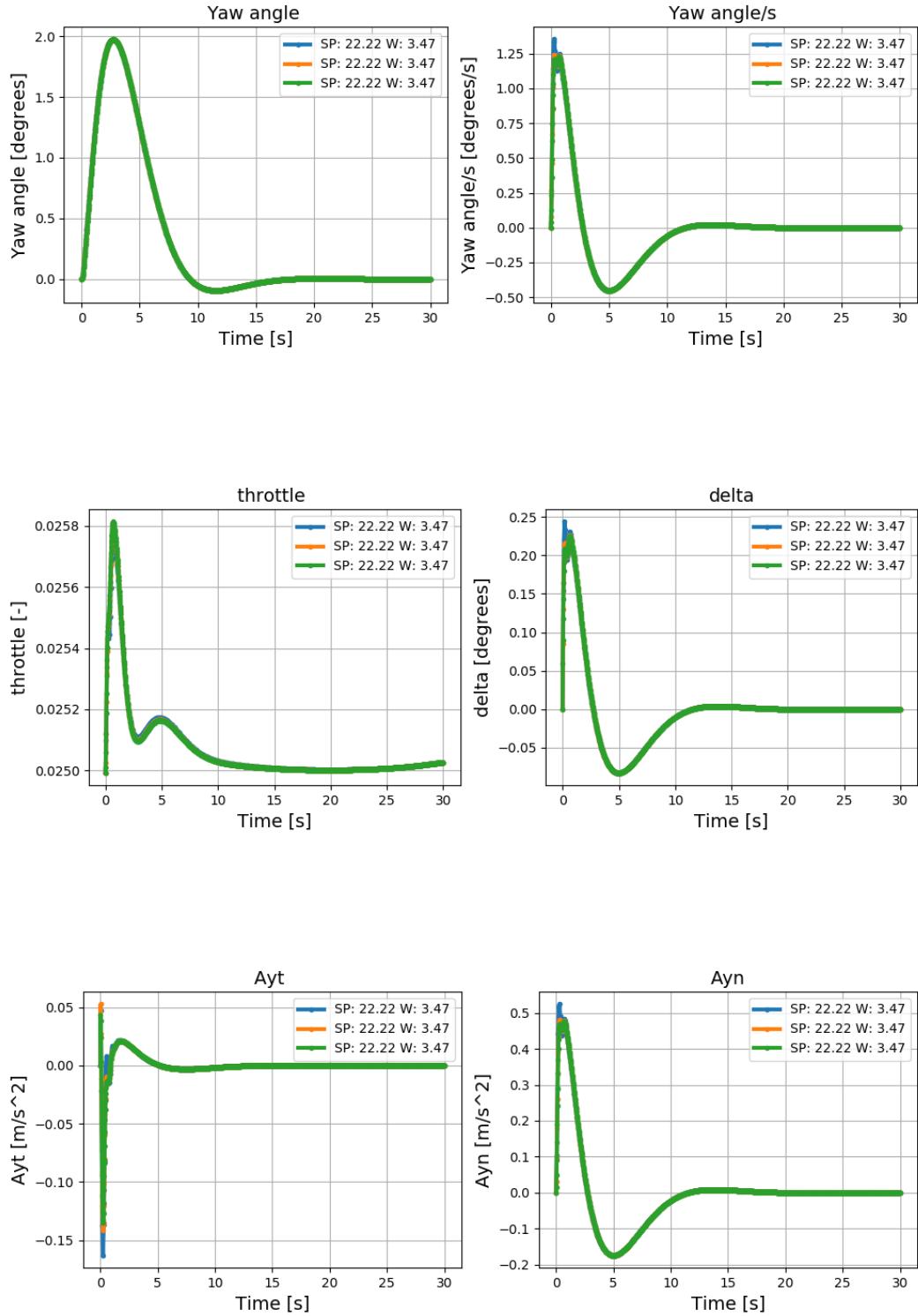
B.2 Amount of optimization points



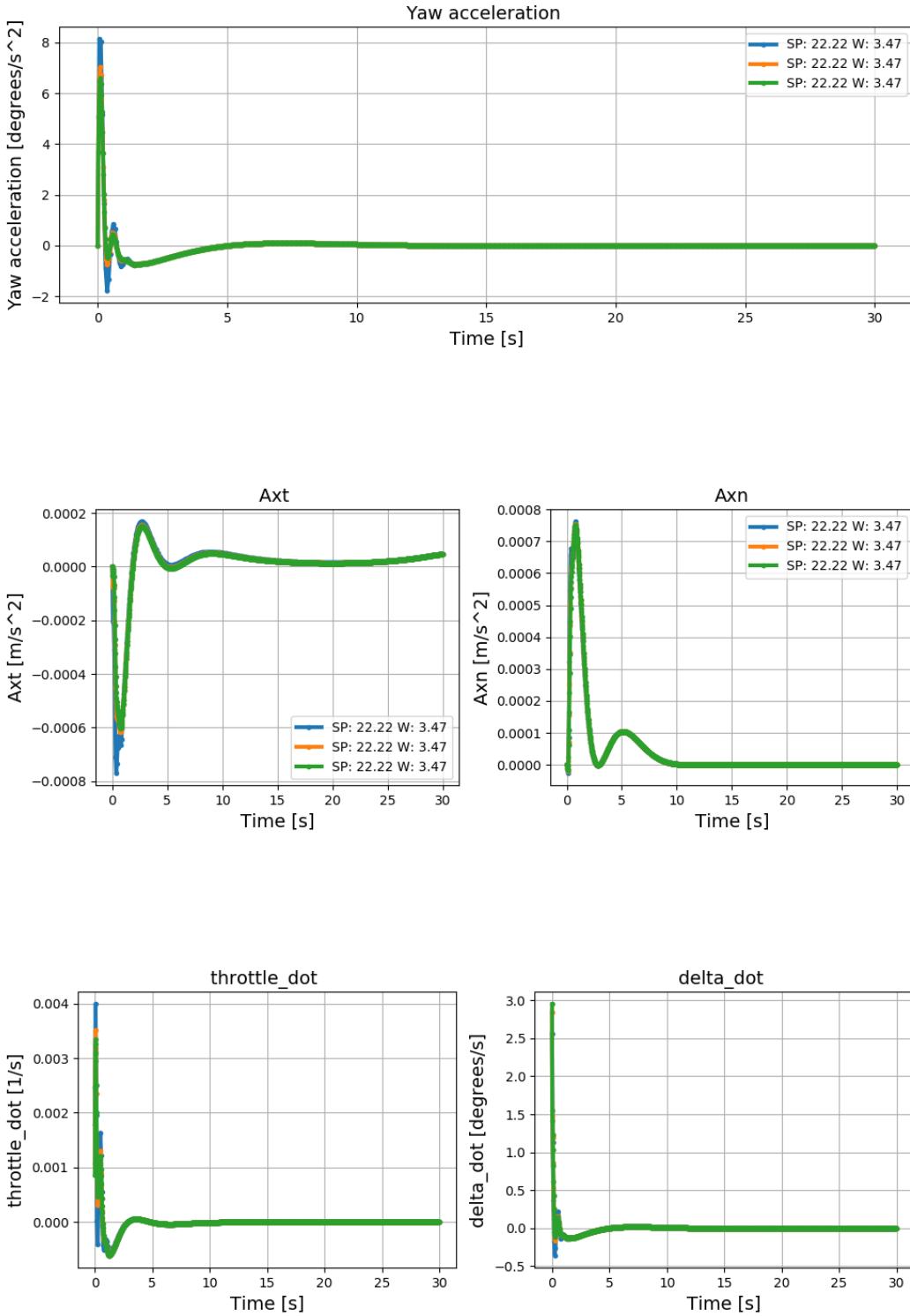
B.2. Amount of optimization points



B. RESULTS OF THE IDEAL DATA VALIDATION



B.2. Amount of optimization points

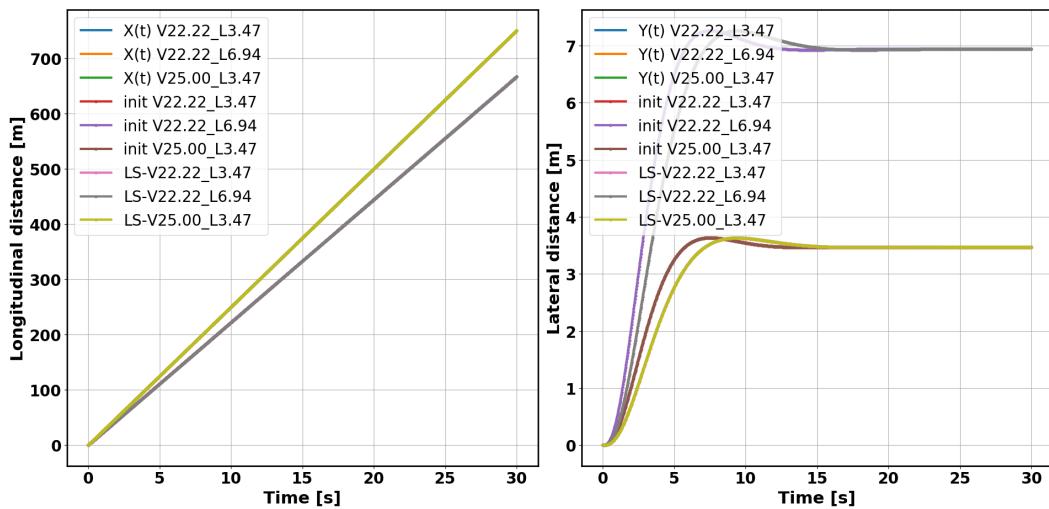


Appendix C

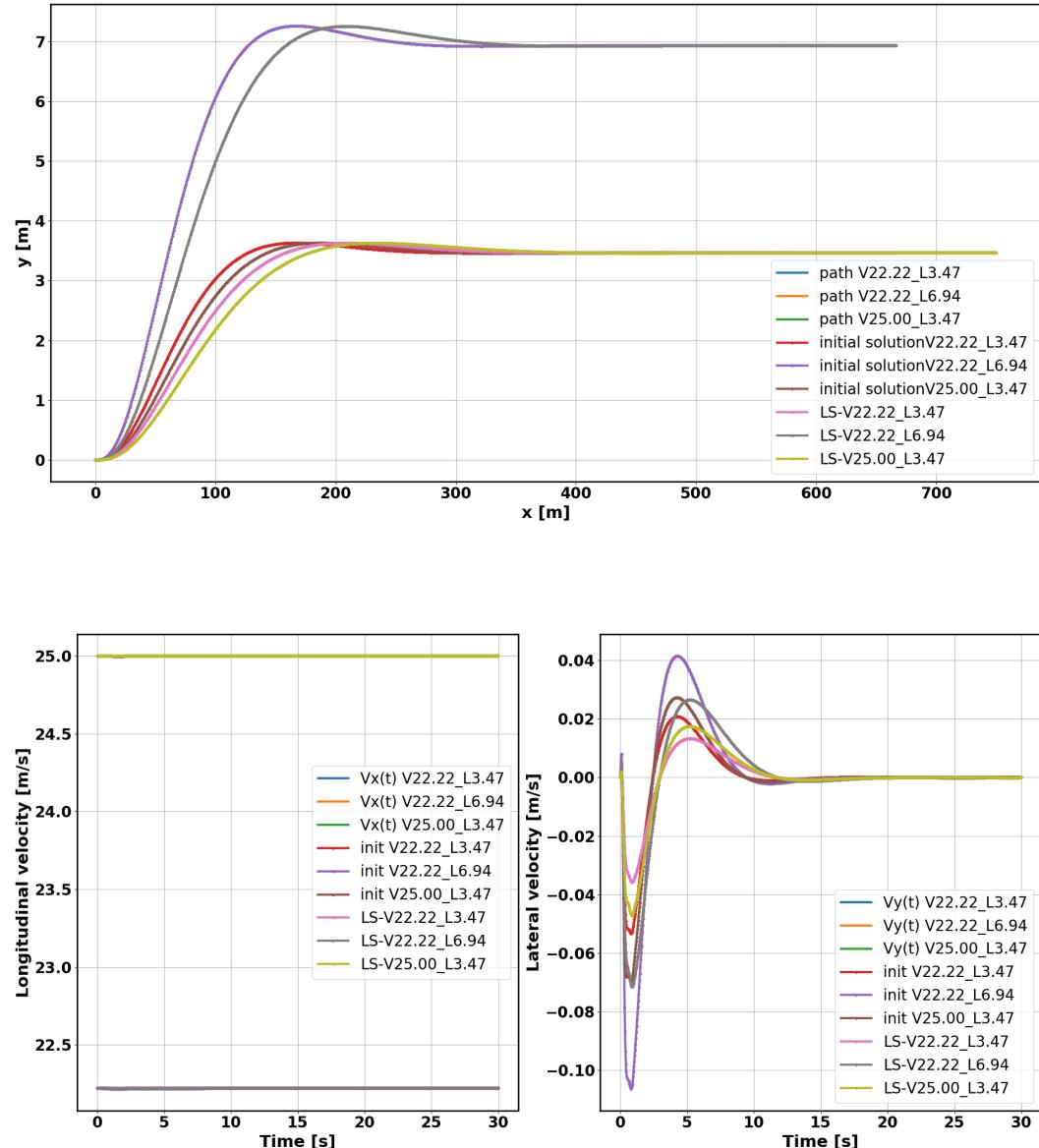
Results of the averaging learning on 3 ideal datasets

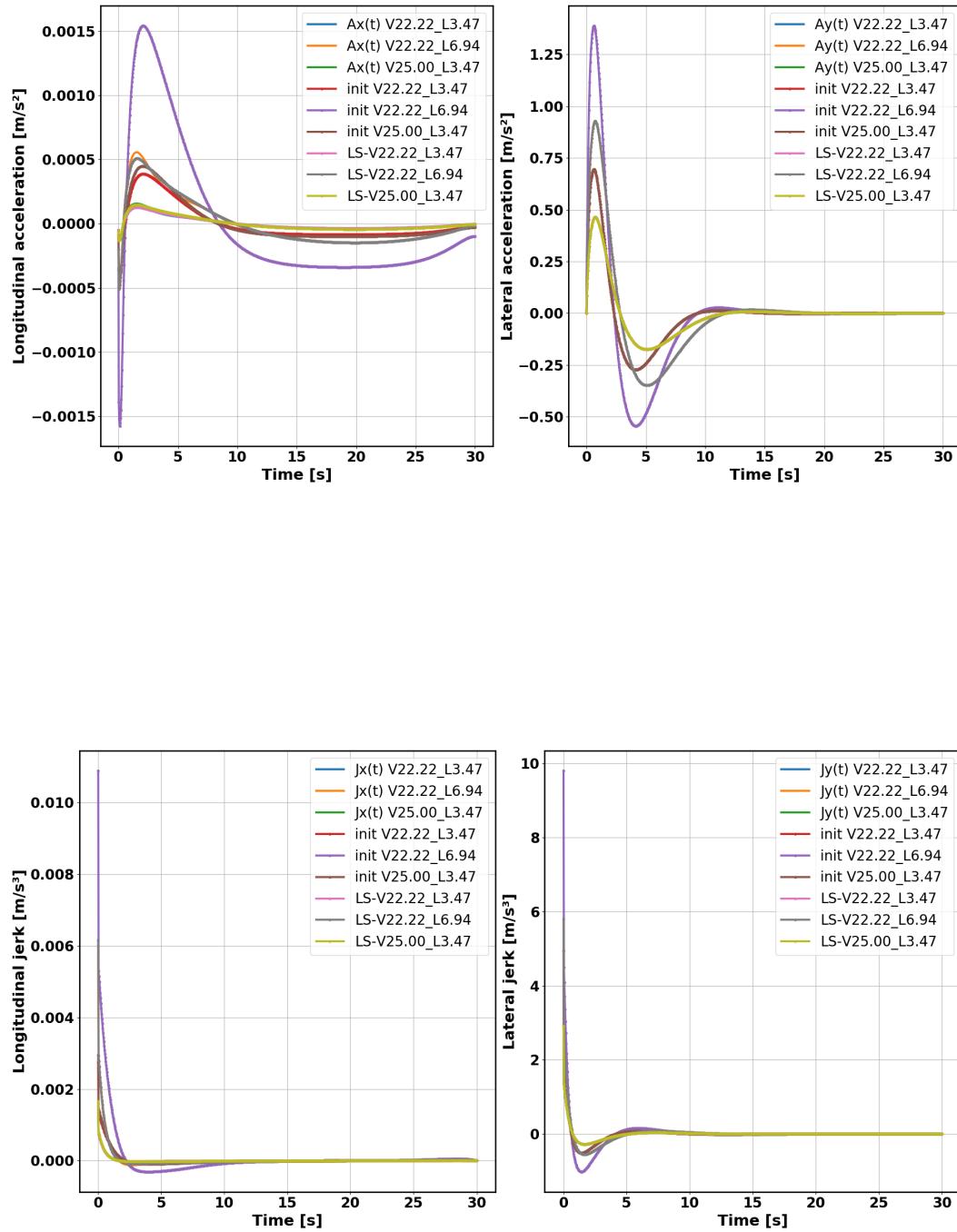
The figures shown in this appendix serve as a completion on the figures that were presented in the discussion of section 4.4.2. It concerns the results of the learning algorithm that learns from three ideal datasets with the amount of control points N equal to 1000 and time limit set to 30 s.

First the resulting kinematic signals of the bicycle model during the different lane changes are shown. It should be noted that Figure C.1 and C.2 show the angle of the front wheel of the bicycle model. In order to obtain the steerwheelangle, this relation is linearised by the factor $G_s = 16.96$ which means that $\delta_{SWA} = G_s \cdot \delta_{front}$. Further Figure C.3 displays the convergence during the learning process and plots f_{rel} over the iterations. Figure C.4 shows the absolute difference between the learned and observed features. In Figure C.5 the learning of the weights towards the final ones are presented. Figure C.6 gives the difference of current weight with respect to the previous one and as last Figure C.7 shows which of the three RPROP cases is used in order to update a certain weight.



C. RESULTS OF THE AVERAGING LEARNING ON 3 IDEAL DATASETS





C. RESULTS OF THE AVERAGING LEARNING ON 3 IDEAL DATASETS

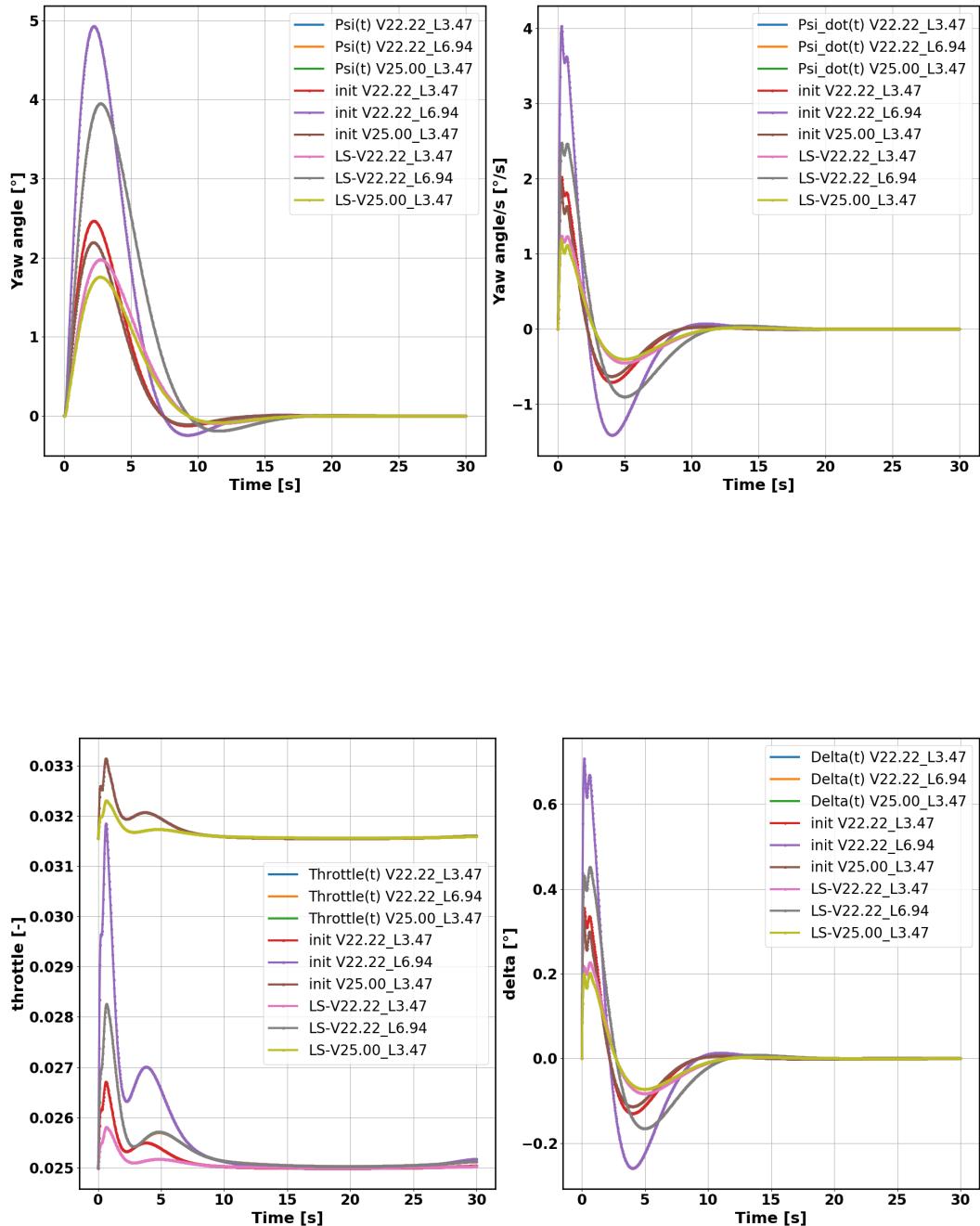
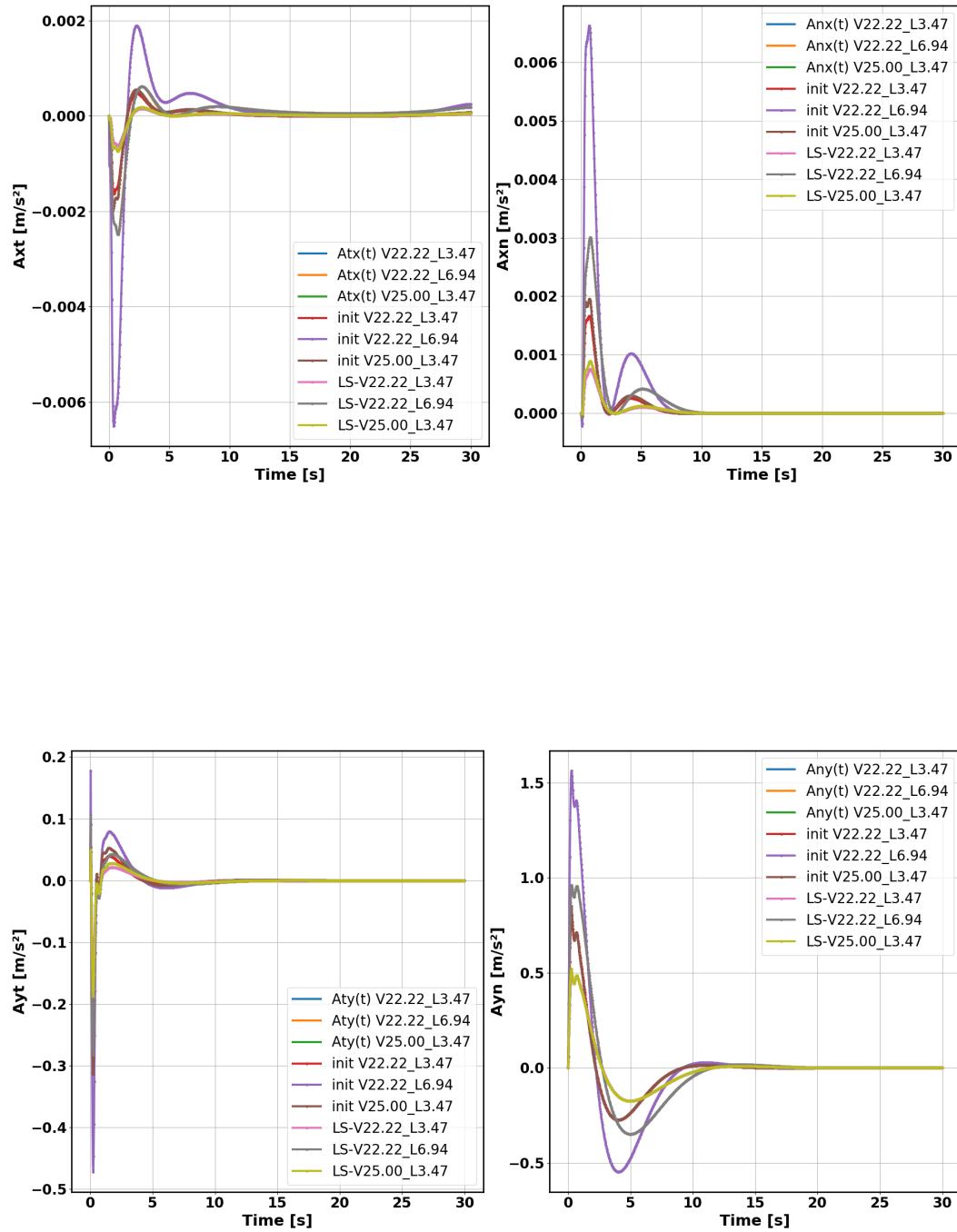


Figure C.1: This figure shows the amount of throttle and the angle of the front wheel of the bicycle model during the lane change maneuvers.



C. RESULTS OF THE AVERAGING LEARNING ON 3 IDEAL DATASETS

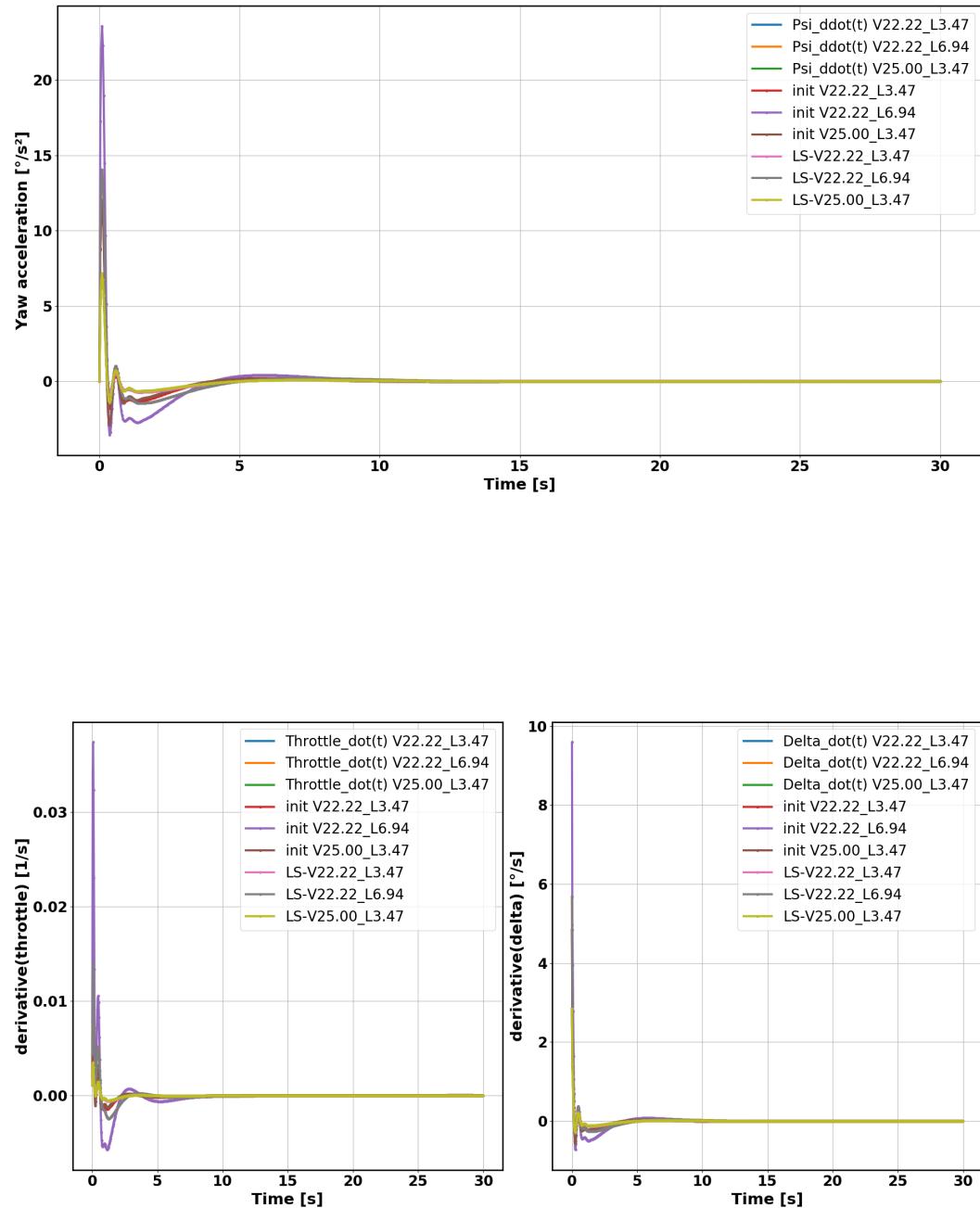


Figure C.2: This figure shows the amount of first derivative of throttle and the first derivative of the angle of the front wheel of the bicycle model is given as input during the lane change maneuvers.

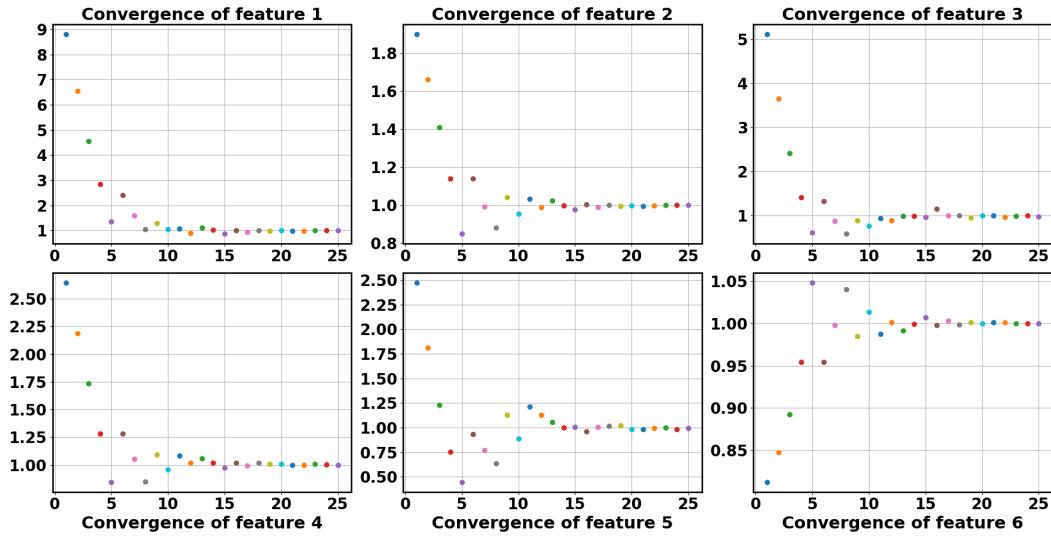


Figure C.3: The convergence of f_{rel} towards one during the learning iterations.

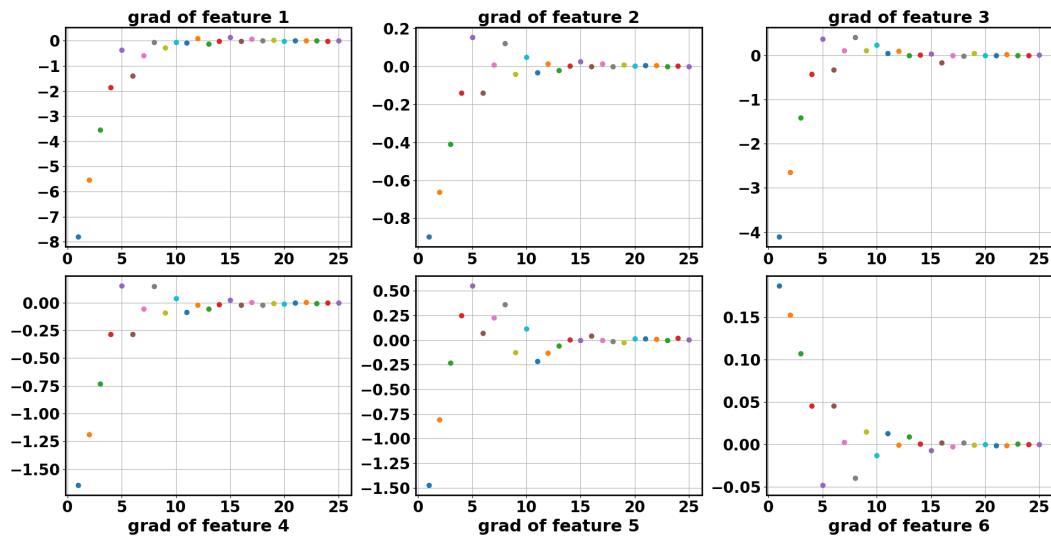


Figure C.4: The gradient $\frac{\partial F_{diff}}{\partial \theta}$ estimated by $f_{obs} - f_{learned}$ towards zero during the different learning iterations.

C. RESULTS OF THE AVERAGING LEARNING ON 3 IDEAL DATASETS

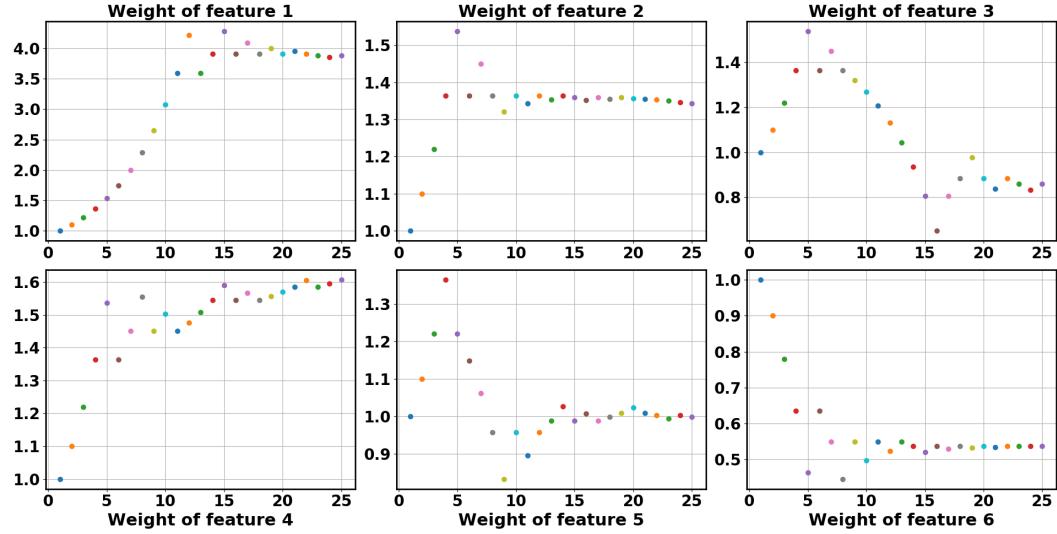


Figure C.5: The learned weights during the different learning iterations.

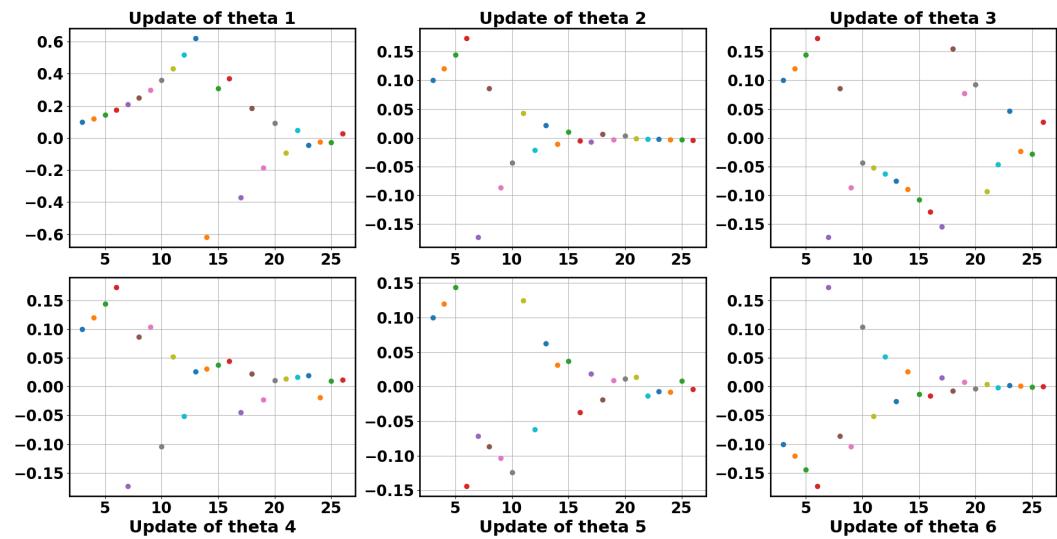


Figure C.6: The difference of θ with respect to one used in the previous iteration.

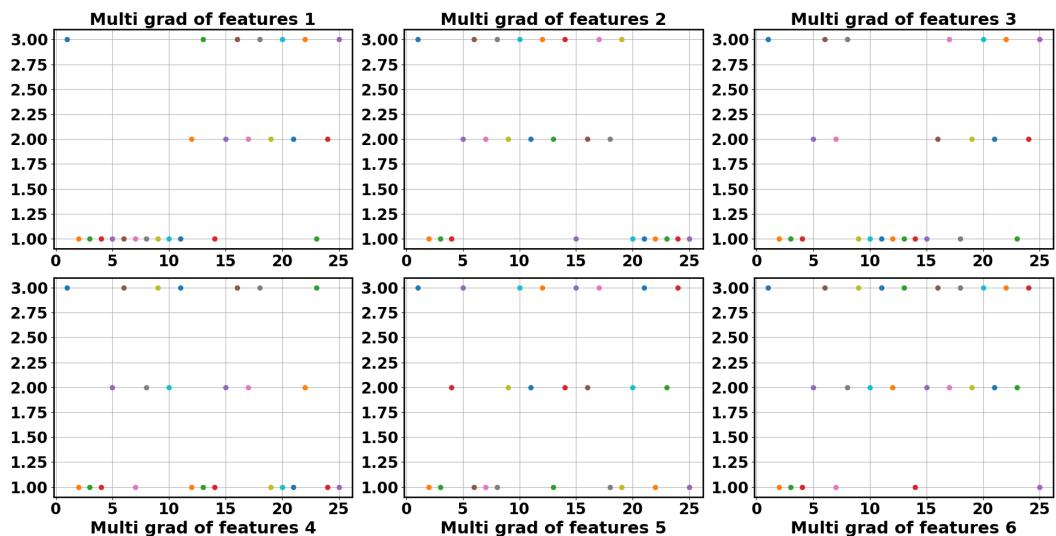
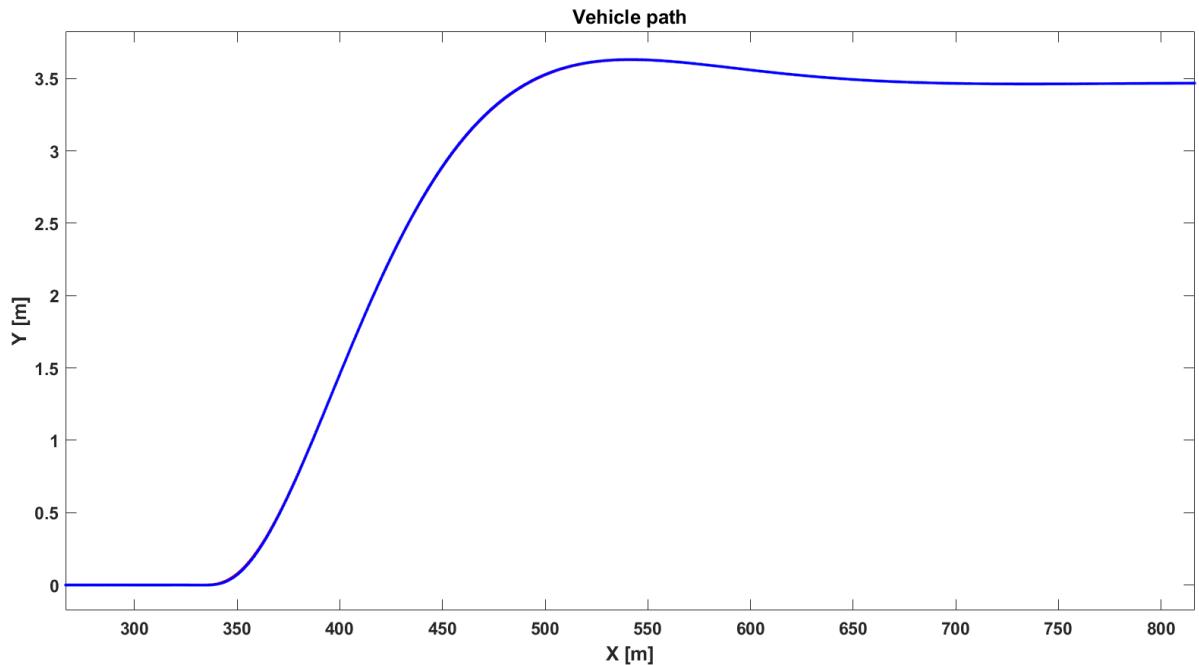


Figure C.7: This figure shows which case of the three available in the RPROP algorithm is chosen during the learning iterations.

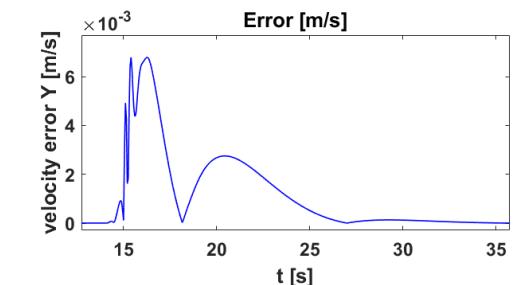
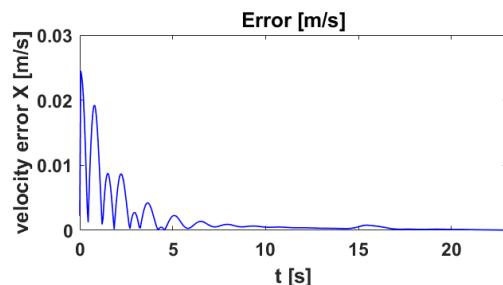
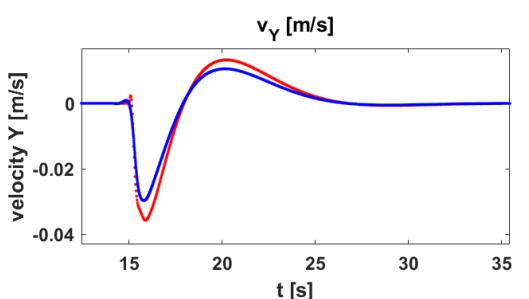
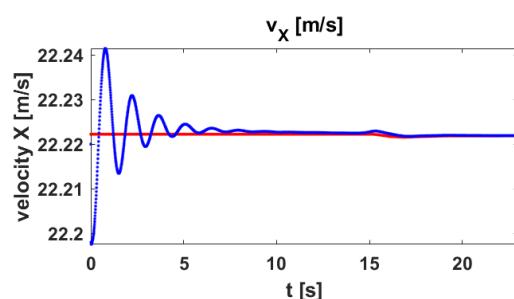
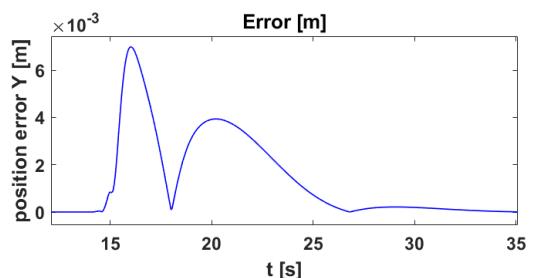
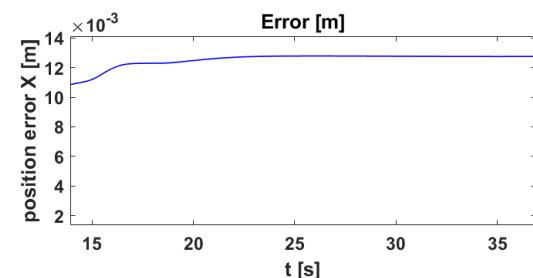
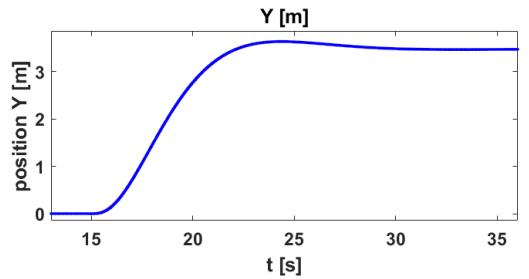
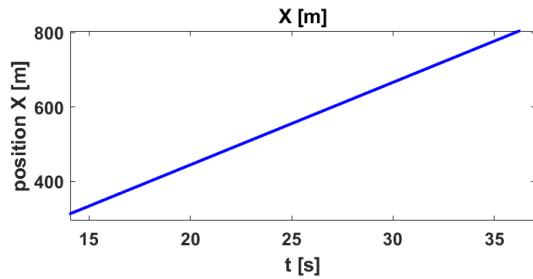
Appendix D

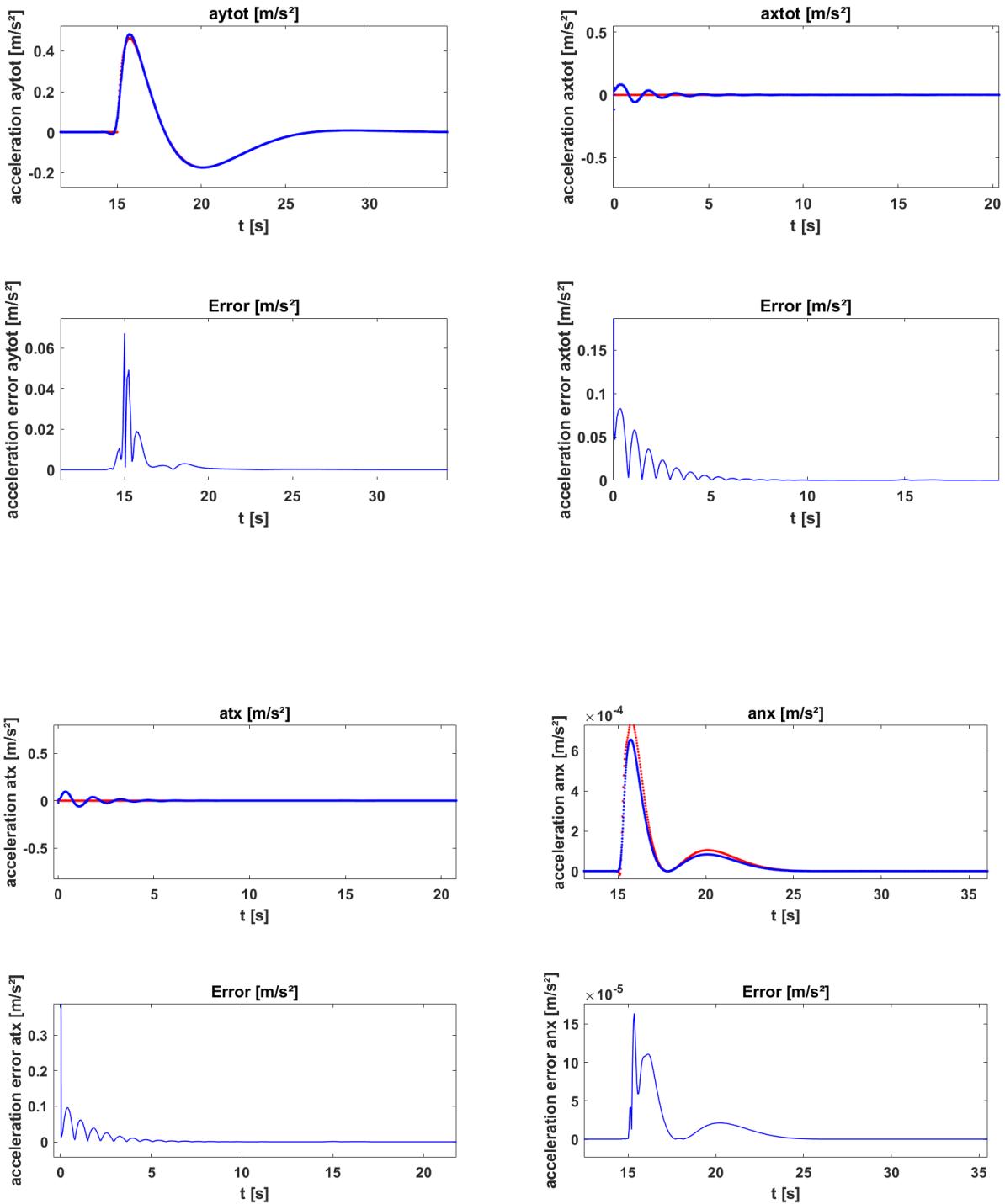
Accuracy results of the tracking MPC

This appendix shows the results that are discussed in 5.2.2. In red the reference is shown that was derived from Eq. (4.8) with using the bicycle model and in blue the trajectory completed by the Amesim model. During the learning process with the Amesim model (section 5.3) the first 10 s of the Amesim trajectory is thrown away in order to remove the unstable longitudinal behaviour at the start of the simulation.

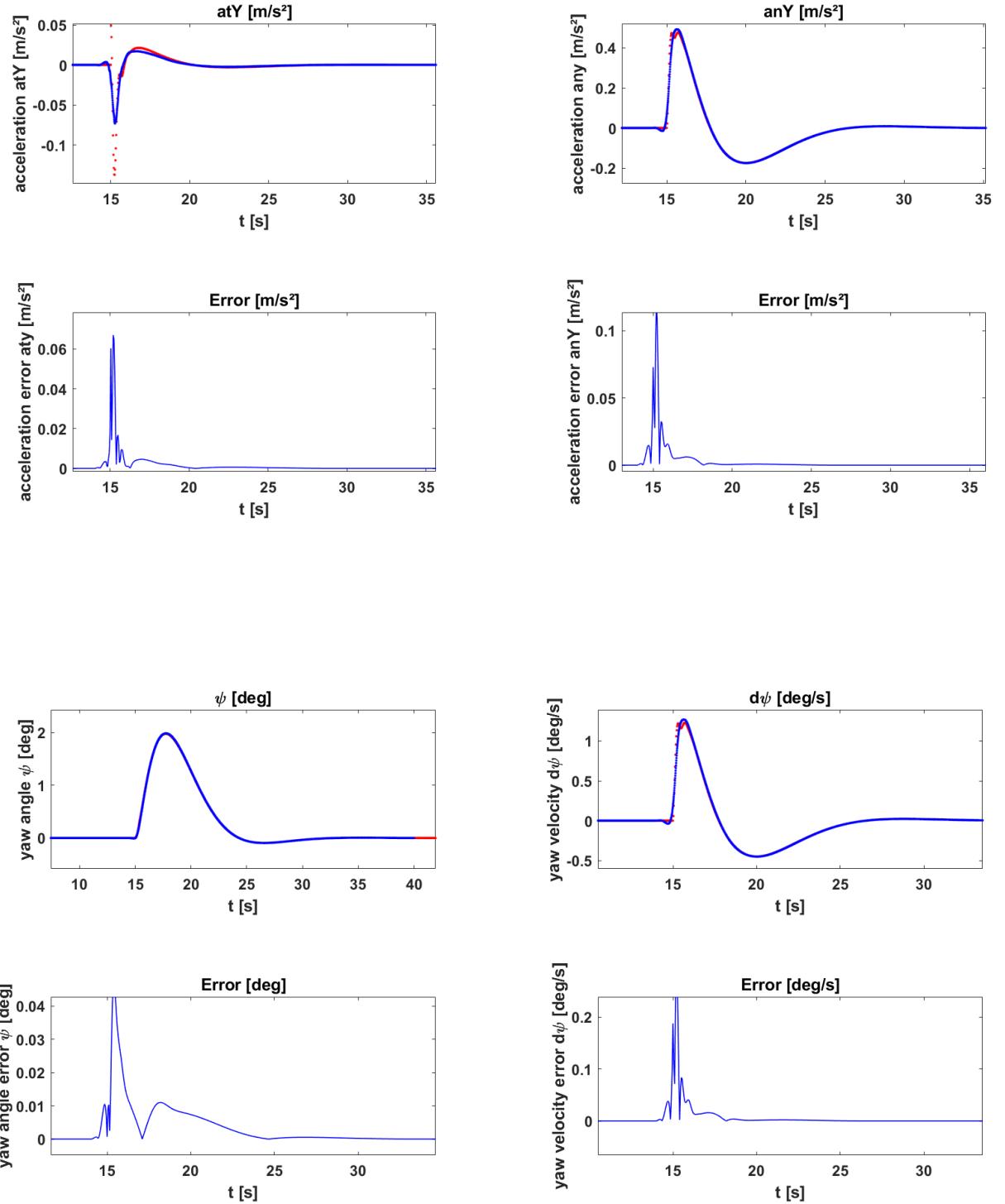


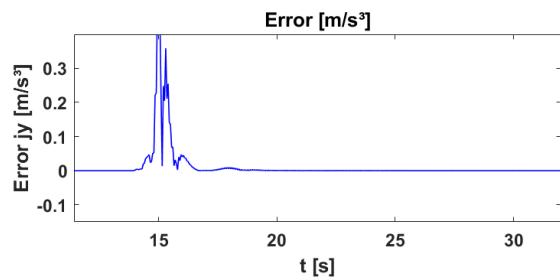
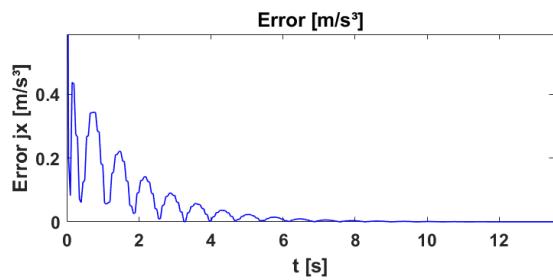
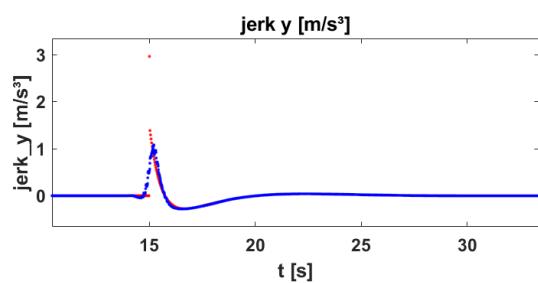
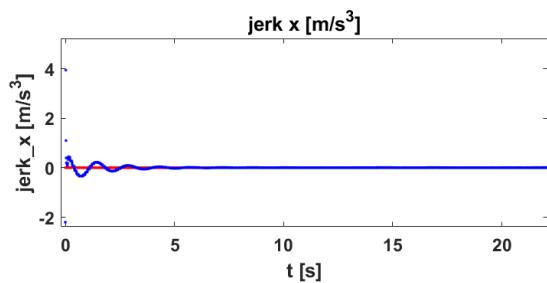
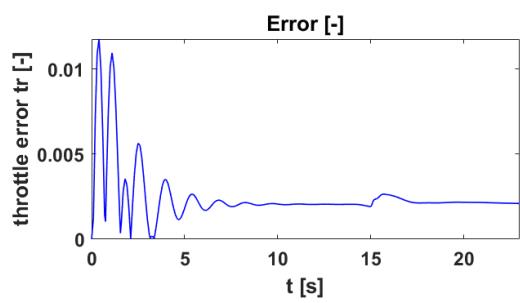
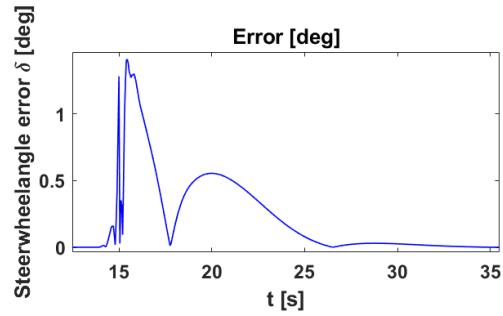
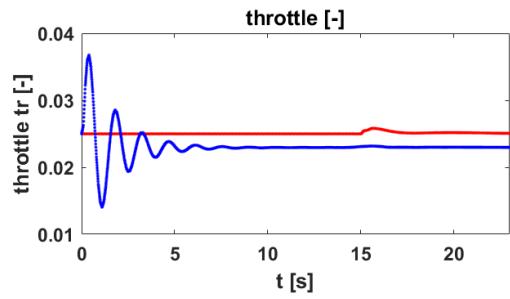
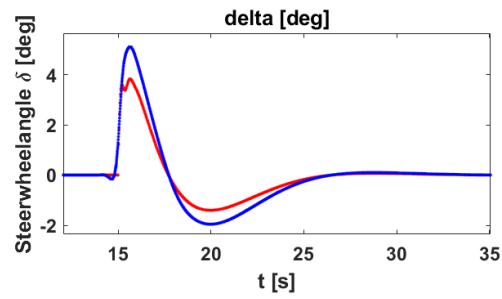
D. ACCURACY RESULTS OF THE TRACKING MPC



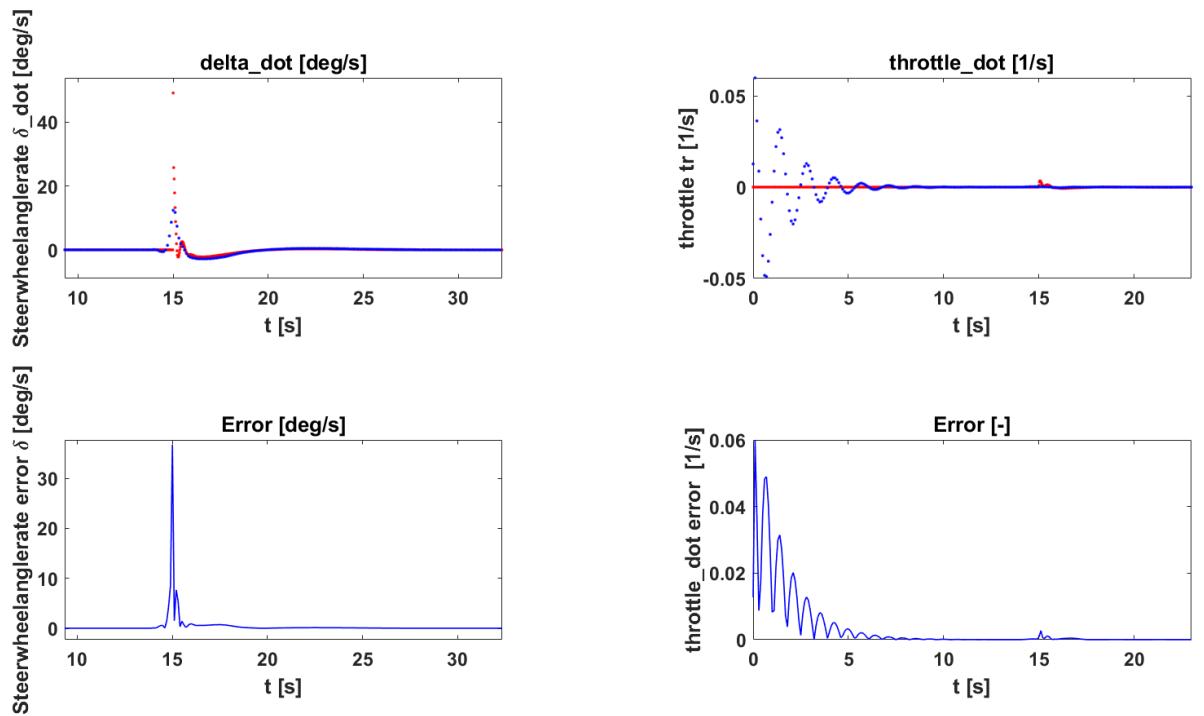


D. ACCURACY RESULTS OF THE TRACKING MPC





D. ACCURACY RESULTS OF THE TRACKING MPC

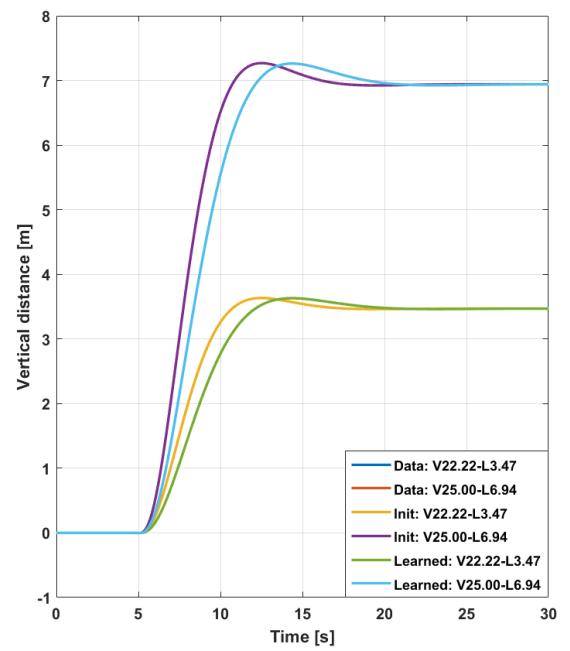
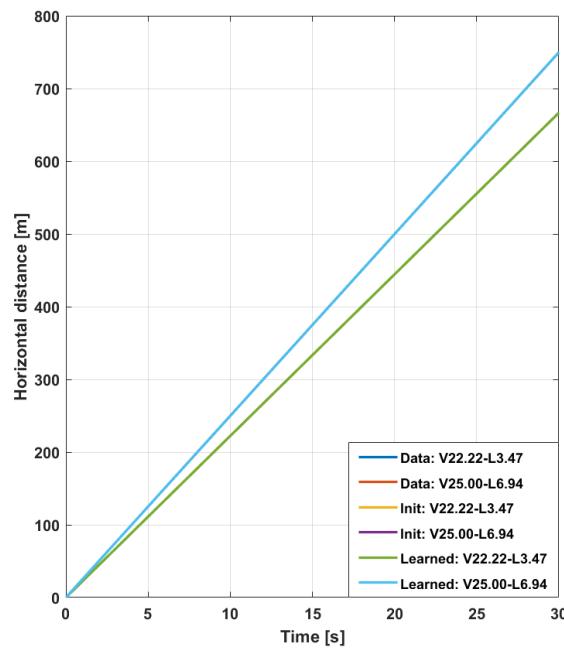
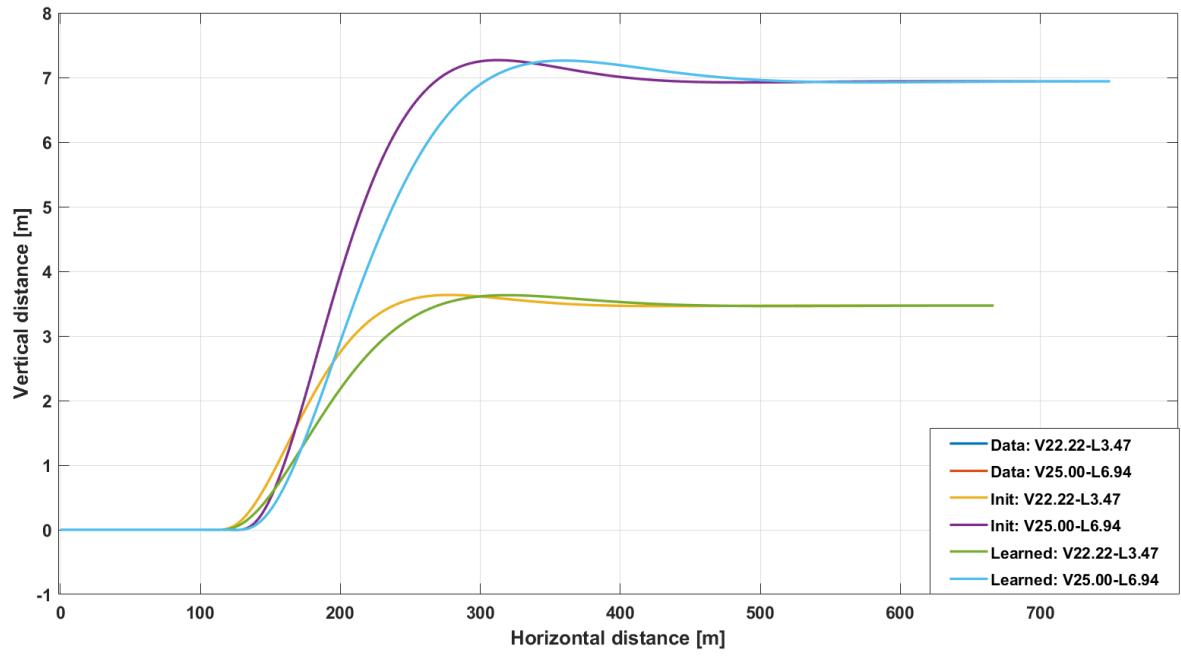


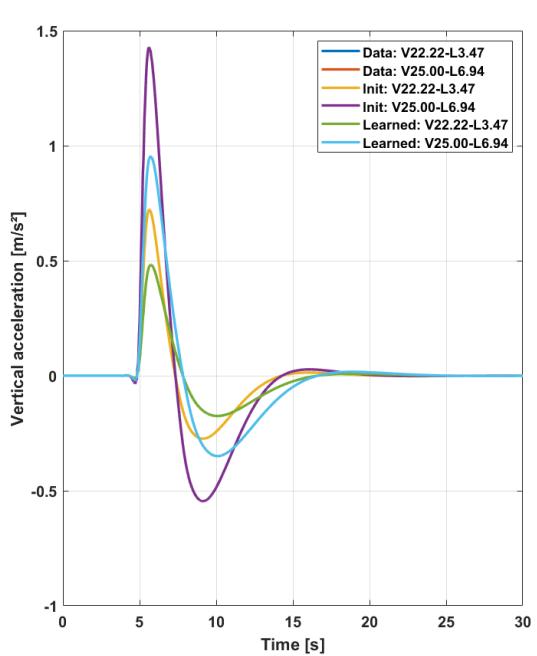
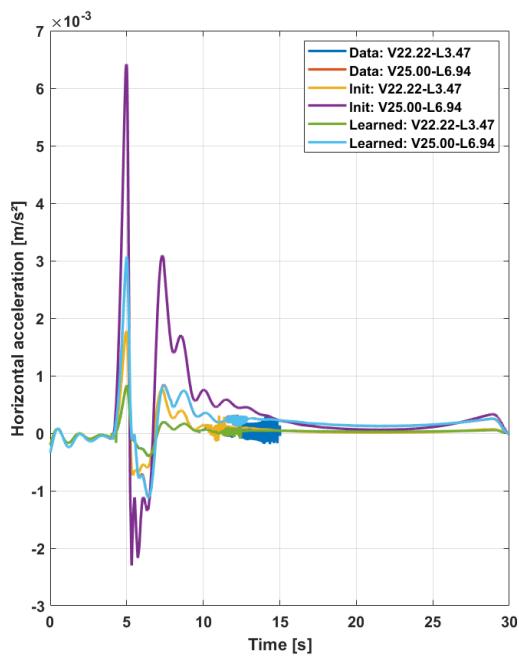
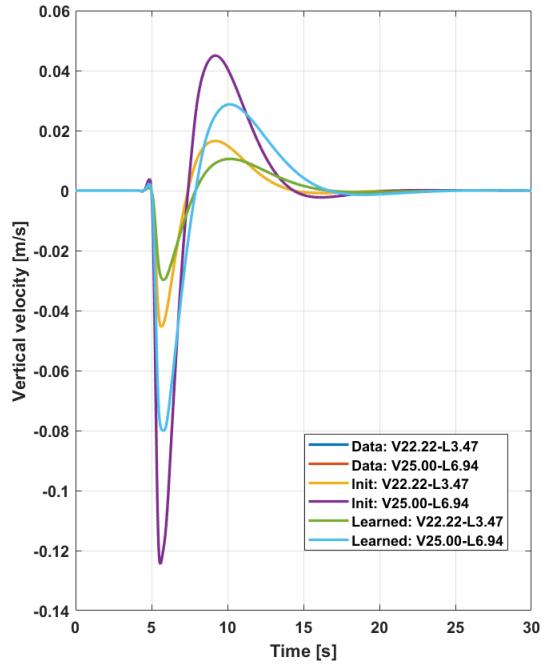
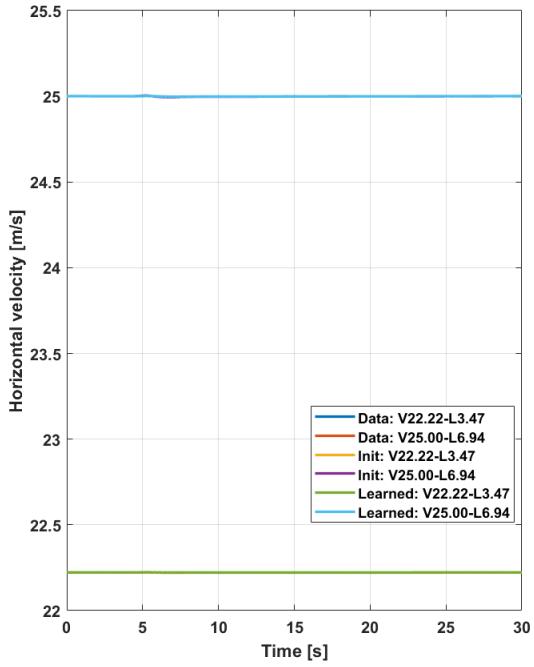
Appendix E

Results of learning with the 15 dof Amesim model

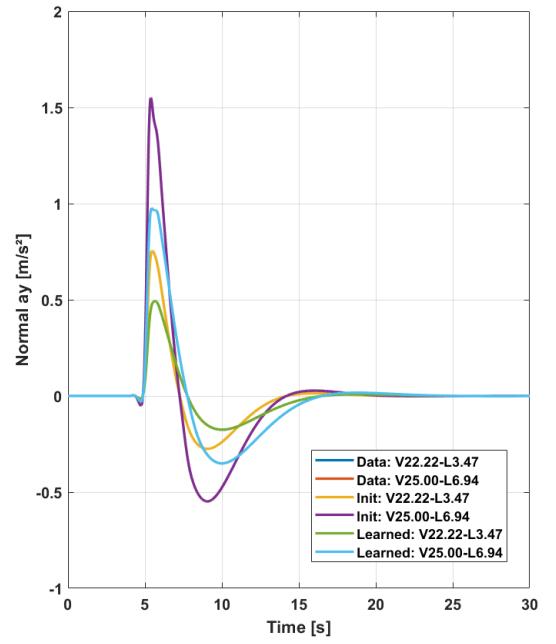
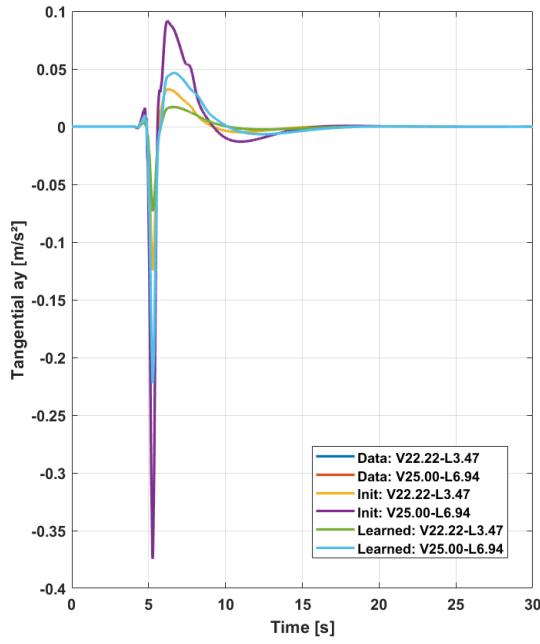
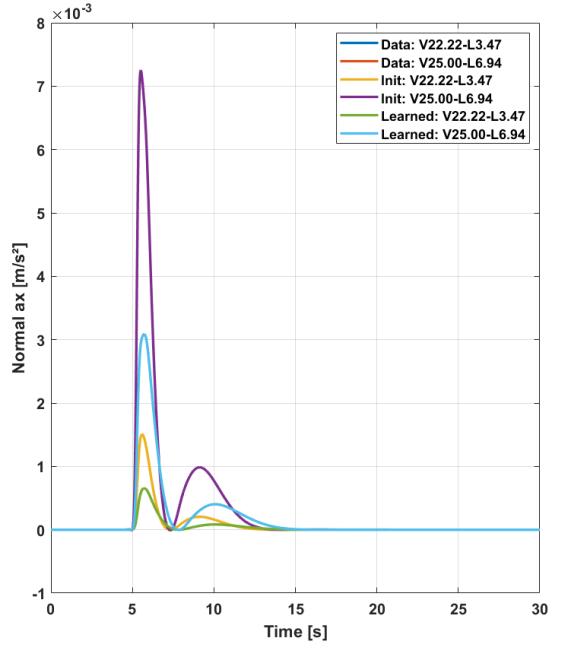
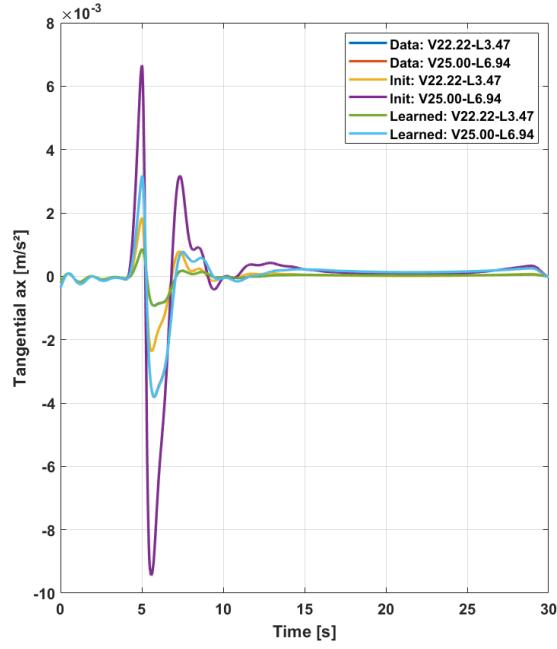
This appendix shows the learning results of the 15 dof Amesim model on two distinct datasets $V022.22 - L3.47$ and $V025.00 - L6.94$. Except for the longitudinal total acceleration and jerk, all the learned kinematic signals match closely with the observed one. The most important conclusions about the results can be found in section 5.3. It should be noted that Figure E.1 and E.2 show the angle of the front wheel of the bicycle model. In order to obtain the steerwheelangle, this relation is linearised by the factor $Gs = 16.96$ which means that $\delta_{SWA} = Gs \cdot \delta_{front}$. Further Figure E.3 displays the convergence during the learning process and plots f_{rel} over the iterations. Figure E.4 shows the absolute difference between the learned and observed features. In Figure E.5 the learning of the weighting factors towards the final ones are presented. Figure E.6 gives the difference of current weighting factors with respect to the previous one and as last Figure E.7 shows which of the three RPROP cases is used in order to update a certain weight factor.

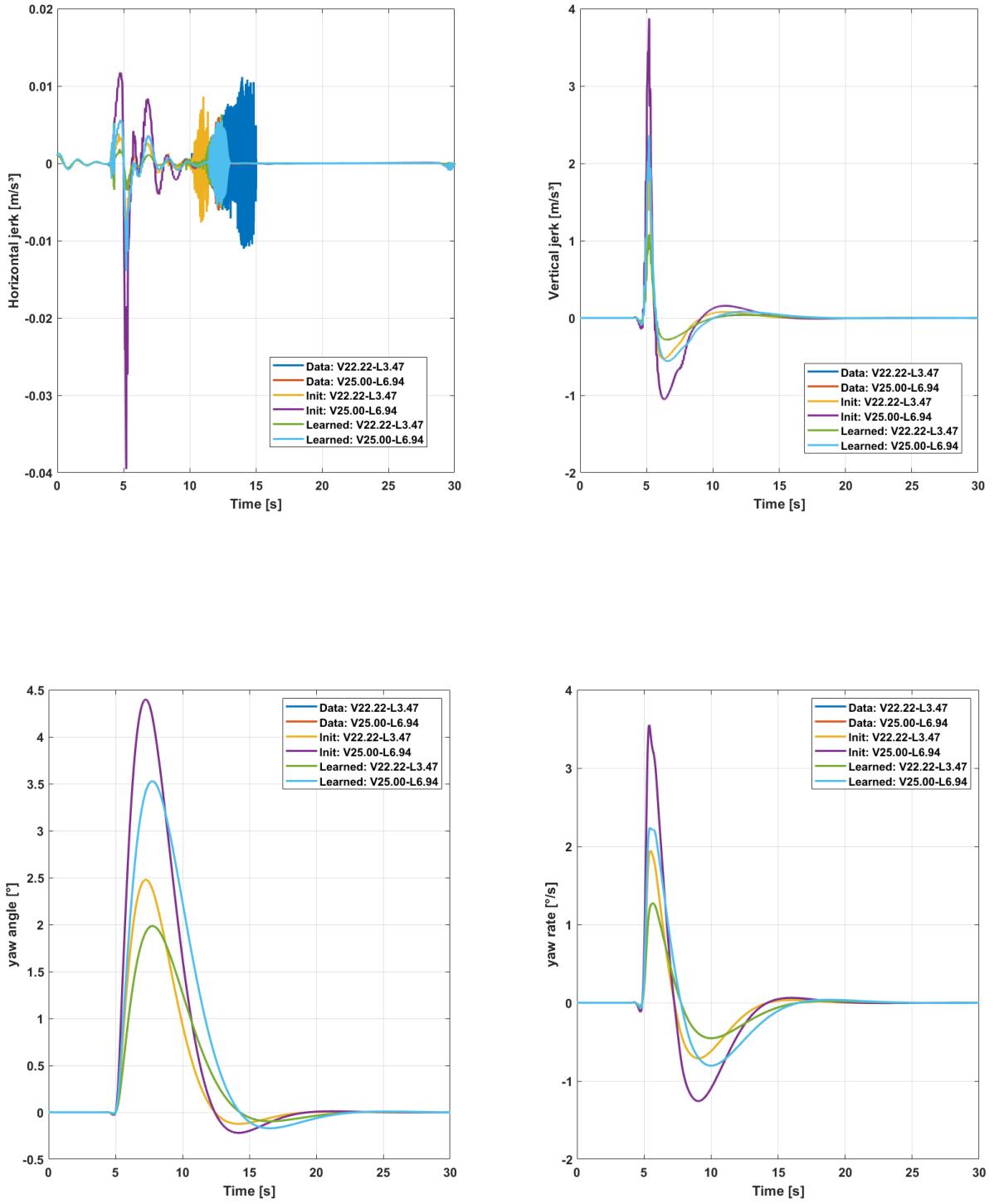
E. RESULTS OF LEARNING WITH THE 15 DOF AMESIM MODEL





E. RESULTS OF LEARNING WITH THE 15 DOF AMESIM MODEL





E. RESULTS OF LEARNING WITH THE 15 DOF AMESIM MODEL

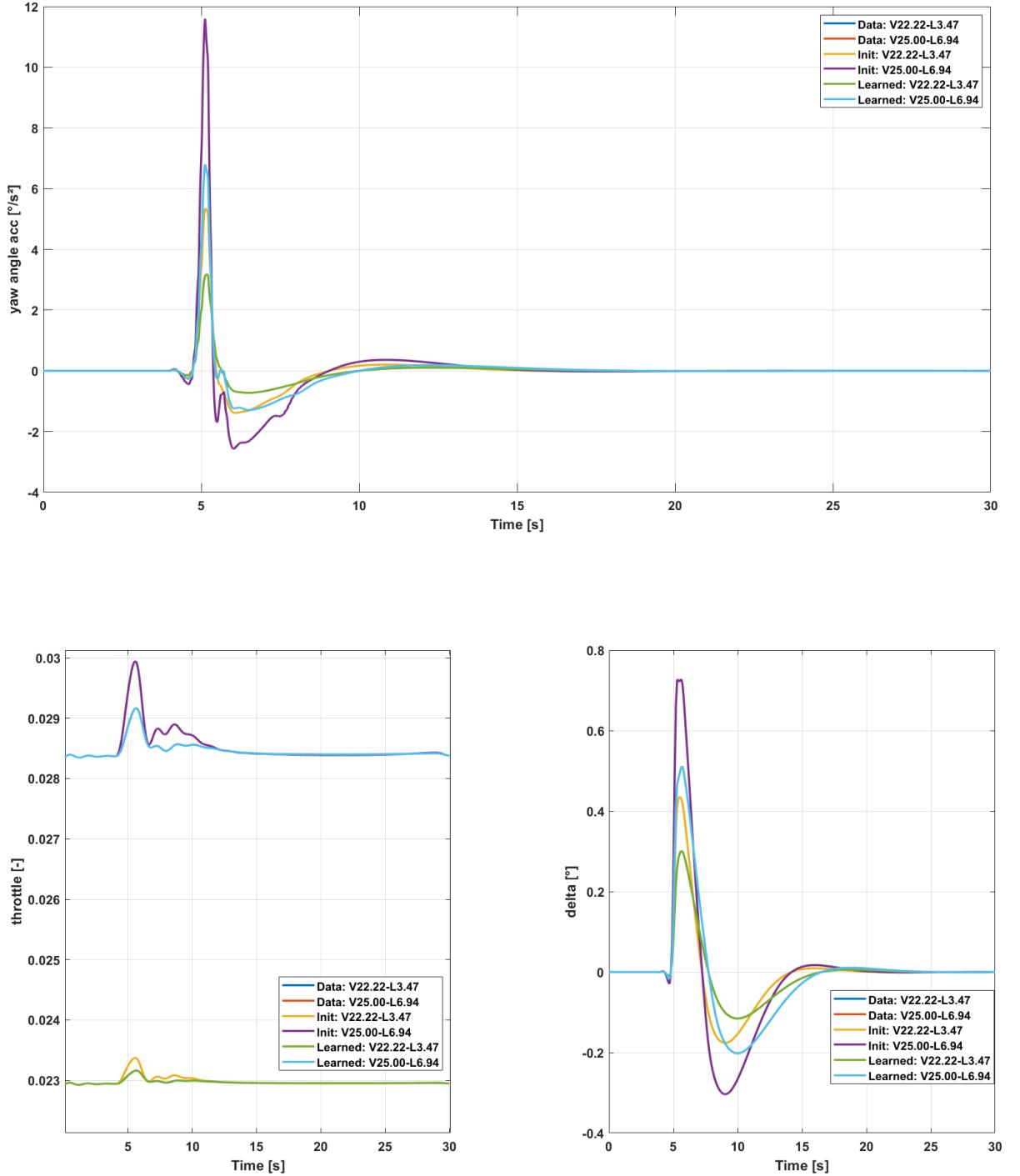


Figure E.1: This figure shows the amount of throttle and the angle of the front wheel of the bicycle model during the lane change maneuvers.

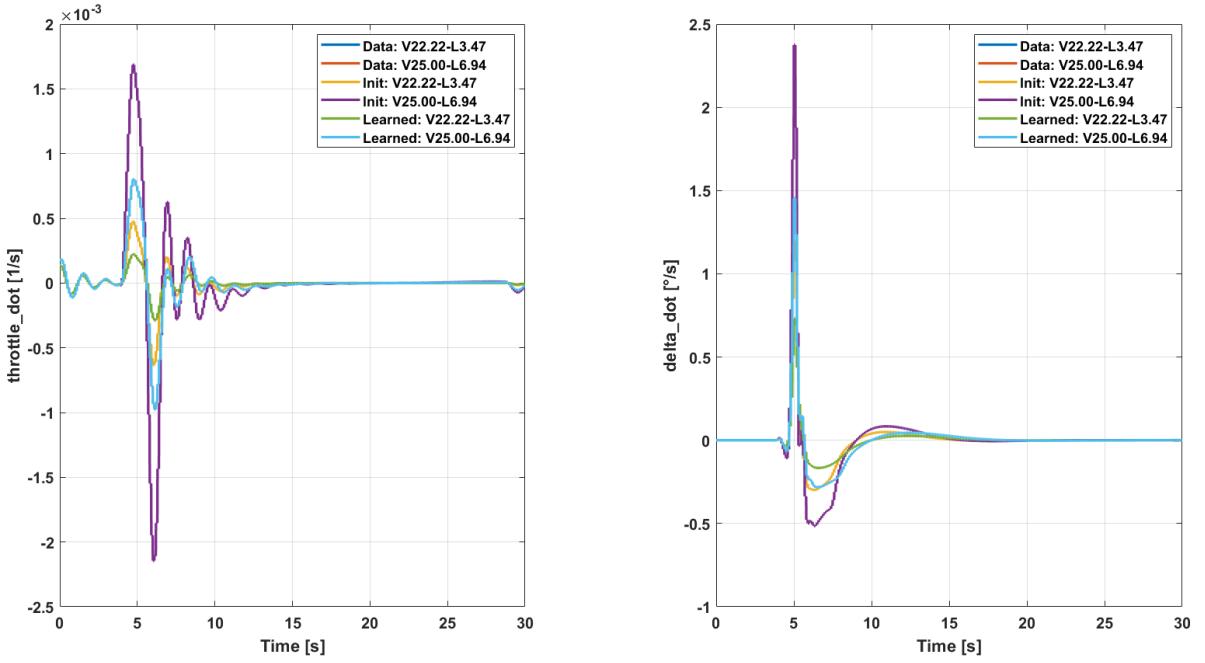


Figure E.2: This figure shows the amount of first derivative of throttle and the first derivative of the angle of the front wheel of the bicycle model is given as input during the lane change maneuvers.

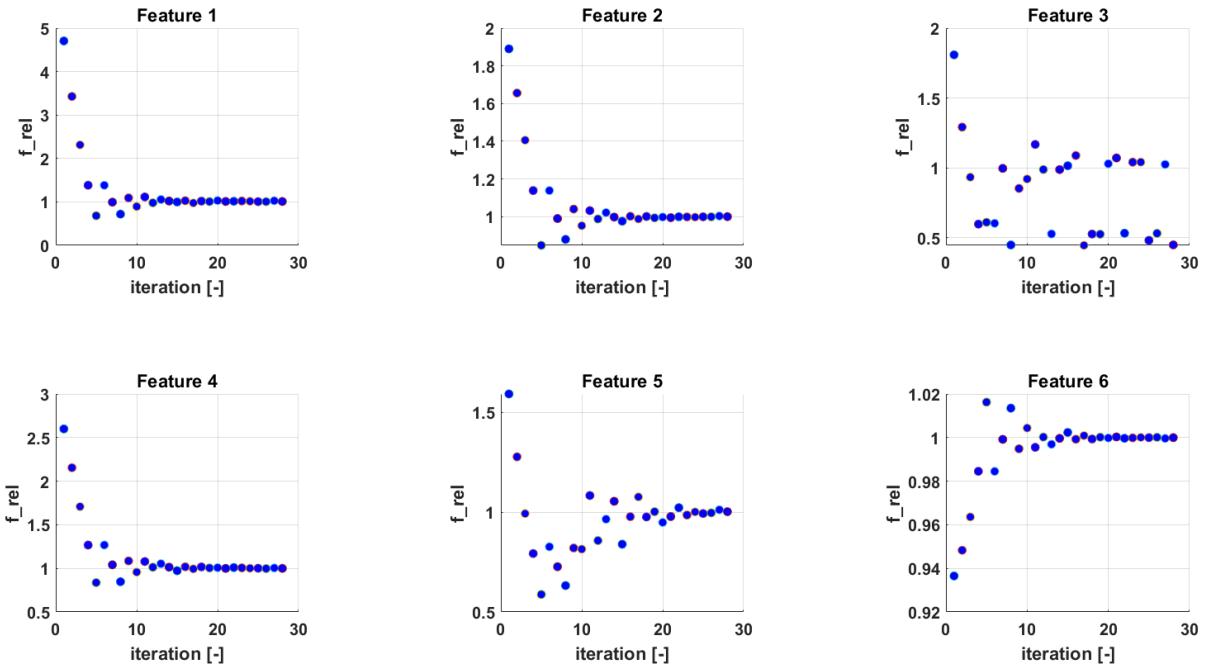


Figure E.3: The convergence of f_{rel} towards one during the learning iterations.

E. RESULTS OF LEARNING WITH THE 15 DOF AMESIM MODEL

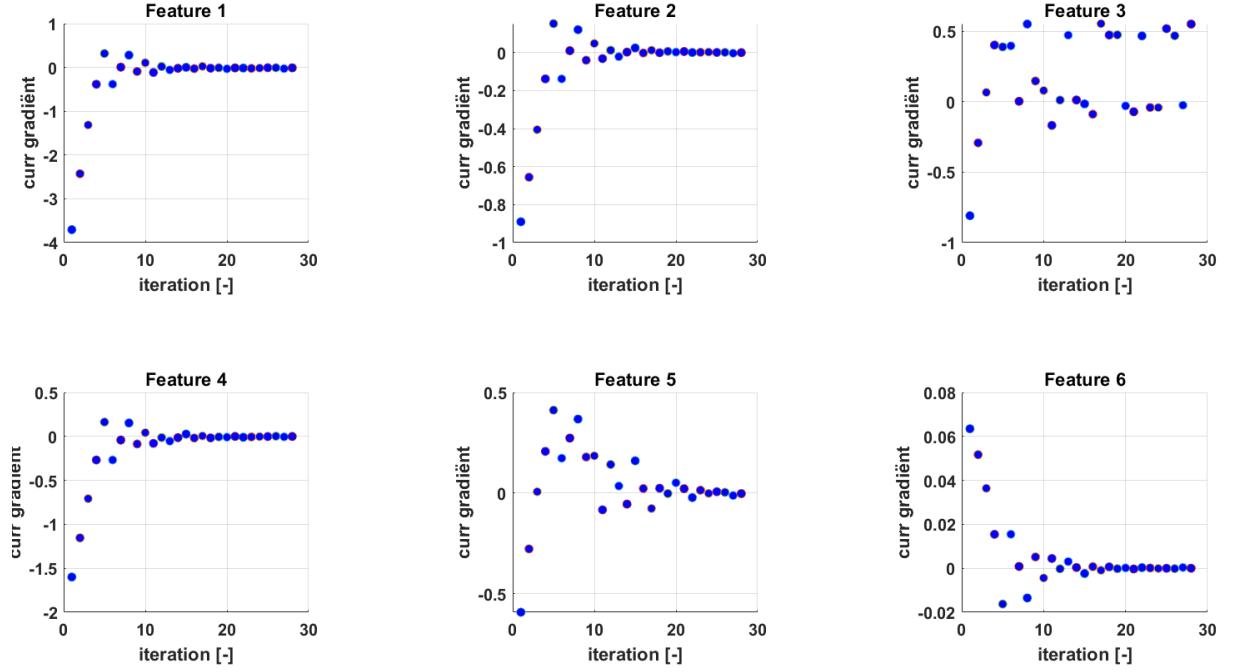


Figure E.4: The gradient $\frac{\partial F}{\partial \theta}$ estimated by $F_{obs} - F(\mathbf{r}_{expected})$ towards zero during the different learning iterations.

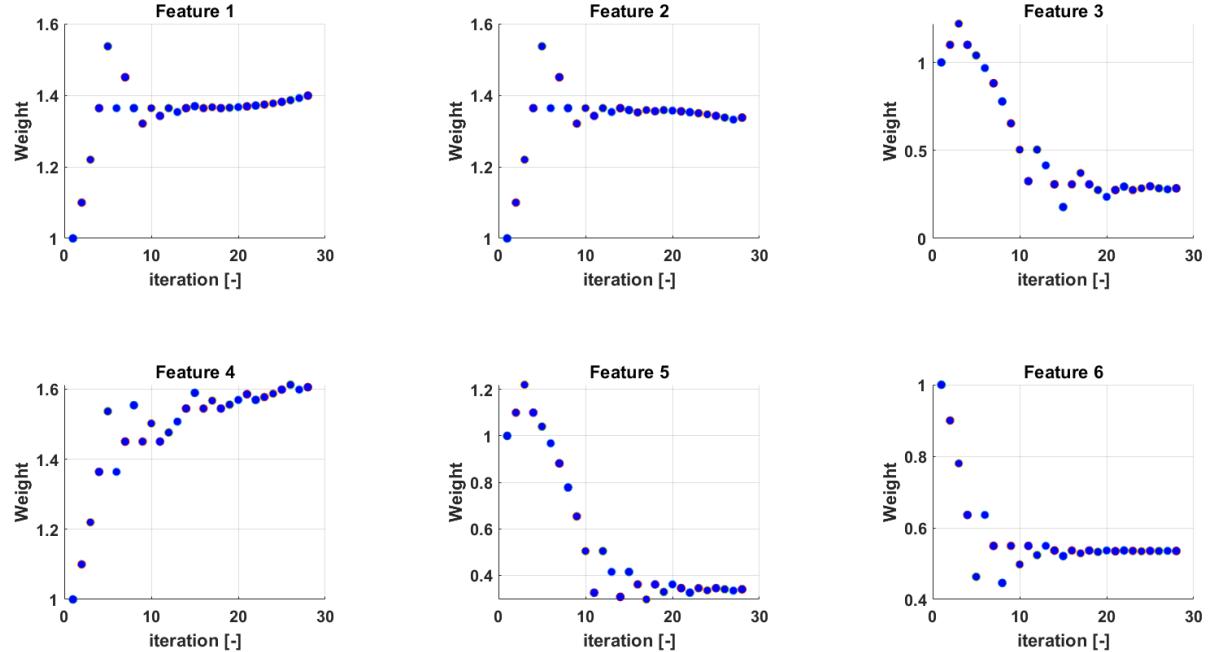


Figure E.5: The learned weighting factors during the different learning iterations.

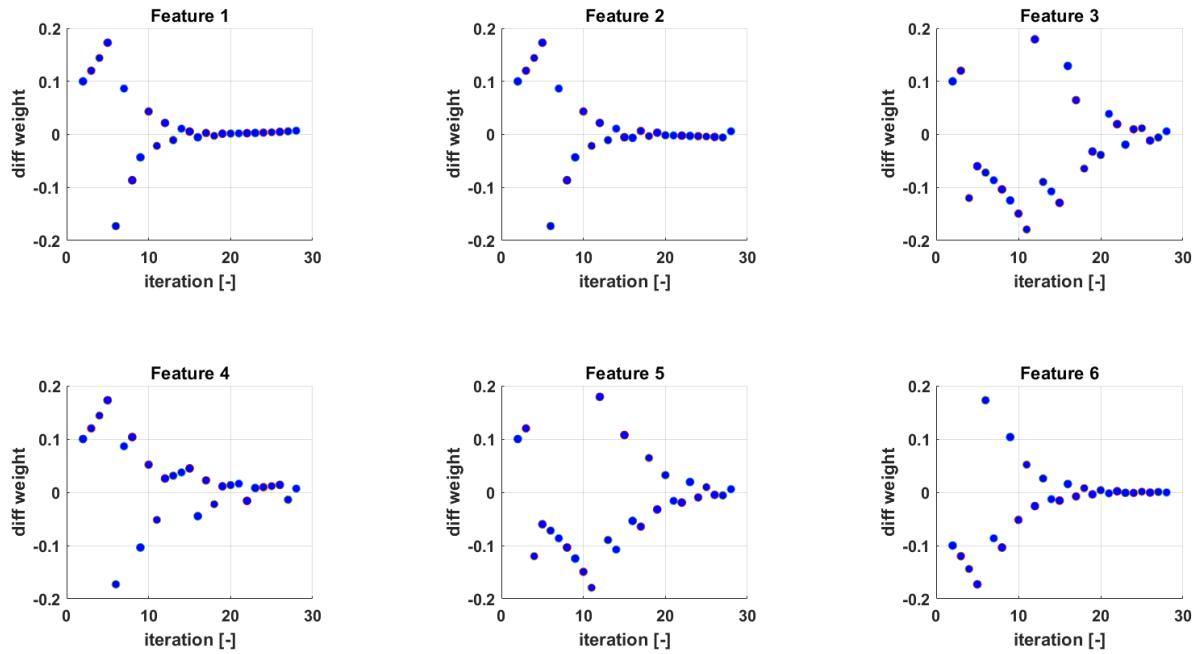


Figure E.6: The difference of θ with respect to one used in the previous iteration.

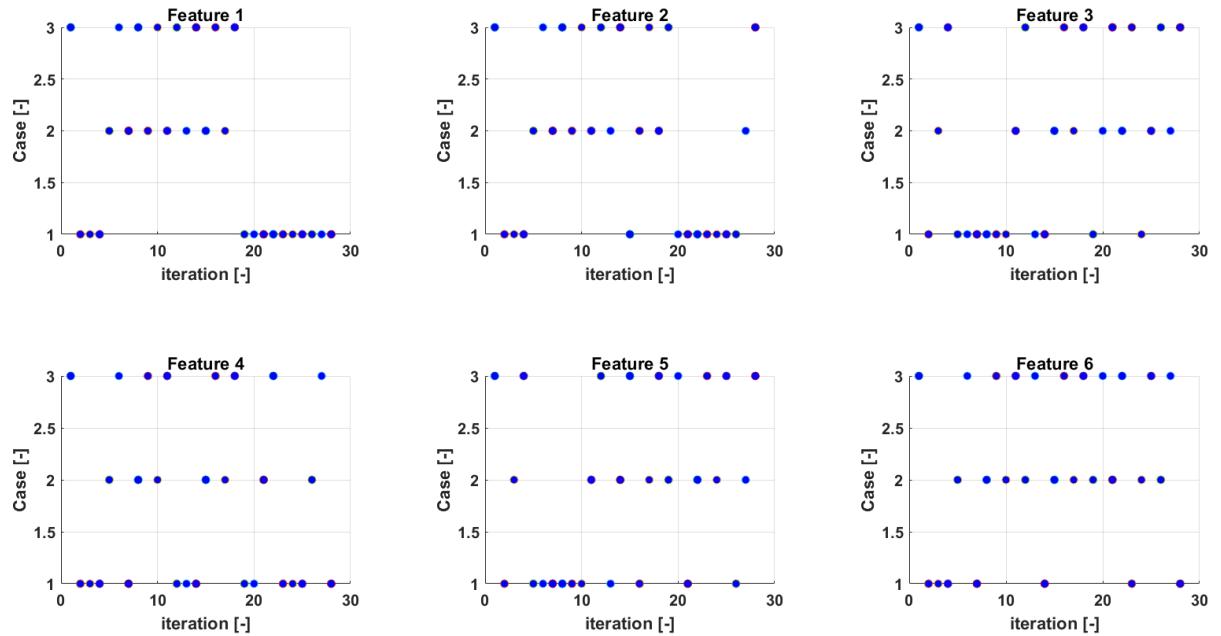


Figure E.7: This figure shows which case of the three available in the RPROP algorithm is chosen during the learning iterations.

Bibliography

- [1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. *Proceedings, Twenty-First International Conference on Machine Learning, ICML 2004*, pages 1–8, 2004.
- [2] O. Arenz. Inverse Optimal Control. *Encyclopedia of Machine Learning and Data Mining*, pages 678–678, 2017.
- [3] I. Bae, J. Moon, and J. Seo. Toward a comfortable driving experience for a self-driving shuttle bus. *Electronics (Switzerland)*, 8(9):1–13, 2019.
- [4] H. Bellem. *Comfort in Automated Driving : Analysis of Driving Style Preference in Automated Driving*. PhD thesis, 2018.
- [5] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell and A. K. Dey. Maximum Entropy Inverse Reinforcement Learning Brian. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, page 6, 2008.
- [6] L. Daniel. Method and system for determining and dynamically updating a route and driving style for passenger comfort - US Patent, 2018.
- [7] E. David. Will autonomous vehicles be safe to use? | Cybersecurity & Technology News | Secure Futures | Kaspersky.
- [8] M. Elbanhawi, M. Simic, and R. Jazar. In the Passenger Seat: Investigating Ride Comfort Measures in Autonomous Cars. *IEEE Intelligent Transportation Systems Magazine*, 7(3):4–17, 2015.
- [9] C. Gianna, S. Heimbrand, and M. Gresty. Thresholds for detection of motion direction during passive lateral whole-body acceleration in normal subjects and patients with bilateral loss of labyrinthine function. *Brain Research Bulletin*, 40(5-6):443–447, 1996.
- [10] J. Gillis. Ya Coda course presentation, 2019.
- [11] H. Kretzschmar, M. Kuderer, and W. Burgard. Learning to predict trajectories of cooperatively navigating agents. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 4015–4020, 2014.

BIBLIOGRAPHY

- [12] M. Kuderer, S. Gulati, and W. Burgard. Learning driving styles for autonomous vehicles from demonstration. *Proceedings - IEEE International Conference on Robotics and Automation*, 2015-June(June):2641–2646, 2015.
- [13] T. Mercy. *Spline-Based Motion Planning for Autonomous Mechatronic Systems*. PhD thesis, 2018.
- [14] J. Meyers. Lecture notes: Numerical Modelling in Mechanical Engineering. 2018.
- [15] P. Patrinos. Model Predictive Control - Lecture Notes, 2019.
- [16] P. Patrinos. Optimization - Lecture notes, 2019.
- [17] V. Powers, C., Mellinger, D., Kushleyev, A., Kothmann, B., Kumar. Experimental Robotics. *ISER*, June, 88(287513):515–529, 2013.
- [18] Prof. Amnon Shashua. Experience Counts, Particularly in Safety-Critical Areas | Intel Newsroom, mar 2018.
- [19] M. Riedmiller and H. Braun. Direct adaptive method for faster backpropagation learning: The RPROP algorithm. *1993 IEEE International Conference on Neural Networks*, pages 586–591, 1993.
- [20] Q. N. Tong Duy Son. Safety-Critical Control for Non-affine Nonlinear Systems with Application on Autonomous Vehicle. (August):7, 2019.
- [21] M. Turner and M. J. Griffin. Motion sickness in public road transport: The effect of driver, route and vehicle. *Ergonomics*, 42(12):1646–1664, 1999.
- [22] K. Yankov. *Potential Field Based Model Predictive Control for Autonomous Vehicle Motion Planning and Control*. PhD thesis.
- [23] Yusof N.B.M. *Comfort in Autonomous Car : Mitigating Motion Sickness by Enhancing Situation Awareness through Haptic Displays Nidzamuddin Md . Yusof*. PhD thesis, Eindhoven, 2019.