KU LEUVEN

FACULTEIT INGENIEURSWETENSCHAPPEN

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:
elektrotechniek, optie Elektronica en
geïntegreerde schakelingen

Promotor:
Prof. dr. ir. Jan Swevers

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

Begeleider:
dr. ir. Son Tong

Academiejaar 2019 – 2020

Preface

I would like to thank my family in the first place. During the history of my studies they always have been my biggest fans and I want to show my gratitude for the opportunities they have given me. I also want to thank my promoter Professor Swevers at the KU Leuven and Dr. Tong my mentor at Siemens for the professional discussions and tips they have given me in order to improve results. As last I also want to thank Flavia Acerbo, an employee at Siemens, who answered my questions on several occasions.

Stijn Staring

Contents

Abstract

The `abstract` environment contains a more extensive overview of the work. But it should be limited to one page.
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Samenvatting

In dit `abstract` environment wordt een al dan niet uitgebreide Nederlandse samenvatting van het werk gegeven. Wanneer de tekst voor een Nederlandstalige master in het Engels wordt geschreven, wordt hier normaal een uitgebreide samenvatting verwacht, bijvoorbeeld een tiental bladzijden.
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

List of Figures and Tables

List of Figures

List of Tables

List of Abbreviations and Symbols

Abbreviations

LoG	Laplacian-of-Gaussian

Symbols

42	"The Answer to the Ultimate Question of Life, the Universe, and Everything" according to [?]
c	Speed of light

Chapter 1

Introduction

1.0.1 Importance of topic

"Society expects autonomous vehicles to be held to a higher standard than human drivers." [16] This quote is setting the tone of the technology in autonomous driving. In order to be accepted to the public, autonomous vehicles should perform as least as good as the conventional human driver on parameters as for example safety. Despite widespread research on self-driving vehicles the acceptance by the user stays only limited.[2] The purchase behaviour of customers can be directly linked with comfort. Also in order 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 the questions what comfort during driving exactly is and how to measure it. A survey was conducted by researchers of the university of Warwick with as research question: Do passengers prefer autonomous vehicles be driven as full efficiency machines or in a way that emulates averaged human behaviour? [20] The result suggested that a blend of both is appealing the most confidence by user of a autonomous vehicle.
Driving comfort is a personal experience and also depend on the current emotional state of the driver. This means that more than one driving style for autonomous vehicle-driving should be identified for a certain vehicle. [22] The state of the driver can be communicated with the vehicle at the start of each ride and different driving styles can be obtained by changing the parameters in the path planning algorithm.

FIGURE 1.1: Concept visualization of autonomous driving. (source: [6])

1.0.2 Link with previous studies and problem formulation

In order to identify the specific comfort preferences of the driver that are quantized by these parameters, the vehicle should be able to learn them by demonstration. [11] Despite that each driver has its own preferences, they are based on a common notion of comfort where only different trade-offs are made. For example some drivers prefer more aggressive driving behaviour than others which will manifests itself in a different trade-offs of different comfort criteria than for example a defensive drivers. This will later in this thesis be translated into a comfort objective where different weights are used in order to quantify different comfort trade-offs made. This approach is in literature called inverse optimal control because it is learning the objective function of the comfort optimization. The lower the outputted value of this comfort objective, the higher the measurement of attained comfort will be in the later path planning algorithm.

In order to find comfort criteria 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 [19] 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 effect were found 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.[7] This means that higher order kinematic variables like accelerations and jerks also should be considered when measuring comfort.

1.0.3 Thesis objective

The goal of this thesis is to build further on the research of learning by demonstration [11] and to refine this idea in a good working practical application. The thesis is more concretely focussed in the ability to explain driver data and the practical implementation and validation of a learning algorithm that is able to capture user specific driving preferences in weights of a comfort objective function. The learning process is to be done offline and is based on an inverse optimal control approach. A comfort objective will be derived from literature to describe the common notion of comfort and this will be fitted on individual driver data to produce driver specific parameters. In the next step this objective function will be used in a path planning MPC formulation which will be calculating online feasible and comfortable paths whereafter an tracking MPC algorithm will follow it.

To conduct the inverse learning control there is first look at data generated by simulations where it is assumed that the vehicle is driving on an straight road and high way speed when executing manoeuvrers as lane changes and longitudinal accelerations.

An example lane change can be seen in Figure 1.2. Assumptions made during this manoeuvre To be able to make the generated data of high quality an MPC approach with a 15 degree of freedom vehicle model is used. Also in the learning algorithm itself a three degree of freedom non-linear bicycle model is used in order to adequately capture the different kinematic signals e.g. jerks and accelerations. Further there were comparisons made of different methods to learn from multiple datasets.

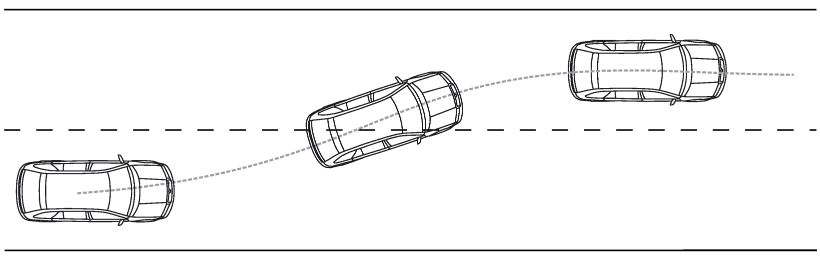

FIGURE 1.2: Example lane change as used as input in the inverse optimal control algorithm.

The execution of this research is conducted with the support of "Siemens Digital Industries Software - NVH R&D engineering department" located in Leuven which made it possible to preserve the direct link with reality. Software was made available e.g. Simcenter Amesim and the possibility to validate the obtained algorithms with real driver data made it possible to make the results that could be obtained more significant.

Chapter 2

Background in optimal control problems

This chapter gives some background information of the theory behind optimal control (OCP). After a global introduction and the discussion about the time discretization and shooting option, the chapter is giving an introduction into model predictive control (MPC).

2.0.1 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 some input and state constraints." [12].

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

$\boldsymbol{q}$ is called the state vector and contains all the states of the system. $\boldsymbol{u}$ is the control vector containing the controls. In theory the amount of states of an system can be infinite, but in order to sufficiently model a system only a good chosen set of states is needed to sufficiently describe the system. In vehicle control in order to describe the driving behaviour often kinematic variables like positions and velocities of the centre of gravity of the vehicle are chosen together with the yaw angle. This fully describes the position of a vehicle in a 2D plane. Typical controls $\boldsymbol{u}$ are steerwheelangle and the amount of throttle which can be directly linked the amount of propulsion force. Also higher order controls are possible.
The objective function l of the optimization problem 2.1 is integrating over the desired control horizon T. In the objective function it is indicated what should be minimized and it is in function of the different states and controls. $E(\boldsymbol{q}(T))$ describes

the terminal cost and the value of the integrated 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 $\dot{\boldsymbol{q}}(t) = \boldsymbol{f}(\boldsymbol{q}(t), \boldsymbol{u}(t))$ and the evolution of the vehicle states during the control horizon are retrieved by integration of this ODE. Q and U represent the constraints that can be put on respectively the optimization variables. It is worth noting that the control horizon time T by itself can also 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. [14]

There is a difference between soft and hard constraints. A hard constraint directly demarcates the feasible solution areas $\boldsymbol{q}(t) \in Q, \quad \boldsymbol{u}(t) \in U$. 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. [21]

2.0.2 Time discretization

The optimal control problem 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. There are several ways to do this which are summarized in Figure 2.1.[9]

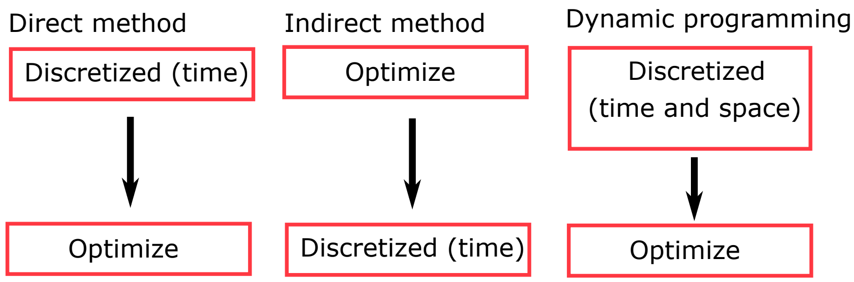

Figure 2.1: Overview of different discretization methods.

"Since direct methods are best suited to solve practically relevant OCPs" [12], this thesis is following the direct method. To implement the discretization of time when using the Direct method, a time shooting approach is used.

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 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 sampling rate should be taken high enough, while bearing in mind the extra optimization variables introduced and therefore calculation load. An other approach to take constraint violation between different sample points into account, is using spline based optimization formulations. [12]

Two different shooting approaches exist:

1. Multiple shooting (MS)
   During multiple shooting every new time sample $ns \in \mathbb{N}$ new states and controls are introduced and taken as optimization variables. Input changes are only allowed on the different time discrete time instances which leads to a piece wise control input signal. The blue bars indicate in Figure 2.2 (left) indicate the control value U_i that is 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}, discrete time integration is used according to $\boldsymbol{q}(k+1) = \boldsymbol{f}(\boldsymbol{q}(k), \boldsymbol{u}(k))$. These connections are put as constraints in the discretized optimization formulation of 2.1 and are called in literature 'path closing constraints' [9].
   Equation 2.2 shows 2.1 when discretized with the multiple shooting approach.

$$
\begin{aligned}
\min_{\boldsymbol{q}(.),\boldsymbol{u}(.)} \quad & \sum_{k=0}^{N-1} l_k(\boldsymbol{q}_k, \boldsymbol{u}_k) + E(\boldsymbol{q}_N) \\
\text{s.t.} \quad & \boldsymbol{q}_{k+1} = \boldsymbol{f}(\boldsymbol{q}_k, \boldsymbol{u}_k) \qquad k = [0, \cdots, N-1] \\
& \boldsymbol{q}_0 = \boldsymbol{q}_{measured} \\
& \boldsymbol{q}_k \in Q, \qquad k = [0, \cdots, N] \\
& \boldsymbol{u}_k \in U \qquad k = [0, \cdots, N-1] \\
& \boldsymbol{q}_N \in Q_f, \quad N \in \mathbb{N}
\end{aligned}
\tag{2.2}
$$

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 and the applied controls. The mathematical formulation of this is shown by 2.3. In Figure 2.2 (right) are the states that result from integration indicated in green. [9]

$$
\begin{aligned}
\boldsymbol{q}^1 &= \boldsymbol{f}(\boldsymbol{q}^0, \boldsymbol{u}^0) \\
\boldsymbol{q}^2 &= \boldsymbol{f}(\boldsymbol{f}(\boldsymbol{q}^0, \boldsymbol{u}^0), \boldsymbol{u}^1) \\
\boldsymbol{q}^3 &= \boldsymbol{f}(\boldsymbol{f}(\boldsymbol{f}(\boldsymbol{q}^0, \boldsymbol{u}^0), \boldsymbol{u}^1), \boldsymbol{u}^2) \\
&\cdots
\end{aligned}
\tag{2.3}
$$

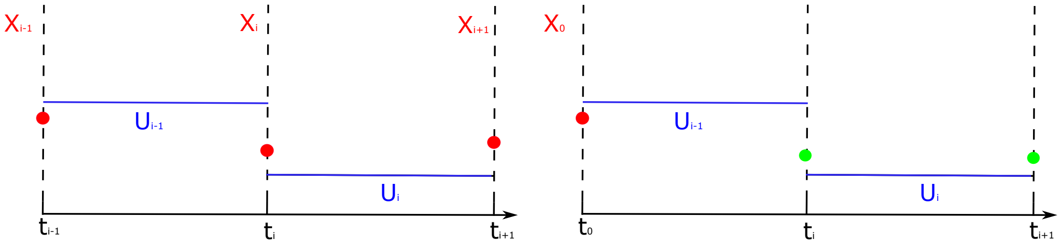

Figure 2.2: Schematic view of the time shooting approaches (left: multiple shooting; right: single shooting).

In this thesis a multiple shooting approach is used together with the use of a Runge-Kutta integration scheme. Runge-Kutta is an explicit integration scheme which has a higher calculation cost 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. [12]

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 control input which means they can be used in calculations more efficiently. In single shooting every state depends on the begin state and different controls which gives smaller, more populated matrices which are more calculation expensive. [9]

2.0.3 Model predictive control

In slow changing environments as for example a chemical plant, MPC is already a mature approach. 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 efficiently algorithms. [12]. In the next section a short introduction of the formulation is made.

MPC is an approach were optimizations are solved in a loop in order to be able to notice disturbances, changing environments and model-plant mismatches. Therefore it makes use of a moving time horizon[1]. Every iteration an OCP is solved over the prediction horizon and as result control signals are given as output. Only the first control is applied for one time interval of the finite prediction horizon $N \cdot T_s$ as is visualized in Figures 2.3and 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. [18, 12]. In Figure 2.3 the solving of an OCP during one MPC iteration on time sample $t+1$ is depicted. Figure 2.4 shows than only the first control of the obtained control signal

[1]There are also other implementations e.g. shrinking time horizon, where the prediction time gets smaller every step.

will be applied, which induces the system to come in a different state which will be the new start state and a new OCP will be solved. This will go on until the final desired conditions are met. A single OCP iteration only takes the model of the system into account as is done in a feedforward controller but through the iterative way of solving the OCPs, the MPC can still deal with unexpected acquired states due to feedback that is given by each initial state. As stated earlier the downside of this approach is a bigger computational load, which makes efficiently written software a necessity. [13]

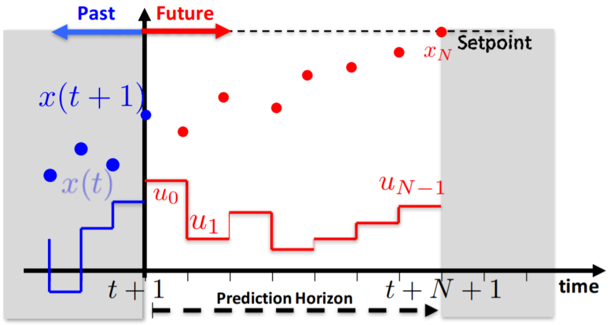

FIGURE 2.3: Visualization of the optimal control problem solved in one iteration of the MPC (Source: [13]).

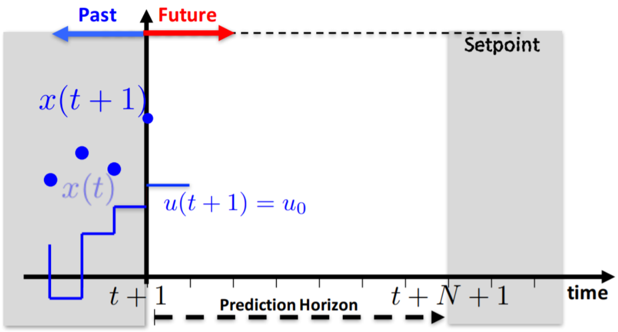

FIGURE 2.4: Visualization of the application of the first step of the calculated control signal during one iteration of the MPC (Source:[13]).

Chapter 3

State of the art modelling of comfort

As discussed in the introduction the goal of the thesis is to learn and implement a method to capture personal experience of comfort in autonomous driving. This will be done by using an inverse optimal control approach where the weights are learned from demonstration. To be able to do this it is necessary that a literature study is done about how to define comfort in a vehicle and to gain information about inverse optimal control.
This chapter will give an overview of the literature that is available and will show how the thesis fits in earlier conducted research.

3.1 What are the parameters that define comfort during driving?

In the following US patent [5] the idea is to assess the amount of comfort by calculating a value for carsickness. This value is calculated by a weighted sum of the 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' [7], 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 also have to be taken into account in 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 hypotheses is that motion sickness will be more prominent in autonomous driving due to the loss of control. Is is also argued that the amount of travel time and the distance to an obstacle are naturally parameters that contribute to a comfortable feeling.

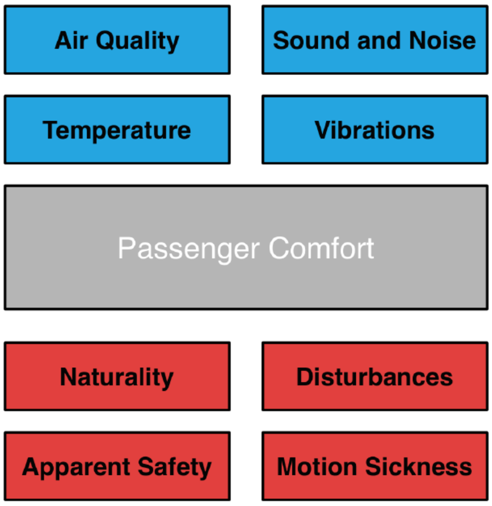

Figure 3.1: Overview of comfort parameters in autonomous vehicle with old parameters (blue) and new ones (red).(Source: [7])

In 'Analysis of Driving Style Preference in Automated Driving' [3] three studies are conducted in order to capture the definition of comfortable driving in autonomous vehicles. In the first study drivers drove manually with their own driving styles and this data was used in order to look for relevant metrics that could be used in defining distinct driver styles. It was found that these metrics are varying across the maneuver. This is a logical result because not in all maneuvers are all the comfort matrices equally dominant.
The results of this research is that accelerations play a key role in comfort but are not the only factor. [3]

The different comfort metrics found:

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

Quickness of completing a lane change or an acceleration maneuver could respectively be modelled as: $q_L = \frac{\frac{\int velocity \cdot dt}{Time}}{\Delta distance}$ and $q_A = \frac{\frac{\int acceleration \cdot dt}{Time}}{\Delta velocity}$.

Comfortable driving as being assessed in [3] can be summarized as driving with small accelerations and jerk to obtain sufficient smooth trajectories and keeping sufficient headway distance in order to have a feeling of control and safety. These results suggest that when an algorithm to assess comfort is suggested for autonomous vehicles, a notion of the surrounding traffic should be inherently present. It was found that in order of the traffic density 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 a feeling of a fast responds of the vehicle translating in early peaks of acceleration. Vertical vibrations come not into scope when roads are assumed sufficiently smooth.

In a second study the main metrics that were found from study one are varied and combinations are rated by the use of a survey about the amount of comfort. "Out of this it followed that accelerations are playing a key factor." [3] For lane changes it could be concluded that the maneuvers with a small lateral acceleration and early perceivable onset were more comfortable. It is further also hypothesised that the location of the acceleration peaks and the amount of symmetry of the acceleration signal also plays a role when the amount of comfort is rated.

The relation between jerks and comfort is also confirmed by [8] where it is stated that: "Jerk has been shown to elicit a stronger influence on comfort than acceleration".

3.1.1 Conclusion

In order to find parameters of driving comfort that will further on in the thesis will be used as comfort criteria, a literature study is conducted which is mainly questionnaire based. The results are that in order to be able to assess comfort higher order kinematic variables as accelerations and jerks should be taken into account. These are important variables in order to obtain a smooth vehicle path and to give a continuous and natural feeling when driving. Also should the quickness of the maneuver and the feeling of safety of the driver be taken into account. As quoted by [19] states that low frequency lateral acceleration are the mean responsible for carsickness. In addition comfort assessment is influenced by the environment because in busier traffic there is a higher tolerance for less smooth trajectories. It was also found that the effect of a fast reaction of vehicle at the start of a maneuver is influencing the amount of comfort perceived in a positive way.

3.2 Inverse reinforcement learning

Because every human has its own driving style it is a cumbersome task to tune these parameters for each individual in order to model a personal perception of comfort. In [15] it is showed that manual tuned parameters will lead besides also to sub-optimal solutions in comparison when the parameters are learned. That is why it is chosen in this thesis to derive these parameters by demonstration of a human driver.

The goal of the learning algorithm explained in this thesis is to derive the parameters of different linearly combined comfort criteria combined in the costfuntion J that when optimized as best as possible explain the observed data. As the match with the observed data gets better, the model as suggested by equation 3.1 is getting closer to the inner comfort model of the individual driver. However it should be noted that the driver is taken unknowingly a lot of comfort criteria into account and they are not always linearly relating. Therefore the suggested comfort cost function consisting of a linear combination of comfort criteria will always be an approximation of the reality. Yet as discussed in [11] the results suggest that the magnitudes of the quantities that contribute to the comfort of the user are obtained.

$$J = \sum_{j=1}^{N} \theta_j \cdot f_j \tag{3.1}$$

In order to create a generative model that create vehicle paths r_i with equivalent kinematic characteristics as the path that was observed $\tilde{r_i}$, a feature-based inverse reinforcement learning is applied. [11, 1] With $i \in [1...m]$ and m the amount of observed trajectories. A feature is encoding relevant kinematic properties and the difference between the demonstrated and calculated features says something about the similarity of the kinematic vehicle signals.

3.2.1 Background on learning algorithm

A feature values maps a trajectory onto a scalar and the higher the scalar value the more discomfort is experienced. An example of a that measures the amount of accelerations in a maneuver is given by equation 3.2.

$$f : \boldsymbol{r} \rightarrow f(\boldsymbol{r}) = \int_0^T \ddot{x}(t)^2 + \ddot{y}(t)^2 dt \tag{3.2}$$

The path that that the centre of gravity of the vehicle is following can be represented by: 3.3.

$$\boldsymbol{r} : t \rightarrow \boldsymbol{r}(t) = \left(\begin{smallmatrix} x(t) \\ y(t) \end{smallmatrix}\right) \tag{3.3}$$

The driver is not a deterministic agent and is modelled by a stochastic distribution: $p(\boldsymbol{r}|\boldsymbol{\theta})$ For certain weights $\boldsymbol{\theta} \in \mathbb{R}^N$ a path $\boldsymbol{r}$ is produced as being a sample of a stochastic distribution. The distribution that is chosen for this is the distribution of maximum entropy (equation: 3.4). [4, 10]. This describes the data best because it is the distribution that is least biased. [11] The distribution with the highest entropy represents the given information best since it does not favour any particular outcome besides the observed constraints. [1]

$$p(\boldsymbol{r}|\boldsymbol{\theta}) = exp(-\boldsymbol{\theta}^T \cdot \boldsymbol{F}(\boldsymbol{r})) \tag{3.4}$$

Equation 3.4 can be interpreted as a linear costfunction $\boldsymbol{\theta}^T \boldsymbol{F}(\boldsymbol{r})$ where agents are exponentially more likely to select trajectories with lower cost. [11] The observed feature vector $\tilde{\boldsymbol{F}} \in \mathbb{R}^N$ has on its entries the different feature values $\tilde{f}_i$ with $i \in [1...m]$. The averaged observed feature vector is $\tilde{\boldsymbol{F}}_{av} = \frac{1}{m} \sum_{j=1}^{m} \tilde{\boldsymbol{F}}_j$.

In order to be able to explain the averaged observed feature vector, the weights $\boldsymbol{\theta}$ need to be found that match the expected features vector obtained from the distribution $\boldsymbol{F}(r_{expected})$ with the averaged observed features vector $\boldsymbol{F}_{obs}$. $\boldsymbol{r}_{expected}$ defined as $E(p(\boldsymbol{r}|\boldsymbol{\theta}))$. To make the expected features vector match the averaged observed features vector, gradient descent method can be used with as gradient $\boldsymbol{F}_{obs} - \boldsymbol{F}(r_{expected})$. Equation 3.5 summarizes the gradient descent method with α the step size taken in the direction of descent.

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^{k} - \alpha \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{\theta}}^{k} \tag{3.5}$$

When $\boldsymbol{\theta}_{optimal}$ is found the gradient is minimized and the features vectors will match as closely as possible. "When the feature expectations match, guaranteeing that the learner performs equivalently to the agent's demonstrated behaviour regardless of the actual reward weights the agent is attempting to optimize" [1]

In order to calculate $\boldsymbol{F}(\boldsymbol{r}_{expected})$ a Hamiltonian Markov chain Monte Carlo stochastic distribution sampling approached is used in [10]. In [11] a simplified method and less calculation demanding approach is proposed where it is assumed that the expected path is the one that is been assessed as the most comfortable by the driver and is thus the path that is minimizing 3.1 for certain weights. In order to be able to match with the observed data it is hereby assumed that the demonstrations not only are samples of a stochastic distribution but are also generated by optimizing an inner cost function which is similar to 3.1. Equation 3.6 summarizes the approximation proposed by [11].

$$\boldsymbol{r}_{expected} = \underset{\boldsymbol{r}}{argmax}\; p(\boldsymbol{r}|\boldsymbol{\theta}) = \underset{\boldsymbol{r}}{argmin}\; \boldsymbol{\theta}^{T} \cdot \boldsymbol{F}(\boldsymbol{r}) \tag{3.6}$$

$$\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{\theta}} = \boldsymbol{F}_{obs} - \boldsymbol{F}(\boldsymbol{r}_{expected}) \tag{3.7}$$

It is checked in chapter 4 if this approach that is assuming that demonstrations are generated by minimizing a cost function also gives acceptable results when the assumption is violated.

3.2.2 RPROP algorithm

From [14] it is known that not every step size in the opposite direction of the gradient is leading to the convergence towards a minimum. When the step size is too small it will take a long time to convergence. However when it is chosen too big, cycling behaviour between limit points can occur. In order to avoid this kind of unwanted behaviour a line-search method is needed in order to change the step size taken during the course of the algorithm. For this the Resilient backpropagation algorithm (**RPROP**) [17] 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 weights for the start of the next iteration in the gradient descent optimization method. The update value Δu is solely dependent of the sign of the current gradient and the sign of the gradient in the previous iteration. The main equations of the algorithm are given for each by:

$$\Delta u_i^t = \begin{Bmatrix} \eta^+ \cdot \Delta u_i^{t-1}, & if\ \frac{\partial f_i}{\partial \theta_i}^t \cdot \frac{\partial f_i}{\partial \theta_i}^{t-1} > 0 \\ \eta^- \cdot \Delta u_i^{t-1}, & if\ \frac{\partial f_i}{\partial \theta_i}^t \cdot \frac{\partial f_i}{\partial \theta_i}^{t-1} < 0 \\ \Delta u_i^{t-1}, & if\ \frac{\partial f_i}{\partial \theta_i}^t \cdot \frac{\partial f_i}{\partial \theta_i}^{t-1} = 0 \end{Bmatrix} \tag{3.8}$$

When the update value of the weight is determined it is applied in the direction of steepest descent which equals the opposite direction of the current gradient.

$$\Delta \theta_i^t = \begin{Bmatrix} -\Delta u_i^t, & if\ \frac{\partial f_i}{\partial \theta_i}^t > 0 \\ +\Delta u_i^t, & if\ \frac{\partial f_i}{\partial \theta_i}^t < 0 \\ 0, & else \end{Bmatrix} \tag{3.9}$$

Exception on 3.9:

$$\Delta \theta_i^t = -\Delta \theta_i^{t-1},\ if\ \frac{\partial f_i}{\partial \theta_i}^t \cdot \frac{\partial f_i}{\partial \theta_i}^{t-1} < 0 \tag{3.10}$$

$$0 < \eta^- < 1 < \eta^+ \tag{3.11}$$

and with $\theta_i^{t+1} = \theta_i^t + \Delta\theta_i^t$, $i \in \mathbb{N}_{[1 \cdots N]}$ and $t \in \mathbb{N}_[1 \cdots \tau]$ the amount of iterations. Every time the partial derivative $\frac{\partial f_i}{\partial \theta_i}^t$ of the corresponding weight θ_i^t changes its sign, it is assumed that the last update was too big and the local minimum was passed. In 3.8 the step size is reduced and in order to go back the previous situation the update of the weight is done as 3.10. In order to not again decrease the update value when going back to the previous situation, in the next iteration $\frac{\partial f_i}{\partial \theta_i}^{t-1}$ is set equal to zero. If the derivative retains its sign, the update-value is slightly increased in order to accelerate convergence in shallow regions." [17] Parameters chosen bij the user are Δ_0, η^+ and η^-. In this thesis following values were chosen: $\Delta_0 = 0.1$, $\eta^+ = 1.2$ and $\eta^- = 0.5$.

3.2.3 Optimization principles

The optimization tool used is the CasADi software. This section discusses the main optimization principles used under the hood and discusses the IPOPT solver and SQP method. (See Handbook opti and slides CasADi 3 lecture vooral)

Conclusion

A background from literature on the inverse reinforcement learning algorithm used in this thesis was given. First inverse reinforcement learning was explained and how it could help to solve the research question how to learn comfort from observations.

Afterwards a more detailed background on the formulation of the specific learning algorithm was given, which was complemented with a discussion on the algorithm used to update the weights being part of the gradient descent method to match expected feature values with observed ones.

Chapter 4

Learning from ideal data

This chapter is focussing on the implementation of the inverse reinforcement learning idea that is used to learn the different weights in the comfort cost function $\boldsymbol{\theta}^T \boldsymbol{F}(\boldsymbol{r})$ explained in chapter 3. The use of ideal data is assumed which means that vehicle model mismatch is avoided by using the same model for learning the weights as generating the data. Thereby is the approximation that comfort of a driver can be modelled by a simple linear relation of features exactly for-filled. Data is generated by choosing a set of weights and generating kinematic vehicle signals by the minimization seen in equation 3.6. Perfect learning occurs when the chosen weights used in generating the data are found back.

First the non-linear bicycle model with the used parameters is explained. Further on, the concrete used algorithm is discussed. After that a detailed discussion is given about the simulations done. It is worth noting that the learning process discussed, concerns an offline optimization which allows for higher calculations loads because of the absence of binding real time constraints.

4.1 Non-linear bicycle model

The free body diagram of the non-linear bicycle model can be seen in figure 4.1.

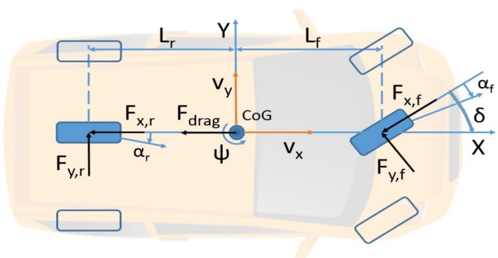

FIGURE 4.1: Non-linear vehicle bicycle model (Source: [18]).

Two variants of this model are discussed which are differentiated by the smoothness of the controls that can be received.

1. Model has 6 states and 2 inputs:

$$\boldsymbol{X} = \begin{bmatrix} x & y & v_x & v_y & \psi & \dot{\psi} \end{bmatrix}^T \textit{ and } \boldsymbol{U} = \begin{bmatrix} t_r & \delta \end{bmatrix}^T \tag{4.1}$$

2. Model has 10 states and 2 inputs:

$$\boldsymbol{X} = \begin{bmatrix} x & y & v_x & v_y & \psi & \dot{\psi} & t_r & \delta & a_x & a_y \end{bmatrix}^T \textit{ and } \boldsymbol{U} = \begin{bmatrix} \dot{t_r} & \dot{\delta} \end{bmatrix}^T \tag{4.2}$$

x and y in the above formulation are the position of the centre of gravity of the car 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 input vector of 4.1 consists of the throttle control input t_r and the angle of the front wheel δ. In the more extended bicycle model of 4.2 throttle and frontwheelangle 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 frontwheelangle.

The equations of motion derived and checked in literature [18] are[1] :

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

The drag force is calculated as:

$$F_{drag} = C_{r0} + C_{r1} v_x^2 \tag{4.4}$$

[1] Appendix A shows the fully worked out jerk equations j_x and j_y.

, 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 tire model. The longitudinal tyre forces are calculated as:

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

R_w is the wheel radius and T_{max} a measure for 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 is 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 slipangles α_f and α_r:

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

, resulting in:

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

The use of this linearised lateral tyre model is valid for small lateral accelerations ($a_y <= 4m/s^2$) and slip angles ($\alpha <= 5^o$) [18]. It is acceptable to use this approximation model in this thesis as the goal is learn a comfortable and thus smooth lane change manoeuvre. These constraints will also be checked during the section 4.2.4.

The vehicle model parameters used are given in table 4.1. These correspond to common used vehicle parameters as provide by Siemens Digital Industries Software - NVH R&D engineering department. The $Gsteerfactor$ approximates linearly the relation between the frontwheelangle and the steeringwheelangle turned by the driver by the relation $\delta = \frac{\delta_s}{Gs}$.

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
Gsteeringfactor Gs [-]	16.96

Table 4.1: Used vehicle model parameters.

4.2 Formulation of the algorithm

The goal of the learning algorithm is to learn the weights $\boldsymbol{\theta}$ in the pre-defined comfort objective function: $\boldsymbol{\theta}^T \boldsymbol{F}(\boldsymbol{r})$. The features that are the entries of the feature vector $\boldsymbol{F}(\boldsymbol{r})$ capture a notion of comfort felled by the driver. Based on the literature study conducted in Chapter 3 and paper [11], the amount of discomfort can be modelled by the features discussed in 4.2.2 during timespan T of the maneuver. The scenario of a lane change on a highway is chosen. The time horizon where over the minimization of the comfort objective is itself also an optimization variable T. The simulations done in this thesis were performed on a notebook provided by Siemens with Intel Core i7-7920HQ CPU @ 3.10GHz and 32 GB of RAM memory.

4.2.1 Flow of the algorithm

The goal of the learning algorithm is to output weights that when applied in the objective $\boldsymbol{\theta}^T \boldsymbol{F}(\boldsymbol{r})$, generate feature vector $\boldsymbol{F}(\boldsymbol{r})$ that are the best possible fit with the observed feature vector. Without a vehicle mismatch a match of feature values will directly induce a good match of the kinematic signals as is extensively discussed in section 4.3. This means that similarity between the observed path and the one that follows from minimizing the objective $\boldsymbol{\theta}^T \boldsymbol{F}(\boldsymbol{r})$ for chosen weights, is quantified by the difference between the feature values of the observed path and the obtained one. The flow of a single data set learning algorithm can be seen in Figure 4.2. The path that is expected to be produced by the driver is the path that is felt the most comfortable and equals $\boldsymbol{r}_{expected} = \underset{\boldsymbol{r}}{argmin}\, \boldsymbol{\theta}^T \cdot \boldsymbol{F}(\boldsymbol{r})$

The learning is started by guessing a set of weights e.g. all equal to one. Equation

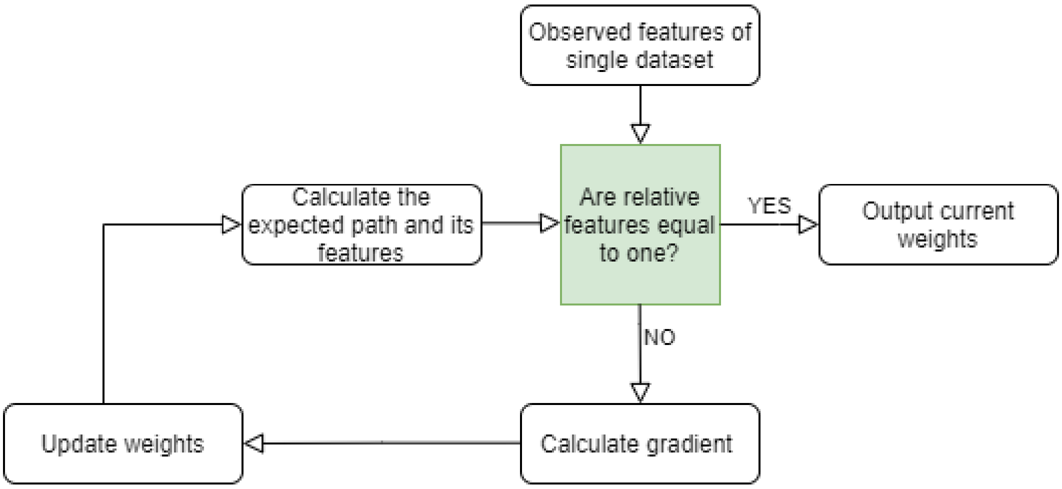

FIGURE 4.2: Basic flow of the reinforced learning algorithm.

3.6 is minimized in order to generate an expected path for these weights. From this calculated path, features for the observation and the calculated path from optimization 3.6,further addressed as learned features, can be retrieved by using the definition of the features discussed in 4.2.2.From this the relative features $f_{rel,i}$ are calculated by dividing the learned features by the observed features. 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}.

While no convergence takes place the weights are updated, making use of the RPROP algorithm explained in Chapter 3. In order to apply the RPROP method only the sign of the gradient is used which means that the size of the gradient is decoupled from the update value of the weights. The gradient is given by $\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{\theta}} = \boldsymbol{F}_{obs} - \boldsymbol{F}(\boldsymbol{r}_{expected})$ and the update of the weight is achieved by applying the gradient method according to $\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - |\Delta\theta^k| sign(\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{\theta}}^k)$ where $\Delta\theta^k$ is calculated according section 3.2.2. If follows that the weight θ_i is decreased if $f_{obs,i} > f_i(\boldsymbol{r_{exptected}})$ and increased when $f_{obs,i} < f_i(\boldsymbol{r}_{exptected})$. the new weights can be used in the minimization of the comfort objective for a set of weights given by 3.6. Next a more detailed description of how this is done is given by 4.8 which uses as vehicle model 4.2.

$$
\begin{aligned}
\min_{\boldsymbol{X}(.),\boldsymbol{U}(.),T} \quad & \boldsymbol{\theta}^T \boldsymbol{F}(\boldsymbol{X},\boldsymbol{U},T) \\
\text{s.t.} \quad & \boldsymbol{X}^{k+1} = I(\boldsymbol{X}^k,\boldsymbol{U}^k) \quad && k=[0,\cdots,N-1] \\
& \boldsymbol{X}^0(1:8) = \boldsymbol{X}_{intitial} \\
& T \leq T_{limit} \\
& \boldsymbol{F}(\boldsymbol{X}^k) \geq 0 && k=[0,\cdots,N] \\
& \boldsymbol{G}(\boldsymbol{U}^k) \geq 0 && k=[0,\cdots,N-1] \\
& \boldsymbol{H}(\boldsymbol{X}^k) = 0 && k=[0,\cdots,N] \\
& \boldsymbol{X}^k \in \mathbb{R}^{6x1} && k=[0,\cdots,N] \\
& \boldsymbol{U}^k \in \mathbb{R}^{2x1} && k=[0,\cdots,N-1] \\
& T \in \mathbb{R}, \quad N,i,l \in \mathbb{N}
\end{aligned}
\tag{4.8}
$$

Where $\boldsymbol{X} \in \mathbb{R}^{10xN+1}$ and $\boldsymbol{U} \in \mathbb{R}^{2xN}$ contain respectively the states and controls 4.2 during the maneuver, complemented with the time of the maneuver T. In order to discretize time, a multiple shooting approach is adopted as is explained in section 2.0.2. The amount of integration intervals N is chosen equal to 1000 and is the same as the amount of control intervals in order to go from the initial state towards the end state. With this a sampling time of 0.025 s is obtained as is discussed in section 4.2.4. Inside the function I the Runge-Kutta time integration method is embedded in order to connect different states over time when a certain control is applied for ΔT. Here the equations of motion 4.3 are inputted in the optimization because to perform the integration, derivatives of the vehicle states are used. The time discretization used is can be categorized as a direct method and a multiple shooting approach is used which means that at every time instance a new set of state optimization variables is introduced as is explained in section 2.0.2. The path constraint vectors $\boldsymbol{F} \in \mathbb{R}^{L_1x1}$ and $\boldsymbol{G} \in \mathbb{R}^{L_2x1}$ demarcate together with the equality constraint vectors $\boldsymbol{H} \in \mathbb{R}^{Q_1x1}$ and $\boldsymbol{J} \in \mathbb{R}^{Q_2x1}$ the feasible area of the solutions for $\boldsymbol{X}$, $\boldsymbol{U}$ and T. An overview of these constraints is given by equations 4.9, 4.10 and 4.11.

$$
\boldsymbol{F} = \left\{
\begin{matrix}
-\frac{Width\ Lane}{2} \leq y^k \leq \frac{3 \cdot Width\ Lane}{2}, & k=[0,\cdots,N] \\
0 \leq x^k, & k=[0,\cdots,N] \\
-\frac{\pi \cdot 150}{180 Gs} \leq \psi^k \leq \frac{\pi \cdot 150}{180 Gs}, & k=[0,\cdots,N]
\end{matrix}
\right\}
\tag{4.9}
$$

$$
\boldsymbol{G} = \left\{ -1 \leq t_r^k \leq 1, \quad k=[0,\cdots,N-1] \right\}
\tag{4.10}
$$

It is not necessary to introduce physical vehicle limits because these are present as soft constraints in the comfort objective $\boldsymbol{\theta}^T \boldsymbol{F}(\boldsymbol{X}, \boldsymbol{U}, T)$.

$$\boldsymbol{H} = \begin{Bmatrix} y^N = Width\ Lane \\ vy^N = 0 \\ \psi^N = 0 \\ \dot{\psi}^N = 0 \\ \delta^N = 0 \end{Bmatrix} \tag{4.11}$$

The constraints displayed in $\boldsymbol{H}$ make sure that at the end of the lane change the slipangles in the tires and the steerwheelangle are zero. From 4.3 this induces that the lateral velocity and acceleration and yaw acceleration also becomes zero and the lane change is completed. y^N makes sure that the wanted lateral distance is covered. This distance can be calculated from the start position of the vehicle in its lane and the width of the lane in order to end up at the centreline of the desired lane.
At the start of the lane change straight driving at constant longitudinal speed is assumed. To obtain this, the constraints of 4.12 are used. No constraints for accelerations are needed because this would give a redundancy due to the other initial states in combination with the motion equations 4.3. In 4.12 all initial states are zero excepts for 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.

$$\boldsymbol{X}_{initial} = \begin{bmatrix} x_{start} \\ y_{start} \\ v_{x,start} \\ v_{y,start} \\ \psi_{start} \\ \dot{\psi}_{start} \\ t_{r,start} \\ \delta_{start} \end{bmatrix} \tag{4.12}$$

The time limit constraint in 4.8 is needed in order to demarcate the optimization solution space. When set it has to take two conflicting criteria. It has to be chosen large enough in order to have a minor influence introduced by this constraint. Secondly it has to be taken small enough in order to preserve good conditions for the numerical integration performed in $\boldsymbol{F}(\boldsymbol{X}, \boldsymbol{U}, T)$. This is because the number of optimization points $N+1$ is fixed which means that a larger time limit will give a coarser time discretization. In order to serve these conditions time limit is set in this thesis on $25s$. This choice is validated in section 4.2.4. It is worth noting that with the removal of the time limit constraint the optimized comfortable lane change takes around 160 . This is not an realistic results because the objective will, as previously explained, not have good numerical properties.

In order to solve 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 4.6 this would give a division by zero.
To further enhance the solving speed of 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. Therefore the initial guesses for $\boldsymbol{X}, \boldsymbol{U}\ and\ T$ are taken from the observed ideal data.

Another initial guess that speeds up the solving time of the IPOPT solver, is setting the initial guess of the lambda multipliers internally equal to the ones found during the previous call of 4.8 during the loop visualized by figure 4.2.

The time needed by the CPU in order to calculate the expected path for a certain set of weights and thus solving 4.8 takes around 6.5 s when an timelimit of 30 seconds and N equal to 1000 is chosen.

4.2.2 Objective function

The objective function used in 4.8 has to contain 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 weights serve as parameters that steer the learning process towards a better match with the observations. It can be expected that the linear relation of features given by $\boldsymbol{F}(\boldsymbol{r})$ will serve as an approximation for the real, more complex comfort objective of a human drive. As is showed in [11] this linear approximation of features can already capture the main trends that contribute to an comfortable maneuver experience. As is suggested in [11] the feature framework that is further discussed in this section, can be validated and adjusted based on an user study in order to better match real driver observations.

When ideal data is used the observations are generated based on a known underlying comfort objective with known weights and feature framework. This permits the validation of the objective learning method discussed in this thesis. The remaining part of this section will discus the feature framework used in this thesis.

$$discomfort = \theta_1 \cdot f_1 + \theta_2 \cdot f_2 + \theta_3 \cdot f_3 + \theta_4 \cdot f_4 + \theta_5 \cdot f_5 + \theta_6 \cdot f_6 \tag{4.13}$$

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

Feature 1: longitudinal acceleration

$$f_1 : \boldsymbol{r} \rightarrow f_1(\boldsymbol{r}) = \int_0^T a_{x,total}^2(t)dt \tag{4.14}$$

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

Feature 2: lateral acceleration

$$f_2 : \boldsymbol{r} \rightarrow f_2(\boldsymbol{r}) = \int_0^T a_{y,total}^2(t)dt \tag{4.15}$$

Feature two is assessing the amount of discomfort by integrating the total lateral acceleration in the local axis. The total lateral acceleration $a_{y,total}$ is the sum of $a_{x,tangential}$ and $a_{x,normal}$ as described in 4.3.

Feature 3: longitudinal jerk

$$f_3 : \boldsymbol{r} \rightarrow f_3(\boldsymbol{r}) = \int_0^T j_x^2(t)dt \tag{4.16}$$

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

Feature 4: lateral jerk

$$f_4 : \boldsymbol{r} \rightarrow f_4(\boldsymbol{r}) = \int_0^T j_y^2(t)dt \tag{4.17}$$

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

Feature 5: desired speed

$$f_5 : \boldsymbol{r} \rightarrow f_5(\boldsymbol{r}) = \int_0^T (v_{des} - v_x)^2 dt \tag{4.18}$$

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

$$f_6 : \boldsymbol{r} \rightarrow f_6(\boldsymbol{r}) = \int_0^T (y_{lane\_change} - y)^2 dt \tag{4.19}$$

y_{des} is a constant and is the desired lateral distance completed after the lane change. If the vehicle reaches faster its desired lateral displacement this is perceived as a good responds and is interpreted as comfort of the driver as is discussed in section 3.1.

In order to implement the above defined integrals in the objective function of 4.8 discretization is needed. For this the Crank-Nicolson numerical integration is used as is shown in 4.20 for feature f_i.

$$\int_{f_i(t^n)}^{f_i(t^{n+1})} df_i = \int_{t^n}^{t^{n+1}} I(t) \cdot dt \tag{4.20a}$$

$$f_i^{n+1} - f_i^n = \frac{1}{2} \frac{I(t^{n+1}) + I(t^n)}{\Delta t} \tag{4.20b}$$

The objectives described by equation 4.13, that consists out of a set of comfort features that model the amount of discomfort experienced during a maneuver by mapping kinematic signals onto scalar feature values by the means of integration. By finding the driver specific weights $\boldsymbol{\theta}$ in 4.13 it is possible to model driver preferences between different comfort features. With this information an autonomous vehicle can perform path planning of the most comfortable path to do a maneuver for a specific driver. As is shortly discussed in Chapter 3, the perception of save driving of an autonomous vehicle contributes to the amount of comfort that is experienced during driving. Because this thesis is focussing on the development and validation of a reinforcement learning algorithm, the environment is not taken into account. However this can be done if data of the state of other vehicles is available by modelling the distances between different vehicles by equation 4.21 as suggested in [11].

$$f_d = \sum_{k=1}^{L} \int_0^T \frac{1}{(x_{o,k}(t) - x)^2 + (y_{o,k}(t) - y)^2} \cdot dt \tag{4.21}$$

$$L \in \mathbb{N}$$

With $[x_o, y_o]_k$ the position of a different vehicle and L the total amount of vehicles in the nearby area.

Not only the bird's eye view distance between two different vehicles plays a role, but also the following distance of the vehicle in front in the same lane is important. This can be modelled as suggested by [11] by:

$$f_d = \int_0^T max(0, \hat{d} - d(t)) \cdot dt \tag{4.22}$$

In equation 4.22, $\hat{d}$ is the desired distance and $d(t)$ is the distance between the vehicle in front in the current lane.

An other assumption that not is discussed in this thesis is the time limit to finish a maneuver. In a real life application however this limit often influence the maneuver. After the weights are identified, the most comfortable path in a limited time span can be calculated for a specific driver. If no time limit serves as bottleneck, the time of the maneuver can be included as optimization variable and long and smooth lane changes are planned.

4.2.3 Normalization factors and initialization

In order to nullify the effect of order of size given by the units in the objective discussed in section 4.2.2, a normalization of the features is used. 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. Then the objective function is divided by the corresponding normalization factor which means that a feature that inherently gives a small feature values, will be divided by a small value and the other way around, an inherently large feature value will be divided by a large value. In this thesis the example lane change is produced as is described in section 4.2.4 with weights equal to $\begin{bmatrix} 4, 5, 6, 1, 2 \end{bmatrix}$ and corresponding to the objective function of section 4.2.2 and making use of normalization factors and an initial guess provided by data of a lane change that was available at Siemens. Table 4.2.3 gives an overview of the lane change normalization factors used in this chapter.

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

Table 4.2: Overview of normalization factors.

The solver used to calculate the states and control signals in 4.8 is the interior point solver IPOPT which is open source code. The idea behind the solver is to smoothing the KKT conditions and transform it to a smooth root finding problem. [14] Because IPOPT is a interior boundary method, a feasible initial guess is needed.
Every time that the expected path has to be calculated as is visualised in flow diagram 4.2, the optimization 4.8 is performed with as initial guess the observed maneuver.

4.2.4 Ideal data

Generation

Schrijf een deeltje over invloed N en T –> maak een tabel met de testen van 5 mei.
As mentioned above the term 'ideal data' concerns data that is generated with a non-linear bicycle model and exactly fulfils the assumption that the observations are

minimizing an underlying discomfort function. More over the discomfort function is the same as the one used in the learning algorithm and this means that the generated path is a the solution of the objective function discussed in section 4.2.2 for weights $\boldsymbol{\theta}$ chosen in advance. Because that the observations are exactly representing the comfort function in the learning algorithm and the absence of vehicle model mismatch, weights can be learned that perfectly explain the observations. This is translated in a accurate matching of the feature values discussed in 4.2.2. Because beforehand it is know what the weights should be in 4.13, the ideal data can serve as a validation of the developed learning algorithm.

The relative weights chosen are $\begin{bmatrix} 4, & 5, & 6, & 1, & 2 \end{bmatrix}$, which gives as absolute weights $\begin{bmatrix} 549.63, & 1.90, & 0.53, & 21.43, & 0.12 \end{bmatrix}$ when the normalization of 4.2.3 is taken into account. The objective that is used in 4.8 and outputs the ideal data is given by 4.23.

$$discomfort = \frac{4}{0.0073} \cdot f_1 + \frac{5}{2.64} \cdot f_2 + \frac{6}{11.28} \cdot f_3 + \frac{1}{0.047} \cdot f_4 + \frac{2}{17.14} \cdot f_5 \quad (4.23)$$
$$f_i \in \mathbb{R}, \; i \in \mathbb{N}$$

Because of the normalization the relative weights will quantify the trade-offs between different comfort features without the disturbance of units used. Aside of this it allows to learn weights faster because of the absence of big size differences that are present in absolute weights. When multiple ideal data sets have to be generated in order to serve as observations, the initial speed and width of the lateral distance
In order to determine if the generated data is a local solution of the optimization of 4.8, three different initial guesses are used. From which two were coming from example lane changes produced by the 15 dof Amesim vehicle model of Siemens and the other one was a generated path with all weights equal to one and using Siemens dataset 1 as initial guess. Figure 4.3 shows the three initial guesses used. It could be concluded that the three guesses gave the same generated signals for all states and controls of the non-linear bicycle model. Figure 4.4 shows the three generated paths which lay exactly on each other. Further on in this chapter the initial guess is set equal to the observed data.

Validation

4.3 Ideal lane change data learning results

This section discusses the results from using the learning algorithm on generated ideal data which is generated making use of the above discussed normalization values in combination with relative chosen weights of $\begin{bmatrix} 4, & 5, & 6, & 1, & 2 \end{bmatrix}$. The first simulation discusses the learning of three datasets that are generated with the same chosen weights. The different datasets are A (V_0 : 22.22 $\frac{m}{s}$, L : 3.47 m), dataset B (V_0 : 22.22 $\frac{m}{s}$, L : 6.94 m) and dataset C (V_0 : 25.00 $\frac{m}{s}$, L : 3.47 m). The convergence

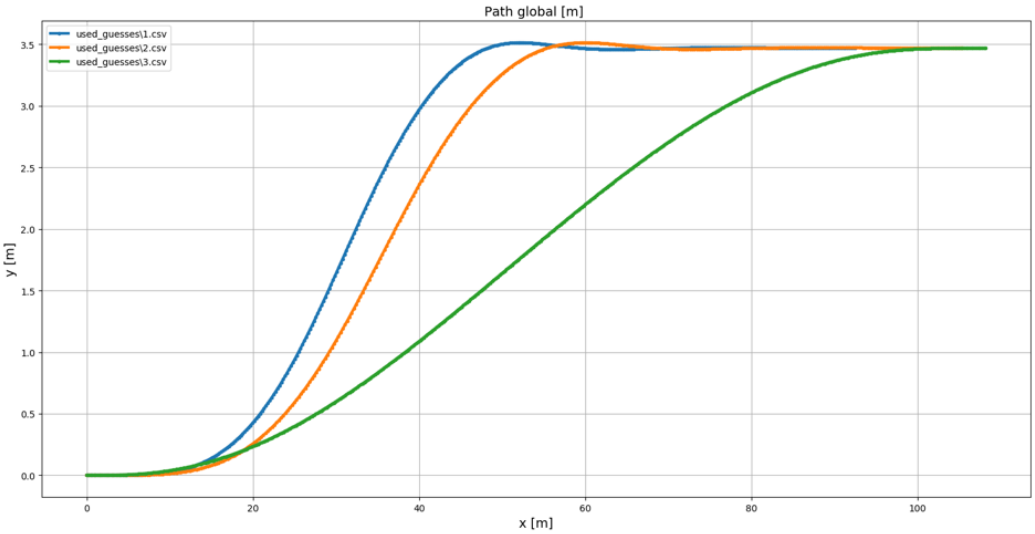

FIGURE 4.3: Different initial guesses used in 4.8 to generate data.

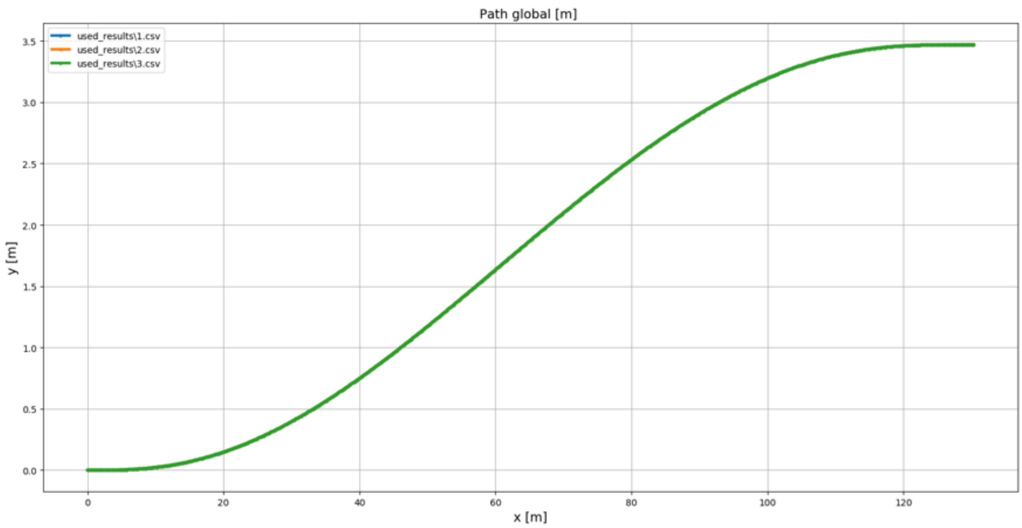

FIGURE 4.4: Different ideal data paths generated with 4.8.

criteria to stop the learning algorithm displayed in Figure 4.2 is when the maximum amount of iterations equal to 300 is reached or if the feature values which use the learned weight match accurately the ones of the observed data according to following formula $f_{rel,i} = \frac{f_{calc,i}}{f_{obs,i}} \leq 10^{-4}$.

4.3.1 Sequential learning

In this section dataset A is learned followed by dataset B and as last dataset C. As initial guess of the weights $\begin{bmatrix} 1.0, 1.0, 1.0, 1.0, 1.0 \end{bmatrix}$ is used. The resulting paths can be seen in Figure 4.5 and Figure 4.6 gives the convergence of $f_{rel,i}$ towards one.

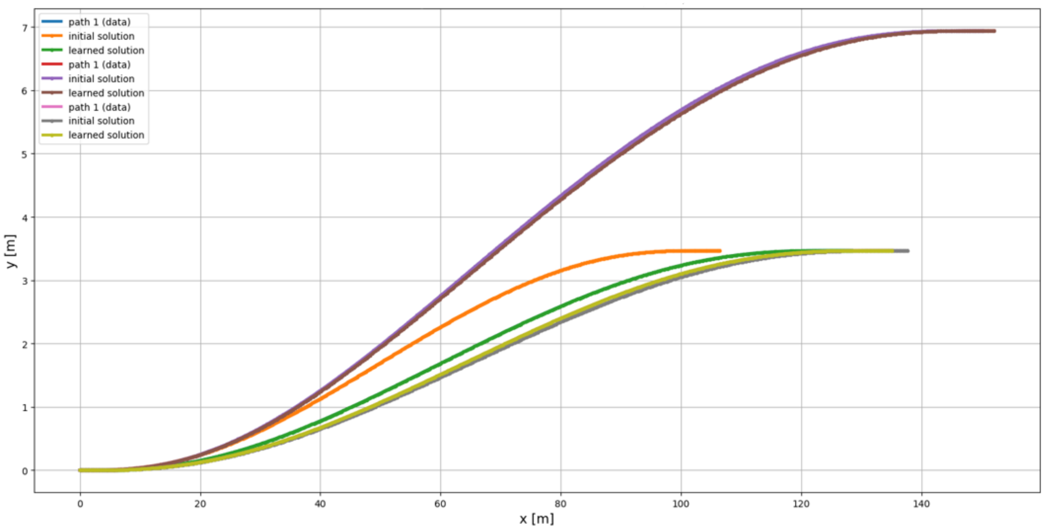

FIGURE 4.5: Overview of initial guesses, learned and observed trajectories.

Convergence is attained after 50, 51 and 49 iterations. From Figure 4.5 it is clear that when features accurately match as visualized in Figure 4.6, the observed path is accurately reconstructed when using ideal observations. It can be concluded that fitting of the features gives a good reproduction of the individual state and control signals. Furthermore convergence is fast obtained because the amount of iterations stays limited and simulation time is under 5 minutes per dataset.

An interesting result however is that the corresponding features are not found back. The learned weights are respectively $\begin{bmatrix} 0.45, 1.44, 1.68, 0.39, 0.60 \end{bmatrix}$, $\begin{bmatrix} 0.45, 1.48, 1.71, 0.35, 0.59 \end{bmatrix}$ and $\begin{bmatrix} 0.63, 1.43, 1.69, 0.37, 0.60 \end{bmatrix}$. This is visualized in Figure 4.6 by the small jump when switched to a different dataset. Figure 4.5 shows in orange the first initial guess and in green the learned solution which matches the observed one in blue. The other generated paths start from a close but not perfect initial guess.

For each different dataset features match exactly but this is not done for the exact same weights because local solutions exist that explain individual datasets. It should be noted that the relative sizes of the learned weights matches the original one except for the feature concerning longitudinal acceleration which don't play a dominant role during a lane change. **Question: how is this when you do an acceleration maneuver??**

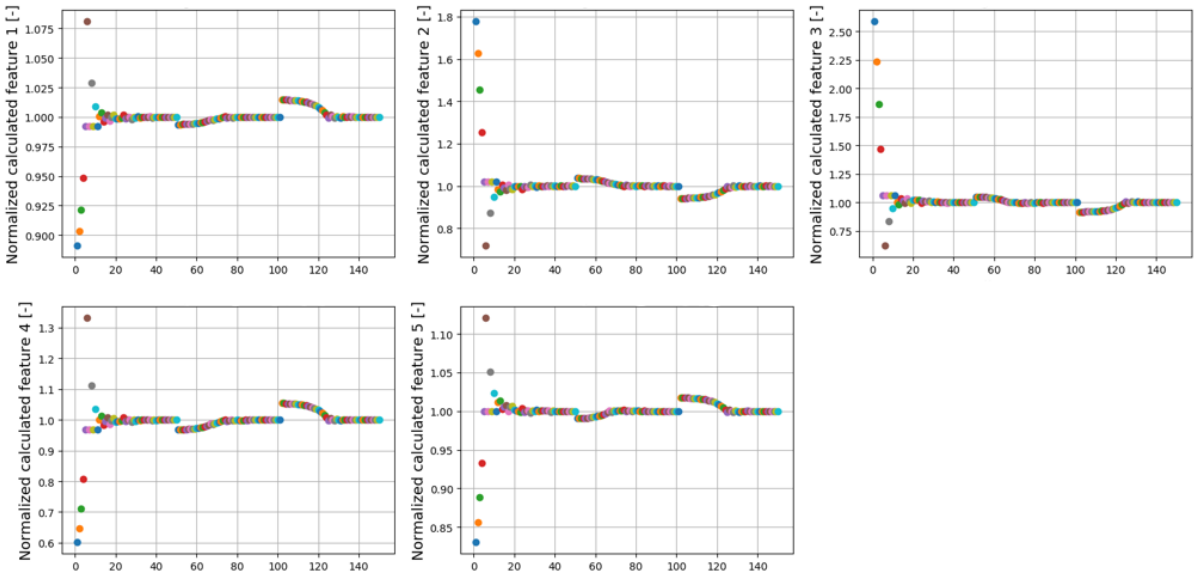

FIGURE 4.6: Convergence of the features with learned weights towards the observed features according to section 4.2.2

Next the influence of the numerical discretization is checked. A better discretization is obtained when a larger amount of optimization points N is chosen which lead to a smaller time discretization dt, but comes at the cost of a higher computational load. Figure 4.7 shows the error on the features discussed in section 4.2.2 when using the learned weights based on dataset A for predicting the observations of dataset B. The convergence criteria of the learned weights are set on $f_{rel,i}$ equal to 10^{-1}. It is concluded that there is no large influence on the amount of optimization points chosen.

In order to further validate the learning algorithm, the chosen weights were given as initial guess making use of one dataset in order to check if it would diverge from the ideal global solution. As expected the algorithm converged after the first iteration. Next a guess close to the global solution was tried. The algorithm converged after 41 iterations and outputted $\begin{bmatrix} 4.01, 5.29, 6.33, 1.09, 2.12 \end{bmatrix}$ and not $\begin{bmatrix} 4.0, 5.0, 6.0, 1.0, 2.0 \end{bmatrix}$ which proofs that a lot of local solutions exist.

From above tests it can be concluded that the local solutions of individual datasets do not match and that the amount of mismatch has only a small dependants on the amount of optimization points. The next step to take is to combine different datasets in simultaneous learning instead of sequential learning in order to try to converge to weights that predict best for multiple datasets.

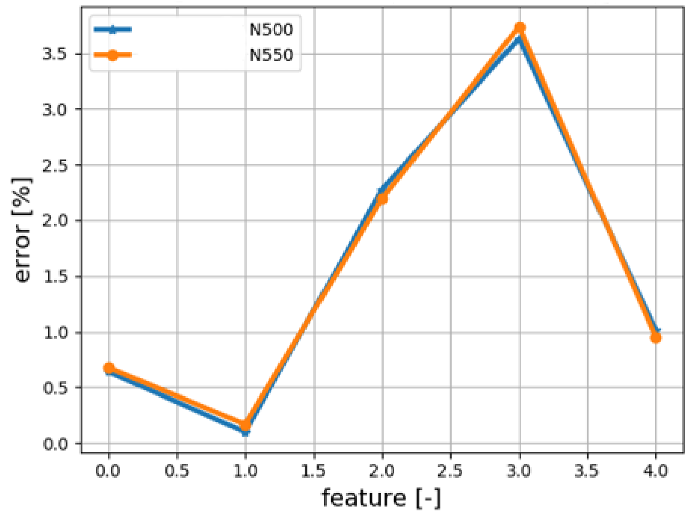

Figure 4.7: Different error made when using a different amount of optimization points in 4.8.

4.3.2 Conflict method

The first method proposed is the conflict method. Is main idea is to update the weights in the common direction of the gradient seen in equation 3.7 of the different datasets. If the direction of update according to gradient method is conflicting between different datasets the update of the weight in question is put on hold. The updating will be resumed when the conflict is resolved by updating the other weights. This is possible because the features are not independent of each other. Figure 4.8 shows how the conflict method will be embedded in the basic flow of the learning algorithm.

Depending if the previous case in the RPROP algorithm was case 2 and the sign difference between the current and previous gradient, three distinct RPROP cases can be identified with other update methods as discussed in section 3.2.2. Inside the conflict test, it is checked if all the signs of the current gradients of the different datasets are checked. If there is a sign difference, this means that for one dataset a feature value is higher than the observed one and the corresponding weight should be increased (negative gradient) and for an other dataset it is the other way around and the corresponding weight 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 also be unity in the decision of the RPROP case. After the conflict is solved, the next case will be automatically equal to case 3 because no decision can be made about the increase or decrease of the update value. The convergence criteria used is aside of the maximum amount of iterations and an accurate match of 10^{-4} with the observed features the criteria if there is still improvement possible in a direction that

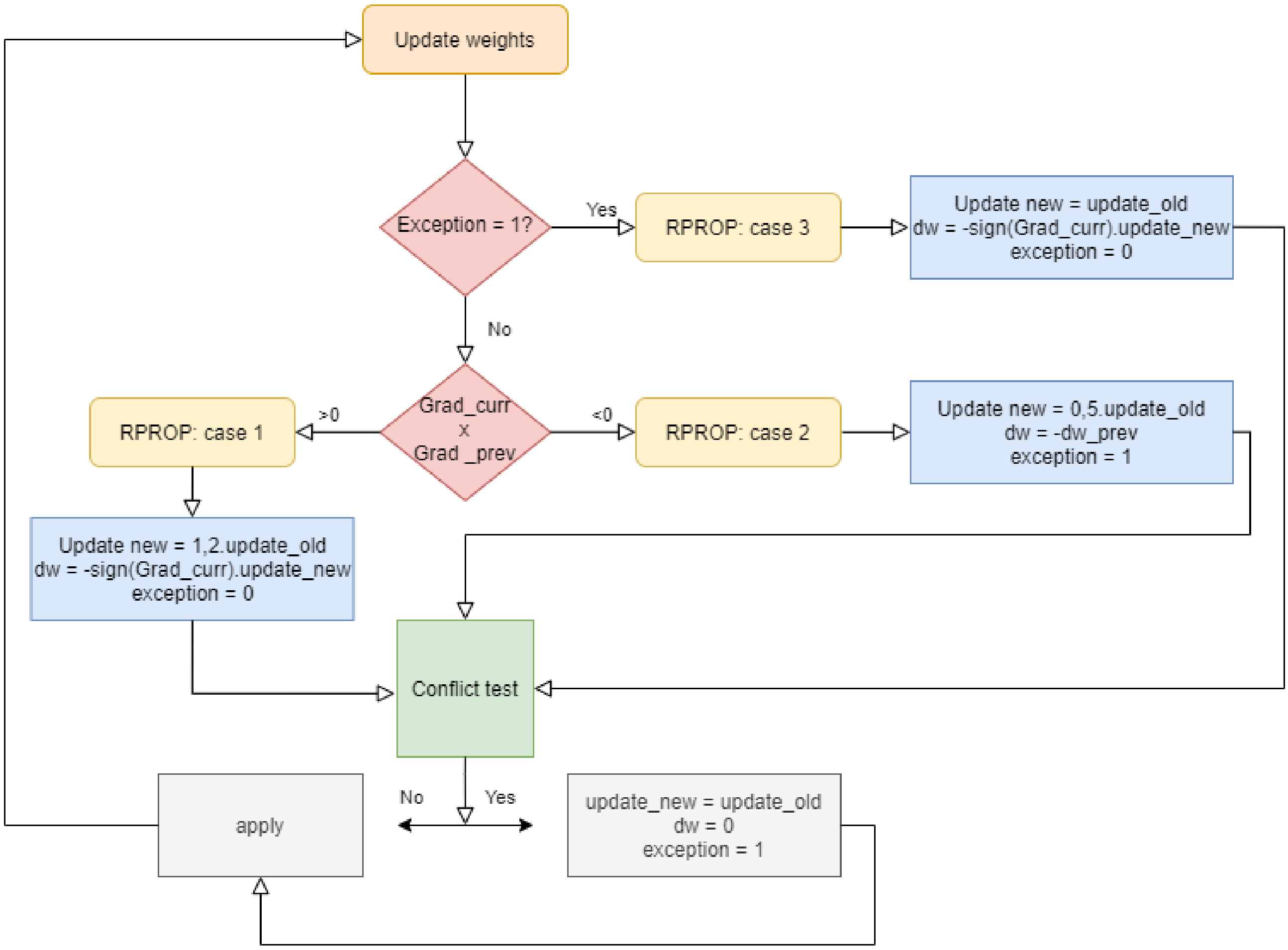

FIGURE 4.8: Flow of the conflict method as part of the basic flow diagram of Figure 4.2

benefits every dataset.

The learning algorithm is applied simultaneously to dataset A,B and C and uses as initial guess for the weights the vector $\begin{bmatrix} 1.0, 1.0, 1.0, 1.0, 1.0 \end{bmatrix}$. The algorithm is ended after 17 iterations with no direction of improvement possible. The observed paths with their initial guesses and learned paths can be seen in Figure 4.9.

The convergence plots are all similar and for dataset A, Figure 4.10, gives the convergence of $f_{rel,i}$ for the different features.

As is indicated by Figure 4.10, the method is clearly converging towards the different sets of observed feature vectors. However it should be noted that despite giving more accurate results for the feature values of the other datasets then when only one dataset A is used to learn the weights, the method is still constricted in its feature matching accuracy due to no available direction of improvement. This effect worsens

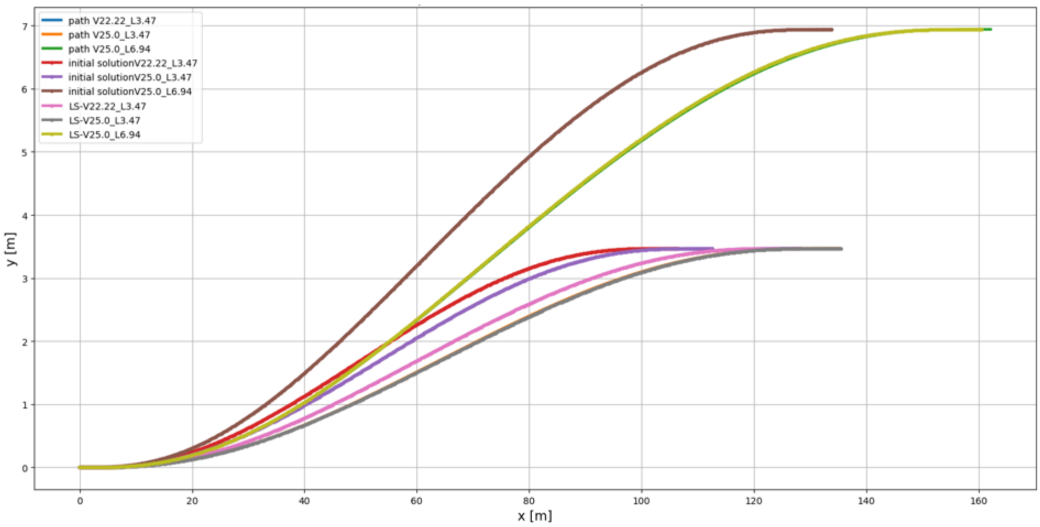

Figure 4.9: Overview of the different observed paths, initial guesses and learned paths.

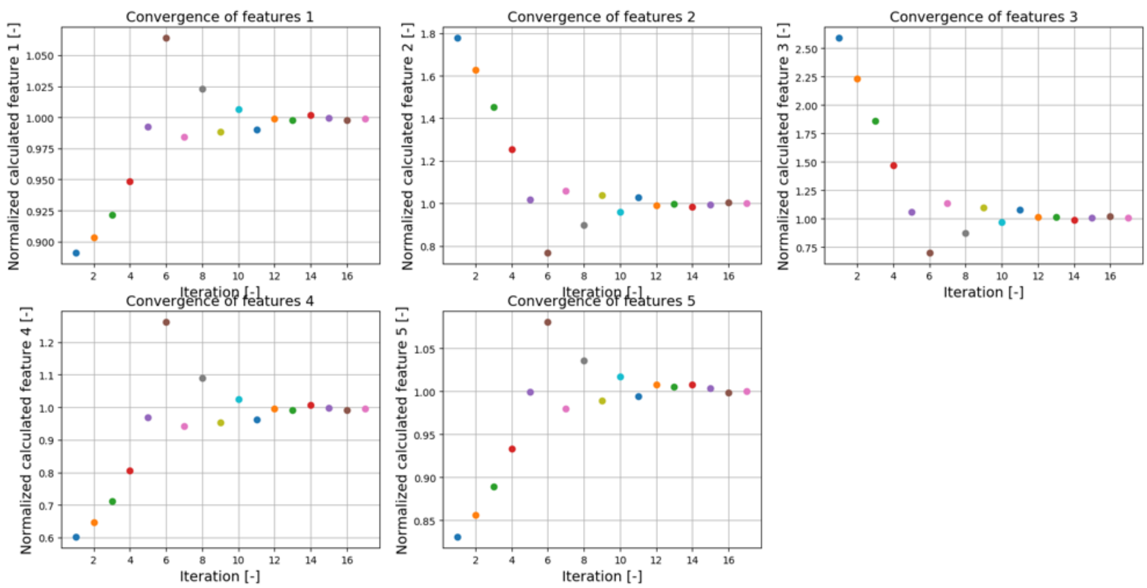

Figure 4.10: Convergence plot of $f_{rel,i}$ for dataset A.

when more datasets are used because conflicts more rapidly occur. Table **??** lists the different learned weights and Figure 4.11 shows the loss of accuracy when more datasets are simultaneously learned.

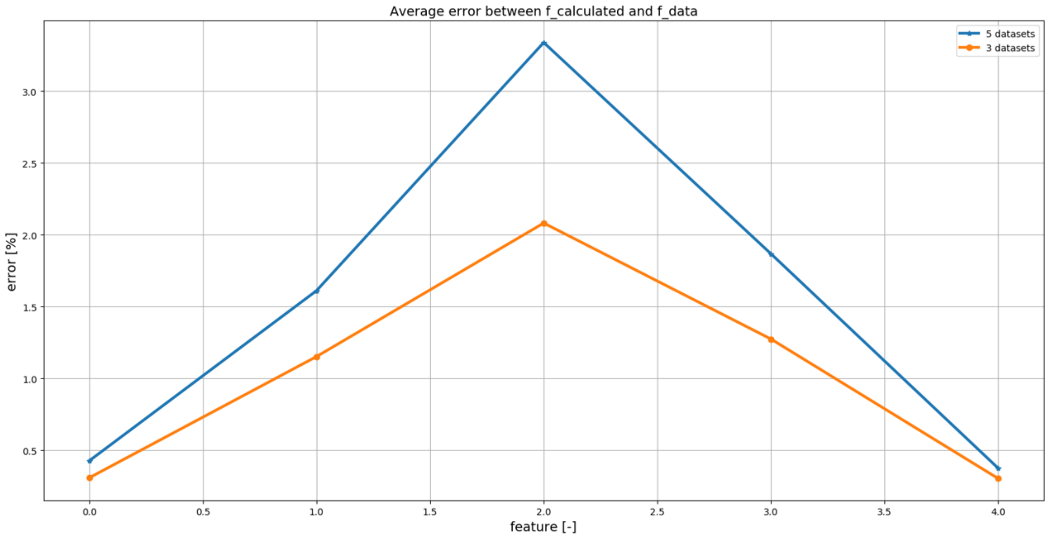

FIGURE 4.11: The average error between the observed and calculated feature values with the learned weights when applying the conflict method.

4.3.3 Averaging method

In this method not all the different observed feature vectors are taken individually into account when updating the weights, but a single averaged observed feature vector is used. This avoids conflicting local solution. The flow of the algorithm is thus similar as shown in Figure 4.2 but instead of a single dataset an averaged is used. In order to calculate the gradient, the difference between the averaged observed feature vector and the averaged calculated feature vector has to be taken. In order to obtain the averaged calculated feature vector, m times the optimization 4.8 is called for each specific observed maneuver. Afterwards the individual found feature vectors are averaged. With m the amount of observed datasets. This method is based on [11].

The learning of the three datasets A,B and C is done with as stop criteria the maximum amount of iterations and convergence towards the averaged feature values with an accuracy of 10^{-4}. The initial guess chosen for the weights is an all one vector. Convergence is reached after 111 iterations because the desired convergence accuracy is reached. The learned weights are $\begin{bmatrix} 1.21, 1.53, 1.83, 0.31, 0.61 \end{bmatrix}$.

4.3.4 Comparison of methods

In this section a comparison is made between the average and conflict method of combining different datasets.

,

4.3.5 Conclusion

Weight	Average	Conflict
Nr.2	0.0	0.0
Nr.4	0.0074	0.032
Nr.6	0.0029	0.00045

Table 4.3: The error made between the learned and chosen weights for respectively the average and conflict method.

Chapter 5

Enhanced learning approach

Next a new learning approach is proposed which is based on a fitting algorithm and makes use of taylor expansions.

Chapter 6

Learning from complex vehicle model

6.1 The First Topic of this Chapter

6.1.1 Item 1

Sub-item 1

Sub-item 2

6.1.2 Item 2

6.2 The Second Topic

6.3 Conclusion

Chapter 7

Validation with expert driver data

Bespreek de validatie van de methode. Implementeer similaties in prescan. Bespreek de verschillende software tools bij Siemens –> Amesim, simulink, prescan. Hoe werken ze samen en hoe wordt de validatie precies gedaan? Wat zijn de resultaten? Install amesim and write a chapter about how the dataset is generated. How is the amesim model defined etc.
how accurate results are learned when the assumption on the observations is violated. (assumping generated by a comfort cost function)
The objective function of the MPC is slightly different of the one used in the learning algorithm. In the training set it was important to explain the observed data but it is known beforehand that the driver demonstrated a feasible path and wants to do a lane change. Now the situation is different and features that assess the environment are a necessity. For this reason the objective function of the learning algorithm is extended with features that handle collision avoidance and following distance. (Lane change behavior is influenced by the environment!)

7.1 The First Topic of this Chapter

7.1.1 Item 1

Sub-item 1

Sub-item 2

7.1.2 Item 2

7.2 The Second Topic

7.3 Conclusion

Chapter 8

Conclusion

The final chapter contains the overall conclusion. It also contains suggestions for future work and industrial applications.
Application: say something about hierarchical control. The comfort controller can work as an inferior controller of the safety of the vehicle.

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

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

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

Appendix B

The Last Appendix

Appendices are numbered with letters, but the sections and subsections use arabic numerals, as can be seen below.

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] 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.

[3] H. Bellem. *Comfort in Automated Driving : Analysis of Driving Style Preference in Automated Driving.* PhD thesis, 2018.

[4] 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.

[5] L. Daniel. Method and system for determining and dynamically updating a route and driving style for passenger comfort - US Patent, 2018.

[6] E. David. Will autonomous vehicles be safe to use? | Cybersecurity & Technology News | Secure Futures | Kaspersky.

[7] 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.

[8] 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.

[9] J. Gillis. Ya Coda course presentation, 2019.

[10] 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.

[11] 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.

[12] T. Mercy. *Spline-Based Motion Planning for Autonomous Mechatronic Systems*. PhD thesis, 2018.

[13] P. Patrinos. Model Predictive Control - Lecture Notes, 2019.

[14] P. Patrinos. Optimization - Lecture notes, 2019.

[15] V. Powers, C., Mellinger, D., Kushleyev, A., Kothmann, B., Kumar. Experimental Robotics. *ISER, June*, 88(287513):515–529, 2013.

[16] Prof. Amnon Shashua. Experience Counts, Particularly in Safety-Critical Areas | Intel Newsroom, mar 2018.

[17] 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.

[18] Q. N. Tong Duy Son. Safety-Critical Control for Non-affine Nonlinear Systems with Application on Autonomous Vehicle. (August):7, 2019.

[19] 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.

[20] University of Warwick. Do passengers prefer autonomous vehicles driven like machines or like humans?, 2019.

[21] K. Yankov. *Potential Field Based Model Predictive Control for Autonomous Vehicle Motion Planning and Control*. PhD thesis.

[22] 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.