CHALMERS

Improved driver model for testing in HIL environment

Master's thesis in Complex Adaptive Systems

HARALD FREIJ

Department of Applied Mechanics
Division of Vehicle Engineering & Autonomous Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2013

Master's thesis 2013:76

MASTER'S THESIS IN COMPLEX ADAPTIVE SYSTEMS

Improved driver model for testing in HIL environment

HARALD FREIJ

Department of Applied Mechanics
Division of Vehicle Engineering & Autonomous Systems
CHALMERS UNIVERSITY OF TECHNOLOGY

Gothenburg, Sweden 2013

Improved driver model for testing in HIL environment
HARALD FREIJ

Master's thesis 2013:76
ISSN 1652-8557
Department of Applied Mechanics
Division of Vehicle Engineering & Autonomous Systems
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone: +46 (0)31-772 1000

Cover:

Chalmers Reproservice
Gothenburg, Sweden 2013

Improved driver model for testing in HIL environment
Master's thesis in Complex Adaptive Systems
HARALD FREIJ
Department of Applied Mechanics
Division of Vehicle Engineering & Autonomous Systems
Chalmers University of Technology

Abstract

This master's thesis concerns driver models, with focus on non perfect driving. On the base of a black box driver behaviour model, perturbations are added for modelling lane drifting, non-alert driving (for example caused by sleepiness) and collision from behind. The driver models are developed and implemented for usage in an HIL environment. This means that both high and low level behaviour is considered.

The running off road is modelled as a small steering wheel angle offset introduced with a smooth ramp. This has support in data from extensive field operation tests. The non alert driving is modelled both as driving with random input update frequency, and a sinusoidal steering wheel angle offset. These models are qualitatively compared to data corresponding to sleepy driving, recognised by a Driver Alert Control safety function. The behaviour connected to collisions from behind is modelled using Gipps' longitudinal model, with small presumed decelerations.

The thesis also presents different low level model approaches, that is how a target vehicle motion is realised, in terms of pedal positions and steering wheel angle. Both PID and open loop methods are discussed. It is shown that PID controlling can give a very good resemblance between target and obtained motion, but also that a high correlation can be acquired without a good resemblance on the low level input.

Keywords: Driver model, HIL, Testing environments, Run off road, Non alert driving, Collision from behind, Field operation test

Preface

The work with the master thesis has been conducted at the Electrical Integration Environments group at Volvo Car Corporation (VCC) in Torslanda, as a part of the Volvo Cars Engineering Student Concept (VESC). Volvo Car Cooperation is a Swedish car manufacturer owned by Zhejiang Geely Holding Group. Most of the development is done in Torslanda.

Acknowledgements

I would like to deeply thank my supervisor at Volvo Cars Corporation, Martin Nilsson, for giving me the opportunity to write this thesis at VCC, and introducing me to the HIL-environment, the company and the automotive industry. I am also very thankful to Krister Wolff for the genuine interest and concern about my report, and for giving the work the academic touch.

A big thanks goes to my colleagues at the Electrical integration environments group; for interesting chats at the coffee table, for lunch walks in sun and rain, for answering all my questions and for asking the right ones yourselves.

I would also like to thank all the people surrounding me during my studies at Chalmers. Friends in FnollK, F-spexet, Chalmersspexet and Raspredax, thank you for all laughs and amazing times, making my time at Chalmers go faster and my life longer. Without you, writing this thesis would have been a hopeless journey. A special thanks to Emma, for always being there and for sharing my time, laughs and problems.

Contents

1 Introduction

The Electrical Integration Environments group at Volvo Cars works with providing testing environments for the electrical systems, so called HIL-rigs. HIL is an abbreviation for Hardware In the Loop, and means that hardware components are tested together with computer models of the surrounding systems. The rigs are used for system and integration testing on a complete electrical system level, and are parts of the system development process from the first prototype cars to the final product. The tests are conducted on both system models and physical components, and can be conducted both automated and manually. The models are mainly built in Simulink, but certain modules are written in C or embedded MATLAB code. Models include for example plant models of car dynamics, models of network communication, models of sensors unavailable in a lab environment.

Parts of the system, such as active safety and navigation, requires a driving environment with roads, crossings and other cars. Simulating realistic driving in such an environment requires a rather complicated driver model. Such a model is today provided together with the environment simulation software, called VIRES.

1.1 Objective

The thesis aims at improving the current driver model so that it can be used to test functions requiring a non-perfect driver, such as lane change warning, drowsiness detection and other functions. The goal was to implement, in Simulink, a modified driver model for the purpose stated above. The driver should preferably be predictable in the sense that running the same simulation repeatedly should give the same outcome. The model should be possible to control both automatically (e.g. for night-tests) and manually. The model should include behaviours for three different kinds of bad driving: drifting off road, non alert driving and collision with another car from behind. The model was to be implemented as an add on to the driver model provided with VIRES. The model should also provide the connection between this driver model and the vehicle dynamics model, and the possible disturbance introduced in this connection should be estimated.

1.2 Limitations

The work has included modelling the desired behaviours, integrating them in the large system model and creating a user interface for selecting driving behaviour. The work was focused on small scale behaviours, and not at for example navigation and route planning. The focus has also been on preparation for testing on an integration level and not on a function level, meaning that variations of realistic behaviours have not always been implemented, but instead models known or supposed to trig the relevant functions.

1.3 Contributions

This thesis contribute to the field by describing a full scale implementation of models of non perfect driving in an HIL environment, and discussing the problems connected to the separation of intended and achieved driving behaviour. It also discusses how to use data from field operation tests to optimise and validate driver models.

2 Background and Theory

2.1 Classes of driver models

Much work has been done on modelling and simulating driver behavior in different kinds of scenarios. A deep and thorough review of near-collision models has been done by Markkula et al. (2012). The authors separate models concerning longitudinal movements from those focusing on lateral dito, as most models do not handle both. That is, either the model deals with accelerations or speeds, or with the angle of the steering wheel. Another distinction can be done between behaviour and control models. Many models are developed in order to study traffic situations, and do only focus on the external dynamics of the car. The input of these models is the current traffic situation (distances to other cars, geometry of the road ahead, etc.), and their output is the dynamics of the car, such as velocity and direction at the next time step. These are what in this thesis is called behaviour models. Other models focus on the actual interface between the driver and the car, such as pedal positions and steering wheel angle. In many situations this is not important, but in other it is of perhaps greater importance than the behaviour. In this thesis, such models are called low level models.

2.1.1 Longitudinal behaviour models

In the literature, there is a wide range of longitudinal models proposed. Some of them are developed from an engineering point of view, and are rather mathematical in their description. Such models may be a bit weak in their motivation. Others are more perceptual, and try to describe why the driver acts in a certain way. These are rather attractive as they, more often, have a well motivated background, and their parameters often can be translated to entities one can relate to. One drawback is however that they can be quite complex (as is the human brain).

A relatively simple, yet partly cognitively motivated, longitudinal model is the one proposed by Gipps (1981). The idea is that the driver, besides having a maximal velocity, acceleration and deceleration, always tries to maintain a situation where he or she, if the driver ahead applies his maximal braking force, has the time to turn to a halt before colliding. With a being the maximal acceleration, b the maximal deceleration, V the maximal velocity, τ the reaction time, $\hat{b}$ the expected maximal deceleration of the car ahead and s the minimal distance the driver wants to have between the own car and the one ahead, this results in the following equation (Gipps 1981):

$$v(t+\tau) = \min\{\left(v(t) + 2.5a\tau(1-\frac{v}{V})\sqrt{0.025+\frac{v}{V}}\right); \left(b\tau + \sqrt{(b\tau)^2 - b(2(\Delta x - s) - v\tau - \frac{v_{Front}^2}{\hat{b}}}\right)\} \quad (2.1.1)$$

The first term is descriptive and seem to be chosen for its nice behavior and simple calculations. It describes a smooth acceleration up to the goal velocity V.The second term deals with the collision avoiding behavior and is physically and cognitively motivated. As long as the car ahead does not brake faster than $\hat{b}$, the driver will maintain a safe distance to the car ahead, with margins for reaction.

It should be noted that this is a discrete model with large time steps, the reaction time τ, which is also used as iteration time, is proposed to be set to 2/3 s.

2.1.2 Lateral behaviour models

As for the longitudinal case, a range of different lateral models are at hand, all with different approaches.

A very simple lateral model is described by Reński (2001). The model is based on pathfollowing, along a path $y_d(x)$, and the steering angle is at every point proportional to the deviation angle to a preview point at a given distance L along the road. The output is then delayed with a time T, representing the reaction time for the driver. This means that the final steering angle is given by

$$\delta = W(\arctan\left(\frac{y_d(x(t-T)+L) - y(t-T)}{L}\right) - \Psi(t-T)). \quad (2.1.2)$$

To this result, he adds an assumption (without mentioning) of small angles and variations in velocity, which allows him to skip the inverse tangent and replace x(t-T) with $x - Tv$. This is however of smaller importance.

Reński also computes optimal parameters for the model, and base these on respons delay times of 0.4 and 0.6 s respectively. It should be noted that the model by Reński has been little used and discussed in the literature. However, it is attractive in its simplicity, and shows the basic idea with path following models.

A more widely used path following model is the one by Sharp, Casanova, and Symonds (2000). The model is based on a linear combination of a series of error terms, wich are to be compensated. The first error term e_Ψ is the angle between the current direction of the car and the direction of the path at it's closest point. The second error term e_1 is the current orthogonal distance to the path (with sign), and the following error terms e_i are the orthogonal distances to the path given that the car would continue in it's current direction for another distance l_i. The resulting steering angle is then given by

$$\delta = K_\Psi e_\Psi + \sum_{i=1}^{i=n} K_i e_i. \tag{2.1.3}$$

Benderius (2012) showed that this driver model can be used to describe a very wide range of dirver behaviours, using the right coefficients. The model has however a large drawback, in that is does not in a satisfying way describe why the driver behaves as he or she does. The parameters are also quite hard to see through, and it is very hard to see in which way manipulating them would change the shown behaviour of the driver. This makes it hard, not to say impossible, to use when one wants to test different behaviours.

Another interesting steering model is proposed by Wann and Wilkie (2004) . This is exciting, as it has a very perceptual, not to say biological approach. The idea is that the steering is driven by the optical entities *visual direction* and *retinal flow*. The retinal flow, or optical flow, is the movement of points in the visual field, while the visual direction is the direction in which the driver is looking. By fixating on a point further along the path, and letting the steering angle act as a damped spring with a linear combination of the visual direction and retinal flow as the driving force, they manage to acquire realistic driving behaviour. The coefficients of the linear combination seem to depend on what they call the quality of the flow. For example, racing drivers tend to focus more on the visual direction, as they are driving in high speed on broad roads without central markings, and thus get less flow information. However, the model by Wann and Wilkie doesn't seem to reproduce panic behaviours very well.

2.2 Low level models

As noted in the article by Markkula et al. (2012), most of the longitudinal models developed are pure behaviour models, and do not regard the pedals of the vehicle. This can be explained the fact that most of them are used to simulate traffic scenarios or collision avoidance, but not the internal whereabouts of the car. As the model developed within this thesis is to be used in a complete electronics environment, it has to give the pedal and steering wheel states as output. Any model used has thus to be supplemented with some transformation from either desired speed or acceleration to two pedal states, $(p_a, p_g) \in [0, 1]^2$. This kind of transformation can be seen as an inverse of the driver's internal vehicle model, which maps the pedal states to the car's dynamics. This model is not to be confused with the vehicle model used in the simulation, that does actual mapping from the driver-to-car interaction to the dynamics of the car.

2.3 Active safety systems

Three different active safety systems are of interest for this thesis. They are each connected to one of the three behaviours modelled. The first one is the Lane Departure Warning (LDW), which is supposed to warn a driver that is accidentally drifting from the current lane. The warning comes as a sound signal as well as a visual sign in the infotainment system. LDW is activated at speeds greater than 65 km/h, and is thus not active in city traffic but during country road and highway driving. The system uses a front camera to monitor the car's position within the lane. In order for LDW to work, the road has to be properly marked with lane marks.

The second safety system is the Driver Alert Control (DAC), which should warn a driver who shows tendencies of drowsiness or loosing focus. This system is also based on the lane offset, but also other factors. The system monitors the driving the driving performance over several minutes, and is also activated at 65 km/h.

The third system is called Collision Mitigation by Braking (CMbB), and it's task is to help a driver that is about to run into another car from behind. The system intervene in three steps. In a potentially critical situation, where the driver seems not to react, a visual and audial warning is given. In a second step the system

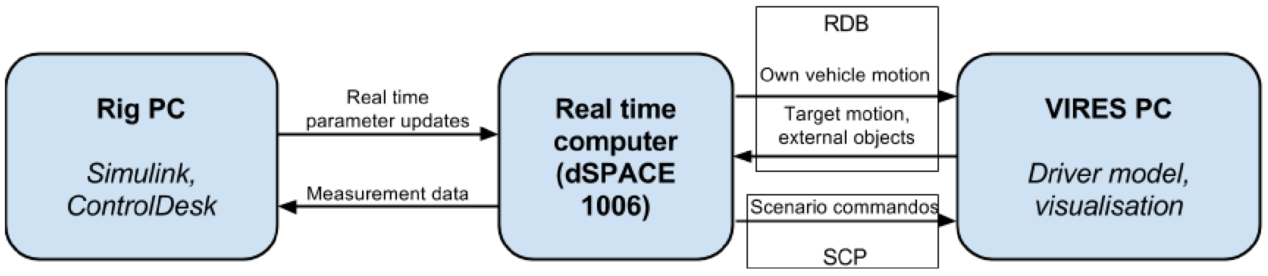
RDB
Own vehicle motion
Target motion, external objects
Scenario commandos
SCP
Rig PC
Simulink, ControlDesk
Real time parameter updates
Measurement data
Real time computer (dSPACE 1006)
VIRES PC
Driver model, visualisation

Figure 2.5.1: *The figure shows the connection between the rig PC, the real time computer and the VIRES PC. The set up is described in section 2.5.*

reacts to and improve the driver's braking by pre-charging the brakes and giving a swifter brake response, and in the third step the system intervenes with fully autonomous braking.

2.4 Development tools

MathWorks

MATLAB 2011b with Simulink was used for all modelling. This is also the framework in which the complete HIL model is built. The state-flow library was used for creating a state machine. All data analysis was done in MATLAB.

dSPACE ControlDesk

dSPACE's ControlDesk Developer Version 3.7.2 and ControlDesk Next Generation was used for interaction with dSPACE boards. This includes loading of application and collecting data logs. It also includes manipulating constant blocks in real time.

VIRES

VIRES Virtual test drive was the software used for generating traffic scenarios and rendering graphics. It was also VIRES that provided the driver model used as the basis for the project. Software from VIRES was also used for logically and graphically building the road and map used for testing and verification. The same map and road are used for the tests at the EIE (Electrical Integrations Environments) group.

2.5 Experimental set up

The set up is shown in figure 2.5.1. The skeleton model, including the developed driver model, is built in Simulink. The model is compiled to a dSPACE 1006 target board. This is via Ethernet connected to an external computer on which the VIRES software is run. On this computer both visualisation, simulation of other vehicles and the black box driver model is run. Data about positioning and movement of the own car is sent from the real time computer, and target acceleration and steering angle, as well as information about the road, other cars etc. is returned from the VIRES computer. This communication is done on the so called Runtime Data Bus (RDB). Specific commands in XML format can also be sent to the VIRES computer via Secure Copy (SCP).

3 Method and Implementation

3.1 Model design

The different scenarios that were to be tested, and for which driving models thus were to be implemented, where

- Run-off-road
- Collision from behind
- Non alert driver.

Each model was first designed conceptually, with one or more free parameters representing physical or cognitive characteristics. The general model should then have a switch indicating which behaviour should be used at the time. Here, one option should be that the normal driver model from VIRES is used. The model is to take the driver's place in the complete model, so it should control the pedal and steering wheel positions.

The model design was done with a couple of different criteria in mind. First of all, it had to be manageable to integrate with the complete Simulink model of the electrical system and surrounding plant models. The generated driver behaviour should also be realistic, and in a good way mimic real driving. The generated behaviours should also be of a kind that would be reacted on by the corresponding safety systems given that they worked as intended. As the main focus for the testing in the EIE rigs is not function verification but verification of a correct integration, this is perhaps even more important than the realism of the model.

The code should also, as always, be easy to read and understand. As driver modelling lies somewhat beyond the normal scope of the group's work, this was perhaps even more important than it would have been for parts closer to the focus area, and this should, if needed, be prioritised before realism.

3.1.1 Collecting driving data

In order to make the model resemble real driver behaviour, we need samples of what real driver behaviour looks like. There is generally three different ways to get this (Markkula 2013). One way is to set up an experiment in a simulator environment, where drivers are presented to some kind of scenario and their behaviour is recorded. Another possibility is to do the same thing on a test track or in a real traffic environment, but still with the recording as the main purpose. A third way to get this kind of data, and the one used in this study, is to record the sensor data during normal car usage, a so called Field Operation Test (FOT). This method generally generates loads of data, with all pros and cons following that. One large FOT is the euroFOT project, a cross European project aiming at making road transportation safer and more efficient (*euroFOT* 2013). The project involves 28 partners, of which Volvo Cars is one. Data collected by logging every sensor in the car with a frequency of 10 Hz was stored from several Volvo cars during one year's driving.

Data from euroFOT was used both in the design process and in the validation of the models. The data used was from driving cycles when the active safety functions Lane Departure Warning (LDW) or Driver Alert Control (DAC/DIMon) where activated. More precisely, sequences of 10 minutes before the system interventions and 1 minute after the same were filtered out. In the cases where the intervention happened before ten minutes driving, or the driving was stopped within one minute after the system interaction, the time series were cut at the driving period's end.

The design process also included discussion with the co-supervisor about what earlier research and hypothesises suggested about how the relevant driving scenarios arise and how they look.

3.2 Low level models

3.2.1 Lateral low level model

The car's lateral motion is controlled with a steering wheel. The motion of the steering wheel, and thus the driver's lateral control, can be described as a function $\theta(t)$. This function implies an angular speed of $\dot{\theta}(t)$, and is obtained by applying a torque T_d on the steering wheel. All these three entities have their own sensors in the car, within different parts of the electrical systems. As a consequence, they are all input to the model of the electrical system used in the HIL environment. This may lead to inconsistency, which may or may not affect certain functionality. Also, the ability to freely set the the steering wheel angle introduces unlimited angle rates, which of course is not realistic and may also affect functionality.

In earlier versions of the model, this has been dealt with by only using the steering wheel angle as input to the model. This has then been smoothed and "continuousised" by making a sliding average, and the (discrete) derivative of this was used as the angle rate input. As the steering servo motor (EPAS - electrical power assisted steering module) works to keep the driver's applied torque to a given value, dependent on the current turning rate, the torque input has been set to this function of $\dot{\theta}$.

However, this approach has a couple of problems. One of them is the problem of introducing a real EPAS, when it's input is calculated based on the assumption that the EPAS is already working perfectly. Another one is the possibility of fully controlling the steering wheel angle, so that interactions from the safety functions and resistance from the tires are not taken into account. The new decided approach was to control the steering wheel angle through the torque. This was done by letting the rotation of the steering wheel be calculated in a torque equation including the braking torque from the wheels and supporting torque from the EPAS. The applied torque from the driver, T_d, was controlled by a PID-controller, using the error in steering wheel angle as input signal.

Letting the steering column be affected by T_d and a lower torque T_c from the steering rack (a force F_c applied at a gear with radius r_c), and letting the steering rack besides F_c be affected by forces F_{wr} and F_{wl} from the wheels as well as F_e from the EPAS gives us theequations

$$\begin{aligned}\ddot{\theta} &= \frac{T_d - F_c r_c}{J_c}\\ \ddot{x}_r &= \frac{F_c + F_e + F_{wl} + F_{wr}}{m_r}\\ x &= r_c\theta\end{aligned}$$

which ends up as

$$\ddot{\theta} = \frac{T_d + r_c(F_{wl} + F_{wr}) + (r_c/r_e)T_e}{J_c + m_r r_s^2} \tag{3.2.1}$$

Calibration of the PID controller was done by hand in a first step. This was done by first setting K_i to zero and K_d to a small value, and then slowly increasing K_p. As this only affected constant blocks in Simulink, it could be done during running in ControlDesk. When the system started to oscillate when changing the target angle, K_p was decreased to a bit under half its value. Then K_i was increased enough to correct the offset in a short time, without leading to instability. Finally, K_d was increased to decrease the settling time of the system. After this, the parameters were fine tuned until the system behaved in a satisfying way.

3.2.2 Longitudinal low level model

Mapping the desired longitudinal dynamics of the car to a required input, i.e. pedal positions, can be done in two fundamentally different ways. One is to find a suitable approximation of the inverse of the function that maps pedal position onto acceleration, and the other one is to apply a classic control model.

If the behaviour model outputs a target velocity, a PID controller can be used to control the pedal positions. Such a controller would need some kind of anti-wind-up, as the pedals have a limited range and the desired output of the controller thus can't always be realised. This would mean that the transfer function for the pedal positions should be

$$p(s) = K(b\hat{v} - v + \frac{T_d s}{1 + \frac{T_d}{N s}}(\hat{v} - v) + \frac{1}{T_i s}(\hat{v} - v)) + \frac{1}{T_{i,sat} s}(p - \tilde{p}) \tag{3.2.2}$$

where $\tilde{p}$ is p saturated to the interval $[-1, 1]$ and the pedal positions are given by $p_a = \min(\max(p, 0), 1)$ and $p_b = -\min(\max(p, -1), 0)$.

Such a model was used in the EIE group before this master's thesis work, when trying to obtain and keep a given velocity.

However, the driver model provided by VIRES doesn't have target velocity as output, but rather a target acceleration. This requires a new low level model, either based on the dynamics of the car or constructed as a controller.

Let's explore the first alternative. An approximation of the kind described in the beginning of this section can be achieved by looking at the complex (external) vehicle model used for simulation, and step by step removing or linearising the parts assumed to be least important, until a sufficiently simple model is achieved.

The vehicle propulsion model used in the Electrical integration environments group has three important non-linear parts. The first one is the so called pedal mapping, which is a function that maps the pedal position and vehicle speed to a requested engine torque. The second one is also the most complicated one, mapping the requested torque to the torque actually applied by the engine. This includes a complex and far from linear model of the engine, as well as the logic and mechanics of the gear box. The last one is a saturation of the acceleration, since the friction between the wheels and the road is limited. The first two of these non-linearities include multiple look-up tables of experimental data from complete cars as well as part systems.

A reasonable assumption is that a normal driver neither has an understanding of this complexity of the power train, nor of the current state of the internal parts. Thus, it is probable that the driver instead has a model that is not based on the internal functionality but on experience of the vehicle dynamics.

A very simplistic model, based on this assumption, is that the applied acceleration depends linearly on the pedal pressure, with different coefficients (and different signs) for the two pedals. Together with an external braking force, dependent only on the vehicle's velocity, this constitutes a first model. Assuming that the driver never presses the two pedals at the same time, and that the braking force is linearly dependent on the velocity, one gets for desired acceleration $\hat{a}$ the equations

$$p_a = k_a(\hat{a} + cv) \tag{3.2.3}$$
$$p_b = k_b(\hat{a} + cv), \tag{3.2.4}$$

with both values saturated to the interval $[0, 1]$. Here, k_a is positive, whereas k_b is negative and typically has a smaller absolute value. In order to optimise this model with respect to the more advanced vehicle model inherit in the simulator, one can measure how the acceleration varies with the pedal pressure and velocity. These measurements can then be used to fit the free parameters using linear regression. The results of such a linear regression of the acceleration as a function of the acceleration pedal position and the velocity is seen in figure 3.2.1. As can be seen, the real function is everything but linear in v and p_a, and at the changes of gears the function jumps. This is most evident for high pedal pressures and low speeds, i.e. when quickly accelerating from being nearly stationary.

Most important for the acceleration's speed dependence is not the braking forces but the current gear. One way to deal with this is to let the coefficient for the pedal pressure be a velocity dependent function, $a = f_{gear}(v)p - c_{drag}v$.

The method above implies open loop controlling, and as always with open loop one can't expect it to exactly reach the desired velocity. The system is also vulnerable to disturbances, as compensation of such requires the information gained through feedback. Thus, perhaps a better approach would be to use a PID controller, controlling the pedals in order to obtain the desired acceleration. This does not at all require the same deep understanding of the car's dynamics, and is because of that likely a better model of how the driver actually acts.

The parameters of a PID controller mapping the target acceleration to pedal positions could be set by starting off with the parameters for the controller mapping velocity to pedal positions. As the acceleration is the derivative of the velocity, the proportional coefficient from the velocity based controller can be used as integral coefficient in the new controller, and the old derivative coefficient would then be used as proportional coefficient. To this a small derivative coefficient should be added added. This method should be further investigated, as lack of time and problems with the model resulted in the approach being abandoned.

A third approach was to convert the target acceleration to a target velocity, by assuming a look ahead time, and using the velocity obtained with constant acceleration during this time as target velocity. This velocity was then used as input to the velocity based PID controller. The choice of preview time was based on inspection of the behaviour of the PID controller, suggesting that this often had a delay of about $2/3\,\mathrm{s}$. This would mean that this preview time would result in a correct acceleration. The $2/3\,\mathrm{s}$ matches well with the preview time proposed by Gipps (1981), even if this was in a slightly different context.

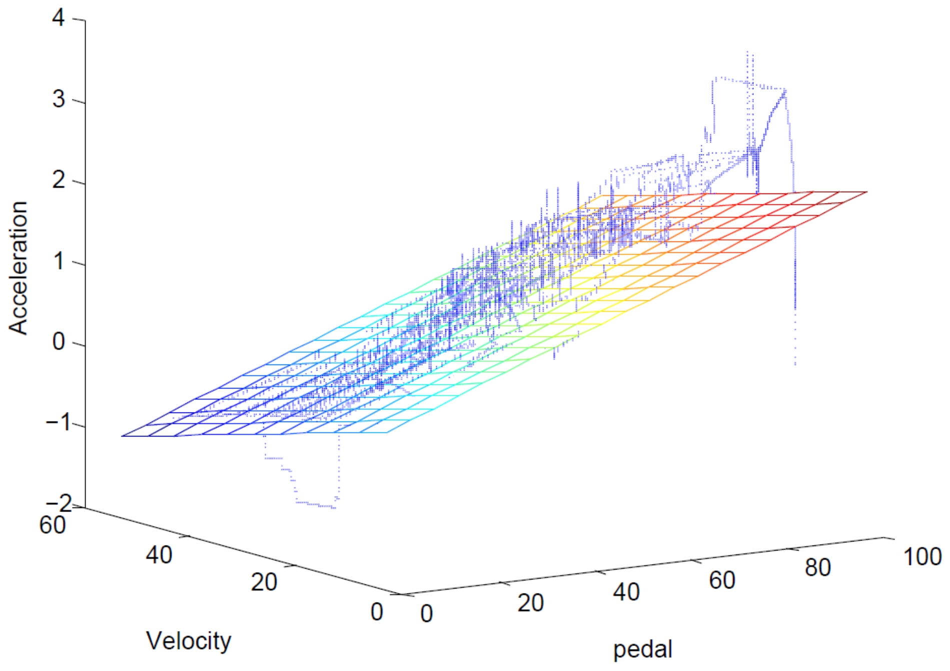

Figure 3.2.1: *The figure shows the acceleration's dependency on pedal position and vehicle velocity. The data is obtained through simulation using the same vehicle model that is used in the testing environment. The surface is a two dimensional linear regression of the data. As can be seen, there is quite a big difference between the regression and the data, showing that a linear model is not that accurate. It is also evident that the same pedal position and velocity can result in very different accelerations. The velocity and acceleration are measured in* m/s *and* $\mathrm{m/s^2}$ *respectively.*

3.3 Behaviour models

3.3.1 Run off road

The model of a run off road scenario is based on the assumption that this kind of situation often arises when the driver is distracted by something (looking at his cell phone, talking to someone in the car, etc.), and by accident turns the steering wheel by a small angle. This is modelled as adding an offset angle to the steering wheel, initiated with a soft ramp. This is realised by letting the angle $\theta(t)$ be a weighted mean between the final angle (correct angle θ_0 plus offset $\Delta\theta$) and the angle at the last time step:

$$\theta_\tau = \frac{\theta_{\tau-1} + c(\theta_0 + \Delta\theta)}{c+1}.$$

Assuming $\theta_0 = \theta(0) = 0$, this gives us

$$\theta_\tau = \Delta\theta(1 - \left(\frac{c}{c+1}\right)^\tau). \tag{3.3.1}$$

As the model time step is 1 ms, we can rewrite this in continuous form as

$$\theta(t) = \Delta\theta(1 - \mathbf{e}^{-\frac{t}{t_0}}), \text{ with} \tag{3.3.2}$$

$$t_0 = -\frac{1}{\ln\left(\frac{c}{c+1}\right)} \text{ ms.} \tag{3.3.3}$$

where t_0 is a characteristic time for the process.

In order not to leave the road entirely, which would make it harder to combine the behaviour with others in a test cycle, the run off road behaviour is limited to a certain driving distance. This is done in a state flow model, by after a certain distance entering a return to road behaviour state. This return was according to the hypothesis only the result of regaining control, that is returning to the normal driving model (from VIRES). The distance was set manually to let car just leave the road, and 100 m proved to be a good number.

As the safety systems are supposed to act differently depending on the direction of the off road drifting, the model was implemented for both directions. Two different offset angles were also used, to give different drifting speeds.

For system testing purposes, the behaviour model was connected with a part of the Simulink model sending commands using SCP (Secure Copy) to the VIRES computer, allowing it to not only control the own car but also the rest of the traffic. That made it possible to, with a single mouse click, trigger the behaviour and place a car in the road lane being drifted into, either approaching or driving alongside the own car. By default, an approaching car driving in 100 km/h was placed in the opposite lane when running of road to the left (into the approaching lane).

3.3.2 Non alert driving

This model is supposed to mimic the driving behaviour of a driver that, of one reason or another, is not alert and thus drifts to and fro on the road. The rationale is to test the trigging of the DAC function, which shall warn a driver that seems to be tired or generally not alert enough to drive safely. Two different models for this was implemented, which have quite opposite approaches.

Slow updates

The background to the first model is a hypothesis of what is the main difference between a driver that is alert and one that is not. The idea is that a non alert driver not keeps the eyes on the traffic or the road, and thus not always knows what the traffic situation looks like. Translated to a driver model, this means that the input to the driver model would not be not updated in every time step, but rather only in the time steps where the driver is focused on the surrounding. This was modelled by only updating the output of the driver model (target angle and acceleration) with a probability p at each time step. If the random number $r_\tau \sim U([0,1])$ was larger than p, the target values were kept from the time step before.

Sine wave

The idea behind the other model was not primarily to mimic a realistic behaviour, but to trig the DAC function. The function is based on a camera image from the front window of the car, and estimates the car's position in the lane. By varying the car's position in the lane, one should be able to trig the function. As the basic driver model was black box, it was not possible to directly affect the target position. Instead, an offset angle was added to the target steering wheel angle. The offset angle function of time was given the shape of a sine wave. As the steering wheel angle (for small angles) is proportional to the second derivative of the lane offset, this should give sine wave shaped oscillations of the lane offset too. This holds if the compensatory steering from the driver model can be neglected. The free parameters of this model are the frequency and amplitude of the steering wheel angle oscillation.

3.3.3 Collision from behind

A collision from behind scenario scenario was created by implementing the longitudinal model proposed by Gipps (1981). According to the paper, the model could mimic critical braking situations by using a "bad" set of parameters. The idea was to use a small value on the parameter $\hat{b}$, which corresponds to the expected maximal deceleration of the vehicle in front. This would lead to a too small safety distance to the car ahead. This in turn would result in a collision if the car ahead braked rapidly.

The driver output from the model of Gipps is a velocity. This means that a velocity dependent low level model has to be used. For this, the PID controller already implemented in the model was used. It also introduces a rather big time delay τ, suggested to be equal to the updating period used in the model. However, the updating time used in the present driver model is 1 ms, and not the 2/3 s proposed by Gipps. It was realised that the desired time delay quite well matched the delay in vehicle velocity introduced by the PID controller, why no further delay was added. This solution, however, means that the driver's reaction time is only 1 ms, as changes in for example speed for the vehicle in front immediately affects the pedal positions of the own vehicle.

In order to create a collision scenario, another car was placed in front of the own car, driving with a steady velocity for fifteen seconds (enough to reach a steady state velocity and distance for the own car), before rapidly braking. This was done with SCP commands, as with the run off road scenario.

3.4 Parameter tuning and verification

When a model or part of a model is developed, it is important to know whether it behaves as intended. This does not only include giving output on the right form (which is easy to check), but also that the output is realistic. This can obviously be a lot trickier, and depending on the type of the model different kinds of methods can be used. The verifications can also be on different levels, spanning from simple qualitative verification, for example just looking whether the result seems reasonable or as intended, to more rigid quantitative ones. The result of the later ones should be some kind of number measuring how well the output data of the model (for a given input, if that is needed) represents the data we want. A problem with this is that in many cases do we not know what data to expect from the model, and finding this out (or finding other ways to verify the model) can very well be the most difficult part of the verification problem.

In both the tuning and validation, selected parts of the FOT data described in section 3.1.1 was used.

Even though all parameters were tuned to a default value, it could be interesting to change them for specific test cases. This can be done in real time during running, through ControlDesk.

3.4.1 Tuning and validation of the longitudinal low level model

As stated earlier in the Method section, one substantial part of the driver model is the transformation from wanted vehicle behaviour, such as velocity, acceleration or turning radius, to physical interaction with the car (i.e. pedal positions and steering wheel angle). For the longitudinal part of this problem, two different features are desired. First of all, the output is supposed to make the (modelled) car follow the intended dynamics. Secondly, the generated outputs, in this case the pedal positions, are supposed to mimic the behaviour of a real driver. As a specific dynamic behaviour of a car can be closely resembled by a range of different interactions from the driver, the way it is obtained is also important when estimating the quality of the model. Does the modelled driver for example always press the pedals fully and control the speed by the lengths of the pressures,

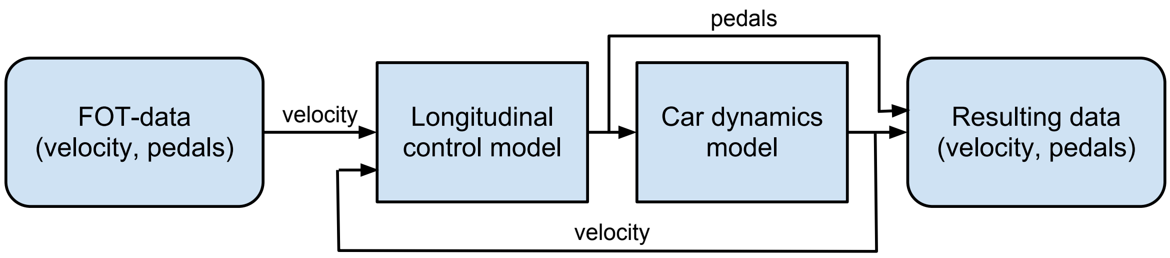
FOT-data (velocity, pedals)
velocity
Longitudinal control model
pedals
Car dynamics model
Resulting data (velocity, pedals)
velocity

Figure 3.4.1: *The set up for verifying the longitudinal control model. The resulting velocities and pedal positions are compared to the ones from the FOT data*

or does he or she find a steady state pedal position or at least a one with smallest possible high-frequency components?

For the validation of the longitudinal control model the FOT-data from the LDW-cases was used. As the data sets were collected from 10 minutes before the LDW-interaction and 1 minute after, it was assumed that the data would correspond well to normal driving and that the lane departure and any reaction to that would be of minimal importance. This should be the case, as the drifting and retaking of control normally take place on the scale of fractions of a minute, so the larger part of the time should be normal driving. These time sets were concatenated, so the result was a driving cycle of more than an hour. The first 30 minutes of these time series were then used, in order to get feasible running times. The velocities and accelerations respectively from this data set were then given as target values longitudinal car control models, resulting in output pedal positions. These pedal positions were then given as input to the car dynamics model used in the rig, which then simulated the car's movement. This movement was also used as feedback signal to the driver models. A sketch of this can be seen in figure 3.4.1.

The resulting velocities and pedal positions were to be compared to the ones from the FOT data. This was done by calculating the correlation between the original velocity and the one obtained by simulation, regarding them as two different stochastic variables. The same was done for the accelerator and brake pedals. The correlation between two random variables is given by

$$\rho_{X,Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}. \tag{3.4.1}$$

With a series of n measurements of the two variables, the correlation coefficient can be estimated by

$$r_{x,y} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \sum_{i=1}^{n}(y_i-\bar{y})^2}}. \tag{3.4.2}$$

This is the value that is calculated when using the MATLAB command `corr(x,y)`, which was done here.

3.4.2 Tuning and validation of the lateral model

As for the longitudinal control, we need to make sure that the target steering wheel angle is followed when using the control model. Two methods were used for this, none of them using the FOT data. The reason that the FOT data was not used here, was mainly that this was done before access was granted to the data.

As input to the PID controller and steering rack dynamics model, a time series of target angles was wanted. The first approach was to generate a random data series. The series should represent real driving, and thus not include any big jumps, and should be more or less centred around 0 degrees. It should also be sufficiently varying, in order to test the reaction time of the driver. Such a time series was generated by

$$x_i = \theta x_{i-1} + \epsilon_i, \tag{3.4.3}$$

where ϵ was a series of Gaussian random variables, and θ was a number slightly smaller than one. The parameters where set in order to keep the series from include too big values (angles larger than 3π radians where not wanted as they could not be realised with the steering wheel) but still be varying enough to show the behaviour of the PID controller.

The second approach, which was the one used in the end, was to generate the time series by driving one lap around the course with the VIRES driver model, and simultaneously using the obtained steering wheel angle to control the car.

Using these time series of target steering wheel angles, the resulting angles after applying the PID controller and dynamics equation were measured, and the correlation between target and resulting angle was used as fitness measure. Then each of the controller parameters were varied separately until a local maximum was found

3.4.3 Tuning the behaviour models

Run off road

The tuning of the parameters for the run off road behaviour was based on FOT data from the situations where the LDW system had been trigged. Of these 133 situations, the 15 ones that had been classified as "Crash relevant", and represented driving on straight roads, were found suitable for parameter tuning. The reason that straight road driving was found most interesting was that this meant that lane offset and steering wheel angle were directly connected ($\ddot{x}_{offset} \sim \theta$), and that the steering wheel angle just before the system intervention would be the one we wished to model. From these 15 situations, the ones where the drifting was fast and that thus matched what was intended to be modelled were chosen and used for model verification and parameter tuning. For each of them an offset angle and rise time were found, and from these parameters the model parameters were chosen.

The parameter c was set to 400, giving $t \approx 400.5\,\text{ms}$. The offset was chosen to $\Delta\theta = 0.04\,\text{rad}$ for fast drifting and $0.02\,\text{rad}$ for slow drifting.

Non alert driving

As the DAC function analyses the driving behaviour over several minutes, it was not relevant to try to find a typical behaviour just before the system intervention. Instead, the statistics of parameters such as lane offset and steering wheel angle for a longer time period before the intervention were studied. Specifically, their distribution over the sample space were investigated. According to Sandberg (2011), both the standard deviation of steering angle and lane offset have been used as measures of driver sleepiness in earlier studies. These were compared to the distributions of the same entities for the first minute of each LDW intervention scenario. This was done since those minutes hopefully show a good representation of normal driving. The hypothesis was that both lane offset and steering wheel angle would vary more when the driver was non alert than during normal driving.

When no characteristics were found in either lane offset or steering wheel angles, the approach was changed to choosing parameters such that the driving behaviour was visibly changed, without making the car running of road when driving in velocities up to 110 km/h on the curvy country road. This was done by manual tuning and visual inspection, for both the random update frequency and added sine wave method.

Collision from behind

As no FOT data was available for collision from behind scenarios, the parameter tuning was based on visual inspection. The goal was to get a driver model that had a steady state safety distance to the car ahead that made it collide when the car ahead braked fast from high speed. Initial parameter values were taken from Gipps (1981), and modified to give the desired scenario.

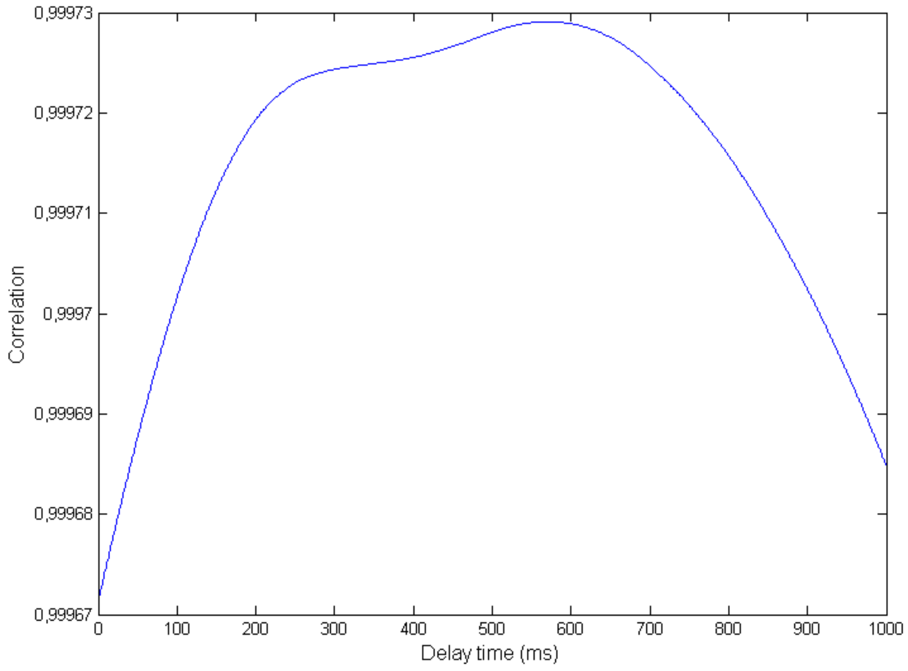

Figure 4.1.1: *The figure shows the correlation between the target and obtained velocity using the velocity based longitudinal controller, with the target velocity delayed. The peak at* 0.58 s *means that the delay before the target velocity is obtained is about this big. In a specific situation, this delay is of course dependent on how much the target signal varies. This delay is close to the recommended updating period in Gipps (1981) of* 0.67 s.

4 Results and discussion

4.1 Low level control models

The verification of the control models are rather straightforward to do quantitatively. Below are the result of those quantifications presented.

4.1.1 Longitudinal control

As mentioned in the method part, we want to know how well the resulting velocity follows the goal velocity, obtained in a field operation test. We also want to know to what extent the output pedal levels match the pedal positions from the FOT. A good control model is a one where these match up good, which is the same as the correlation between the FOT and the lab time series being close to 1 for velocity, acceleration pedal and braking pedal.

Velocity based controller

The PID controller mapping velocity to pedal positions, that was already implemented in the model, did follow the target velocity in a smooth way. This was also shown when feeding it with FOT data. The correlation between target and obtained velocity was 0.9935, which means that the signals were close to identical. The correlation between the acceleration pedal position in the FOT data and the obtained position in the controller was 0.83, thus significantly smaller. Most of this difference can however probably be explained by the fact that road state such as slope and curves are not fed to the vehicle model. This means that identical pedal positions would not generate identical velocities.

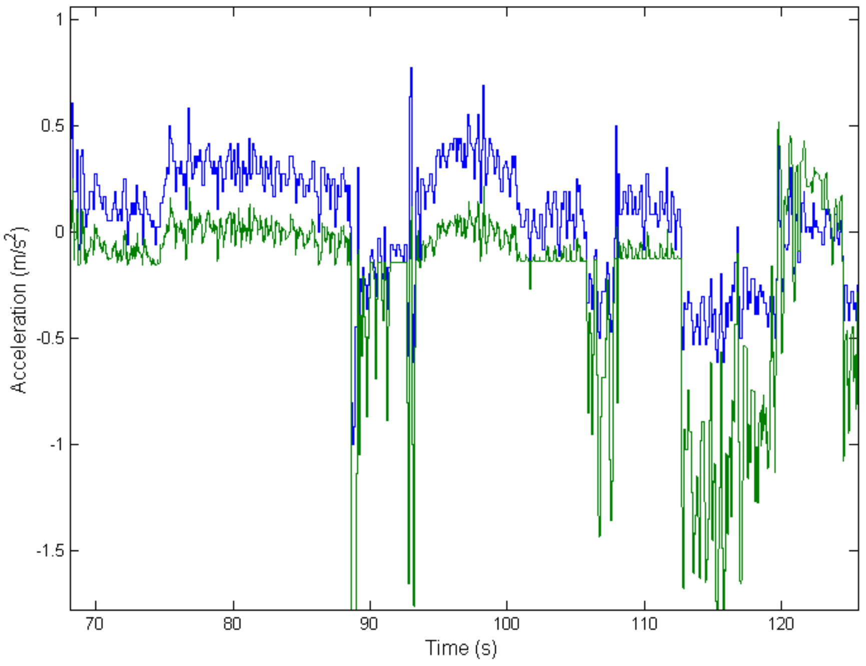

Figure 4.1.2: *The figure shows the acceleration in a part of the validation of the longitudinal open loop low level model. The blue line shows the target acceleration and the green line the acquired acceleration. It is evident that for these (small) accelerations, the obtained acceleration is much too low.*

Open loop model

The open loop model turned out to be very weak, and was not able to follow the target acceleration in a satisfying way. This was perhaps not too surprising, as open loop control generally have this tendency.

The correlation between the target and obtained acceleration, using target data from the FOT time series, was 0.5538. A plot of the target and obtained accelerations as function of time can be seen in figure 4.1.2. In figure 4.1.3, the two series are plotted against each other.

Conversion of target acceleration to velocity

The approach to convert the target acceleration to a target velocity, and then use the already implemented PID, was the one that turned out to work best of the three acceleration based low level models. The response made this approach highly usable for following the target acceleration given by the VIRES driver model, and these two can thus be used as a baseline driver model.

The correlation between the target and resulting acceleration was 0.7447. The two accelerations plotted together can be seen in figure 4.1.4, while figure 4.1.5 shows them plotted against each other.

4.1.2 Lateral control

Optimising the lateral low level control gave very good correspondence between target and achieved steering wheel angle. A (local) optimum for the PID controller was found for the parameters $k_p = 0.15$, $k_i = 0.05$ and $k_d = 8$. For these values, the correlation between target and achieved angles was as high as 0.9999965. However, optimising the PID controller, resulted in a driver that was stronger that what is realistic. Typical values measured in the torque sensor in a real car are up to 3 Nm, with significantly lower values as default. The torque applied by the modelled driver was at the entry of the curves often as high as 7-8 Nm. The torque was also oscillating with very high frequency to keep the steering wheel in a steady state. The torque as a function of time is shown in figure 4.1.6. The oscillating torque generated oscillations also in the steering wheel, as seen in figure 4.1.7.

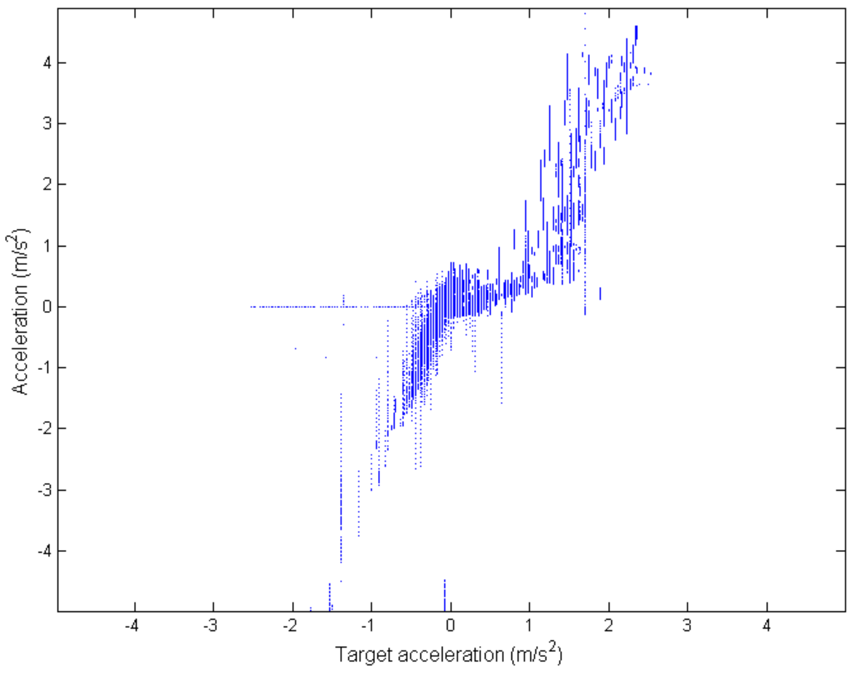

Figure 4.1.3: *The obtained acceleration using the open loop model is plotted against the target acceleration. A perfect model would generate a straight line through the origin with slope 1. One can see that for small accelerations, the obtained acceleration is too low, while the opposite is true for high accelerations. The correlation between the two series are 0.55*

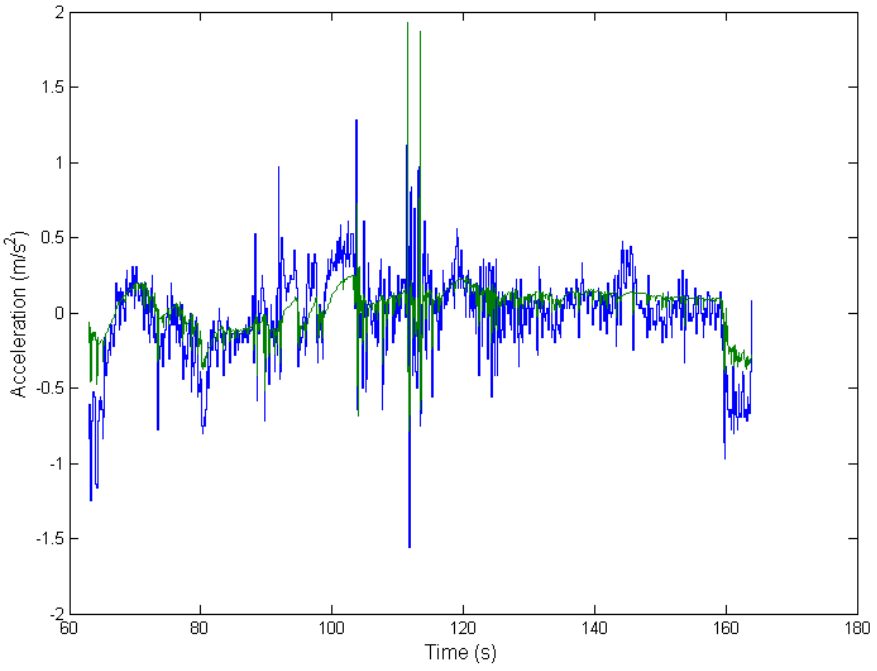

Figure 4.1.4: *The blue line shows target acceleration, collected from FOT data, and the green line shows the corresponding obtained acceleration using the low level model converting the target acceleration to a target velocity (using a preview time of* $2/3\,\mathrm{s}$*).*

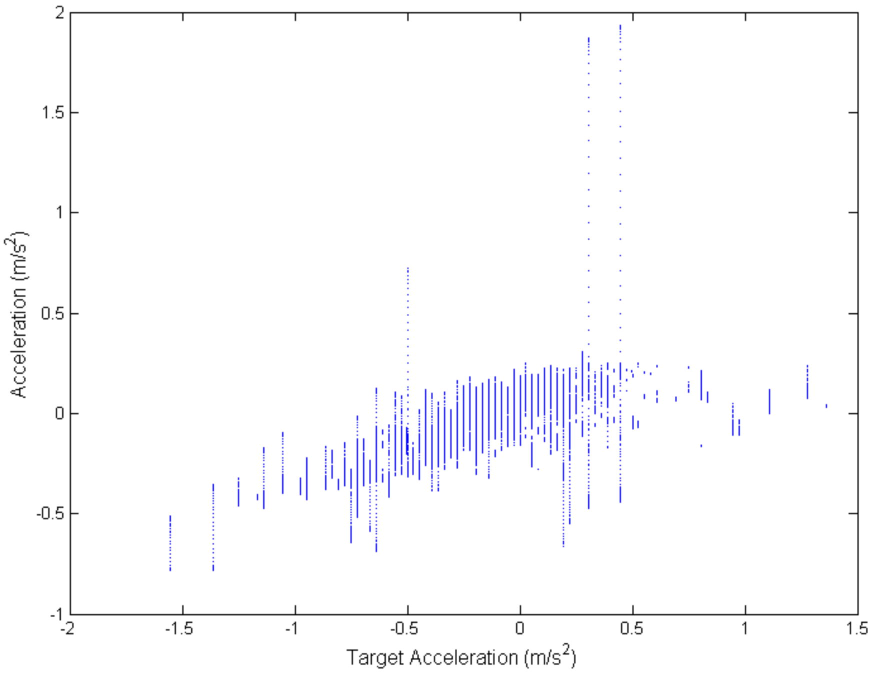

Figure 4.1.5: *The figure shows the target and obtained accelerations plotted against each other using the acceleration to velocity conversion approach. The correlation was 0.74, which is much better than the result using the open loop model.*

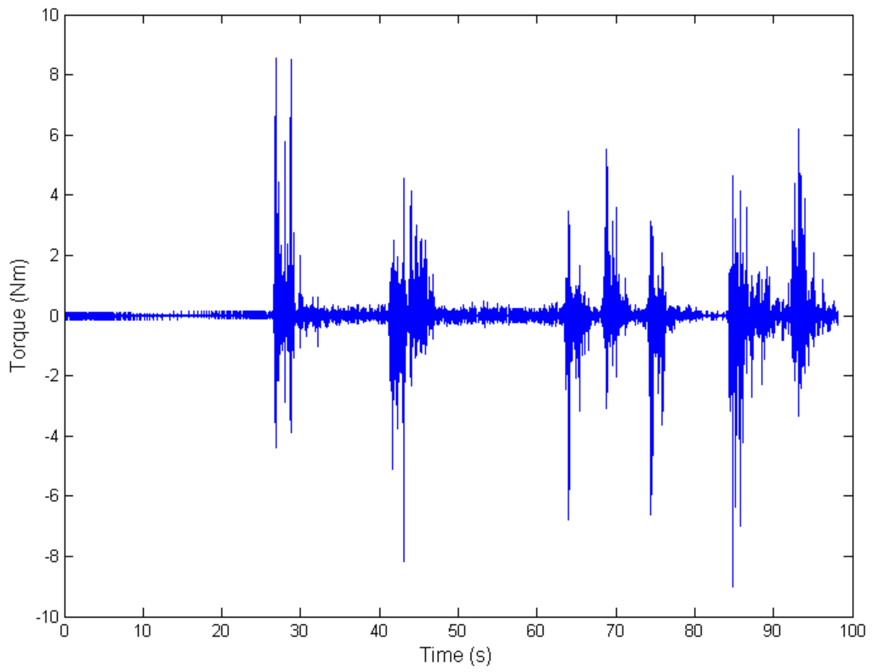

Figure 4.1.6: *The torque applied on the steering wheel by the model driver using the optimised lateral low level model, when driving around the country road lap at* 110 km/h. *The torque peaks are much larger (factor* ~ 3*) than what is usually measured in the car. It is also a lot more high frequent. This is explained by the fact that only the angle error is used as optimisation criteria for the model, and not the torque.*

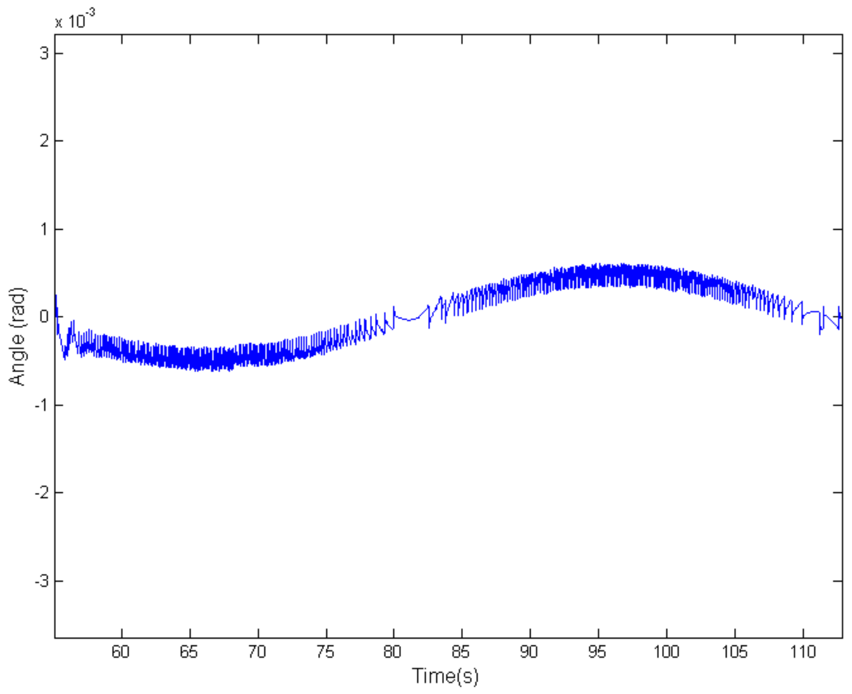

Figure 4.1.7: *High frequency variations in the steering wheel angle, introduced by the low level model. As this model applies large, and oscillating, torques, this behaviour is introduced.*

4.2 Analysis of FOT data

4.2.1 Lane departure warning interventions

From the 15 time series representing crash relevant situations on straight road driving, 8 were found to reflect a sudden run off road, of the kind that was intended to be modelled. It turned out that the characteristics of 6 of these where more or less identical. These characteristics matched the model very well The steering wheel angle from the time about ten seconds before the intervention to about two to three seconds before the intervention varied between -0.05 and 0.05 radians. At two to three seconds before intervention there was a quick change of angle with between 0.025 and 0.04 radians, before the angle was stable for about one and a half second. During this time, the lane offset was rapidly increasing. This was followed by a very fast turn back with about 0.5 radians. This turn was typically started half a second *before* the safety system intervention. A figure of a typical log of the steering wheel angle can be seen in figure 4.3.4.

4.2.2 Driver alert control interventions

The distributions of lane offset as well as steering wheel angle for the last minute before each intervention of the DAC function can be seen in figuers 4.2.3 and 4.2.1 respectively. The data was collected from 1114 interventions from the DAC function. Corresponding entities from the first minute of each of the LDW time series, which would reflect normal driving, are seen in figures 4.2.2 and 4.2.4. As noted in chapter 3, the hypothesis was that the variance would be significantly higher for both distributions in the case with non alert drivers. This would be so, as slower reactions would lead to further drifting from the centre of the lane. This in turn would force the driver to apply a greater steering wheel angle in order to return to the lane centre. However, it is clear from the figures that the case is the very opposite. The sample standard deviation in lane offset from the LDW cases was 0.78 m, while it is only 0.47 m in the DAC intervention scenarios. For the steering wheel angle the difference was even more evident, with a standard deviation of 3.8° for the non alert drivers but 40.5° for the LDW case.

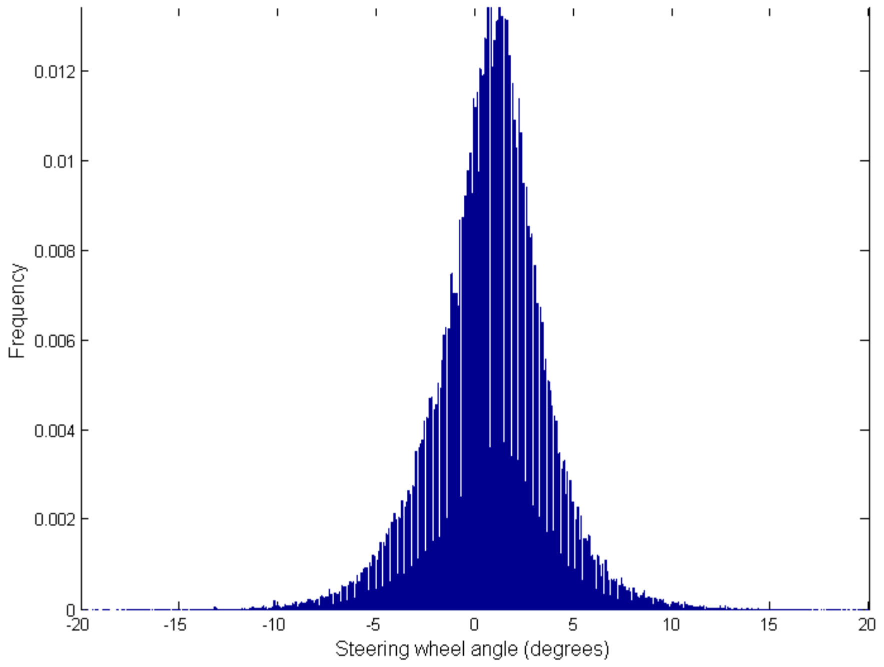

Figure 4.2.1: *This figure shows the distribution of steering wheel angles in the FOT data from the last two minutes before each intervention from the DAC system, that detects driver sleepiness. The distribution is bell shaped with a much smaller variance than for the data from long before LDW interventions, seen in figure 4.2.2.*

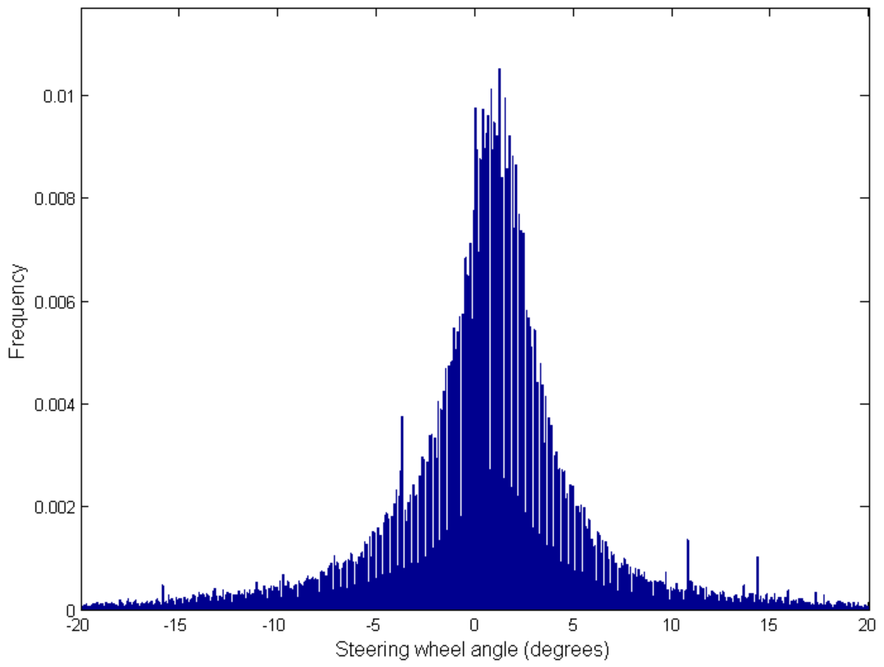

Figure 4.2.2: *The figure shows the distribution of steering wheel angles ten to eight minutes before LDW interventions. The hypothesis was that this would correspond to normal driving. However, this distribution has much larger variance than the corresponding distribution from just before DAC interventions, seen in figure 4.2.1*

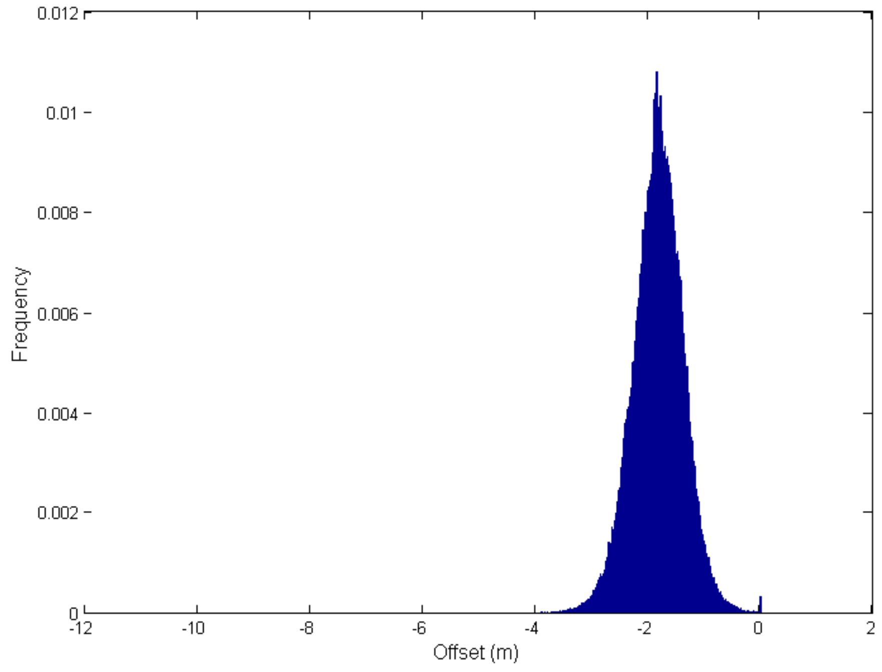

Figure 4.2.3: *The figure shows the distribution of lane offset (from the line) during the last two minutes before an intervention from the DAC system. The data come from the FOT. The offset is limited to negative values. The variance is significantly smaller than in the data from what was supposed to be normal driving, seen in figure 4.2.4.*

4.3 Behaviours

4.3.1 Normal driving

First of all, we take a look at the model of normal driving, which is the driver model from VIRES used together with the lateral low level model described above. The lane offset during one lap around the course can be seen in figure 4.3.1. The corresponding steering wheel angle series is shown in figure 4.3.2. One can see that the offset spikes in the enter and exit of every turn, followed by small oscillations in steering wheel angle. It is also interesting to see the low frequency oscillation in the offset obtained during the long straight part of the road. What gives rise to this behaviour is not known, as the driver model is black box.

4.3.2 Run off road

Out of the 8 fast run of road scenarios on straight road that were found in the FOT data, 6 were found to match the hypothesised steering wheel angle behaviour well enough for the two parameters to be estimated. As the storage frequency in the FOT data is only 10 Hz, the certainty in the time parameter is quite small. The mean offset in angle was 0.037 rad and the mean characteristic time was 0.4 s. The estimated parameters from the six scenarios are found in table 4.3.1.

After the return to normal driver behaviour, the steering wheel angle oscillates quite heavily. Oscillations are per se not wrong, but can be seen in the FOT data as well. However, these oscillations are both heavier and more even than the ones in the FOT data. These oscillations do most probably occur of the same reason as the ones occurring in the model when overtaking other vehicles and (to some extent) when entering sharp curves, as unwanted behaviour of the VIRES driving model.

The steering wheel angle as a function of time for an activation of the run off road behaviour can be seen in figure 4.3.3. The corresponding lane offset is shown in figure 4.3.5. Comparison can be done with a typical real scenario from the FOT data, with angle and offset in figures 4.3.4 and 4.3.6.

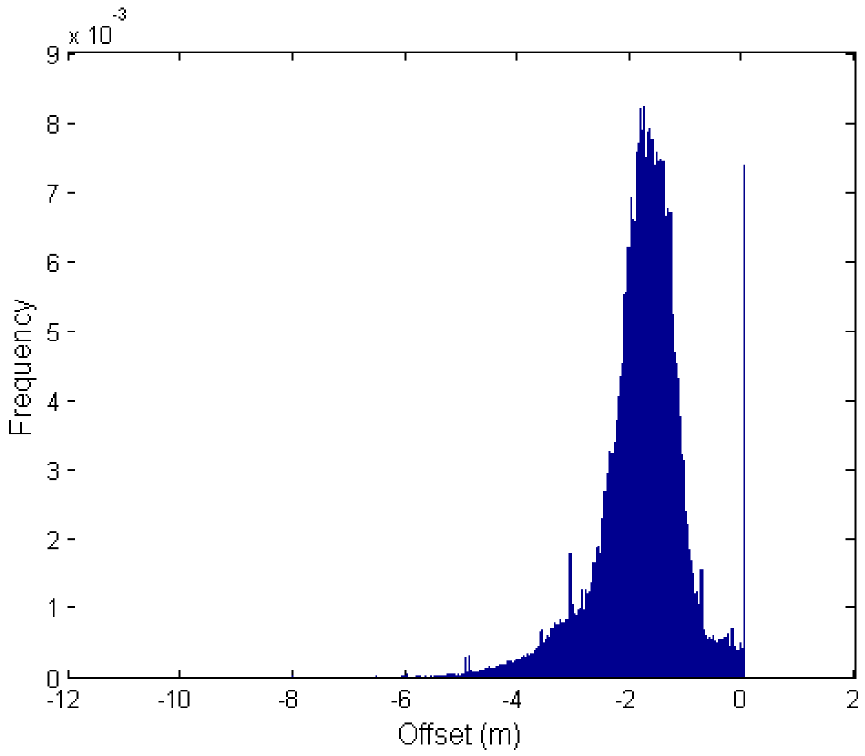

Figure 4.2.4: *The distribution of lane offset (from the line) from ten to eight minutes before each LDW intervention in the FOT data. This was supposed to correspond to normal driving. Just as in the case with steering wheel angles (figures 4.2.2 and 4.2.1, the variance is larger in this data than in the one with sleepy drivers, which is not the expected result.*

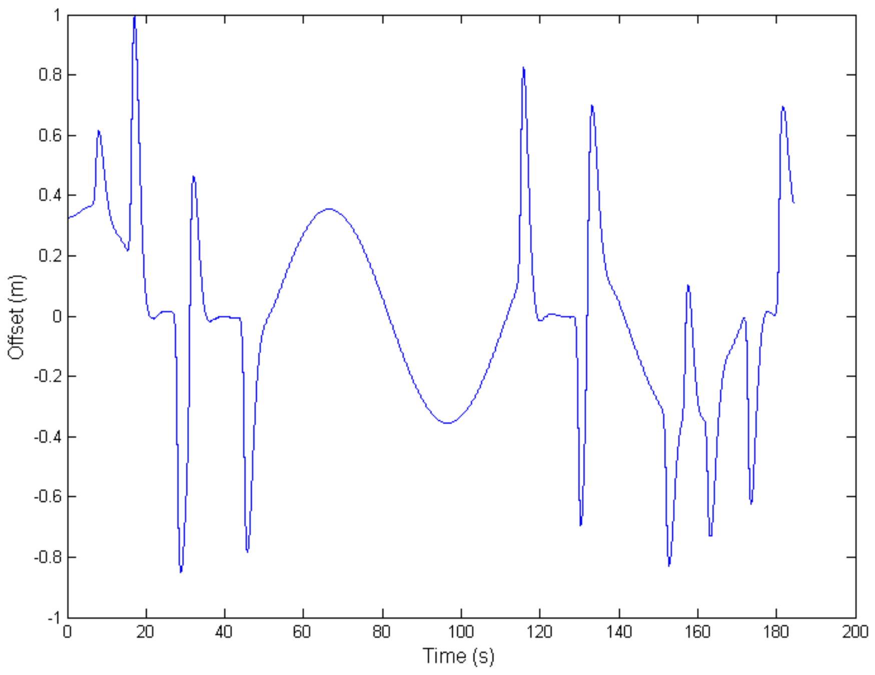

Figure 4.3.1: *The figure shows the model driver's offset from the centre of the lane during one lap on a country road course, using the VIRES driver model and the optimised lateral low level model. The peaks come from entries exits of curves, while the sine shaped part in the middle is a long straight road.*

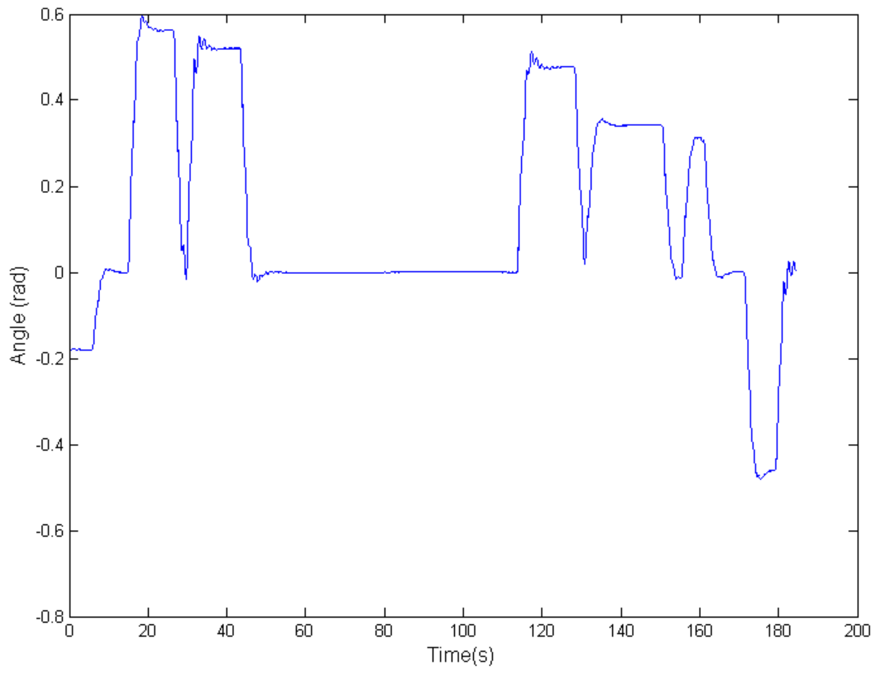

Figure 4.3.2: *The obtained steering wheel angle as a function of time during the same run that is represented in figure 4.3.1. The angle does a small overshoot in the entry of each curve, to compensate for the initial offset peak.*

$\Delta\theta$	t_0
0.068	0.8
0.041	0.4
0.032	0.4
0.032	0.2
0.045	0.1
0.008	0.6

Table 4.3.1: Data from the six LDW cases from the FOT data that matched the model for fast run off road scenarios

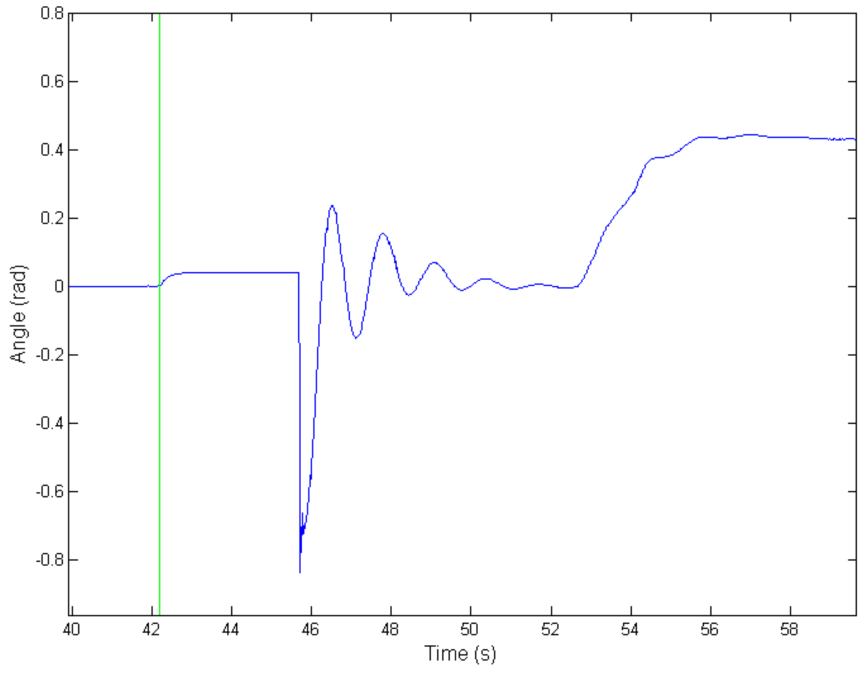

Figure 4.3.3: *Steering wheel angle as a function of time in a modelled run off road scenario. The green line marks the activation of the scenario. Compared to a real near accident situation (figure 4.3.4), the angular velocity when returning to the track is much higher.*

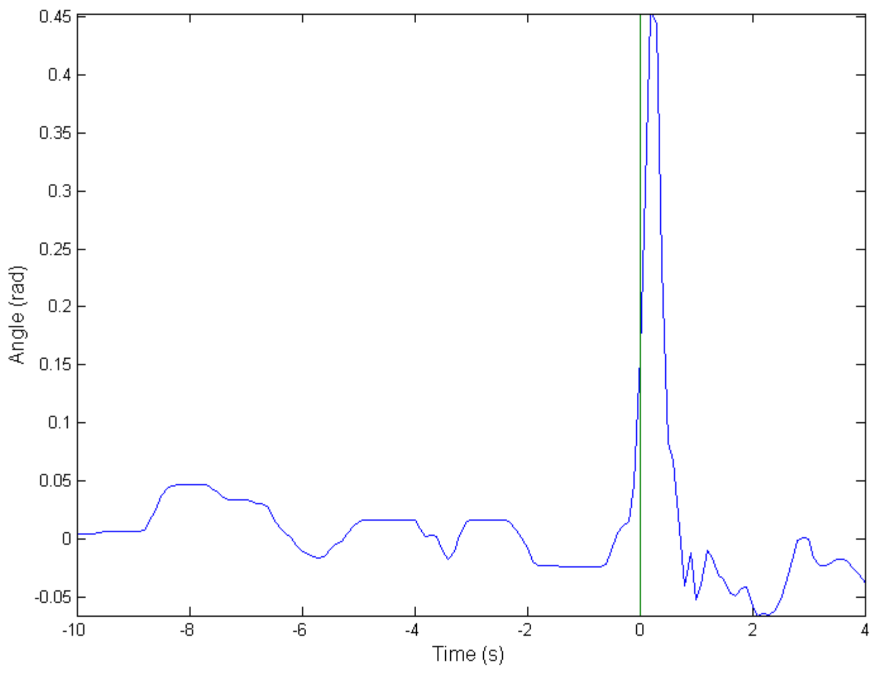

Figure 4.3.4: *Steering wheel angle as a function of time in a typical fast run off road scenario. The data is collected from the euroFOT project. The green line marks the intervention from the LDW system.*

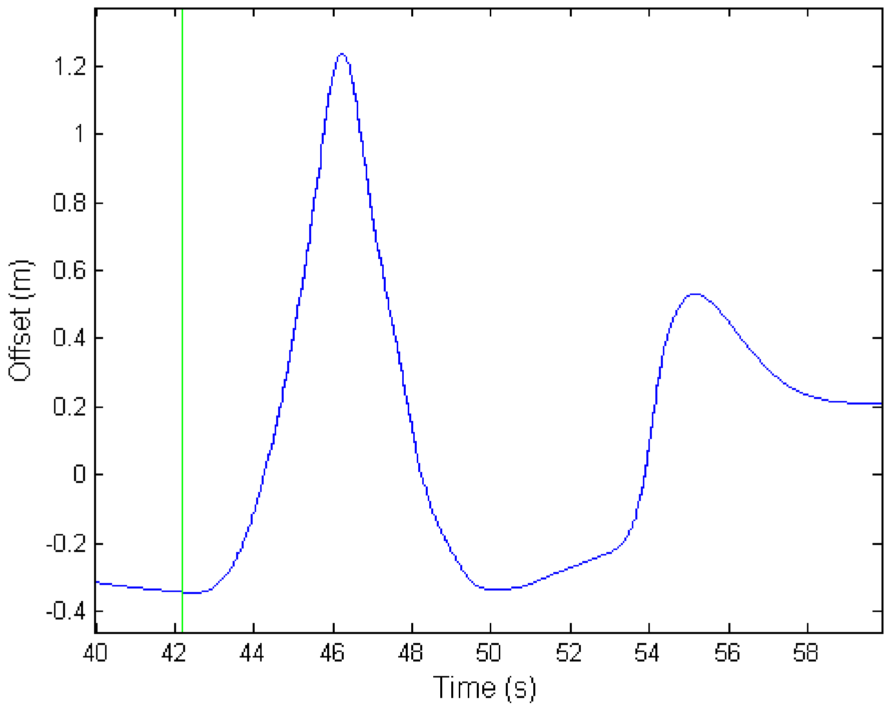

Figure 4.3.5: *Lane offset as a function of time when modelling a run off road scenario. The green line marks the activation of the scenario.*

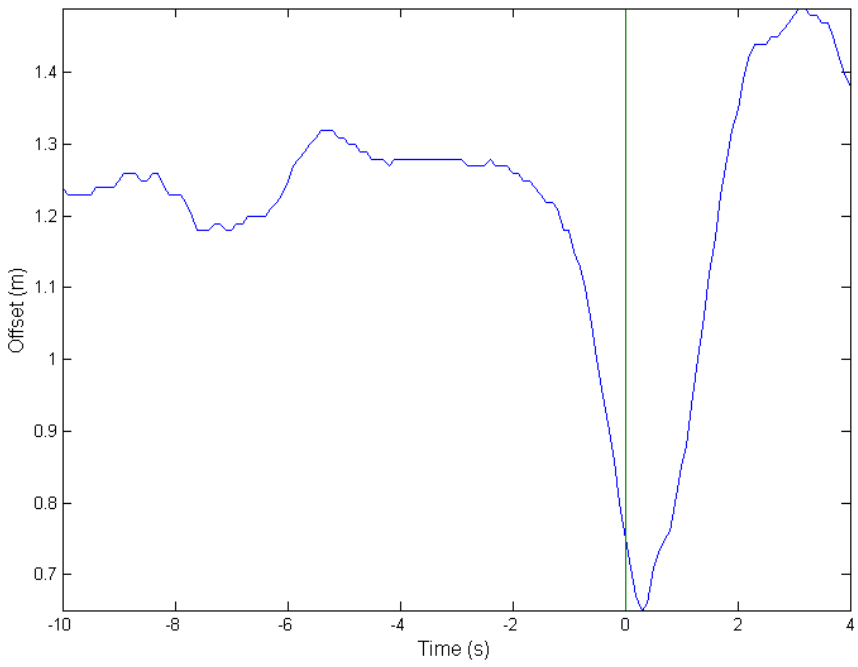

Figure 4.3.6: *Lane offset as a function of time in a typical fast run off road scenario. The data is collected from the euroFOT project. The green line marks the intervention from the LDW system.*

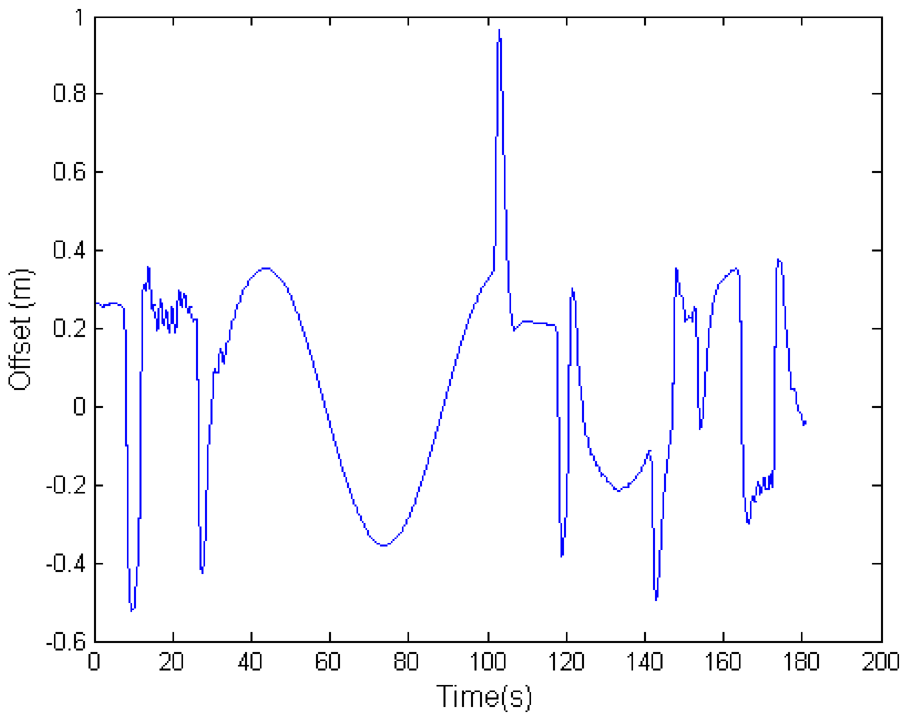

Figure 4.3.7: *The lane offset is here shown as obtained using the non alert driver model based on random updating frequency. Some of the peaks from figure 4.3.1 have disappeared, but the figures are quite similar.*

4.3.3 Non alert driving

Random updating frequency

The model based on a random updating frequency turned out to be hard to get stable and at the same time get to move notably within the lane. In the end, an updating probability of 0.01, giving an expected updating frequency of 10 Hz, was chosen. The lane offset during one lap on the course using this model can be seen in figure 4.3.7. This should be compared with the offset using the plain VIRES driver model, which is seen in figure 4.3.1. The steering wheel angle during the same lap is seen in figure 4.3.8. As can be seen, oscillations in the steering wheel occur at every curve entry. However, this gives small or no effect on the lane offset. This suggest that, if it indeed is in lane offset the low alertness is seen, this is not a good model of a non alert driver.

Sine wave offset

The model with an added sine wave offset on the steering wheel angle was significantly easier to tune in order to keep the car on the road in a range of velocities and road sates, but still get variations in lane offset. This is what is wanted, in order to trig the DAC function without risking that the car leaves the road, and that the test thus has to be aborted. The chosen parameters were a frequency $f = 1\,\text{rad/s}$ and amplitude $A = 0.1\,\text{rad}$. Early tests show that this was able to trig the DAC function. The resulting steering wheel angle and lane offset as function of time can be seen in figure 4.3.9 and 4.3.10.

A problem with this approach is that it is highly unrealistic, and gives no explanation on why the behaviour would arise. Form a testing point of view, this is of minor (if any) importance, but as driver model it is a big weakness.

4.3.4 Collision from behind

As noted in section 3.4.3, no FOT data was available for verification of this behaviour. Instead, the evaluation was based only on visual inspection. It turned out that in order to get the desired behaviour, placing the car with a short steady state distance to the car ahead and unable to brake fast enough to avoid a collision, was very hard. However, it was possible with the parameters $b = 2.1$ and $\hat{b} = 2$ (see section 2.1.1 for explanation). However, using more reasonable parameters, such as the ones proposed by Gipps, the car follows the car ahead

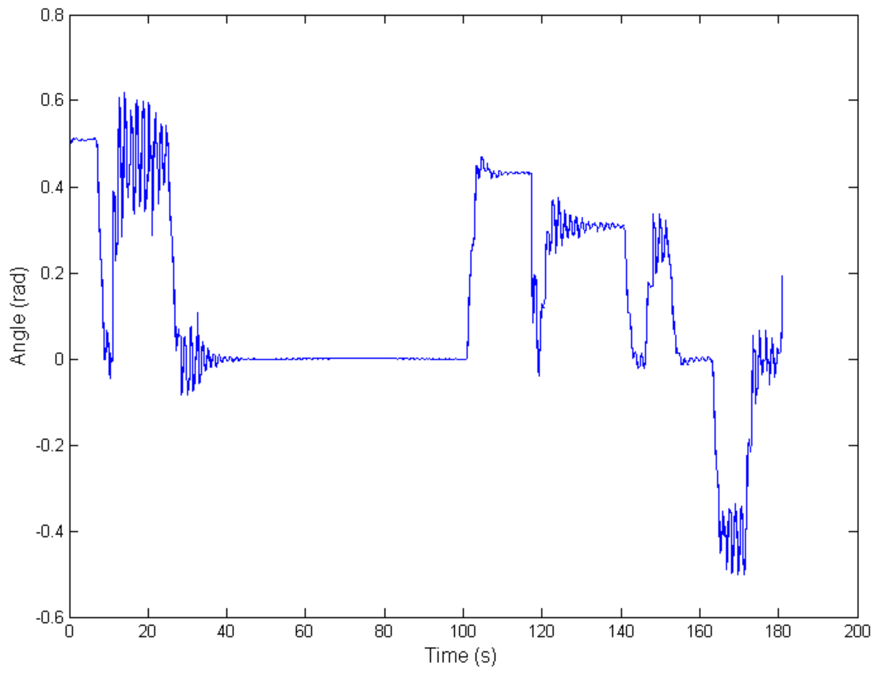

Figure 4.3.8: *The figure shows the steering wheel angle during one lap using the random updating frequency model. As can be seen, each curve introduces a small oscillation, which is damped as the normal target angle remains constant. Using smaller updating probability, the oscillations can instead increase, making the car run off road.*

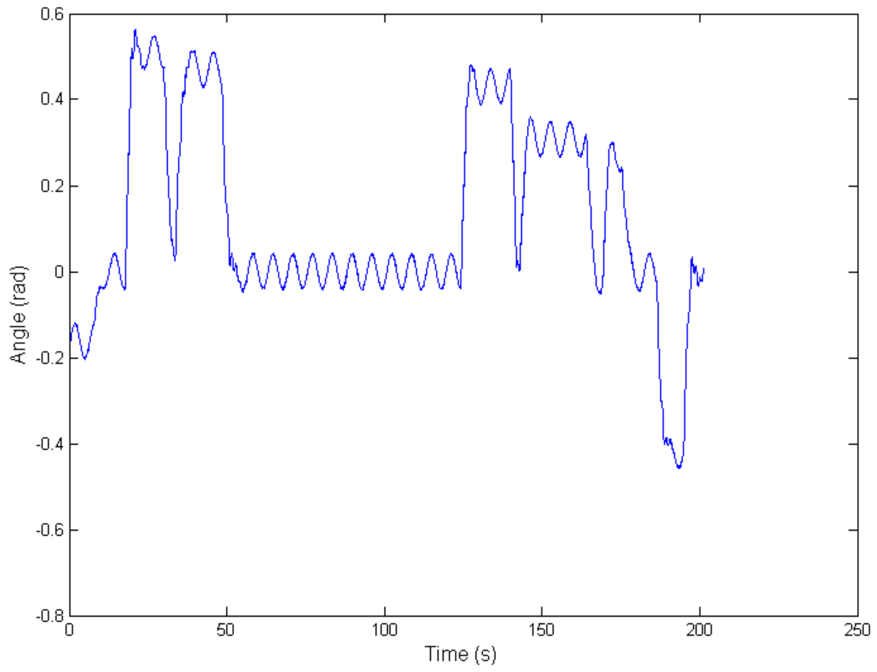

Figure 4.3.9: *The figure shows the steering wheel angle as a function of time when using a sine wave offset model. Of course this is not a realistic model.*

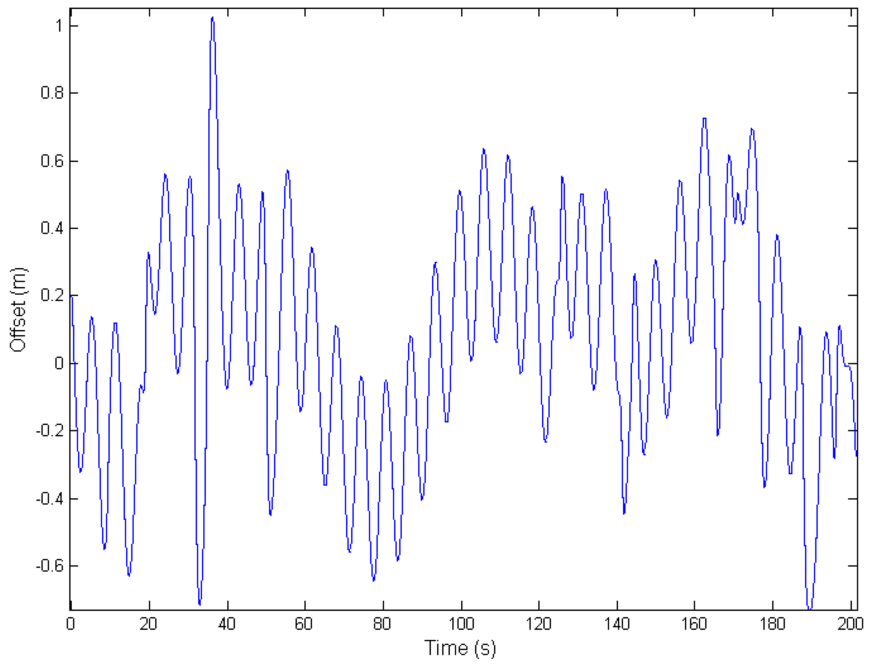

Figure 4.3.10: *The figure shows the lane offset as a function of time when using the sine wave offset model of non alert driving. The variations in figure 4.3.1 introduced by the base model are almost completely hidden in the high frequency oscillations.*

very well. This suggests that if one would like to implement a complete own driver model for the rig set up, Gipps would be a suitable longitudinal model.

5 Conclusions and future work

5.1 Low level models

As noted in chapter 4, the PID controller controlling the steering wheel resulted in much higher torques than a real driver. This was the result as the optimisation only was done with respect to the resulting steering wheel angle. Whether this is a problem or not depends on what uses the driver model has. If only the steering angle is of importance in the application, this will not be a problem. However, if the torque itself is important, this makes the driver model weak. A way to improve this would be to optimise with respect to the values of all three sensors connected to the steering rack: angle, angular velocity and torque. Using logged steering wheel angles from FOT as target angle, one could compare the resulting sensor values with the ones retrieved during driving. The very high fitness for the lateral low level model could also be an effect of over fitting, as the training and validation sets were not separated.

Another problem is the verification of the longitudinal low level models using acceleration as input. If one uses FOT data as input, the obtained velocity will eventually start drifting off from the velocity in the logged data. This makes the comparison between logged and obtained pedal positions irrelevant, as they come from different conditions. Thus, the pedal positions, i.e. the way to obtain the target acceleration, has not been compared, but only the correlation between target and obtained acceleration.

The VIRES software runs on 60 frames per second, which corresponds to a time step of 16.7 ms. The Simulink model, however, uses a time step of 1 ms. This means that the input from VIRES is considered constant for about 17 computation cycles, before making a step change. As the lateral model was very fast in its response, this could be seen in the steering wheel motion, and in the end the ability to follow these steps became important in the optimisation. As these steps are not realistic or wanted, it could even be so that a slower lateral model would mimic even steering wheel motion better, not only torque characteristics. A possible solution to this would be to filter or extrapolate the target steering angle signal before applying the low level model.

These problems with tuning and evaluating the low level models makes evaluating the behaviour models hard. It is also not certain that the complete model will be useful or good, even with realistic high level models, as long as the low level models aren't more stable and well tuned than they are.

5.2 High level models

5.2.1 VIRES' driver model

The driver model from VIRES, that is used for normal driving and as a foundation for the other behaviours, does in principle seem good and stable. However, it shows some odd behaviour when applying large steering wheel angles. Such situations are when taking over a slower car, and when returning to the road after a run off road scenario. In such cases, the car starts oscillating before settling to a stable lane offset and steering wheel angle. This is a behaviour that could be introduced by for example a controller with too large P-coefficient, but also if the target angle has too large absolute value. This could be the case if the steering wheel target angle delivered by the driver model is not the optimal angle at the specific time but the angle that the driver should target at in a longer perspective.

This uncertainty shows the perhaps greatest weakness with the provided driver model: that it is a black box. A black box model is not a bad thing as long as it works as intended, and is to be used as is. However, when being modified and added on, it is a problem not to know how the original model will react. This problem is further discussed in section 5.4.

5.2.2 Run off road

The run off road behaviour is perhaps the one closest to mimic real driver behaviour. Still, it has some problems that could be looked further into.

After the behaviour is activated, the steering wheel angle is locked, apart from the offset. This is done in order to not get a compensation from the underlying driver model. However, this leads to two problems. First of all, the model does not take into account if the road starts turning during the drifting. This is perhaps not that unrealistic, given that the model is to mimic a driver not looking at the road, but it can result in that a wanted run off road to the right instead becomes a drifting to the left, or the opposite. This can be a problem from a testing point of view. The locking of steering wheel angle also results in the steering wheel being close to constant during the drifting. This is not the case in reality, as a matter of fact the FOT data suggest that the steering wheel is only very seldom constant for more than a second at a time. Whether this is a problem in testing depends, as many of the unrealistic features, on how the functions to be tested are implemented.

The FOT data also shows quite clearly that a run off road is almost never an isolated accident, but follows a period of wobbly driving. No matter if the driver is sleepy, speaking in the cell phone or distracted by something else, he or she is not so only for the few seconds when the drifting occurs. This is not something that one has to take into account for the application of the model at hand, but is worth noting for other applications.

5.2.3 Non alert driving

None of the two approaches where satisfying in the sense that they were realistic and showed the desired behaviour. As the FOT data from DAC interventions showed little significance, and according to one of the persons working with it to a great extent captured completely normal driving, it is hard to say whether the random update approach was realistic, but it seemed not to be. Perhaps a better approach would be to add longer, but more rare, periods of distraction, using some kind of Markov model.

The use of random numbers also means that the model will not show exactly the same behaviour every time. From a testing point of view this is problematic, as it means that problems are not always possible to reproduce. The hypothesis is that the variations between different runs are so small that they won't matter, but this is just a hypothesis.

5.2.4 Collision from behind

The collision from behind model does indeed create a collision scenario, and can be manipulated in order to get more or less critical situations. However, there are things to work with here as well. Perhaps most evident is the fact that the delay, or reaction time, has been removed. This was motivated by the fact that there already is a delay in the realisation of a target velocity. But this is an implicit delay, and does not remove the unreality in a driver that directly presses the brake pedal as the driver in front of him does. A plausible assumption is even that the reaction time should be enlarged in collision scenarios. Gipps (1981) state that the model can describe congested traffic and sudden brakes, however it is possible that the model fail to mimic collision behaviours. This was not possible to verify, as no FOT data for collisions from behind was available.

5.3 Verification and parameter tuning

Both the parameter tuning and verification process could, and perhaps should, be done more rigorously. More quantitative studies could be done on both non alert driving and collision from behind, and the study of the run off road behaviour is quite weak. One should in principle always strive to separate parameter tuning and verification, and for example use different sets of data points for the two tasks. With the selection of LDW scenarios at hand, this was not possible, as there was only six data series left. One could look at a broader spectrum of scenarios by for example not only looking at the interventions marked as *Crash relevant*.

A problem with the verification of the longitudinal low level models, was that it assumed that the dynamics model used in the simulations was perfect. At least, it was not possible to separate differences between FOT and modelled data occurring due to weaknesses in the driver model from those coming from errors in the dynamics model. A way to get a grip on how big the second part is, would be to use the pedal data from the FOT's as input to the dynamics model, and compare the resulting velocity to the one stored in the same FOT data.

The verification of the non alert driver model suffered from the FOT data not being trustworthy. However, it is not clear how a good validation would have been done even if the data was good. The idea was to compare the distributions of lane offsets, steering wheel angle and steering wheel angle rate between the FOT data and model generated data. However, this would have been hard as the road section used for simulations are only a couple of kilometres, of which more than a third is a completely straight road, and the rest is a connection of a series of circle arcs. As can be seen in figure 4.3.2 this results in more or less discrete steps in steering wheel angle, and the distribution would then be very strange. In order to get a good distribution shape, a much more diverse road net would be needed. Perhaps, one would even need it to resemble the one driven on during the FOT test.

The verification and parameter tuning of the collision from behind model is even more deficient. No FOT data was at hand, meaning that visual inspection was the only thing to go on. With FOT data, one could at least have seen if there was any characteristic behaviour. This would however require that the distance to the car ahead was stored in the FOT data. If this is an entity calculated locally within the active safety system, it is probably not available in FOT data and would thus not be possible to use. How to validate the model quantitatively, or rigorously tune the parameters, using FOT data is not obvious and would need some investigation.

5.4 Future work at Volvo

A question one has to think about is how realistic driver model one really wants. This question is tightly connected to the one about what one wishes to use the driver model for. My understanding is that this is still somewhat of an open question. If the driver model should only be used for trigging specific functions, there is no need to have a realistic driver model.Then, the focus should be to keep it as simple as possible, and make the behaviours rather hard coded. Thus, the driver models implemented in this thesis don't have to be developed much further. If the driver model instead is supposed to really test the functionality of the systems, or if one wants to test the systems for a range of behaviours, more advanced driver models should be used. For this purpose, one would need to look further into both the model for non alert driving and for collision from behind. One would also need to improve at least the lateral low level model, and optimise it not only regarding angle error.

One should also consider the possibility to implement a driver model from scratch, and not use the one provided by VIRES. The VIRES driver does absolutely well enough for normal driving, but modifying a black box model with perturbations will always lead to problems as one doesn't know how the underlying model will react. The conclusion from this work is than that Gipps would state a good longitudinal model. As a simplified version of Reński (2001) (with only offset distance and angle as errors) has been previously used and found "nearly good enough", a more elaborate Reński model would probably be good as lateral model.

References

Benderius, O. (2012). “Driver modeling: Data collection, model analysis, and optimization”. Licentiate Thesis. Göteborg: Chalmers University of Technology.

Benderius, O. et al. (2011). “A Simulation Environment for Analysis and Optimization of Driver Models”. In: *Digital Human Modeling*. Ed. by Duffy, V. G. Vol. 6777. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 453–462.

Black, J. D. (2010). “Vehicle steering systems – Hardware-in-the-loop simulator, driving preferences, and vehicle intervention”. PhD thesis.

Dogan, U., Edelbrunner, H., and Iossifidis, I. (Oct. 2008). “Towards a Driver Model: Preliminary Study of Lane Change Behavior”. In: *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pp. 931–937.

euroFOT (July 2013). URL: `http://www.eurofot-ip.eu/`.

Gipps, P. (1981). “A behavioural car-following model for computer simulation”. In: *Transportation Research Part B: Methodological* 15.2, pp. 105–111.

Huang, J. (2012). “Vehicle Longitudinal Control”. In: *Handbook of Intelligent Vehicles*. Ed. by Eskandarian, A. Springer-Verlag London, pp. 168–190.

Inagaki, T. (2003). “Adaptive Automation: Sharing and Trading of Control”. In: *Handbook of Cognitive Task Design*. Ed. by Hollnagel, E., pp. 147–169.

Itoh, M., Horikome, T., and Inagaki, T. (2010). “Effectiveness and Driver Acceptance of a Semi-Autonomous Forward Obstacle Collision Avoidance System”. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 54.24, pp. 2091–2095.

Markkula, G. (2013). “Evaluating vehicle stability support systems by measuring, analyzing, and modeling driver behavior”. Licentiate Thesis. Göteborg: Chalmers University of Technology.

Markkula, G. et al. (2012). “A Review of Near-Collision Driver Behavior Models”. In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 54.6, pp. 1117–1143.

Pellecchia, A. et al. (Mar. 2005). “Making driver modeling attractive”. In: *Intelligent Systems, IEEE* 20.2, pp. 8–12.

Reński, A. (2001). “Identification of Driver Model Parameters”. In: *International Journal of Occupational Safety and Ergonomics* 7.1, pp. 79–92.

Sandberg, D. (2011). “Detecting Driver Sleepiness”. PhD thesis. Chalmers University of Technology.

Sharp, R., Casanova, D., and Symonds, P. (2000). “A Mathematical Model for Driver Steering Control, with Design, Tuning and Performance Results”. In: *Vehicle System Dynamics* 33.5, pp. 289–326.

Wann, J. P. and Wilkie, R. M. (2004). “How do we Control High Speed Steering?” In: *Optic Flow and Beyond*. Ed. by Vaina, L., Beardsley, S., and Rushton, S. Vol. 324. Springer Berlin Heidelberg, pp. 371–389.