Variable Vehicle Dynamics Design -Objective Design Methods

Master's thesis
performed in **Vehicular Systems**

by
Magnus Oscarsson

Reg nr: LiTH-ISY-EX-3348-2003

11th July 2003

Variable Vehicle Dynamics Design -Objective Design Methods

Master's thesis

performed in **Vehicular Systems**,
Dept. of Electrical Engineering
at **Linköpings universitet**

by **Magnus Oscarsson**

Reg nr: LiTH-ISY-EX-3348-2003

Supervisor: **Dipl.-Ing. Avshalom Suissa**
DaimlerChrysler AG
Dipl.-Ing. Manuel Stumpf
DaimlerChrysler AG
Anders Fröberg
Fordonssystem, LiTH

Examiner: **Professor Lars Nielsen**
Linköpings Universitet

Linköping, 11th July 2003

Avdelning, Institution
Division, Department

Vehicular Systems,
Dept. of Electrical Engineering
581 83 Linköping

Datum
Date

11th July 2003

Språk
Language

☐ Svenska/Swedish
☒ Engelska/English
☐ ______

Rapporttyp
Report category

☐ Licentiatavhandling
☒ Examensarbete
☐ C-uppsats
☐ D-uppsats
☐ Övrig rapport
☐ ______

ISBN
—

ISRN
LITH-ISY-EX-3348-2003

Serietitel och serienummer
Title of series, numbering

ISSN
—

URL för elektronisk version

http://www.vehicular.isy.liu.se
http://www.ep.liu.se/exjobb/isy/2003/3348/

Titel Variabel Fordonsdynamik - Målinriktade Designmetoder

Title Variable Vehicle Dynamics Design -Objective Design Methods

Författare Magnus Oscarsson
Author

Sammanfattning
Abstract

The goal of this thesis has been to study the behaviour of the closed loop driver-vehicle-environment in simulation and to find parameters of the synthetic vehicle model, which minimise certain optimisation criteria. A method of optimising parameters using genetic algorithms has been implemented and has proven to work well. Two different driving strategies have been tried in the optimisation of an ISO lane-change manoeuvre. The first approach has simulated a beginner driver and his or her behaviour. The second approach simulates an experienced driver and also the possibility of driver adaption to different vehicle types. The implemented driver model has shown to be sufficient to describe the driver´s behaviour during lateral manoeuvres. A parameter set which minimises the lateral acceleration response on steering wheel angle has proven to be the optimum. This includes a small steering wheel ratio, and a small but positive under steer gradient. The driver has demonstrated the ability to adapt to different vehicles, and therefore different parameter sets, describing the driver, should be used for different problems.

Nyckelord
Keywords

driver models, mental workload, parameter optimisation, genetic algorithms

Abstract

The goal of this thesis has been to study the behaviour of the closed loop driver-vehicle-environment in simulation and to find parameters of the synthetic vehicle model, which minimise certain optimisation criteria. A method of optimising parameters using genetic algorithms has been implemented and has proven to work well. Two different driving strategies have been tried in the optimisation of an ISO lane-change manoeuvre. The first approach has simulated a beginner driver and his or her behaviour. The second approach simulates an experienced driver and also the possibility of driver adaption to different vehicle types. The implemented driver model has shown to be sufficient to describe the driver´s behaviour during lateral manoeuvres. A parameter set which minimises the lateral acceleration response on steering wheel angle has proven to be the optimum. This includes a small steering wheel ratio, and a small but positive under steer gradient. The driver has demonstrated the ability to adapt to different vehicles, and therefore different parameter sets, describing the driver, should be used for different problems.

Keywords: driver models, mental workload, parameter optimisation, genetic algorithms

Acknowledgment

This work has been carried out at research department RIC/AR, DaimlerChrysler AG. I want to thank all the people who have made this time in Esslingen such a pleasant experience for me. Especially I want to thank my supervisors Dipl.-Ing. Avshalom Suissa and Dipl.-Ing. Manuel Stumpf for their support and ideas. I would also like to thank my supervisor at Linköping University, Anders Fröberg, for his support.

Contents

Chapter 1

Introduction

The vehicles of today are more and more dependant on electronic systems, for driving safety and ease of control for the driver. Some systems are more or less standard in vehicles on the market, examples of such systems are ABS, TRC, ABC, etc. Other systems are still in the research area or on their way onto the market. One example of these systems is the drive-by-wire, in which the mechanical steering of today will be replaced with either electrical, hydraulic or electro-hydraulic steering. The main benefit of this is that the steering angle on the wheels can be controlled independent from the steering wheel angle. This gives a new freedom for design of steering algorithms, which will ease the task of driving, for example the vehicle can have variable steering ratio. That is a high ratio at low speeds, for example making it easy to park the vehicle, and low ratio at high speeds for safety reasons.

Background

In the scope of drive-by-wire systems, several reference models for driving dynamics were developed and implemented at the research departments of DaimlerChrysler. Evaluation and optimisation of these synthetic models was done both in the driving simulator in Berlin and by test drives in test vehicles, see figure 1.1. The driving dynamics are described by parameter sets and can be fully customised by the driver, see figure 1.2, for certain behaviour of the vehicle. In the existing models, the driver is considered to give the command input only with steering wheel and pedals, see figure 1.3. The closed-loop relation between the driver and the vehicle has not been taken into account until now.

Figure 1.1: Test vehicles Pegasos and Technoshuttle.

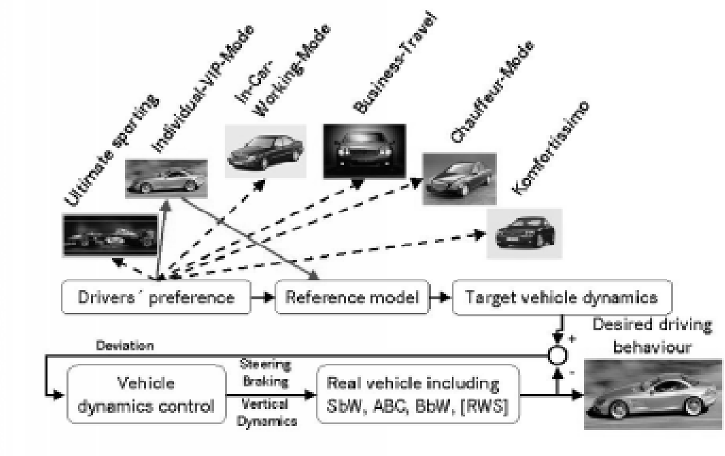

Figure 1.2: Overview of the integrated chassis control.

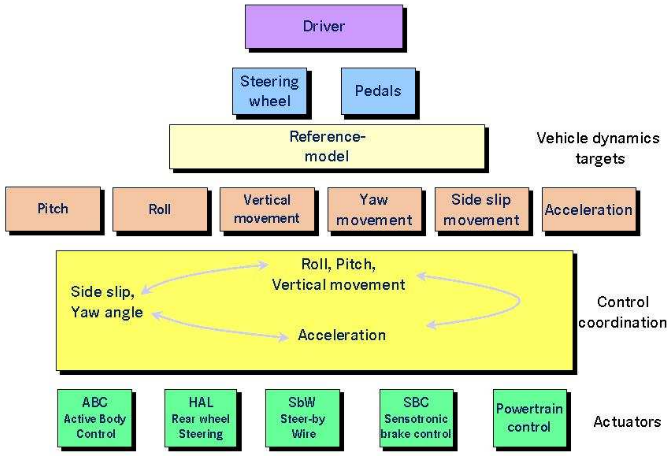

Figure 1.3: Overview of the VFD system in Technoshuttle.

Objectives

The goal of this thesis project has been to evaluate and design the driver model as a part of a control system, and to implement this in Matlab/Simulink for simulation. Using the driver models, several mathematical evaluation criteria were to be developed, taking different evaluation criteria into account, e.g. the driver´s mental workload, required steering energy, etc. The third part was to optimise parameters in the vehicle reference model regarding these criteria, during different driving manoeuvres.

Methods

This work started with a literature search for suitable driver models, after which some promising candidates were implemented in Matlab/Simulink, for evaluation together with the VFD reference model. After that, the evaluation criteria were derived and implemented together with the driver-vehicle model. Finally, the optimisation program was connected to the simulation environment and parameter optimisation was executed.

Thesis outline

The work done and the results achieved are explained in the thesis, structured in the following way.

Chapter 2 Vehicle model Explains the basics of the vehicle model used for this thesis.

Chapter 3 Driver models Contains background theory about driver modelling and an explanation of the driver models used for this work.

Chapter 4 Optimisation Explains genetic algorithms and the optimisation algorithm which has been used.

Chapter 5 Simulation results Contains information about the results achieved.

Chapter 6 Conclusions and future work Contains the conclusions drawn from this project and some suggestions for extensions and future work about the same topic.

Appendix A Contains description of the ISO lane-change track, the penalty function and an overview of the Simulink model.

Chapter 2

Vehicle model

2.1 Introduction

Vehicle modelling has been more and more used in the automotive industry due to both the need for more rapid construction and evaluation time, but also with the development of faster and cheaper computers, providing computer power and simulation tools, such as Matlab/Simulink, ADAMS or Cascade. Also, vehicle modelling can be used to test the behaviour of the vehicle in dangerous situations, e.g. crash tests, in a safe way. Several different approaches to vehicle modelling exist, from simple linear one-track bicycle models up to extremely complex nonlinear models. The modelling is often a question of how simple the model can be made, but still be valid for its intended purpose, as a more complex model inevitably requires more computation power and is also more sensitive to modelling errors.

2.2 Co-ordinate systems

When modelling vehicles, different co-ordinate systems are used depending on what is modelled, and which behaviours are to be studied. The most important co-ordinate systems are:

- *The center of gravity co-ordinate system*, (CoG), can be seen in figure 2.1 and has its origin at the vehicle center of gravity and is used as the reference for all movements of the vehicle body.

- *The fixed inertial system*, figure 2.2, is a non-moving co-ordinate system used as a reference for the vehicle´s position relative to earth.

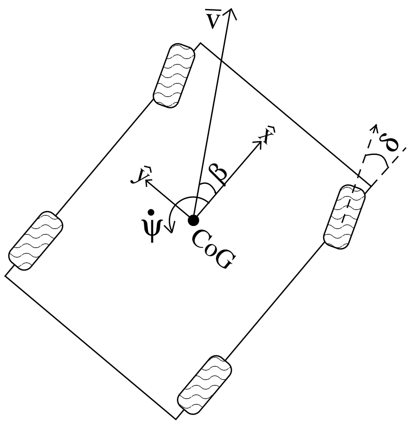

Figure 2.1: The Center of Gravity co-ordinate system.

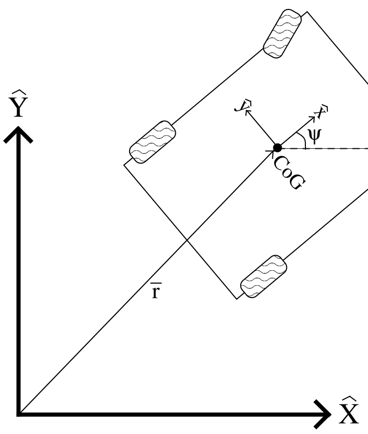

Figure 2.2: The CoG co-ordinate system in relation to the inertial.

In order to translate positions between the different co-ordinate systems the following base vector equations apply:

$$\hat{x} = \cos\psi\hat{X} + \sin\psi\hat{Y} \tag{2.1}$$

$$\hat{y} = -\sin\psi\hat{X} + \cos\psi\hat{Y} \tag{2.2}$$

Also, for this work the position of an arbitrary point on the vehicle had to be calculated, to test whether the vehicle passed the *ISO lane-change*, manoeuvre. The co-ordinates of a point $\bar{p}$ is given by:

$$\bar{p} = \begin{pmatrix} x \\ y \end{pmatrix}_{\hat{x}\hat{y}} = \bar{r} + \begin{pmatrix} x\cos\psi - y\sin\psi \\ y\cos\psi + x\sin\psi \end{pmatrix}_{\hat{X}\hat{Y}} \tag{2.3}$$

where x and y are the distances along the $\hat{x}$ and $\hat{y}$ axis from the center of gravity, $\bar{r}$, to the point $\bar{p}$, seen from the inertial system, as in figure 2.2.

2.3 VFD reference model

VFD, or Variable Fahrzeug Dynamik (Variable Vehicle Dynamics) is a synthetic model for use in lateral dynamics problems. Many of the parameters in this model can be adjusted freely to achieve different behaviours. The driver gives a steering command, δ_{LR}, on the steering wheel and a longitudinal velocity, v_x, which is transferred to the reference model. Then the reference model generates target values for the yaw-controller, which gives the steering commands to the vehicle, δ_f and δ_r, see figure 2.3. In the following subsections, the most important dynamics will be discussed in more detail, the models are taken from [2].

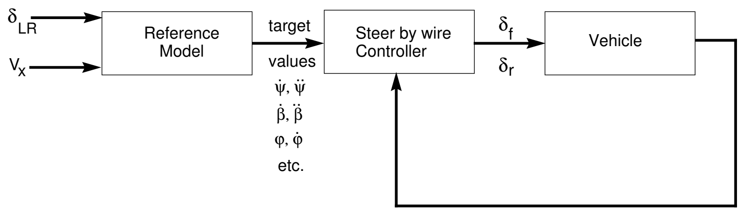

Figure 2.3: The complete VFD system.

2.3.1 Yaw

When driving in a circle with constant radius R, the yaw amplification can be described by three parameters; steering wheel ratio i_l, wheelbase l, and under-steer gradient elg. The linear equation in steady-state is:

$$\delta_{LR} = i_l(elg\, a_y + \frac{l}{R}) \tag{2.4}$$

Where the under-steer gradient, elg, is given by the following equation:

$$elg = \frac{c_h l_h - c_v l_v}{c_v c_h l} m \tag{2.5}$$

Together with the force equation:

$$m a_y = \frac{m v^2}{R} = m v \dot{\psi} \tag{2.6}$$

that gives the yaw amplification of the single-track model:

$$\dot{\psi}_s = \frac{v}{i_l(l + elg\, v^2)} \delta_{LR} = K_{\dot{\psi}} \delta_{LR} \tag{2.7}$$

With $elg > 0$ the vehicle tends to under-steer, with a characteristic velocity v_{ch} and a maximum yaw rate $\dot{\psi}_{smax}$:

$$v_{ch} = \sqrt{\frac{l}{elg}} \tag{2.8}$$

With 2.8 inserted in 2.7, the maximum yaw rate becomes:

$$\dot{\psi}_{smax} = \frac{v_{ch}}{2 l i_l} \delta_{LR} = \frac{1}{2 i_l} \sqrt{\frac{1}{lelg}} \delta_{LR} \tag{2.9}$$

For $elg < 0$, oversteer, the yaw amplification grows with velocity, with a critical velocity:

$$v_{krit} = \sqrt{\frac{-l}{elg}} \tag{2.10}$$

This velocity is critical, since it gives a pole in 2.7. This gives three freely adjustable parameters; l, i_l and elg. In a real vehicle, l is the fixed wheelbase, but when using drive-by-wire, this can be designed as desired in the reference model.

2.3.2 Body slip

In steady-state the body side-slip angle β_s is given by the following equation:

$$\beta_s = -\frac{l_h}{R} + swg\, a_{ys} \tag{2.11}$$

where the body side-slip gradient, swg, and the reference lateral acceleration, a_{ys}, are given by the following equations:

$$swg = \frac{m l_v}{c_h l} \tag{2.12}$$

$$a_{ys} = \frac{\delta_{LR}}{i_l elg} = k_{a_y}\delta_{LR} \tag{2.13}$$

Together with (2.6) and (2.7)

$$\beta_s = (\frac{-l_h}{v} + swg\, v)\dot{\psi}_s \tag{2.14}$$

2.3.3 Steady-state roll

The roll angle depends on the lateral acceleration in the following way:

$$\varphi_s = wwg\, a_{ys} = wwg\, v\dot{\psi}_s \tag{2.15}$$

where wwg is the freely adjustable roll angle gradient.

2.3.4 Steering wheel torque

The torque on the steering wheel is modelled as follows:

$$M_L = M_{Ls} + M_{LZ} + M_{LD} + M_{LR} \tag{2.16}$$

Where M_{Ls} is the main torque, M_{LZ} is the centering torque, M_{LD} is the damping torque and M_{LR} is the friction torque, respecively.

$$M_{Ls} = lmg\, a_{ys} = lmg\, v\dot{\psi}_s \tag{2.17}$$

is the main contributor, depending on both velocity and yaw rate. The other components are the centering torque:

$$M_{LZ} = \begin{cases} c_{MZ}\delta_{LR} & \mid \delta_{LR} \mid \leq \delta_{LRZ} \\ c_{MZ}\delta_{LRZ} & \mid \delta_{LR} \mid > \delta_{LRZ} \end{cases} \tag{2.18}$$

the damping torque, proportional to the angular velocity of the steering wheel:

$$M_{LD} = d_{ML}\dot{\delta}_{LR} \tag{2.19}$$

and the friction torque:

$$M_{LR} = \begin{cases} M_{LReib}\frac{\dot{\delta}_{LR}}{\dot{\delta}_{Lreib}} & \mid \dot{\delta}_{LR} \mid < \dot{\delta}_{LReib} \\ M_{LReib} sgn(\dot{\delta}_{LR}) & \mid \dot{\delta}_{LR} \mid \geq \dot{\delta}_{LReib} \end{cases} \tag{2.20}$$

The parameter d_{ML} can be expressed as:

$$d_{ML} = 2D_{ML}\sqrt{c_{ML}\Theta_{LR}} \tag{2.21}$$

where D_{ML} is a freely chosable damping mass, Θ_{LR} is the moment of inertia in the steering wheel and c_{ML} is the stiffness which depends on velocity and elg:

$$c_{ML} = \frac{lmg\, v^2}{i_L(l + elg\, v^2)} \tag{2.22}$$

For high velocities c_{ML} will be approximately constant and d_{ML} can be approximated with:

$$d_{MLmax} = 2D_{ML}\sqrt{\frac{lmg\,\Theta_{LR}}{elg\,i_L}} \tag{2.23}$$

2.3.5 Dynamic Behaviour

When looking at the dynamic behaviour of the vehicle, notice must be taken of the fact that the yaw and slip dynamics are not isolated, but coupled together in the following way:

$$a_y = v(\dot{\psi} - \dot{\beta}) \tag{2.24}$$

The equations for $\dot{\psi}$ and β are:

$$\dot{\psi} = \frac{T_z s + 1}{\frac{s^2}{\omega_0^2} + \frac{2\xi s}{\omega_0} + 1}\dot{\psi}_s \tag{2.25}$$

$$\beta = \frac{T_{z\beta} s + 1}{\frac{s^2}{\omega_0^2} + \frac{2\xi s}{\omega_0} + 1}\beta_s = \frac{(\frac{-l_h}{v} + swg\,v)(T_{z\beta} s + 1)}{\frac{s^2}{\omega_0^2} + \frac{2\xi s}{\omega_0} + 1}\dot{\psi}_s \tag{2.26}$$

This gives four parameters which can be chosen freely, two time constants, T_z and $T_{z\beta}$, damping, ξ, and eigenfrequency, ω.

2.4 One-track bicycle model

The linear bicycle model, or Riekert-Schunck model, can be seen in figure 2.4. This model has both front and rear-wheel steering. From figure 2.4 the following equations can be derived:
Body side-slip angle and rate, assuming small angles:

$$\beta = -\arctan\frac{v_y}{v_x} \approx -\frac{v_y}{v_x} \tag{2.27}$$

$$\dot{\beta} = \dot{\psi} - \frac{a_y}{v_x} \tag{2.28}$$

$$\dot{v}_y = -\dot{\psi}v_x + a_y \tag{2.29}$$

Front and rear tyre side-slip angles:

$$\alpha_v = \beta - \frac{l_v\dot{\psi}}{v_x} + \delta_v \tag{2.30}$$

$$\alpha_h = \beta + \frac{l_h\dot{\psi}}{v_x} + \delta_h \tag{2.31}$$

Lateral forces, front and rear tyre:

$$S_v = c_v \alpha_v \tag{2.32}$$

$$S_h = c_h \alpha_h \tag{2.33}$$

Newton´s second law and momentum around the z-axis:

$$m a_y = S_v + S_h \tag{2.34}$$

$$\ddot{\psi} = \frac{S_v l_v - S_h l_h}{I_z} \tag{2.35}$$

Combining (2.28), (2.29) and (2.34) gives:

$$\dot{\beta} = \dot{\psi} - \frac{S_v + S_h}{m v_x} \tag{2.36}$$

$$\dot{v}_y = -\dot{\psi} v_x + \frac{S_v + S_h}{m} \tag{2.37}$$

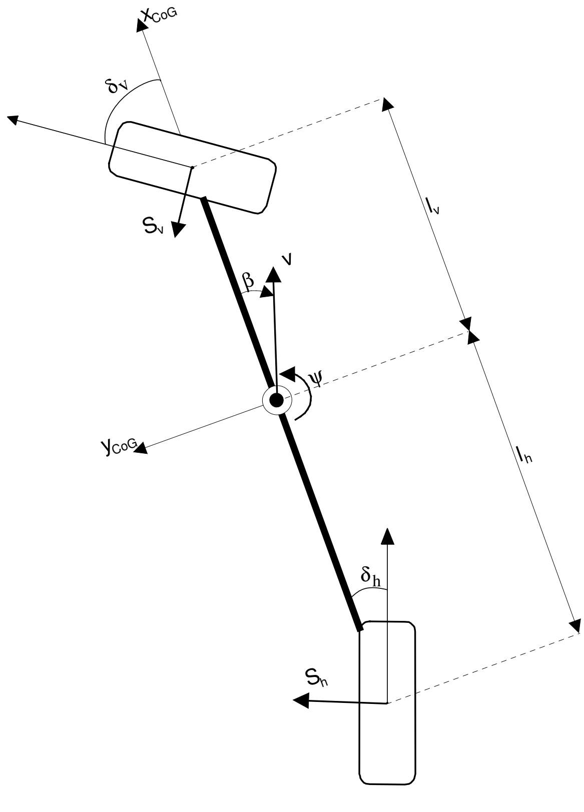

Figure 2.4: Bicycle model

With (2.30), (2.31), (2.32) and (2.33) into (2.35) and (2.37)we get the differential equations:

$$\ddot{\psi} = \frac{(c_h l_h - c_v l_v)v_y - (c_v l_v^2 + c_h l_v^2)\dot{\psi}}{I_z v_x} + \frac{c_v l_v \delta_v - c_h l_h \delta_h}{I_z} \tag{2.38}$$

$$\dot{v}_y = \frac{(c_h l_h - c_v l_v)\dot{\psi} - (c_v + c_h)v_y}{m v_x} - \dot{\psi} v_x + \frac{c_v \delta_v + c_h \delta_h}{m} \tag{2.39}$$

which describes the lateral dynamics of the bicycle model.

Chapter 3

Driver models

3.1 Introduction

Modelling human drivers is a hard task, mainly because there are no general equations describing the complex human mind, and because the driver adapts to different vehicles and traffic situations, [1], thereby changing his or her strategy and tactics. Much research has been done in the field of modelling humans, but there is still much left to explore. A general driver model is not possible to find today, but several control models exists which are more or less suited for specific tasks, e.g. keeping distance or changing lanes, the former being a *longitudinal* and the latter a *lateral* controller. The lateral controllers can be further divided into compensation tracking models and preview tracking models, both of which are explained in later sections. For a more thorough explanation about modelling humans as controllers, see [11] or [7].

3.1.1 Compensation Tracking Models

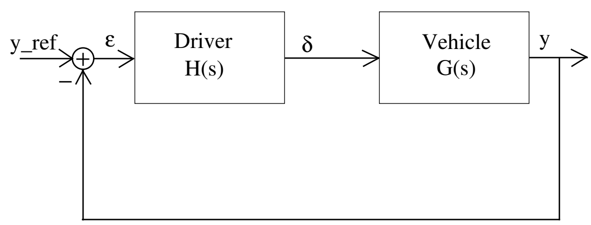

Figure 3.1: Basic structure of compensation tracking model

According to [5] a compensatory driver/vehicle model can be described in block diagram form as seen in figure 3.1. This driver model use only the lateral displacement error, ε, as input and produces a steering wheel angle, δ, as output. The simplest way of describing the driver in a compensatory way is the *PID model* which gives the driver transfer function:

$$H(s) = \frac{K_d s^2 + K_p s + K_i}{s} \tag{3.1}$$

where K_d, K_p and K_i are the derivative, proportional and integral coefficients, respectively. The major drawback of this description is that the coefficients are hard to determine. Another model presented in [5] is:

$$H(s) = \frac{Ke^{-t_d s}(1 + T_L s)}{(1 + t_h s)(1 + T_I s)} \tag{3.2}$$

where the parameters of brain response delay, t_d, and driver action delay, t_h, were introduced to represent the agility of the driver. The other time constants, the lead time T_L and the lag time T_I, and the gain K represent the driver's experience. Another approach is the *crossover frequency model*, where the driver parameters are adjusted so that the open-loop function $H(s)G(s)$ matches the following equation:

$$H(s)G(s) = \frac{\omega_c e^{t_d s}}{s} \tag{3.3}$$

where ω_c is the crossover frequency.

3.1.2 Preview tracking models

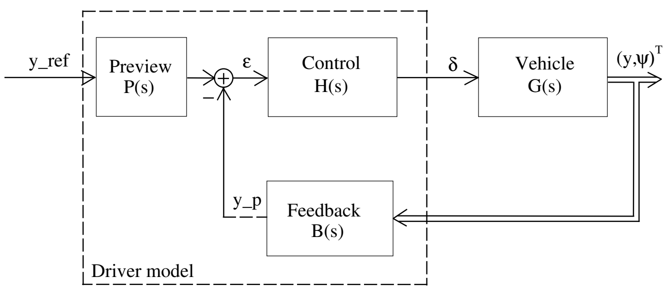

Figure 3.2: Basic structure of preview model

Preview or *look-ahead* models are a group of models which unlike the compensatory models, use future information about the path to be followed as controlling inputs. The general structure of such a model can be seen in figure 3.2 where

$$\begin{cases} P(s) = e^{T_p s} \\ H(s) = K \\ B(s) = (1, T_p v) \end{cases} \tag{3.4}$$

with the parameters preview time, T_p, system gain, K, vehicle speed, v, and the feedback vector

$$\bar{y}(t) = [y(t), \psi(t)]^T$$

This is the earliest preview tracking model, so the driver response delay was ignored. If $V\psi(t)$ is replaced with $\dot{y}(t)$, the feedback B(s) becomes a single variable $y(t)$, in equation 3.4. A more advanced model, the *second order predictable correction model* can be described with:

$$\begin{cases} P(s) = e^{T_p s} \\ H(s) = \frac{K}{s} e^{-t_d s} \\ B(s) = 1 + T_p s + \frac{T_p^2}{2} s^2 \end{cases} \tag{3.5}$$

This model includes the driver response delay, $e^{-t_d s}$, and a second order prediction feedback. Also, an integration block with gain K is introduced, to represent the driver´s correction ability. Another preview model presented in [12] gives the control input

$$\varepsilon(t) = \frac{y_d(x_{0s} + L_a) - y_{0s}(x_{0s})}{L_a} - \Psi(x_{0s}) \tag{3.6}$$

where x_{0s} is the longitudinal position, y_{0s} is the lateral position, y_d is the desired path deviation, L_a is the *look-ahead distance* and Ψ is the heading angle of the vehicle. With a steering wheel ratio i_L and a driver response delay T_k, the resulting steering command would be:

$$\delta(t) = \frac{i_L}{L_a} y_d(t + \frac{L_a}{v} - T_k) - \frac{i_L}{L_a} y_{0s}(t - T_k) - i_L \Psi(t - T_k) \tag{3.7}$$

This gives a driver model which can be described by the three parameters: aim point distance, L_a, driver response delay, T_k, and steering wheel ratio, W.

3.1.3 Multi-Input Driver model

For this work, a multi-input driver model from [6] has been used. The structure of the driver-vehicle system can be seen in figure 3.3 and the driver model is further explained in figure 3.4. The model uses both the lateral position error and the current yaw-angle as inputs. The transfer functions are

$$K_y(T_{Ly} s + 1) \tag{3.8}$$

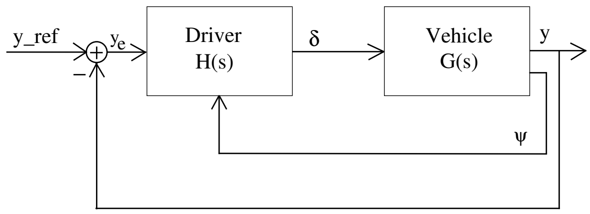

Figure 3.3: Basic structure of multi-input model

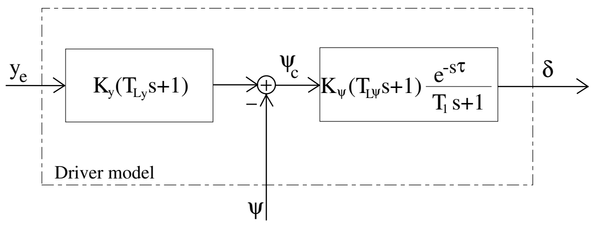

Figure 3.4: Multi-input driver model

and

$$K_\psi(T_{L\psi}s+1)\frac{e^{-\tau s}}{T_1 s+1} \tag{3.9}$$

The outer loop, as seen in figure 3.3, feeds back the lateral displacement and thus makes the vehicle follow the desired path, and the inner loop feeding back the yaw rate is necessary to give sufficient damping of the closed-loop system. This approach gives in practice four parameters to work with: K_y, T_{Ly}, K_ψ and $T_{L\psi}$. The gain parameters, K_y and K_ψ represents the proportional action of the driver with respect to lateral error and yaw angle, respectively. The $(T_L s + 1)$ factors are modelling the lead or predictive action, meaning that the driver controls the vehicle by predicting future values, also known as preview in the previous section. The last two parameters, τ, and T_1 should be kept constant, representing dead time and the delay due to the muscular system, respectively. The sum of both the lead time constants $T_{Ly}+T_{L\psi}$ can also be taken as a measure of the driver´s mental workload, large sum denoting high mental workload and vice versa. In figure 3.5 to 3.10, the influence of the different parameters on performance in a single lane-change can be seen.

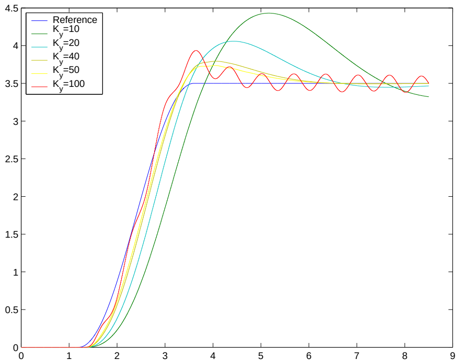

Figure 3.5: Influence of proportional gain, K_y

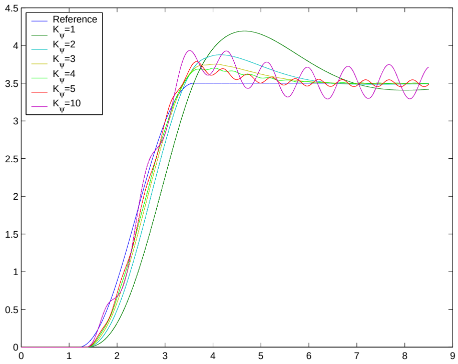

Figure 3.6: Influence of proportional gain, K_ψ

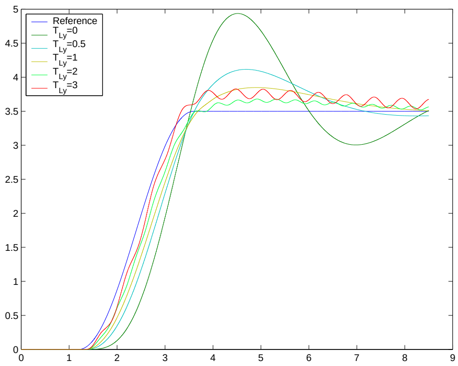

Figure 3.7: Influence of driver lead time, T_{Ly}

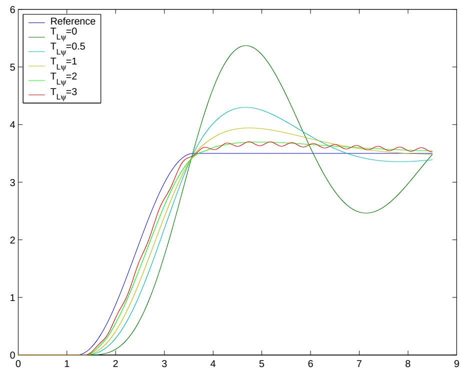

Figure 3.8: Influence of driver lead time, $T_{L\psi}$

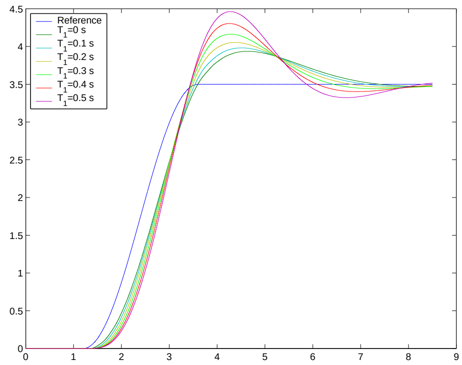

Figure 3.9: Influence of driver lag time, T_1

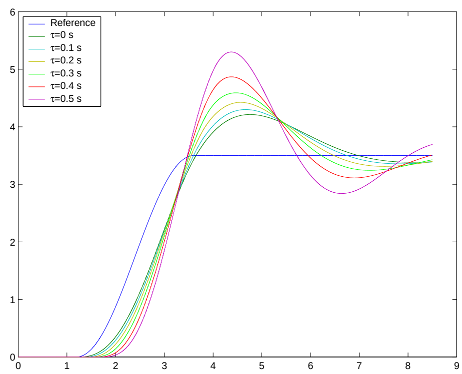

Figure 3.10: Influence of driver reaction time, τ

3.2 Handling Qualities

The handling qualities of the vehicle can be defined as the aspects which affect ease and accuracy when performing a certain task. These can be subdivided into two groups:

- Task Performance
- Driver Workload

The task comprises in general cornering, lane keeping, lane changing, driving a certain distance, etc. It is known that a driver can compensate for somewhat decreased vehicle performance, and therefore no difference in task performance would appear, within certain limits. The driver´s workload can be further subdivided into physical and mental workload, where the former corresponds to the amount of physical work the driver has to do and the latter is containing factors such as stress, fatigue, etc, but also the task of keeping the vehicle stable and within secure distances from the surrounding vehicles. For this work, the goal is to minimise the mental and physical workload while keeping the performance at acceptable levels.

3.2.1 Task Performance

This work has been considering lateral dynamics only, so as a measurement of the task performance, the lateral deviation from a desired path was chosen:

$$J_1 = \int_0^t (y_{ref} - y)^2 dt \tag{3.10}$$

where y is the actual position of the vehicle´s CoG. The driver´s proportional constants, K_y and K_ψ have a significant effect on the performance, J_1.

3.2.2 Physical Workload

In [6], the physical workload of the driver is considered to be small if he or she can perform a certain task by keeping the steering wheel angle, δ_{LR}, small. This gives the following measurement of physical workload:

$$J_2 = \int_0^t \delta_{LR}^2 dt \tag{3.11}$$

Another approach is considering the necessary force required from the driver to complete a certain task. The general torque equation is:

$$\alpha I = \sum_{i=0}^{n} \tau(n) \tag{3.12}$$

For the steering wheel:

$$\ddot{\delta}_{LR} I = M_L + M_D \tag{3.13}$$

where M_L is the feedback torque, the driver feels from the steering wheel and M_D is the required torque from the driver. I is the moment of inertia of the steering wheel. If the steering wheel is considered as a rotating cylinder, the moment of inertia can be calculated as:

$$I = \frac{mr^2}{2} \tag{3.14}$$

where m is the mass and r is the steering wheel cylinder radius. The required torque from the driver can be expressed as:

$$M_D = Fr \tag{3.15}$$

Together with 3.13 and 3.14 this gives the necessary force, F:

$$F = \frac{mr\ddot{\delta}}{2} - \frac{M_L}{r} \tag{3.16}$$

which has been used alternating with the δ_{LR} as integrand in 3.11.

3.2.3 Mental Workload

Much research has been done on the mental workload of the driver, see [3] and [10]. Vehicle driving is a dynamic control activity in a continuously changing environment, affected not only by the drivers themselves, but also by the behaviour of other traffic participants. If the mental workload exceeds the capacity of the driver, this may result in affected performance, e.g. a beginner driver cannot perform all control tasks automatically, and workload with respect to vehicle control is high. In a new traffic environment, e.g. driving in heavy traffic in an unknown city, manoeuvre tasks may put high demands on visual and central resources, leading to affected performance. Sources of driver mental workload may be found both inside and outside the vehicle and since driving is to a very large extent a visual task, demands on visual and central resources will be highest. It is still impossible to find a general equation which describes the mental workload of the driver, but much qualitative research has been done, e.g. examining the differences between alert and fatigued drivers, from a medical and/or physiological point of view. This thesis is only concerning the lateral movements, so to describe the drivers mental workload, an assessment of the steering actions is appropriate. In general, the mental workload of a driver increases with his or her derivative actions, see [6], and as noted in section 3.1.3, the sum of the driver´s lead time constants can be used as a measurement of the mental workload:

$$J_3 = T_{Ly} + T_{L\psi} \tag{3.17}$$

Another approach is to directly measure the angular velocity of the steering wheel, thereby measuring the derivative action of the driver. This gives a mental workload definition as:

$$J_3 = \int_0^t \dot{\delta}_{LR} dt \tag{3.18}$$

Both 3.17 and 3.18 have been used alternately in this project.

Chapter 4

Optimisation

4.1 Introduction

Optimisation theory is a branch within applied mathematics, which contains the usage of mathematical models to find the best possible solution to a certain problem. Examples of optimisation problems can be production planning, schedule planning, profit maximisation, structural optimisation, etc. The general mathematical structure of a optimisation problem is:

$$\begin{aligned} &min\, f(\mathbf{x}) \\ &when\; \mathbf{x} \in X \end{aligned}$$

where $f(\mathbf{x})$ is the cost function, depending on the variables $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$. The set X contains the permitted solutions to the problem. Usually X is expressed with conditional equations $g_1(\mathbf{x}), \ldots, g_m(\mathbf{x})$, which gives the alternate general structure:

$$\begin{aligned} &min\, f(\mathbf{x}) \\ &when\; g_i(\mathbf{x}) \leq b_i \quad i = 1, \ldots, m \end{aligned}$$

where $b_i, \ldots, b_m$ are constant values. The solution $\mathbf{x} \in X$, which minimises $f(\mathbf{x})$ is called the optimal solution or optimum. Several optimisation algorithms exist, e.g. linear programming with the simplex method or the Frank-Wolfe method for solving nonlinear problems. For this work, another branch of algorithms has been used, namely the Genetic Algorithms. How they work will be explained in more detail in the following sections.

4.2 Evolutionary and Genetic Algorithms

Evolutionary algorithms are a set of optimisation methods which attempt to solve optimisation problems with methods from the Darwinian principles of reproduction and survival of the fittest. Evolutionary algorithms model natural processes such as selection and mutation, discarding bad results and trying to find better candidates in the neighbourhood of a promising result. In [9], *genetic algorithms* are introduced as an algorithm which tries to find a good solution to an optimisation problem by genetically breeding a set, *population*, of candidate solutions, *individuals*, which are then transformed into a new generation using *reproduction, selection and mutation.* If the optimisation criteria are not met, the algorithm starts calculating a new generation. The indiviuals are ranked and the best are selected for production of *offspring*. Parents are recombined and the offspring are mutated with a certain probability. The offspring then replace their parents and are inserted into the population, creating a new generation. This procedure is repeated until the optimisation criteria are reached; see figure 4.1. The main difference between evolutionary algorithms and genetic algorithms is that the genetic algorithms model the sexual behavior of reproduction, with mating parents, while the evolutionary algorithms are asexual, using only mutation and selection to model the evolution. When using multiple *subpopulations,* each population evolves over a

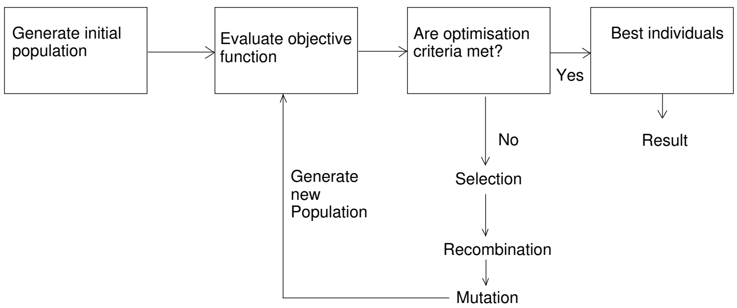

Figure 4.1: Evolutionary Algorithm Structure

few generations before one or more individuals *migrate* between the subpopulations. In figure 4.2 the general algorithm is explained. In the following sections, the procedure of an genetic algorithm will be explained in more detail.

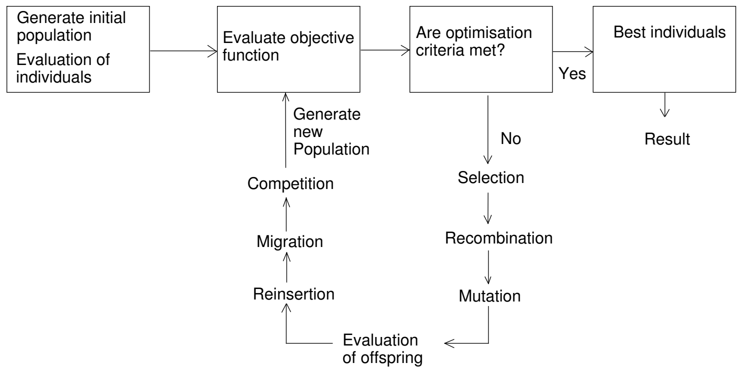

Figure 4.2: Evolutionary algorithms structure with multiple populations

4.2.1 Selection

The selection process chooses which individuals in a population are to reproduce and create offspring. The first step in that process is the *fitness assignment,* whereby each population member gets a probability for reproduction, depending on its objective value and the objective value of all the other individuals. Many different algorithms have been developed for producing fitness assignment, for example *rank-based, roulette-wheel and local selection.* In rank-based fitness assignment, the individuals are sorted according to their respective position in terms of objective value. This solves the stagnation problem whereby a premature converge can occur. In roulette-wheel selection, the individuals are chosen for breeding in a random manner, but the chance of being chosen is proportional to the fitness of the individual. Finally, local selection introduces the *neighbourhood*, where each individual interacts only with other individuals inside this area. The neighbourhood can be interpreted as the obtainable mating partners for a certain individual. The selection works in two steps; first one half of the population is chosen at random and then a local neighbourhood is selected for each chosen individual. The structure of the neighbourhood can be linear, two-dimensional or three-dimensional, or more complex with combinations of these. Then for each individual a mating partner is selected within the neighbourhood according to some rules or at random.

4.2.2 Recombination

Recombination is the process which mixes the information from the parents and thereby produces new individuals. The information can be transferred in several different ways, depending on how the information is stored in the parental individuals. Some methods, eg *intermediate recombination* can only be used on real valued variables, while *discrete recombination* and *binary valued recombination* can be used on all types of variables. In discrete recombination an exchange of values between the parents takes place, randomly choosing which parent will give its value to the offspring. In intermediate recombination the offspring variable gets its values from those which are between the parents, following the rule:

$$var_i^0 = var_i^{P_1} a_i + var_i^{P_2}(1 - a_i) \qquad i \in 1, 2, \ldots, Nvar \tag{4.1}$$

$a_i \in [-d, 1 + d]$ uniform at random, $d = 0.25, a_i$ for each i new

where a is a scaling factor chosen randomly over the interval $[-d, 1+d]$ for each variable anew. The parameter d defines the region allowed for possible offspring, with $d = 0$ defining the area allowed as the same as that of the parents. This can have the drawback of a shrinking area, because most offspring will be created in the center of the area and not on the borders. A larger value of d will prevent this, with $d = 0.25$ ensuring that the offspring will (statistically) span the area of the parents. The binary valued recombination is similar to discrete recombination, but mostly working on binary variables.

4.2.3 Mutation

In mutation, the values of certain variables are varied randomly. These variations are normally small and will be applied to the individual variable after recombination with a low probability, the *mutation rate*. The probability of mutation is inversely proportional to the number of variables within each individual, i.e. the more dimensions one individual has, the smaller the probability of mutation. The mutation step-size is difficult to choose, the optimal step-size depends on the optimisation problem and it may even vary during the optimisation process. A small step-size is usually preferred when the individual is already well adapted, while a larger step-size can often produce good results much faster, if successful. A mixture of step-sizes in the mutation process producing small steps with high probability and large steps with low probability is often the best mutation operator.

4.2.4 Reinsertion

When a new set of offspring has been created, it must be reinserted into the population, to make the new generation. If less offspring are produced than the original population all should be inserted. Similarly, if more offspring than needed are generated, a *reinsertion scheme* must be used to select which offspring are to exist in the new generation. Some global reinsertion schemes are:

- pure reinsertion - produce as many offspring as parents, replacing all parents with their offspring
- elitist reinsertion - produce less offspring than parents and replace the worst parents
- fitness-based reinsertion - produce more offspring than parents, and reinsert only the best offspring

Pure reinsertion is the simplest scheme, where every individual lives only one generation. The major drawback is that good individuals are likely to be replaced by worse offspring, thus losing good information. This is prevented by using elitist and/or fitness-based recombination, which allows the good individuals to live for many generations.

4.2.5 Multiple subpopulations

Multiple subpopulations, see figure 4.2, is a model which also incorporates the *migration* between several subpopulations, thus creating a regional model. The subpopulations evolve independently for a number of generations, the *isolation time,* after which a few individuals are exchanged between the subpopulations, *migration.* The migrating individuals can be selected randomly or according to fitness-based reasoning, then the best individuals migrate. There are also many possible migration structures, for example neighbourhood or unrestricted migration, determining the range of migration.

4.3 Optimisation with Pigeno

Pigeno or **P**arametric **i**dentification using **gen**etic **o**ptimisation is described in [4] and the general structure can be seen in figure 4.3. Pigeno is a Matlab/Simulink tool developed by DaimlerChrysler AG to identify parameters, which optimises a certain *cost function* using the genetic algorithm described in section 4.2, or to match measurement data. Pigeno can also be used to find a good parameter set in a model, by adjusting the cost function to describe the optimisation problem.

4.4 Cost Function

For this work, the cost function has to show how the driver rates the vehicle, as described in section 3.2. The cost function was chosen as:

$$J = q_1 J_1 + q_2 J_2 + q_3 J_3 + q_4 J_4 \tag{4.2}$$

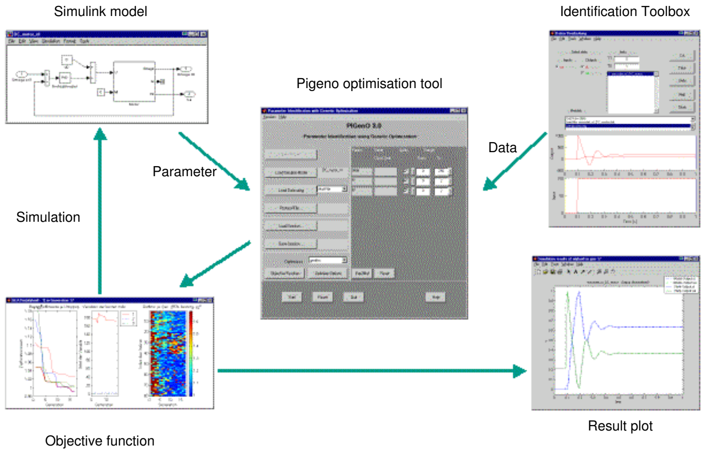

Figure 4.3: Pigeno Structure

where q_i denotes the weighting factors of the respective costs, J_i. J_1, J_2 and J_3 represent performance, physical workload and mental workload respectively, as described in previous sections. J_4 was introduced to keep all penalties, e.g. penalty for humanly impossible manoeuvres or knocking down cones in the ISO-lanechange, so that unacceptable results died out in the genetic algorithm.

Chapter 5

Simulation results

A framework has been implemented in Matlab/SIMULINK, see appendix A.3. It has been built by modules and is therefore easy to expand or change. Different reference trajectories can be chosen, and the selection of driver models is also possible. The cost function for the optimisation algorithm can be fully customised, and initial parameter settings can be loaded from m.files. The manoeuvre which has been studied in detail is the ISO-TR 3888 double lane-change, see appendix A.1. Three different approaches have been made in the optimisations:

- Which driver parameter set minimises the cost function, with respect to performance, mental and physical workload for a fixed trajectory and vehicle model.

- Which vehicle parameter set minimises the cost function, with respect to performance, mental and physical workload for a fixed trajectory and driver model.

- Which trajectory minimises the cost function, with respect to performance, mental and physical workload for a fixed driver and vehicle model.

Due to the fact that a driver adjusts his or her driving behaviour dynamically, optimisations were made on both driver and vehicle parameters at the same time, simulating the adaption process. Also, it was found that the optimal path varied with the driver and vehicle, and therefore the trajectory was varied at the same time as the other parameters. The simulations have mainly had a longitudinal velocity of $v_x = 60\,km/h$, but driver model parameters have been found, which complete the lane-change manoeuvre with up to $v_x = 100\,km/h$. Two different driving strategies have been studied:

- The first driver represent a beginner driver, who tries to stay in the middle of the track, thus keeping the maximum distance to the nearest cones, see figure 5.1.
- The second driver is more experienced and tries to follow a path which minimises his or her distance, cutting corners etc, see figure 5.2.

The initial parameter set of the VFD model represented a limousine-type car, and for the comparison of driver adaption, parameter sets representing a sports car and a road cruiser were used, together with the limousine parameter set.

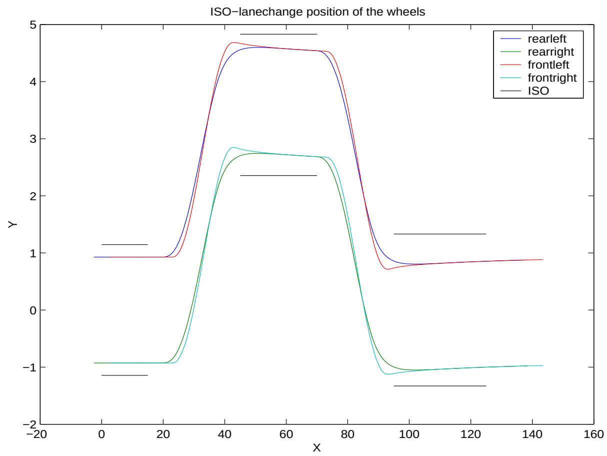

Figure 5.1: The wheels of the vehicle as the driver takes a middle path in the ISO lane-change

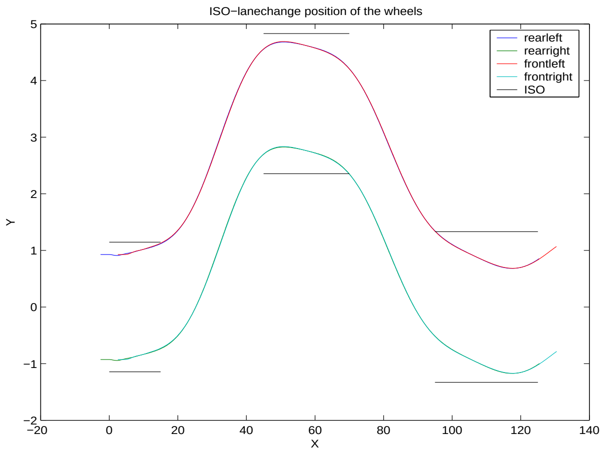

Figure 5.2: The wheel positions in a smoother ISO lane-change manoeuvre

5.1 Beginner driver results

As seen in figure 5.1, the first driver tries to stay in the middle of the track, with maximum possible distance to the cones. He or she makes the lane-change very quickly, which gives high lateral acceleration. This behaviour can be taken as typical beginner behaviour, when the driver feels insecure about how to handle the vehicle. The optimisation results can be seen in table 5.1. The optimal parameter set for performing this task includes a very fast lateral acceleration response, high ω_0, with low damping, ξ. This however, would result in an uncontrollable vehicle, because such low damping would result in a self-oscillating vehicle, and therefore the driver would have to compensate for this with the steering wheel. This can be seen in the driver parameters, where the lead constant, $T_{L\psi}$, is very high, which implies that the driver must pay much attention to the yaw angle. In diagram 5.4, the most interesting parameters are compared with the original VFD parameters. As can be seen, the damping, ξ, is much higher when optimising on mental workload than on the other optimisations. This indicates that too little damping results in a high mental workload. Also noticeable is that the optimal steering wheel ratio, i_l, and the optimal under steer gradient, elg, are very small. If we recall the lateral dynamics from the vehicle model

$$a_y = v(\dot{\psi} - \dot{\beta}) = v(\frac{T_z s + 1}{\frac{s^2}{\omega_0^2} + \frac{2\xi s}{\omega_0} + 1} \frac{v}{i_l(l + elg\, v^2)} \delta_{LR} - \dot{\beta}) \tag{5.1}$$

and insert the driver model transfer function

$$\delta_{LR} = K_\psi (T_{L\psi} s + 1) \frac{e^{-\tau s}}{T_1 s + 1} (K_y (T_{Ly} s + 1) y_e - \psi) \tag{5.2}$$

we get the open-loop transfer function from lateral position to lateral acceleration:

$$\begin{aligned} a_y = v(&\frac{T_z s + 1}{\frac{s^2}{\omega_0^2} + \frac{2\xi s}{\omega_0} + 1} \frac{v}{i_l(l + elg\, v^2)} K_\psi (T_{L\psi} s + 1) \cdot \\ &\frac{e^{-\tau s}}{T_1 s + 1} (K_y (T_{Ly} s + 1) y_e - \psi) - \dot{\beta}) \end{aligned} \tag{5.3}$$

This work has only concentrated on the yaw dynamics, and therefore the influence of body side slip has not been considered, since only one actuator is used. From this, we can draw the conclusion that a rapid lateral acceleration response on the steering wheel angle is the optimum, which is what we get with a small steering wheel ratio and under steer gradient. Another result is the driver adaption to the vehicle. The driver model parameters vary quite a lot with the different optimisation goals, for example when optimising on mental workload the driver pays more attention to the yaw angle. This can be seen in the parameter K_ψ, which is big compared to the other optimisations.

Parameter	Original	Performance	Physical work	Mental work
K_ψ		1.0967	0.6509	1.9382
K_y		1.2337	3.0387	2.7915
$T_{L\psi}$		2.3255	2.6488	2.3219
T_{Ly}		0.0623	0.0879	0.2261
elg	0.30	0.0028	0.0065	0.0519
i_l	15.43	0.7990	4.0738	8.6844
ω_0	10	43.6103	29.4635	25.1321
ξ	1.50	0.0880	0.0071	0.9545
c_{MZ}	0.20	0.0810	0.0825	0.2008
d_{ML}	0.015	0.0128	0.0111	0.0094
lmg	1.00	0.8203	0.9153	0.5718
M_{LReib}	0.30	1.2611	1.1776	1.2399

Table 5.1: Driver and vehicle parameters, optimisation results

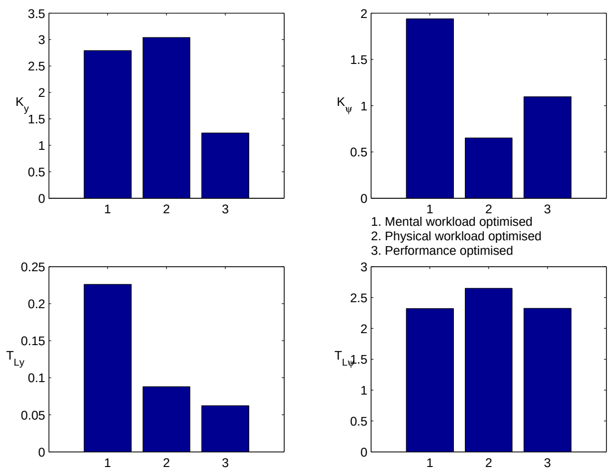

Figure 5.3: Optimised driver model parameters

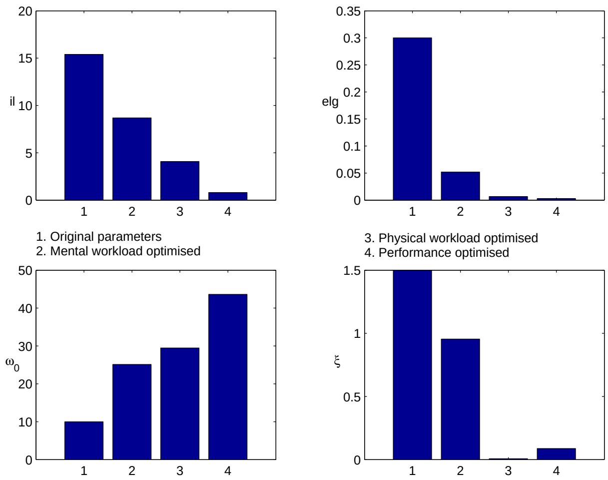

Figure 5.4: Vehicle parameters, beginner trajectory

5.2 Experienced driver results

The second driver strategy was implemented as a seventh order polynome, representing the experienced driver, who knows the vehicle and therefore tries to cut corners, to give a more smooth path and thus minimise the lateral acceleration. Two different approaches were tried

- Which are the optimal VFD parameters, for a certain driver, i.e. a driver model with fixed parameters.
- If the driver adapts, which are the optimal parameters in the driver model.

If the driver adapts to the vehicle, then the driver model parameters K_y, T_{Ly}, K_ψ and $T_{L\psi}$ can be chosen freely, while the neuromuscular response delays T_1 and τ should be kept constant.

5.2.1 Fixed driver model

To find the parameters for the driver model and the trajectory, the genetic algorithms were used to find a suitable combination, to pass the ISO lane-change with acceptable performance, whilst having low mental and physical workload, i.e. a good balance between task and workload. The time constants T_1 and τ were chosen to represent an alert driver. The resulting driver parameters can be seen in table 5.2, and the resulting VFD parameters can be seen in table 5.3. In diagram 5.5, the optimised parameters can be seen compared to the original VFD parameters, and in diagram 5.6, the cost functions are compared. Here the resulting parameters are more close to the original VFD parameters, with the exception of the eigenfrequency, ω_0, and the damping, ξ, which varies quite a lot with the different optimisation goals. This is an indication that if the eigenfrequency and the damping could be controlled, the workload of the driver could be decreased and the performance could be increased. As can be seen, the steering wheel ratio i_l and the time constant T_z are almost the same in all results, thus indicating that for a specific driver, the steering wheel ratio could be kept constant regardless of optimisation goals.

Parameter	Value
K_ψ	3.2001
K_y	39.1328
$T_{L\psi}$	0
T_{Ly}	0.32117
T_1	0.1
τ	0.1

Table 5.2: Driver parameters

Costfcn	Original	Performance	Force	Steering angle	Mental work
Parameter					
i_l	15.43	15.6724	15.672	15.671	15.669
ω_0	10	3.87572	15.14	5.3608	11.752
ξ	1.50	1.33179	1.6284	2.0647	1.2812
T_z	0.10	0.105275	0.10526	0.10525	0.10522

Table 5.3: Vehicle parameters, optimisation results

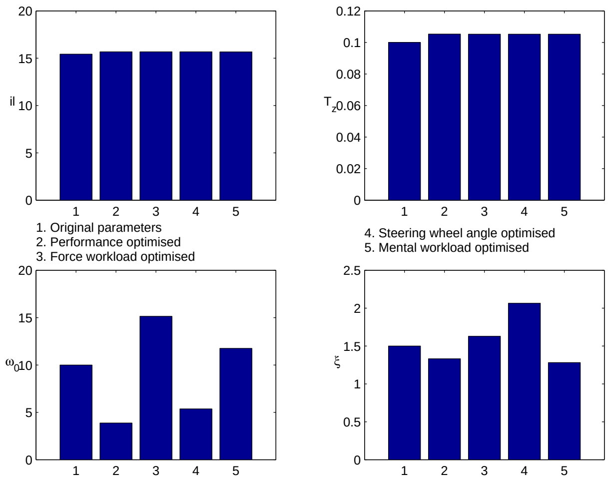

Figure 5.5: Vehicle parameters, optimised

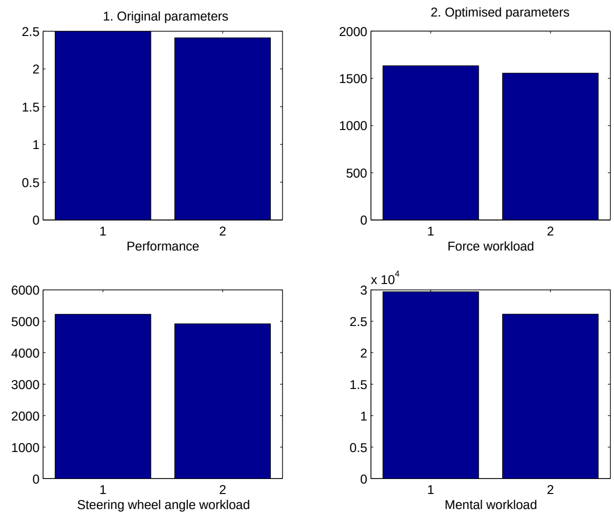

Figure 5.6: Comparison of cost functions

5.2.2 Driver adaption

It is known that a driver can adapt to different vehicles, and thus compensate for differences in behaviour. To test the driver model adaption, three different vehicle parameter sets were used:

- A limousine
- An extremely sporty vehicle
- A road cruiser

These parameter sets represent three different vehicle types, the limousine is a comfortable vehicle, the sporty vehicle has a more direct steering response, i.e. low i_l, etc, and the road cruiser is relatively slow in steering response. The driver model parameters were optimised on performance, i.e. trajectory- following, during the ISO lane-change manoeuvre. The results can be seen in table 5.4, together with the distinguishing vehicle parameters. Here the results indicate that the sports car demands the least proportional action from the driver, while the other models demand more, while performing this task. This is a result from the lower steering wheel ratio and the quicker lateral acceleration response of the sports car.

Parameter	Limousine	Sporty	Road cruiser
K_ψ	3.2643	2.9704	3.7878
K_y	42.6064	28.6082	46.4463
$T_{L\psi}$	0.2924	0.0054	0.1389
T_{Ly}	0.0183	0.2394	0.2442
elg	0.30	0.30	0.60
i_l	15.40	12.00	17.00
ω_0	10.00	10.00	8.00
ξ	1.50	1.00	1.00
c_{MZ}	0.20	0.20	0.10
d_{ML}	0.015	0.015	0.0075
lmg	1.00	1.20	0.50

Table 5.4: Driver model optimisation results, to see adaption

K_ψ	i_l	$\frac{K_\psi}{i_l}$
1.0967	0.7990	1.373
0.6509	4.0738	0.160
1.9382	8.6844	0.223
3.2001	15.43	0.207
3.2001	15.67	0.204
3.2643	15.40	0.212
2.9704	12.00	0.248
3.7878	17.00	0.223

Table 5.5: Ratio between K_ψ and i_l

With the exception of the performance optimisation with the beginner driver, a ratio between K_ψ and i_l of about 1:5 seems to be the optimum, see table 5.5. One reason for the abnormal result of the first optimisation could be the extremely low steering wheel ratio, and the high eigenfrequency and low damping, which compensate the transfer function of lateral acceleration and thus keep that constant. This result could be used to tune a driver model to a vehicle model, or to tune a vehicle steering wheel ratio to a driver, if the driver parameters could be identified.

Chapter 6

Conclusions and future work

6.1 Conclusions

The closed-loop driver-vehicle-environment has been implemented in matlab/simulink. The driver adapts to the vehicle and it is therefore impossible to describe the driver with one general equation. To describe the lateral actions of the driver, a model working with the position error and the heading angle was implemented, and has proven sufficient for the ISO lane-change. The optimisation tool has proven useful to find the parameters of the driver model, to adjust them to a certain vehicle reference model or for the opposite function; to find vehicle parameters to optimise certain criteria, e.g. mental workload. In the lane change task, a fast lateral acceleration response seems to improve the handling qualities of the vehicle. For a fixed driver, the steering wheel ratio, i_l, should be fixed, and a ratio between K_ψ and i_l of 1:5 has proven to be optimal. The optimum in lateral acceleration includes an eigenfrequency, ω_0, between 5 and 15, and a damping, ξ, between 1 and 2.

6.2 Future work

This work can be continued in several different ways, e.g. the simulation environment developed does only take into account lateral dynamics. Only the yaw dynamics have been taken into account in this thesis work, therefore as future work, the side-slip should be examined and also the complete VFD model with yaw-rate controller, full vehicle dynamics, etc. The coupling between lateral and longitudinal dynamics would be interesting to examine, with an extended driver model, capable not only of steering, but also acceleration and braking. In addition, a more realistic environment could be implemented, to examine external disturbances, like wind and different friction coefficients, μ, on the tyres. This work has mostly studied the ISO lane-change manoeuvre, which has a very short duration, so it could be interesting to build a model of a full test track for example, to simulate endurance driving. The results obtained here are going to be tested in a real research vehicle, to hopefully be verified by human drivers.

References

[1] A. Apel and M. Mitschke. Adjusting vehicle charachteristics by means of driver models. *Int. J. of Vehicle Design*, 18(6):583–596, 1997.

[2] F. Böttiger. VFD Synthetische Modelle. 7 1999.

[3] D. de Waard. *The measurement of driver´s mental workload*. PhD thesis, Traffic Research Centre, University of Groningen, University of Groningen, The Netherlands, 1996.

[4] Dennis Giesa and Stefan Lachmann. *Dokumentation zur Parameter Identification with Genetic Optimization(PIGenO for SIMULINK)*. DaimlerChrysler AG.

[5] K. Guo and H. Guan. Modelling of driver/vehicle directional control system. *Vehicle System Dynamics*, 22:141–184, 1993.

[6] Shinichiro Horiuchi and Naohiro Yuhara. An analytical approach to the prediction of handling qualities of vehicles with advanced steering control system using multi-input driver model. *Journal of Dynamic Systems, Measurement, and Control*, 122:490–497, September 2000.

[7] U. Kiencke and L. Nielsen. *Automotive Control Systems*. Springer-Verlag, Berlin, Germany, 2000.

[8] Lars Klumparend. Variation von fahrverhaltensparametern unter besonderer berücksichigung des lenkradmoments. Diplomarbeit, Fachhochschule Aalen Fachbereich Maschinenbau, August 1999.

[9] John R. Koza. Genetic programming. In James G. Williams and Allen Kent, editors, *Encyclopedia of Computer Science and Technology*, volume 39, pages 29–43. Marcel-Dekker, 1998.

[10] B. Messick Huey and D. Wickens. *Workload Transition*. National Academy Press, Washington DC, USA, 1993.

[11] A. Murray-Smith, R. Johansen and D. Murray-Smith. Cooperative human and automatic control, 1997.

[12] Andrzej Renski. Identification of driver model parameters. *International Journal of Occupational Safety and Ergonomics*, 7(1):79–92, 2001.

Notation

Variables and parameters

ψ	Yaw angle
$\dot{\psi}$	Yaw rate
$\ddot{\psi}$	Yaw acceleration
β	Body side-slip angle
v	Vehicle velocity
v_x	Longitudinal velocity
v_y	Lateral velocity
a_y	Lateral acceleration
b	Vehicle width
l	Wheel base
l_v	Distance from CoG to front axle
l_h	Distance from CoG to rear axle
I_z	Moment of Inertia around z-axis
m	Vehicle mass
ω	Eigenfrequency
ξ	Natural damping
elg	Under steer gradient
swg	Body side-slip gradient
wwg	Roll angle gradient
c_v	Cornering stiffness front wheel
c_h	Cornering stiffness rear wheel
δ_v	Steering angle front wheel
δ_h	Steering angle rear wheel
δ_{LR}	Steering wheel angle
α_v	Front tyre side-slip angle
α_h	Rear tyre side-slip angle
i_l	Steering wheel ratio
c_{MZ}	Steering wheel centering stiffness
c_{ML}	Steering wheel stiffness
d_{ML}	Steering wheel damping
lmg	Steering torque gradient

Abbreviations

ABC	Active Body Control
ABS	Anti-lock Braking System
CoG	Centre of gravity
ISO	International Organization for Standardization
TRC	Traction Control
VFD	Variabel Fahrzeug Dynamik (Variable Vehicle Dynamics)

Appendix A

A.1 ISO-TR 3888

Description of the ISO-TR 3888 lane-change manoeuvre, see also [8].

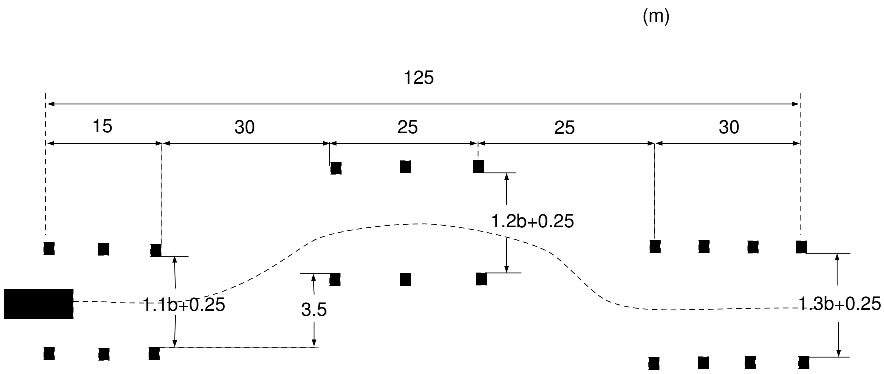

Figure A.1: Sketch over ISO lane-change track

A.2 Penalty function

As noted earlier, a penalty part of the cost function was used to keep all penalties, thereby limiting the survival of a bad solution of the optimisation algorithm. The different limits are:

- Max steering wheel angle, δ_{LRMax}, ensuring that the driver does not turn the steering wheel more than physically possible.
- Max steering wheel angular velocity, $\dot{\delta}_{LRMax}$, due to the limited capacity of the human muscular system.
- Max lateral acceleration, the reference model is not valid for high lateral accelerations.
- Passing or failing the ISO lane-change.

The first three limits were implemented as simple simulink switches, returning 0 if pass, and 1 if fail. The function for checking the ISO lane-change passing is more complex, and for performance reasons was implemented as an S-function, written in C. The code can be seen in figure A.2.

```
#define S_FUNCTION_NAME  lsolanechange2
#define S_FUNCTION_LEVEL 2

#include "simstruc.h"
#include <math.h>

/*================*
 * Build checking *
 *================*/


/* Function: mdlInitializeSizes ===============================================
 * Abstract:
 *   Setup sizes of the various vectors.
 */
static void mdlInitializeSizes(SimStruct *S)
{
    ssSetNumSFcnParams(S, 0);
    if (ssGetNumSFcnParams(S) != ssGetSFcnParamsCount(S)) {
        return; /* Parameter mismatch will be reported by Simulink */
    }

    if (!ssSetNumInputPorts(S, 1)) return;
    ssSetInputPortWidth(S, 0, DYNAMICALLY_SIZED);
    ssSetInputPortDirectFeedThrough(S, 0, 1);

    if (!ssSetNumOutputPorts(S,1)) return;
    ssSetOutputPortWidth(S, 0, 1);/*DYNAMICALLY_SIZED*/

    ssSetNumSampleTimes(S, 1);

    /* Take care when specifying exception free code - see sfuntmpl_doc.c */
    ssSetOptions(S, SS_OPTION_EXCEPTION_FREE_CODE |
             SS_OPTION_USE_TLC_WITH_ACCELERATOR);
}


/* Function: mdlInitializeSampleTimes =========================================
 * Abstract:
 *    Specifiy that we inherit our sample time from the driving block.
 */
static void mdlInitializeSampleTimes(SimStruct *S)
{
    ssSetSampleTime(S, 0, INHERITED_SAMPLE_TIME);
    ssSetOffsetTime(S, 0, 0.0);
}


static int checkpoint(double x, double y, double b)
{

    if ((0.0<=x) && (x<=15.0))/*point in first corridor?*/
        {
        if (((-(1.1*b+0.25)/2) <= y) && (y < ((1.1*b+0.25)/2))) /*correct pass*/
            return 0;
        else return 1;
        }
    if ((45.0<=x) && (x<=70.0))/* point in second corridor?*/
        {
        if (((3.5-(1.1*b+0.25)/2) <= y) && (y< (3.5+(1.2*b+0.25)-(1.1*b+0.25)/2)))
            return 0;
        else return 1;
        }
    if ((95.0<=x) && (x<=125.0))/*point in third corridor?*/
        {
        if ((-((1.3*b+0.25)/2) <= y) && (y< ((1.3*b+0.25)/2)))
            return 0;
        else return 1;
        }
    else
        return 0;/* point ok, out of corridor*/
}
```

```
/* Function: mdlOutputs =======================================================
 * Abstract:
 *    calculates the cost function for crossing the boundaries of a lanechange
 */
static void mdlOutputs(SimStruct *S, int_T tid)
{
    /*int_T            i;*/
    InputRealPtrsType uPtrs = ssGetInputPortRealSignalPtrs(S,0);
    real_T           *penalty   = ssGetOutputPortRealSignal(S,0);
    int_T            width = ssGetOutputPortWidth(S,0);
    double          x,y,psi,cf,cle,cri,cr,b;
    x=*uPtrs[0];
    y=*uPtrs[1];
    psi=*uPtrs[2];
    cf=*uPtrs[3];
    cr=*uPtrs[4];
    cle=*uPtrs[5];
    cri=*uPtrs[6];
    b=(cri+cle);

    *penalty++ = 1000000000.0*(checkpoint(x,y,b)
    +checkpoint((x+cf*cos(psi)-cle*sin(psi)),(y+cf*sin(psi)+cle*cos(psi)),b)/*front left corner*/
    +checkpoint((x+cf*cos(psi)+cri*sin(psi)),(y+cf*sin(psi)-cri*cos(psi)),b)/*front right corner*/
    +checkpoint((x-cr*cos(psi)-cle*sin(psi)),(y-cr*sin(psi)+cle*cos(psi)),b)/*rear left corner*/
    +checkpoint((x-cr*cos(psi)+cri*sin(psi)),(y-cr*sin(psi)-cri*cos(psi)),b));/*rear right corner*/
}


/* Function: mdlTerminate =====================================================
 * Abstract:
 *    No termination needed, but we are required to have this routine.
 */
static void mdlTerminate(SimStruct *S)
{
}



#ifdef  MATLAB_MEX_FILE    /* Is this file being compiled as a MEX-file? */
#include "simulink.c"      /* MEX-file interface mechanism */
#else
#include "cg_sfun.h"       /* Code generation registration function */
#endif
```

Figure A.2: C-code

A.3 Simulink Model

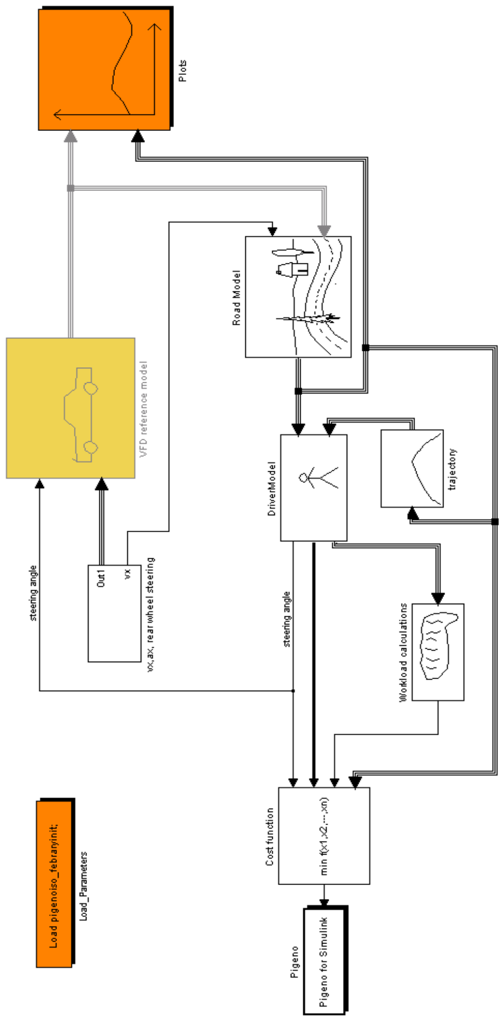

Figure A.3: Simulink model, top level