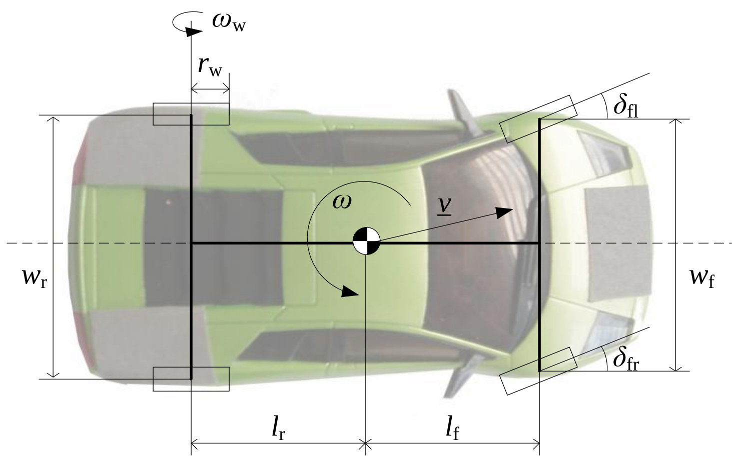
Semester Thesis
March 2010 - June 2010

Modeling of 1:43 scale race cars



Authors: Patrick Spengler & Christoph Gammeter

Supervisor: Sean Summers, Dr. Colin Jones

Professor: Prof. Dr. Morari

Abstract

In this thesis two models for the Kyosho d-Nano RC cars have been developed and a test track for future controlling algorithms has been created. In a first step models were derived from literature. In a second step a measurement setup was built, and in a third step the theoretical models were analyzed, validated and their parameters were fitted, which resulted in two proposed models.

Acknowledgment

This report is the result of the semester thesis 'Modeling of 1:43 scale race cars' written at the Automatic Control Laboratory.

We want to express our thanks to Sean Summers, our supervisor, who always helped and encouraged us to find new ways to solve our problems. And a special thanks to Martin Rutschmann, who provided us with a vision system and helped us with numerous implementations. Without them we would have never been able to complete our thesis.

Zürich, august 4 2010

Patrick Spengler Christoph Gammeter

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für Automatik

SA, FS 08	Issue Date:	08 March 2010
	Termination Date:	13 August 2010

Title of the project

Real-time control of 1:43 scale race cars

Author C.N. Jones, D. Raimondo and S. Summers jones@control.ee.ethz.ch

Description

In this project you will develop an autonomous RC-car racetrack in order to study high-speed, real-time control. Mini 1:43 scale RC cars have just come into their own with the release of the Kyosho dNano in 2008, which can drive on a desktop but moves at extreme pace. You will develop the hardware and software for a race track in which up to 40 of these cars can compete simultaneously, as well as the controllers to enable them to win.

In this project you will investigate the application of real-time MPC techniques to high-speed remote control cars. These RC cars can achieve scale-speeds of several hundred kilometers per hour (5 m/s) and are challenging for a human to control, let alone a computer in a dynamic and changing race environment. The goal of the project is to build a system in which humans and computers can race head-to-head - the best algorithm wins!

This project summary encompasses five semester projects and one master project that will together develop a system capable of optimally racing several autonomous cars simultaneously while avoiding human opponents.

Project : Modeling and trajectory tracking	
Christoph Gammeter and Patrick Spengler	
Primary Supervisor : Sean Summers	
Goal	Develop validated model of the cars and simple controller to track a given path.

- Review literature on modeling of cars
- Review literature on nonlinear system identification
- Develop experiments to identify / measure car parameters
- Fit model to recorded data and validate
- Estimate uncertainty in developed model
- Review literature on trajectory-tracking controllers
- Implement most promising controller and interface to Matlab. Tune and validate performance against fixed target trajectories.

Procedures

- Literature study: Read the provided material and get acquainted with the subject of the project.
- Project plan: Fix a project plan which consists of the major work phases, corresponding milestones and due dates. Define the requirements that the components and system parts will have to fulfil.
- Testing: Test the plant configuration and the implemented software, document errors and bugs.
- Assessment: Assess your system concept and make proposals for improvements.
- Documentation: Complete documentation is considered an important part of any student project at the Automatic Control Lab. Therefore it is a good idea to start writing the documentation while the work is progressing.
- Evaluation of work in progress: The progress of the work is to be constantly monitored and evaluated on the basis of the milestones that have been defined in the project plan. Changes in the project plan have to be accounted for in the written report.
- Workplace: A workplace including a PC is provided in the ETL Building of ETH Zurich
- Final Presentation: Present your results to the members of the Automatic Control Laboratory at the end of the project. (Date: To be determined, duration 15 to 20 minutes)
- Meetings: Discuss the progress of your work with your supervisor in regular meetings.

Project Conclusion

- Hand in a copies of the final report to your supervisors
- Should be handed in to M. Mariani (ETL I23) 6 months after initial work commencement. To be handed in are:
- CD-ROM containing the project (pdf), the final presentation as well as any other relevant material
- Cleaning up your computer account: Delete all irrelevant data on your Unix/PC account and leave only relevant files and directory structures in place.

Supervisor	C.N. Jones, jones@control.ee.ethz.ch
Place of work	Institut für Automatik, ETL
Supervising professor	Prof. M. Morari, Institut für Automatik

Contents

Chapter 1

Introduction

This thesis is a follow up to a thesis by Marc Osswald and Florian Engeler about Real-time control of 1:43 scale race cars [1], which covers the design and implementation of an autonomous real-time RC car racing system. They managed to implement PID controllers, which worked sufficiently after a great deal of tuning. They stated ”..., one of our biggest problems was the lack of a useful model.” in their conclusion and in their outlook they said ”As mentioned before a first step toward more advanced control will definitely be the derivation of a reliable model. To obtain such, accurate knowledge of the car is essential, e.g. the configuration manner of the motor. Furthermore it could be very useful to setup a plain racing area without boundaries to be able to do more significant tests and analysis.”

The goal of this thesis was to develop validated models of these cars and a simple controller to track a given path by reviewing literature, developing experiments and fitting models to recorded data.
Chapter 2 describes the cars, which are the subjects to be modeled, including their interfaces. It further describes the test setup and measurement system that was used for data collection. Chapter 3 will give an analysis of vehicle physics for a specific set of RC cars as well as an analysis of its relevant components and functions. Chapter 4 describes the derivation of the final models. Chapter 5 shows the results that were achieved, what kind of experiments were used to validate the models. Chapter 6 provides our conclusion and gives an outlook on future projects.

Chapter 2

System description

2.1 Kyosho D-Nano race cars

The car that has been modeled in this thesis is a 1:43 scale Kyosho D-Nano Lamborghini Murcielago. The car has a trapezoid steering mechanism driven by a small servo motor with feedback from a potentiometer. The motor is a 7mm Coreless/XSpeed DC motor and located in front of the rear axle. The scope of delivery supplies multiple gear ratio's from 5.17:1 to 3.44:1, in this thesis the standard delivery gear ratio of 5.17:1 has been used. The rear axle is suspended by an H-bar and contains a tiny disk differential. The whole car including battery weights 50 grammes and is of 107x50x27 millimeters in size. Table 2.4 shows the technical specifications and Fig 2.1 shows some pictures of the Kyosho D-Nano Lamborghini Murcielago. For this thesis no tuning has taken place.

2.2 Kyosho D-Nano controller

The Kyosho D-Nano controller is the interface designed for human control usually powered by four AAA batteries. The controller consists of a power button, a potentiometer designed as a wheel to control the steering input and a potentiometer designed as a gun trigger to control the throttle input. Each of the two inputs can be trimmed, which will be discussed further in Section 3.3. The two digitalized inputs for throttle and steering are then transmitted wireless with a carrier frequency of 2.4Ghz.

2.3 Vision system

The vision system consists of two Pointgray cameras, with each having an infrared LED array, an infrared filter, and they are connected to a computer with firewire 800 cables. The RC cars are tagged with different shapes of reflectors. The tags reflect the light from the infrared arrays back to the camera through the infrared

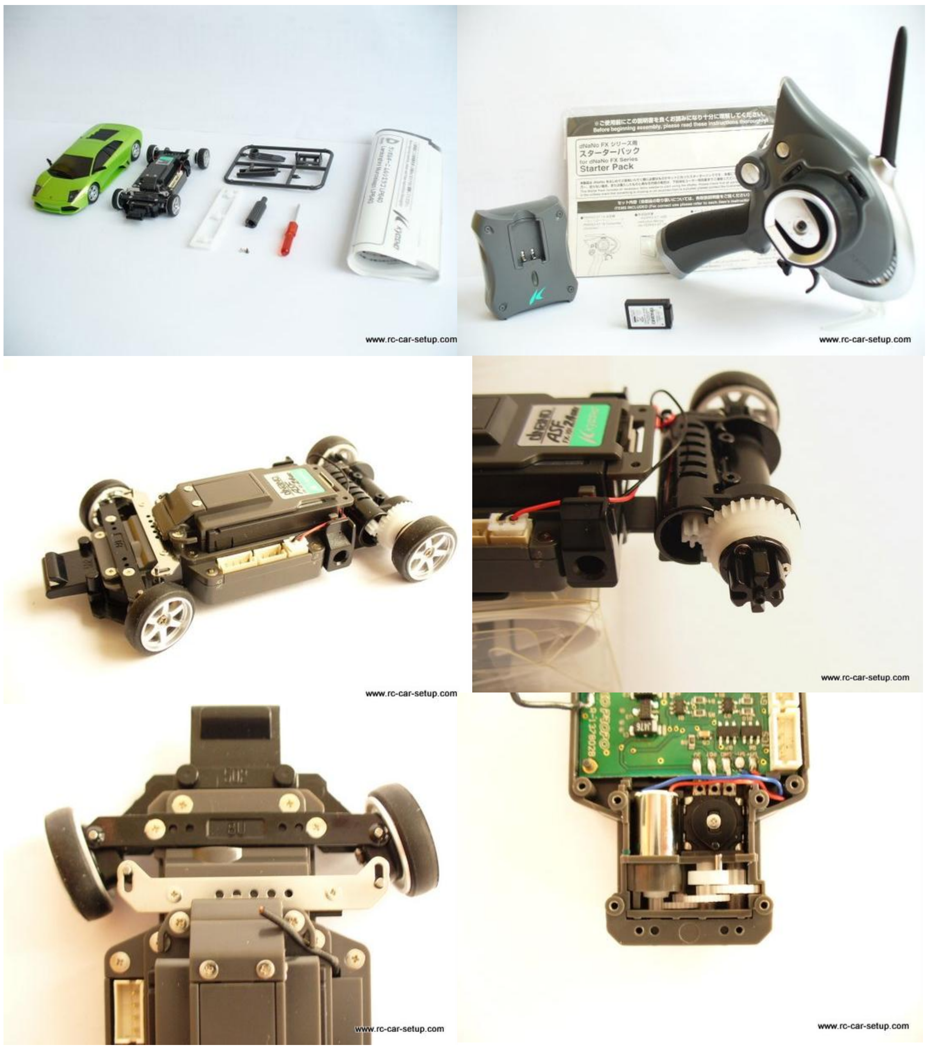


Figure 2.1: Pictures of the Kyosho D-Nano Lamborghini Murcielago showing the scope of delivery (top row) placement of the motor and gear attached via an H-bar as suspension (center row) and trapezoid steering driven by a servomotor and measured by a potentiometer (bottom row) [2].

filter.
With this method a resolution of 5 mm and 3° can be achieved. Limited by the transmission via the firewire cable the vision system is able to cover either a 2 by 3 meter area with a frame rate of 200Hz or a 4 by 3 meter area with a frame rate of 100Hz. The outputs of the vision system are coordinates in centimeters [cm] and degrees [°] for the car's orientation. The origin can be chosen arbitrarily. Further details can be found in 'Infrared based vision system for tracking 1:43 scale race cars' [3].

2.4 Measurement setup

The measurement setup has been kept as simple as possible. An interface was programmed within the main loop of the vision systems C++ code. The interface allows the user to create two voltages with a digital analog converter (DAC) for each Kyosho controller attached. The voltages of the DAC are connected to the two potentiometers of the Kyosho controller, which transmits the digital input to its car. The vision system then measures the car's position and orientation. So far the interface can create input sequences either by keyboard or via a play station controller. The interface can also record and save data. Figure 2.2 shows that a controlling algorithm could be directly inserted into the interface since the feedback already exists.

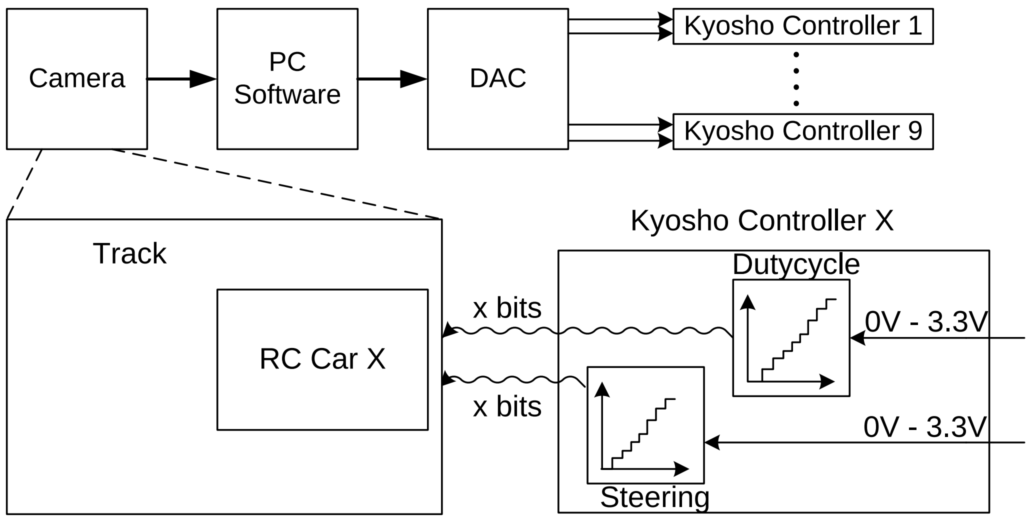


Figure 2.2: Block diagram of the system setup.

Parameter	Value			
mass	50	g		
gear ratio	5.17:1			
wheelbase	62	mm		
length	107	mm		
width	50	mm		
hight	27	mm		
	front		rear	
track	41.6	mm	42.5	mm
tire-diameter	15	mm	16	mm
tire-width	4	mm	5.5	mm

Table 2.1: Specifications of the Kyosho D-Nano Lamborghini Murcielago [2].

Chapter 3

Analysis

The models of the dNano car presented in chapter 4 is based on an in-depth analysis of the dynamics of the car and its traction drive. Furthermore special attention is paid to the remote controller, which is the interface to the motors of the car (i.e. the traction motor and the servo motor for steering). Since all control inputs are sent to the car via remote controller, it is important to know its behavior.

3.1 Dynamics of the dNano car

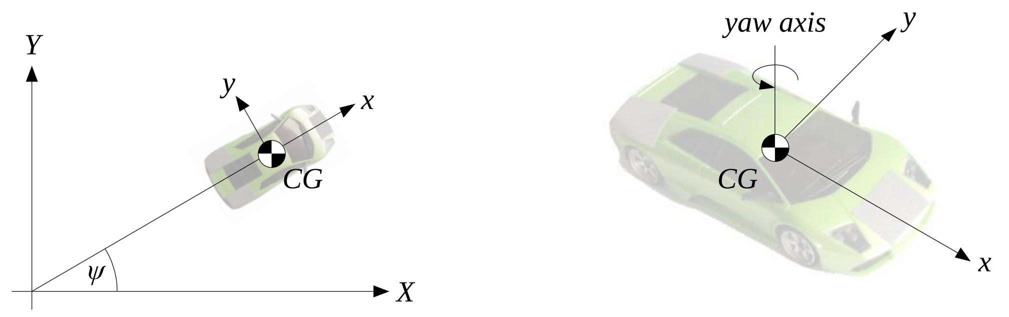


Figure 3.1: Global coordinate system (left) and car coordinate system (right)

The derivation of the equations of motion follows straight Giancarlo Genta's book: Motor vehicle dynamics [4] and therefore is kept short. The coordinate system is shown in figure 3.1. In order to simplify the calculation of forces acting on the car we introduce the car coordinate frame, where the x-axis is always aligned with the lateral axis of the car. The position of the car is described in global coordinates denoted with the capital letters X and Y. They represent the location of the center of gravity (CG). The orientation is given by the heading angle denoted with ψ. In terms of X, Y and ψ the general equations of motion for the rigid car body may be written as

$$\begin{aligned}\ddot{X} &= mF_X \\ \ddot{Y} &= mF_Y \\ \ddot{\psi} &= I_Z M_Z\end{aligned} \tag{3.1}$$

where F_X and F_Y are the total forces in the X- and Y-direction acting on the CG, m is the mass of the car, I_Z is the momentum of inertia about the Z-axis and M_Z is the yawing torque. In terms of velocities (3.1) can be rewritten and transformed in the car-coordinate system:

$$\begin{aligned}\dot{v_x} &= mF_x - mv_y\omega \\ \dot{v_y} &= mF_x + mv_x\omega \\ \dot{\omega} &= I_z M_z\end{aligned} \tag{3.2}$$

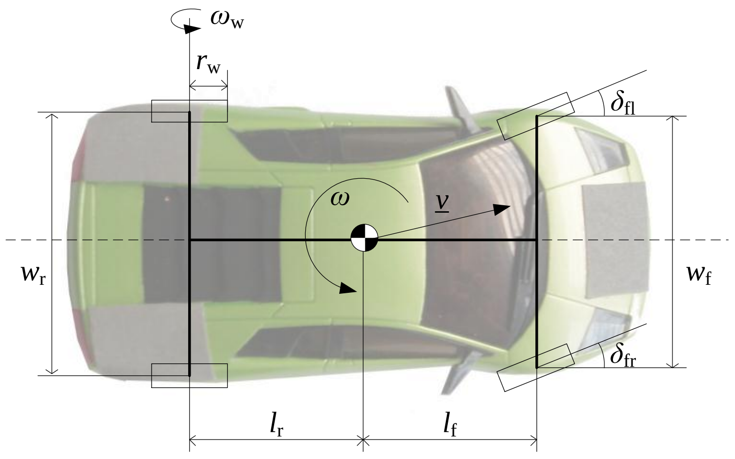


Figure 3.2: Car body from top view

Note that the third equation remains the same in car coordinates. This is because for planar motion the global Z-axis is always aligned with the z-axis of the car. The first two equations have slightly changed by the additional terms $mv_y\omega$ and $mv_x\omega$. They are a result of the transformation into a coordinate system rotating with ω. The total forces F_x and F_y are compositions of tire forces generated at the front and rear wheels, air drag and roll resistance. Figure 3.2 shows the car body from top. l_f and l_r are the distances of the front axle respectively the rear axle from the CG. They can be obtained by running a balancing experiment. The steering angles of the front wheel are denoted with δ_{fl} and δ_{fr}. r_{w} is the radius of the rear wheels and ω_{w} is the rotational velocity of the rear axle. Because of the displacement of the wheels from the CG the tire forces also generate a torque M_z about the yaw-axis.

3.1.1 Tire forces

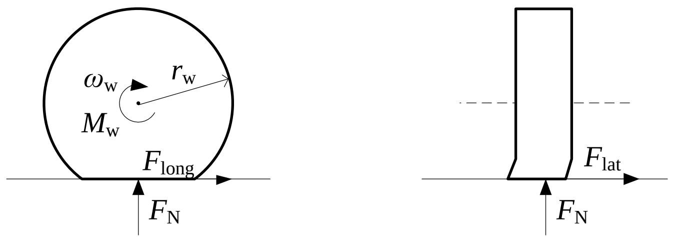


Figure 3.3: Tire from side and front view

Tire forces occur at the contact point between the tire and the ground surface due to deformation of the tire material. Figure 3.3 shows a tire from side and front view. The tire force has been split into its longitudinal and lateral component. The longitudinal tire force is generally expressed as a function of slip

$$\sigma = \frac{\omega_{\text{w}} r_{\text{w}} - v_{\text{x}}}{v_{\text{x}}} \tag{3.3}$$

where $\omega_{\text{w}} r_{\text{w}}$ is the tangential velocity of the wheel at the contact point to the ground and v_{x} is the longitudinal component of the velocity of the car $\underline{v}$ (see figure 3.2). During acceleration, the wheels are spinning slightly faster than the car is moving over the ground (the tire is squeezed longitudinally) and is positive. When the car brakes, the tire is stretched and σ becomes negative. The longitudinal tire force F_{long} may be written as a general function of σ:

$$F_{long} = f(\sigma) \tag{3.4}$$

f strongly depends on the ground surface, the tire material and the normal force acting on the tire. Figure 3.5 shows a typical example of $f(\sigma)$ for real car tires. In literature there exist many analytical and empirical models describing this function [5]. In this project we confine us to a linear approximation of $f(\sigma)$. For small values of σ (3.4) is approximated by

$$F_{\text{long}}(\sigma) = C_\sigma \sigma \tag{3.5}$$

where C_σ is the slope of f at the origin. At higher values of σ, f tends to settle down to a constant value. This is due to the limited availability of grip. To take this effect into account, the linear approximation in (3.5) must be modified in the following way:

$$F_{\text{long}}(\sigma) = \begin{cases} C_\sigma \sigma, & \text{for } |C_\sigma \sigma| < F_{\text{long,max}} \\ F_{\text{long,max}} \text{sgn}(\sigma), & \text{for } |C_\sigma \sigma| \geq F_{\text{long,max}} \end{cases} \tag{3.6}$$

The force exerted in the lateral direction F_{lat} is defined as a function of the slip angle α. The slip angle is the angle between the velocity vector at the hub of the wheel and the longitudinal axis of the tire (see figure 3.4). $\underline{v}_{\text{hub}}$ is given by the transformation law

$$\underline{v}_{\text{hub}} = \underline{v} + \underline{\omega} \times \underline{r} \tag{3.7}$$

where $\underline{v}$ is the velocity at the CG, $\underline{\omega}$ is the angular velocity about the yaw-axis and $\underline{r}$ is the vector pointing from the CG to the hub. For the geometry shown in figure 3.4 the hub velocity is therefore

$$\underline{v}_{\text{hub}} = \begin{bmatrix} v_x - \omega w \\ v_y + \omega l \end{bmatrix} \tag{3.8}$$

and the slip angle can be written as:

$$\alpha = \arctan(\frac{v_{\text{y}} + \omega l}{v_{\text{x}} - \omega w}) - \delta \tag{3.9}$$

Lateral tire forces may be expressed with the same kind of function as longitudinal forces [5]. However, the parameters are different. Analogically to the longitudinal case, the function is linearized about the origin and constrained by $F_{\text{lat,max}}$:

$$F_{\text{lat}}(\alpha) = \begin{cases} -C_\alpha \alpha, & \text{for } |C_\alpha \alpha| < F_{\text{lat,max}} \\ -F_{\text{lat,max}} \text{sgn}(\alpha), & \text{for } |C_\alpha \alpha| \geq F_{\text{lat,max}} \end{cases} \tag{3.10}$$

The minus sign in the equation above is a consequence of the definition of α in (3.9).
So far longitudinal and lateral forces where treated separately. In reality they often occur in combination and interact together. As there is only a limited amount of grip available per tire, longitudinal and lateral forces have to 'share' grip. F_{long} and F_{lat} become dependent on each other. One way to describe this dependency is

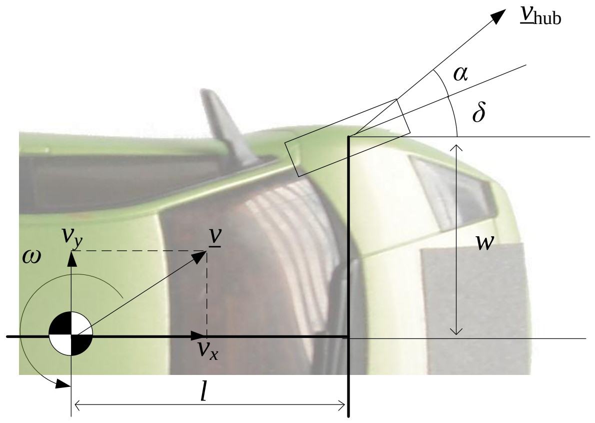


Figure 3.4: Definition of the slip angle α

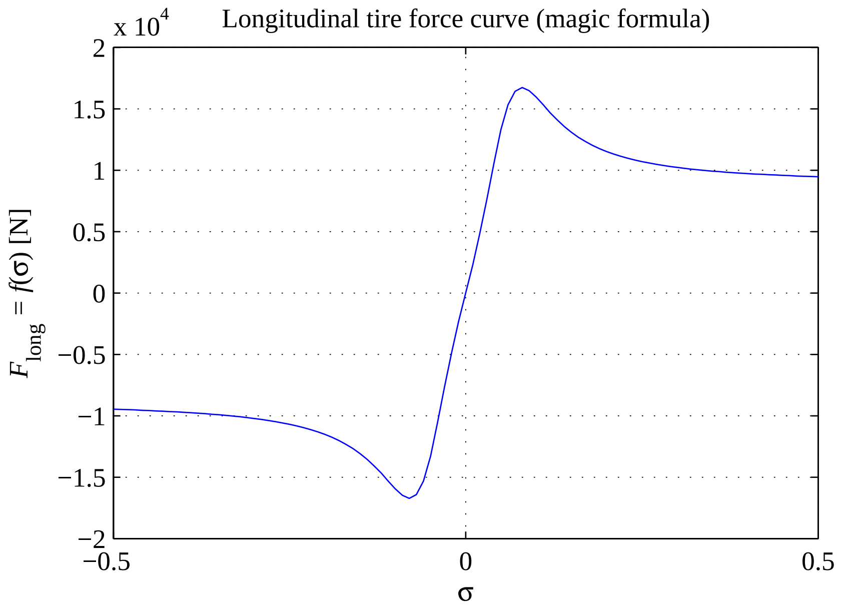


Figure 3.5: Curve $f(\sigma)$ obtained by using the "magic formula" [5].

the so-called friction ellipse [4]. The maximal available grip is given by the ellipse equation:

$$\left(\frac{F_{\mathrm{x}}}{F_{\mathrm{x0}}}\right)^2 + \left(\frac{F_{\mathrm{y}}}{F_{\mathrm{y0}}}\right)^2 = 1 \tag{3.11}$$

F_{x0} and F_{y0} represent the available grip in the longitudinal respectively lateral direction at a given slip angle α. F_{x} and F_{y} are the resulting forces. This idea of combining longitudinal and lateral tire forces will be used in the slip based model (see section 4.1).

3.1.2 Resistive forces

Beside the tire forces described in the previous section, several resistive forces act on the system. There are many sources for these forces such as air drag, rolling resistance of the wheels, friction in the transmission and friction in the motor. Beside the magnitude of different sources it is hard to describe the resistive forces by modeling the exact physics, e.g. aerodynamic drag at low speed. Therefore an empirical approach was taken to model the effects of these forces. More details about the friction model are given in chapter 4.

3.2 Traction drive

3.2.1 Coreless motor

The 7mm coreless motor, which drives the car, may be described by the equivalent circuit shown in figure 3.6. R is the clamp resistance and can be measured directly with a simple ohmmeter. For the motors on the cars we used in our project it is about $4.8\,\Omega$. L is the winding inductance and V_{Ind} is the induced voltage. L can be estimated by applying a voltage step to the clamps while the motor is standing still, i.e. $V_{\mathrm{Ind}} = 0$ and measuring the motor current i_{a} at the same time. In this case i_{a} only depends on L and R and is given by:

$$i_{\mathrm{a}}(t) = \frac{V_{\mathrm{a}}}{R}\left(1 - \mathrm{e}^{-\frac{R}{L}t}\right) \tag{3.12}$$

Figure 3.7 shows the step response for a step of $V_{\mathrm{a}} = 3.1\,\mathrm{V}$. According to (3.12) this result corresponds to an inductance of about $27.75\,\mu\mathrm{H}$.
The dependency between the inner voltage V_{Ind} and the rotational velocity ω of the motor was examined in a no-load experiment. A voltage measurement device was attached to the motor clamps to measure the induced voltage. Since no load was applied to the motor, i_{a} is equal to zero and there is no voltage drop over the clamp resistance. Therefore the induced voltage could be measured directly at the

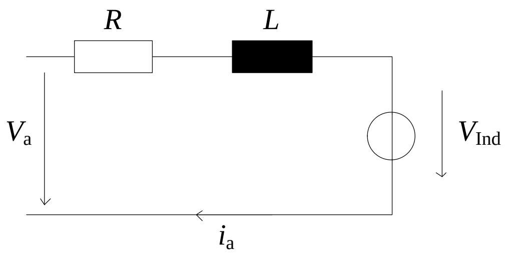


Figure 3.6: Equivalent circuit of the motor

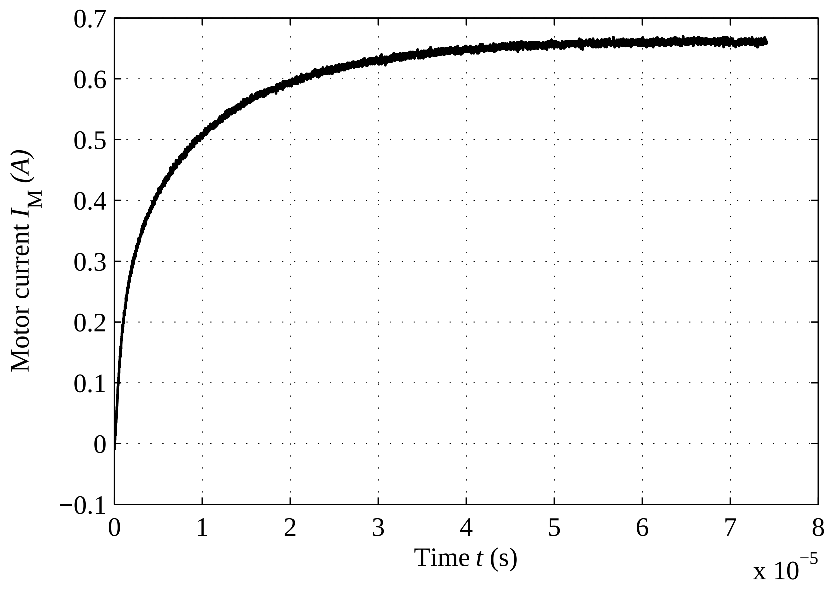


Figure 3.7: Input voltage step response of the motor current at stand still.

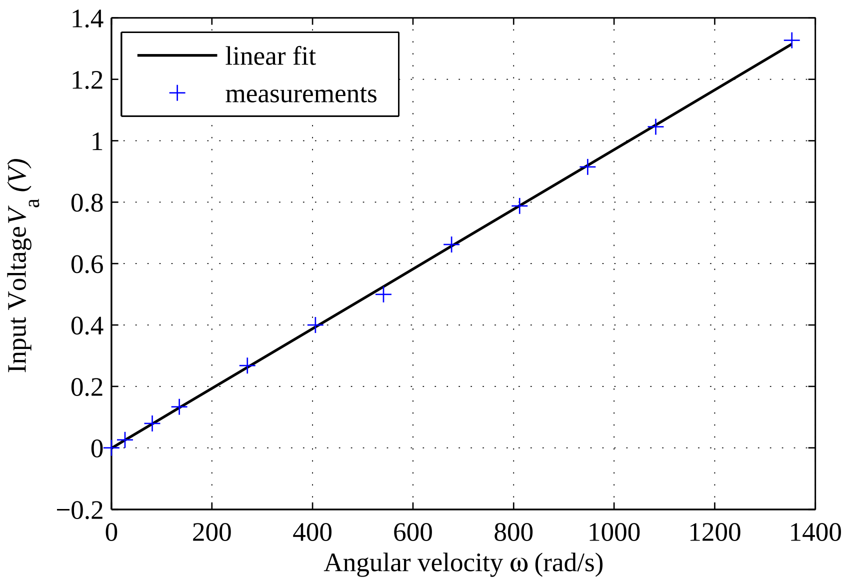


Figure 3.8: Induced voltage at different rotational velocities.

clamps. Figure 3.8 shows V_{Ind} measured at certain velocities ω. As it is common for electric motors V_{Ind} is proportional to ω:

$$V_{\text{Ind}} = k_{\text{M}}\omega \tag{3.13}$$

In literature, the constant k_{M} is often referred to as the motor constant. For the measurement in figure 3.8 k_{M} has a value of 0.97 mVs. The torque M_{el} generated in the motor is proportional to the motor current i_{a} with the same constant

$$M_{\text{el}} = k_{\text{M}} i_{\text{a}} \tag{3.14}$$

Table 3.1 summarizes the motor parameters for the car used in this project.

Parameter	Value	
R	4.8	Ω
L	27.8	µH
k_{M}	0.97	mVs

Table 3.1: Motor parameters

3.2.2 Drive circuit

The motor is driven by an on board h-bridge, which steers the motor current. Figure 3.9 shows the circuit diagram. The motor current i_{a} is set by operating the circuit as a buck converter. Unlike common PWM techniques, e.g. unipolar and bipolar PWM, where all four transistors are used to control the voltage applied to the motor, the buck converter uses two transistors to drive in one direction and the other two to drive in the other direction.
To drive the motor forward transistor T_4 is turned on permanently and T_1 is controlled by a PWM signal. To drive the motor backwards T_2 is turned on permanently and T_3 switches according to the PWM signal. Because the electrical time constant $\tau_{\mathrm{el}} \approx 5.8$ µs given by the motor is very small compared to the switching period T_{T}, which is 50 ms in forward driving mode, the buck converter operates in a discontinuous conduction mode, i.e. the motor current i_{a} reaches and remains zero for a certain while.
The PWM signal is a constant frequency duty cycle variant signal $D \in [0, 1]$. Figure 3.10 shows schematically the voltage and current at the motor clamps for two switching periods in forward driving mode. While T_1 is turned on, which is for DT_{T}, v_{a} equals v_{Bat}. Since the inductance L is very small ($\tau_{\mathrm{el}} << T_{\mathrm{T}}$) we can neglect it and immediately calculate the motor current for the time DT_{T} in the on-state:

$$i_{\mathrm{a}}(t) = \frac{V_{\mathrm{Bat}} - V_{\mathrm{Ind}}}{R}, \text{ for } 0 \le t < DT_{\mathrm{T}} \tag{3.15}$$

For the remaining time $(1 - D)T_{\mathrm{T}}$ transistor T_1 is turned off, preventing i_{a} from flowing. In reality i_{a} does not cease to flow immediately. Directly after T_1 is turned off, the current continues to flow over the diode D_2. While D_2 is still conducting and therefore short-circuits the motor together with T_4. This causes i_{a} to reach zero rapidly. As soon as i_{a} reaches zero, the diode D_2 stops conducting and the motor is disconnected; i.e. $M_{\mathrm{el}} = 0$. Since τ_{el} is very small, i_{a} will reach zero almost immediately and can be approximate to be zero for the whole time T_1 is turned off:

$$i_{\mathrm{a}}(t) = 0, \text{ for } DT_{\mathrm{T}} \le t < T_{\mathrm{T}} \tag{3.16}$$

By applying (3.15) and (3.16) to (3.14) we finally get an expression for the motor torque M_{el} dependent of the duty cycle D:

$$M_{\mathrm{el}}(t) = \begin{cases} k_{\mathrm{M}} \frac{V_{\mathrm{Bat}} - V_{\mathrm{Ind}}}{R}, & \text{for } 0 \le t < DT_{\mathrm{T}} \\ 0, & \text{for } DT_{\mathrm{T}} \le t < T_{\mathrm{T}} \end{cases} \tag{3.17}$$

If all sources of friction are neglected, the dynamics of the drive system may be expressed by

$$J\dot{\omega} = M_{\text{el}}(t) \tag{3.18}$$

where J is the total inertia, taking into account the complete drive train including the rear axle and the rear wheels of the car. The mechanical time constant given by $1/J$ is much greater as the switching period T_{T} and (3.18) may therefore be simplified by introducing the mean value of M_{el} over a switching period;

$$\bar{M}_{\text{el}} = \int_0^{T_{\text{T}}} M_{\text{el}}(t)dt = Dk_{\text{M}}\frac{V_{\text{Bat}} - V_{\text{Ind}}}{R} \tag{3.19}$$

Equation (3.18) can now be written as:

$$J\dot{\omega} \approx \bar{M}_{\text{el}} = Dk_{\text{M}}\frac{V_{\text{Bat}} - V_{\text{Ind}}}{R} \tag{3.20}$$

Replacing V_{Ind} with the expression given in (3.13) leads to:

$$J\dot{\omega} = Dk_{\text{M}}\frac{V_{\text{Bat}} - k_{\text{M}}\omega}{R} \tag{3.21}$$

With a voltage source V_{Bat} of 3.1 V and a duty cycle $D = 1$, the motor can generate a torque of 0.63 mNm at take-off, i.e. $\omega = 0$. This corresponds to an acceleration of about 8.1 m/s^2 for a car with a mass of approximately 50 g.
In reverse the equation for the motor torque remains the same (3.17), but the corresponding duty cycle D would be negative. The switching period T_{T} is much larger when the motor is driven in reverse, but this does not influence any of the previous simplifications.

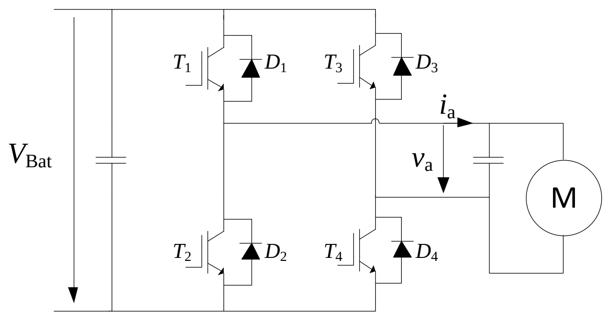


Figure 3.9: Simplified diagram of the drive circuit.

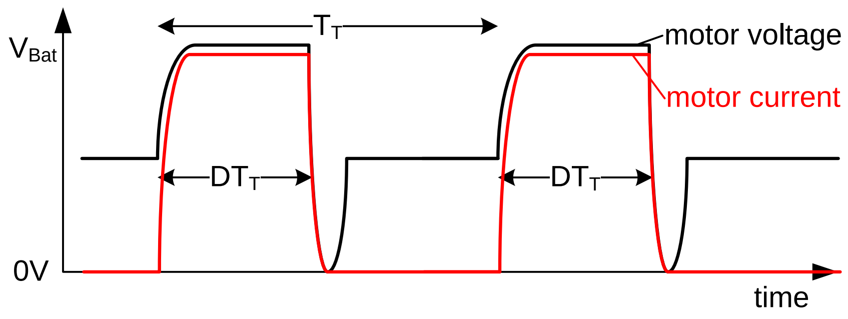


Figure 3.10: Schematic of the voltage measured at the motor clamps.

3.3 Remote controller

As described in section 2.2 the remote controller has two inputs, a throttle voltage and a steering voltage. Because the remote controller is designed as a human interface, it allows the user to adjust its behavior by trimming it with the buttons A, B, R and L (see Fig A.2).
The models derived by this thesis are not dependent on the controller, but any prediction made with one of the models is highly sensitive to the input sequences, especially to the throttle input. This section describes how to deal with the remote controller.

3.3.1 Throttle input

As described in section 3.2.2 the motor is controlled by a duty cycle D. The remote controller transmits a wireless signal, which is digital and contains a value for the duty cycle and a value for the steering angle. Therefore the duty cycle can only take on discrete levels. This is a great disadvantage for control purposes, since incrementing the duty cycle by one step, which is about 0.077, may result in a difference of up to 0.62 m/s^2 in terms of acceleration. The voltage sent to the potentiometer, which defines the throttle input, can be set with the DAC, but depending on the trim settings, the voltages applied don't correspond to the same values for this duty cycles. The best way to deal with this situation is to create a mapping function/table describing the relation between the voltage applied by the DAC and the duty cycle actually set on the car. And because the models are very sensitive to the throttle input, the simulations should only be fed values that can actually be achieved. Otherwise the velocity cannot be predicted precisely and every state is dependent on the velocity. Table 3.2 shows an example of such a table. Note that the mapping is not necessarily linear. By choosing the voltage values equidistant to the steps, the noise on the throttle voltage created by the DAC does not affect the duty cycle.

Throttle control Voltage V_{th} (V)	Duty cycle D
1.800	0.000
1.915	0.070
1.955	0.146
1.990	0.224
2.035	0.300
2.065	0.378
1.115	0.455
2.150	0.532
2.175	0.608
2.215	0.686
2.260	0.762
2.295	0.840
2.315	0.920
2.370	1.000

Table 3.2: Mapping table between the throttle voltage and the duty cycle. The controller has been trimmed such as the neutral voltage is $V_0 = 1.8$V. The voltage values have been chosen equidistant to the transition points.

3.3.2 Steering input

Analog to the throttle input the steering input is also discrete, but it has a higher resolution. Because the steering angle isn't as easily measured as a duty cycle and the wheels have a significant amount of freedom, the mapping can be described by a function

$$V_{\text{steer}}(\alpha) = \begin{cases} \frac{\alpha - c_{0\text{l}}}{c_{1\text{l}}} & ,\text{if } \alpha > 0 \\ \frac{\alpha - c_{0\text{r}}}{c_{1\text{r}}} & ,\text{if } \alpha < 0 \end{cases} \tag{3.22}$$

where $c_{0\text{l}}$, $c_{1\text{l}}$, $c_{0\text{r}}$, $c_{1\text{r}}$ are coefficients that can be measured as described in the appendix. Figure 3.11 shows the measurements and the according mapping function given by (3.22).

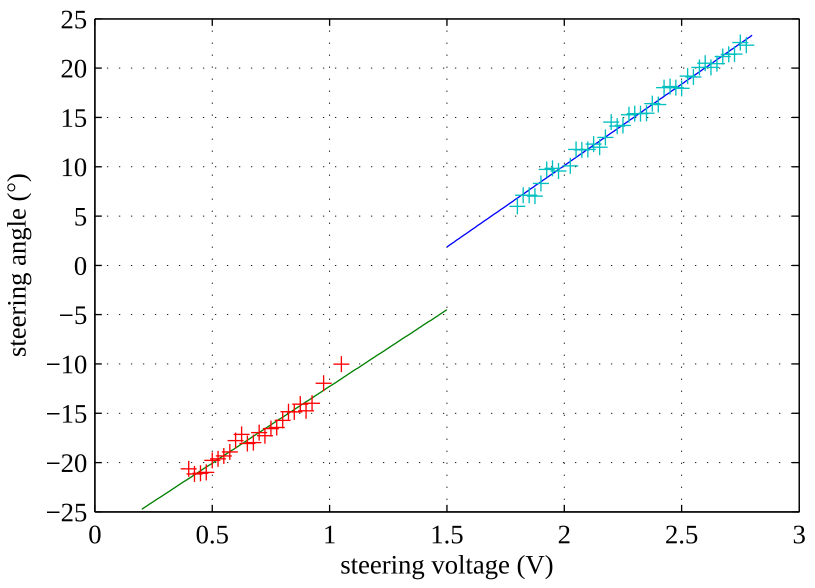


Figure 3.11: Measurements of the steering angle and the linear approximations.

Chapter 4

Modeling

Based on the analysis done in the previous chapter we will derive two different models describing the dynamics of the complete car system. The first model is based on tire force calculations and slip as described in section 3.1.1. The second model is a completely slip-free four-state model. Both models use a simplified version of the car geometry shown in figure 3.2.

4.1 Slip based model

The slip based model derived in this section uses equation (3.10) from section 3.1.1 to calculate lateral tire forces. Longitudinal tire forces are calculated differently, because it is hard to determine σ without any measurement of the tire's rotational speed. The model is simplified by reducing the car's four-wheeled geometry into a two-wheeled bicycle geometry, which is shown in figure 4.1. The slip angles at the front and the rear wheel are obtained by transforming the velocity from CG to the point, where the wheels are attached to the car frame (see section 3.1.1).

$$\begin{aligned}
\alpha_f &= \arctan\left(\frac{v_y + l_f\omega}{v_x}\right) - \delta \\
\alpha_r &= \arctan\left(\frac{v_y - l_r\omega}{v_x}\right)
\end{aligned} \tag{4.1}$$

4.1.1 Front tire forces

Equations (3.6) and (3.10) are approximations of the forces acting between the ground surface and the tires. As the front wheels are not driven by any motor, the longitudinal force component is set to zero, which implies a slip ratio of zero. This means that the front wheels always spin according to the ground velocity of the car. The lateral force is calculated through the slip angle:

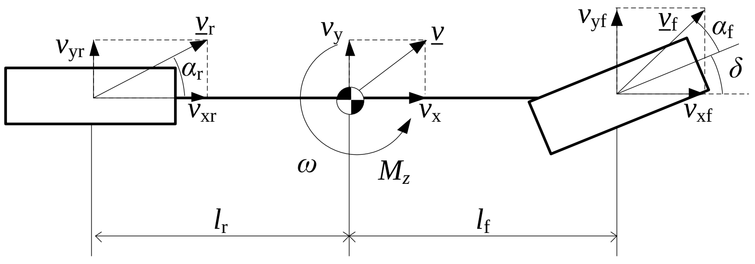


Figure 4.1: Simplified car geometry

$$F_{\text{lat}}(\alpha_f) = \begin{cases} -C_\alpha \alpha_f, & \text{for } |C_\alpha \alpha_f| < F_{\text{lat,max}} \\ -F_{\text{lat,max}} \operatorname{sgn}(\alpha_f), & \text{for } |C_\alpha \alpha_f| \geq F_{\text{lat,max}} \end{cases} \tag{4.2}$$

As the front wheel is not aligned with the car-coordinate system (see figure 4.1) the force has to be rotated into the xy-system. The total force contributed by the front axle in xy-coordinates is therefore given by:

$$\begin{aligned} F_{x,f} &= -\sin(\delta) F_{lat,f} \\ F_{y,f} &= \cos(\delta) F_{lat,f} \end{aligned} \tag{4.3}$$

4.1.2 Rear tire forces

The rear wheels are driven by the electrical motor. They are connected to the motor via a transmission with the transmission ratio N_{Tr} defined as

$$N_{Tr} = \frac{\omega_M}{\omega_w} \tag{4.4}$$

where ω_M is the velocity, at which the motor is turning and ω_w is the rotational speed of the rear wheel. The torque from the motor is transffered to the wheel accordingly. The equation of motion for the rear wheel is

$$J\dot{\omega}_w = N_{Tr} M_{el} - F_{long} r_w \tag{4.5}$$

M_{el} is the torque provided by the motor and is described in section 3.2. F_{long} is the longitudinal tire force. Solving (4.5) for F_{long} and replacing M_{el} with the term given in (3.21) leads to

$$F_{long} = D k_M \frac{V_{\text{Bat}} - k_M N_{Tr} \omega_w}{R r_w} - \frac{J}{r_w} \dot{\omega}_w \tag{4.6}$$

Note that we already used (4.4) to get rid of ω_M. With the assumption that the slip ratio σ remains small, the tangential velocity of the rear wheel $\omega_w r_w$ is always close to the ground velocity v_x in longitudinal direction:

$$\omega_w r_w \approx v_x \tag{4.7}$$

This kinematic constraint allows us to replace ω_w in (4.6), which enables us to define the longitudinal force F_{long} in terms of v_x:

$$F_{long} = D k_M N_{Tr} \frac{V_{\text{Bat}} r_w - k_M N_{Tr} v_x}{R r_w^2} - \frac{J}{r_w^2} \dot{v}_x \tag{4.8}$$

To eliminate $\dot{v}_x$ we need to introduce the equation of motion for v_x according to (3.2) in section 3.1:

$$m\dot{v}_x = F_x \tag{4.9}$$

F_x is the sum of all forces acting in the x direction including F_{long}. Hence we can split up $m\dot{v}_x$ in F_{long} and F_x', containing all the remaining forces:

$$m\dot{v}_x = F_x' + F_{long} \tag{4.10}$$

Now $\dot{v}_x$ can be replaced in (4.8) and the equation can be solved for F_{long}:

$$F_{long} = \frac{m}{r_w^2 m + J} \left(D k_M N_{Tr} \frac{V_{\text{Bat}} r_w - k_M N_{Tr} v_x}{R} \right) - \frac{J}{r_w^2 m + J} F_x' \tag{4.11}$$

In order to keep the notation simple, two constants are introduced:

$$\begin{aligned} C_{m1} &= \frac{m k_M N_{Tr} V_{\text{Bat}} r_w}{R(r_w^2 m + J)} \\ C_{m2} &= \frac{m (k_M N_{Tr})^2}{R(r_w^2 m + J)} \end{aligned} \tag{4.12}$$

These are called the motor constants as they contain all the information of the motor model. Equation (4.11) becomes:

$$F_{long} = C_{m1} D - C_{m2} D v_x - \frac{J}{r_w^2 m + J} F_x' \tag{4.13}$$

The calculation of the lateral tire force is straight forward similar to equation (4.2):

$$F_{\text{lat}}(\alpha_r) = \begin{cases} -C_\alpha \alpha_r, & \text{for } |C_\alpha \alpha_r| < F_{\text{lat,max}} \\ -F_{\text{lat,max}} \ \text{sgn}(\alpha_r), & \text{for } |C_\alpha \alpha_r| \geq F_{\text{lat,max}} \end{cases} \tag{4.14}$$

Due to the presence of the longitudinal force F_{long}, F_{lat} is reduced by a factor of

$$\sqrt{1 - \left(\frac{F_{long}}{F_{long,max}}\right)^2} \tag{4.15}$$

which is calculated from equation (3.11).

4.1.3 Equations of motion

As all tire forces are known now, we can write down the equations of motion according to (3.2):

$$\begin{aligned} m\dot{v}_x &= F_{x,f} + F_{x,r} + F_{x,res} + m v_y \omega \\ m\dot{v}_y &= F_{y,f} + F_{y,r} - m v_x \omega \\ J_z \dot{\omega} &= F_{y,f} l_f - F_{y,r} l_r \end{aligned} \tag{4.16}$$

The term $F_{x,res}$ was added to model the effect of driving resistance, such as air drag and roll resistance. A commonly used approximation is given by Giancarlo Genta [4]:

$$F_{x,res} = -C_{r0} - C_{r2} v_x^2 \tag{4.17}$$

The coefficients C_{r0} and C_{r2} are obtained by experiments. In the appendix a method is described of how to find these coefficients. For the calculation of $F_{x,r}$ we need to determine F'_x. By inspection we can see that F'_x is equal to:

$$F_{x,f} + F_{x,res} + m v_y \omega \tag{4.18}$$

4.1.4 Model equations

So far all the calculations were done in the car frame coordinates. In order to obtain an expression for $\dot{X}$ and $\dot{Y}$ the velocities v_x and v_y need to be transformed into global coordinates:

$$\begin{aligned} \dot{X} &= v_x \cos(\psi) - v_y \sin(\psi) \\ \dot{Y} &= v_x \sin(\psi) + v_y \cos(\psi) \end{aligned} \tag{4.19}$$

The equation for $\dot{\psi}$ is simply:

$$\dot{\psi} = \omega \tag{4.20}$$

By collecting all the equations we arrive at the complete system:

$$\begin{aligned}
\dot{X} &= v_x \cos(\psi) - v_y \sin(\psi) \\
\dot{Y} &= v_x \sin(\psi) + v_y \cos(\psi) \\
\dot{\psi} &= \omega \\
m\dot{v}_x &= F_{x,f} + F_{x,r} + F_{x,res} + m v_y \omega \\
m\dot{v}_y &= F_{y,f} + F_{y,r} - m v_x \omega \\
J_z \dot{\omega} &= F_{y,f} l_f - F_{y,r} l_r
\end{aligned} \tag{4.21}$$

4.2 Slip free model

The slip free model is derived from the car geometry showed in figure 4.2. Generally, it is valid under the following assumptions:

1. The tire force limit is never exceeded.
2. The slip angles α_f and α_r are small.
3. The steering angle δ is small.

Assumption 1 is fulfilled under normal driving conditions, where longitudinal and lateral forces remain small. However, in extreme cornering maneuvers, where throttle is applied while the car is turning sharply, the forces will saturate due to insufficient grip, and the car may break out. Such behavior is not covered by this model. In such extreme driving situations assumption 2 would also be violated since the car gets very slippy.

4.2.1 Steady state turning

As the slip angles are assumed to be small ($\alpha \approx 0$), the lateral component of the velocity vector $v_{y,r}$ at the rear must vanish:

$$\alpha = 0 \Leftrightarrow \arctan\left(\frac{v_{y,r}}{v_{x,r}}\right) = 0 \Leftrightarrow v_{y,r} = 0 \tag{4.22}$$

$v_{y,r}$ may be obtained by transforming the velocity of the car from the CG to the wheel's hub:

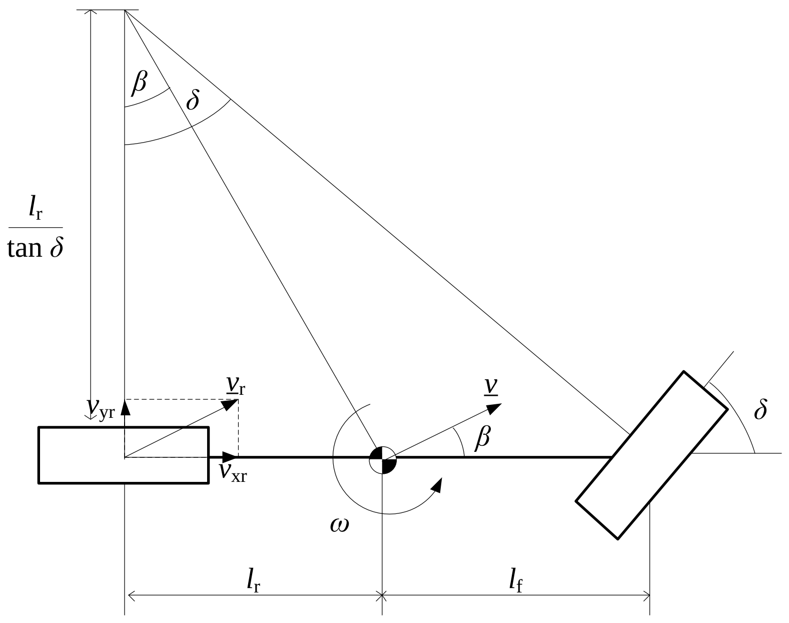


Figure 4.2: Geometry for slip free model

$$v_{y,r} = v\sin(\beta) - \omega l_r \tag{4.23}$$

According to (4.22) the right-hand side of the equation (4.23) must be equal to zero. Setting $v_{y,r}$ to zero and solving for ω results in:

$$\omega = \frac{v\sin(\beta)}{l_r} \tag{4.24}$$

This can be further simplified by taking into account that δ is small. For small steering angles β can be approximated as follows:

$$\tan(\beta) = \frac{l_r}{l}\tan(\delta) \approx \frac{l_r}{l}\delta$$
$$\Rightarrow \beta \approx \frac{l_r}{l}\delta \tag{4.25}$$

Applying this to (4.24) and setting $\sin(\beta) \approx \beta$ we get the following expression for the yawing velocity of the car:

$$\omega = \frac{v\delta}{l} \tag{4.26}$$

4.2.2 Driving force

The driving force model is taken completely from the slip based model, but v_x and v_y are replaced according to:

$$\begin{aligned} v_x &= v\cos(\beta) \approx v \\ v_y &= v\sin(\beta) \approx v\beta \approx v\frac{l_r}{l}\delta \end{aligned} \tag{4.27}$$

Applying these changes to the equation for the velocity in (4.21) leads to:

$$\dot{v} = C_{m1}D - C_{m2}Dv - C_{r2}v^2 - C_{r0} - \omega v\frac{l_r}{l}\delta \tag{4.28}$$

Note that the mass m has been included in the constants. The physical meaning of the constants is explained in section 4.1. ω is given by the expression (4.26) and is no longer a state of the system. Replacing ω in (4.27) gives the final expression for the derivative of the velocity:

$$\dot{v} = C_{m1}D - C_{m2}Dv - C_{r2}v^2 - C_{r0} - (v\delta)^2\frac{1}{l} \tag{4.29}$$

4.2.3 Model equations

The complete car system will be described in terms of four state variables, namely the global coordinates X and Y, the orientation ψ and the velocity v. The equations for X and Y are derived directly from the geometry shown in figure 4.2:

$$\begin{aligned} \dot{X} &= v\cos(\psi+\beta) \approx v\cos\left(\psi + \frac{l_r}{l}\delta\right) \\ \dot{Y} &= v\sin(\psi+\beta) \approx v\sin\left(\psi + \frac{l_r}{l}\delta\right) \end{aligned} \tag{4.30}$$

The derivative of the orientation $\dot{\psi}$ is equal to ω and is given in (4.26). An expression for $\dot{v}$ is shown in (4.28). Summarizing we can write down all the state equations for the complete system

$$\begin{aligned} \dot{X} &= v\cos(\psi + C_1\delta) \\ \dot{Y} &= v\sin(\psi + C_1\delta) \\ \dot{\psi} &= v\delta C_2 \\ \dot{v} &= C_{m1}D - C_{m2}Dv - C_{r2}v^2 - C_{r0} - (v\delta)^2 C_2 \end{aligned} \tag{4.31}$$

where $C_1 = \frac{l_r}{l}$ and $C_2 = \frac{1}{l}$ have been introduced as geometrical parameters. Table 4.1 gives an overview over all the parameters used in this model.

Parameter		Expression	Typical value	Description
C_1	-	$\frac{l_r}{l}$	0.5	Geometrical parameter, see figure 4.2
C_2	m^{-1}	$\frac{1}{l}$	16.13	Geometrical parameter, see figure 4.2
C_{m1}	$\mathrm{m/s}^2$	$\frac{k_M N_{Tr} V_{\mathrm{Bat}} r_w}{R(r_w^2 m+J)}$	11.52	First motor parameter, see (4.12)
C_{m2}	1/s	$\frac{(k_M N_{Tr})^2}{R(r_w^2 m+J)}$	2.74	Second motor parameter, see (4.12)
C_{r2}	1/m	–	0.05	Second order friction parameter, see (4.17)
C_{r0}	$\mathrm{m/s}^2$	–	0.54	Zero order friction parameter, see (4.17)

Table 4.1: Overview of the parameters in the slip free model

Chapter 5

Applications and Results

In this chapter the slip free model will be examined by looking at two different applications. In the first application, the model is used to predict the trajectory of the car given the input sequence for the whole prediction horizon and the initial condition. The second application is a nonlinear controller which uses the inverse model to navigate the car along a given path.

5.1 Trajectory prediction

One of the main goals of this semester project was to provide a model of the 1:43 scaled dNano car, which has the ability to predict the trajectory for given input sequences. The slip free model presented in chapter 4 provides this ability as the trajectory is described by X, Y and ψ, which are all state variables of the model. Additionally the velocity v is also predicted, because it is a state of the model. For simulation and prediction purposes the model was implemented in Simulink (see appendix B). To test the prediction performance, a random trajectory was generated by a human driver. Figure 5.1 shows four different predictions starting at four different points in time for a horizon of $T_h = 5\,\mathrm{s}$. The model is able to predict the trajectory quite accurately for a limited time span. However, it tends to drift away at some point, especially when the steering angle changes. This may be caused by an inaccurate mapping of the control voltage for the steering angle as discussed in section 3.3.2. Furthermore the quantization of the steering angle may also lead to deviations from the measured values. The accuracy of the throttle control voltage map is much higher, since it was obtained by measuring the control voltage and the output, i.e. the duty cycle D directly. This allows to predict the velocity even for a time horizon of $T_h = 30\,\mathrm{s}$ with a mean square error $\bar{e}$ less than $0.02\,m/s$, as shown in figure 5.2.

The deviation between the simulated trajectory $\underline{r}_{\mathrm{sim}}(t)$ and the real trajectory

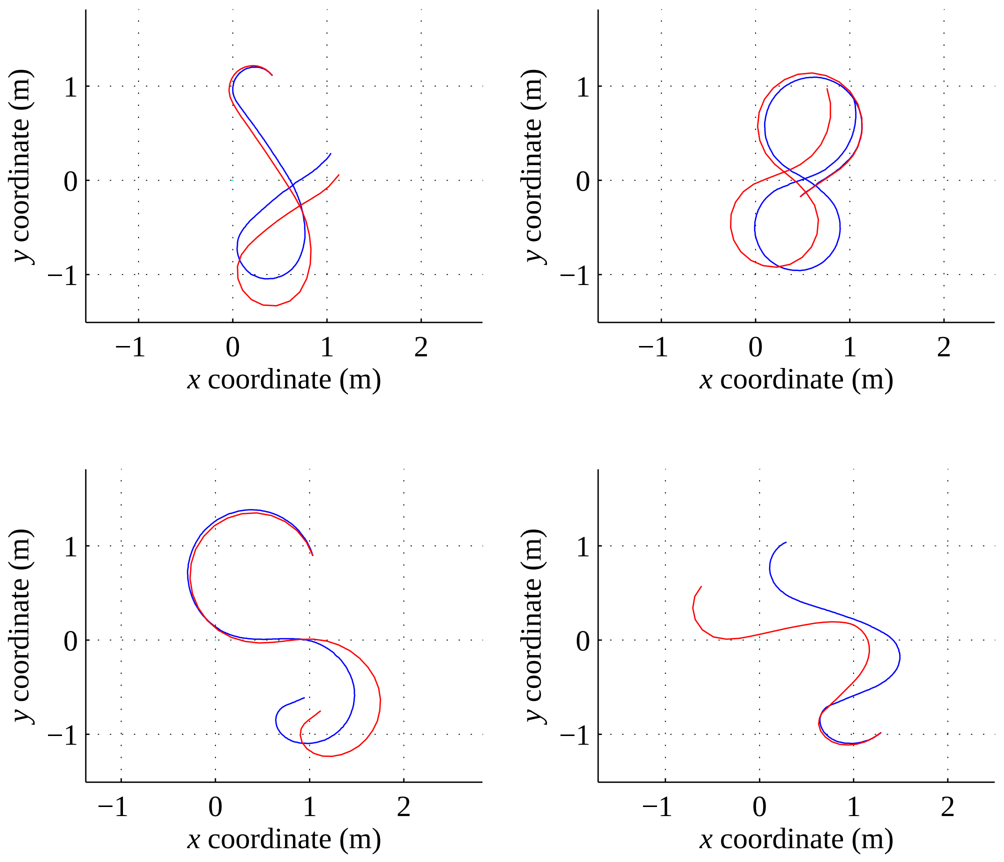


Figure 5.1: Four independent trajectory predictions. The blue line represents the true trajectory, the red line is the trajectory predicted by the model. All trajectories were predicted for a time span of 5 s.

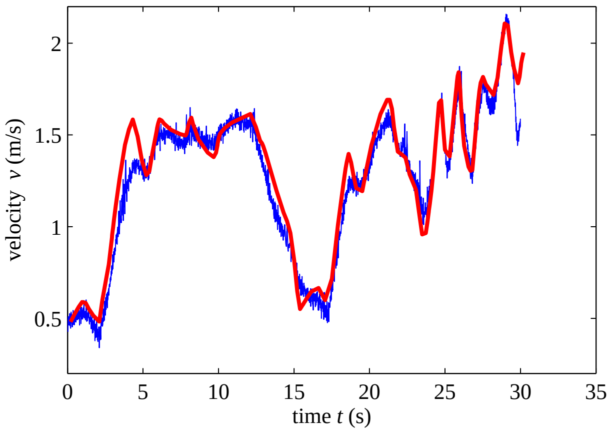


Figure 5.2: Prediction of the velocity for a given input sequence. The blue line shows the real velocity, the red line shows the velocity predicted by the model. The prediction was done for the time span of 30 s.

$\underline{r}_{\mathrm{real}}(t)$ may be quantified as

$$e_{\mathrm{r}}(t) = \sqrt{\frac{1}{T_{\mathrm{h}}} \int_{t}^{t+T_{\mathrm{h}}} \|\underline{r}_{\mathrm{sim}}(\tau) - \underline{r}_{\mathrm{real}}(\tau)\|^2 d\tau} \tag{5.1}$$

where T_{h} is the time horizon of the prediction. Figure 5.3 shows the deviation for $T_{\mathrm{h}} = 1\,\mathrm{s}$ and for $T_{\mathrm{h}} = 0.1\,\mathrm{s}$. Figure 5.4 shows the side slip angle β measured over time compared with the modeled value. As soon as the true side slip angle β deviates from the modeled value the error becomes significantly larger, because the assumption of small slip angles is violated. The two large peeks in figure 5.3 are a result of the car breaking out.
It is important to keep in mind that the model is based on the assumption that slip remains small. If the car is driven at its limits of stability it cannot be guaranteed that the model provides any useful prediction.

5.2 Trajectory tracking controller

To demonstrate the abilities of the slip free model a controller for tracking a given trajectory was implemented. The main principle of the controller is to use some kind of an inverse model to calculate the control inputs D and δ, which have to be applied to achieve a desired acceleration. The tracking algorithm may then calculate the acceleration, which is needed to track the trajectory, and does not have to provide D and δ directly.

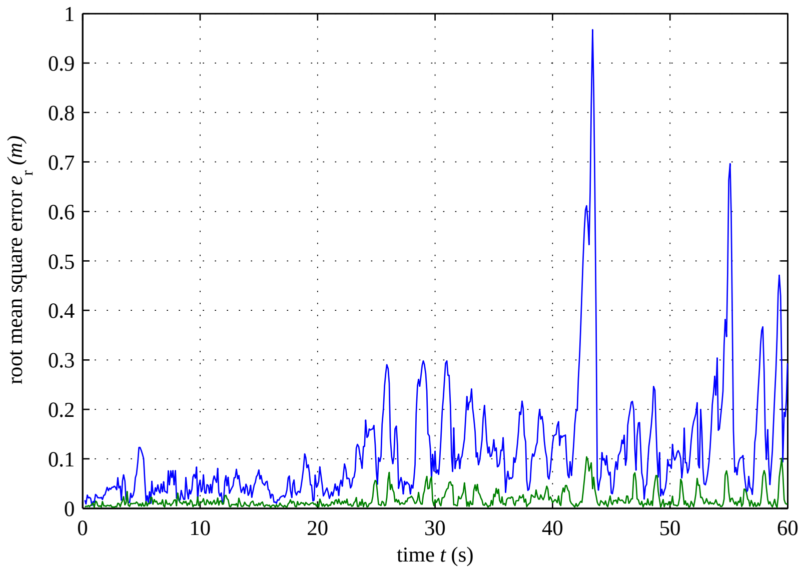


Figure 5.3: Deviation $e_{\mathrm{r}}(t)$ of the simulated trajectory from the real trajectory for $T_{\mathrm{h}} = 1\,\mathrm{s}$ in blue and $T_{\mathrm{h}} = 0.1\,\mathrm{s}$ in green.

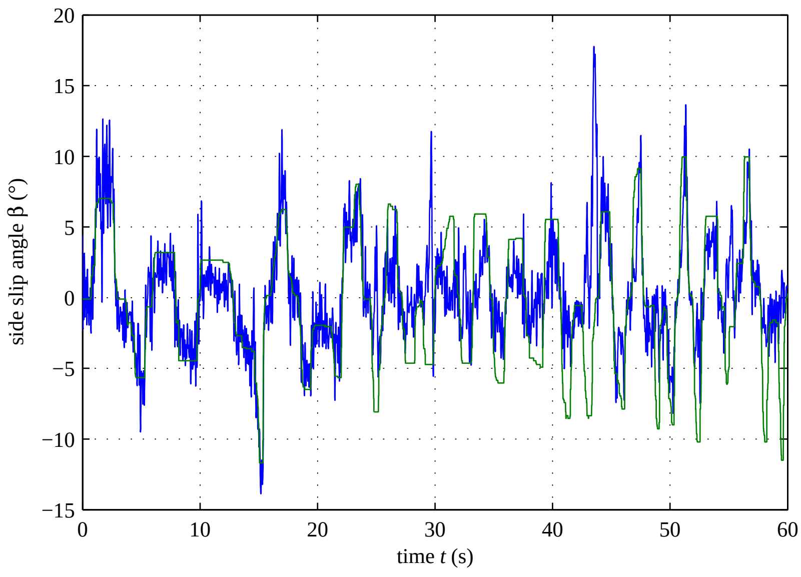


Figure 5.4: Side slip angle β measured over time (blue) and calculated from model (red).

5.2.1 Inverse model

The slip free model provides the following equation for the cars acceleration:

$$a_{\|} = C_{m1}D - C_{m2}Dv - C_{r2}v^2 - C_{r0} - (v\delta)^2 C_2 \tag{5.2}$$

The acceleration is parallel to the velocity vector $\underline{v}$ (parallel to the x_v axis in figure 5.5) and therefore is denoted with $a_{\|}$. In order to maintain a circular movement (e.g. driving in a curve) a second acceleration perpendicular to $a_{\|}$ (parallel to the y_v axis in figure 5.5) must be acting on the car, namely the centripetal acceleration:

$$a_{\perp} = v\omega \tag{5.3}$$

ω is well known from the model equations (4.31) and may be replaced:

$$a_{\perp} = v^2 \delta C_2 \tag{5.4}$$

Now equations (5.2) and (5.4) are solved for the control inputs D and δ. Starting with (5.4) we get the following expression for the steering angle:

$$\delta = \frac{a_{\perp}}{v^2 \delta C_2} \tag{5.5}$$

Note that δ is not defined for $v = 0$, which is obvious since it is not possible to drive in a circle, while the car is not moving. The second equation (5.2) is solved for the duty cycle

$$D = \frac{a_{\|} + C_{r2}v^2 + C_{r0} + (v\delta)^2 C_2}{C_{m1} - C_{m2}v} \tag{5.6}$$

where δ was calculated previously. The duty cycle D may become greater than 1 especially when the denominator is small. This is clear since the power of the electrical motor is limited, and therefore it is not possible to achieve an arbitrary large acceleration $a_{\|}$.

5.2.2 Tracking algorithm

Now that we have expressed the control inputs in terms of the accelerations $a_{\|}$ and $a_{\perp}$ we need to determine these accelerations in a way, that the car moves along a given trajectory. Figure 5.5 shows the car moving a bit off the trajectory represented by the dotted line. $\underline{r}_{\text{dh}}$ is the vector pointing from the cars CG to the nearest point p on the trajectory. $\tilde{x}$ and $\tilde{y}$ define a coordinate system with the $\tilde{x}$ axis

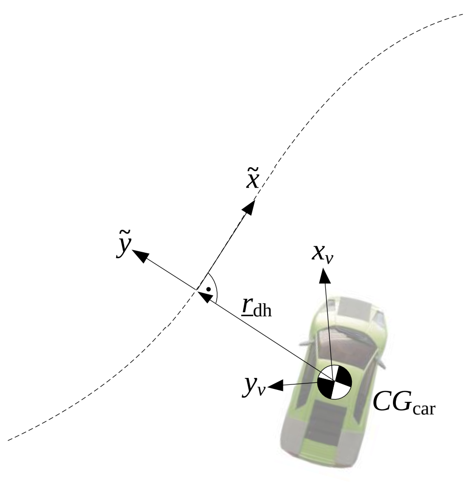


Figure 5.5: Car driving along the trajectory (dotted)

being perpendicular to the trajectory in p. The $x_v y_v$ coordinate system is orientated with respect to the velocity vector of the car.
The acceleration is now determined by emulating a particle moving in the potential field shown in figure 5.6. The potential has the shape of a parabola and has its minimum at the origin, which is at the point p lying on the trajectory. If the potential is given by

$$U(\tilde{y}) = \frac{1}{2} k_{\mathrm{P}} \tilde{y}^2 \tag{5.7}$$

the force acting on the particle can be calculated as:

$$F_{\tilde{y}} = -\frac{\partial U(\tilde{y})}{\partial \tilde{y}} = -k_{\mathrm{P}} \tilde{y} \tag{5.8}$$

The force causes the particle to accelerate toward the trajectory. In order to stabilize the particle a damping term is added to (5.8):

$$F_{\tilde{y}} = -k_{\mathrm{P}} \tilde{y} - k_{\mathrm{D}} \dot{\tilde{y}} \tag{5.9}$$

This can be seen as a standard PD controller controlling to deviation of the particle, i.e. the car from the trajectory to zero. However, to move the car along the trajectory, a second acceleration in the $\tilde{x}$ direction is needed. The necessity of this second acceleration becomes evident when the car is moving right on the trajectory. In this case $\underline{r}_{\mathrm{dh}} = 0$. Therefore no acceleration will be present according to (5.9). But we want the car to be moving along the trajectory even in the presence of friction, which slows the car down. To overcome friction and to maintain a constant velocity a PI controller acting in the $\tilde{x}$ direction is used. The controlled variable is the car's velocity v and the manipulated variable is the acceleration in the $\tilde{x}$ direction. The control law is defined by the following expression:

$$F_{\tilde{x}} = -k_{\mathrm{P,v}}(v - v_{ref}) - k_{\mathrm{I,v}}\frac{d}{dt}(v - v_{ref}) \tag{5.10}$$

With expression (5.9) and (5.10) the reference acceleration for the car is determined. The only remaining step in order to apply these accelerations to the inverse model is to rotate them from the $\tilde{x}\tilde{y}$ system into the coordinate system orientated with respect to the velocity vector $\underline{v}$. Matrix T describes this transformation:

$$T = \begin{bmatrix} \cos(\psi + \delta C_1) & \sin(\psi + \delta C_1) \\ -\sin(\psi + \delta C_1) & \cos(\psi + \delta C_1) \end{bmatrix} \tag{5.11}$$

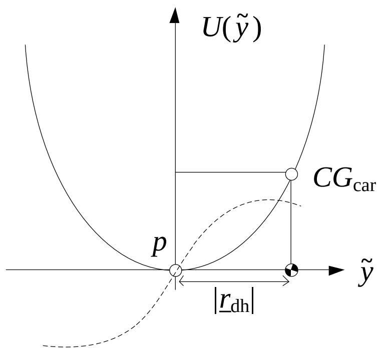


Figure 5.6: Potential function in the direction of $\tilde{y}$

5.2.3 Tracking controller operating in the loop

The described trajectory tracking controller was implemented in MATLAB (see appendix C) and tested on the real system. During the test, the X and Y position of the car and the velocity v where recorded. They are shown in figure 5.7 respectively 5.8 along with their reference values. Note that the car remains a little bit off while driving the curve, which seems plausible since the algorithm behaves like a PD controller and does not perform any integral action.
The main advantage of using the inverse model is the reduction of complexity of the input-output behavior of the system. The more complex car dynamics are hidden in the system, while from the outside, the system behaves like a particle moving in

a potential field. The algorithm has though to be used with special care because the inverse model is not valid for all driving situations. Especially when driving at the limits of stability the requirements for the simplifications made during the derivation of the model are not met. Therefore the controller parameters have to be adjusted in a way that the car does not leave a certain sub space of the state space, for which the model is valid.

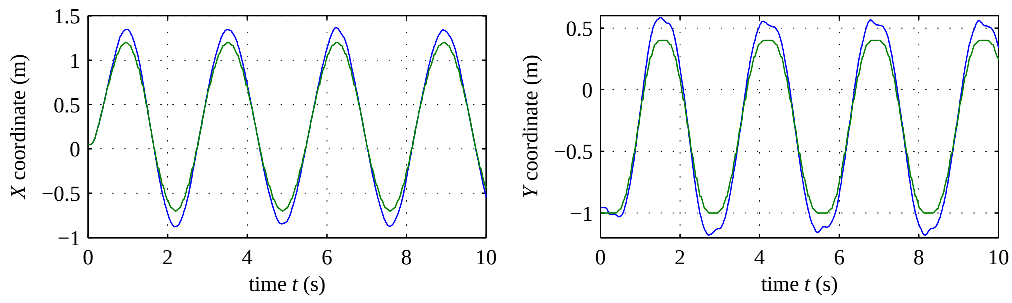


Figure 5.7: X and Y coordinates of the car during the controller test: Measured values shown in blue, reference values shown in green.

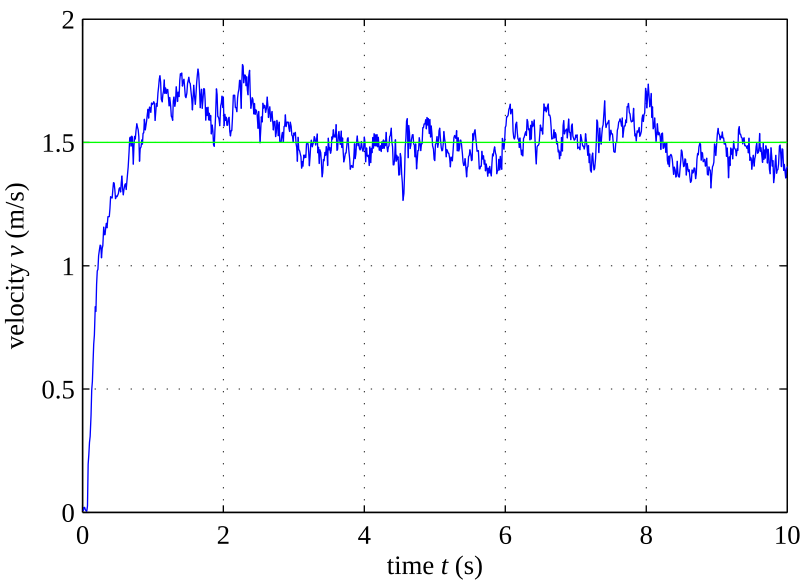


Figure 5.8: Measured velocity v (blue) and reference velocity (green) for the controller test.

Chapter 6

Conclusion

In this thesis we first covered the vehicle physics, which are well documented in literature. In a second step the D-Nano Lamborghini Murcielago remote control race car was picked apart and analyzed in depth. From the knowledge gained in the first two steps we developed two models suiting this exact race car, which are both highly non linear. Because these two models needed to be validated, we created a measurement setup, which was very time consuming. Supplied with a measurement setup the proper engineering work could take place. The actual car physics did not present the greatest difficulties, but the interface, i.e. the Kyosho remote controller, did so.

Because the interface was meant as a human interface it provides many features to personalize how the controller behaves. It interprets and transmits the two input signals, which are basically two voltage levels, to the car. This problem was solved by creating a mapping function between the voltage levels applied and the inputs used by the models. It was important though not to touch the Kyosho controller anymore after the mapping function was measured. Nevertheless the simulation results did not satisfy our expectations. With further investigations we discovered that the models are highly sensitive to the duty cycle driving the car's motor. This lead us to restrict the input values to discrete values that can actually be set by the Kyosho controller. At this point our simulation results were finally satisfying. The parameter estimation produced useful results, however with the downside of having created a hybrid system, which is much harder to deal with in terms of control. Even though we were forced to describe the car as a hybrid system we achieved to create a simple model based control algorithm for tracking a given trajectory. So we reached the goals of our task.

Outlook

At the moment discrete control inputs have to be translated by a mapping function into voltage levels applied to an interface i.e. the Kyosho controller, about which we only posses very limited information. The interface then translates the applied voltage levels back into discrete control inputs, which have an insufficient resolution to be approximated by a continuous function. Since the aim of future projects will be to create and compare controlling algorithms - where hybrid systems in general are harder to deal with in terms of control - we believe that the performance of the two models could be greatly improved by the development of a new interface between the computer and the car. Such an interface should at least provide a greater resolution for the control inputs and might even contain additional sensors to create more feedback than just the car's position and orientation provided by the vision system.

Bibliography

[1] **Marc Osswald, Florian Engeler**. *Real Time Control of 1:43 scale race cars.* Master's thesis, Automatic Control Laboratory, ETHZ, June 2009.

[2] supersonic. http://www.rc-car-setup.com. Review.

[3] **Martin Rutschmann**. *Infrared based vision system for tracking 1:43 scale race cars*, June 2010.

[4] **Giancarlo Genta**. *Motor vehicle dynamics, Modeling and Simulation.* World Scientific Publishing Co. Pte. Ltd, 1997.

[5] **H. B. Pacejka**. *Tyre and vehicle dynamics.* Butterworth-Heinemann, 2nd edition, 2006.

[6] **Ryszard Andrzejewski and Jan Awrejcewicz**. *Nonlinear dynamics of a wheeled vehicle.* Springer Science+Business Media, Inc., 2005.

[7] *Estimation of Tire Cornering Stiffness Using GPS to Improve Model Based Estimation of Vehicle States. Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, 2005.

[8] *Manual for dNaNo FX Series Starter Pack.*

List of Figures

List of Tables

Appendix A

Parameter Identification Guide

This guide describes how to install the interface between the RC cars and the computer and provides a routine to identify parameters for other D-nano cars.

1 Step

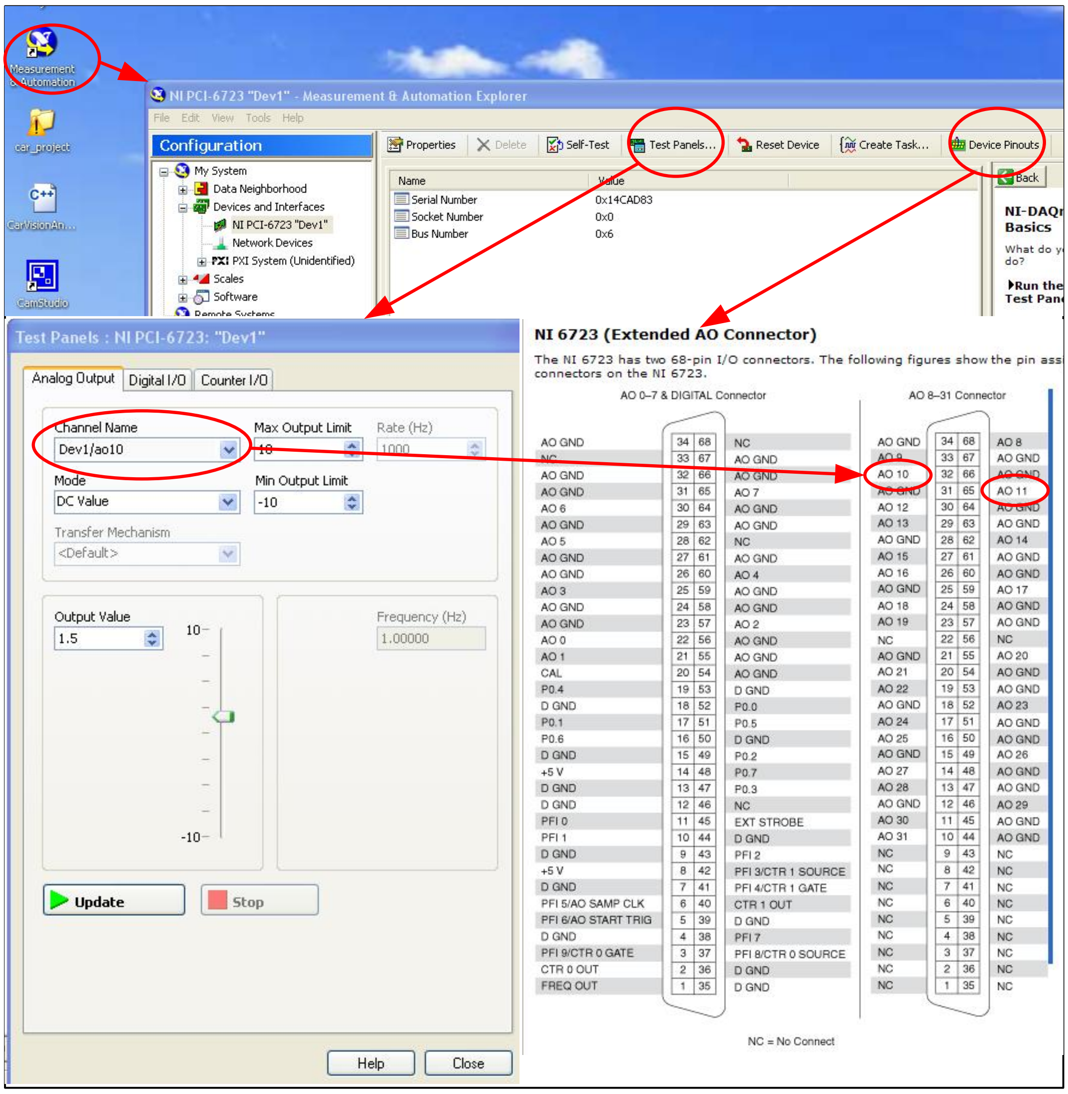
Make sure a DAC is properly installed. Connect the Kyosho remote controller to the DAC, figure A.2 shows the connections needed. Supply the board with a voltage source of $6\,V$. Connect the potentiometers pins on the remote controller to the DAC. When the remote controller is connected (not turned on) make sure the analog signals are set to $1.5\,V$, this can be achieved with the Test Panel see figure A.1.

2 Step

Reset the remote controller to its factory settings and create a connection between the remote controller and the car.
Press and hold button B, turn the remote controller on with button P, after 3 seconds release button B. Turn the remote controller off. At this point the factory settings are reset.
Press and hold button A, turn the remote controller on, once the blue LED ceases to glow brightly release button A. Turn on the car, press the connect button (see figure A.3 with a pointy object until the red LED glows brightly. Turn the car and remote controller off and on again. The controller is now connected to the car. Further information can be found in the manual [8].

3 Step

The goal of this step is to create a mapping table between the throttle voltages applied to the remote controller and the duty cycle D sent to the car. The possible values for the duty cycles are discrete and the mapping function can be represented as a step function. It is recommended to use values more or less equidistant between the steps (see figure A.4). The mapping function can be changed by trimming the



Name	Value
Serial Number	0x14CAD83
Socket Number	0x0
Bus Number	0x6

Channel Name: Dev1/ao10
Max Output Limit: 10
Rate (Hz): 1000
Mode: DC Value
Min Output Limit: -10
Transfer Mechanism: <Default>
Output Value: 1.5
Frequency (Hz): 1.00000
10
-10

NI 6723 (Extended AO Connector)

The NI 6723 has two 68-pin I/O connectors. The following figures show the pin ass
connectors on the NI 6723.

AO 0–7 & DIGITAL Connector

AO GND	34	68	NC
NC	33	67	AO GND
AO GND	32	66	AO GND
AO GND	31	65	AO 7
AO 6	30	64	AO GND
AO GND	29	63	AO GND
AO 5	28	62	NC
AO GND	27	61	AO GND
AO GND	26	60	AO 4
AO 3	25	59	AO GND
AO GND	24	58	AO GND
AO GND	23	57	AO 2
AO 0	22	56	AO GND
AO 1	21	55	AO GND
CAL	20	54	AO GND
P0.4	19	53	D GND
D GND	18	52	P0.0
P0.1	17	51	P0.5
P0.6	16	50	D GND
D GND	15	49	P0.2
+5 V	14	48	P0.7
D GND	13	47	P0.3
D GND	12	46	NC
PFI 0	11	45	EXT STROBE
PFI 1	10	44	D GND
D GND	9	43	PFI 2
+5 V	8	42	PFI 3/CTR 1 SOURCE
D GND	7	41	PFI 4/CTR 1 GATE
PFI 5/AO SAMP CLK	6	40	CTR 1 OUT
PFI 6/AO START TRIG	5	39	D GND
D GND	4	38	PFI 7
PFI 9/CTR 0 GATE	3	37	PFI 8/CTR 0 SOURCE
CTR 0 OUT	2	36	D GND
FREQ OUT	1	35	D GND

AO 8–31 Connector

AO GND	34	68	AO 8
AO 9	33	67	AO GND
AO 10	32	66	AO GND
AO GND	31	65	AO 11
AO 12	30	64	AO GND
AO 13	29	63	AO GND
AO GND	28	62	AO 14
AO 15	27	61	AO GND
AO 16	26	60	AO GND
AO GND	25	59	AO 17
AO 18	24	58	AO GND
AO 19	23	57	AO GND
NC	22	56	NC
AO GND	21	55	AO 20
AO 21	20	54	AO GND
AO 22	19	53	AO GND
AO GND	18	52	AO 23
AO 24	17	51	AO GND
AO 25	16	50	AO GND
AO GND	15	49	AO 26
AO 27	14	48	AO GND
AO 28	13	47	AO GND
AO GND	12	46	AO 29
AO 30	11	45	AO GND
AO 31	10	44	AO GND
NC	9	43	NC
NC	8	42	NC
NC	7	41	NC
NC	6	40	NC
NC	5	39	NC
NC	4	38	NC
NC	3	37	NC
NC	2	36	NC
NC	1	35	NC

NC = No Connect

Figure A.1: How to define voltages with the DAC test panel.

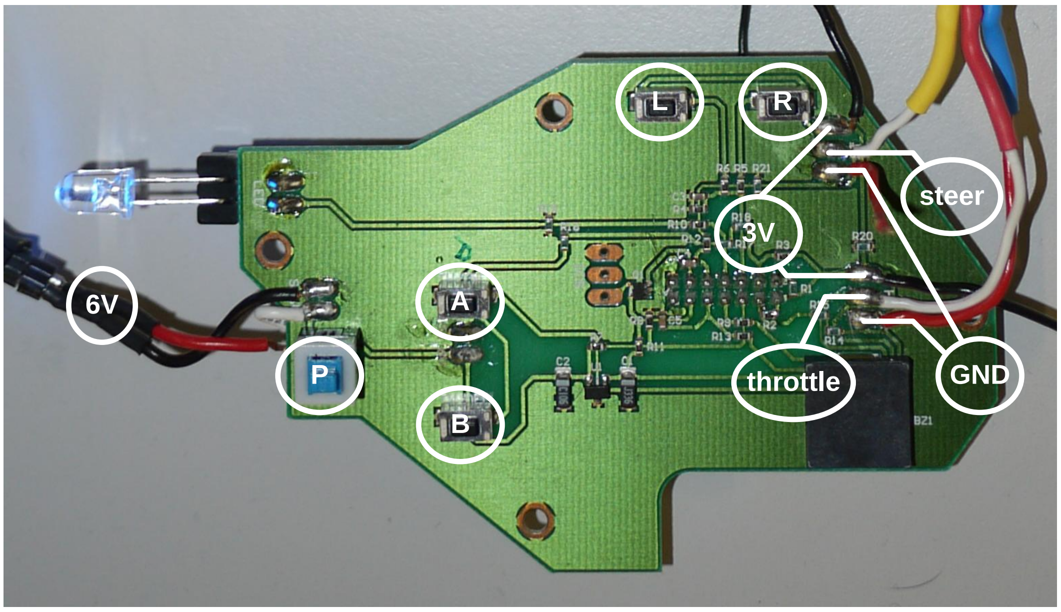


Figure A.2: The Kyosho controller board......

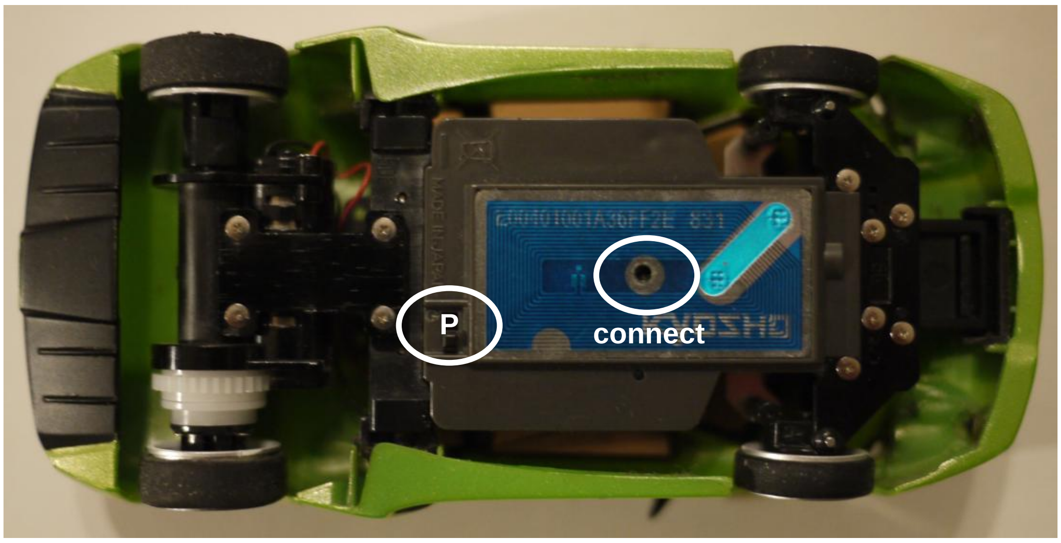


Figure A.3: Power and connect button on the car.

controller with buttons A and B (see figure A.2). Do not trim the controller after the mapping table has been measured!
The best way to measure the duty cycle is to measure the voltage at a load replacing the motor with an oscilloscope (see figure A.5).

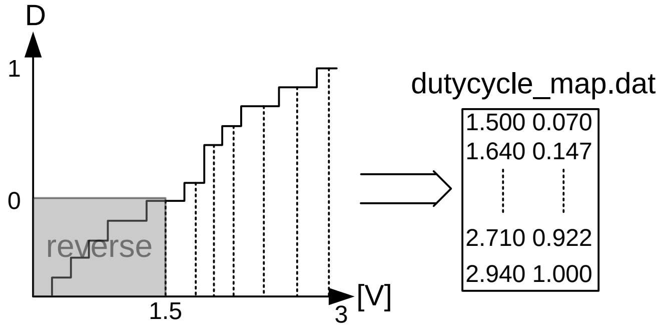


Figure A.4: Visualization of step 3 creating a duty cycle mapping table using voltage values about equidistant between the steps.

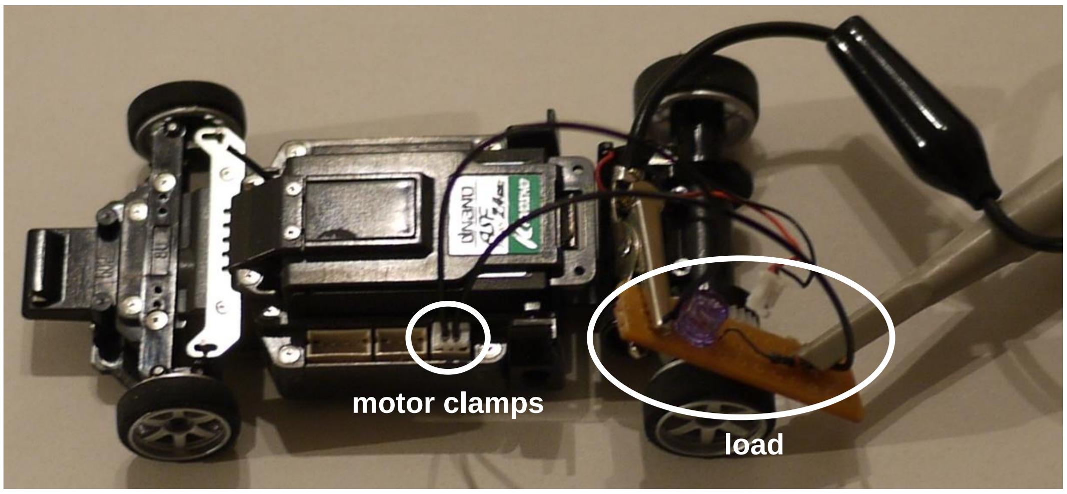


Figure A.5: A possible setup to measure the duty cycle at the motor connection clamps.

4 Step

The goal of this step is to create a mapping between the steering input voltage applied to the remote controller and the steering angle sent to the car. Similar to the throttle mapping the angle mapping is discrete, yet more difficult to measure. From the radius the car drives for a certain value of the steering voltage it is possible to extrapolate the actual steering angle. It is important that the car does not over

or under steer, therefore this experiment has to be done at low speeds.
Measure the steering voltage applied and the xy-coordinates for the following two experiments. First drive slowly with the greatest possible steering angle to the left and slowly decrease the steering voltage applied (at least one complete circle per voltage step). Repeat this experiment but start with the greatest possible steering angle to the right and increase the steering voltage applied.
Once the data has been collected execute the Matlab function

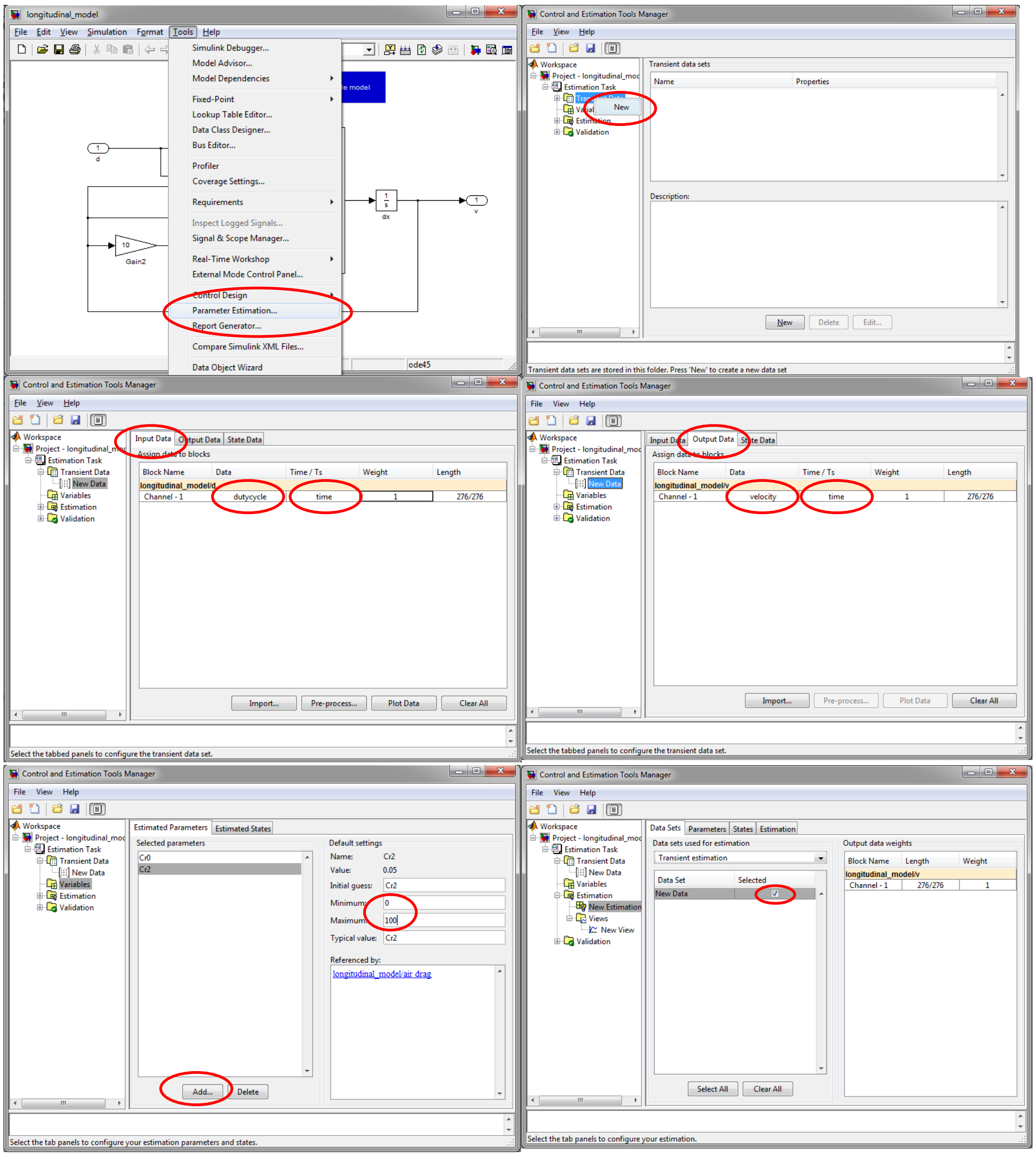
steering_map(angle_input_left,xy_data_left,angle_input_right,xy_data_right)

where angle_input_left is the steering voltage sequence applied for the first experiment and xy_data_left are the corresponding xy-coordinates and angle_input_right is the steering voltage sequence applied for the second experiment and xy_data_right are the corresponding xy-coordinates. The function returns the coefficients describing the mapping function between the steering voltage applied and the actual steering angle (see Section 3.3.2).

5 Step

Identify the models deceleration parameters by performing a deceleration experiment. Produce a data set where the duty cycle applied to the motor is zero, the car has an initial velocity and decelerates until it stops. The recorded data has to contain time, duty cycle and the cars velocity. Use the Simulink model to identify the deceleration parameters with the Parameter estimation Toolbox (see figure A.6 & A.7).

- execute initialize.m and define the vectors for time, duty cycle and velocity into the Matlab workspace, and set v0 to the initial velocity (for example v0=mean(velocity(1:4))).
- Open the Parameter Estimation Toolbox in the Simulink model longitudinal_model.mdl.
- Import the time, velocity and duty cycle data.
- Add the Parameters Cr0 and Cr2 and set their limits.
- Create a new Estimation and select the Data and Parameters.
- Create a new View to plot ”Measured and simulated”.
- Start the estimation.
- Check in the New View with Show Plots if the simulation and measurements match.



Transient data sets are stored in this folder. Press 'New' to create a new data set

Block Name	Data	Time / Ts	Weight	Length
longitudinal_model/d				
Channel - 1	dutycycle	time	1	276/276

Select the tabbed panels to configure the transient data set.

Block Name	Data	Time / Ts	Weight	Length
longitudinal_model/v				
Channel - 1	velocity	time	1	276/276

Select the tabbed panels to configure the transient data set.

Selected parameters: Cr0, Cr2

Name: Cr2
Value: 0.05
Initial guess: Cr2
Minimum: 0
Maximum: 100
Typical value: Cr2

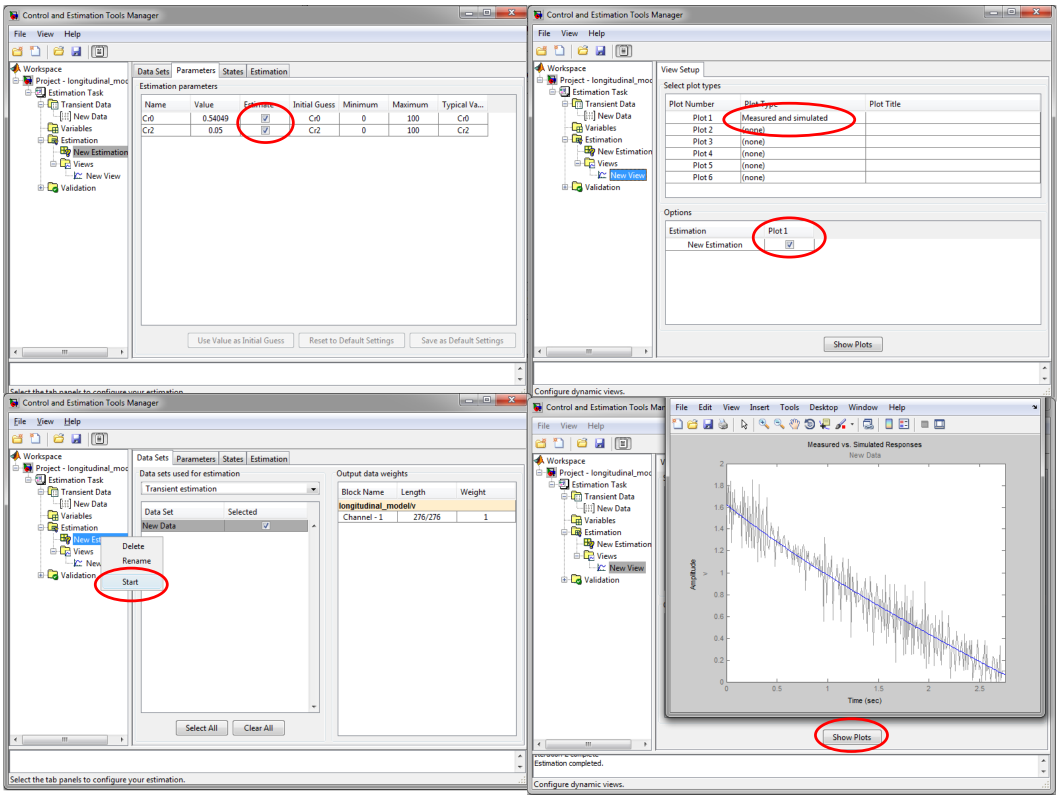
Select the tab panels to configure your estimation parameters and states.

Data Set	Selected
New Data	✓

Block Name	Length	Weight
longitudinal_model/v		
Channel - 1	276/276	1

Select the tab panels to configure your estimation.

Figure A.6: Visual guide for the parameter estimation with the Simulink model (part 1).


Control and Estimation Tools Manager

Estimation parameters

Name	Value	Estimate	Initial Guess	Minimum	Maximum	Typical Va...
Cr0	0.54049	✓	Cr0	0	100	Cr0
Cr2	0.05	✓	Cr2	0	100	Cr2

Select the tab panels to configure your estimation.

View Setup

Select plot types

Plot Number	Plot Type	Plot Title
Plot 1	Measured and simulated	
Plot 2	(none)	
Plot 3	(none)	
Plot 4	(none)	
Plot 5	(none)	
Plot 6	(none)	

Options

Estimation	Plot 1
New Estimation	✓

Configure dynamic views.

Data sets used for estimation: Transient estimation

Data Set	Selected
New Data	✓

Output data weights

Block Name	Length	Weight
longitudinal_model/v		
Channel - 1	276/276	1

Select the tab panels to configure your estimation.

Estimation completed.

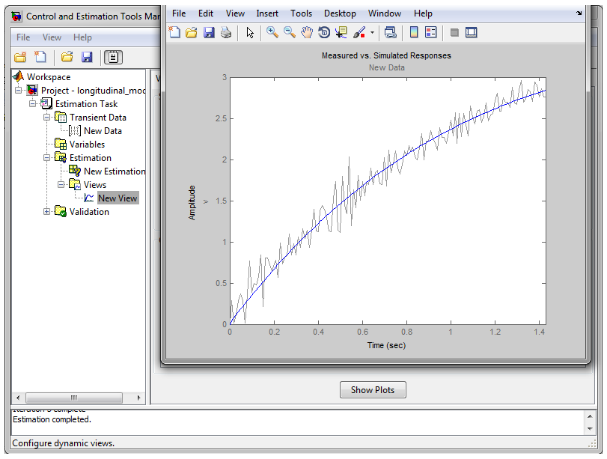
Configure dynamic views.

Figure A.7: Visual guide for the parameter estimation with the Simulink model (part 2).

6 Step

Identify the models acceleration parameters by performing an acceleration experiment. Produce a data set where the initial velocity is zero and a duty cycle step is applied to the motor. The recorded data has to contain time, duty cycle and the cars velocity. Use the Simulink model to identify the acceleration parameters with the Parameter estimation Toolbox, analog to step 5.

- execute initialize.m and define the vectors for time, duty cycle and velocity into the Matlab workspace, and set v0 zero.
- Open the Parameter Estimation Toolbox in the Simulink model longitudinal_model.mdl.
- Import the time, velocity and duty cycle data.
- Add the Parameters Cm1 and Cm2 and set their limits.
- Create a new Estimation and select the Data and Parameters.
- Create a new View to plot ”Measured and simulated”.
- Start the estimation.
- Check in the New View with Show Plots if the simulation and measurements match (see figure A.8).



Estimation completed.

Configure dynamic views.

Figure A.8: Verification if the simulation with the parameters Cm1 and Cm2 match the measurements.

Appendix B

Simulink Models

B.1 Slip free model

Usage

In order to be able to use the Simulink model you first have to set all variables defined in the Model. Use the initialize script the set the variables. The simulate script can be used to set up and execute a simulation.

Implementation notes

The implemented model differs in the following points from the theoretical model presented in chapter 4

- Instead of the geometrical constants C_1 and C_2, the lengths l, l_r and l_f where used directly (refer to figure 3.2).
- The friction parameters $C_{\mathrm{r}2}$ and $C_{\mathrm{r}0}$ are named differently:

$$\begin{aligned} C_{\mathrm{r}2} &= C_\mathrm{d} \\ C_{\mathrm{r}0} &= C_\mathrm{r} \end{aligned}$$

- To reduce numerical problems at low speed the zero order friction term is faded down to zero for small values of v

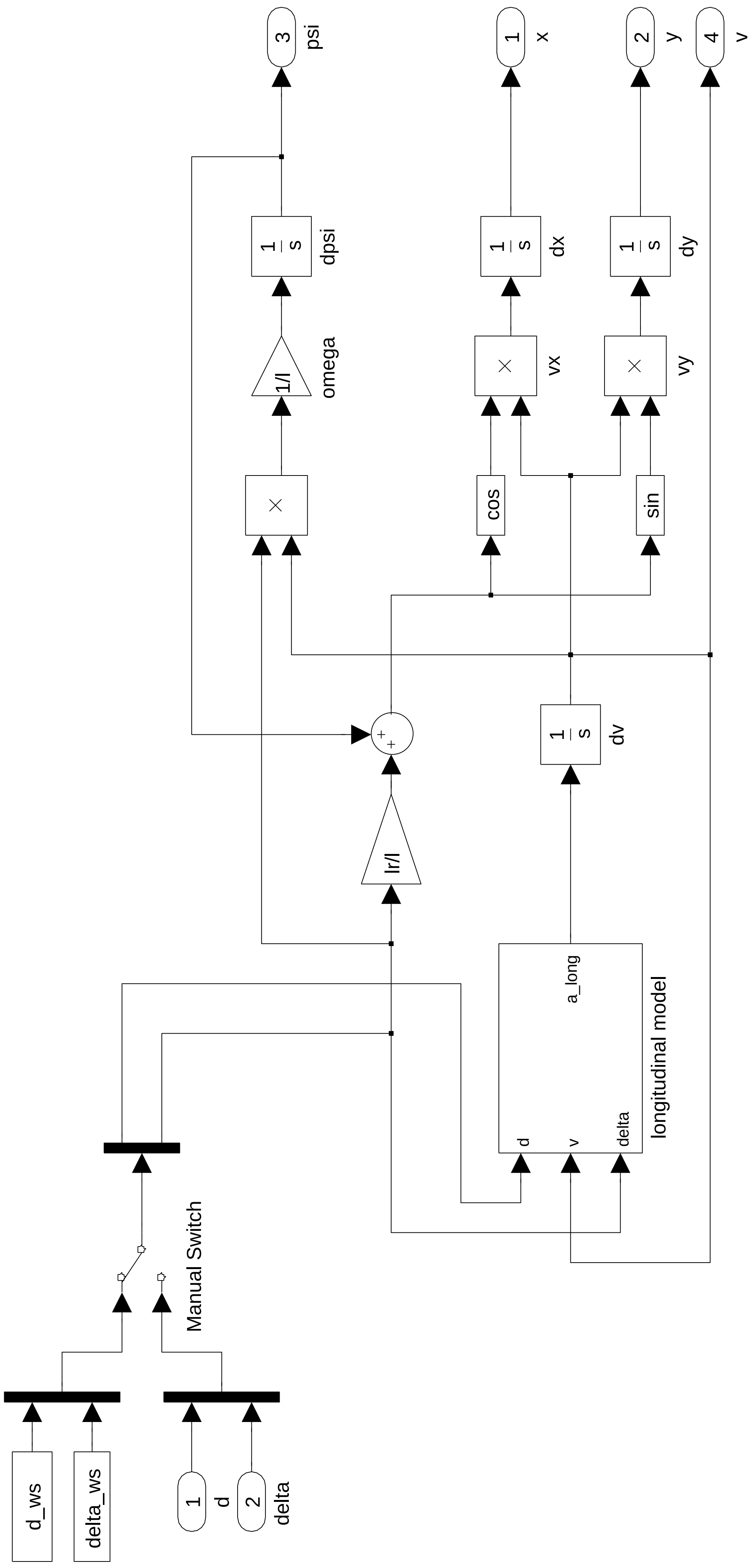
Initialize script

```
% initialize parameters
Cm1 = 11.519;
Cm2 = 2.7443;
Cd = 0.05;
Cr = 0.54049;
```

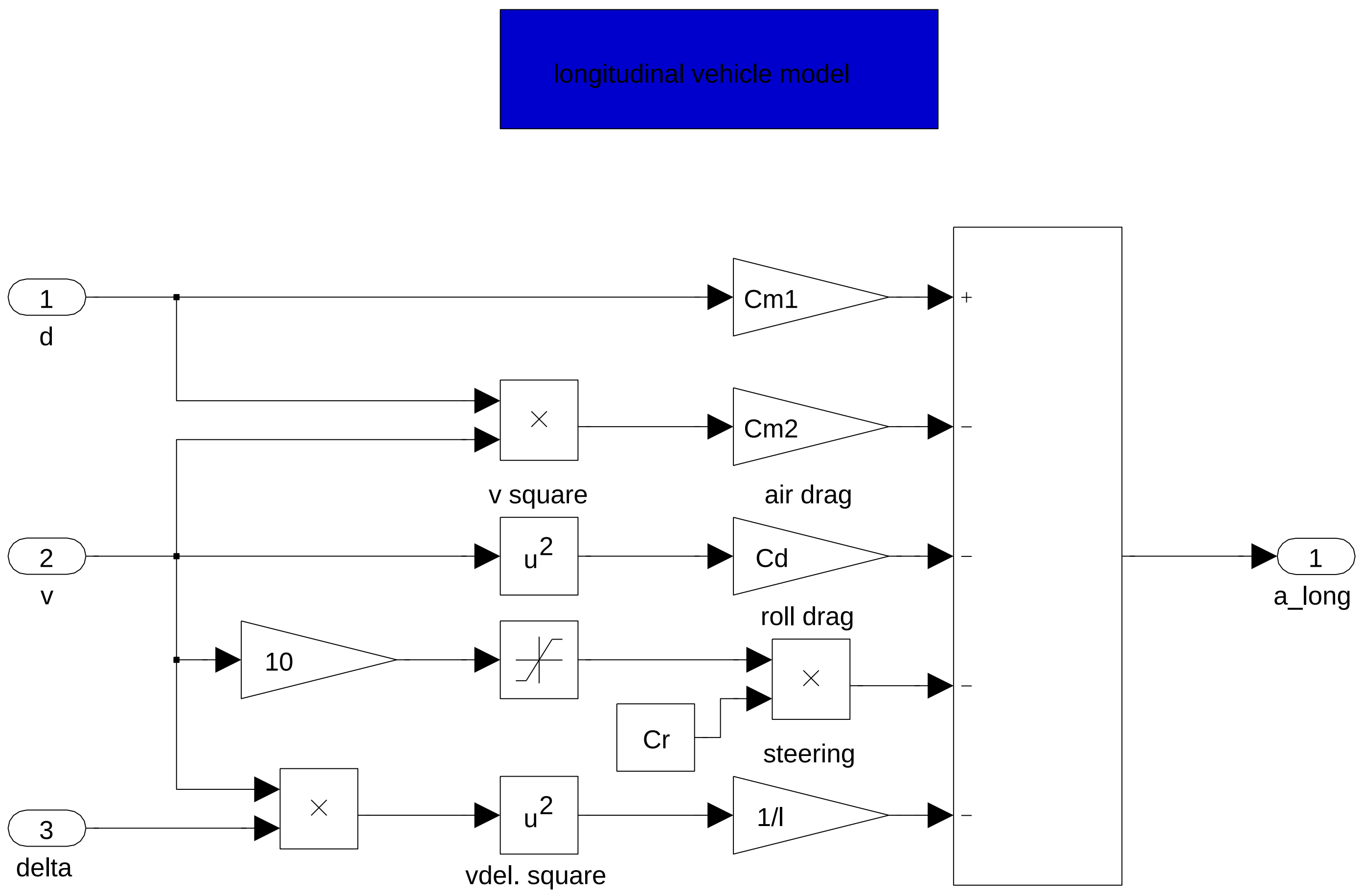
```

ksp = 0.29;
ksn = 0.34;

lr = 0.031;
lf = 0.031;
l = lr + lf;

% initialize initial conditions
x0 = 0;
y0 = 0;
psi0 = 0;
v0 = 0;

Tsim = 10;
```

Simulate script

```
initialize();

% set simulation time
Tsim = 0;

% set initial conditions
x0 = 0;
y0 = 0;
psi0 = 0;
v0 = 0;

% set input data
% d_ws:    duty cycle
% delta_ws: steering angle
%
% use the following format:
%   d_ws     = [t,d]
%   delta_ws = [t,delta]
% where t is the time vector
%
% all vectors must be column vectors
d_ws = [t,d];
delta_ws = [t,delta];

% simulate
[t,x] = sim('slipfree_model');
```



d_ws
delta_ws
1
d
2
delta
Manual Switch
longitudinal model
d
v
delta
a_long
lr/l
1/s
dv
cos
sin
1/l
omega
1/s
dpsi
vx
vy
1/s
dx
1/s
dy
3
psi
1
x
2
y
4
v


longitudinal vehicle model
1
d
2
v
3
delta
10
v square
u2
vdel. square
Cr
Cm1
Cm2
air drag
Cd
roll drag
steering
1/l
+
−
−
−
−
1
a_long

Appendix C

Trajectory tracking algorithm

pathfinder.m

```matlab

% IP-address of the PC running CarLab
server_ip = '127.0.0.1';

% TCP port for receiving car data (e.g. position) from CarLab
% (default 20101)
port_in = 14445;

% TCP port for sending control inputs (throttle and steering)
% to CarLab
% (default 20100)
port_out = 14444;

% setup a simple track
track_data(1:7,1) = [0 -1 0.5 -1  1  0  0.5];
track_data(1:7,12) = [0.5 0.4 0 0.4 -1 0  0.5];
for i = 1:10
    sinphi = sin(i/10*pi);
    cosphi = cos(i/10*pi);
    R = 0.7;
    track_data(1:2,i+1) = track_data(3:4,i);
    track_data(3:4,i+1) = [0.5;-0.3] + R*[sinphi;-cosphi];
    track_data(5:6,i+1) = track_data(3:4,i+1) - track_data(1:2,i+1);
    track_data(7,i+1) = norm(track_data(3:4,i+1) - track_data(1:2,i+1));
    track_data(5:6,i+1) = track_data(5:6,i+1) / track_data(7,i+1);

    track_data(1:2,i+12) = track_data(3:4,i+11);
    track_data(3:4,i+12) = [0;-0.3] + R*[-sinphi;cosphi];
    track_data(5:6,i+12) = track_data(3:4,i+12) - track_data(1:2,i+12);
    track_data(7,i+12) = norm(track_data(3:4,i+12) - track_data(1:2,i+12));
    track_data(5:6,i+12) = track_data(5:6,i+12) / track_data(7,i+12);
end
```

```


iCurrent = 1;
N = 22;

% define the model parameters
C1  = 0.5;
C2  = 16.129;
Cm1 = 11.519;
Cm2 = 2.7443;
Cd  = 0.05;
Cr  = 0.54;

% define controller parameter(s)
k_P = 30;
k_D = 10;
k_Pv = 3;
k_Iv = 0.025;

% data storage
data_record = zeros(11,1000);
record_index = 1;

% connect to the CarLab server
con_in = comm_connect(server_ip,port_in);
con_out = comm_connect(server_ip,port_out);

% initialize integral part of the velocity controller
ev0 = 0;

% controller loop
tic
while(record_index < 1001)

    % read car data from CarLab
    % car data is stored like this:
    % data(1): timestamp
    % data(2): x-position
    % data(3): y-position
    % data(4): psi (orientation angle)
    % data(5): velocity in x-direction (w.r.t. global coordinates)
    % data(6): velocity in y-direction (w.r.t. global coordinates)
    % data(7): omega (angular velocity)
    % data(8): throttle
    % data(9): steering angle
    data = fread(con_in,9,'float32');
    while(con_in.BytesAvailable > 0)
        data = fread(con_in,9,'float32');
    end

    if(find(isnan(data)))
        error('NaN ERROR!!!');
    end

    % store some car data in local variables to make the code
    % more readable
    rcar  = data(2:3);
    vx    = data(5);
    vy    = data(6);
    psi   = data(4);
    delta = data(9);
    v     = sqrt(vx^2 + vy^2);

    % create rotation matrix to rotate from global frame
    % to car frame (x-axis aligned with velocity
    % vector of the car)
    % NOTE: We assume the slipangle to be small
    %      (like the model does)
    R = [ cos(psi + delta*C1) sin(psi + delta*C1); ...
         -sin(psi + delta*C1) cos(psi + delta*C1)];

    % map the car to the nearest point on the track
    iStart = 1; iEnd = N;
    [rnearest,distance,iCurrent] = maptotrack(track_data,rcar);

    % calculate pseudo downhill force
    %a = (rcar - rnearest) * k_downhill/distance;
    n_downhill = (rcar - rnearest)/distance;
    n_forward = track_data(5:6,iCurrent);
    prj = dot(n_downhill,[vx;vy]);

    % velocity controller
    ev1 = 1.5-v;
    ev0 = ev0 + ev1;

    % rotate the downhill force into the car frame
    a = n_downhill*(-distance*k_P - prj*k_D) + n_forward*(k_Pv*ev1 + k_Iv*
        ev0);
    a = R*a;

    % extract x- and y-component
    ax = a(1);
    ay = a(2);

    % calculate control inputs
    if(v < 0.1)
        delta = 0;
    else
```

```
        delta = ay/(C2*v^2);
    end

    if(((Cm1 - v*Cm2)) < 0.1)
        d = 1;
    else
        d     = (ax + C2*(v*delta)^2 + v^2*Cd + Cr)/(Cm1 - v*Cm2);
    end

    delta = max(-0.3491,min(0.3491,delta));
    d = max(0,min(1,d));

    data_record(1:9,record_index) = data;
    data_record(10:11,record_index) = rnearest;
    record_index = record_index + 1;


    % send control input to CarLab

    i1 = find(input_map(:,2) >= d, 1);

    if(isempty(i1))
        i1 = length(input_map);
    end

    if(i1 == 1)
        i1 = 2;
    end

    if(delta > 0.01)
        delta = (delta - ppos(2))/ppos(1);
    else
        if(delta < -0.01)
            delta = (delta - pneg(2))/pneg(1);
        else
            delta = 1.5;
        end
    end

    if(delta > 2.8)
        delta = 2.8;
    end

    if(delta < 0.2)
        delta = 0.2;
    end

    comm_sendinput(con_out, input_map(i1-1,1), delta);

```

```
end
toc
% close all open connections
comm_close(con_in);
comm_close(con_out);

figure(1)
hold on;
axis square
axis([-1 4 -1 4]);
angle = 0:0.1:2*pi;
myones = ones(size(pi));
plot(track_data(1,:),track_data(2,:),'r')
pt_car = line(0,0,'Marker','o','Color','blue');
pt_target = line(0,0,'Marker','o','Color','red');
pt_radius = plot(cos(angle),sin(angle));

for i = 1:length(data_record)
    r = norm(data_record(2:3,i)-data_record(10:11,i));
    set(pt_car,'XData',data_record(2,i));
    set(pt_car,'YData',data_record(3,i));
    set(pt_target,'XData',data_record(10,i));
    set(pt_target,'YData',data_record(11,i));
    set(pt_radius,'XData',data_record(2,i)*myones + r*cos(angle));
    set(pt_radius,'YData',data_record(3,i)*myones + r*sin(angle));
    pause(0.01);
end
```

maptotrack.m

```
function [ r min_distance index ] = maptotrack( track_data, point )

    % number of track segments
    N = size(track_data,2);

    min_distance = 1e6;

    % loop trough all track segments and find the nearest point
    % on the track
    for i = 1:N

        % calculate the projection of the car onto the
        % track segment
        proj = dot(point - track_data(1:2,i), track_data(5:6,i));

        % if the projection is less than 0, the point does
        % not lie on the segment and the start point
        % of the segment is the nearest point
        if(proj < 0)
            rnear = track_data(1:2,i);
        % if the projection is larger than the length of
        % the segment, the end point is nearest point
        elseif(proj > track_data(7,i))
            rnear = track_data(3:4,i);
        % else the point lies on the segment
        else
            rnear = track_data(1:2,i) + track_data(5:6,i)*proj;
        end

        % calculate the distance between the track
        % and the point
        distance = norm(point - rnear);

        % find the minimal distance
        if(distance < min_distance)
            min_distance = distance;
            r = rnear;
            index = i;
        end

    end

end
```

Appendix D

CD ROM content

- Admon
  final presentation (pptx) and report (pdf)
- Models MATLAB code for the models
- Presentation final presentation and movies showed during the presentation
- Report final report and tex source
- Tools C++ and MATLAB tools for data acquisition