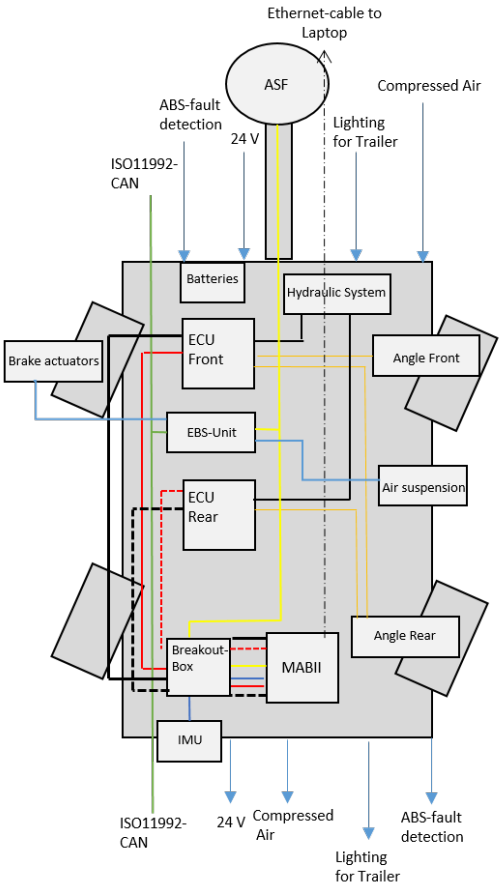
CHALMERS



Real-Time Control Interface for a Steered and Braked Converter Dolly for Long Heavy Vehicle Combinations

Master's thesis in Automotive Engineering

SEBASTIAN FRANZ
MICHAEL HOFMANN

Department of Applied Mechanics
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2015

MASTER'S THESIS IN AUTOMOTIVE ENGINEERING

Real-Time Control Interface for a Steered and Braked Converter Dolly for Long Heavy Vehicle Combinations

SEBASTIAN FRANZ
MICHAEL HOFMANN

Department of Applied Mechanics
Division of Vehicle Dynamics
Vehicle Dynamics Group
CHALMERS UNIVERSITY OF TECHNOLOGY

Göteborg, Sweden 2015

Real-Time Control Interface for a Steered and Braked Converter Dolly for Long Heavy Vehicle Combinations
SEBASTIAN FRANZ
MICHAEL HOFMANN

© SEBASTIAN FRANZ, MICHAEL HOFMANN, 2015

Master's thesis 2015:01
ISSN 1652-8557
Department of Applied Mechanics
Division of Vehicle Dynamics
Vehicle Dynamics Group
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: +46 (0)31-772 1000

Cover:
Some explanation

Chalmers Reproservice
Göteborg, Sweden 2015

Real-Time Control Interface for a Steered and Braked Converter Dolly for Long Heavy Vehicle Combinations
Master's thesis in Automotive Engineering
SEBASTIAN FRANZ
MICHAEL HOFMANN
Department of Applied Mechanics
Division of Vehicle Dynamics
Vehicle Dynamics Group
Chalmers University of Technology

Abstract

Keywords: Some stuff, More stuff, Stuff

Contents

Abstract **i**

Contents **iii**

1 Introduction (3 Seiten) **1**
1.1 Purpose . . . 1
1.2 Objectives . . . 1
1.3 Limitations . . . 2
1.4 Structure of this thesis . . . 2

2 Overview (6-7 Seiten) **4**
2.1 Background of long vehicle combinations (LVC) . . . 4
2.2 Ongoing research . . . 5
2.3 Market overview for existing solutions . . . 6
2.4 Rapid Control Prototyping . . . 6

3 Vehicle Model controllers (3 Seiten) **9**
3.1 Low-Speed controller . . . 9
3.1.1 Input parameters . . . 10
3.2 High-Speed controller . . . 10
3.2.1 Input parameters . . . 10
3.3 Overview of the model . . . 10
3.4 Real-Time implementation of controllers . . . 11
3.5 Interface with Real-Time environment . . . 11

4 Hardware architecture(9 Seiten) **13**
4.1 Hardware overview . . . 13
4.2 Utilized dolly system . . . 14
4.3 Real-Time Environment . . . 14
4.3.1 CAN-bus extension . . . 15
4.4 Interfaces and connections with dolly . . . 16
4.5 Interfaces with truck . . . 17
4.6 Measurment Setup . . . 17
4.6.1 On-board sensors . . . 17
4.6.2 Inertial measurement unit . . . 17
4.6.3 Arduino Due . . . 18

5 Software Architecture (8 Seiten) **24**
5.1 Functionality architecture . . . 24
5.2 Matlab/Simulink environment . . . 24
5.2.1 dSpace RTI-blockset . . . 24
5.2.2 Virtual Truck Modelling Library . . . 26
5.2.3 CAN-bus extension . . . 26
5.3 ControlDesk monitoring environment . . . 26

5.3.1 Maneuver control . . . 26
5.3.2 Monitoring and logging . . . 26
5.4 Arduino IDE and applications . . . 26

6 Fault detection and system ability (5 Seiten) 28
6.1 Failure Mode and Effects Analysis (FMEA) . . . 28
6.2 Safety concepts . . . 28
6.3 Maximum capabilities of the system . . . 28
6.4 Warning and state-info system . . . 29

7 Verification and validation of steering (2-3 Seiten) 30
7.1 Overview . . . 30
7.2 Bench-Testing . . . 30
7.2.1 ECU-setup . . . 30
7.2.2 VTM maneuver verification . . . 30
7.2.3 CAN verification . . . 30
7.2.4 Fault detection system verification . . . 30
7.3 Processing time evaluation . . . 30
7.3.1 Background . . . 30
7.3.2 Measured input delay . . . 31
7.3.3 Delays in logging measuring chain . . . 33
7.4 Vehicle testing . . . 34
7.4.1 System calibration . . . 34
7.4.2 Actuator tests . . . 34
7.4.3 Algorithm evaluation . . . 34
7.4.4 Sensor testing . . . 34
7.5 Hardware-in-the-loop testing . . . 34
7.5.1 Low-speed controller . . . 35
7.5.2 High-speed controller . . . 35
7.6 Track testing . . . 35
7.6.1 Testmaneuvers . . . 35
7.6.2 Testenvironment AstaZero . . . 36
7.6.3 Testmatrix . . . 36
7.6.4 Test setup and instrumentation . . . 36

8 Discussion (3-4 Seiten) 37
8.1 Results from bench testing . . . 37
8.2 Results from processing time evaluation . . . 37
8.3 Results from in vehicle testing . . . 37
8.4 Results from hardware-in-the-loop testing . . . 37
8.5 Comparison . . . 38

9 Conclusion (2 Seiten) 39
9.1 Recommendation . . . 39
9.2 Future Work . . . 39

APPENDICES 40

A Appendix **40**

References **42**

1 Introduction (3 Seiten)

1.1 Purpose

Heavy goods-transport on the road has constantly increased over the last decades. Coupled with the stricter environmental regulations concerning CO_2-emissions and pollution, the call for more economical transport solution has led to the wider introduction of long combination vehicles. Those truck-trailer combinations have a longer history in geographical areas with low population density, mining and transport within factory sites where rail-road transport is not a viable option but transportation of large volumes and tonnages are called for. The prospects of saving costs on driver's salarys, reduced fuel consumption and decreased costs suggests the introduction of those combinations in other environments as well. Introduction of a new vehicle class leads to many challenges in safety, research and development, and legislation.

The driving behaviour of LCVs is in many ways different to that of standard trucks and needs to be researched in great detail to gain an understanding of the vehicle's dynamic properties, that is equally detailed as it is for other vehicle classes. This will lead to development of better safety and assistance systems and thus reduce threat potential, accidents and fatalities involving this emerging mode of transportation. Different usage patterns of LCVs have to be considered as well, when developing functions for LCVs. For example inner-city use is no prevalent use-case for LCVs, whereas highway safety features and handling properties at higher speeds are prime goals due to high percentage of highway-driving for LCVs. Nevertheless maneuverability for docking is also a development goal.

Besides the technical implementation, socio-economic aspects have to be considered. Legislation has to be adjusted to allow for longer vehicle classes, including new certification processes and driver training. Furthermore infrastructure might have to be modified or reviewed to accommodate the needs and dimensions of extended truck combinations.

The research project in which this thesis is embedded aims to develop an active dolly, meaning that steering will be autonomously conducted by the dolly based on the driving situation at hand and various vehicle parameters (e.g. speed, steering wheel angle). Furtheron braking capabilities are to be implemented to act in a similiar fashion as an electronic stability control system (ESC) by creating a yaw-moment countering undesired vehicle movements. This counter-stearing will be achieved through wheel-individual brake-application.

This high-level control algorithm will be executed on a rapid-prototyping system which is linked to and controls the dolly. To supply this connection between the hardware and control-algorithm implemented in the modeling-environment Simulink is the main-task of this thesis.

1.2 Objectives

The main-goals that are supposed to be achieved within this thesis' scope of work are:

- Develop a software interface for the high-level control algorithm implemented in Simulink to be run on a rapid-prototyping system to control the steering system on an active dolly.
- Develop the physical hardware interface with the dolly; establish a suitable environment connection for the rapid-prototyping system on-board of the dolly

- Develop a measuring solution to determine the processing delays in the sensing system as well as the delays introduced by computation and actuator reaction times. Determine and try to minimize these occuring delays.
- Verification of designed systems through different stages of hardware-in-the-loop tests (HIL)
- Develop a safety system that continously monitors the active steering system, prevents malicious inputs and triggers necessary warnings.
- Develop an interface that allows to contiously determine the system's maximum actuation capabilities depending on the system's current properties (axle load, speed, steering angle, yaw-behaviour). Utilize a vehicle dynamics model to find out the critical boundries for LCV's behaviour.
- prepare on-track testing (design road-ready measuring equipment, start-up procedure, repeatable pre-recorded maneuver decription for automated testing)

1.3 Limitations

To implement the actual high-level algorithm to compute the desired angle for the dolly's steerable axles is not in the scope of this thesis. It was developed in the research project in which this thesis is embedded. Nevertheless to establish an easier insight into the interfaces' parameters, an overview of the structure, in- and outputs of the underlying computational steering model is needed and shall be presented in chapter 3.

The hardware- and low-level control-system of the hydraulic actuators is in place already and thus will not be part of this thesis. It is supplied as turn-key software by the manufacturer and readily available on the dolly's electrical control units (ECU). Substantial modifications are necessary to achieve the desired goals, though. The ECU's software version will be available fully calibrated and parametrized for the dolly at hand and thus provide a reliable working base to build upon.

Braking the dolly's axles individually will not be part of the testing and development scope of this work. Nevertheless the conceptual solution for including this functionalities furtheron will be outlined and a foundation for sending actuation request to these systems will be included in the rapid-protoyping interface block resulting from this thesis.

1.4 Structure of this thesis

In the first two sections of this thesis a brief overview of the legal situation concerning LCV for different countries, the current state of the art and ongoing research in the field of LCV shall be presented (chapter 2). Furthermore an introduction to the model, that will be run on the rapid-prototyping system will be given (chapter 3). Those two chapters are meant to give an introduction into the matter and are mainly based on literature review.

In the succeeding chapters the conducted development work will be described in detail. As a loose guiding structure for this work the development process after the V-model (see 2.4) will be referred to. Starting with a description of the utilizied hardware-systems and

their interconnections in chapter 4, followed by detailing the different software-tools and environments running on those hardware-platforms in chapter 5. In chapter 7.3 the measuring concepts and theoretical details for the determination of the overall processing delays in the control-chain will be discussed. As at the planned high speeds and great inertia for testing safety is a major concern, safety functions will be implemented and systematically evaluated. This will be outlined and discussed in chapter 6. Aftet the development work was described the created (sub-)systems will subsequently be validated and peu-à-peu put together and following the V-model evaluated at different stages of integration (chapter 7).

The work closes with a discussion of the results collected during testing and a conclussion where the authors will try to give recommendation for pratical implemenation and outline future research work in the field.

2 Overview (6-7 Seiten)

2.1 Background of long vehicle combinations (LVC)

In countries of the EU the permitted maximum length of a truck-trailer combination is 18.75 m and its maximum weight is 44 tonnes. There is however the possibility for countries to make exceptions from that rule.[26] For example in Sweden and Finland road trains can be up to 25.25 m long with a maximum weight of 60 tonnes.[27] In several other countries of the EU LVC are allowed on certain roads for testing purposes. Table 2.1 shows the maximum length and weight of LVC in different countries.

Table 2.1: LVC in different countries[27][25][1][9][4]

Country	Max. Length [m]	Max. Weight [t]
EU (general)	18.35	44.00
Sweden	25.25	60.00
Finland	25.25	60.00
Australia	53.50	132.00
USA(trailers without truck)	26.07	59.86
Canada	36.88	63.50
Mexico	31.00	75.50

In Sweden there are attempts to have LVC that exceed the current dimensions. One example is an A-double combination, which consists of a tractor, two semitrailers and a dolly between the semitrailers. This type of combination was used for the simulations and hardware-in-the-loop tests conducted in this project.

There are certain regulations for LVC in terms of driver requirements. In countries of the European Union a person who drives a LVC is required to have the CE drivers license, which allows you to drive a tractor with a mass over 3.5 tonnes with a trailer, whose mass is over 750 kg. To be able to acquire that drivers license the person has to be over 21 years old. [15] On top of that, there have been more requirements for the drivers that took part in trials of LVC in European countries. For example for the trials in the Netherlands drivers were required to have their drivers license for at least five years, to not have lost it during the last three years and they had to acquire a special certificate. To get that certificate the drivers had to take theoretical as well as practical courses. [**LVC˙test˙netherlands**]

Road wear is a big concern when it comes to road transport. The main factor of road wear is the axle load of the vehicles. The following equation for a load equivalency factor shows how different axle loads impact road wear.

$$\frac{N_{ref}}{N_x} = \left(\frac{W_x}{W_{ref}}\right)^4 \tag{2.1}$$

Where W_x and W_{ref} are axle loads and N_x and N_{ref} are the corresponding numbers of load applications.[**road˙wear**] Equation (2.1) shows that for example doubling the axle load leads to a road wear that is 16 times higher.

Assuming an equal distribution of the load over the different axles, the axle load for a vehicle is calculated as follows:

$$W_x = \frac{m_{tot}}{n_{axl}} \tag{2.2}$$

Where m_{tot} is the total mass of the vehicle and n_{axl} is the numbers of axles. A standard tractor-semitrailer combination has five or six axles and a maximum mass of 40 tonnes. With equation (2.2) the axle load, if fully loaded, for that combination is 8 tonnes/axle and 6.67 tonnes/axle, respectively. An A-double-combination, which has ten or eleven axles and a maximum mass of 80 tonnes, also reaches a maximum axle load of 8 tonnes/axle and 7.27 tonnes/axle, respectively. For a 25.25 m combination, which has nine or ten axles and a maximum mass of 60 tonnes, the maximum axle load is even lower than for the standard combination. The maximum axle load for this case is 6.67 tonnes/axle and 6 tonnes/axle, respectively.
Based on that it can be concluded that high capacity transport vehicles don't lead to higher road wear than standard combinations.

2.2 Ongoing research

There is a lot of research going on concerning high capacity transport vehicles. The main topics are the safety of HCT, their environmental impact as well as economical aspects.

There are many concerns about the safety of HCT in public opinion, so there have been quite a lot studies that investigated this issue. One concern is the extended length of HCT, which causes an increase of time needed to overtake a combination. Studies showed that accident risk for overtaking a long combination is not statistically higher than for standard combinations. [**EMS**] For overall safety, the majority of studies show that the use of HCT increases traffic safety due to the reduced number of vehicles on the roads, which is induced by the higher capacity of HCT. However, HCT might have a higher accident rate for individual vehicles. [**EMS**] The analysis of crash data from the Swedish Traffic Accident Data Acquisition showed that there is no higher rate of fatal or severe crashes per traveled vehicle km for LVC compared to standard vehicle combinations. [3]

To reduce the effects on safety, which might occur with the use of HCT, different safety systems are being developed.

One aim of safety systems for HCT is to reduce the rearward amplification of yaw rate and lateral acceleration. A possibility to achieve that is to use a steerable dolly. In section 3.2 a controller to reduce rearward amplification for high speed is described.
Another problem of HCT is the off-tracking of the last trailer for low speed cornering. Off-tracking means that the trailer doesn't follow the path of the tractor but moves on a radius that is smaller then the tractors. This issue can also be addressed by using a steerable dolly. In section 3.1 a controller to improve the path-following is described.

One reason for the use of HCT is the expected positive influence on the environment compared with standard combinations. Studies showed that with the use of LVC the environmental impact of goods transportation can be reduce significantly for high loading factors. The fuel consumption is about 15% lower compared to the standard vehicle combinations used in the EU. In the number of trips a reduction of 32% can be seen. This reduction is achieved because

two LVC can carry the same amount of goods as three standard vehicle combinations. [2] However, the environmental impact of LVC can become negative if due to a high penetration of LVC goods transportation is shifted from rail to the road. [7]

Probably the main reason for introduction of HCT is their economical advantage. First of all, the transport costs decrease by 1.8-3.4%. Together with the environmental costs, that are lower compared to standard combinations, an overall benefit of HCT can be seen.[**EMS**]

2.3 Market overview for existing solutions

There are some solutions for dollys with steerable axles that exist today. However, the steering angles of the axles is only based on the articulation angle between the dolly and the trailer or truck it is attached to.
One of these solutions is the "active articulated dolly" manufactured by "Fahrzeugwerk Bernard Krone". The front axle of this dolly is steerable. In order to achieve that, the drawbar of the dolly is mechanically connected to the frontaxle and is rotateable. So if the drawbar is turned the wheels of the front axle steer. The transmission of the steering can be varied by changing the length of the drawbar, which is adjustable. For safety reasons the steering is mechanically locked automatically depending on the velocity. The main reason for implementing this steering capability is to reduce off-tracking. This dolly is equipped with air suspension, an electronic breaking system as well as a stability system. [**Krone˙dolly**]
Other manufacturers that offer steerable dollys, that work the same way, are "Schmitz Cargobull" and "Kögel Trailer". [**Kogel˙dolly**][**Schmitz˙dolly**]
In order to use the dolly to improve the vehicle dynamics of a combination in a more advanced way, a solution that allows more independently steering is preferable. The dolly used in this project is capable of steering both front and rear axles individually and independent of the articulation angle between the dolly and the trailer in front of it. In section 4.2 this system is described in detail.

2.4 Rapid Control Prototyping

In automotive software development standardized processes are applied to assure structured work. The V-model which derives its name from the characteristic shape constitutes one graphic representations of the sequence of steps towards the finished overall system. It also gives an approximate impression about the level of detail of each of the steps on the vertical axle, as well as chronological progress on the horizontal. A simplified version of this V-model can be seen in figure 2.1. First the user or customer needs are analyzed and put into a schematic formulation of the system showing the logic dependencies and underlying physical working principles. This is usually done in a semi-standardized fashion utilizing Unified Markup Language (UML) or similiar diagrams, preventing non-specific prose to ensure less room for mis-communication and better readability. Technical decisions and limitations are not yet considered. After the logical system and dependencies have been established they will be analyzed and broken down to actual technical system descriptions defining also which subsystem will take over which function, what parts are to be realised as software or hardware (e.g. filtering, safety functions). The different parts of the technical system are then specified in detail, defining interfaces between different software sub-system, operational states and distribution of functions over sub-systems.

The fourth step will define the specifications for the software in details for example: data types, control sequences, decision structures, real-time behaviour. Finally the actual design step will implement these specification with regards to the hardware limitations (RAM/ROM, processing power), computation methods, data handling (detailed format, variable types, utilization of parameters or variables). To compile the system from the component-level sub-system a similar hierachy is used in inverse. The corresponding steps of the project definition and specification phase (left side of the schematic) are supposed to provide test-procedures and goals. If necessary after insufficient test outcomes, another iteration of the previous step will be performed (vertical iterations). A similar sequence of development steps is applied to develop the hardware platform in parallel.[20]

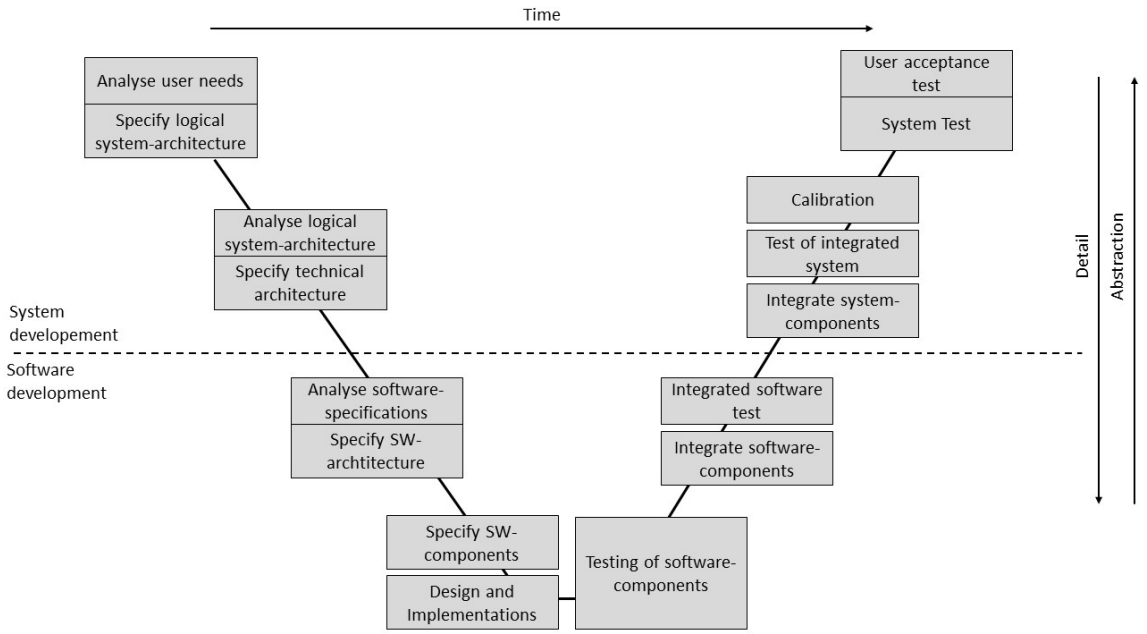


Figure 2.1: Overview for a process in automotive software development after the V-model

The increasing complexity of mechatronic systems and shorter product life-cycles lead to new development methods that made it possible to take 'short-cuts' in the established V-model process. This methods are summarized under the term rapid prototyping - in the case of software functions for mechatronic systems the term rapid control prototyping (RCP) was coined. It is possible to conduct testing and verification with early soft- and hardware versions which are still under development by having the remaining system and environmental influences simulated by the RCP-platform. In utilizing these new methods such as rapid control prototyping and software/hardware-in-the-loop testing, a tremendous decrease in development time can be achieved. It is also possible to validate specifications early in the process, eliminating possible cost-intensive changes in later stages.[19] For the research project in which this theses is embedded it even opens up the possibility of on-road testing, as it eliminates the low-level development steps of the actual implementation and some of the detailed design work. The implementation of the steering algorithm (see section 3) on a stand-alone ECU, hydraulic actuator control, hardware layout, etc. would by far exceed the dimensions of a research project. Applying rapid control prototyping is the only feasible way to establish a functioning on-road prototype vessel.

- V-model
- real-time capability
- abstract high level model to programm ==¿ focus on function development and modelling, no low level coding needed
- time critical processes
- extensive lecture notes from [19]

3 Vehicle Model controllers (3 Seiten)

There are two different controllers used to control the steerable axles of the dolly. One is used for low-speed and one for high-speed maneuvers. The main purpose of the low-speed controller is the reduction of off-tracking. For the high-speed controller the aim is to reduce the rearward amplification.

3.1 Low-Speed controller

The low-speed controller is implemented in path-distance domain η. The path-distance is calculated by integrating the forward speed $v_{x,1}(t)$.
The steering angle of the first axle of the dolly is calculated as follows:

$$\delta_{31}(\eta) = k_1\delta_{11}(\eta - \eta_1) + k_2\Delta\psi_1(\eta - \eta_2) + k_3\Delta\psi_2(\eta - \eta_3) + k_4\Delta\psi_3(\eta - \eta_4) \tag{3.1}$$

The steering angle of the second axle of the dolly is calculated using the steering angle of the first dolly axle as follows:

$$\delta_{32}(\eta) = k_5\delta_{31}(\eta - \eta_5) \tag{3.2}$$

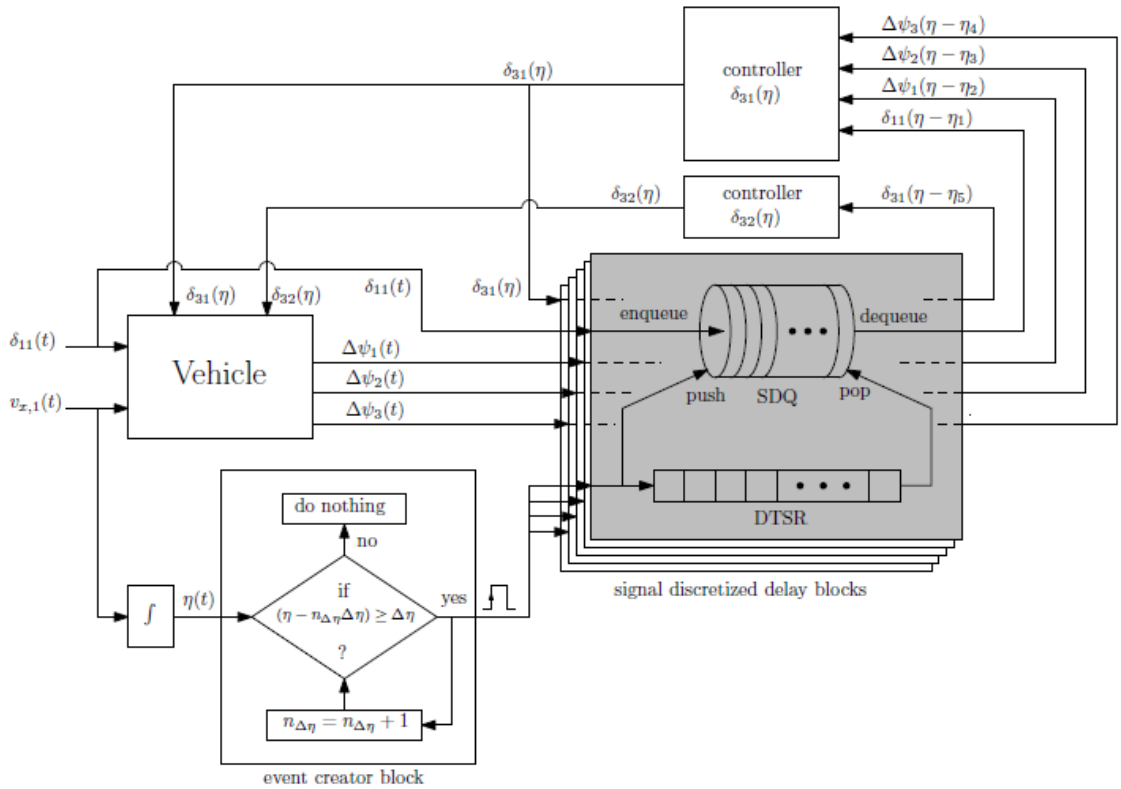


Figure 3.1: Controller implementation in the functional architecture [10]

- controller uses path-distance domain instead of time domain
- path distance is calculated by integrating vx
- optimization of gains using particle swarm optimization
- path distance in time domain transformation needed

3.1.1 Input parameters

- for delta 31: delta phi 1-3 (n) + delta11(n)
- for delta 32: delta31

3.2 High-Speed controller

- The controller is implemented using a nonlinear inverse of a dynamic vehicle model.
- same path of all units –¿ time delay of units considered
- lateral acceleration of dolly(t) = lat. accel. of truck(t-tau)
- modelica used because it can do inverse models
- imported to simulink as a functional mock-up unit (FMU) using FMI-toolbox

3.2.1 Input parameters

  - ay3ref = delayed ay1
  - ay3; delta11, vx
  - derivative of delta11 needed for inversion operation
  - explain feedback/inverse path

3.3 Overview of the model

- based on lit from MI-paper
- overview graph of model
- single track model
- filtering of inputs?
- feedback-loop
- start condition
- what can be concluded with the model?

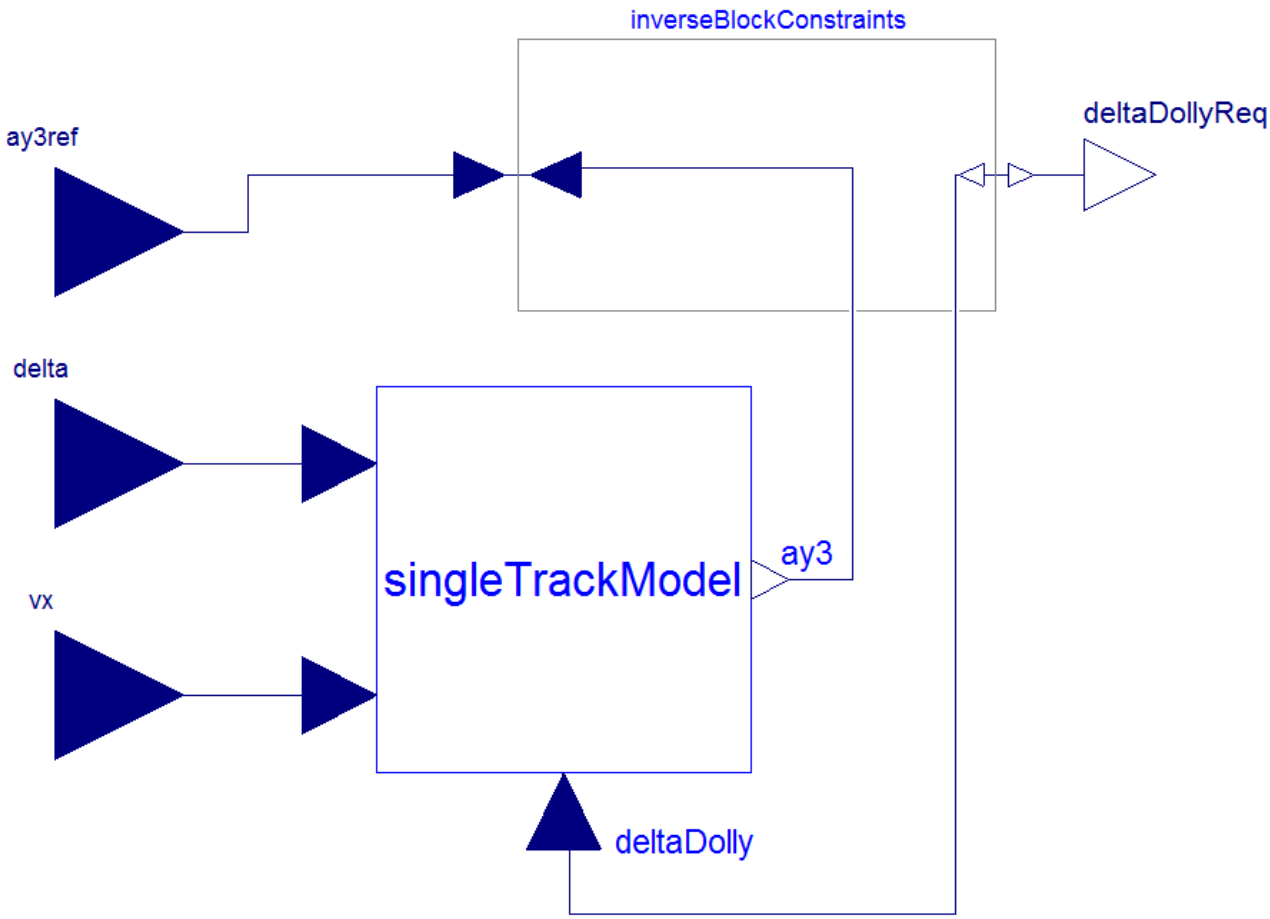


Figure 3.2: Inverse of the single track A-double model [11]

3.4 Real-Time implementation of controllers

- this paragraph in close cooperation with MI
- what had to be changed to allow for MABII execution?
- how will feedback loop be handled? item incorporate measurings?
- simulation step size?
- utilized computational method

3.5 Interface with Real-Time environment

- capabilities of system (see section 6.3)
- signals passed forward from environment + restriction
- actuator signals
- signal modification (correct frequency for CAN)

- safety flags/signals

4 Hardware architecture(9 Seiten)

4.1 Hardware overview

As already outlined in chapter 2 the LCV consists of a truck with two semi-trailers. To achieve the goals of this thesis, some modifications had to be one to the combination as well as equipment mounted in addition to the legacy fixtures. The utilized combination can be seen in figure 4.1, the shown number will be referenced further-on to give easier insight into the implemented systems' mounting points.

The first semi-trailer's (number 2 in figure 4.1) rear needs to be equipped with a drawbar eye to accommodate the dolly's drawbar. Besides this obvious physical requirement power supply for dolly and second trailer, pressurized air hoses for brake actuation, e-brake CAN-buses and lighting connectors have to be routed through the first trailer and supplied to the second trailer with a robust enough cable. This solution had to mainly be custom-made, as these appliances usually would not be available in the rear of a semi-trailer as usually no further towed unit is present. These installations would also apply for any passive combination or for the legacy VSE ETS system. Furthermore CAN networks have to be patched through the first semi-trailers (see section 4.5). The Real-Time system will be mounted on the dolly directly (position 3) to ensure short signal pathes and minimize wiring efforts. However the Host-PC will have to sit in the tractor (position 1) to sufficiently protect it against the track environment, which would be impossible if sitting on the dolly directly. This leads to an additional ethernet-cable for the Host-PC connection in the cabin which also needs to be routed through the first semi-trailer. The second semi-trailer does not require these modifications, as it is merely a passively towed unit.

Figure 4.1 also depicts another important aspect. Round bullets on the unit-connecting lines indicate a connection with one rotational degree of freedom (DOF) square bullets mean fixed connection. As can be gathered from the figure there are three DOF in the whole combination: between tractor and firstsemi-trailer, between first-semitrailer and the dolly, and between the dolly and the second semi-trailer. As the angle between the different units greatly influences the steering performance of combination, the angles have to be accounted for in the simulation and of course also the steering algorithm. It is thus necessary to measure them and provide the measurings via CAN to the rapid-prototyping system which is described in great detail in section 3.5 for the model-side and in section 4.5 for the hardware details. These articulation angle sensors also need to be wired from their mounting point to the dolly, where they are fed into the RapidPrototyping environment.

- give overview of system on truck
- sketch for where which unit sits (with numbers to be referenced)
- different tasks for Arduino
- make communication diagramm with all protocols (CAN, i2c, SPI, serial, ethernet) <== this could also be moved to software chapter?

4.2 Utilized dolly system

The utilized dolly by Parator Industri AB (Parator) is equipped with two steerable axles. They are controlled by an after-market solution called ETS Electronic Trailer Steering System (ETS) supplied by V.S.E. Vehicle Systems Engineering B.V. (VSE), of which figure 4.2 gives an overview. Their product includes sensors, ECUs and hydraulic systems which come in a ready-to-mount housing, which is placed on the trailer/dolly. This solution is usually sold as a low-speed active steering system for truck-trailer combinations to provide better maneuverability at low speeds in inner-city areas. Besides electrical power and compressed-air (see "2" in the figure) supply there is no connection with the truck in place. This allows for use with many different truck/trailer original equipment manufacturers (OEM), as no insight into proprietary Controler Area Network communication (CAN) is needed. In the original VSE system the two parameters that influence the actuation of the dolly's steering are vehicle speed ("4") and kingpin-deflection ("3"). This deflection is the angle between the truck and trailer, which is measured by an additional kingpin-angle sensor supplied by VSE and mounted in the eye of the kingpin hub. Furthermore every steering-knuckle of the dolly is equipped with an angle sensor to provide appropriate wheel-individual feedback for the VSE control-system. A diagnosis screen is available in the VSE-unit, which allows for relatively simple set-up, calibration and parametrization to be done.[5]

VSE provides assisted steering up to a speed of 25km/h over which intervention until it reaches zero at 55km/h. At this treshhold the steerable axles are locked and thus behave like normal rigid axles. This according to the manufacturer is to ensure stability at higher speeds.[5] Locking the steering at higher speeds leads to a more predictable behaviour for the user and system robustness. However, performance during high speed maneuvers can be improved by uniformly steering the dolly as well.[12] The desired demonstration of the algorithm presented in chapter 3 will as outlined in the introduction to this thesis, include steering at higher speeds, thus a work-around had to be established.

Short overview for the dolly, including:

- mechanical properties in short (weight, turning radius, max. tonnage)
- description of function (brake, steer, countersteer, lock at highspeed)
- difference low <=> high speed
- control system by VSE, diagnosis, display connection with truck
- system overview picture/schematics

4.3 Real-Time Environment

- mechanical properties of MABII (dimension, currents, mounting points in dolly)
- computational power/limitations
- explain interfaces with truck/dolly (abstract)
- explain technical realisation of HW interface (ZIF)

- explain rapid-prototyping
- robustness
- programming with software ref to 5.2
- runtime interface ref to 5.3

4.3.1 CAN-bus extension

The MABII comes with a preset number of in- and output ports. The maximum number of CAN-buses, that can be connected to the MABII's native controllers is limited to six. As there is no off-the-shelf solution by dSpace for extending the available bus-connections, it was necessary to come up with a gateway solution that allows to patch the needed amount of additional CAN-buses through to the ethernet connection which is also available on the MABII.

To do so, a gateway from CAN-protocol to the standardized User Datagram protocol (UDP) used for communication on ethernet infrastructure was implemented. It is a very leight-weight protocol, that is straight-forward to implement and runs well, even with limited processing power on a microcontroller. For future purposes the broadcasting capabilities of UDP might also proof useful, as many nodes could be connected to this CAN-to-UDP gateway for example for visualization on different computers or additional logging outside of the MABII environment. One limitation that was decided on, is to have receiving capabilities only for the MABII to eliminate the need for extensive computation and CAN-matrix handling on the Arduino.

Excursus: The CAN-protocol is the most widespread protocol to allow for communication between different ECUs in the automotive field. Development began at the Robert Bosch GmbH, but is now standardized and enhanced and adjusted for special purposes internationally. Messages are broadcasted by the bus-participants and stamped with their unique identifier, which also doubles as an arbitration token to handle message prioritization. Each bus-participant can listen to all available messages and "only picks from the bus what he needs". The standard message has a size of eight byte à eight bit, transmitted after the identifier and followed by an ending sequence. Hardware-wise CAN-communication relies on only a pair of twisted wires, where a very robust voltage difference signal is transmitted. [6]

By utilizing the UDP-protocol to acquire data into the simulation, the MABII environment's very convenient possibilty to incorporate CAN database files (.dbc-file)[1], which is the usual way to exchange information and instructions for CAN-networks is no longer available. To decompose the UDP packets into the original signals it was necessary to implement the function of a .dbc-file in the underlying Simulink-model.

All messages that the CAN-gateway is supposed to handle and forward are put into one UDP frame in succession with respective CAN-identifiers and additional spacer bytes between signals. The UDP frame is broadcasted every time a new CAN-message reaches the gateway

[1]Correlates physical human readable/understable signals with units to the actual distribution over the different bytes of a CAN-frame. It allows to "decrypt" the information which is available on a CAN-bus and allows to code and send messages in the respective format that the other bus-participants (ECUs, sensors) expect.

on one of its CAN-buses. Messages that were not updated with the incoming CAN-message will be kept at their previous value. This is called for, as UDP is a connection-less protocol without any loss-prevention mechanisms like acknowledgment-handling or retransmission of messages. Holding the values if not available ensures, that at least some value is available on the bus. UDP also doesn't have native

- Skalierbarkeit
- daten-konvertierung und packaging zu UDP und von UDP zu Daten in CD!!
- Skizze des Systems
- verwendete bibliotheken
- verwendete Controller
- I2C-anbindung und bus-layout

4.4 Interfaces and connections with dolly

Figure 4.5 gives an overview of all the in- and outputs of the dolly. To the first semitrailer the dolly is connected with a standard 7-pin ISO 7638-2 connector. This connector includes 24 V electrical power, a signal for ABS fault detection as well as the ISO 11992 CAN. There's a second connector, which includes different pins for different kinds of lighting of the dolly, like the turn indicators and brake light. The dolly also receives compressed air from the first semitrailer. For controlling the MABII an Ethernet connection is established between the dolly and the tractor unit, where a laptop is located. The interfaces of the dolly with the second semitrailer are the same as those to the first one. To control and measure signals of the dolly a connection to the dollies ETS- and EBS-system had to be established. To achieve this, different kinds of CAN-buses had to be used. That was necessary because the various systems of the dolly use different CAN-protocols. In total five CAN-buses connect the MABII with the dolly. Two for each of the ETS-ECUs and one for the ASF- and EBS-Signal.

One of the CAN-buses contains the signals from the EBS-ECUs and the articulation angle sensor, which measures the angle between the dolly and the first semitrailer. It uses the extended CAN-standard. In the original state this CAN is directly routed to the CAN of the two ETS-ECUs (see section 4.2). In order to achieve steering that is both independently from the articulation angle as well as the velocity of the vehicle the connection between the ASF/EBS-CAN and the CAN of the ETS-ECU was cut. Instead the ASF/EBS-CAN is connected to the MABII.
The MABII is connected to the CAN bus of each ETS-ECU. Because of the use of different CAN standards of the ASF/EBS-CAN, which uses the extended CAN standard, and the ETS-ECU CAN, which uses the J1939 protocol, two physical CAN ports are needed on the MABII for each ETS-ECU. Those two CANs are merged inside the breakout-box. Figure 4.6 shows the breakout box which is used to as an interface between the different CANs that were used and the MABII. The merging of the two CANs that contain signals for the ETS-ECU can be seen on the right side of the scheme.

Furthermore there is the ISO 11992 CAN bus which is used for communication between tractor and the attached semitrailers. In the current design this CAN is only connected to the

EBS-units of the dolly. For future projects it is preferable to connect the MABII to that CAN, too. If the connection is established and a matching DBC-file for the ISO 11992 standard is acquired, there would be different benefits. For one thing more information about the state of the trailers, like for example axle loads and wheel speeds, could be received and used to improve quality of the capability-state of the dolly (see section 6.3), for another thing there would be the possibility to also control breaking and the leveling of the air suspension of the dolly. With this a more sophisticated control of the vehicle dynamics of the dolly can be achieved.
Another possibility to get control over the braking and leveling system could be over the diagnosis port of the EBS. This port is usually used for diagnostic purposes. With the matching diagnostic software the braking and leveling can be controlled manually. If the EBS supplier provides insight into how the diagnostic software communicates with the EBS unit, this port could be used to access the EBS units and the brake and leveling control could be implemented.

- **distinguish between HIL/bench-testing and track-testing!!**

4.5 Interfaces with truck

The vehicle model controllers depend on inputs from the tractor unit(see section 3.3). One of the inputs is the steering angle of the tractors front axle, another the current vehicle speed. For track-testing the MABII is connected to the Chassis CAN of the tractor, in order to get those signals. For the HIL-testing the tractor unit is simulated using VTM (see section 5.2.2).

- CAN communication with truck
- steering system conncetion (VDS!)
- physical CAN path to dolly?

4.6 Measurment Setup

4.6.1 On-board sensors

Sensor	**Measurings**	**Type**	**Mounting point**
Draw bar sensor	Kingpin angle Temperature	External	TBD!!!
EBS	wheel speed	External	wheel brakes
Steering angle sensor	steering angle (first, second axle)	External	steering knuckles

Table 4.1: Available sensors of VSE's ETS

4.6.2 Inertial measurement unit

To determine the processing delays in the control chain (refer to chapter 7.3) as well as logging implementation for verification and analyses purposes a number of inertial measurement units (IMU) where utilizied throughout this thesis' work. The system at hand combined a gyroscope

(L3GD20H), and an accelero- and magnetometer (LSM303D) into an IMU put on one circuit board.[16] This one-chip solution allowed for a convenient access to the sensor measurings, as the sensor outputs could be received via Inter-Integrated Circuit-protocol (I^2C) which eliminates the need for a transducer. Furthermore a high-pass filter is integrated into the IMU's accelerometer, which leads to simpler compensation of the immanent drift of the magnetormeter. These units supply the measurings for three axes each at a maximum frequency of 1600Hz for the accelerometer and 757.6Hz for the gyroscope. [23][22]

The IMU-chip was put in a suitable plastic housing and cast into resin, to fixate the IMU securily and prevent movement relative to the truck-frame on which it will is mounted as well as ensuring water-proofing.

Figure 4.7 shows, how the IMU is connected to the Arduino Due utilizing the I^2C-interface on the Due. Not pictured are the power supplies (both 5V) for the MCP2515 CAN-transceiver and the IMU itself. To minimize cables that have to be rooted through the combination and avoid problems with cable length for the power cables, it was decided to equip each of these assemblies with a rechargeable battery-pack. This also allows to measure in remote location and collect data locally on the Arduino Due during testing, eliminating the need for CAN-communication completely if needed in the future.

A maximum of two IMUs can run on one I^2C line as it is possible to override the last bit of the otherwise hard-coded I^2C address via a hardware jumper. If both I^2C lines of the Due are used, it would be possible to connect up to four IMUs to one Arduino. Though eight IMUs are desired for the whole LCV, this was outruled as a possibility for this project, due to the limitations of the physical bus-length of the I^2C lines of only a few meters (as the name already suggests I^2C is not meant for long distance communication). As can also be seen in the overview figure of the setup on the truck in figure 4.1 this distance would be exceeded with the distance between the different mounting points (15-20m).

In the final measuring setup the assembly shown in figure 4.7 will be present four times in the whole combination, to gather motion data from each independently moving part. Figure 4.1 shows those placements in an overview sketch. The correct placement will be as close as possible to the center of gravity (CoG) of the respective unit to have as little influence of the vehicle's rotary motions as possible, which drastically reduces work in data analysis further on. CoG locations can be gathered from Volvo's internal database and are available at the test site. Putting two IMUs in each spot allows for averaging and ensure redundancy should one IMU chip fail or produce corrupt outputs, which occured from time to time during the bench-testing of the assembly if the voltage of the preliminary power-supply dropped due to other loads.

4.6.3 Arduino Due

To parse the IMU's sensor output (see also section 4.6.2) and convert it into a measurable format and for the implementation of the CAN-to-UDP gateway, it was decided to rely on the cost-efficient and flexible Arduino microcontroller platform. The Arduino platform series was developed by SmartProjects company in Italy and consists of a micro-controller board with several in- and output pins, and a set of programming tools to program said micro-controller (see also section 5.4). Hardware layouts as well as software of this solution are released under an open-source licence, so changes at any level of detail can be made and many other suppliers are now offering solution that tie in with the Arduino. Furthermore this lead to a broad user base and availability of vast supporting resources.

SmartProjects offers various Arduino boards mainly differing in number of communication pins, memory size, clock frequency and physical size. It was decided to utilize the to date most powerful Arduino Due, offering 84 MHz of clock-speed on a ARM Cortex-M3 processor, 512kB flash memory for programm-code and a 96kB working SRAM. It is the first Arduino with 32-bit architecture. This was done in order to allow for future use of this platform in other projects and having enough space for different sub-programs on the controller as well as enough processing power to deal with basic signal filtering and parsing at appropriate speeds.

The Due has a native I^2C interface, which was used to gather the measurings from the IMU. It also has a Serial Peripheral Interface interface (SPI) which is needed to access the ethernet controller necessary for the CAN-UDP-gateway (refer to section 4.3.1), serial communication is also handled on hardware level. The availability of these ports in hardware form allow for robust systems and eliminate the need to implement those protocols on software level (bit-banging), which frees up memory resources. In addition to this digital communication possibilities the Due offers an abundance of 54 digital and 12 analog freely configurable I/O-pins. The Due is also the first Arduino to host an on-board CAN-controller, which in this thesis will be used for measurement transfer to the logging system (MABII), eliminating the need for additional hardware for CAN-bus interfacing on the Arduino side.

Besides the power-supply and the IMU-chip a MCP2551 CAN-transceiver by Microchip Technology Inc. was incorporated to take care of the pyhsical layer of CAN-bus communication by converting the digital signals from the Due's CAN-controller to the standardized voltage levels of the CAN. The MCP2551 is capable of different CAN standards and fully ISO-11898 compatible, which makes future use in different environments or as unit for in-vehicle CAN-interfacing feasible.

The Arduino Due was used to determine the delays in the system (see section 7.3.2) as well as a light-weight solution to quickly read CAN-outputs of the various systems during this thesis' work. This proofed to be an easy debugging solution. Additionally it was used to trigger some error codes to test the developed safety mechanisms (see section 7.2.4).

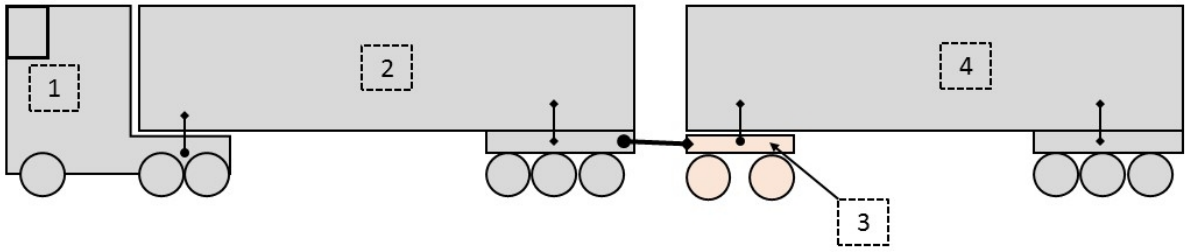


Figure 4.1: Positioning of different sensors on the LVC

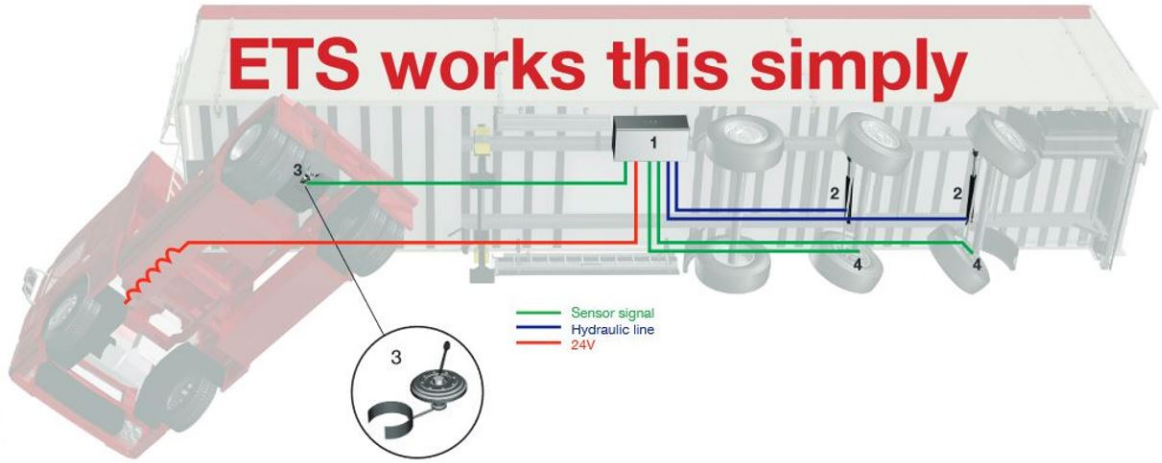


Figure 4.2: Active dolly legacy steering system supplied by VSE[5]

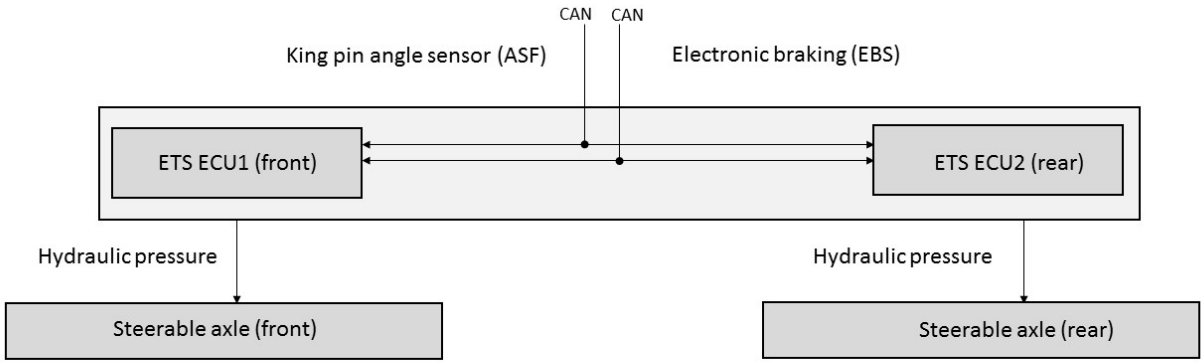


Figure 4.3: System overview for sensors and signal path for ETS legacy system by VSE

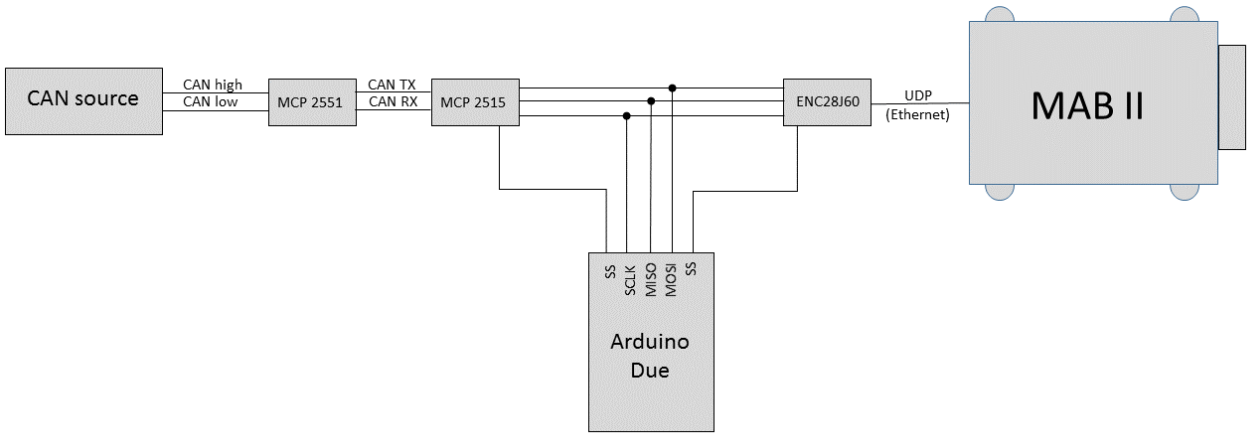


Figure 4.4: Extension of MABII CAN-buses (hardware overview)

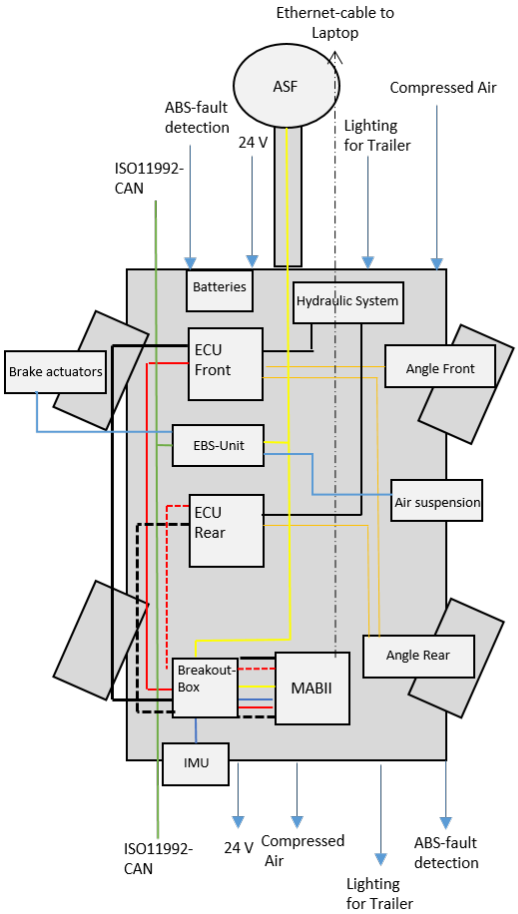


Figure 4.5: Dolly Interfaces

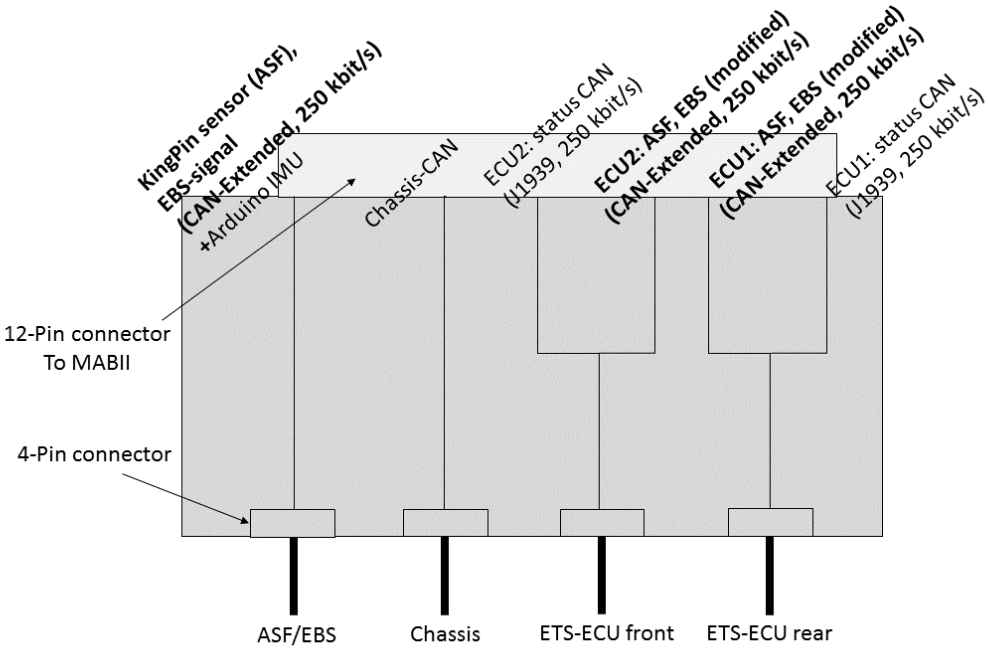


Figure 4.6: Breakout box scheme

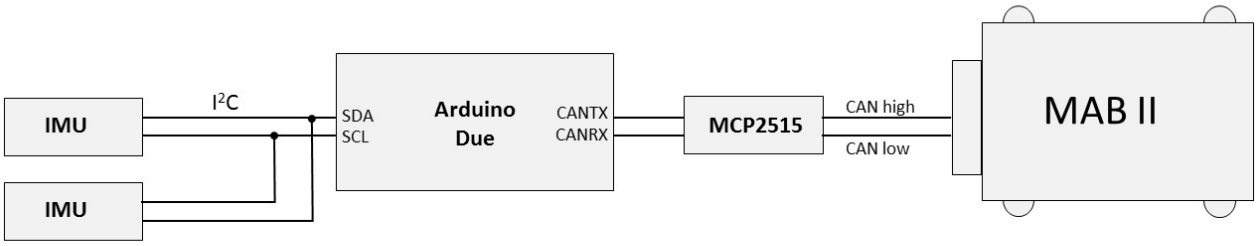


Figure 4.7: Connection of IMU with Arudino Due to MABII

5 Software Architecture (8 Seiten)

5.1 Functionality architecture

- describe VTT funct. artchitecture
- make and insert sketch for (1)
- outline which software system correspondes to what
- differentiate between HiL, Bench and track

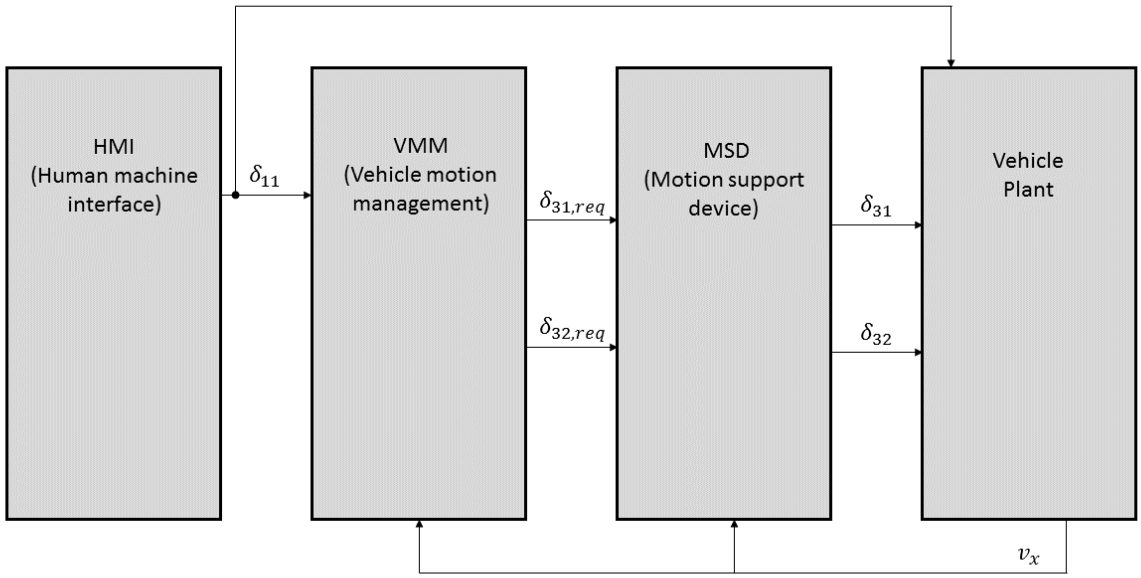


Figure 5.1: Functionality Architecture

5.2 Matlab/Simulink environment

5.2.1 dSpace RTI-blockset

The dSpace RTI-Blockset (Real-time interface) is a plug-in for MATLAB/Simulink that allows you to connect a simulink-model to the different inputs and outputs of the MABII. There are RTI-blocks for CAN, Ethernet, LIN, FPGA and the analog and digital outputs of the MABII. In this project the RTI CAN MultiMessage and the RTI Ethernet (UDP) Blocksets were used.

RTI CAN MultiMessage Blockset

The RTI CAN MultiMessage blocks establish an interface between the physical CAN-Buses of the MABII and the simulink-model running on the MABII. There are four different blocks in this blockset:

In the "GeneralSetup" block the paths of the model root and the destination folder for generated files are set.

In the "ControllerSetup" block first the name of the controller has to be set, then the physical CAN-Bus, that should be used, is set by choosing a module number and a controller number. How the module- and controller numbers have to be set for the different CAN-buses of the MABII is shown in 5.1.

Table 5.1: CAN-layout MABII

CAN	Module number	Controller number	ZIF-Pin CAN-High	ZIF-Pin CAN-Low
CAN1	1	1	c2	c3
CAN2	1	2	b2	b3
CAN3	2	1	B2	B3
CAN4	2	2	A2	A3
CAN5	3	1	P2	P3
CAN6	3	2	N2	N3

After setting the module- and controller number the identifier format has to be set to either standard or extended format, the transceiver type must be chosen between ISO11898-2 and ISO11898-6. ISO11898-2 is used for a high-speed medium access unit and ISO11989-6 for the selective wake-up functionality of a high-speed medium access unit. If needed, a termination resistance of 120 Ohms can be set in the block as well. As a last step the Baud rate of the CAN-bus has to be defined.

In the "MainSetup" block a dbc-file is connected to one of the controller blocks, that were created before. A "ControllerSetup" block can only be connected to one "MainSetup" block at a time. When the dbc-file is loaded in the "MainSetup" block, the different messages and signals of the dbc-file can be chosen as inputs and/or outputs to the simulink-model.

RTI Ethernet (UDP) Blockset

The RTI Ethernet (UDP) Blockset was used for two purposes: On the one hand for the HIL-testing (see section 7.5), on the other hand for the CAN-bus extension of the MABII (see section 5.2.3). The RTI Ethernet (UDP) Blockset contains four different blocks: In the "Ethernet_UDP_SETUP" block Blablbalba
The "Ethernet_UDP_RX" block is used to receive data over Ethernet. Blablbalba
The "Ethernet_UDP_TX" block is used to send data over Ethernet. Blablbalba
The "Ethernet_UDP_INTERRUPT" block is used to send data over Ethernet. Blablbalba

5.2.2 Virtual Truck Modelling Library

The Virtual Truck Modelling Library (VTM) is a computational framework used within Volvo Trucks to simulate the dynamic behaviours of trucks and combinations. As a library it extends Simulink, where maneuvers, track layout and the trucks kinematic and dynamic properties are linked together and then computed. This toolbox was used as a base for the simulation of the LVC including the dolly for this project. It is possible to run simulation offline with a predefined maneuver and a given environment as well online where these parameters are fed into the calculation live or as measurings from the real-world environment. For online use all relevant parameters and states can be accessed in Simulink or in case of execution on the MABII through ControlDesk as their respective representation of the Simulink variable.

5.2.3 CAN-bus extension

- outline data diagram structure
- show limitations uf UDP (time, reliability, retrans)
- timestamping

5.3 ControlDesk monitoring environment

5.3.1 Maneuver control

5.3.2 Monitoring and logging

- data-format
- frequency
- synchronizing over different CANs

5.4 Arduino IDE and applications

The Arduino system provides an integrated development environment (IDE) written in Java, providing cross-platform support. It is used to develop the code as well as compiling the code and subsequently uploading it into the microcontroller via the computers serial interface. Within the IDE it is also possible to load some of the officially supported libraries directly. It conveniently is possible to monitor the computer's serial interface as well, which is the most practical way to monitor and debug code that is executed on the Arduino.

Library name	**Purpose**	**Comment**
LSM303	read magnetometer on IMU via I2C	[18]
L3G	read gyro & accelerometer on IMU via I2C	[17]
UIPethernet	control ENC28J60 via SPI	[14]
TinyGPSPlus	acquire and parse GPS signal from EM-506 via serial	v0.94b[13]
mcp_can	implement CAN via MCP2515 and MCP2551 via SPI	[8]

Table 5.2: List of utilized libraries on the Arduino platform

6 Fault detection and system ability (5 Seiten)

6.1 Failure Mode and Effects Analysis (FMEA)

FMEAs were first used by NASA in the apollo project in the 1960s. A FMEA is used to detect possible failures before they appear. Therefore the FMEA is done in a early stage of a project in order to be able to take the results of the FMEA into consideration when developing a system. In the process every possible failure of a system are taken into consideration.

In a first step a block diagram of the system with all of its inputs, outputs and subsytems was created to gain a complete understanding of the system. Every subsystem was broken down to the lowest level and for those subsystems block diagrams with all the components, inputs and outputs were created as well.

Starting from the block diagrams all potential failure modes were determined for every component of the system respectively subsystem. As a next step the potential effects of these failure modes were determined and the severity of these effects were evaluated on a scale from one to ten, with 1 being the lowest severity and ten being the highest severity. For the evaluation a table that shows how different severities correlate with the numbers was used. picture of severity-table Following this, potential causes for every failure mode were defined. Then the probability of occurrence for each cause was evaluated, also using a scale from one to ten. After that the current control mechanisms, that detect the failure when it should appear, were listed for each failure mechanism and the detectability of the failure mechanism was evaluated using a scale from one to ten. After the severity, probability and detectability were evaluated, a risk priority number is calculated as follows:

$$RPN = severity * probability * detectability$$

This risk priority number is used to identify the failure mechanisms that need to be addressed. In table 6.1 it is shown how the risk priority number For the failure mechanisms that need to

Table 6.1: Risk priority number

RPN	Action
0<RPN<40	No action needed
40<RPN<100	Decide if action is needed after review
RPN>100	Action needed

be addressed actions were decided, that either lower the severity, probability or detectability of them or more than one of those factors. After the performance of these actions the severity, probability and detectability were evaluated again and a new RPN was calculated. If the RPN still is to high a iteration of the process was done.

6.2 Safety concepts

6.3 Maximum capabilities of the system

- give indicator of maximum angle/angle rate

- describe "algorithm"/lookuptable
- explain underlying physical correlation (ref to MA from)

6.4 Warning and state-info system

- warnings from dolly ECU
  In the original state, the ETS-ECU monitors all safety critical components of the dolly. In case of a malfunction, an error code is sent via the CAN-Bus and the error message is visible on the diagnose displays of the dolly. Due to the required modification of the ASF- and speed-signal, the inbuilt supervision of the ETS-ECU is no longer fully functioning. Therefore a additional supervision of the system has to be implemented in the Simulink-model.
- warnings from EBS
- warnings from vehicle
- 'own' error codes and warnings (e.g. logging, MABII related, arduino-IMU related)

7 Verification and validation of steering (2-3 Seiten)

7.1 Overview

In different states of the project different kind of tests were performed. As a first step bench tests were done to verify the developed software. Following that, several tests on the actual dolly, with the dolly in standstill and no trailers connected, were made.

7.2 Bench-Testing

7.2.1 ECU-setup

7.2.2 VTM maneuver verification

- include simulation results for maneuver of controller being run inside VTM
- perhaps include some picture of the video-output, with the actuated truck
- plots for maneuver (steering input, speed over time)

7.2.3 CAN verification

- check output from ASF with arduino
- check ECU CANs, make sure they differ in output

7.2.4 Fault detection system verification

- send faulty inputs
- show plot of faulty input and system reaction

7.3 Processing time evaluation

7.3.1 Background

The desired solution is supposed to operate at any speed. For high speeds a quick processing and transmission time is required to ensure prompt and realtime intervention of the control system based on the measured input signals. If the delays induced by the different components in the complete system are known or can be estimated, they can be compensated for in the steering-algorithm running on the rapid-prototyping system.

As outlined in section 4.6 the dolly is equipped with a system to determine the deflection angle of the draw-bar and both the kingpin angles of the LVC. The sensors' raw signal is then parsed and filtered in an integrated low-level system which feeds the filtered signals to a private CAN-bus, where it is picked up by the MABII. The filtering operation takes a certain time and

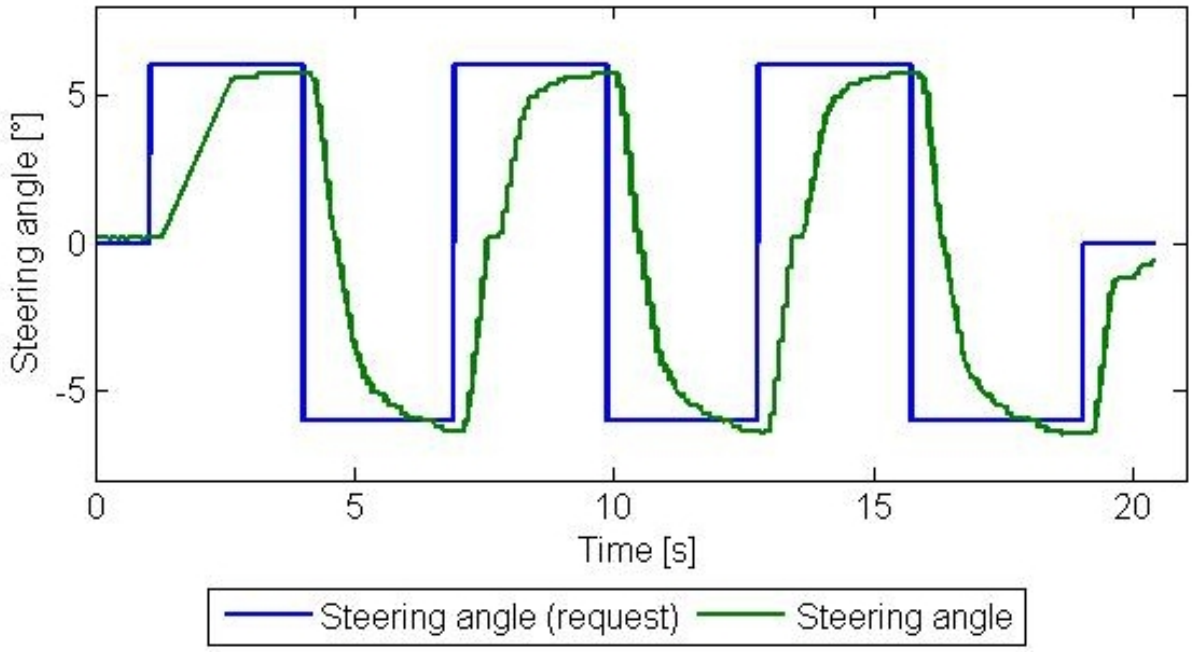


Figure 7.1: Example of step input for delay determination, front axle lifted, 8° amplitude, six repetitions.

thus induces a delay. Furthermore the model running on the rapid-prototyping system needs a certain time to calculate the current desired steering angle for the dollys' wheels. This has to be determined as well. The steering mechanisms on the dolly are also a delay-inducer due to the inertia in the hydro-mechanic system. This is as well unavoidable, but when measured can as well to a certain degree be compensated for.

The measuring chain for the logging solution will also be discussed and the different delays will be discussed.

7.3.2 Measured input delay

The response time of the dolly's steering system was determined in workshop environment in stand-still without any additional axle loads. Tests were conducted for both axles individually to ensure that maximum power was available preventing further delays through insufficient electrical power or hydraulic pressure. A repeated steering angle in the form of a step input was used to determine the reaction time of the system. The step amplitude was increased over different tests to account for possible dependencies on articulation of the steering system. The tests each started and ended with a steering angle of 0°. An example for the test sequence can be seen in figure 7.1. Clearly the difference between the requested angle and the actual articulation of the wheel can be seen. Further discussion of the results can be found in section 8.2.

To eliminate tire friction the investigated axle was suspended in the air in further series of testing by slightly craning up the dolly's front/rear (see figure A.1) to allow for free movement of the tires. An example for the utilized testing routine can be gathered from table 7.3.2. Between the different steps (double amplitude) a sufficient wait time of 3500ms was used to

allow the system to reach the requested steering angle. This waiting time was kept constant throughout all tests.

#	Amplitude [°]	Axle	Comment
1	1	front	lift front axle, ETS dead band
2	2	front	
3	3	front	
4	4	front	
5	5	front	
6	6	front	
7	7	front	
8	8	front	
9	9	front	error mode, step too[1]
10	1	rear	lift rear axle, ETS dead band
11	2	rear	
12	3	rear	
13	4	rear	
14	5	rear	
15	6	rear	
16	7	rear	
17	8	rear	
18	9	rear	error mode, step too big[1]

Table 7.1: Test matrix for delay measuring with lifted axle (analogous matrix for non-lifted testing (#19-#36))

As also described in section 8.2 additional response tests were conducted utilizing a ramp input. Here a maximum steering amplitude of 15° was used and the slope of the ramp was varied. As the request was not in the form of a step input, no error was triggered like in test #9 and #18 (see table 7.3.2). The test matrix for this setup is outlined in table 7.3.2.

#	Amplitude [°]	Rate	Sample time [s]	Comment
1	15	0.01	0.001	
2	15	0.001	0.001	
3	15	0.1	0.001	
4	15	0.3	0.001	
5	15	0.4	0.001	error mode, slope too steep

Table 7.2: Test matrix for delay measurings with ramp input

All testing was automated in ControlDesk using Python scripts to loop through the different tests. The utilized graphical user interface (GUI) can be seen in figure A.2. Logging within ControlDesk was started and stopped manually after each test and automatically exported to MATLAB for parsing and analysis. All data was acquired from the steering knuckle sensors originally present in the ETS. This sensor's signal is available on the ETS-CAN (also see

[1]If a too big of a step is requested from the steering system, VSE's ECU will go into emergency mode. Steering is completely locked, turning the dolly completely passive.

section 4.2).

It was decided to neglect the delays induced by the CAN communication. The reason for this is, that both the request as well as the measuring are send over the CAN. Which means that the two transmission times are counted with different signs and thus almost exactly cancel out only leaving double the worst case transmission (maximum stuffing bits, worst case arbitration) time as a maximum error of 232μs. This is sufficiently low regarding the fact, that the simulation on the MABII is executed every 0.001s.

7.3.3 Delays in logging measuring chain

To have a complete picture of the occuring delays in the measuring chain all relevant bigger execution blocks of the code running cyclically on the Arduino Due to broadcast the IMU data to the CAN-bus were evaluated for execution time. Figure 7.2 gives an overview, into which functions the code was grouped as well as the respectively used protocol for communication. The output via the serial port (Universal Asynchronous Receiver Transmitter (UART)) for visualization purposes on the computer while logging takes very long compared to the execution times of the other depicted code blocks.[1]It therefore is disabled while actually logging.

The delays were evaluated by inserting time-stamped variables into the code before and after each examined function's call. Using the Due's *micros*() function a resolution in microseonds can be achieved[2] with an accuracy of the inverse of the processors clock frequency (i.e. 11.9ns for the Due). *micros*()' call/execution time was neglected as it is in the magnitude of a couple of dozen nanoseconds. As

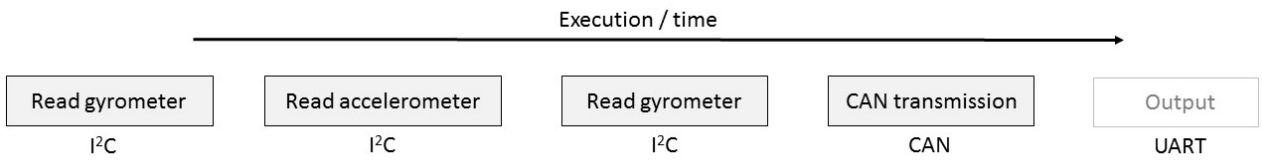


Figure 7.2: Execution blocks for IMU data acquisition on the Arduino Due (overview)

[1]The maximum transmission rate on the UART for the Due is 115200bit/s. Transmitting the payload (3x3 sensor axes, signed integer) only excluding all control bits accumulates to 3 sensors x 3 axes x 16bit = 144bit or 1.25ms

[2]On earlier/smaller Arduinos *micros*() resolution is 4μs!

7.4 Vehicle testing

7.4.1 System calibration

7.4.2 Actuator tests

7.4.3 Algorithm evaluation

7.4.4 Sensor testing

- test angle sensor offline
- zero IMUs, make sure paired sensors show the same outputs

7.5 Hardware-in-the-loop testing

- description of HIL
- Functional architecture diagram (MSD replaced with actual dolly)
- Limitations (axle loads, standing still)

Insufficient processing power does not allow for VTM's execution in the dSpace environment on the MABII. To perform HIL-testing it was thus necessary to split the comptuational load and accomplish real-time data exchange between the hardware controlling system on the MABII and the rest of the simulation which will be run parallely in Simulink in real-time on a standard PC. Though there are dedicated real-time platforms available to achieve real-time execution it was decided to rely on a standard PC to minimize costs and have a lean work-process without extra steps of code conversion to different platforms. Figure 7.3 illustrates the distribution of the HIL-setup's different components over two computers, the MABII and the actual hardware.

The Host-PC fulfills the usual task to start-up the dolly and simulation during HIL testing and log the behaviour of the dolly system and the IMUs. This is done through the ControlDesk GUI which was described in section 5.3 and of which a screenshot can be found in the appendix (figure A.2). It already runs the final version of the control and capturing environment and does not require changes for track testing. It is connected to the MABII via the Host-PC port.

On the Simulation-PC the simulation of the complete rest of the hardware-system is computed in real-time. Mainly truck, trailers, environment and steering behaviour of the truck are part of this model. All these sub-systems are modeled in VTM which runs on this computer in Simulink.

Usually Simulink simulations are computed as fast as possible and then afterwards analysis of the results and system's behaviour is conducted offline. This is an unsuitable approach for this thesis' purposes, as it is important to have real-time execution for this sub-system to provide the actual hardware of the HIL-test with the needed parameters and simulate the environment at the correct instance also considering the hardware's feedback. I.e. the hardware runs in real-time, so the rest of its environment has to as well. Simulink's Desktop Real-Time toolbox provides the functionality to execute models in real-time on a desktop PC as well aas support for a variety of hardware systems. This toolbox was used to connect the MABII's ethernet port to the simulation-PC's ethernet port using the UDP protocol.

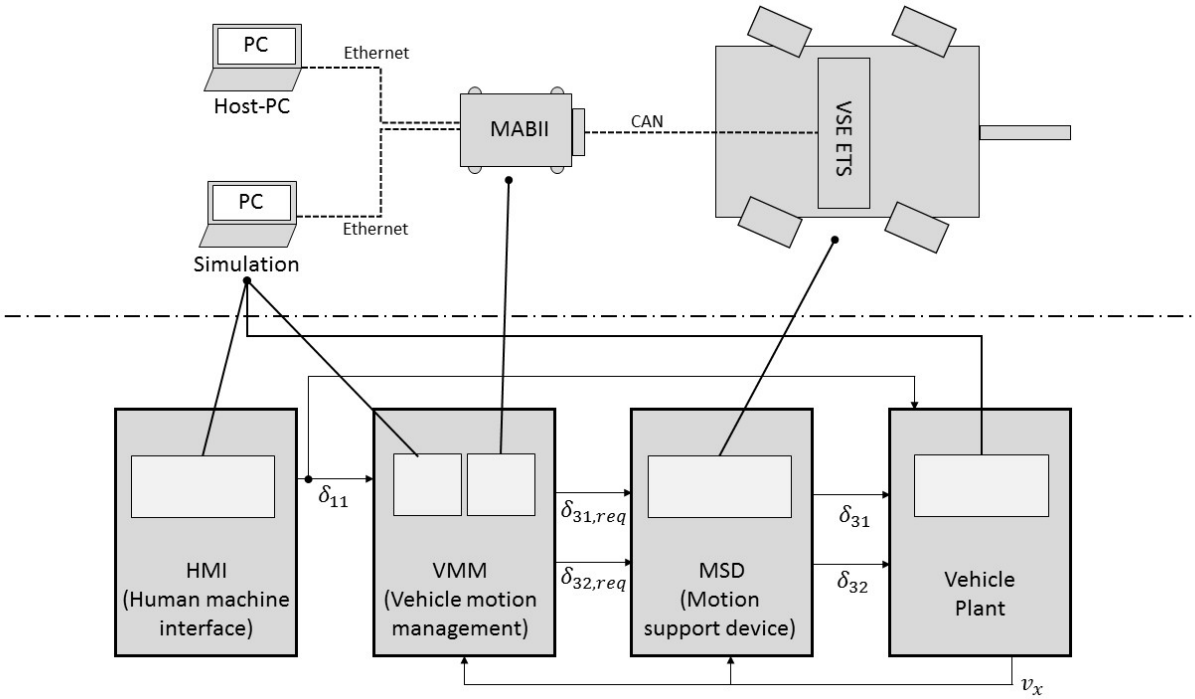


Figure 7.3: Overview of HIL-simulation, distribution of sub-functions over different physical platforms (top) and correlation to Volvo functional architecture (bottom)

Limitations: HIL-testing took place in workshop environment and thus the dolly was not rolling compromising the tire friction compared with a rolling combination. As outlined in section 8.2 this does not increase the reaction time of the system, though. Furthermore the additional axle load was zero for both axles, as it was not possible to increase the load without a trailer in the lab environment at hand. This is also unlikely to change the results, as is suggested in 8.2 the axle-load is no deciding factor in the wheel articulation's performance.

7.5.1 Low-speed controller

7.5.2 High-speed controller

7.6 Track testing

7.6.1 Testmaneuvers

- lit research for standard maneuvers
- sine-wave
- outline critical parts of maneuver
- figure with SA over time
- expected behaviour from simulation

7.6.2 Testenvironment AstaZero

- overview of AZ
- map in appendix?
- restrictions of environment

7.6.3 Testmatrix

- checklist for launch
- parameters that very varied
- different runs
- planned maneuvers

7.6.4 Test setup and instrumentation

- detailed description of placement of sensors, wiring, logging-PC

8 Discussion (3-4 Seiten)

8.1 Results from bench testing

- lessons-learned?
- adaptation for future projects
- what was taken over for further tests?
- what couldnt be simultaed?

8.2 Results from processing time evaluation

- diagrams (over amplitude, show time)
- explain different curves
- explain data gathering
- mention ramp input
- show reaction time

During plot analysis of preliminary results it was discussed, whether it would be possible to by-pass the occuring delay by feeding a ramp-input instead of a step-input into the ETS. It was assumed that there is a low-pass filter in place which would not affect ramp inputs. Subsequent tests with ramp inputs with varying slopes and amplitudes showed that it was indeed possible to eliminate the low-pass filter leading to no delay between the target angle and the request angle. Nevertheless it was not possible to get any faster change than with the request of a step input. It thus can be concluded that the maximum steering rate of $5°/s$ can not be circumvented.

8.3 Results from in vehicle testing

- CAN-analysis
- robustness?
- reliability of safety features

8.4 Results from hardware-in-the-loop testing

- compare plots from HIL with simulation

8.5 Comparison

- VTM <=> testing

9 Conclusion (2 Seiten)

9.1 Recommendation

9.2 Future Work

- braking (indivi)
- leveling
- wireless configuration?
- track testing
- propulsion
-

A Appendix

Figure A.1: Dolly craned up to determine delay of steering actuation

Figure A.2: ControlDesk GUI for basic testing and start-up of the dolly

References

[1] U. E. P. Agency. *Longer combination vehicles. a glance at clean freight strategies.* 2010. URL: http://www.epa.gov/smartway/forpartners/documents/trucks/techsheets-truck/420f10053.pdf (visited on 02/09/2015).

[2] H. Backman and R. Nordström. Improved performance of European long haulage transport. *Transport Research Institute (TfK), Stockholm* (2002).

[3] A. Bálint et al. "Correlation between truck combination length and injury risk". *Australasian College of Road Safety Conference, 2013, Adelaide, South Australia, Australia.* 2013.

[4] T. R. Board. *REVIEW OF MEXICAN EXPERIENCE WITH THE REGULATION OF LARGE COMMERCIAL MOTOR VEHICLES.* Oct. 2011. URL: http://onlinepubs.trb.org/onlinepubs/nchrp/nchrp_rrd_362.pdf (visited on 02/09/2015).

[5] V. V. S. E. B.V. *Product information ETS for trailers.* 2014.

[6] S. Corrigan. Introduction to the controller area network (CAN). *Texas Instrument, Application Report* (2008).

[7] C. Doll et al. Long-Term Climate Impacts of the Introduction of Mega-Trucks, Study to the Community of European Railways and Infrastructure Companies (CER). *Fraunhofer ISI (Study Co-Ordinator, Karlsruhe) TRT (Milan), NESTEAR (Gentilly), Fraunhofer-ATL (Nuremberg), Fraunhofer-IML (Dortmund), Karlsruhe* (2009).

[8] GitHub. *Seeed-Studio/CAN$_B$US$_S$hield.* 2015. URL: https://github.com/Seeed-Studio/CAN_BUS_Shield/.

[9] A. Infrastructure and Transportation. *Highway Provider View of Long Combination Vehicles.* Mar. 2005. URL: http://www.transportation.alberta.ca/Content/docType59/Production/FAQs%28US%29.pdf (visited on 02/09/2015).

[10] M. M. Islam, L. Laine, and B. Jacobson. "Improve Safety by Optimal Steering Control of a Converter Dolly using Particle Swarm Optimization for Low-Speed Maneuvers".

[11] M. M. Islam, L. Laine, and B. Jacobson. "Inverse Model Control Including Actuator Dynamics for Active Dolly Steering in High Capacity Transport Vehicle".

[12] M. S. Kati et al. "Performance Improvement for A-double Combination by introducing a Smart Dolly". *13th International Heavy Vehicle Transport Technology Symposium, San Luis, Argentina.* 2014.

[13] *mikalhart/TinyGPSPlus.* 2013. URL: https://github.com/mikalhart/TinyGPSPlus/.

[14] *ntruchsess/arduino$_u$ip.* 2014. URL: https://github.com/ntruchsess/arduino_uip.

[15] T. E. PARLIAMENT and T. C. O. T. E. UNION. *DIRECTIVE 2006/126/EC.* Dec. 20, 2006. URL: http://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32006L0126&from=DE (visited on 05/07/2015).

[16] PololuCorporation. *AltIMU-10 v4 Gyro, Accelerometer, Compass, and Altimeter (L3GD20H, LSM303D, and LPS25H Carrier).* 2015 (accessed February 12, 2015). URL: https://www.pololu.com/product/2470/.

[17] *pololu/l3g-arduino.* 2015. URL: https://github.com/pololu/l3g-arduino.

[18] *pololu/lsm303-arduino.* 2015. URL: https://github.com/pololu/lsm303-arduino.

[19] "Rapid Control Prototyping". German. *Rapid Control Prototyping.* Springer Berlin Heidelberg, 2006, pp. 295–318. ISBN: 978-3-540-29524-2. DOI: 10.1007/3-540-29525-9_7. URL: http://dx.doi.org/10.1007/3-540-29525-9_7.

[20] J. Schäuffele and T. Zurawka. *Automotive software engineering: Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen*. Springer-Verlag, 2012.

[21] P. Steenhof, C. Woudsma, and E. Sparling. Greenhouse gas emissions and the surface transport of freight in Canada. *Transportation Research Part D: Transport and Environment* **11**.5 (2006), 369 –376. ISSN: 1361-9209. DOI: http://dx.doi.org/10.1016/j.trd.2006.07.003. URL: http://www.sciencedirect.com/science/article/pii/S1361920906000411.

[22] STMicroelectronics. *L3GD20H - MEMS motion sensor: three-axis digital output gyroscope: Datasheet - production data*. 2013.

[23] STMicroelectronics. *LSM303D - Ultra compact high performance e-Compass 3D accelerometer and 3D magnetometer module: Datasheet — preliminary data*. 2012.

[24] K. Tindell, H. Hansson, and A. J. Wellings. "Analysing real-time communications: controller area network (CAN)". *Real-Time Systems Symposium, 1994., Proceedings*. IEEE. 1994, pp. 259–263.

[25] D. of Transport and M. Roads. *Guideline for Multi-combination Vehicles in Queensland*. July 2013. URL: http://www.tmr.qld.gov.au/~/media/busind/Heavyvehicles/guidepermits/Guideline_mcv_form1.pdf (visited on 02/09/2015).

[26] T. C. O. T. E. UNION. *Council Directive 96/53/EC*. July 25, 1996. URL: http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:31996L0053&from=en (visited on 02/05/2015).

[27] Vägverket. *Weight and Dimensions for Road Traffic*. URL: http://www.unece.org/fileadmin/DAM/trans/wp24/wp24-presentations/documents/pres08-04.pdf (visited on 02/05/2015).