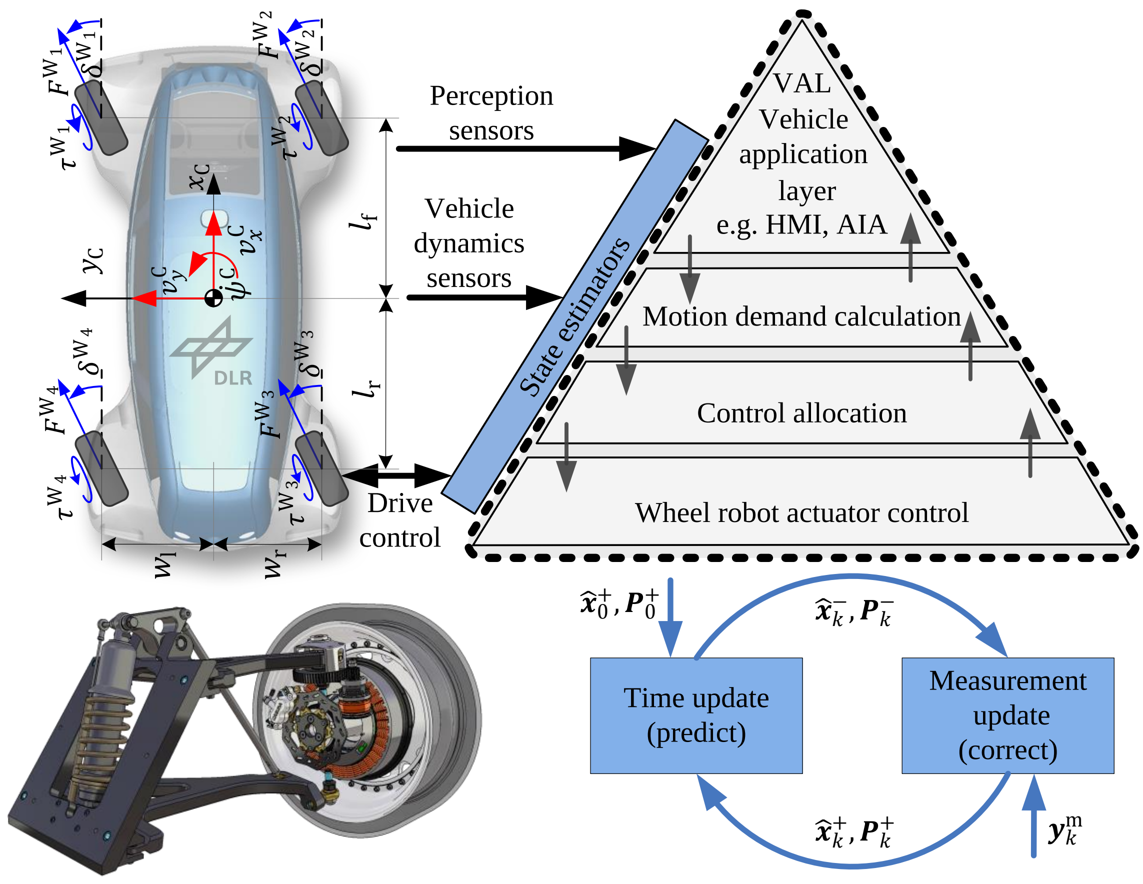
Technische Universität München

Fakultät für Elektrotechnik und Informationstechnik

Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik

Model Based Energy Management and State Estimation for the Robotic Electric Vehicle ROboMObil



Doctoral thesis

Dipl.-Ing. (Univ.) Jonathan Brembeck

Technische Universität München

Fakultät für Elektrotechnik und Informationstechnik

Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik

MODEL BASED ENERGY MANAGEMENT AND STATE ESTIMATION FOR THE ROBOTIC ELECTRIC VEHICLE ROBOMOBIL

Dipl.-Ing. (Univ.) Jonathan Brembeck

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:	Prof. Dr.-Ing. Hans-Georg Herzog
Prüfer der Dissertation:	1. Prof. Dr.-Ing. Ralph Kennel
	2. Hon.-Prof. Dr.-Ing. Gerhard Hirzinger
	3. Hon.-Prof. Dr.-Ing. Martin Otter

Die Dissertation wurde am 19.12.2017 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 05.07.2018 angenommen.

Acknowledgement

Foremost, I would like to express my gratitude to my advisors (in alphabetic order):
Prof. Dr.-Ing. Gerhard Hirzinger, Prof. Dr.-Ing. Ralph Kennel, and Prof. Dr.-Ing. Martin Otter.

My sincere thanks go to Dr.-Ing. Johann Bals, director of the DLR institute of system dynamics and control, for offering me the working opportunities in his institute and for supporting the project.

I would also like to thank Dr. rer. nat. Andreas Pfeiffer for his support in software implementation of the DLR Kalman Filter Library.

Moreover, my special thanks go to the BayStMWIVT for their financial support for the development of ROboMObil's wheel robots within the public funded "Radroboter – eCorner" EMO-1002-0002[1] as well as for the financial support by the BMBF for the research on the Modelica based observer framework in the ITEA2 project MODRIO[2].

Last but not the least, I would like to thank my family especially my parents for their continuous moral and financial support throughout my studies.

[1]https://www.bayern.landtag.de/www/ElanTextAblage_WP16/Drucksachen/Schriftliche%20Anfragen/16_0015878.pdf – Retrieved 03, 2017

[2] https://itea3.org/project/modrio.html – Retrieved 03, 2017

Abstract

The totally from scratch developed robotic electric vehicular research platform of the German Aerospace Center (DLR), called ROboMObil, enables completely new approaches for future energy management systems. Its x-by-wire technology and robotic inspired centralized control architecture offers the freedom for intelligent algorithms to design future economical individual transport solutions. By means of these technologies the disbandment between the driver motion demands and the actual commanded values at the wheels of ROboMObil, the so called wheel robots, is accomplished. In the first part of this doctoral thesis an innovative energy management framework – focusing on the spatial vehicle motion – is developed and simulatively evaluated by means of a multiphysical Modelica model of ROboMObil with several realistic test scenarios. It is itemized in three levels: The first one describes the path optimization within the road-boundaries and the generation of a feasible velocity profile. The second level is an iterative search based path following component and the third an actuating energy minimizing control allocation approach. In the second part of this thesis a novel model based Kalman filter framework is outlined, which enables to automatically incorporate multiphysical Modelica models in discrete-time estimation algorithms extended with constraint handling and real-time capable nonlinear moving horizon estimation. To conclude the connection to the energy manager framework the necessary quantities for the controller are estimated in two application examples of the framework which are successfully tested with data from real ROboMObil experiments. First, a hybrid modeled state of charge battery observer with constraints incorporation for the calculation of the state of charge and current power availability is developed. Second, a constrained vehicle position, orientation and velocity observer with time delayed position measurements from a global navigation satellite system using an extended real-time capable moving horizon estimation approach is investigated.

List of Figures

List of Tables

Lists of Symbols, Nomenclatures and Abbreviations

Formula symbol	Unit	Description
$P_{\mathrm{inv,const}}$	[W]	Inverter losses constant
k_{hyst}	$[\mathrm{Ws/rad}]$	Hysteresis losses constant
τ_{fric}	$[\mathrm{Ws/rad}]$	Motor friction constant
k_{eddy}	$[\mathrm{Ws^2/rad^2}]$	Eddy current losses constant
k_{inv}	$[\mathrm{W/I}]$	Current dependent inverter losses
R_{s}	[Ω]	Warm resistance per phase
z_{p}	$[-]$	Pole pair number
ψ_{PM}	[Wb]	Magnetic flux of permanent magnets
l	$[-]$	State of charge
h	[V]	Cell hysteresis voltage
k_{i}	$[-]$	MESC model cell capacity correction factor
C_{N}	[Ah]	Cell nominal capacity
η_{Ah}	$[-]$	Coulombic cell efficiency
h	[V]	Transient cell hysteresis voltage
M	[V]	Cell polarization voltage depend from l, T
β^{C}	[rad]	Vehicle’s side slip angle
v^{C}	$[\mathrm{m/s}]$	Vehicle’s velocity over ground
$\dot{\psi}^{\mathrm{C}}$	$[\mathrm{rad/s}]$	Vehicle’s yaw rate
ψ_{C}	[rad]	Vehicle’s yaw angle
x_{C}	[m]	Vehicle’s position in x direction
y_{C}	[m]	Vehicle’s position in y direction

Nomenclature	Explanation
$[K_{\mathrm{D}}] = {}^{1}\!/_{\mathrm{s}}$	Denotes that the variable’s K_{D} unit is ${}^{1}\!/_{\mathrm{s}}$
e	Lower case letter variable is a scalar
$\boldsymbol{e}$	Bold lower case letter variable is a vector
$\boldsymbol{E}$	Bold upper case letter variable is a matrix
$\boldsymbol{e} = \{e_1, e_2\} = (e_1, e_2)^T$	Equivalent vector notations
$\boldsymbol{E} = \begin{bmatrix} e_{11} & e_{12} \\ e_{21} & e_{22} \end{bmatrix}$	Matrix notation
$\mathrm{E}(\cdot)$	Expectation value of a random variable
$\boldsymbol{CP}(\cdot)$	Lower triangular Cholesky factorization of $\boldsymbol{P}$
$(\cdot)^{\mathrm{W}_x}$	Quantity expressed in the x-th wheel robot coordinate system parallel to the car coordinate system

Nomenclature	**Explanation**
$(\cdot)^{\mathrm{C}}$	Quantity expressed in the car coordinate system with origin in CoG
$(\cdot)_{\mathrm{C}}^{\mathrm{P}}$	Car quantity expressed in the path coordinate system with origin in CoG
$(\cdot)_{\mathrm{C}}$	Car quantity expressed in the inertial coordinate system – short for $(\cdot)_{\mathrm{C}}^{\mathrm{I}}$
$(\cdot)^{\mathrm{ST},\boldsymbol{W}}$	Quantity belongs to steering powertrain
$\min_{\boldsymbol{x}} \lVert \boldsymbol{e}(\boldsymbol{x}) \rVert_{\boldsymbol{W}}$	With $\boldsymbol{W}$ weighted least squares minimization of function $\boldsymbol{e}$ with respect to $\boldsymbol{x}$
$\mathrm{s.t.}$	“subject to” if the problem is restricted
$\mathrm{O}(\cdot)$	Denotes the number of necessary floating point operations
ϵ	Denotes the machine precision according to IEEE 754-2008

Abbreviation	**Explanation**
A/D	Analog to digital signal conversion
ABX	AutoBox – rapid prototyping real-time controller from dSPACE GmbH
AIA	Artificial intelligence agent – part of ROMO’s autonomous functionalities
BC	Boundary constraint
BMS	Battery management system
BNCU	On-board network control unit of ROMO’s electrical system
CA	Control allocation
CAD	Computer aided design
CAN	Controller area network – vehicle bus standard
CoG	Center of gravity
DC	Direct current
DC/DC	Conversion between different voltage levels
DLR	German Aerospace Center
DOF	Degree of freedom
Dymola	Modelica simulator from Dassault Systèmes
EKF	Extended Kalman filter
EM	Energy management
ESC	Enhanced self-correcting (model)
ESTM	Extended single track model
EV	Electric vehicle
FIFO	First in first out (buffer)
FMI	Functional mockup interface
FMU	Functional mockup unit
GNSS	Global navigation satellite system
GPS	Global positioning system
GRV	Gaussian random variable
HIL	Hardware-in-the-loop
HMI	Human machine interface
ICE	Internal combustion engine – e.g. a gasoline motor

Abbreviation	**Explanation**
IIR	Infinite impulse response (filter)
MABX2	MicroAutoBox II – embedded rapid prototyping real-time controller from dSPACE GmbH
MESC	Modified enhanced self-correcting (model)
MHE	Moving horizon estimation / estimator
Modelica	Object oriented modeling language for multiphysical systems
NG	Nonlinear gradient (descent search)
NMPC	Nonlinear model predictive control
OCV	Open circuit voltage
PDF	Probability density function
PFC	Path following control
PHEV	Parallel hybrid electric vehicle
PMSM	Permanent magnet synchronous machine – electric drive type
QL	Quadratic program solver
QP	Quadratic program
ROboMObil	see ROMO
ROMO	short for ROboMObil – DLR's robotic electric vehicle
RTI	Real-time iteration
SHEV	Serial hybrid electric vehicle
SOC	State of charge
SP	Sigma points
SPKF	Sigma point Kalman filter
SPT	Sigma point transformation
SQP	Sequential quadratic program
SR	Square-root
SR-EKF	Square-root extended Kalman filter
SR-UKF	Square-root unscented Kalman filter
TIPI	Time independent path interpolation
TM	Traction motor
UKF	Unscented Kalman filter
UML	Unified modeling language
URNDDR	Unscented recursive nonlinear dynamic data reconciliation
UT	Unscented transformation
VAL	Vehicle application layer of ROMO's central control architecture
Wheel robot	Wheel unit that integrates propulsion, steering and brake
WLS	Weighted least squares (estimation)
WSLR	Weighted statistical linear regression
X-by-wire	Digital commanded actuator without physical interconnection to the driver
XML	Extensible markup language

Content

1. Introduction

Growing emissions of $\mathrm{CO_2}$ and other greenhouse gases of the world's industry nations are a major concern of today's and future generations. Due to the increasing demands of the rapid industrialization in highly populated countries like China and India, the number of users of individual transport has been increasing dramatically in the last decade. Therefore, reducing consumption of fossil resources for future generations and releasing congested areas from exhaust gases is a necessity. One-step towards this is the exchange of the propulsion systems of vehicles from internal combustion engines to electric drives. That's why the German government formulated the aim to register one million electric vehicles (EV) by 2020, and up to six millions by 2030. This regulation is followed also by other industry nations; the USA defined a goal of one million EVs by 2015 and China six millions by 2020[3].

Thanks to this technology, it is possible to shift the local exhaust gas emission away from the world's mega cities to power plants outside these areas. Despite the fact that the worldwide production of renewable energy is still low[4], the Well-to-Wheel $\mathrm{CO_2}$ efficiency of electric vehicles[5] – also when considering the higher effort for production, transportation and storage losses – outperforms the one of conventional cars. Studies conducted by [Jrc13] have shown a theoretical improvement of 20% in comparison to vehicles powered by an internal combustion engine. In contrast to conventional cars, the range of EVs is still limited (for most series EVs it is up to five times lower[6]). This limitation is strongly connected to the high acquisition costs of the battery systems. Moreover, extreme temperature up- and downturns and excessive power demands may reduce the range and even the battery life time[7]. This amongst other factors, e.g. the limited heating comfort and the actual low gasoline costs, may have contributed to the fact that the admission statistic aims for the USA were mistaken in 2015[8].

In the last decade many researches engaged in optimal powertrain configuration, storage optimization or drivetrain and engine optimization. Besides, management strategies of the powertrain components and intelligent load management of comfort components with focus on energy efficiency were developed (see Chapter 3.1 for references and details). As a consequent step in this thesis the possibilities for future energy management strategies of complete x-by-wire electric vehicles with autonomous functionalities are investigated. The here proposed solution is inspired by a robotic centralized control architecture, which is hierarchically organized and refines the motion demands over the different abstraction layers with focus on the minimization of

[3]http://www.wiwo.de/unternehmen/auto/elektroautos-deutschland-verliert-den-anschluss-an-china-und-die-usa/12883950.html – Retrieved 03, 2017

[4] https://en.wikipedia.org/wiki/World_energy_consumption#By_fuel – Retrieved 03, 2017

[5] https://de.wikipedia.org/wiki/Well-to-Wheel – Retrieved 03, 2017

[6] https://en.wikipedia.org/wiki/Energy_density – Retrieved 03, 2017

[7] https://de.wikipedia.org/wiki/Elektroauto#Energiespeicher – Retrieved 03, 2017

[8] https://en.wikipedia.org/wiki/Plug-in_electric_vehicles_in_the_United_States#Markets_and_sales – Retrieved 03, 2017

the necessary power demand for solving the driving task. By means of these additional degrees of freedom for an energy management system a contribution for reaching the aim of reducing the propulsion's power consumption is imaginable.

1.1 Motivation and Objectives

The entirely newly developed robotic electric vehicular research platform of the German Aerospace Center (DLR) called ROboMObil, short ROMO, enables completely new approaches for future energy management systems. Its x-by-wire technology and a centralized control architecture inspired by robotics open up the freedom for intelligent algorithms to design future economical individual transport solutions. By means of these technologies the disbandment between the driver motion demands and the actual commanded values at the wheels of ROboMObil is accomplished. That way the driver gives either only rough commands which are refined by an autonomous arbiter module or ROMO can drive fully autonomously. A future scenario can be that many ROMOs are stored in a depot, which are called via a smartphone application by the user to drive to his location. After having finalized the driving task ROMO automatically returns back to its depot.

Within the ROboMObil project context different researches have focused on partial aspects such as fault detection and isolation [Ho16] or artificial intelligence agents [Sca16]. The goal of this doctoral thesis is, on the one hand, to develop an innovative energy management system (EM), focusing on the spatial motion. On the other hand, in the second part of the thesis a novel Modelica [Mod17] model based state estimation framework for the accurate control of the robotic electromobility research platform ROMO is developed. Its design pattern enables a flexible continuous-time prediction model formulation in Modelica without the need of hand discretization, linearization or event handling (e.g. vehicle standstill) by the user. Moreover, a relevant quantity of discrete estimation algorithms, extended with state constraints and delayed measurement handling, constitutes the need for a powerful tool to enhance complex control applications. These highly accurate state estimates stand out with exceptional time savings in implementation and reduction of errors in numerical efficient and correct implementations. It is worth mentioning that these technologies can also help to improve energy efficiency, since complex and accurate model knowledge is crucial (e.g. in [Arn12] 4 % fuel saving could be achieved by means of model based injection control for internal combustion engines).

1.2 Thesis Outline and Overview

The thesis is divided into six chapters, which are structured as follows:

In Chapter 2, the design patterns and system architecture of the robotic electric vehicular research platform ROMO is outlined. In addition the interfaces and system structures which are the basis for the energy manager and observer framework are described.

Chapter 3 presents the model based energy manager framework with its different layers of energy optimal control refinement. By means of a multiphysical Modelica model of ROMO the complete control framework is tested in closed loop with several test scenarios, which show the capability of the energy manager approach.

In Chapter 4, the theory of a novel model based Kalman filter framework, which is able to automatically incorporate multiphysical Modelica models in discrete-time estimation algorithms extended with constraint handling and real-time capable nonlinear moving horizon estimation, is outlined.

In Chapter 5, two applications of the proposed estimation framework are presented. A hybrid modeled state of charge battery observer and a constrained vehicle position, orientation and velocity observer for the EM framework are discussed.

The thesis is concluded by Chapter 6, in which a brief summary and a global overview about the work is elaborated and directions of future and ongoing research are mentioned.

2. Analysis and Design of the Robotic Electric Vehicle ROMO

This chapter is an extended version of [Bre11].

ROboMObil, short ROMO, is an electromobility concept based on intelligent central control of four wheel robots, which integrate the drivetrain, brakes, steering and semi-active dampers. This section outlines the conceptual design and development of DLR's robotic electromobility testing platform ROboMObil, which was part of the work during this doctoral thesis project. The motivation behind ROMO's wheel robot concept and the implementation details together with the suspension design are described in the following. Additionally, the electric power system, consisting of a lithium-ion battery storage system, providing a high voltage power for propulsion and a low voltage supply for vehicle control, is discussed. Finally, an overview of the central control architecture, inspired by the one which is used in today's modern robotic applications is provided. This architecture, both from constructive and cyber-physical system perspective, will be the basis for the proposed model based design of innovative operation strategies for ROMO in the upcoming chapters.

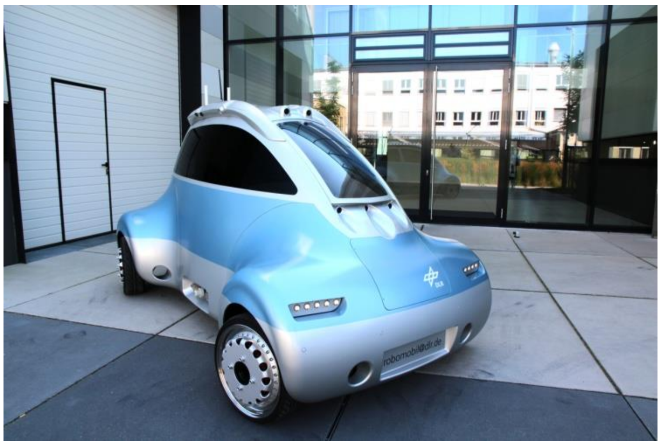


Figure 2.1: ROMO – the robotic electric vehicle in front of DLR's TechLab

2.1 State of the Art

The integration of the vehicle dynamic actuation systems in the wheel itself is a relatively new development, and in recent years, several concepts and prototypes of this idea have emerged. Examples include the Brembo In-Wheel [Brm15], the Schaeffler E-Wheel Drive [Sch14], Michelin Active Wheel [Gas08], Siemens VDO eCorner [Bry06], MIT Wheel Robot [Scm07], Volvo ACM [Jon07] and the Nissan Metamo system [Aso08]. In case of Michelin (Heuliez Will and Venturi Volage in [Dlr09]) and Schaeffler (Ford Fiesta E-Wheel-Drive), full size electrically powered vehicle prototypes with integrated wheel units have been demonstrated. Some developments focussed on extending the vehicle with the ability to rotate about its central axis or even to drive sideways, requiring extended steering angle ranges on all four wheels (e.g. Toyota Fine-X, Nissan Pivo II, or the MIT prototype with its wheel robot).

2.2 The Mechatronic System Architecture of ROMO

ROMO was designed completely new from a clean sheet, which enabled the conception of an innovative robotic research platform. This made it possible to layout the future of mobility without the constraints applied by the modification of a conventional vehicle, making use of advancements in intelligent systems from the field of robotics. The constraints for its development were to design an innovative mobility concept which is mainly operated in unstructured environment, with high maneuverability and a high level of autonomous control to cope with today's and future requirements of mega cities. Therefore, the decision was made to design a vehicle with four mechanically independent modules, the wheel robots, which do not have a physical linkage between the driver inputs and the commanded actuating variables. This design has a strong analogy to today's robots which also have separate integrated drivetrains in each joint, and all of them are controlled by a centralized architecture.

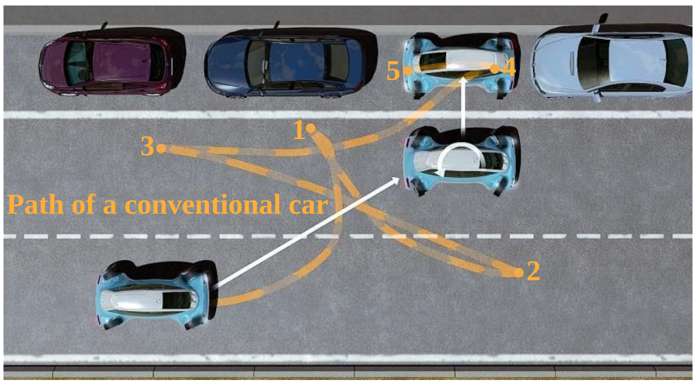


Figure 2.2: ROMO's high maneuverability enables simplified parking

2.2.1 The Wheel Robots Concept

The requirement of high maneuverability (compare Figure 2.2) on driven wheels spurs the use of wheel-mounted drive systems, capable of overcoming the wheel steering angle limitation that is typically present in conventional cardan shaft mechanisms. This fact leads to the development of the so called wheel robot concept (Figure 2.3).

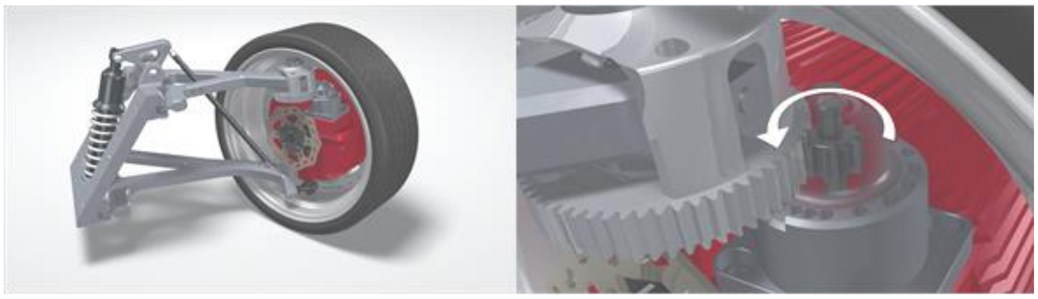


Figure 2.3: The wheel robot with in-wheel actuators and suspension (left);
The wheel carrier mounted steering mechanism (right)

The wheel robots approach is derived from robotics and Mars rovers [Mic08], with all the actuators integrated in-wheel. With the integrated in-wheel steering mechanism and in-hub traction motors (Figure 2.3), an extended steering angle of 95° to −25° degrees is realized. As a result, ROboMObil is able to rotate around its own vertical axis (centrically and eccentrically) and move sideways. Such a degree of steering freedom would be difficult to achieve with a chassis mounted drive motor and drive shafts.

The Wheel Robot Actuators

The permanent magnet synchronous motor (PMSM) in-wheel direct drive architecture provides a nominal motor speed of 1000 rpm without field weakening, which is equivalent to a maximum vehicle velocity of 100 km/h with tires of the dimension 165/35 R17 installed on ROMO. Each motor can deliver a peak torque of 160 Nm which is reduced to 40 Nm during continuous operation. The motors have an air-cooled inner-rotor design, with the aluminum stator housing simultaneously acting as the heat sink and carrier for the steering mechanism. Analyses have shown air-cooling to be sufficient due to the high efficiency factor of the electric drive, leading to a significantly simpler and lighter design compared to liquid-cooling. The stator housing also plays the role of the wheel carrier with its connections to the ball joints at the end of the upper and lower wishbones (see Figure 2.6 – steering axis).

The individual wheel steering consists of a rotary electric actuator mounted on the traction motor housing (compare Figure 2.3 – right). The pinion gear rotates around a larger gearwheel, which is rotationally constrained to the upper wishbone, while remaining coaxial with the king pin axis (Figure 2.4). Compared to steering actuation via a conventional tie-rod, this direct rotational actuation of the steering axis allows a very large steering angle range limited only by the physical contact between the wheel and the wishbones. This rotational constraint on the large gear wheel presents a mechanical challenge as it must be effectively connected to the upper wishbone via a cardan joint. This is solved by the use of a novel “sliding-block” mechanism (see Figure 2.3 – right and Figure 2.4 – top) that showed profound reliability during testing.

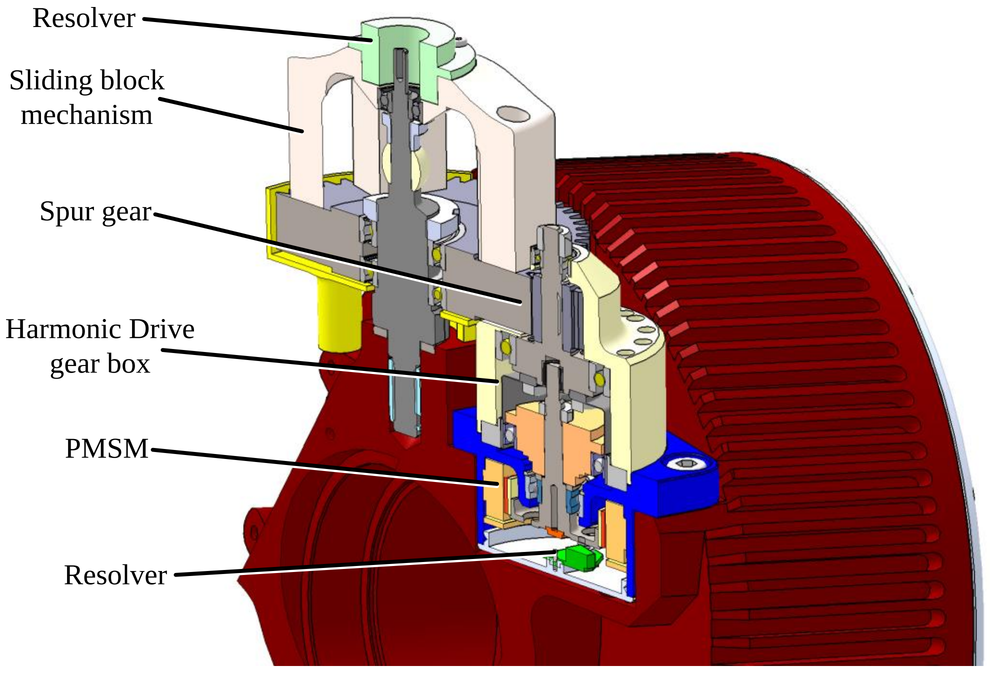


Figure 2.4: Cutaway view of the wheel hub integrated steering mechanism

The steering actuator uses a $370\,\mathrm{W}$ permanent magnet synchronous machine (PMSM) with a strain wave gearing from Harmonic Drive AG. The system is specified to achieve similar road wheel steering speeds to that of a human driver in extreme situations, which is accepted to be approximately $80\,°/\mathrm{s}$. This corresponds to $1200\,°/\mathrm{s}$ at the steering wheel for a sportive steering ratio of 15:1 in a conventional car, well above the $500\,°/\mathrm{s}$ specified in test procedures [Iso88] and the $1000\,°/\mathrm{s}$ used in industrial circles. For the control of the steering angle two angular sensors are used. The first, a resolver, is used to measure the rotation between upper wishbone and the wheel carrier. The second sensor is located at the electric drive side and is also used for field-oriented control of the motor. The use of the second wheel mounted resolver compensates motor oscillations due to elasticity in gears and guarantees stationary exactness.

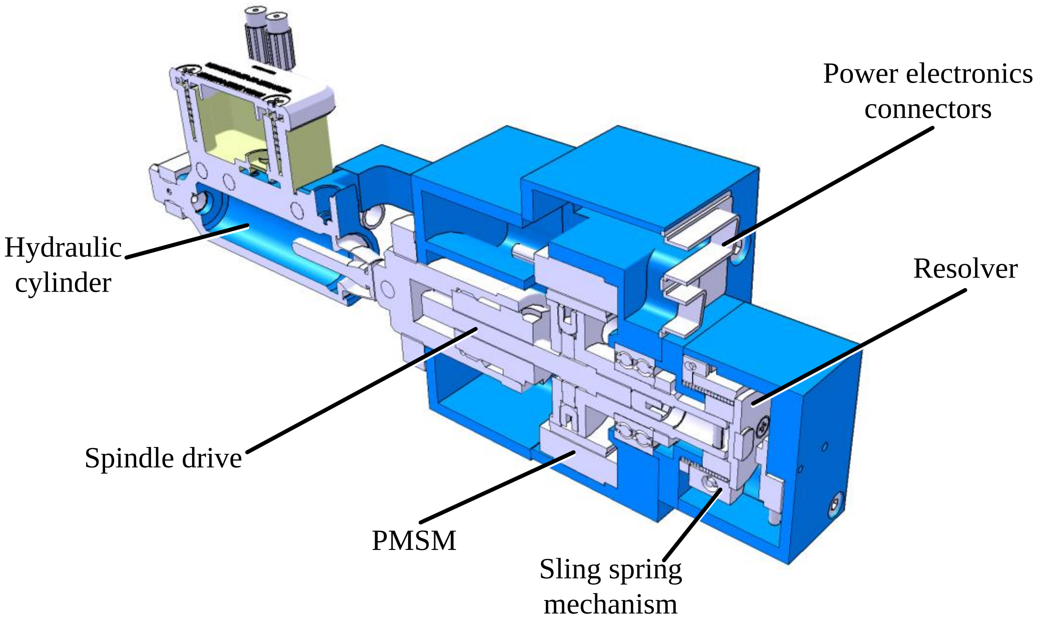


Figure 2.5: Cutaway view of the electro hydraulic brake actuator

The hydraulic friction disc brakes are based on go-kart calipers produced by Magura GmbH. They were selected due the similar weight per brake ratio between go-karts and ROMO. The master cylinder is driven by an electro mechanical linear actuator comprising a spindle drive and a PMSM motor (see Figure 2.5). This system allows braking with a deceleration of $8.8\,\mathrm{m/s^2}$, exceeding the ECE R13 service braking requirement of $6.4\,\mathrm{m/s^2}$. This additional friction brake system, lockable via a sling spring mechanism and powered by redundant low volt batteries, is necessary on ROMO for three main reasons: First, to provide a parking brake function, and to hold the vehicle on an incline during standstill. Second, to provide an emergency brake system in case of loss of the high voltage power supply for the traction motors (see energy system Chapter 2.3.1). Third, to assist the traction motors, working in generator mode, for providing a higher deceleration potential.

The Wheel Robot Suspension Design

The demanding requirements of the wheel steering mechanism, such as a steering lock angle of 90°, and a strut capacity of 1000 kg total mass in maximum lateral accelerations of 1 g, presented a challenge for the suspension design. The considerations of stiffness, simplicity for prototyping and availability of proven design methods for high load vehicle dynamics led to the adaption of the well proven double wishbone layout. Other designs with movable linkages or top-hinged wheel carriers (like on a shopping trolley) [Aso08] suffer from high complexity, significant design effort and stiffness issues, which could not be overcome within the scope of this project.

In terms of mechanical component design, the large steering angle range demand a shaped, slim wishbones construction (cf. Figure 2.6). Since this wishbone also had to carry the steering moments (the steering actuator lies on the wheel end of the wishbone), the loads led to a challenging task. An analysis with modelling of the lower wishbone as a flexible body using Modelica demonstrated the sufficient stiffness and strength of the design in a vehicle driving simulation, with the deflections having minimal effects on the vehicle driving behavior [Har10].

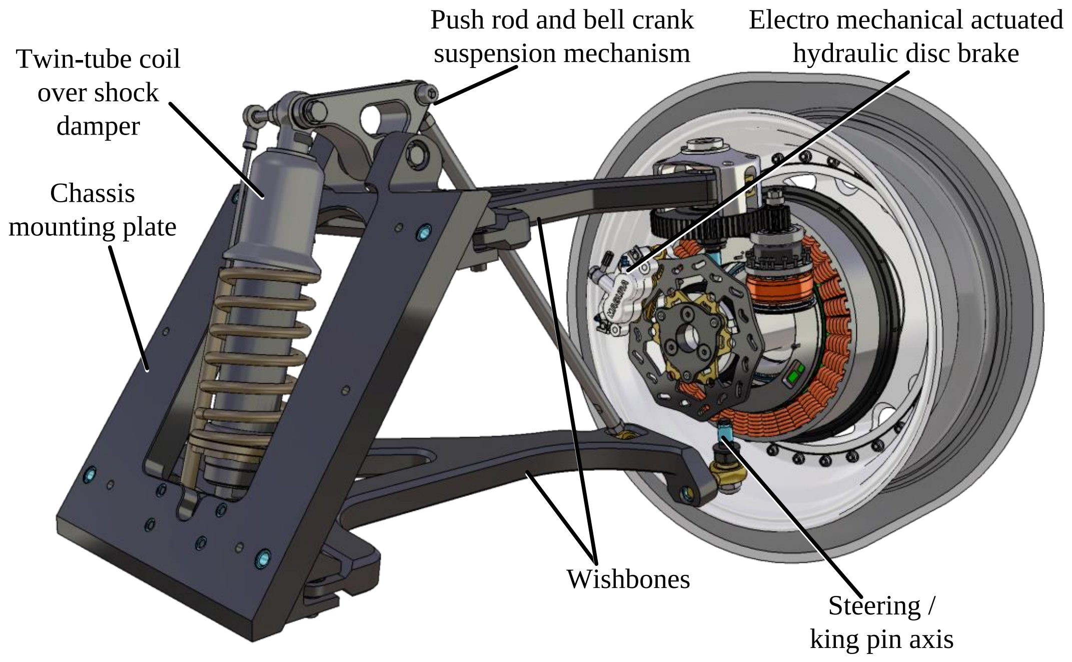

Push rod and bell crank suspension mechanism
Electro mechanical actuated hydraulic disc brake
Twin-tube coil over shock damper
Chassis mounting plate
Wishbones
Steering / king pin axis

Figure 2.6: Suspension design in CAD (wheel hub motor housing is faded out)

In order to verify these challenging requirements before the construction of physical prototypes, the design of the suspension geometry was preliminary validated using a 3D multi-body simulation. The detailed suspension was modelled with rigid links in Modelica (see Figure 2.7), and the static geometric relationships were evaluated. The first designs of the wheel robots were carried out with identically long wishbones, which resulted in the lack of pitch angles and a wheel contact point in line with the wheel axle. Although this makes it easy to integrate the steering mechanism, the mechanical advantages of a state of the art vehicle suspension [Trz08] would not have been utilized.

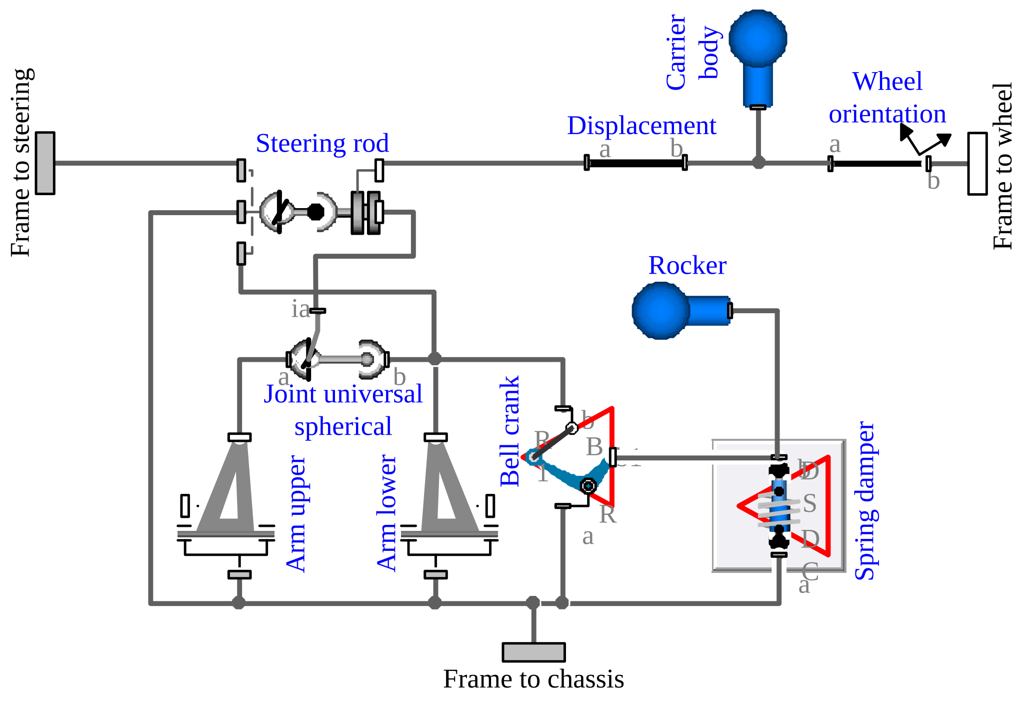


Figure 2.7: Modelica multi-body suspension model of one wheel robot

By use of the Modelica DLR Vehicle Control Library (multi-body simulation) and the DLR Optimization Library (optimization of the hard points within the constructed space) the suspension was optimized to meet the criteria of vehicle stability, drivability, energy efficiency and self-stabilization in case of malfunction of the by-wire steering. As tuner variables the length of the wishbones, the height position of the hardpoints on the chassis and wheel carrier side, the relative pitch angle between the upper and the lower wishbone, as well as the expansion axis inclination and offset were chosen. In Table 2.1 a complete list of the static characteristic values in k_0 (vehicle height at static compression stroke) of ROMO's axles, after an iterative optimization process, is given and will be discussed in the following. For more information regarding the influence of the single characteristics please refer to [Trz08].

Table 2.1: ROMO's kinematic characteristics in static compression stroke

Characteristic	**Front axle**	**Rear axle**
Camber angle	−0.5°	−0.5°
Toe angle	0.0°	0.0°
Caster angle	4.00°	4.00°
Inclination	8.01°	8.01°
Steering offset	2.29 mm	2.29 mm
Caster trail	25.16 mm	25.16 mm
Roll center height	72.53 mm	81.31 mm
Spring ratio	0.81	0.81
Anti-dive / Anti-squat effect	55.38 %	54.55 %

The requirement of a high operating speed (up to 100 km/h) demands automotive specifications regarding the steering axis design, such as the wheels being self-stabilizing in the absence of actuator moment. In short, this requires a positive trail value and caster angle. This constructive measure supports the energy efficiency of the by-wire steering due to reduction of the necessary energy in case of straight-on running (wheel self-centering) and steering back from cornering (wheel self-recirculation). Additionally, the steering offset is designed to a minimum value to minimize the necessary holding steering torque during acceleration and deceleration; thereby the loads on the steering actuators are reduced. The in-wheel motor design offers advantages in anti-dive and anti-squat design because the same lever for both drive and braking is effective. Here a good trade-off between vehicle body movement and wheel-load fluctuation was chosen, which is necessary to maximize the energy recuperation during braking. The low center of mass (the high voltage battery pack and other components are positioned close to the bottom of ROMO – see Figure 2.13) in combination with a good value of the geometric roll center height enabled an axle design that does not need the presence of an anti-roll bar. This simplifies the construction and gives a stable cornering and transient vehicle dynamics behavior to the vehicle. Whereas in the first suspension design the coil over shock damper could be placed between the two wishbones, a constructed space analysis made it necessary to move the damper to the chassis side transmitted by a narrow push rod and bell crank suspension mechanism (cf. Figure 2.7.). Beside the constructive benefit, this enabled to design a progressive spring ratio, which gives advantages to the road holding with limited available shock travel of about 160 mm (see trajectories in Figure 2.8). The resulting vehicle trajectories characteristics are given in the following plot. The symmetrical camber gradients at front and rear axle lead to a good support in lateral direction and help to reduce the tire friction during cornering. The toe-in trajectory (especially at the rear axle) is designed to self-stabilize the vehicle during emergency brakes and the progressive spring rate (by use of the push rod and bell crank mechanism – see Figure 2.6) assists to reduce the vehicle body movement and harsh collisions with the damper bump stops.

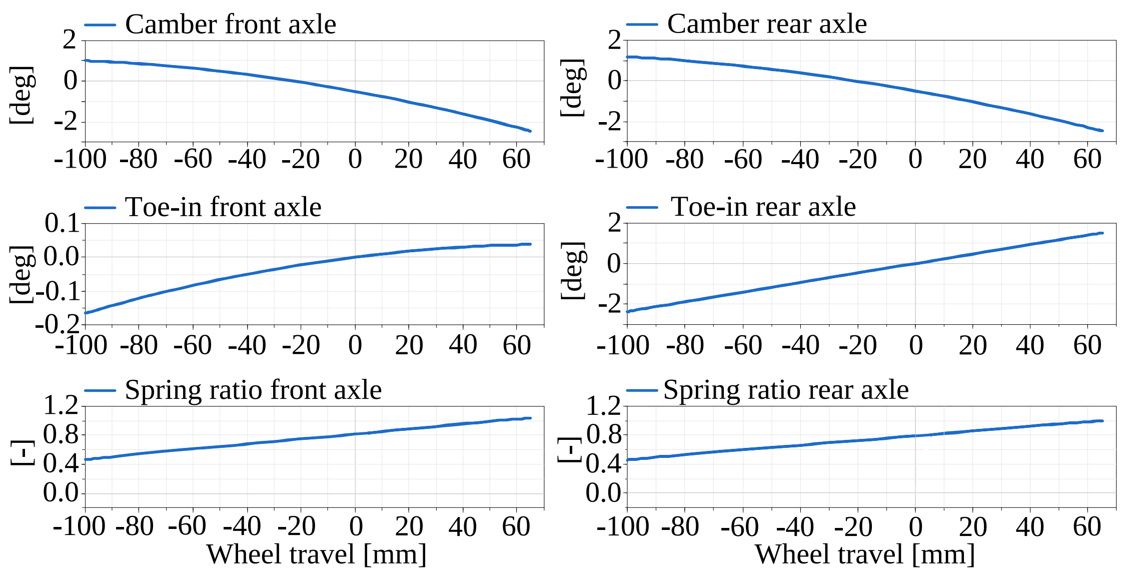


Figure 2.8: ROMO's wheel trajectories after optimization

To cope with the mass ratio of one (wheel) to four (body), the twin tube dampers (passive) are equipped with an elastomer bump stop in compression and a hydraulic rebound stop. These measures in combination with an optimized compression and rebound damping setup guarantee secure wheel ground contact and mechanical stability even in critical situation, when the wheel loses ground contact.

Another usage of the simulation model (cf. Figure 2.7) has been the evaluation of wheel camber variation as the wheels steered through the angle range from 0° to 90°, as shown in Figure 2.9. The front wheels take on positive camber values of 3.4°, and the rear wheels values of 11°. By reason of the limited time of operation in the sideward driving mode, these values can be seen as acceptable for the investigation on ROMO’s high maneuverability features.

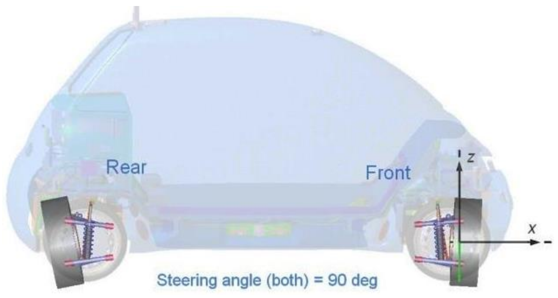


Figure 2.9: Suspension geometry during driving sidewards

Additionally it is worth mentioning that in nominal operation mode ($-25°$ to $+25°$ steering angle) the camber trajectories stay in optimal boundaries to support the vehicle dynamics with the benefit for self-steering behavior (compare Figure 2.10).

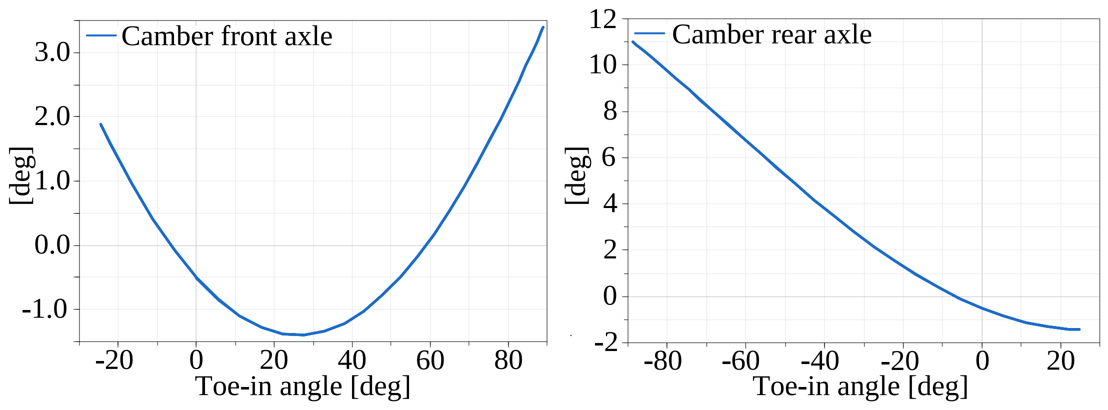


Figure 2.10: The wheel robots camber angles trajectories

Moreover, the loads acting on the vehicle axles are important criteria for the design of the mechanical components and mechatronic vehicle dynamic systems. They define, together with the above explained kinematics, constraints for the CAD construction regarding the strength of the components. These are evaluated through dynamic simulations of the complete vehicle model, for example in virtual prints from the curb maneuvers or maximum bump-stop force evaluation in case a wheel loses contact to the road.

2.2.2 The Four Modules Concept

The four wheel robots are pairwise condensed in two axle modules, which have identical electrical systems of electric drive inverters, backup batteries and step-down DC/DC converters (compare Figure 2.13). Thereby ROMO's architecture allows modifications to the chassis without affecting the powertrain, which is located completely in the two axle modules, with the actuators fully integrated within the wheel robots. The other modules are the driving module (body), which forms the structure of the vehicle and carries the cockpit, and the energy module mounted beneath the cockpit floor (see arrangement in Figure 2.11).

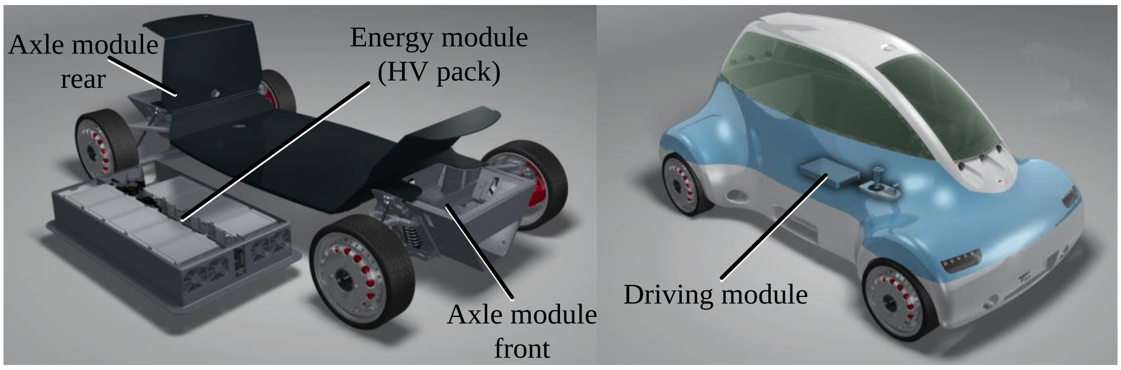


Figure 2.11: ROMO's four module concept

This four module concept (two axle modules, body and battery) enables an interchangeable design for other mobility concepts. For instance the exchange or extension of the battery module with an internal combustion engine range extender would be possible. Another option could be a body module that offers space for more than two persons or a pure transport module with more than two axles.

The fourth vehicle module is the energy unit – in the ROMO application it is based on a lithium-ion battery storage (see also Chapter 2.3.1). It is inserted and removed from the bottom side (see Figure 2.11 – left) of the vehicle, which corresponds to the design proposed in [Wik17]. This concept was proposed to allow exchanging the battery instead of charging it at the service station to shorten waiting times, and in the context of the ROMO prototype this allows the possibility of using alternative electrical energy sources in future developments, such as hydrogen fuel cells or the DLR-developed free-piston linear alternator [Poh05].

This interchangeability is achieved through local intelligent control units for each actuator (integrated into an axle module). A central control computer (part of the driving module) is communicating with these local units to coordinate the vehicle's motion.

With the independent control of four wheel steering actuators, four electric wheel hub motors and two friction brake actuators, the vehicle dynamics variables of yaw rate $\dot{\psi}$, side slip angle β and vehicle velocity $\boldsymbol{v} = \{v_x, v_y\}$ can be decoupled and independently controlled. In Chapter 3 this property will be used to design an energy optimal control for ROMO's actuators.

2.3 ROMO's Electrical System and Control Architecture

In this chapter the electrical system, which enables the x-by-wire architecture of ROboMObil is discussed. It supplies the central control architecture and mechatronic infrastructure with energy in a complex multi-voltage network in combination with safe fallback strategies in case of a malfunction.

2.3.1 The Vehicle Electrical System Design

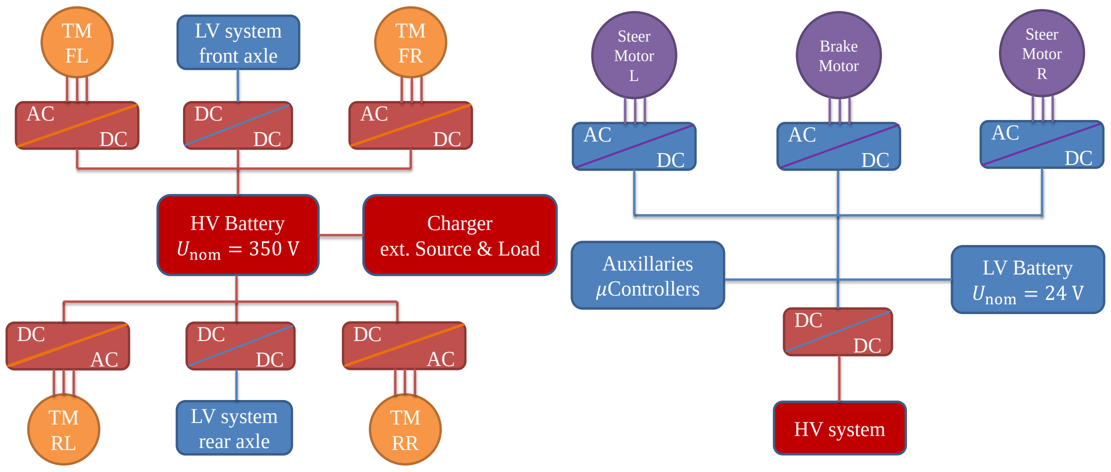


Figure 2.12: Electrical architecture scheme of ROMO – high voltage System (left) – low voltage system of one axle module with identical composition for front and rear (right)

The electrical architecture consists of a high voltage (HV) battery as the main power source (see Figure 2.12). It is directly connected to the inverters of the four traction motors (TM), and also to the two step-down DC/DC converters to supply the low voltage (LV) system. A connection with an external high voltage DC power supply allows on-board electrical system investigations, like hardware-in-the-loop (HIL) controlled external sink and source simulations, as well as fast charging of the vehicle. The operation modes of the electrical system are controlled by the on-board network control unit (BNCU) which is implemented on an embedded rapid prototyping central controller of ROMO, one of the components of the hierarchical controller architecture.

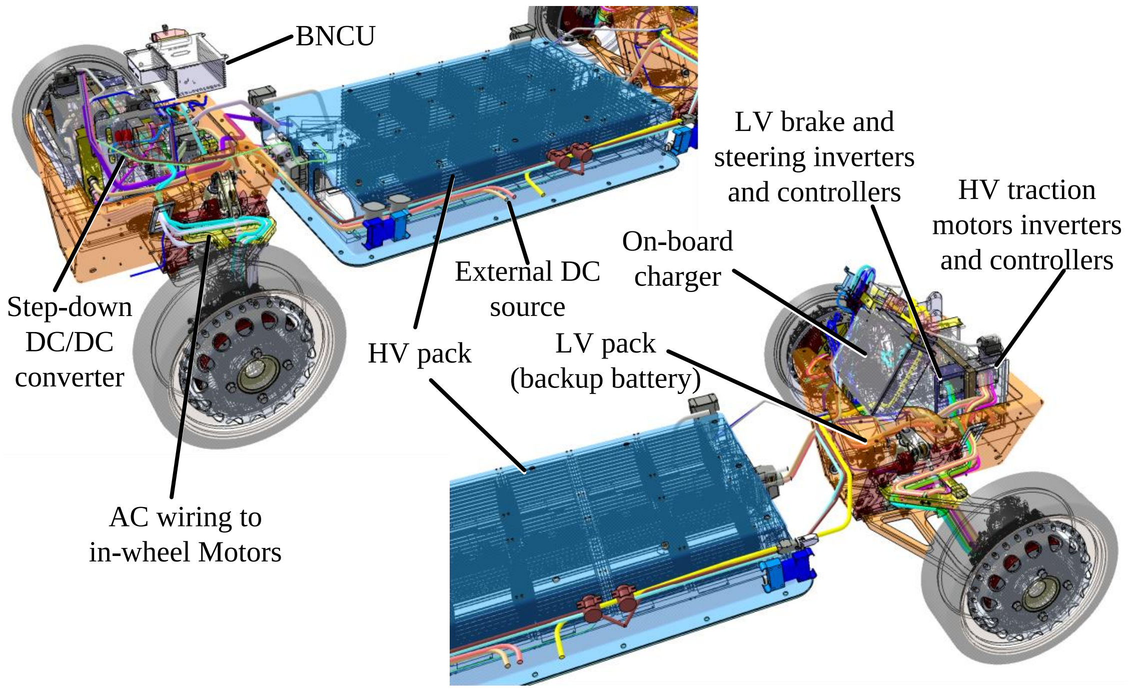

BNCU
LV brake and steering inverters and controllers
HV traction motors inverters and controllers
On-board charger
External DC source
Step-down DC/DC converter
HV pack
LV pack (backup battery)
AC wiring to in-wheel Motors

Figure 2.13: CAD detail views of ROMO’s electrical systems

The air-cooled HV battery unit consists of 90 pouch type cells divided into nine stacks, providing a nominal capacity of 14 kWh at 350 V (cf. Figure 2.13). Simulations using DLR’s Modelica PowerTrain Library [Tob07] showed that ROMO has a range of approximately 100 km. To ensure operational safety, the HV electrical system is designed in accordance to the ECE R100 regulations for electric and hybrid vehicles. An isolation monitor is implemented, as safety measure for the high voltage system, alongside a HV-interlock signal wire, which ensures that all HV components are connected and secured. Any breach of these security measures leads to a release of the battery main relays. In this case the LV network in the axle modules prevent the car from an uncontrolled behavior and guarantees sufficient energy to bring the vehicle to a safe state by use of the steering and mechanical brake actuators.

The development of operating strategies to optimize the energy management requires high-fidelity models of the energy sources and sinks. In the case of the HV system, these are the lithium-ion battery pack and the in-wheel drive motors respectively. The information from systematic battery cell testing was used to configure and parametrize a real-time capable battery model. Accordingly, a model based state observation concept was developed to predict the states within the battery, including state of charge (SOC) and power availability, see Chapter 5.1.

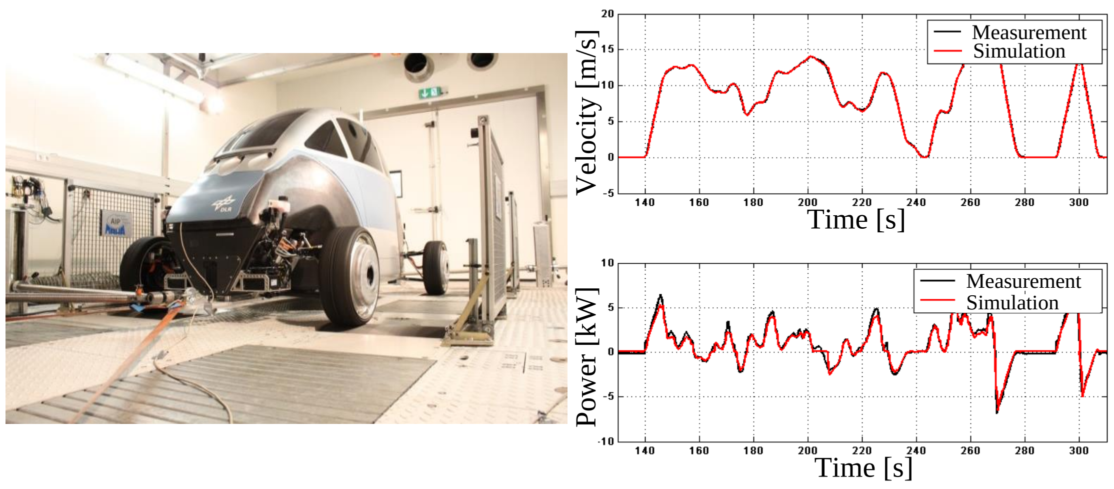


Figure 2.14: Test results of ROMO’s longitudinal dynamics on DLR’s roller test rig

Longitudinal dynamics tests with the ROMO prototype were performed on DLR’s chassis dynamometer in Stuttgart. Amongst other tests, driving cycle investigations were performed using an automatic controller on rollers simulating the velocity dependent resistances on the vehicle. The data was then validated with the results from a driving cycle simulation to check the accuracy of the models. In Figure 2.14 the vehicle drove the so-called “Stuttgart suburban” cycle, a DLR driving cycle based on a speed profile recorded in a real-life drive in suburban traffic of Stuttgart. The collected data during the tests were also used to identify ROMO’s powertrain and electric propulsion behavior for ROMO’s multiphysical Modelica model used in Chapter 3.7.

2.3.2 The Central Control Architecture

ROMO’s central control architecture is driven by modern robot applications with a hierarchical structure feeding high level commands (from perception, cognition or autonomy) to a general motion demand calculation layer. This layer is a controller that uses the feedback of the vehicle dynamics sensor system and calculates a triple of plane movement requests (e.g. $v_x^{\mathrm{C}}, v_y^{\mathrm{C}}, \dot{\psi}^{\mathrm{C}}$) of the vehicle body in the $\mathbb{R}^3$ space for the actual high level command. An underlying control allocation algorithm determines, by use of intelligent optimization methods, a feasible set of commanded actuator values. Finally, these set-points are applied by the local controllers of the intelligent wheel robot units and commanded by the power electronics to the distinguished drive.

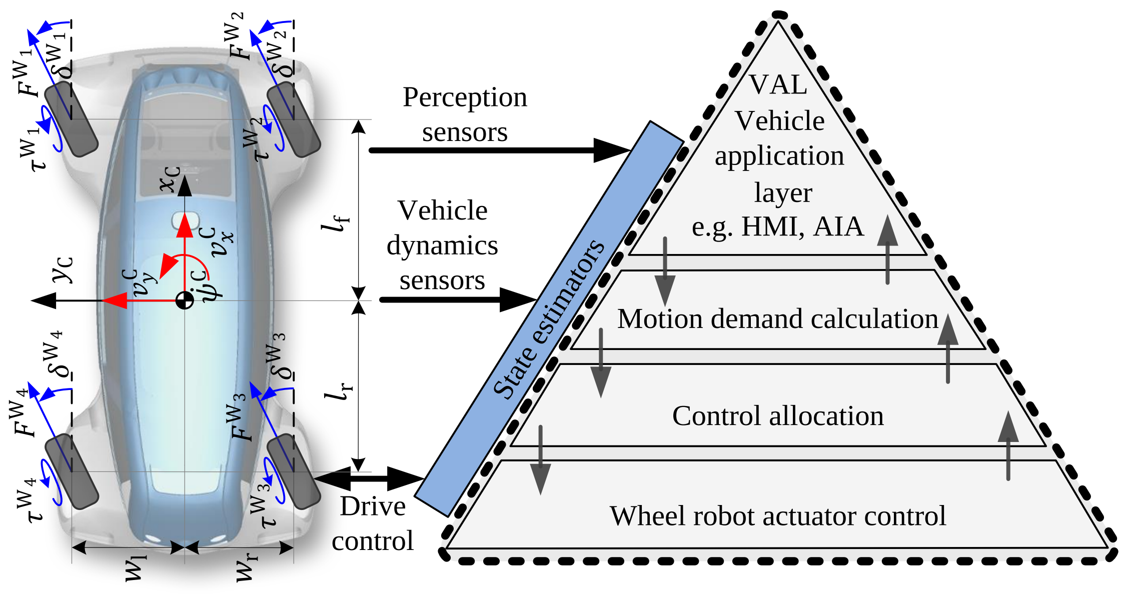


Figure 2.15: ROMO's pyramidal control architecture

The presented pyramidal control architecture in Figure 2.15 has the main advantage that the complexity is reduced by defined responsibilities, interfaces and intra-system communication between the different levels. In comparison to this approach, today's vehicle architectures are mostly based on function grouping of mechatronic components and the associated microcontrollers. Nowadays, there is a slight change in upcoming vehicle systems by grouping different functionalities on one microcontroller [Scu15], but nevertheless they do not have an overall integrated chassis management.

In detail, the flow of commands in ROMO begins with environmental sensing and recognition of a motion demand, which can be set by the in-vehicle human machine interface (HMI), by a driver in a teleoperation station, or by a path planner for autonomous driving – the artificial intelligence agent (AIA). The vehicle is equipped with a 360° 3D-stereo camera system to orientate e.g. in today's mega cities. With the raw camera data different algorithms, developed at DLR for computer vision in robot application [Hir08], can be applied to detect obstacles and road users. Beyond that, information from car2x networks or low distance ultrasound sensors can improve the data mining [Sca11]. This sensor data is then processed in the vehicle application layer (VAL). In this level different algorithms can be integrated and investigated – e.g. a conventional human control interface for performance comparison studies up to algorithms for autonomous driving. One main research focus of the ROMO project is to implement algorithms in this shared autonomy layer in which the human operator expresses a raw motion demand that is subsequently refined on its own by the subordinate system, using the available information about the environment and the vehicle's system states. Consequently, the vehicle can refuse to cause an accident, by preventing the driver from leaving the road. In the underlying motion demand calculation layer the planar vehicle motion demand is calculated by use of the VAL trajectory information. This layer feeds back the kinematic and driving dynamics constraints to the VAL to incorporate them in the trajectory generation module. The motion request is then (as mentioned above) refined within the control allocation module that generates the set-points for the local intelligent controllers. For this purpose the motion information – gathered by a global navigation satellite system (GNSS) and an inertial measurement system in combination with a vision based ego motion estimation – and the actual performance capacity of the wheel robots are taken into account. Finally, the high fidelity wheel robot controllers adjust precisely the demanded set-points of the ten mechatronic driving dynamics actuators.

2.3.3 The Vehicle Dynamics Sensors and Actuators Network Architecture

As a motivation for the following chapters, where the focus is mainly on ROMO's vehicle dynamics sensors and actuators, in Figure 2.16 the – for these task relevant – connections and components are given. The arrangement is designed to meet the central control architecture introduced in Chapter 2.3.2 with two synchronized rapid prototyping controllers (dSPACE MABX2 and ABX) which represent ROMO's central control unit. The wheel robots with the traction and steering drive torques $\tau^{(\mathrm{ST}),\mathbf{W}}$, angular velocities $\omega^{(\mathrm{ST}),\mathbf{W}}$ and steering angles $\delta^{\mathbf{W}}$ quantities, as well as the Correvit optical measurement unit (used to measure the vehicle's longitudinal and lateral speed $\boldsymbol{v}_{\mathrm{act}}^{\mathrm{C}} = \{v_{\mathrm{act}_x}^{\mathrm{C}}, v_{\mathrm{act}_y}^{\mathrm{C}}\}$), are connected via high-speed CAN busses. The inertial measurement system with a differential corrected global navigation satellite system (GNSS) to measure the vehicle states such as the positon $\boldsymbol{p}_{\mathrm{C_{act}}}^{\mathrm{I}}$, yaw angle $\psi_{\mathrm{C_{act}}}^{\mathrm{I}}$ and yaw rate $\dot{\psi}_{\mathrm{C_{act}}}^{\mathrm{I}}$, or the velocity $\boldsymbol{v}_{\mathrm{act}}^{\mathrm{C}}$ is wired by an Ethernet bus to the central control. Additionally, the wheel $\boldsymbol{a}^{\mathbf{W}}$ and chassis body $\boldsymbol{a}_{\mathrm{body}}$ accelerations as well as wheel travel sensors $s^{\mathbf{W}}$are tethered via analog inputs. Both, the OxTS measurement unit and the optical odometry sensor are suited for the experimental evaluation of driving experiments.

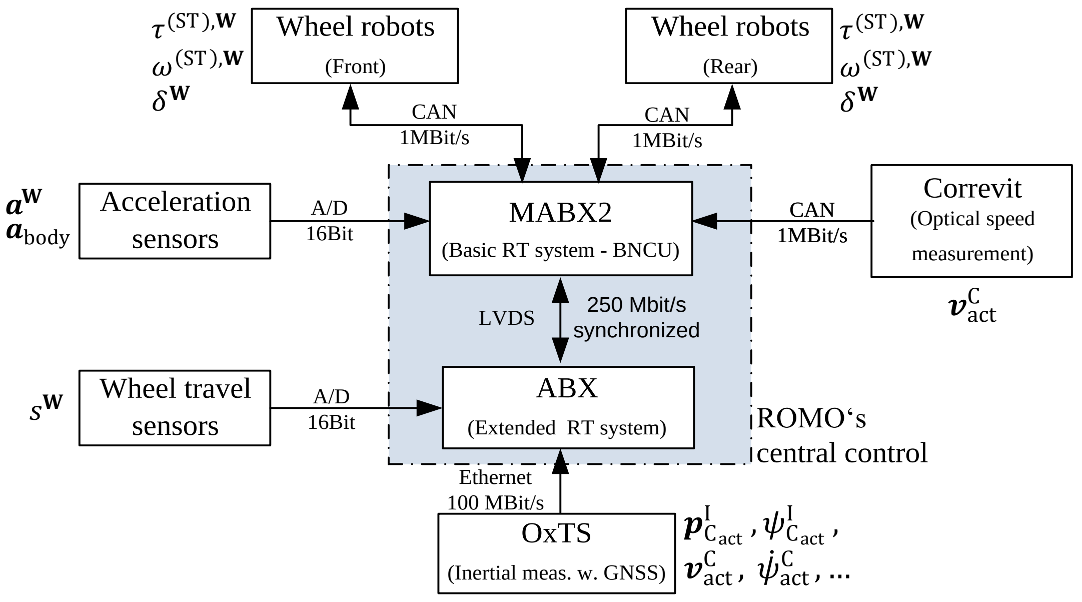

$\tau^{(\mathrm{ST}),\mathbf{W}}$
$\omega^{(\mathrm{ST}),\mathbf{W}}$
$\delta^{\mathbf{W}}$
Wheel robots (Front)
Wheel robots (Rear)
$\tau^{(\mathrm{ST}),\mathbf{W}}$
$\omega^{(\mathrm{ST}),\mathbf{W}}$
$\delta^{\mathbf{W}}$
CAN 1MBit/s
CAN 1MBit/s
$\boldsymbol{a}^{\mathbf{W}}$
$\boldsymbol{a}_{\mathrm{body}}$
Acceleration sensors
A/D 16Bit
MABX2 (Basic RT system - BNCU)
CAN 1MBit/s
Correvit (Optical speed measurement)
$\boldsymbol{v}_{\mathrm{act}}^{\mathrm{C}}$
LVDS
250 Mbit/s synchronized
$s^{\mathbf{W}}$
Wheel travel sensors
A/D 16Bit
ABX (Extended RT system)
ROMO's central control
Ethernet 100 MBit/s
OxTS (Inertial meas. w. GNSS)
$\boldsymbol{p}_{\mathrm{C_{act}}}^{\mathrm{I}}, \psi_{\mathrm{C_{act}}}^{\mathrm{I}},$
$\boldsymbol{v}_{\mathrm{act}}^{\mathrm{C}}, \dot{\psi}_{\mathrm{act}}^{\mathrm{C}}, \ldots$

Figure 2.16: ROMO's vehicle dynamics sensors and actuators architecture

The design of ROboMObil has been a joint research project of the ROboMObil team (consisting of eight core team members), where the author has been the project leader and his contributions can be summarized as follows:

- Simulative dimensioning of the high voltage battery system of ROMO by means of multiphysical simulation models, as well as the specification for security measures, heat management and vehicle package integration.
- Design, parameterization, and validation of Modelica based (electrical) component models for ROMO’s powertrain dimensioning and assessment of different concepts.
- HIL testing of the high voltage battery system by means of a high performance and freely programmable high voltage source and sink power supply.
- Significant co-development of ROMO’s electrical system and central control architecture with the focus on vehicle dynamic control and sensor systems.
- Overall vehicle packaging concept development including interior, HMI, as well as electrical and mechanical components namely the four module concept consisting of a chassis, two axle modules and a battery module.
- Design and optimization of ROMO’s suspension geometry and characteristics under the restriction of available constructive space in the axle modules and the avoidance of collision with the sliding block mechanism of the steering strut.

3. The Model Based Energy Manager Framework

This chapter is an extended and enhanced version of [Bre12], [Bre14]. The deployed path interpolation algorithm from [Rit15] is a modified development to match the here developed trajectory controller (Chapter 3.4.2) and its outputs to the energy optimal control allocation problem formulation (Chapter 3.5).

3.1 State of the Art

In today's conventional vehicle architectures there are a large number of electrical loads, many of them related with comfort aspects, like seat heating but also safety critical ones like advanced driver assistance functionalities. Therefore, in the last two decades a lot of effort was spent to protect the conventional $12\ \mathrm{V} - 14\ \mathrm{V}$ on-board electrical system from under/over voltage states or exceeding power requests. Strategies to avoid these flaws are on the one hand topology modification of the on-board electrical system through the application systems like a voltage-stabilization-unit, by means of a buck converter during motor starting, or a decoupled on-board electrical system, i.e. a second battery [Fro08]. On the other hand, control interventions during dynamic performance requests through measures like increasing the engine idle speed, soft switching of the loads or reducing the power request of the loads during performance peak situation may also alter the on-board electrical system stability [Buc08].

To further reduce the $\mathrm{CO_2}$ emissions hybrid powertrain configurations like parallel hybrid electric vehicle (PHEV) or serial hybrid electric vehicle (SHEV) have been developed. They have different types of power sources – e.g. internal combustion engines (ICEs) – and energy storage – e.g. high voltage (HV) lithium-ion battery packs – and open up a manifold of optimization possibilities for the energy management control. A good overview and detailing of the most convenient architectures is given in [Kam11].

Especially in the above mentioned hybrid powertrain configurations, on-board electrical system management techniques achieve profound reliability in stability and energy efficiency. For example in [The07] an energy management approach with pre-allocated table based prioritization of the loads, classified in safety critical classes and preferred comfort assignments in a real-time management system is proposed. Defining the on-board electrical system as a market model, according to theory from economy science, showed good results for vehicle applications in [Ens08], [Buc08] and was recently also successfully applied in an improved form to an aviation application [Scl15]. The here used market theory employs the optimal allocation of limited resources from providers to allocators. The actual market demand and offer of energy yields a market price and every component has a certain cost function in dependency of it. An auctioneer module supervises permanently the excess demand and recalculates at every sample time instance the market price to achieve a balanced equilibrium. To ensure reliable system operation safety critical functions do not have a price limitation for their power demand.

Other strategies for energy management aim on the efficient propulsion source distribution (especially for PHEV), energy recovery during deceleration and traction battery charging management. By means of on-board sensors, map data systems with virtual horizon, actual drive style classification algorithms, wireless communication, and efficient energy management systems for the longitudinal powertrain could be developed in the past decades. Appreciable research studies for these kinds of energy management strategies can be found in [Won03], [Wid08], [Toe08] and most recent refined in [Kno16].

3.2 A Motion Request Approach for Future Energy Management

As presented in Chapter 2.3.1, ROMO has a complex electrical on-board architecture with only one central power source for propulsion controlled by the on-board network control unit. The conception of this doctoral thesis project is not to optimize the energy flow between the components or to influence the power distribution between a combustion engine and electric motor (for details see Chapter 3.1). Instead here it is investigated how energy flows can be optimized on the level of the motion demand – in other words, how energy can be saved by travelling a route under the utilization of the complete road width and model based knowledge for intelligent control algorithms.

The research aims for the motion demand driven energy manager approach are:

- Utilizing the full road space for the vehicle movement,
- integrating the vehicles physical limits and road incline in the planning and
- commanding the wheel robots actuators of ROMO in sense of energy efficiency.

In [Ros07] it is proposed to apply today's common pyramidal organized management structure to energy management systems for multi-propulsion vehicle architectures (as described in Chapter 3.1). Inspired by this approach an inverse pyramid for energy optimal driving strategy management is proposed on the left side of Figure 3.1.

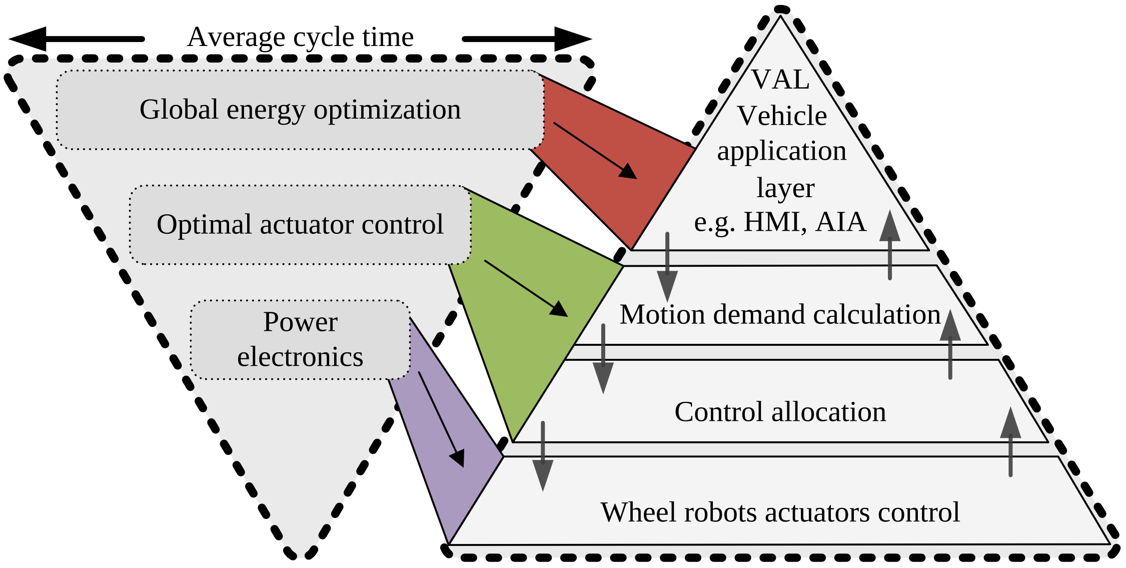

Average cycle time
Global energy optimization
Optimal actuator control
Power electronics
VAL Vehicle application layer e.g. HMI, AIA
Motion demand calculation
Control allocation
Wheel robots actuators control

Figure 3.1: ROMO's energy manager scheme

The width of the inverse pyramid levels donates a proportion for the average cycle-time of one hierarchical control level. Moreover, a mapping of these levels to the control strategy of ROMO (compare also Figure 2.15) is given. On the top of the inverse pyramid – with the longest possible calculation time – the global energy optimization module is present. Its task is to define an energy optimal path, with corresponding vehicle velocity, for a given route. It should also be able to integrate information like road obstacles and other vehicles. On the level of optimal actuator control the task is to generate the set-points for the wheel robots intelligent actuators in an energy optimal manner, trying to keep the vehicle on the precalculated path and velocity predetermined by the top management level. The lowest level of the pyramid, the power electronics and the corresponding local current control loops for the actuators' electric drives, are not part of this dissertation project. Their development and optimization was part of the BayStMWIVT "Radroboter – eCorner" EMO-1002-0002 public funded project.

The development of the complete control strategy is divided into three chapters. In Chapter 3.3 a real-time capable path planning module is developed which optimizes the spatial path, combined with constrained vehicle speed profile generation, in an energy optimal way. In addition, investigations on efficient real-time capable problem solving formulations are carried out. Subsequently in Chapter 3.4 the extraction of the actual controller set-point for the vehicle motion control from time independent path and velocity representation and the design of the error compensating controllers for the planar vehicle movement are discussed. As a last step in Chapter 3.5 an energy optimal control allocation algorithm distributes the change in the demanded movement to ROMO's over-actuated wheel robots. The complete control flow diagram with their interfaces to other modules is explained in Chapter 3.6 and a comprehensive simulation study together with a high-fidelity Modelica model of ROboMObil is assessed.

3.3 The Real-Time Capable Path Planning Module

The main task of the proposed algorithm is to calculate an energy optimal parametric path in a real-time capable way to be able to incorporate data from actual traffic situations (e.g. oncoming traffic) or changed road conditions.

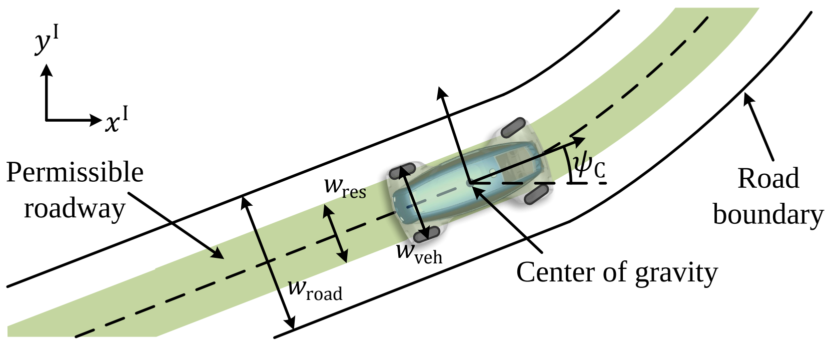


Figure 3.2: A simplified vehicle model – its width and orientation define the valid area of the roadway (the graphic is based on [Dan11]).

The resulting trajectory is then fed forward to a path following controller (Chapter 3.4) that calculates the motion demands for the energy optimal control allocation (Chapter 3.5). This in turn distributes the demand to the actuators of the over-actuated vehicle and a numerical reliable way is shown to formulate the energy optimal path planning optimization objective. Besides this, different types of optimization methods are evaluated for their computational effort and the influence of the path horizon assessed. Based on the optimized spatial path a velocity profile is calculated that considers the drag forces and the physical limitations of ROMO.

3.3.1 The Real-Time Capable Path Planning Approach

In this chapter an optimization problem formulation for a real-time capable path planning algorithm is introduced that combines the benefits of two approaches, which are a small scale non-linear optimization problem and linear inequality constraints which represent the driveway boundary accurately. In this approach a situation as shown in Figure 3.2 is assumed. A vehicle is travelling along a predefined road with defined roadway boundaries (here the navigation in a road network is not considered) and it is able to use the available lateral space w_{res} on the street to minimize an optimization criteria (e.g. the path curvature) over the considered road segment.

Preceding Works in Vehicle Path Planning

In the recent publication [Pad16] a comprehensive overview of different numerical methods for path planning strategies appropriate for semi-autonomous vehicles, like DLR's ROboMObil, is given. The authors group the methods shortly described as follows [Pad16]:

- Variational methods represent the path as a function parameterized vector and the (locally) optimized path is obtained by optimizing these parameters with a (non-)linear optimization method.
- Graph-search methods discretize the configuration space of a vehicle as a graph, in which vertices are the finite collection of vehicle configurations and the edges are transitions between the vertices. The minimal cost path in such a graph represents the searched optimal solution.
- Incremental search methods sample the configuration space and build incrementally a reachability path that maintains the discrete set of reachable configurations and feasible transitions between them.

This thesis focuses on the first type, the variational methods, since ROMO is a non-holonomic vehicle and therefore the parameterized path representation, e.g. with a spline, is a natural description formulation for a feasible path with a continuous curvature. An approach is the path representation with clothoids segments which are iteratively optimized to achieve a path with minimal curvature [Fun12] or the representation as Bézier curves (e.g. [Ma12]) which can also be transformed to a spline representation. Within this work it is decided to focus on spline representations of the vehicle path, since modification in Bézier curves control points do always influence the whole path and clothoids approaches are only analytically solvable under special assumptions [Wid09] or in an iteratively manner [Fun12]. A literature research for the here defined goals (normally shaped roadway with boundaries) led to the promising approaches in [Dan11] and [Bra08] which are examined in detail in the following. The first approach in [Dan11] is based on the idea that every road boundary as well as the vehicle path itself can be represented as set of two dimensional cubic splines with the spline parameter q (Figure 3.2)):

$$\boldsymbol{o}_i(q) = \begin{Bmatrix} o_{x,i}(q) = a_{x,i}(\theta_i)^3 + b_{x,i}(\theta_i)^2 + c_{x,i}(\theta_i) + d_{x,i} \\ o_{y,i}(q) = a_{y,i}(\theta_i)^3 + b_{y,i}(\theta_i)^2 + c_{y,i}(\theta_i) + d_{y,i} \end{Bmatrix}; \ \theta_i = q - q_i \tag{3.1}$$

Each spline consists of $n-1$ polynomial functions $\boldsymbol{o}_i(q)$ between the n interpolation points q_i $\{i \in \mathbb{N}: 1 \leq i \leq n-1\}$, this implies that for every spline segment $\{\theta_i \in \mathbb{R}: 0 \leq \ \theta_i \leq 1\}$ holds. The authors of [Dan11] derived a linearized quadratic program (QP) (see also eq. (3.42)) with inequality constraints which is based on a polynomial coefficient comparison of the boundaries and the optimized path. As the minimization criteria they defined a linearized criteria function $\tilde{E}$ of the nonlinear curvature κ along the path:

$$\tilde{E} = \sum_{i=1}^{n-1} \sum_{p \in \{x,y\}} (q_{i+1} - q_i)\left(b_{p,i}^2 + b_{p,i} b_{p,i+1} + b_{p,i+1}^2\right) \tag{3.2}$$

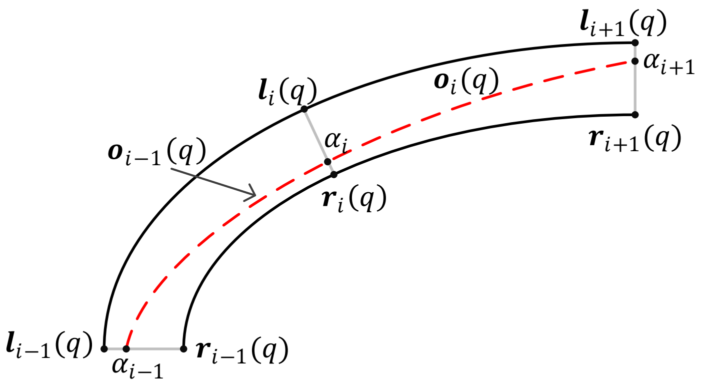
The QP enables good real-time capability due to the availability of powerful solvers e.g. *QL* [Sch05]. However, experimental implementations of this approach showed only unsatisfactory results. This can be constituted as follows: if the path changes a quadrant, it is necessary to redefine the inequality constraints, to be consistent with the left and right street boundary. Therefore, it is necessary to place an extra interpolation point at the transition of the quadrants, which leads to a poor weighting of the spline segments, since some of them may be very small. Due to the choice of the polynomial coefficients as optimization variables and their weak coupling between each other, the quadratic matrix of the QP gets sparse. The linearization of the curvature objective – which is only valid for very small distances – in combination with an increasing number of decision variables, depending on the route length and number of quadrant changes, makes it difficult for the QP solver to find a valid solution. An attempt to cope with this disutility by introducing weighting in the objective function did not lead to a numerical robust solution.

The second algorithm under evaluation [Bra08] simplifies the optimization approach, in the way that only the scalars $\alpha_i \in [0,1]$ at the junction points q_i of the polynomial sections can be tuned by the optimizer. These scalars α_i define the point on the connection line between the right $\boldsymbol{r}_i(q)$ and the left $\boldsymbol{l}_i(q)$ roadway boundaries at the point $q = q_i$ (see Figure 3.3); $\alpha_i = 0$ states that the path is located on the left and $\alpha_i = 1$ denotes it is located on the right side. In this way the optimization problem has a dense formulation and numerically reliable results are achievable. Nevertheless, with this approach the authors cannot guarantee that the calculated path will stay within the boundaries between two control points $q_i \to q_{i+1}$.

In the following subsection a new approach is proposed that combines the polynomial representation of [Dan11] with the simplification of the representation of [Bra08] and is moreover extended with an efficient boundary control mechanism.

The New Computational Efficient Approach

The aim of this approach is to give a small scale, well posed optimization problem and a reliable formulation of the boundaries inequality constraints, that guarantee the vehicle to stay within the roadway also between the interpolation points. The new approach uses polynomials for the path representation (similar to [Dan11]), but bounds the interpolation nodes on a line with a scaling parameter $\alpha_i \in [0,1]$ (inspired by [Bra08]) between the left $\boldsymbol{l}_i(q)$ and the right $\boldsymbol{r}_i(q)$ road boundaries (compare Figure 3.3). The benefit of this representation is that the number of optimization variables reduces from eight (compare eq. (3.1)) to only three per segment. The reduced set of variables for one spline segment $\boldsymbol{o}_i(q)$ are the polynomial coefficients $\boldsymbol{b}_i = \{b_{i,x}\ b_{i,y}\}$ and the scalar scaling factor α_i.



$\boldsymbol{l}_{i+1}(q)$
α_{i+1}
$\boldsymbol{l}_i(q)$
$\boldsymbol{o}_i(q)$
$\boldsymbol{o}_{i-1}(q)$
α_i
$\boldsymbol{r}_{i+1}(q)$
$\boldsymbol{r}_i(q)$
$\boldsymbol{l}_{i-1}(q)$
$\boldsymbol{r}_{i-1}(q)$
α_{i-1}

Figure 3.3: The new spline representation with reduced number of parameters

All other spline parameters can be expressed as functions of $\boldsymbol{b}_i$ and α_i. This representation can be derived from the constraint that a $\mathcal{C}^2$ continuous transition between the spline segments is guaranteed. This demand can be expressed in a set of equality constraints: the position and its first and second derivative between two spline segments have to be equal at each junction point. This leads to the following equality constraints

$$h_i^3 \boldsymbol{a}_i + h_i^2 \boldsymbol{b}_i + h_i \boldsymbol{c}_i + \boldsymbol{d}_i = \boldsymbol{d}_{i+1} \tag{3.3}$$

$$3h_i^2 \boldsymbol{a}_i + 2h_i \boldsymbol{b}_i + \boldsymbol{c}_i = \boldsymbol{c}_{i+1} \tag{3.4}$$

$$3h_i \boldsymbol{a}_i + \boldsymbol{b}_i = \boldsymbol{b}_{i+1} \tag{3.5}$$

wherein $i = 1 \dots n-2$ and $h_i = q_{i+1} - q_i$. Eq. (3.3) and eq. (3.5) contain the four unknown spline coefficients $\boldsymbol{a}_i, \boldsymbol{c}_i, \boldsymbol{d}_i$ and $\boldsymbol{d}_{i+1}$. To obtain a determined system of linear equations an additional eq. (3.6) is added to the set. It denotes the shifting of the interpolation points between the left $\boldsymbol{l}_i$ and the right $\boldsymbol{r}_i$ side of the street boundaries via the scalar scaling factor α:

$$\boldsymbol{d}_i = \boldsymbol{l}_i + (\boldsymbol{r}_i - \boldsymbol{l}_i) \cdot \alpha_i \quad \forall i = 1 \dots n \tag{3.6}$$

Solving the system of linear equations (3.3), (3.5) and (3.6) and substituting the results into eq. (3.1) yields the compact path representation with $3n$ optimization variables in $\boldsymbol{b}_i$ and α_i.

This compact formulation is used for the new equality constraint, which results from substituting the derived dependencies from $\boldsymbol{a}_i, \boldsymbol{c}_i$ and $\boldsymbol{d}_i$ on $\boldsymbol{b}_i$ and α_i into the remaining original constraint (3.4). Equation (3.7) evaluates the reduced equality constraint for $h_i = 1$ and $i = 2 \dots n-1$

$$\begin{aligned} &\boldsymbol{b}_{i-1} + 4\boldsymbol{b}_i + \boldsymbol{b}_{i+1} = \\ &3(\boldsymbol{l}_{i-1} + (\boldsymbol{r}_{i-1} - \boldsymbol{l}_{i-1})\alpha_{i-1}) - 6(\boldsymbol{l}_i + (\boldsymbol{r}_i - \boldsymbol{l}_i)\alpha_i) + \\ &+3(\boldsymbol{l}_{i+1} + (\boldsymbol{r}_{i+1} - \boldsymbol{l}_{i+1})\alpha_{i+1}) \end{aligned} \tag{3.7}$$

As a result, the number of equality constraints can be decreased, since they are partly contained in the new path representation. Beyond that, the inequality constraints at the nodes can be more simply described with the introduced optimization variable α_i:

$$0 \leq \alpha_i \leq 1; \quad i = 1 \dots n \tag{3.8}$$

In the next section it is shown that with this approach it can be ensured that the optimized path also lies in the constrained area of the roadway between two nodes. The simplest method to achieve this is to put the interpolated points closer together, with the major disadvantage to unnecessarily increasing the number of optimization variables. Therefore, the orientation independent inequality constraints are extended by additional control points, without adding optimization variables. The same principle as for α_i is used at control points between the spline sampling points. The i-th optimized path segment $\boldsymbol{o}_i(q)$ evaluated at k linearly spaced points $\rho_j = q_i + j \cdot \frac{1}{k+1}$ for $j = 1 \dots k$ must be located between the left boundary $\boldsymbol{l}_i(q)$ and the right boundary $\boldsymbol{r}_i(q)$ at the same points. This leads to the inequality formulation:

$$0 \leq \frac{\boldsymbol{r}_i(\rho_j) - \boldsymbol{l}_i(\rho_j)}{\left\|\boldsymbol{r}_i(\rho_j) - \boldsymbol{l}_i(\rho_j)\right\|^2}\left(\boldsymbol{o}_i(\rho_j) - \boldsymbol{l}_i(\rho_j)\right) \leq 1 \tag{3.9}$$

The introduced inequality control constraints of eq. (3.9) are visualized in Figure 3.4.

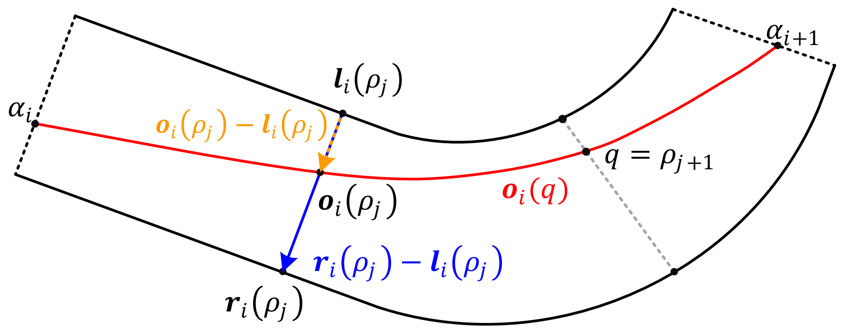

α_{i+1}
$\boldsymbol{l}_i(\rho_j)$
α_i
$\boldsymbol{o}_i(\rho_j) - \boldsymbol{l}_i(\rho_j)$
$q = \rho_{j+1}$
$\boldsymbol{o}_i(\rho_j)$
$\boldsymbol{o}_i(q)$
$\boldsymbol{r}_i(\rho_j) - \boldsymbol{l}_i(\rho_j)$
$\boldsymbol{r}_i(\rho_j)$

Figure 3.4: Sample spline segment with an additional control point at $q = \rho_j$

In words, the vector between the left boundary and the optimized path $\boldsymbol{o}_i(\rho_j) - \boldsymbol{l}_i(\rho_j)$ in direction of the vector between the left and right boundary $\boldsymbol{r}_i(\rho_j) - \boldsymbol{l}_i(\rho_j)$ must be longer than zero and shorter than $\boldsymbol{r}_i(\rho_j) - \boldsymbol{l}_i(\rho_j)$. The frequency of these constraints can be defined independently by the number of interpolated points. Here it is worth to mention that in exceptional cases despite valid inequality constraints the optimized path lies outside of the boundaries, but due to the curvature minimization it is very unlikely that a violating path is computed (e.g. loops between two control points may cause a large value in the curvature cost function eq. (3.10)).

It has been decided to formulate the optimization objective only in terms of the spline coefficients $\boldsymbol{b}_i$ and the control points α_i to be able to make benefit from the problem structure in the real-time capable implementation (see the later Chapter 3.3.2). Other optimization objective approaches that make use of the energy loss in the powertrain $E_{\mathrm{loss}}(v)$ (as it is discussed later in Chapter 3.5.3) or do incorporate the traveled time t or distance s result in a more complex optimization problem with $v(s)$ as a decision variable. E.g. incorporating the vehicle's lateral acceleration limit $\kappa(s) \cdot (\mathrm{d}s/\mathrm{d}t)^2 < a^{\mathrm{C}}_{y_{\max}}$ leads to nonlinear inequality constraints. Justified by the aim of being real-time capable the spatial path optimization is separated from the velocity profile generation (see details in Chapter 3.3.2).

The here considered optimization objective is now formulated as a linear problem in terms of the constraints (equalities = junctions; inequalities = path boundaries and extra control points) and a nonlinear cost function (curvature quadrat of the path [Hor83]). This objective has been already successfully used in publications for energy optimal path generation in [Dan10] and [Car10]. The assumption is that with a smaller curvature κ of the path, a turn can be passed with a higher velocity according to the relation $v^2 \leq a^{\mathrm{C}}_{y_{\max}}/\kappa$ and thus with less energy consumption caused by acceleration (braking and reaccelerating) of the vehicle. Furthermore, the necessary steering angles will stay small and therefore the tire losses due to cornering. The optimization problem with the spline coefficients $\boldsymbol{x} = \{b_{1,x}, b_{1,y}, \alpha_1, \dots, b_{n,x}, b_{n,y}, \alpha_n\}$ is given as follows:

$$\begin{aligned} \boldsymbol{x}^* = \ & \underset{\boldsymbol{x}}{\operatorname{argmin}} \int_0^{q_n} \kappa^2(\boldsymbol{x}, q)\mathrm{d}q \\ & \text{s.t. } \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \\ & \quad\ \ \boldsymbol{C}\boldsymbol{x} \leq \boldsymbol{d} \end{aligned} \tag{3.10}$$

Since eq. (3.7) is linear in the optimization variables $\boldsymbol{b}_i$ and α_i it can be symbolically transformed to the form $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$. The inequality constraints are expressed via the control points at the spline knots eq. (3.8), which lead to $2n$ equations in $\boldsymbol{C}\boldsymbol{x} \leq \boldsymbol{d}$. Additionally, the control points inequality constraints in eq. (3.9) are expanded with cubic splines. Their coefficients are represented in the dependency of the decision variables and finally transformed to the linear inequality equation form $\boldsymbol{C}\boldsymbol{x} \leq \boldsymbol{d}$. Moreover, all derivatives of the polynomials in the integral over the quadratic path curvature (cf. eq. (3.10)) can be calculated analytically

$$\kappa(\boldsymbol{x}, q) = \frac{o''_y(q)o'_x(q) - o''_x(q)o'_y(q)}{\left(o'^2_x(q) + o'^2_y(q)\right)^{\frac{3}{2}}} \tag{3.11}$$

with $o'_p(q) = o_p(q)\mathrm{d}/\mathrm{d}q$. With this combined approach the dimensions of the constraint matrices are reduced to $\boldsymbol{A} \in \mathbb{R}^{[2(n-2)+4]\times[3n]}$ and $\boldsymbol{C} \in \mathbb{R}^{[2k\cdot(n-1)+2n]\times[3n]}$ compared to $\boldsymbol{A} \in \mathbb{R}^{[6(n-2)]\times[8(n-1)]}$ and $\boldsymbol{C} \in \mathbb{R}^{[16(n-1)]\times[8(n-1)]}$ in [Dan11]. Summarizing all beneficial properties of this new approach the improvements, compared to [Dan11], [Bra08] are:

- The dimension of the optimization variables reduce from $8(n-1)$ to only $3n$,

- the nonlinear objective function can be analytically calculated from the spline itself,
- all constraints are linear and the inequality constraints are orientation independent,
- the number of equality constraints shrink to one third compared to [Dan11] and
- the number of inequality constraints could be reduced to one eighth (without extra control points).

By these achievements the computational performance gain can be estimated as follows: a quadratic program e.g. QL [Sch05] solver makes it necessary to calculate the Cholesky decomposition ($\mathrm{O}(n^3/3)$ floating operations [Gol13]) of the quadratic matrix $\boldsymbol{H}$ in eq. (3.42). When using the nonlinear gradient method [Ros07] it is necessary to calculate the pseudo invers for projecting the gradient on the linear constraints by means of a singular value decomposition which is extremely costly ($\mathrm{O}(12n^3)$, cf. [Gol13]). This gives a good rule of thumb that an increase of optimization variables or number of extra constraint equation raises the computational effort at least to the power of three of floating point operations.

3.3.2 Implementation and Testing

In this section an efficient solution of the optimization task is analyzed and experimental results in offline simulations with different path horizons are given.

Problem Structure Exploiting Optimization Methods and Testing

In order to calculate in real-time a solution for the constraint optimization problem of eq. (3.10), an optimization method needs to be chosen that can exploit the problem structure. For the assessment of the appropriate optimizer algorithms a 2 km long rural road segment of the Vires virtual landscape [Vir17b] is used. The same road segment (see Figure 3.29), but with a length of 3 km, is investigated later in Chapter 3.7.3 to evaluate the overall energy manager. The here considered track with 2 km length is formulated with the proposed new optimization approach according to eq. (3.10). It consists of 88 interpolation knots of the spline, i.e. 264 optimization variables have to be determined by the algorithm and the roadway boundaries have to be extracted from the Vires track description. The number of spline segments is chosen by a heuristic procedure which showed good results in a manifold of experiments. The courses of the road boundaries as well as the initial guess (the road middle) lane are checked in before to not have any oscillations or loops within them. With this setup a case study with different appropriate optimization algorithms has been performed, listed in Table 3.1.

Table 3.1: Optimizer performance comparison for constrained path curvature minimization

Algorithm	**Time**	**Iterations**	**Cost Function** $\boldsymbol{c_i}$	**Relative Increase** $(\boldsymbol{c_i} - \boldsymbol{c}_{\mathbf{min}})/\boldsymbol{c}_{\mathbf{min}}$
QP	0.09 s	1	$3.490 \cdot 10^{-2}$	38.7 %
NG	2.67 s	73	$2.549 \cdot 10^{-2}$	1.31 %
fmincon	7.13 s	201	$2.568 \cdot 10^{-2}$	2.07 %
SQP	38.6 s	481	$2.516 \cdot 10^{-2}$	Reference

The table shows the performance index of a (linearized) quadratic program (QP), a nonlinear gradient descent (NG) search, `fmincon` (the built-in solver in MATLAB) and a sequential quadratic program (SQP) solver. The solution of the SQP showed the best result with respect to the minimization of the cost function. But it also is the most computational demanding and therefore acts as reference for the other solvers. The best tradeoff between calculation time and accuracy showed the nonlinear gradient descent method (NG), in a similar implementation as proposed in [Ros60], and is hence selected as method of choice.
The reason for its good performance in this optimization task (eq. (3.10)) can be explained as follows: first, the gradient function can be efficiently calculated when approximating the curvature integral with a Riemann sum of the quadrat of the analytical curvature function (eq. (3.11):

$$E_i = \int_{q_i}^{q_{i+1}} \kappa^2(\boldsymbol{x}_i, q)\mathrm{d}q \approx \hat{E}_i = \sum_j \kappa_i^2\big(\boldsymbol{x}_i, q_j\big) \cdot \Delta q_j \tag{3.12}$$

It is assumed that the constant interval size Δq_j is chosen to be small. This approximation is necessary since no analytic integral solution of the quadratic curvature term eq. (3.12) can be found. The NG minimization objective writes therefore as follows:

$$\begin{gathered} \boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}}\, f(\boldsymbol{x}) = \sum_{i=1}^{n-1} \hat{E}_i(\boldsymbol{x}_i) \\ \text{s.t. } \boldsymbol{Ax} = \boldsymbol{b} \\ \boldsymbol{Cx} \leq \boldsymbol{d} \end{gathered} \tag{3.13}$$

Through this method, the gradient computation can be divided into sub problems due to the piecewise polynomial representation of the spline functions. Second, the linear equality and inequality constraints enable projection methods to determine a feasible descent search direction. The negative gradient $-\boldsymbol{\nabla} f(\boldsymbol{x})$, i.e. the search direction, is projected onto the equality constraints $\boldsymbol{A}$ by computing a Moore-Penrose pseudoinverse $\boldsymbol{P}^\dagger$ [Gol13]. This results in the projected descent direction $\boldsymbol{r}_{\text{pro}}$ as described in [Ros60] and shown in eq. (3.14)

$$\boldsymbol{r}_{\text{pro}} = -\boldsymbol{P}^\dagger(\mathbf{A}) \cdot \boldsymbol{\nabla} f(\boldsymbol{x}) \tag{3.14}$$

The resulting projected descent direction $\boldsymbol{r}_{\text{pro}}$ lies in the null space of the linear equality constraints $\boldsymbol{Ax} = \boldsymbol{b}$, which means it is ensured that no component of the descent direction point out of the admissible space. The pseudo inverse $\boldsymbol{P}^\dagger(\boldsymbol{A})$ calculation has to be performed only once per optimization run, since the equality constraints do not change during the descent search, and can be kept in storage for solving the linear equation system $\boldsymbol{r} = \boldsymbol{P}^\dagger(\boldsymbol{A}) \cdot -\boldsymbol{\nabla} f(\boldsymbol{x})$ during the iterations. However, the calculation of the pseudo inverse by means of a singular value decomposition is very computational demanding ($\mathrm{O}(12n^3)$, cf. [Gol13]) it is proposed to transform eq. (3.14) into an equivalent least squares problem:

$$\boldsymbol{r}_{\text{pro}} = \underset{\boldsymbol{x}}{\operatorname{argmin}}\, \big\|\boldsymbol{Ax} - \big(-\boldsymbol{\nabla} f(\boldsymbol{x})\big)\big\|^2 \tag{3.15}$$

This minimization task can be solved by the LAPACK function DGELSX [And99] which uses a more efficient QR decomposition with $O(2n^3/3)$ floating operations [Gol13]. Unfortunately, the QR decomposition would be necessary in every iteration step of the NG descent search. To overcome this issue the solution of $\boldsymbol{r}_{\text{proj}}$ can be calculated in the same way as it is explained in Chapter A.3.1 (cf. eq. (A.7) to eq. (A.11)):

$$\boldsymbol{r}_{\text{pro}} = -\underbrace{\boldsymbol{P}\boldsymbol{Z}^T\boldsymbol{U}^{-1}\boldsymbol{Q}_1^T}_{\text{constant}}\nabla f(\boldsymbol{x}) \tag{3.16}$$

Through this measure it is possible to calculate the matrices $\boldsymbol{P}, \boldsymbol{Z}, \boldsymbol{U}^{-1}, \boldsymbol{Q}_1$ only once per optimization call and keep them in memory for the subsequent iterations of the descent projection on the equality constraints. For the inequality constraints a more complex Gram-Schmidt procedure is chosen, which projects the search direction into the null space of the active inequality constraints. This avoids computational expensive matrix inversions if the set of active inequality constraints changes, which is an advantage especially with large scale problems that can occur in the case of paths with long distances (for algorithm details please see Chapter 4.4.1). In Figure 3.5 the outcome of the curvature minimization by optimization with the NG algorithm compared to the non-optimized is depicted for the above mentioned test track segment. The quadratic characteristic of the cost criterion causes a higher weighting on high curvatures. Thus, the peak curvature of sharp turns is reduced more than the curvature of wide turns.

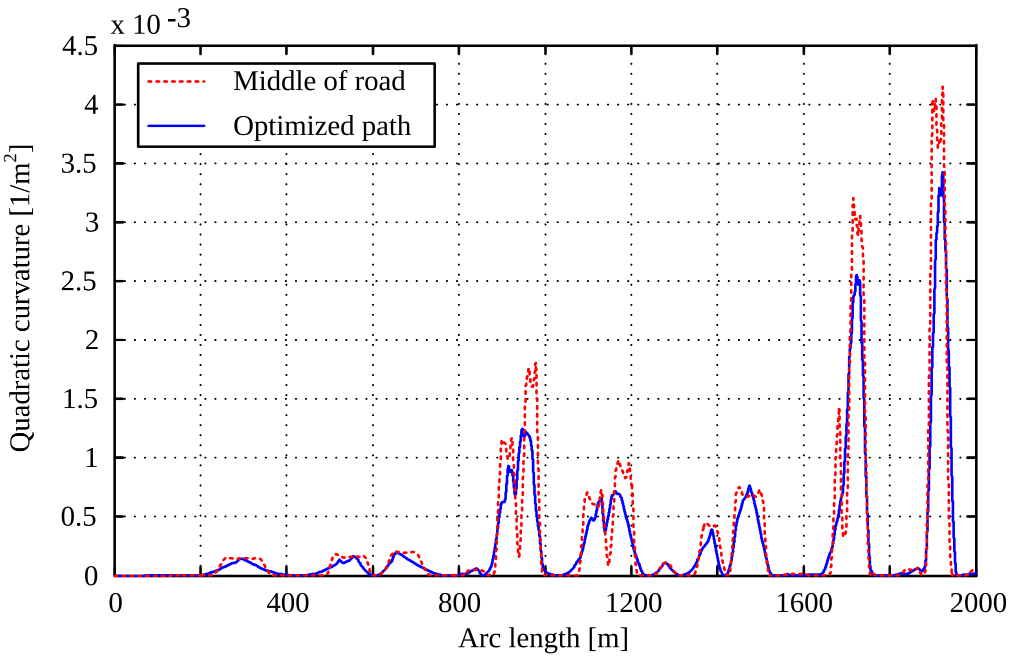


Figure 3.5: Objective function comparison of the track middle lane and the optimized path

Additionally, smoother transitions between the turns are realized by the optimization, which gives a gentler handling to the vehicle's passengers.

Figure 3.6 shows a detail cut of the optimized path of the test track. It can be seen that the consideration of the road boundaries in the inequality constraints is effective for path planning and the smooth transition is similar to the one race car drivers choose to keep the speed while cornering [Bra08].

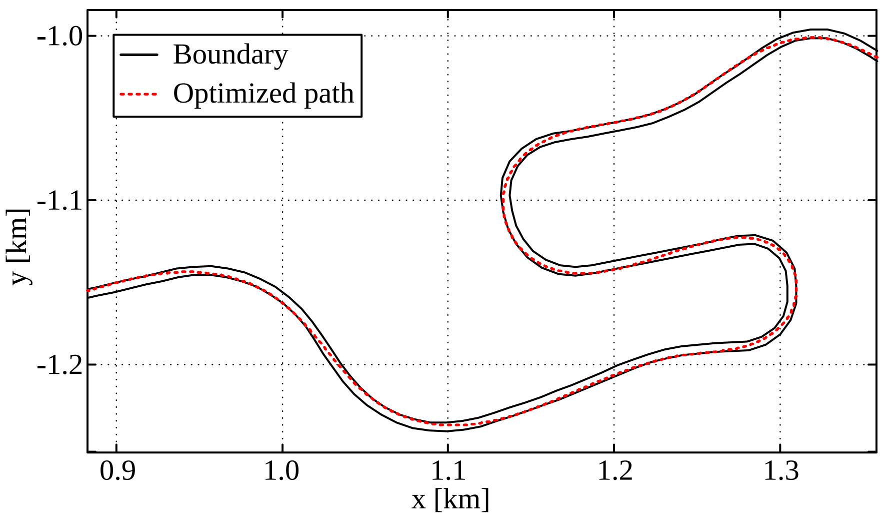


Figure 3.6: Detail view of optimized path in a road section

So far, the complete length of a track segement has been considered for the spatial path optimization. However, for a potential application in a real-time environment, like ROboMObil's central control (cf. Figure 2.16), the planning horizon must be limited to stay within the time constraints due limited calculation power. In Table 3.2 the outcome of a quantitative study of the influence of the optimization horizon length Δs, by hands of the above mentioned test segment, is given.

Table 3.2: Cost function (eq. 3.10) assessment in dependency of the optimization horizon Δs

Algorithm	**Cost Function**	**Cost Reduction**
Track middle lane	$626.8 \cdot 10^{-3}$	Reference
$\Delta s = 100$ m	$554.4 \cdot 10^{-3}$	11.5 %
$\Delta s = 200$ m	$510.9 \cdot 10^{-3}$	18.5 %
$\Delta s = 2000$ m (global opt.)	$509.2 \cdot 10^{-3}$	18.8 %

The comparison shows that the global optimization reduces the objective cost eq. (3.10) by almost 19 % in comparison to the curvature of the middle lane of the road. On the one hand, the cost function reduction is limited, since a road width of 7 m gives a limited space for finding smoother paths. On the other hand, Table 3.2 depicts the dependency of the cost function on the chosen optimization horizon size. The longer the optimization horizon is, the more the cost function can be reduced. With a length of $\Delta s = 200$ m no major difference in the cost function compared to the global optimization is realized.

The Velocity Profile Generation

Besides this spatial optimization it is necessary to generate the temporal movement along the optimized path. In this work a velocity profile generation is chosen according to [Bue11b] and is subsequently executed to the path optimization. In order to have a real-time capable path and velocity profile generation, both tasks are separated: First, a desired velocity profile is defined considering the maximum vehicle velocity (constrained by technical limits, e.g. the maximum wheel robot rotational speed – compare App. Table C.3) and the road speed limits. In the second step this desired profile is modified to satisfy a lateral acceleration limit with the given path curvature regarding the following equation: $v^2 \leq a^{\mathrm{C}}_{y_{\max}}/\kappa$. Note that the velocity profile itself is not energy optimal since it is not part of the minimization (compare Chapter 3.3.1); it only guarantees a feasible velocity demand in dependency of the physical and technical limits.

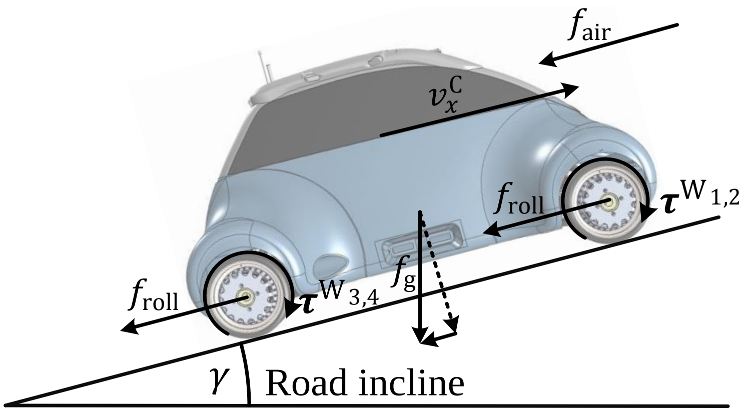


Figure 3.7: Acting forces on ROboMObil in longitudinal direction

This lateral acceleration limit $a^{\mathrm{C}}_{y_{\max}}$ is mainly based on comfort requirements. The last step is a forward and backward filtering of the velocity profile to satisfy the longitudinal acceleration limits. These acceleration constraints are due to the forces acting on the vehicle, which are the available motor torques $\boldsymbol{\tau}^{\mathrm{W}}$ and the counteracting driving resistances: the air resistance f_{air}, downhill-slope force $f_{\mathrm{g}}(\gamma)$, and the tire rolling resistance f_{roll} as depicted in Figure 3.7. Details of the algorithm and the mathematical velocity profile description can be found in [Bue11b].

The Parametric Path Description

The interface to the underlying path following control (explicated in Chapter 3.4) is a time independent motion demand $\boldsymbol{\lambda}(s) \in \mathbb{R}^5$ presentation with respect to the path arc length s. It consists of the following five quantities: the absolute position $\boldsymbol{p}^{\mathrm{I}}_{\mathrm{P}} = \{x^{\mathrm{I}}_{\mathrm{P}}(s), y^{\mathrm{I}}_{\mathrm{P}}(s)\}$ of the reference path, the corresponding path orientation $\psi^{\mathrm{I}}_{\mathrm{P}}(s)$, its curvature $\kappa_{\mathrm{P}}(s)$ and the generated longitudinal velocity $v^{\mathrm{P}}_x(s)$.

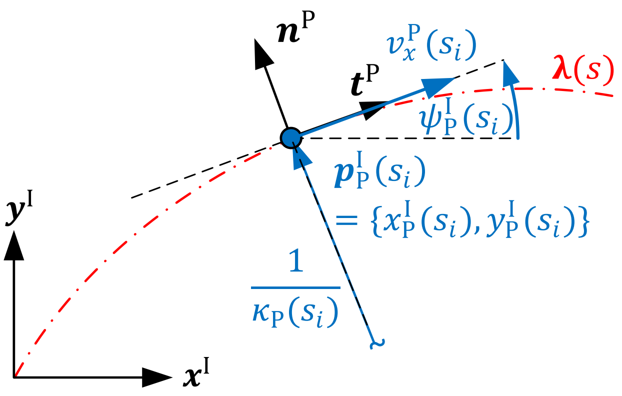

$\boldsymbol{n}^{\mathrm{P}}$
$v^{\mathrm{P}}_x(s_i)$
$\boldsymbol{\lambda}(s)$
$\boldsymbol{t}^{\mathrm{P}}$
$\psi^{\mathrm{I}}_{\mathrm{P}}(s_i)$
$\boldsymbol{p}^{\mathrm{I}}_{\mathrm{P}}(s_i)$
$= \{x^{\mathrm{I}}_{\mathrm{P}}(s_i), y^{\mathrm{I}}_{\mathrm{P}}(s_i)\}$
$\boldsymbol{y}^{\mathrm{I}}$
$\frac{1}{\kappa_{\mathrm{P}}(s_i)}$
$\boldsymbol{x}^{\mathrm{I}}$

Figure 3.8: Graphical interpretation of the parametric path $\boldsymbol{\lambda}(s)$ at point s_i

In Figure 3.8 all quantities are spanned at a certain point s_i along the parametric path $\boldsymbol{\lambda}(s)$. The desired vehicle longitudinal velocity $v_x^{\mathrm{P}}(s_i)$ is parallel to the tangent vector $\boldsymbol{t}^{\mathrm{P}} = \boldsymbol{x}^{\mathrm{P}}$ at $\boldsymbol{p}_{\mathrm{P}}^{\mathrm{I}}(s_i)$ and the reciprocal curvature $1/\kappa_{\mathrm{P}}(s_i)$.

3.4 The Path Following Control

In this chapter a path following controller is developed that generates the vehicle motion demand in the horizontal plane – details on the energy optimal distribution of the motion demand between ROMO's actuators are given in Chapter 3.5. In comparison to a classic trajectory tracking controller, which tracks the scheduled time depended velocity commands, the path following controller compensates errors between the actual vehicle states and the corresponding demands on the parametric path $\boldsymbol{\lambda}(s)$.

3.4.1 The Time Independent Path Interpolation

For the energy optimal control allocation of ROMO's wheel robots (compare Chapter 3.5.3) the values of $\boldsymbol{\lambda}(s)$ (see Chapter 3.3.1 – Figure 3.8) cannot be used directly since the allocator inputs are, as described later, defined by $\Delta\dot{v}_x, \Delta\dot{v}_y, \Delta\dot{\psi}$. In order to generate these magnitudes by a controller an online derivation of the set-points and their derivatives is developed that does not use the time explicitly. With this time independent path interpolation (TIPI) [Rit15] and the tracking controller (Chapter 3.4.2) a dynamical state dependent vehicle motion demand trajectory can be generated. Ideally, the actual path positon and the position of the car coincide $\boldsymbol{p}_{\mathrm{P}}(s^*) = \boldsymbol{p}_{\mathrm{C}}$. To approximate this condition it is necessary to minimize the displacement $\boldsymbol{e}(s)$ between vehicle and reference path as depicted in Figure 3.9:

$$s^* = \underset{s}{\operatorname{argmin}} \left\| \underbrace{\boldsymbol{p}_{\mathrm{P}}(s) - \boldsymbol{p}_{\mathrm{C}}}_{\boldsymbol{e}(s)} \right\|_{\mathbf{2}} \tag{3.17}$$

The geometrical interpretation of this minimization objective is that $\boldsymbol{p}_{\mathrm{P}}(s^*)$ can be determined by projecting $\boldsymbol{p}_{\mathrm{C}}$ orthogonally on the path $\boldsymbol{p}_{\mathrm{P}}(s)$. For this issue it is proofed in [Mor08] that $\boldsymbol{p}_{\mathrm{P}}(s^*)$ exists and is unique if $\|\boldsymbol{e}(s^*)\|_2$ is smaller than the lower bound of the curve radius. For the TIPI this condition implies that the inverse of the maximal curvature of the demanded vehicle path defines the maximum lateral displacement for which s^* exists: $e_y^{\mathrm{P}}(s^*) \leq 1/\kappa_{\mathrm{P}}(s^*)$.

The solution of the optimization problem eq. (3.17) is illustrated in Figure 3.9. Investigating this figure it can be seen that $\|\boldsymbol{e}(s)\|_2$ is minimal if the vectors $\boldsymbol{e}(s)$ and $\boldsymbol{x}^{\mathrm{P}}(s)$ are orthogonal to each other. This condition implies that the tangential displacement component $e_x^{\mathrm{P}}(s)$ is zero. Hence, the actual parameter value s^* can be determined by the following scalar root finding problem:

$$e_x^{\mathrm{P}}(s^*) \overset{!}{=} 0 \tag{3.18}$$

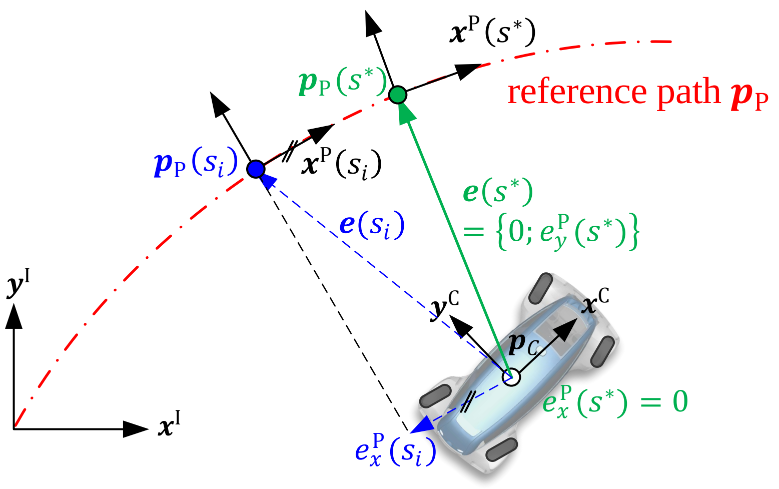


Figure 3.9: Graphical representation of the dynamic root finding to determine s^*

The derivation of this root finding by means of an iteration free error minimization problem is given in detail in [Rit15]. It is designed – as depicted in Figure 3.10 – as a feedback control loop which minimizes the longitudinal displacement $\dot{s} = -K_P \cdot e_x^P$ combined with an estimated curve parameter rate $\hat{\dot{s}}$ in the feed forward path of the control loop.

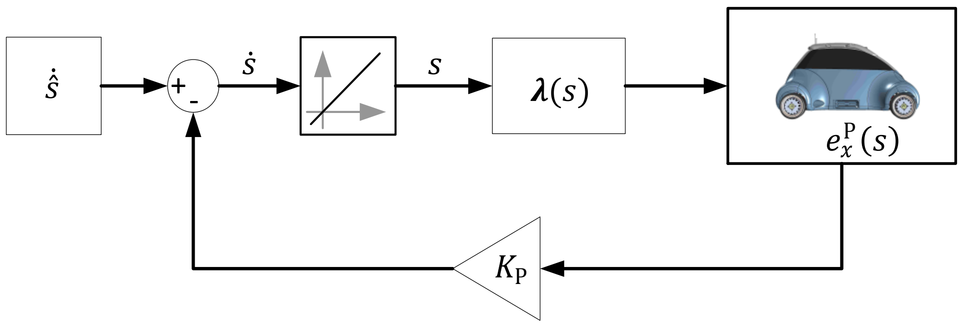


Figure 3.10: Block diagram of the dynamic root finding control loop

3.4.2 The Path Tracking Controller

In this subsection a path tracking control law is developed that calculates references inputs for an underlying control allocation algorithm (cf. Chapter 3.5). Its task is to compensate the tracking error between the actual vehicle states $p^I_{C_{act}}, v^C_{act}$ and the demands of the parametric curve $\lambda(s)$ at the actual path arc length s_k, determined by the method introduced in Chapter 3.4.2.

Preceding Works in Path Tracking Control

The authors in [Pad16] group current tracking controller methods (for self-driving urban vehicles) into path stabilization and trajectory stabilization approaches. This is done in dependency of the reference provided by motion planner. In the previous chapter a parametric path description has been developed and therefore only previous work is addressed, which copes with path stabilization. The authors in [Pad16] group the methods shortly described as follows:

- Path stabilization for kinematic models without consideration of the wheel slip. An example for this is the pure pursuit controller. It determines a semi-circle cutting the vehicle's current position and a point on the reference path. This lays ahead of the vehicle with a predefined look ahead distance L. In this way an additional yaw angle can be calculated to control the vehicle's steering. Other methods are front or rear wheel position based feedback laws which minimize the distance error along the normal vector from the reference path to the front respectively the rear wheel of the vehicle.
- Model predictive control approaches solve an optimization problem to track the path under the restriction of constraints (e.g. limited wheel slip) and incorporate the vehicle dynamics.
- Linear parameter varying controllers take system varying states, as the vehicle speed, into account. This improves the control performance and stability in comparison to the approaches in the first bullet. These may be inaccurate in case of large position errors, caused by the linearization error.

In contrast to these methods, here a path stabilization controller which compensates the spatial errors between ROMO and the reference track is designed. Since ROboMObil has an over actuated drivetrain configuration, this controller calculates the set-points for an underlying nonlinear control allocator which distributes the demands to the steering and traction drives of ROMO.

Parametric Path Stabilization Based on Three Separated PD Controllers

To track the motion request of the path planning and the corresponding velocity three separate controllers are incorporated by use of the time independent path interpolation and the actual vehicle states. The necessary information for this controller setup are the actual vehicle position in the inertial coordinate system $\boldsymbol{p}^{\mathrm{I}}_{\mathrm{C_{act}}} = \{x^{\mathrm{I}}_{\mathrm{C_{act}}}, y^{\mathrm{I}}_{\mathrm{C_{act}}}, \psi^{\mathrm{I}}_{\mathrm{C_{act}}}\}$, as well as the longitudinal and lateral vehicle velocity $\boldsymbol{v}^{\mathrm{C}}_{\mathrm{act}} = \{v^{\mathrm{C}}_{x_{\mathrm{act}}}, v^{\mathrm{C}}_{y_{\mathrm{act}}}\}$ with respect to the vehicle frame itself. The following algorithm is described in a discrete-time representation where s_k denotes the arc length value at time-discrete instance k.

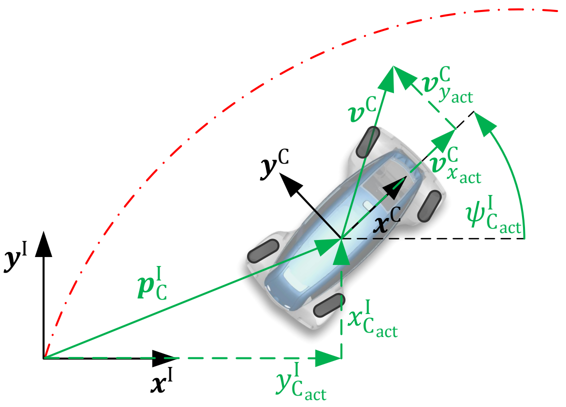

$\boldsymbol{v}^{\mathrm{C}}$
$v^{\mathrm{C}}_{y_{\mathrm{act}}}$
$v^{\mathrm{C}}_{x_{\mathrm{act}}}$
$\boldsymbol{y}^{\mathrm{C}}$
$\boldsymbol{x}^{\mathrm{C}}$
$\psi^{\mathrm{I}}_{\mathrm{C_{act}}}$
$\boldsymbol{y}^{\mathrm{I}}$
$\boldsymbol{p}^{\mathrm{I}}_{\mathrm{C}}$
$x^{\mathrm{I}}_{\mathrm{C_{act}}}$
$\boldsymbol{x}^{\mathrm{I}}$
$y^{\mathrm{I}}_{\mathrm{C_{act}}}$

Figure 3.11: Vehicle state quantities for the path tracking controller

In the first step it is necessary to evaluate the demanded control values $\boldsymbol{\lambda}$ in dependency of the current path parameter s. It is calculated in the path interpolation module, in dependency of the actual vehicle states $\boldsymbol{p}^{\mathrm{I}}_{\mathrm{C_{act}}}$, $\psi^{\mathrm{I}}_{\mathrm{C_{act}}}$ and $\boldsymbol{v}^{C}_{act}$ with the discrete algorithm explained in the following. The values of $\boldsymbol{\lambda}(s)$ are stored in a varying buffer matrix that can be updated, consistent and state dependent, by the path planning module (see Chapter 3.3) if it is used in a real-time application. Values between the discrete sampling points s_k are gathered by interpolation.

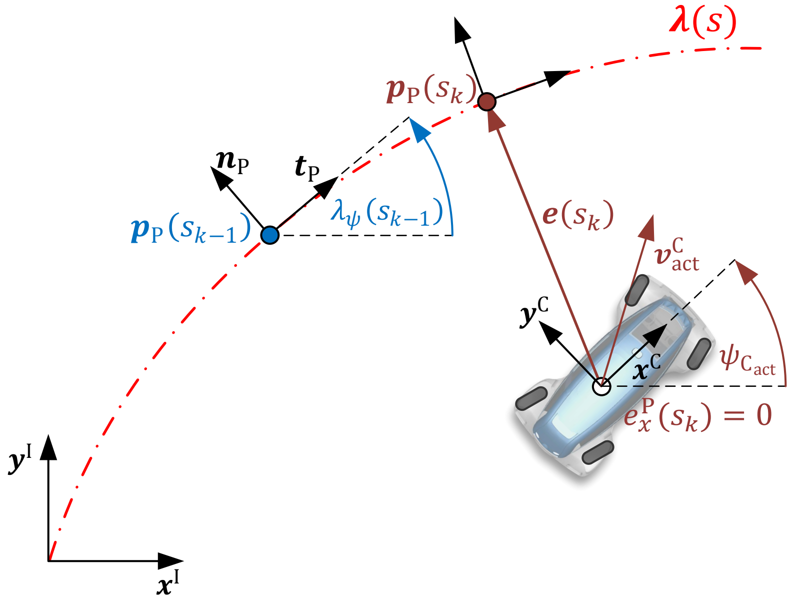

$\boldsymbol{\lambda}(s)$
$\boldsymbol{p}_{\mathrm{P}}(s_k)$
$\boldsymbol{n}_{\mathrm{P}}$
$\boldsymbol{t}_{\mathrm{P}}$
$\boldsymbol{p}_{\mathrm{P}}(s_{k-1})$
$\lambda_\psi(s_{k-1})$
$\boldsymbol{e}(s_k)$
$\boldsymbol{v}^{\mathrm{C}}_{\mathrm{act}}$
$\boldsymbol{y}^{\mathrm{C}}$
$\boldsymbol{x}^{\mathrm{C}}$
$\psi_{\mathrm{C_{act}}}$
$e^{\mathrm{P}}_x(s_k) = 0$
$\boldsymbol{y}^{\mathrm{I}}$
$\boldsymbol{x}^{\mathrm{I}}$

Figure 3.12: Schematic representation of discrete-time path interpolation

For the following calculation steps it is useful to comprehend Figure 3.12. To calculate the error vector $\boldsymbol{e} = \{e_x, e_y\}$ the vehicle velocity state $\boldsymbol{v}^{\mathrm{C}}_{\mathrm{act}}$ must be transformed with a rotation matrix $\boldsymbol{R}$ into the inertial coordinate system.

$$\boldsymbol{v}_{\mathrm{C}} = \left[\underbrace{\begin{matrix} \cos(\psi_{\mathrm{C_{act}}}) & -\sin(\psi_{\mathrm{C_{act}}}) \\ \sin(\psi_{\mathrm{C_{act}}}) & \cos(\psi_{\mathrm{C_{act}}}) \end{matrix}}_{\boldsymbol{R}}\right] \cdot \boldsymbol{v}^{\mathrm{C}}_{\mathrm{act}} \tag{3.19}$$

In the next step the longitudinal and lateral displacement error $\boldsymbol{e}$ between the path and the vehicle is decomposed into a tangential and normal part along the path:

$$\begin{aligned} \boldsymbol{t}_{\mathrm{P}} &= \left\{\cos\left(\lambda_\psi(s_{k-1})\right), \sin\left(\lambda_\psi(s_{k-1})\right)\right\} \\ \boldsymbol{n}_{\mathrm{P}} &= \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \cdot \boldsymbol{t}_{\mathrm{P}} \\ \boldsymbol{e} &= \left\{x_{\mathrm{C_{act}}} - \lambda_x(s_{k-1}),\ y_{\mathrm{C_{act}}} - \lambda_y(s_{k-1})\right\} \cdot [\boldsymbol{t}_P \quad \boldsymbol{n}_P] \end{aligned} \tag{3.20}$$

Finally in combination with the estimated arc length rate $\hat{\dot{s}}$ [Rit15] the control law for the iteration free root finding of $\dot{s}$ (compare Figure 3.10) can be written as:

$$
\begin{aligned}
\dot{s} &= -K_{\mathrm{P}_x} \cdot \boldsymbol{e}_{x}^{\mathrm{P}}. + \dot{\hat{s}} \\
\Rightarrow \dot{s} &= K_{\mathrm{P}_x} \cdot e_x + \frac{\boldsymbol{v}_{\mathrm{act}}^{\mathrm{C}} \cdot \boldsymbol{t}_{\Delta\psi}}{1 - e_y \cdot \lambda_\kappa(s_{k-1})} \\
\boldsymbol{t}_{\Delta\psi} &= \{\cos(\Delta\psi), \sin(\Delta\psi)\} \\
\Delta\psi &= \lambda_\psi(s_{k-1}) - \psi_{\mathrm{C_{act}}}
\end{aligned}
\tag{3.21}
$$

and the actual arc length parameter s_k is determined by a discrete-time integration:

$$
s_{k|k-1} = s_{k-1} + \int_{t_{k-1}}^{t_k} \dot{s}(s_{k-1}, \dots)\, \mathrm{d}t \tag{3.22}
$$

Since the value for $\boldsymbol{\lambda}(s_k)$ is now available, in a second step a longitudinal PD-controller, which minimizes the error between the actual vehicle velocity $v_{x_{\mathrm{act}}}^{\mathrm{C}}$ and the reference vehicle velocity $v_{x_{\mathrm{ref}}}^{\mathrm{P}}$, can be formulated as follows:

$$
\begin{gathered}
\Delta v_x^{\mathrm{C}} = K_{\mathrm{P}_x} \cdot \left(\lambda_{v_x}(s_k) - v_{x_{\mathrm{act}}}^{\mathrm{C}}\right) + K_{\mathrm{D}_x} \cdot \frac{\mathrm{d}\left(\lambda_{v_x}(s_k) - v_{x_{\mathrm{act}}}^{\mathrm{C}}\right)}{\mathrm{d}t} \\
\text{with } \left[K_{\mathrm{P}_x}\right] = 1, \left[K_{\mathrm{D}_x}\right] = {}^{1}\!/_{s}
\end{gathered}
\tag{3.23}
$$

In the last step the two lateral controllers, which minimize the displacement e_y and the orientation offset e_ψ of the vehicle, can be designed as two separated PD-controllers in a similar way:

$$
\begin{aligned}
\boldsymbol{n}_{\mathrm{C}} &= \left\{-\sin\left(\psi_{\mathrm{C_{act}}}\right), \cos\left(\psi_{\mathrm{C_{act}}}\right)\right\} \\
e_y &= \left(\lambda_x(s_k) - x_{\mathrm{C_{act}}}, \lambda_y(s_k) - y_{\mathrm{C_{act}}}\right) \cdot \boldsymbol{n}_{\mathrm{C}} \\
e_\psi &= \lambda_\psi(s_k) - \psi_{\mathrm{C_{act}}} \\
\Delta v_y^{\mathrm{C}} &= K_{\mathrm{P}_y} \cdot e_y + K_{\mathrm{D}_y} \cdot \frac{\mathrm{d}e_y}{\mathrm{d}t} \\
\Delta\dot{\psi}^{C} &= K_{\mathrm{P}_{\dot{\psi}}} \cdot e_\psi + K_{\mathrm{D}_{\dot{\psi}}} \cdot \frac{\mathrm{d}e_\psi}{\mathrm{d}t}; \\
\left[K_{\mathrm{P}_y}\right] &= \left[K_{\mathrm{P}_{\dot{\psi}}}\right] = {}^{1}\!/_{\mathrm{s}^2}\ ;\ \left[K_{\mathrm{D}_y}\right] = \left[K_{\mathrm{D}_{\dot{\psi}}}\right] = {}^{1}\!/_{\mathrm{s}}
\end{aligned}
\tag{3.24}
$$

Assessment of the Controller Tracking Stability

The here proposed controller is a linear controller applied to a nonlinear system. A simulative case study in Chapter 3.7 showed a reliable and robust control behavior of this approach. This can be reasoned since the nonlinearity of the vehicle is considered in the underlying nonlinear control allocator (cf. Chapter 3.5). In Figure 3.13 the error tracking is shown for the three PD path tracking controllers used for the experiment in Chapter 3.7.3. The vehicle starts in the middle of the road, whereas the planned path is located on the right roadway (cf. error $e_y = 2$ m at $t = 0$ s in Figure 3.13). The initial vehicle velocity is close to the reference velocity of the parametric path. The tracking error of the lateral displacement e_y is fast and without an overshoot compensated, as well as the yaw error e_ψ is robustly compensated to zero during the half lane-change

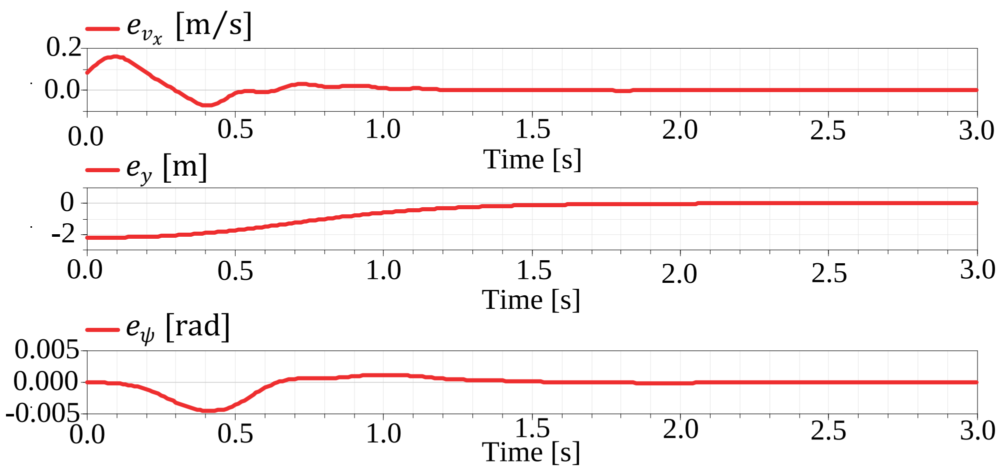


Figure 3.13: Robust error tracking of the vehicle's initial displacement

Nevertheless, this controller will get unstable, when the tracking error gets too large. E.g. the road friction changes strongly and the tracking error gains strongly. By this reason an improved path tracking controller for real world tests has been developed in [Rit15]. It is based on nonlinear I/O linearization and successfully tested in an interactive driving experiment with ROMO in [Rit16b].

3.5 The Energy Optimal Actuator Control

In this chapter a new control allocation based approach for energy optimal motion demand distribution for the over-actuated electric robotic vehicle ROMO is proposed. The focus of the control strategy lies on the model based minimization of the actuator losses and power consumption for driving along a trajectory to optimize the overall efficiency. The approach is based on a real-time capable nonlinear control allocation (CA) algorithm, using quadratic programming, implemented in the object oriented modeling language Modelica. Different optimization objectives are analyzed and the performance is presented in simulation results.

In vehicle dynamics control theory one can find a couple of approaches focusing on vehicle control by force allocation methods [And07], [Jon09], [Kno08]. The main emphasis of their research was on the centralized controlled vehicle stabilization. The objective here is to drive ROboMObil along the predetermined path $\boldsymbol{\lambda}(s)$ (compare Chapter 3.4) with the lowest possible power demand and losses so that the range is maximized and simultaneously sufficient vehicle dynamics are retained. So the focus in this section is to reduce energy consumption in the drivetrain actuators; the four wheel hub motors, the four in-wheel electric steerings and the two electro hydraulic brakes. In contrast to [Lai07], the energy management presented in this work is a preceding mechanism and not part of the control allocation optimization itself.

3.5.1 The Robotic Approach

Since ROMO's development is inspired from robotic technology its control strategy should be closely connected to it. The task for the control strategy of an industrial 6-degree of freedom

(DOF) robot is to calculate the joint angles for a given 6-DOF end effector trajectory (the robot manipulator) of a high level planner. One algorithm for doing this is the selective damped least squares algorithm proposed in [Bus05]. It calculates the null space solution along a given trajectory, trying to prevent kinematic singularities. Based on this analogy an energy optimal control for RObоMObil is developed in the following.

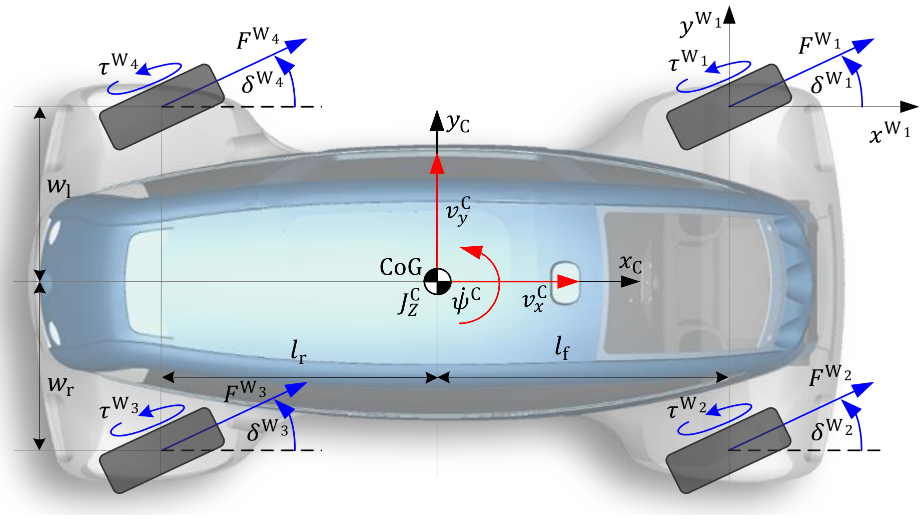


Figure 3.14: Mapping of planar movement requests to the over-actuated ROMO

3.5.2 The RObоMObil Approach

The proposed control allocation method considers the planar movement of RObоMObil, depicted in Figure 3.14. Compared with the robotic approach, the end effector can be seen as a 3-DOF flat movement of ROMO at the center of gravity (CoG). These degrees of freedom are two translational velocities and one rotational velocity:

$$\boldsymbol{v}^{\mathrm{C}} = \{v_x^{\mathrm{C}}, v_y^{\mathrm{C}}, \dot{\psi}^{\mathrm{C}}\} \tag{3.25}$$

In contrast to an industrial robot, ROMO has 10 actuators in total in its joints. In the following only eight actuators, four steering angles and four drive torques are considered. It is not distinguished between the two electro mechanical brakes (one per axle) and the recuperative feature of the electric drives. It is assumed that there is an underlying control mechanism that maximizes the recuperation even in wheel slip situation controlled by an algorithm as proposed in [Sat16]. In addition, this investigation focuses on vehicle operation modes in which the maximum recuperation torque of the traction motors is sufficient to achieve the motion demands. The actuation control vector with respect to the wheels is:

$$\boldsymbol{u}^{\mathbf{W}} = \{\delta^{\mathrm{W}_1}, \delta^{\mathrm{W}_2}, \delta^{\mathrm{W}_3}, \delta^{\mathrm{W}_4}, \tau^{\mathrm{W}_1}, \tau^{\mathrm{W}_2}, \tau^{\mathrm{W}_3}, \tau^{\mathrm{W}_4}\} \tag{3.26}$$

Now it is clear that the control of ROMO is an underdetermined optimization problem, in contrast to the robotic approach. Hence, it is possible to exploit the null space to find an optimal solution which minimizes a cost function. This technique is well known as control allocation and it is used for example in avionic applications where the amount of actuators is also greater than the degrees of freedoms of the body movement [Hae03].

3.5.3 The Concept of Energy Optimal Control Allocation

The technique of control allocation can be used in a wide range of over-actuated control problems. In comparison to conventional feedback control without optimization the main advantage is that it is easier to take actuator limits into account. Due to the control variable independent design it is possible to adapt the controller online without reconfiguring it. For example in case of an actuator fault the distribution mechanism can be easily modified to no longer comprising the defective actuating variable. Moreover, one can design secondary objectives within the problem formulation. This opens up the possibility to exploit the actuator redundancy for an energy optimal actuating variable determination.

Control Allocation Problem Formulation

In Figure 3.15 a typical closed loop control allocator scheme is shown. A higher-ranking controller calculates the demanded virtual control variables $\boldsymbol{v}$ based on the desired command variable reference $\boldsymbol{y}_{\mathrm{ref}}$ (this equates to the procedure described in Chapter 3.4). These are fed forward to the CA where they are mapped to the actuating variables and then commanded (input $\boldsymbol{u}$) to the over-actuated plant.

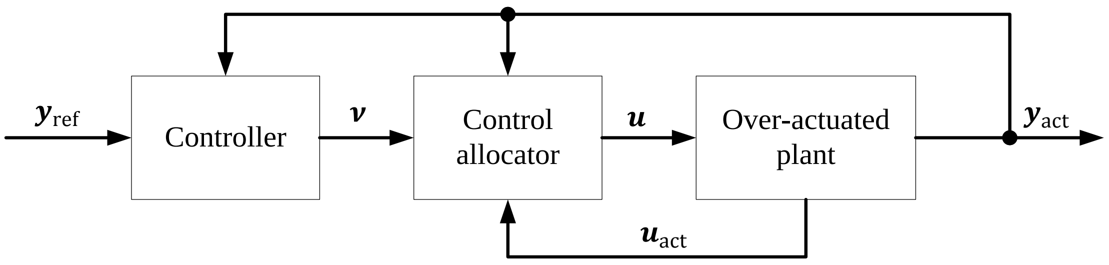

$\boldsymbol{y}_{\mathrm{ref}}$
Controller
$\boldsymbol{v}$
Control allocator
$\boldsymbol{u}$
Over-actuated plant
$\boldsymbol{y}_{\mathrm{act}}$
$\boldsymbol{u}_{\mathrm{act}}$

Figure 3.15: Control scheme of a closed loop control allocator

The mapping between the virtual control variables and the actuating variables is concatenated with the so called control actuating variables efficiency function $\boldsymbol{b}$. Mathematically, the general problem formulation of the control allocator writes as follows:

$$\begin{gathered} \boldsymbol{u} = \boldsymbol{b}^{-1}(\boldsymbol{v}) \\ \underline{\boldsymbol{u}} \leq \boldsymbol{u} \leq \overline{\boldsymbol{u}} \\ \boldsymbol{v} \in \boldsymbol{\Phi} \subseteq \mathbb{R}^{k}, \boldsymbol{u} \in \Omega \subseteq \mathbb{R}^{m};\ k < m \\ \Omega = [\underline{u}_1; \overline{u}_1] \times \cdots \times [\underline{u}_m; \overline{u}_m] \\ [\boldsymbol{v}] \text{ demanded virtual control variable} \\ [\underline{\boldsymbol{u}}, \overline{\boldsymbol{u}}] \text{ min/max (rate) actuator limits} \end{gathered} \tag{3.27}$$

There are several methods to solve the CA problem. Many of them are only appropriate for linear systems. The most common ones are the daisy chain method or a solution via a cascaded generalized inverse solution [Bec02], [Bus04]. As sketched in the beginning of this section, the goal is to select a technique for handling two objectives. For this reason an optimization based

CA is chosen [Hae03]**.** The principle is to minimize a cost function while handling the CA problem and the limits within the constraints. Formulated for the linear case, this leads to [Bec02]:

$$\begin{aligned} &\min_{\boldsymbol{u}}\big(J(\boldsymbol{u})\big) \\ \text{s.t. } &\boldsymbol{B}\cdot\boldsymbol{u}=\boldsymbol{v} \\ &\underline{\boldsymbol{u}}\le\boldsymbol{u}\le\overline{\boldsymbol{u}} \\ &\boldsymbol{b}(\boldsymbol{u})\equiv\boldsymbol{B}\cdot\boldsymbol{u};\ \boldsymbol{B}\in\mathbb{R}^{k\times m} \end{aligned} \tag{3.28}$$

If several solutions for the equality constraints $\boldsymbol{B}\cdot\boldsymbol{u}=\boldsymbol{v}$ exist, the one that minimizes the cost function $J(\boldsymbol{u})$ is chosen. In case there is no solution of the equation system, the optimization would fail and therefore a second step becomes necessary. Two of the most common solutions are “preserve direction”, not discussed here, or to “minimize the objective error” [Bec02]:

$$\begin{aligned} &\min_{\boldsymbol{u}}\|\boldsymbol{W}_\nu(\boldsymbol{B}\boldsymbol{u}-\boldsymbol{v})\| \\ \text{s.t. } &\underline{\boldsymbol{u}}\le\boldsymbol{u}\le\overline{\boldsymbol{u}} \end{aligned} \tag{3.29}$$

In eq. (3.29) $\boldsymbol{W}_\nu$ denotes the user tuneable weighting of the virtual control variable $\boldsymbol{v}$**.** Instead of this sequential optimization approach it can be transformed to a weighted one-step optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{u}}(J(\boldsymbol{u})+\gamma\|\boldsymbol{W}_\nu(\boldsymbol{B}\boldsymbol{u}-\boldsymbol{v})\|) \\ \text{s.t. } &\underline{\boldsymbol{u}}\le\boldsymbol{u}\le\overline{\boldsymbol{u}} \end{aligned} \tag{3.30}$$

The tuning parameter γ allows the prioritization between the primary $\boldsymbol{B}\cdot\boldsymbol{u}=\boldsymbol{v}$ and the secondary $\min J(\boldsymbol{u})$ optimization objective. By means of the analysis of the controlled plant (compare Figure 3.14) there are severe nonlinearities between the actuating variables and the virtual control variables. Therefore, the proposed method – eq. (3.28) & eq. (3.29) – has to be extended to handle a nonlinear control efficiency matrix [Hae03]. For this purpose the nonlinear CA problem is locally approximated by means of a Taylor series expansion of the control efficiency function $\boldsymbol{b}(\boldsymbol{u})$ cropped after the first term:

$$\boldsymbol{b}(\boldsymbol{u})\approx\boldsymbol{b}(\boldsymbol{u}_0)+\underbrace{\frac{\partial\boldsymbol{b}}{\partial\boldsymbol{u}}(\boldsymbol{u}_0)}_{\boldsymbol{B}(\boldsymbol{u}_0)}\cdot\underbrace{(\boldsymbol{u}-\boldsymbol{u}_0)}_{\Delta\boldsymbol{u}} \tag{3.31}$$

In this way a linear complementary problem for the nonlinear control efficiency function can be formulated:

$$\begin{aligned} \Delta\boldsymbol{v}&=\boldsymbol{B}(\boldsymbol{u}_0)\cdot\Delta\boldsymbol{u} \\ \underline{\boldsymbol{u}}-\boldsymbol{u}_0\le\Delta\boldsymbol{v}&\le\overline{\boldsymbol{u}}-\boldsymbol{u}_0 \\ \Delta\boldsymbol{v}&=\boldsymbol{v}-\boldsymbol{v}_0\,,\Delta\boldsymbol{u}=\boldsymbol{u}-\boldsymbol{u}_0 \end{aligned} \tag{3.32}$$

The solution of eq. (3.32) is a change of the variables – $\Delta\boldsymbol{u}$ follows now the commanded control variable variation $\Delta\boldsymbol{v}$. The approximated solution $\tilde{\boldsymbol{u}}$ of eq. (3.27) can be calculated as $\tilde{\boldsymbol{u}}=\boldsymbol{u}_o+\Delta\boldsymbol{u}$. In addition to the linearization of the actuating efficiency function also the boundary conditions are substituted to correspond to the actuating variable variation $\Delta\boldsymbol{u}$. Since the CA is used within a discrete-time system, the rate of change $\mathrm{d}\boldsymbol{u}/\mathrm{d}t$ can be expressed as $\Delta\boldsymbol{u}/T_s$, where T_s constitutes the sample time of the discrete control system.

The optimization procedure which minimizes a certain objective $J(\Delta\boldsymbol{u})$ (e.g. the minimum actuator energy effort) can be divided in two steps. In step one, a set of possible solutions Ω is found that respects the boundaries $\Delta\underline{\boldsymbol{u}}, \Delta\overline{\boldsymbol{u}}$ and incorporates the prioritization $\boldsymbol{W}_\nu$ of the virtual control variable $\Delta\boldsymbol{\nu}$. If the solution is not unique, the second step of the optimization process performs a search for a solution in Ω that minimizes the objective function $J(\Delta\boldsymbol{u})$:

$$\begin{aligned} &\text{Step 1: } \Omega = \underset{\boldsymbol{u}}{\operatorname{argmin}} \|\boldsymbol{W}_\nu - (\boldsymbol{B}\cdot\Delta\boldsymbol{u} - \Delta\boldsymbol{\nu})\|_2 \\ &\qquad\qquad \text{s.t. } \Delta\underline{\boldsymbol{u}}(T_s) \le \boldsymbol{u} \le \Delta\overline{\boldsymbol{u}}(T_s) \\ &\text{Step 2: } \Delta\boldsymbol{u} = \underset{\Omega}{\operatorname{argmin}} J(\Delta\boldsymbol{u}) \end{aligned} \tag{3.33}$$

In terms of computational effort solving the one-step optimization problem (eq. (3.30)) may be more efficient, but more difficult to tune. So it is decided to apply the two step approach, eq. (3.33), for controlling ROboMObil as described in the following chapters.

The Actuating Variables Efficiency Matrix of ROMO

The correlation between the demanded virtual control variable $\boldsymbol{\nu}^{\mathrm{C}}$ (eq. (3.25)) and the actuation control variable $\boldsymbol{u}^{\mathrm{W}}$ (eq.(3.26)) is described by a nonlinear actuating variable efficiency function $\boldsymbol{b}(\boldsymbol{u}^{\mathrm{W}})$. In a subsequent step the function is linearized so that algorithm eq. (3.33) can be utilized. The planar vehicle movement (compare Figure 3.14) is described by two linear momentum equations and one angular momentum equation. The first two specify the longitudinal (eq. (3.34)) and lateral (eq. (3.35)) movement, the third the rotation around the vertical axis with respect to the car coordinate system C ($x_{\mathrm{C}}, y_{\mathrm{C}}$ in Figure 3.14).

$$\dot{v}_x^{\mathrm{C}} = \frac{1}{m}\cdot\sum_i \left(F_x^{\mathrm{W}_i}\right) + \dot{\psi}^{\mathrm{C}}\cdot v_y^{\mathrm{C}} \tag{3.34}$$

$$\dot{v}_y^{\mathrm{C}} = \frac{1}{m}\cdot\sum_i \left(F_y^{\mathrm{W}_i}\right) - \dot{\psi}^{\mathrm{C}}\cdot v_x^{\mathrm{C}} \tag{3.35}$$

$$\begin{aligned} \ddot{\psi}^{\mathrm{C}} = \frac{1}{J_z^{\mathrm{C}}}\cdot\Big[&-\left(F_x^{\mathrm{W}_1} + F_x^{\mathrm{W}_3}\right)\cdot w_{\mathrm{l}} + \left(F_x^{\mathrm{W}_2} + F_x^{\mathrm{W}_4}\right)\cdot w_{\mathrm{r}} \\ &+\left(F_y^{\mathrm{W}_1} + F_y^{\mathrm{W}_2}\right)\cdot l_{\mathrm{f}} - \left(F_y^{\mathrm{W}_3} + F_y^{\mathrm{W}_4}\right)\cdot l_{\mathrm{r}}\Big] \end{aligned} \tag{3.36}$$

Here m indicates the vehicle mass, J_z^{C} is the vehicle yaw inertia and w_i, l_i are the distances between the wheel robots and the CoG. The unknown forces $F_x^{\mathrm{W}_i}, F_y^{\mathrm{W}_i}$ are calculated by the linear continuity of the wheel slip s^{W}, respectively the side slip angle α^{W}, and constant longitudinal respectively cornering tire stiffness factors. A complex tire model like Pacejka's magic formula [Ril01] is not considered here, since in this work the main focus of the investigation lies on moderate driving within the linear range of vehicle dynamics. By means of rearranging all equations for the virtual control variables $\boldsymbol{v}^{\mathrm{C}} = \left(v_x^{\mathrm{C}}, v_y^{\mathrm{C}}, \dot{\psi}^{\mathrm{C}}\right)$ it is possible to obtain the solution of the actuating variable efficiency function $\boldsymbol{b}\left(\boldsymbol{u}^{\mathrm{W}}, \boldsymbol{v}^{\mathrm{C}}\right)$:

$$\boldsymbol{b}(\boldsymbol{u}^{\mathrm{W}}, \boldsymbol{v}^{\mathrm{C}}) = \begin{bmatrix} 1/m & & \\ & 1/m & \\ & & 1/J_z^{\mathrm{C}} \end{bmatrix} \cdot \begin{pmatrix} \sum_i F_x^{\mathrm{W}_i} \\ \sum_i F_y^{\mathrm{W}_i} \\ \sum_i M_z^{\mathrm{W}_i} \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ \dot{\psi}^{\mathrm{C}} \end{pmatrix} \times \begin{pmatrix} v_x^{\mathrm{C}} \\ v_y^{\mathrm{C}} \\ 0 \end{pmatrix} \tag{3.37}$$

The function linearization, using eq. (3.31), yields a nonlinear actuating variables efficiency matrix:

$$\boldsymbol{B}\left(\Delta\boldsymbol{u}^{\mathrm{W}}, \Delta\boldsymbol{v}^{\mathrm{C}}\right) = \begin{bmatrix} \frac{\partial b_1}{\partial u_1} & \cdots & \frac{\partial b_m}{\partial u_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial b_1}{\partial u_k} & \cdots & \frac{\partial b_m}{\partial u_k} \end{bmatrix} \in \mathbb{R}^{3\mathrm{x}8} \tag{3.38}$$

Note that the nonlinearities are stated by the trigonometric functions of the vehicle kinematics and the velocity vectors in their corresponding coordinate systems. In the following sections two optimization cost functions for the second step of the proposed optimization problem formulation (eq. (3.33)) are investigated.

The Physically Motivated Cost Function

The first approach considers a physically motivated cost function design which is based on multiphysical models of the electric drives and their drivetrains. For this reason ROMO’s field-oriented controlled permanent magnet synchronous motor (PMSM) models (see Chapter C.2) and steering actuator models (compare Chapter 2.2.1), have been developed. The modeling depth is determined on the level of real-time capability, quasi-stationary behavior and linear loss factors. The models use a dq-frame approach [Ong98], which neglects the inverter switching. This leads to a good value in terms of simulation performance. The modeled energy losses, with respect to the flow from electric (input) to mechanic power (output), in a PMSM can be outlined as follows:

- inverter losses (switching and basic load)
- copper losses (coil resistance)
- iron losses (reversal of magnetism)
- mechanical losses (friction effects e.g. bearings)
- inverter losses (efficiency dependent)

Note that the field weakening operation mode is neglected here, since it’s not implemented in the field-oriented control of ROMO’s traction drives. That means solely the I_{q} phase current is commanded. The electric losses of the wheel robots traction motor $E_{\mathrm{loss}}^{\mathrm{TM}}$can be formulated as a second order polynomial function with respect to the actual wheel speeds $\omega^{\mathbf{W}}$and the wheel torques $\tau^{\mathbf{W}}(I_{\mathrm{q}})$:

$$E_{\mathrm{loss}}^{\mathrm{TM}}(\omega^{\mathbf{W}},\tau^{\mathbf{W}}) = \int_t^{t+T_s} \sum_{i=1}^{4} \Big(P_{\mathrm{inv,const}}^{\mathrm{W}_i} + \big(\tau_{fric}^{\mathrm{W}_i,\mathrm{TM}} + k_{\mathrm{hyst}}\big)\cdot\big|\omega^{\mathrm{W}_i,\mathrm{TM}}\big|$$
$$+ k_{\mathrm{eddy}}\cdot\omega^{2\mathrm{W}_i,\mathrm{TM}} + a\cdot\big|\tau^{\mathrm{W}_i}\big| + b\cdot\tau^{2\mathrm{W}_i,\mathrm{TM}}\Big)\,\mathrm{d}t$$
$$a = \frac{k_{\mathrm{inv}}\cdot 2}{3\cdot z_{\mathrm{p}}\cdot\psi_{\mathrm{PM}}}\ ;b = \frac{2\cdot R_{\mathrm{s}}}{3\cdot z_{\mathrm{p}}^2\cdot\psi_{\mathrm{PM}}^2} \tag{3.39}$$
$$\tau^{\mathbf{W},\mathrm{TM}} = I_{\mathrm{q}}^{\mathbf{W},\mathrm{TM}}\cdot {}^{3}\!/_{2}\cdot z_{\mathrm{p}}\cdot\psi_{\mathrm{PM}}$$

The parameter significances of ROMO's traction motors loss function are stated as follows:

Table 3.3: Explanation of the PMSM model parameters

Parameter	Unit	Description
$P_{\mathrm{inv,const}}$	[W]	Inverter losses constant
k_{hyst}	$\left[{}^{\mathrm{Ws}}/_{\mathrm{rad}}\right]$	Hysteresis losses constant
τ_{fric}	$\left[{}^{\mathrm{Ws}}/_{\mathrm{rad}}\right]$	Motor friction constant
k_{eddy}	$\left[{}^{\mathrm{Ws}^2}/_{\mathrm{rad}^2}\right]$	Eddy current losses constant
k_{inv}	$\left[{}^{\mathrm{W}}/_{\mathrm{I}}\right]$	Current dependent inverter losses
R_{s}	[Ω]	Warm resistance per phase
z_{p}	[-]	Pole pair number
ψ_{PM}	[Wb]	Magnetic flux of permanent magnets

The electric machine parameter values for the implemented drives (traction and steering) can be found in App. Table C.4. The steering actuators are mounted on the wheel carriers and transfer their torque to the upper wishbone of the chassis (compare Figure 2.4). Their drivetrain consists of a low torque, high speed PMSM and a cycloidal gear box connected to a spur gear which meshes with a larger gear-wheel that is mounted on the upper wishbone.

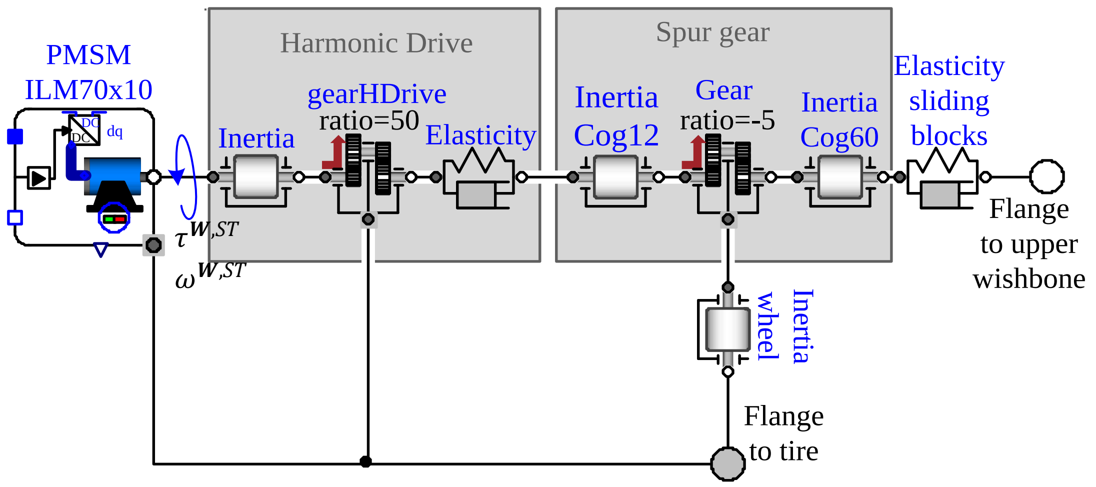


Figure 3.16: Schematic steering drivetrain and PMSM model in Modelica

In addition to the losses of the PMSM itself, the following effects are taken into account:

- the acceleration work $\tau_{\mathrm{acc}}^{\mathrm{ST},\mathbf{W}}$, due to the wheel hub inertia and the steering drivetrain,
- the gearboxes losses $\tau_{\mathrm{gearloss}}^{\mathrm{ST},\mathbf{W}}$ and
- the work to overcome tire bore torque $\tau_{\mathrm{bore}}^{\mathrm{ST},\mathbf{W}}$ and the speed dependent self-aligning torque $\tau_{\mathrm{align}}^{\mathrm{ST},\mathbf{W}}$.

The upper index ST, denotes that it refers to the designated steering actuator W_i. In total the mechanical losses of the steering driveline can be written as:

$$P_{\mathrm{mech}}^{\mathbf{W},\mathrm{ST}} = \left(\tau_{\mathrm{acc}}^{\mathbf{W},\mathrm{ST}} + \tau_{\mathrm{gearloss}}^{\mathbf{W},\mathrm{ST}} + \tau_{\mathrm{bore}}^{\mathbf{W},\mathrm{ST}} + \tau_{\mathrm{align}}^{\mathbf{W},\mathrm{ST}}\right) \cdot \omega^{\mathbf{W},\mathrm{ST}} \tag{3.40}$$

Adding the mechanical losses from eq. (3.40) to the formulation of the electrical losses the total losses of the steering drivelines E_{los}^{ST} results in:

$$\begin{aligned} E_{\mathrm{loss}}^{\mathrm{ST}}\left(\omega^{\mathbf{W},\mathrm{ST}}, \tau^{\mathbf{W},\mathrm{ST}}\right) = \int_t^{t+T_s} \sum\nolimits_{i=1}^{4} \Big(P_{\mathrm{mech}}^{\mathrm{W}_i,\mathrm{ST}} + P_{\mathrm{inv,const}}^{\mathrm{W}_i,\mathrm{ST}} + \\ +\left(\tau_{\mathrm{fric}}^{\mathrm{W}_i,\mathrm{ST}} + k_{\mathrm{hyst}}\right) \cdot \left|\omega^{\mathrm{W}_i,\mathrm{ST}}\right| + k_{\mathrm{eddy}} \cdot \omega^{2\mathrm{W}_i,\mathrm{ST}} + a \cdot \left|\tau^{\mathrm{W}_i,\mathrm{ST}}\right| \\ +b \cdot \tau^{2\mathrm{W}_i,\mathrm{ST}} \Big) \,\mathrm{d}t \end{aligned} \tag{3.41}$$

To avoid the evaluation of the integral, especially when linearizing the cost function (see QP solver interface in eq. (3.42)), it is presupposed, when minimizing the overall power consumption also the energetic effort is minimized. This assumption is valid by reasons that the power function is monotonic during one sample optimization step T_s. In Figure 3.17 two possible trends (colored lines) for the electric power consumption within one discrete optimization step T_s are shown. The area enclosed between two time steps (i.e. $[t \;\; t + T_s]$) underneath the electric power function curves equates the integral and therefore the expended energy.

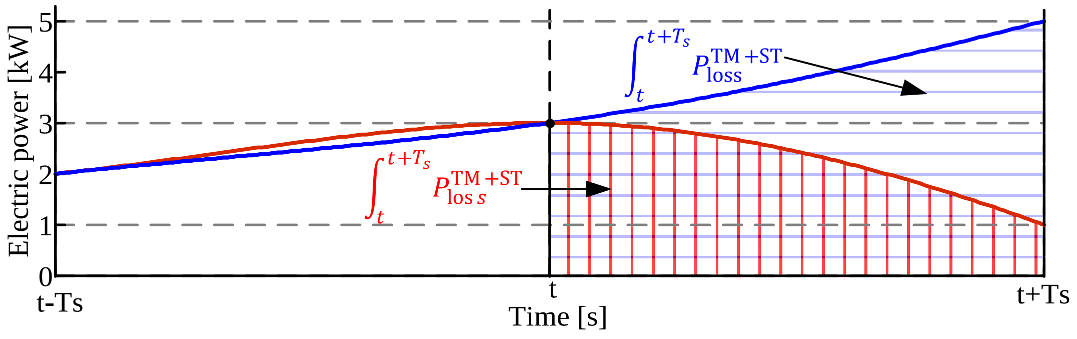


Figure 3.17: Schematic motivation for energy minimization through power minimization

Through this simplification the power consumption also minimizes the energetic effort, which is the integral over the power consumption function. I.e. the integrals in eq. (3.39) and eq. (3.41) can be neglected and for the latter only the sum of all power losses is considered ($E_{\mathrm{loss}}^{\mathrm{ST}}\left(\omega^{\mathbf{W},\mathrm{ST}}, \tau^{\mathbf{W},\mathrm{ST}}\right)$ and $E_{\mathrm{loss}}^{\mathrm{TM}}\left(\omega^{\mathbf{W}}, \tau^{\mathbf{W}}\right)$).

The Heuristic Motivated Cost Function

Complementary to the complex physically motivated cost function, an alternative simple cost function is formulated in this section, which bases on heuristic considerations. It is assumed that for minimizing the vehicle's energy consumption the actuating variables should be controlled by these rules:

- The steering rate $\dot{\delta}^{\mathbf{W}}$ should be minimized (tending to zero) to avoid mechanical losses (compare eq. (3.40)) and
- the traction motor torque $\tau^{\mathbf{W}}$ should be chosen in a way that recuperation is maximized.

These regulations can be realized by a simple cost function, which can be tuned offline. As pointed out in the beginning of this section, this method results in a slim function expression and is therefore very appropriate for real-time applications. Details of the mathematical formulation are given in the next section.

3.5.4 Control Allocator Implementation in Modelica

Since ROMO's simulation model is implemented in Modelica [Mod17] (for details see Chapter C.2), also the discrete-time controllers for the evaluation are implemented in the Modelica language (compare Figure 3.18). For the solution of the CA optimization problem a quadratic program solver (QP), that can handle equality and inequality constraints, is interfaced to the Modelica simulator Dymola (compare eq. (3.42)). In this particular case the QL solver [Sch05] showed best performance and reliability and was therefore chosen for the further work.

$$\begin{aligned} \boldsymbol{x}^* &= \underset{\boldsymbol{x}}{\operatorname{argmin}} \left(\frac{1}{2} \boldsymbol{x}^T \boldsymbol{H} \boldsymbol{x} + \boldsymbol{f}^T \boldsymbol{x} \right) \\ \text{s.t. } & \boldsymbol{x}_{\min} \leq \boldsymbol{x} \leq \boldsymbol{x}_{\max} \\ & \boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b} \, ; \, \boldsymbol{L} \cdot \boldsymbol{x} \leq \boldsymbol{k} \end{aligned} \tag{3.42}$$

The optimization problem from eq. (3.33) is stated as a constrained least squares minimization problem and must be transformed by matrix calculus to a QP compliant one. In this way the first step of the optimization is calculated by:

$$\begin{gathered} \min_{\Delta \boldsymbol{u}^{\mathrm{W}}} \left(\frac{1}{2} \Delta \boldsymbol{u}^{\mathrm{W}^T} \boldsymbol{H} \Delta \boldsymbol{u}^{\mathrm{W}} + \boldsymbol{f}^T \Delta \boldsymbol{u}^{\mathrm{W}} \right) \\ H = 2 \cdot \boldsymbol{B}^T \boldsymbol{W}_\nu^T \boldsymbol{W}_\nu \boldsymbol{B} \ ; \ \boldsymbol{f} = -2 \cdot \boldsymbol{B}^T \boldsymbol{W}_\nu^T \boldsymbol{W}_\nu \Delta \boldsymbol{\nu}^C \\ \boldsymbol{W}_\nu = \text{tuning matrix} \end{gathered} \tag{3.43}$$

In the second step, if there is a null space solution for eq. (3.43) – this can be assumed, due to finite machine precision, if $\boldsymbol{B} \cdot \Delta \boldsymbol{u}^{\boldsymbol{W}} < \sim 100 \cdot \epsilon$ holds – the cost function needs to be reformulated as follows.

$$\begin{gathered} J(\Delta \boldsymbol{u}^{\mathrm{W}}) = \frac{1}{2} \Delta \boldsymbol{u}^{\mathrm{W}^T} \boldsymbol{E} \Delta \boldsymbol{u}^{\mathrm{W}} + \boldsymbol{e}^T \Delta \boldsymbol{u}^{\mathrm{W}} \\ \text{s.t. } \boldsymbol{B} \cdot \Delta \boldsymbol{u}^{\mathrm{W}} = \Delta \boldsymbol{\nu} \end{gathered} \tag{3.44}$$

Derivation of $J_{\text{phys}}(\Delta\boldsymbol{u}^{\mathbf{W}})$

The physically motivated cost function, proposed in Chapter 3.5.3, has to be linearized via a second order Taylor series expansion to meet the QP interface of eq. (3.41). The operating point needs to be selected separately for the steering and the traction motor power. The steering actuator power demand depends on the change of the steering angle, i.e. its rate. Therefore the expansion is chosen around the operating point from the last sample time step $\Delta\boldsymbol{\delta}^{\mathbf{W}}(t-T_s)$ $(\boldsymbol{D_1}, \boldsymbol{f_1}$ in eq. (3.45)). On the contrary, the drive torques $\boldsymbol{\tau}^{\mathbf{W}}$ operation point is set to the current time step t $(\boldsymbol{D_2}, \boldsymbol{f_2}$ in eq. (3.45)).

$$\begin{aligned}\boldsymbol{P}_{\text{quad}} &= \frac{1}{2}\cdot\boldsymbol{y}^{T}\cdot\begin{pmatrix}\boldsymbol{D}_1 & 0\\ 0 & \boldsymbol{D}_2\end{pmatrix}\cdot\boldsymbol{y}+(\boldsymbol{f}_1,\boldsymbol{f}_2)^T\cdot\boldsymbol{y}\\ \underline{\boldsymbol{y}} &= \begin{pmatrix}\Delta\boldsymbol{\delta}_t^{\mathbf{W}}-\Delta\boldsymbol{\delta}_{t-T_s}^{\mathbf{W}}\\ \boldsymbol{\tau}_{t+T_s}^{\mathbf{W}}-\boldsymbol{\tau}_t^{\mathbf{W}}\end{pmatrix}\end{aligned} \tag{3.45}$$

For the desired formulation as an optimization objective applicable to eq. (3.44), the quadratic function $\boldsymbol{P}_{\text{quad}}$ of eq. (3.45) is symbolically manipulated to obatain $\boldsymbol{E}, \boldsymbol{e}$ in dependency of $\Delta\boldsymbol{u}^{\mathbf{W}}$.

Derivation of $J_{\text{heur}}(\Delta\boldsymbol{u}^{\mathbf{W}})$

The heuristic cost function (see Chapter 3.5.3) is considered as a weighted minimization problem of the distance between the actuating variable $\Delta\boldsymbol{u}^{\mathbf{W}}$ and the demand $\Delta\boldsymbol{u}_{\text{d}}^{\mathbf{W}}$. The entries of the weighting matrix $\boldsymbol{W}_{\text{u}}$ are the outcome of an offline closed loop optimization with the DLR optimization software MOPS [Joo08].

$$\begin{aligned}&J(\Delta\boldsymbol{u}^{\mathbf{W}}) = \| \boldsymbol{W}_{\text{u}}(\Delta\boldsymbol{u}^{\mathbf{W}}-\Delta\boldsymbol{u}_{\text{d}}^{\mathbf{W}}) \|_2\\ &\xrightarrow{\text{yields}} \boldsymbol{E} = 2\cdot\boldsymbol{W}_{\text{u}}^{T}\boldsymbol{W}_{\text{u}}\ ;\ \boldsymbol{e} = -2\cdot\boldsymbol{W}_{\text{u}}^{T}\boldsymbol{W}_{\text{u}}\Delta\boldsymbol{u}_{\text{d}}\end{aligned} \tag{3.46}$$

The heuristic optimization objective of Chapter 3.5.3 is expressed in dependency of the actuating variation vector $\Delta\boldsymbol{u}_{\text{d}}$. The steering rates are set to zero and the drive torque rates are affected to maximum available recuperation dependent on the current vehicle state.

$$\begin{aligned}\Delta\boldsymbol{u}_{\text{d}}^{\mathbf{W}}(t) &= \left(0,0,0,0,\tau_{\text{d}}^{\text{W}_1}(t),\tau_{\text{d}}^{\text{W}_2}(t),\tau_{\text{d}}^{\text{W}_3}(t),\tau_{\text{d}}^{\text{W}_4}(t)\right)\\ \boldsymbol{\tau}_{\text{d}}^{\mathbf{W}}(t) &= \operatorname{sgn}\left(\boldsymbol{\omega}^{\mathbf{W}}(t)\right)\cdot\boldsymbol{\tau}_{\text{mot,max}}^{\mathbf{W}}-\boldsymbol{\tau}^{\mathbf{W}}(t)\end{aligned} \tag{3.47}$$

3.5.5 Evaluation of the Energy Manager Control Allocator Performance

In this chapter an experimental setup with the proposed energy manager control allocator is investigated. Figure 3.18 shows the simulation control setup implemented in Modelica. A path interpolation module generates the parameterized spatial path $\boldsymbol{\lambda}(s)$ for the path following controller (compare Chapter 3.4). By hands of the path following PD controllers, incorporating $\boldsymbol{\lambda}, \boldsymbol{p}_{\text{act}}^{\text{C}}, v_{x_{\text{act}}}^{\text{C}}$, the commanded change of the virtual control variables $\Delta\boldsymbol{v}^{\text{C}}$ is calculated.

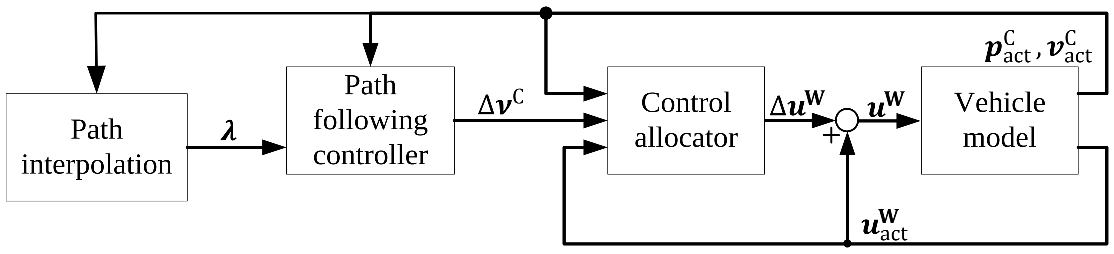


Figure 3.18: Scheme for the vehicle control by means of a control allocator

The CA block is composed of three parts: the linearization of the actuating variables efficiency matrix, the calculation of the dynamic cost function and the solution of the linear complementary problem. ROMO's vehicle model is based on a parameterized skate board vehicle model (compare Chapter 3.7) which gives a good tradeoff between accuracy and simulation speed. Besides longitudinal and lateral movement it also reflects vertical dynamics, like wheel-load fluctuations (cf. Chapter 3.7 and Chapter C.2).

Simulative Evaluation of the Control Allocation Algorithm

For the comparison of the different control strategies an ISO double lane-change driving maneuver with a predefined velocity profile is used (cf. Figure 3.19). The focus of the investigations is on reproducibility. To achieve this, a lateral acceleration range is stated to the vehicle reference speed generation (compare Chapter 3.3.1) which does not reach the threshold of the linear region of the tire and vehicle dynamics.

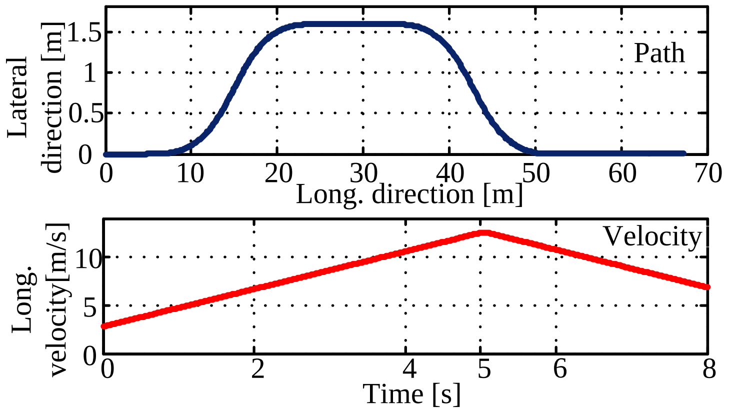


Figure 3.19: ISO 3888-1 similar double lane change maneuver

The maneuver has a total simulation time of $t = 8$ s. The initial car velocity $v^C_{x_0}$ is 3 m/s and from the beginning the car accelerates with 2 m/s^2 up to time $t = 5$ s and then changes volatile to -2 m/s^2. This guarantees a considerable demand change especially to the traction motors to gain the potential of the CA optimization strategy. The conclusion of the investigations is that both simulations with the physically motivated and the heuristic (cf. Chapter 3.5.3) cost function show similar results in the overall energy consumption (see Table 3.4). This observation will be discussed in more detail in the following Chapter 3.7.

Table 3.4: Overall energy consumption for exemplary lane-change maneuver

Variant	Energy consumption
Physically motivated cost function	59.875 kJ
Heuristic cost function	59.882 kJ

Moreover, the characteristics of the actuator variable commands differ only negligibly from each other and both of them generate feasible commands:

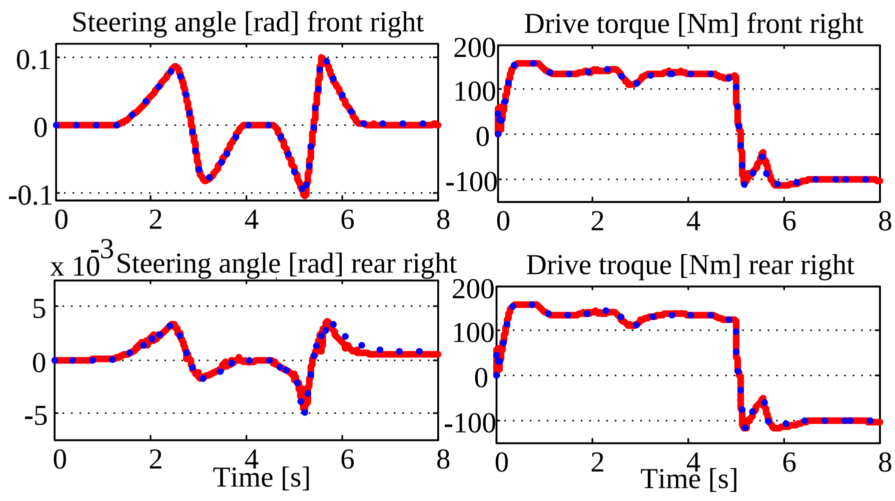


Figure 3.20: Actuating variables trajectories (solid = physical, dotted = heuristic)

3.6 The Interconnectivity of the Energy Manager Controller Levels

In this final section of the EM framework derivation the complete signal flow diagram of the multi-layer control structure is given in Figure 3.21. The background colors are matched to the one of the EM concept in Figure 3.1, the hatched colored modules are not part of this thesis. For further understanding following points are assumed for the controller implementation. The spatial description of the road and the navigation between two points are given by a system like ADAS RP (Advanced Driver Assistance System Research Platform by HERE Inc.) with an electric horizon or in case of an simulation a system like the DLR Vehicle Controls Library (VCL) using the Open DRIVE standard (compare Chapter C.2). Information about upcoming cars or other road narrows must be accomplished by vehicle perception or car2x technologies. Vehicle states $\boldsymbol{p}^{\mathrm{I}}_{\mathrm{C_{act}}}, \psi^{\mathrm{I}}_{\mathrm{C_{act}}}, \boldsymbol{v}^{\mathrm{C}}_{\mathrm{act}}, \dot{\psi}^{\mathrm{C}}_{act}$, denoted with red arrows, are assumed to be available and precise – in Chapter 5.2 a model based method for estimating these values is given. In addition, it is assumed that the power source of ROMO is sufficiently capable to fulfill the motion request. A model based battery state observer able to estimate the state of charge l_{HV} and the current battery power availability $P_{\mathrm{HV_{max}}}$ (cf. Chapter 5.1) feeds additional information into the control structure (orange arrows). Potential degradation or switches in control strategy are discussed in Chapter 5.1.3. The controller is implemented in a discrete-time rapid prototyping controller that is capable to fulfill the calculation within one deterministic calculation step (i.e. a hard real-time system).

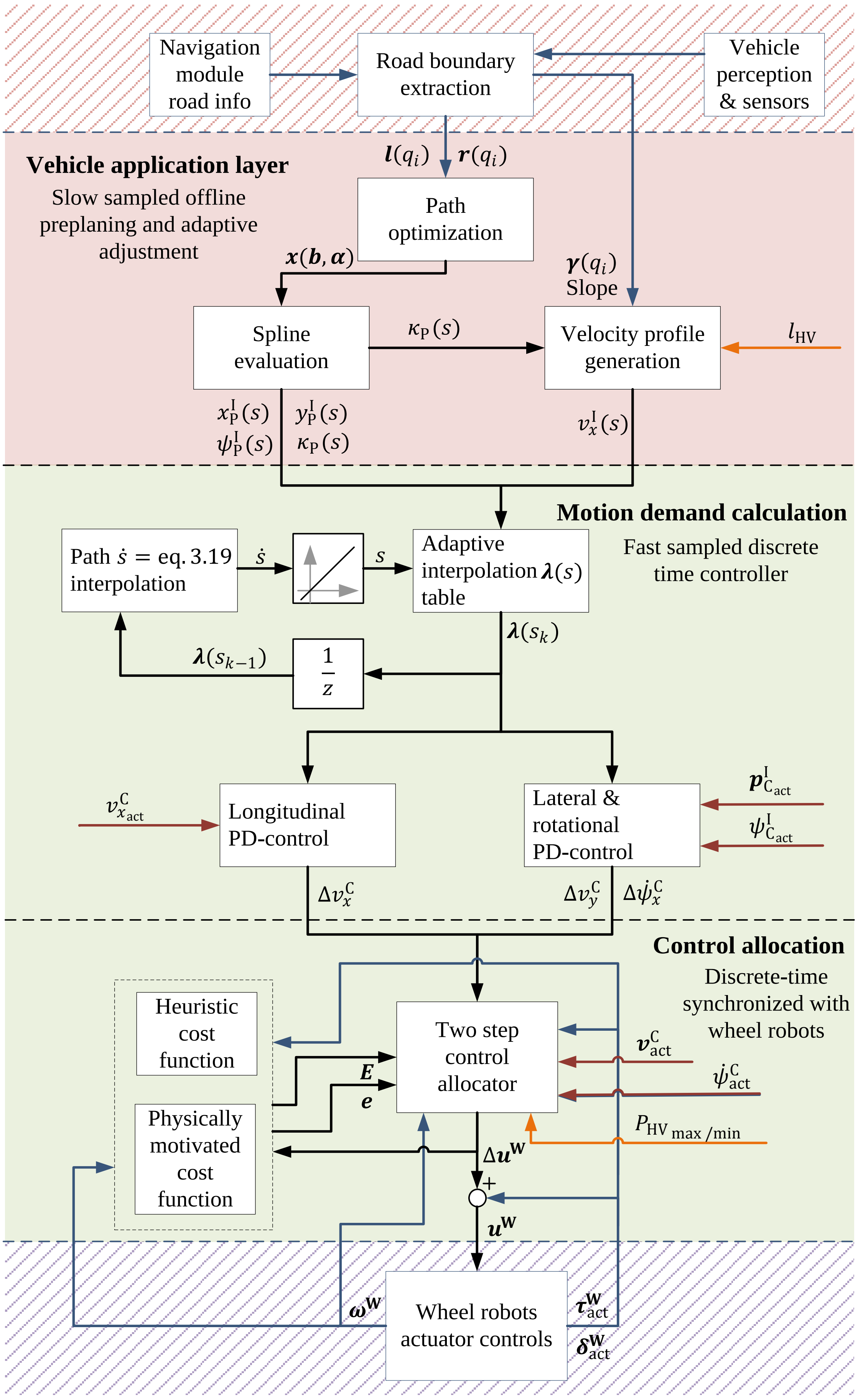

Navigation module road info
Road boundary extraction
Vehicle perception & sensors
$l(q_i)$ $r(q_i)$
Vehicle application layer
Slow sampled offline preplaning and adaptive adjustment
Path optimization
$x(b,\alpha)$
$\gamma(q_i)$ Slope
Spline evaluation
$\kappa_P(s)$
Velocity profile generation
l_{HV}
$x_P^I(s)$ $y_P^I(s)$ $\psi_P^I(s)$ $\kappa_P(s)$
$v_x^I(s)$
Motion demand calculation
Fast sampled discrete time controller
Path interpolation $\dot{s}$ = eq. 3.19
$\dot{s}$
s
Adaptive interpolation table $\lambda(s)$
$\lambda(s_k)$
$\lambda(s_{k-1})$
$\frac{1}{z}$
$v_{x_{act}}^C$
Longitudinal PD-control
Lateral & rotational PD-control
$p_{C_{act}}^I$
$\psi_{C_{act}}^I$
Δv_x^C
Δv_y^C $\Delta\dot{\psi}_x^C$
Control allocation
Discrete-time synchronized with wheel robots
Heuristic cost function
Physically motivated cost function
E
e
Two step control allocator
v_{act}^C
$\dot{\psi}_{act}^C$
$P_{HV\,max/min}$
Δu^W
+
u^W
ω^W
Wheel robots actuator controls
τ_{act}^W
δ_{act}^W

Figure 3.21: Overall control concept of the spatial energy manager

3.7 Evaluation of the Proposed Energy Manager for ROMO

For the assessment of the EM algorithm a detailed multiphysical Modelica model of ROboMObil was designed providing a good tradeoff between level of detail and computational complexity (Chapter C.2). The simulation environment (cf. Figure 3.22) is composed by electrical models i.e. the high voltage (HV) battery (for details see Chapter 5.1.2) the powertrain of ROMO's axle modules with its in-wheel motors, wheel carrier mounted by-wire steering systems, DC/DC buck converter, a low voltage (LV) backup battery supply, ROMO's chassis composed of a multi-body dynamics suspension system with a Pacejka tire model [Pac12] and the discrete-time EM algorithm implementation according to the flow diagram in Figure 3.21.

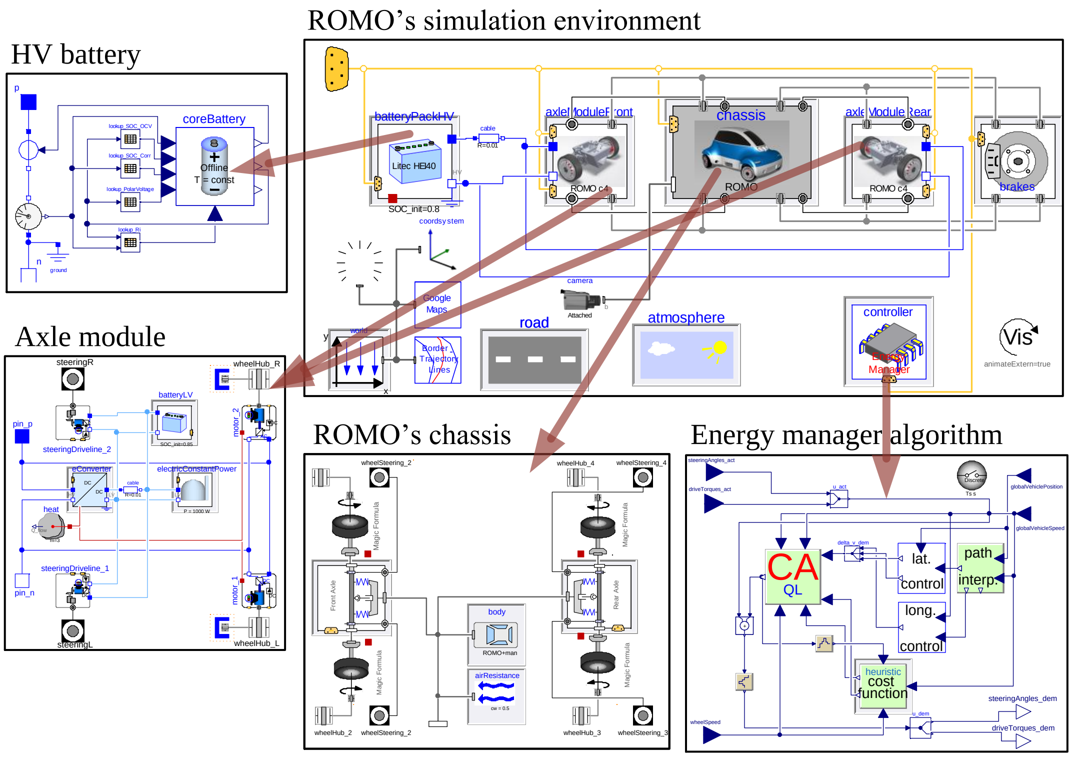


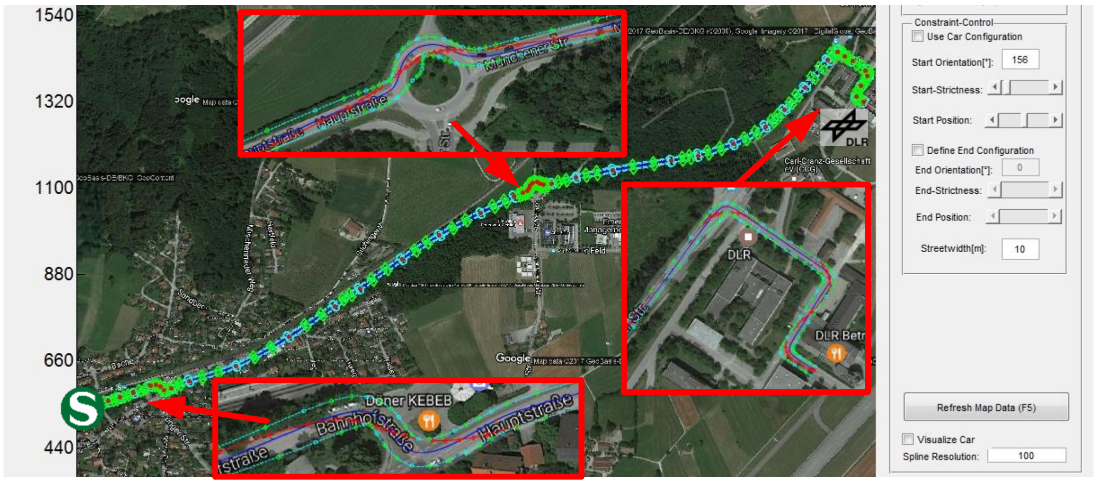
Figure 3.22: ROMO's multiphysical Modelica model and EM controller

In the following subsections different driving scenarios are analyzed. In all cases the parametric path description $\boldsymbol{\lambda}(\mathbf{s})$ is computed offline with a horizon length lasting across the whole test track.

In total three different virtual test tracks have been simulated, all of them with a path description that in the non-optimized case relies on the right lane of the road (non-opt.) and in the optimized case being able to use the full width of the driveway (opt.). To generate comparable results, the maximum longitudinal acceleration and velocity are reduced in the optimized case so that the planned journey time is identical to the one of the non-optimized path. The reason is that the optimized path has larger and smoother arc radii and therefore higher velocities are possible (compare Chapter 3.3.1), but that would lead to a shorter travelling time raised from higher energy demands from the propulsion.

3.7.1 Flat Test Track from DLR's TechLab to S-Bahn Station Weßling

This first experiment track is generated with DLR's interactive path planning GUI, a MATLAB tool that makes use of Google Maps satellite pictures and enables the user to design routes and road boundaries in the real world (Figure 3.23). The so designed routes are then forwarded to the here proposed path optimization and velocity profile generation algorithms.



Constraint-Control
☐ Use Car Configuration
Start Orientation[°]: 156
Start-Strictness:
Start Position:
☐ Define End Configuration
End Orientation[°]: 0
End-Strictness:
End Position:
Streetwidth[m]: 10
Refresh Map Data (F5)
☐ Visualize Car
Spline Resolution: 100

Figure 3.23: Excerpt from DLR's interactive path planning GUI

The above picture details show the route between DLR's TechLab (Figure 3.23 – right), where the ROboMObil laboratory is located, to the next public train station S-Bahn in Weßling (Figure 3.23 – left). The length of the track is about 2.5 km, it is assumed to be completely plane, and a common commuter way for DLR's co-worker who use public transportion from and to the inner city of Munich. The motivation of this experiment is that ROMO, as a semi-autonomous vehicle, could overcome this flaw, at least here in the simulation case study. The track begins with three sharp curves and then leaves the DLR facilities to the rural road. In the middle of the way to Weßling a roundabout is located, followed by another rural road segment. Just before the railway station a s-curve leads to the end of the track.

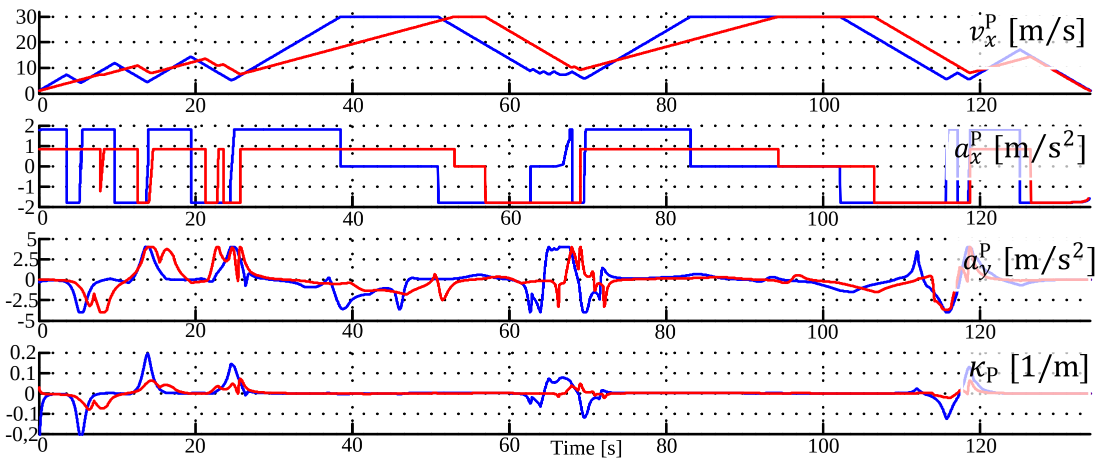


Figure 3.24: Trajectories from DLR TechLab to the railroad station (blue non-opt. – red opt.)

In Figure 3.24 the temporal path description of the vehicle dynamics quantities $v_x^{\mathrm{P}}, a_x^{\mathrm{P}}, a_y^{\mathrm{P}}, \kappa_{\mathrm{P}}$ is given. The demanded longitudinal velocity gradient as well as the acceleration is considerably reduced in the optimized case compared to the non-optimized case. The curvature is most significantly reduced which partly yields a reduced lateral acceleration which is more comfortable to the passenger. Table 3.5 summarizes the necessary energy for the realization of the path demands by the path-following controller and the energy optimal control allocator.

Table 3.5: Energy manager effectiveness assessment – test track DLR to S-Bahn Weßling

Experiment	E_{con}	ΔE	**Difference**
Non-opt. – Heuristic	0.3348 kWh	Reference	–
Non-opt. – Physical	0.3386 kWh	+0.0038 kWh	+1.135 %
Opt. – Heuristic	0.2968 kWh	−0.0380 kWh	−11.350 %
Opt. – Physical	0.3006 kWh	−0.0342 kWh	−10.215 %

In this test scenario the approach yields a saving of about 10 % in comparison to the non-optimized case. Furthermore, the difference between the heuristic and the physically motivated cost function is small, with the heuristic cost function performing a little better. The justification can be found in the analysis of the torque and steering demands, which show jittering during constant driving phases. A reason for this is the aggressive tuning of the overlaying path following controller, as well as the control allocation approach, which only concerns a very small step $T_s = 4$ ms without consideration of future or past demand trajectories (compare Figure 3.17).

3.7.2 Rural Road Test Track Vires Road Segment 2501

After the first promising test scenario a second one is chosen using the Vires landscape and road modelling (Figure 3.25) [Vir17b]. In this virtual world also the road height profile can be considered in the vehicle velocity profile generation (see drag force consideration in Figure 3.7).

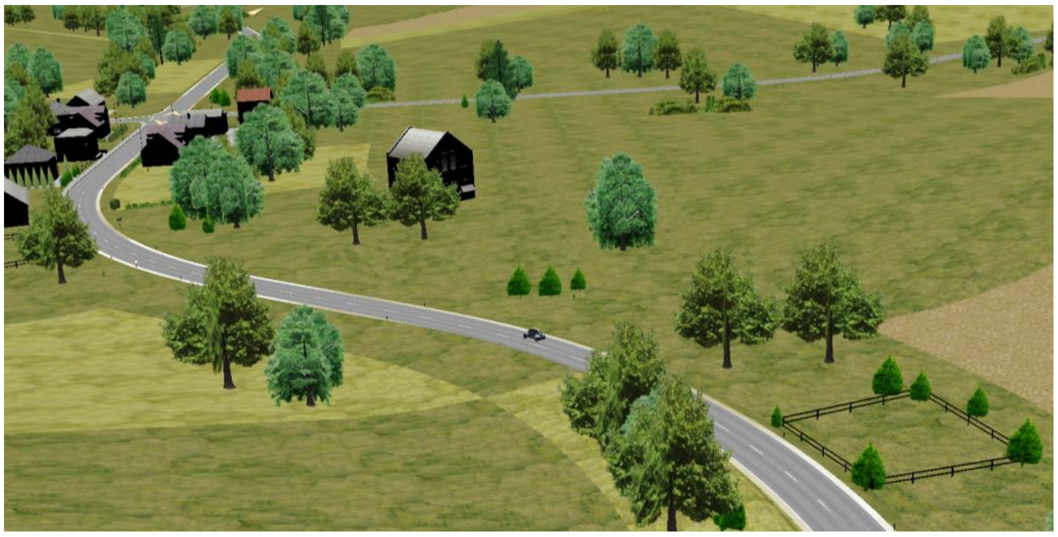


Figure 3.25: ROboMObil on Vires 2501 rural road

In Figure 3.26 the selected 3 km open circuit track and its height profile is depicted. Besides long bend curves there are two sharp bends at crossings.

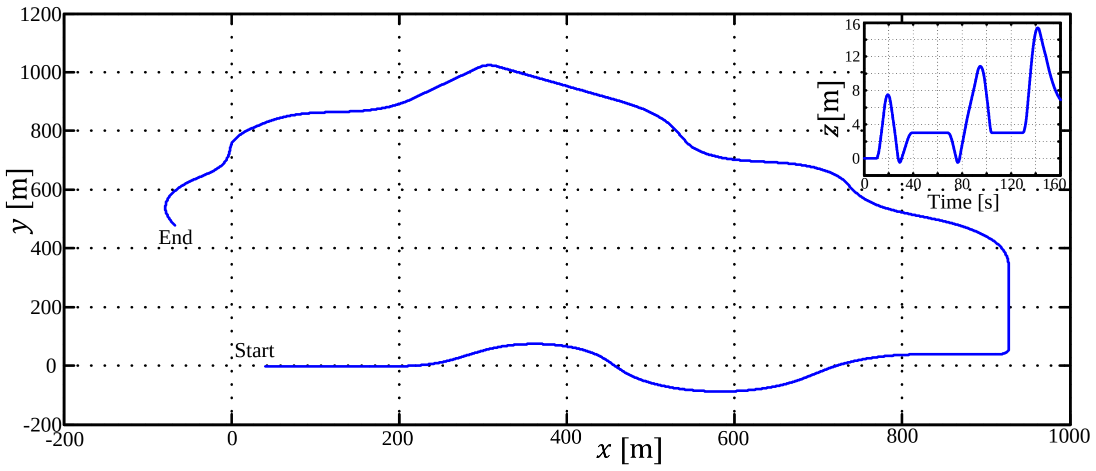


Figure 3.26: Path and height profile of Vires 2501 road segment

As before, the demanded values are given in Figure 3.27. It is eye-catching that the trajectories do not differ too much from each other, meaning that the lateral acceleration and the longitudinal velocity are very similar in both cases. The longitudinal acceleration is kept longer on a high level in the optimized case in comparison to the non-optimized case although the maximum level is reduced to meet the constraint that both versions should have the same travel time for the route. This leads in the optimized case to shorter phases of negative acceleration demands in which the traction motors are able to recuperate. Additionally, the path curvature could only be slightly reduced by the optimizer in the crossing segments.

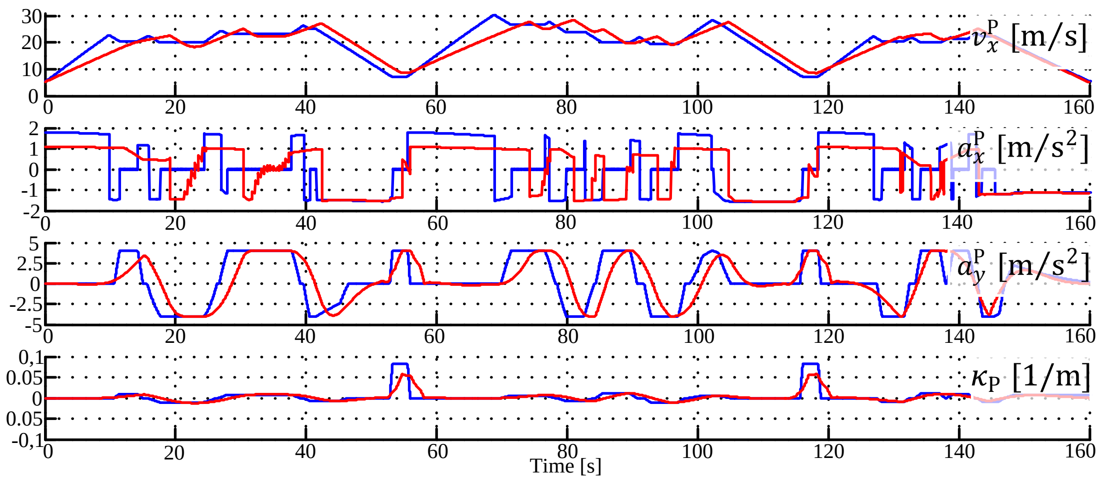


Figure 3.27: Vires 2501 experiment trajectories (blue non-opt. – red opt.)

These observations are reflected in the comparison of the energy consumption of the different path and control allocator configurations in Table 3.6. The maximum energy saving is reduced to only ∼4 % and as before, the trend of the physical motivated cost function is the same.

Table 3.6: Energy manager effectiveness assessment – Vires road segment 2501 – 3000 m

Experiment	E_{con}	ΔE	**Difference**
Non-opt. – Heuristic	0.4110 kWh	Reference	–
Non-opt. – Physical	0.4173 kWh	+0.0063 kWh	+1.532 %
Opt. – Heuristic	0.3955 kWh	−0.0155 kWh	−3.771 %
Opt. – Physical	0.4009 kWh	−0.0101 kWh	−2.457 %

3.7.3 Mountain road descent test track Vires road segment 2716

The last test track is a descent rural road segment of 3 km length (Figure 3.28). ROboMObil starts from the top of the mountain and then drives downhill for 120 m of height (Figure 3.29).

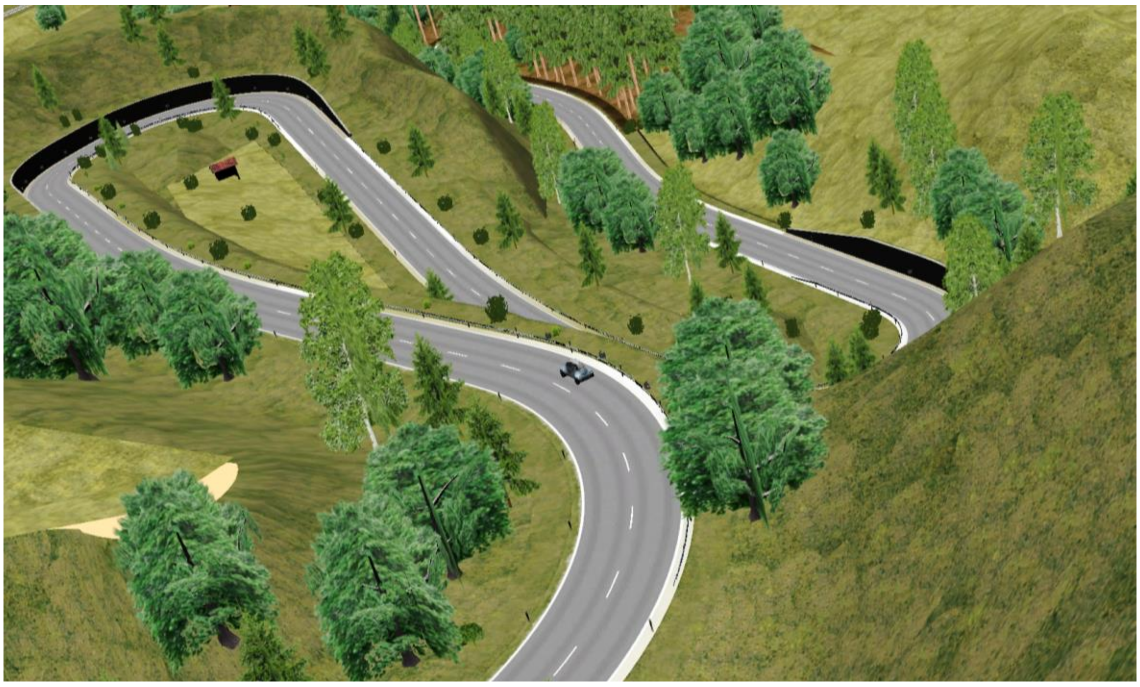


Figure 3.28: ROMO descending Vires 2716 mountain road

The intention here is to evaluate whether the EM framework can optimize the recuperation during downhill driving scenarios. The track itself has several long and smooth curves and in the middle a follow up of two sharp serpentine segments (see Figure 3.28 and Figure 3.29).

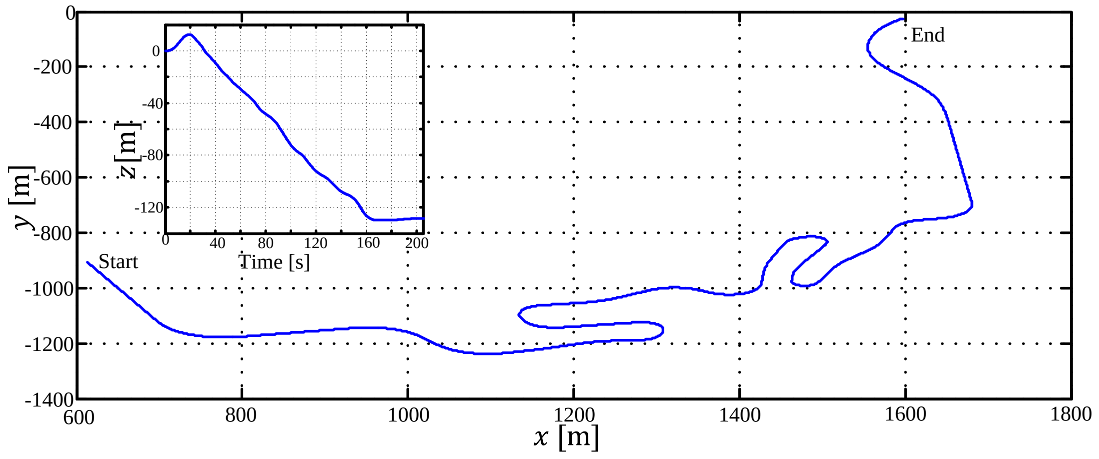


Figure 3.29: Path and height profile of Vires 2716 road segment

In comparison to the previous plot (Figure 3.28) of the demanded trajectories the difference between the both version – optimized and non-optimized – is even smaller.

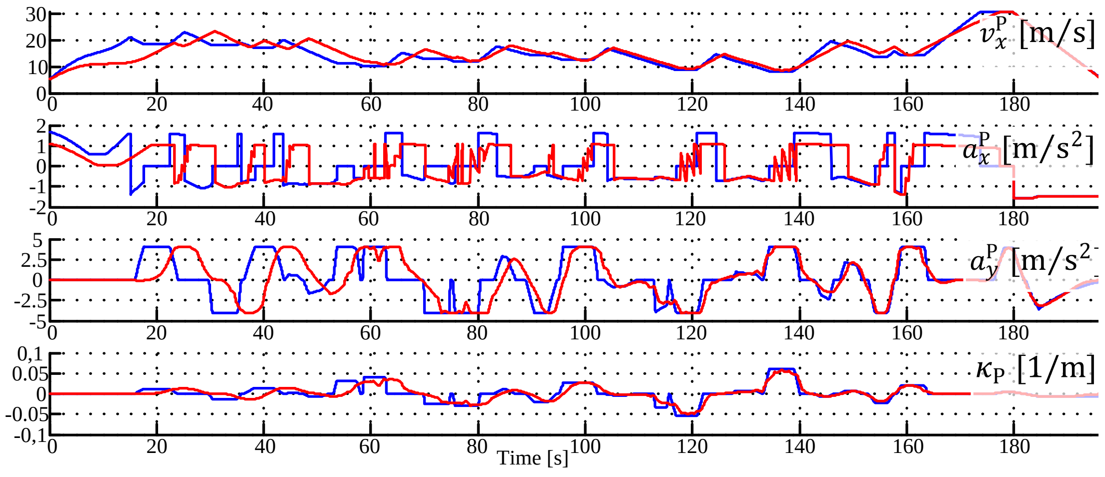


Figure 3.30: Vires 2716 experiment trajectories (blue non-opt. – red opt.)

The curvature as well as the lateral acceleration course is smoothened through the optimization which can be seen as an advantage for the riding comfort for the passenger. Already from the beginning of the course the velocity profiles of both versions are very close to each other whereas the longitudinal acceleration shows a little bit jitter in the two s-curve segments between driving and recuperating demands.

Table 3.7: Energy manager effectiveness assessment – Vires road segment 2716 – 3000 m

Experiment	E_{con}	ΔE	**Difference**
Non opt. – Heuristic	0.0182 kWh	Reference	–
Non-opt. – Physical	0.0242 kWh	+ 0.006 kWh	+32.967 %
Opt. – Heuristic	0.0122 kWh	– 0.006 kWh	−32.970 %
Opt. – Physical	0.0185 kWh	+ 0.0003 kWh	+1.648 %

As can be seen, ROboMObil consumes nearly no energy and therefore the nominal difference between the different control allocations and path variants is small in absolute values, although the relative difference may appear high.

3.7.4 Conclusion of the Energy Manager's Simulative Assessment

The comprehensive conclusion of the simulation case study has shown, that with the here proposed energy manager approach a reduction of the necessary energy for travelling a particular route can be achieved through the optimization based curvature minimization of the path. However, the improvement may vary, depending on the geometry and the slope of the particular road. In the first test scenario the roundabout and other sharp turns led to respectable energy saving. On the other hand, in the second and third scenario the long smooth turns reduced this effect. As already mentioned in Chapter 3.3.1 the quadratic characteristic of the cost criterion (eq. (3.10)) causes that the peak curvature of sharp turns is reduced more than the curvature of wide turns. Moreover, in case of the descending road the saving tended to be marginal. Since the geometric path planning and velocity profile generation should be able to be executed in real-time, the subsequent velocity profile generation leads not to a global optimum of the trajectory $\boldsymbol{\lambda}(s)$. This matter will be further investigated in research activities based on the results of this thesis. The energy optimal control allocation approach as well as the path following controller showed in total good and stable results. The physically motivated cost function did not bring considerable advantage to the overall energy consumption. An approach for further investigations is a model predictive control (MPC) based control allocation approach that does also take the future demanded actuating variable into account (see outlook in Chapter 6.3). Another point worth mentioning is that performance and energy consumption is dependent on the tuning of the three linear controllers of the path tracking controller. For the studies here the parameters were kept constant throughout the case studies, but time-varying gains might reduce the energy consumption.

4. The Model Based Observer Framework

Many demanding mechatronic systems, like the DLR ROboMObil, employ state dependent nonlinear optimization based control (compare control scheme Figure 3.21), which need an accurate knowledge of the system states. Frequently, these states cannot be gathered directly through sensors, since an appropriate measurement principle for the searched quantity is not available (e.g. determination of the state of charge l of a battery) or the sensor is expensive and therefore desirable to be economized (e.g. vehicle over ground velocity sensing $\boldsymbol{v}^C$). The goal of this research is to develop a complete toolchain to assist the control systems engineer in the development of complex, continuous-time formulated prediction models. These should be applied to discrete-time state estimation algorithms and the automatically generated code should be afterwards easily downloadable to an embedded microcontroller target. In this chapter a novel framework for state estimation using detailed multiphysical continuous-time models designed in Modelica [Mod17] is developed. It employs an intelligent separation of the model and the estimation algorithm by utilizing modern computer technologies and recent developments in the Modelica language, which enable automated discretization, integration, and derivative calculation of an object oriented, equation based prediction model. Beyond this a direct usage of the state estimator on a cross platform embedded microcontroller target is also enabled. In addition to the implementation of the well-proven Kalman filter algorithms (for details see Chapter A.3 & A.4), theory extensions for constraints handling and numerically efficient moving horizon estimation are given in Chapter 4.3 & 4.4. In Chapter 5, examples which make use of this framework, where both of them are directly connected to state estimation tasks in the EM framework, are discussed. This chapter is an extended and improved version of the publications [Bre14b] and [Bre11c].

4.1 State of the Art

The textbooks [Gre15], [Sim06], [Hay01] are excellent starting points for the theoretical background and algorithm extensions on Kalman filtering and optimal state estimation. In addition to these textbooks a very comprehensive and condensed chapter of the most relevant algorithms, their derivation and comprehensible relationships for discrete-time state estimation is given in Chapter A of the appendix. For this reason only the most important steps of discrete estimation and their interface to the estimation framework are given in Chapter 4.2.1 to guarantee readability and comprehension.

To the best of the author’s knowledge, the MATLAB toolbox EKF/UKF [Har11] and the ACADO toolkit [Hou10] can be seen as the most advanced publicly available toolboxes for state estimation problems. Both require an analytically derived, by hand or via symbolic preprocessing, (time-discrete) prediction model provided by the user. This is nontrivial, error-prone and time-consuming to utilize for a concrete estimation application with a complex nonlinear model. In contrast, a Modelica simulator e.g. Dymola is able to analyze the structure of an acausal interconnected multiphysical model, to inline the discretization method, and finally to transform it to an order reduced ordinary differential equation representation with analytic derivative calculation features [Mod17]. Another issue is that, in case of the first toolbox, no functionality is provided to directly cross compile, i.e. compile for an execution system different to the one of the development system, and download the result to a real-time target.

In this work a formulation and code generation framework is proposed that makes use of a for estimation problems specific, extended FMU 2.0 co-simulation interface [Mod13], giving the possibility to formulate the prediction model equations in continuous-time by means of the Modelica language.

In [Bon14] an approach is given that makes use of the older FMI 1.0 standard [Mod10], which is not able to directly handle inline integration or model state events. This is similar to the approach in [Bre11c], besides that the authors in [Bon14] use Python with PyFMI [Mab17] and not a Modelica environment, which facilitates the integration of the observer in combination with model based nonlinear inverse controllers [Thu05] or the Modelica Synchronous Control System Library [Ott12] in a single development environment.

4.2 Design of a Modelica Kalman Filter Estimation Framework

In this chapter a method is developed to automatically generate nonlinear state estimators based on continuous-time Modelica models. The approach is based on an extended FMI 2.0 co-simulation interface [Mod13] that interacts with the state estimation algorithms implemented in the DLR Kalman Filter Library [Bre14b].

With the raise of computational power in the last decades the possibilities to implement complex control strategies in real world applications has been enhanced tremendously. For most of them a good knowledge of the actual states is necessary. Often these are not directly measurable due to cost limitations or missing sensors (for example, it is not practical to measure in-tire forces in an automotive application). In the ITEA2 project MODRIO [Ite12] one aim has been to develop state estimation technologies for plants that use the knowledge of complex models of the controlled system itself. These models are often designed, parameterized and optimized as multi-domain models in Modelica [Mod17]. To reuse these models for estimation and control purposes the functional mockup interface [Mod13] turns out to be very helpful. At the beginning of this Ph.D. project a first concept for the state estimation framework was developed using FMI 1.0 for model exchange [Mod10]. It was implemented by handwritten discretization and integration algorithms and Modelica function pointers to separate the prediction model from the observer algorithms with the aim to create an easy to reconfigure framework for state estimation purposes [Bre11c]. This approach had several limitations and difficulties (e.g. no event handling) and therefore an extended version based on the FMI 2.0 co-simulation interface [Mod13] has been developed afterwards.

In Chapter 4.2.1 a brief introduction to the most import estimation algorithms, the extended Kalman filter and unscented Kalman filter, are given and the steps are identified where the model evaluations are necessary by means of the FMI. For time-discrete Kalman filter theory details, derivative-free methods and numerical robust filter design using matrix decomposition and propagation, please inspect Chapter A in the appendix.

4.2.1 Model Evaluations in State Estimation Algorithms

For many control system tasks the plant model to be used in state estimation is naturally described as a nonlinear continuous-time state-space system:

$$\begin{aligned} &\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}), \\ &\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}), \\ &\boldsymbol{u}(t) \in \mathbb{R}^{n_u}, \boldsymbol{x}(t) \in \mathbb{R}^{n_x}, \boldsymbol{y}(t) \in \mathbb{R}^{n_y}, t \in \mathbb{R} \end{aligned} \tag{4.1}$$

Where t is the time, $\boldsymbol{u}(t)$ is the vector of inputs, $\boldsymbol{x}(t)$ is the vector of states and $\boldsymbol{y}(t)$ is the vector of outputs. Besides of manual, time consuming and error prone derivation of this representation, in this work the plant models are defined in Modelica and exported as a functional mockup unit (FMU) [Mod13] using the Modelica simulator Dymola. Note that all the research results presented here are also valid if FMUs are generated by other tools and/or non-Modelica environments, as long as the FMU supports the extended FMI 2.0 co-simulation interface according to Table 4.3.

In a sampled data system (e.g. a microcontroller) the continuous-time model representation in eq. (4.1) cannot be used directly. Instead a time-discrete representation is needed and therefore the time-discrete transformation of eq. (4.1) with additive Gaussian noise is used in the sequel:

$$\begin{aligned} \boldsymbol{x}_k &= \boldsymbol{f}_{k|k-1}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) + \boldsymbol{w}_{k-1}, \\ \boldsymbol{y}_k &= \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{v}_k, \\ &\boldsymbol{w}_k \sim N(0, \boldsymbol{Q}_k), \\ &\boldsymbol{v}_k \sim N(0, \boldsymbol{R}_k). \end{aligned} \tag{4.2}$$

Here t_k is the k-th sample time instant of a periodically sampled data system, $\boldsymbol{u}_k = \boldsymbol{u}(t_k)$, $\boldsymbol{x}_k = \boldsymbol{x}(t_k)$, $\boldsymbol{y}_k = \boldsymbol{y}(t_k)$. The vectors $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ represent zero biased Gaussian white noise. The following eq. (4.3) describes the discrete-time integration rule, like a Runge-Kutta or Euler integration method, and therefore must be integrated into the framework.

$$\boldsymbol{f}_{k|k-1} = \boldsymbol{x}_{k-1} + \int_{t_{k-1}}^{t_k} \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1})\, \mathrm{d}t. \tag{4.3}$$

The Extended Kalman Filter Calculation Steps

In this section the steps of Kalman filter based state estimation are briefly summarized and the prediction step of the extended Kalman filter (EKF) and the unscented Kalman filter (UKF) are examined. At this stage of the estimation algorithm the need for an efficient and reliable way for the prediction simulation model raises tremendously, which can be handled by the proposed approach.

In Figure 4.1 the cycle flow diagram of a recursive Kalman filter algorithm is depicted. The filter is initialized with the initial state vector guess $\hat{\boldsymbol{x}}_0^+$ and the initial guess of the state covariance matrix $\boldsymbol{P}_0^+$. These can be seen as a stochastic expectation for the trust in the first guess of the estimation procedure.

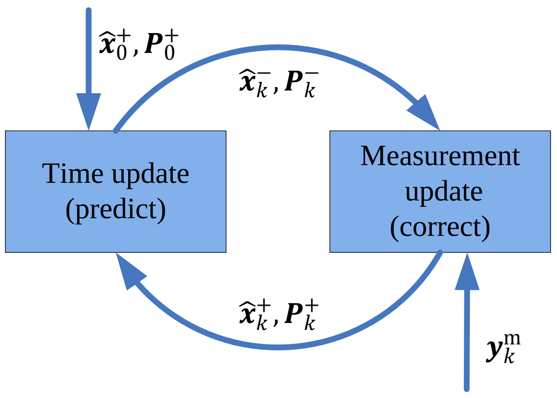

$\hat{\boldsymbol{x}}_0^+, \boldsymbol{P}_0^+$
$\hat{\boldsymbol{x}}_k^-, \boldsymbol{P}_k^-$
Time update (predict)
Measurement update (correct)
$\hat{\boldsymbol{x}}_k^+, \boldsymbol{P}_k^+$
$\boldsymbol{y}_k^{\mathrm{m}}$

Figure 4.1: Principle of a Kalman filter based estimation algorithm

After the initialization process the cycle of the two calculation steps (predict) and (correct) begins and is executed with a predetermined static sample time T_s. The additive Gaussian noise assumption in eq. (4.2) is handled by the user defined covariance matrices $\boldsymbol{Q}, \boldsymbol{R}$. These enable the user to tune the filter to the specific task requirements (e.g. trust in the accuracy of the measurements $\boldsymbol{y}_k^{\mathrm{m}}$). For nonlinear model state estimation the widely used EKF algorithm is given as pseudo code in Table 4.1. The operand $\mathrm{E}(\cdot)$ calculates the expectation value of a random variable [Sim06].

Table 4.1: The extended Kalman filter algorithm in context with FMU evaluations

Initialization:

$$\hat{\boldsymbol{x}}_0^+ = \mathrm{E}(\boldsymbol{x}_0)$$
$$\boldsymbol{P}_0^+ = \mathrm{E}\left((\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)^T\right)$$

for $k = 1,2,\dots$ $(k \in \mathbb{N}^+)$:

Predict:

$$\hat{\boldsymbol{x}}_k^- = \boldsymbol{f}_{k|k-1}(\hat{\boldsymbol{x}}_{k-1}^+, \boldsymbol{u}_{k-1})$$
$$\boldsymbol{P}_k^- = \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}^+\boldsymbol{F}_{k-1}^T + \boldsymbol{Q}$$
$$\text{with } \boldsymbol{F}_{k-1} = e^{\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\Big|_{\hat{\boldsymbol{x}}_{k-1}^+} \cdot T_s\right)}$$

Correct:

$$\boldsymbol{K}_k = \boldsymbol{P}_k^-\boldsymbol{H}_k^T \cdot \left(\boldsymbol{H}_k\boldsymbol{P}_k^-\boldsymbol{H}_k^T + \boldsymbol{R}\right)^{-1}$$
$$\text{with } \boldsymbol{H}_k = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\Big|_{\hat{\boldsymbol{x}}_k^-}$$
$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k \cdot (\boldsymbol{y}_k^{\mathrm{m}} - \boldsymbol{h}(\hat{\boldsymbol{x}}_k^-))$$
$$\boldsymbol{P}_k^+ = (\boldsymbol{I} - \boldsymbol{K}_K \cdot \boldsymbol{H}_k) \cdot \boldsymbol{P}_k^-$$

The red marked sections – this holds for Chapter 4 & 5 – indicate where the evaluation of the underlying system model equations, compare eq. (4.2), are necessary. The calculation of $\hat{\boldsymbol{x}}_k^-$ is performed by integrating the prediction model eq. (4.1) from t_{k-1} to t_k. $\boldsymbol{F}_{k-1}$ is the state-transitions matrix of $\boldsymbol{f}$ with respect to $\boldsymbol{x}$ at $\hat{\boldsymbol{x}}_{k-1}^{+}$ and $\boldsymbol{H}_k$ is the partial derivative matrix of $\boldsymbol{h}$ with respect to $\boldsymbol{x}$ at $\hat{\boldsymbol{x}}_k^{-}$. The Jacobians $\boldsymbol{J}_{k-1}$ and $\boldsymbol{H}_k$ must either be provided directly (according to the FMI 2.0 specification, this feature is optional), or they can be determined numerically, for example with a forward difference quotient – note ϵ is the machine precision of the particular computer architecture and n_x is the number of system states:

$$\begin{aligned}&\text{for } i = 1,2,\dots,n_x;\ \Delta \cong \sqrt{\epsilon}\\ &(\boldsymbol{J}_{k-1})_i = \frac{\boldsymbol{f}(\hat{\boldsymbol{x}}_{k-1} + \Delta\, \boldsymbol{e}_i, \boldsymbol{u}_{k-1}) - \boldsymbol{f}(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_{k-1})}{\Delta}\end{aligned} \tag{4.4}$$

The UKF Sigma Point Prediction Step

The so called sigma point transformation (SPT) is based on the idea that it is easier to approximate a Gaussian distribution, than it is to approximate an arbitrary nonlinear function or transformation; see [Jul04], [Sim06], and Chapter A.4. The parts of the UKF algorithm, in which model evaluations are necessary, are given in Table 4.2. The selection of the sigma points in matrix $\boldsymbol{X}$ is performed via a static scaling factor $\gamma(n, \alpha, \kappa)$ and the matrix square-root of the a posteriori covariance matrix. The number of states is denoted by $n = n_x$, α is the spread around the last state value $\hat{\boldsymbol{x}}_{k-1}^{+}$ and κ is a parameter for the stochastic distribution assumption. In total $2n + 1$ points must be created and then used as initial values for $2n + 1$ simulations from t_{k-1} to t_k to compute $\boldsymbol{X}_{k|k-1}$.

Table 4.2: The unscented Kalman filter prediction step in context with FMU evaluations

$$\begin{aligned}
\boldsymbol{X}_{k-1} &= \left[\hat{\boldsymbol{x}}_{k-1}, \hat{\boldsymbol{X}}_{k-1} + \gamma\sqrt{\boldsymbol{P}_{k-1}^{+}}, \hat{\boldsymbol{X}}_{k-1} - \gamma\sqrt{\boldsymbol{P}_{k-1}^{+}}\right]\\
\boldsymbol{X}_{k|k-1} &= \boldsymbol{f}_{k|k-1}(\boldsymbol{X}_{k-1}, \boldsymbol{u}_{k-1})\\
\hat{\boldsymbol{x}}_k^{-} &= \sum_{i=0}^{2\cdot n} \boldsymbol{w}_i^{\mathrm{m}} \cdot \boldsymbol{X}_{k|k-1,i}\\
\boldsymbol{P}_k^{-} &= \sum_{i=0}^{2\cdot n} \boldsymbol{w}_i^{\mathrm{c}} \cdot \left(\boldsymbol{X}_{k|k-1,i} - \hat{\boldsymbol{x}}_k^{-}\right)\left(\boldsymbol{X}_{k|k-1,i} - \hat{\boldsymbol{x}}_k^{-}\right)^T + \boldsymbol{Q}\\
\boldsymbol{X}_k' &= \left[\hat{\boldsymbol{x}}_k^{-}, \hat{\boldsymbol{X}}_k^{-} + \gamma\sqrt{\boldsymbol{P}_k^{-}}, \hat{\boldsymbol{X}}_k^{-} - \gamma\sqrt{\boldsymbol{P}_k^{-}}\right]\\
\boldsymbol{Y}_k &= \boldsymbol{h}(\boldsymbol{X}_k')\\
\hat{\boldsymbol{y}}_k^{-} &= \sum_{i=0}^{2\cdot n} \boldsymbol{w}_i^{\mathrm{m}} \cdot \boldsymbol{Y}_{k,i}
\end{aligned}$$

The predicted values $\hat{\boldsymbol{x}}_k^{-}$, $\hat{\boldsymbol{y}}_k^{-}$, $\boldsymbol{P}_k^{-}$ are calculated via weighted sums with the predetermined weights $\boldsymbol{w}_i^{\mathrm{c,m}}(n, \alpha, \kappa)$. It is defined $\hat{\boldsymbol{X}} := [\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}, \dots, \hat{\boldsymbol{x}}] \in \mathbb{R}^{n\times 2n+1}$ and in the notation here a vector depending function (e.g. $\boldsymbol{f}_{k|k-1}$ or $\boldsymbol{h}$) with a matrix argument returns a matrix with columns that are equal to the evaluated columns of the matrix argument.

It can be shown that the nonlinear approximation accuracy of the UKF is at least twice higher as the EKF. This becomes an important feature in case of strong nonlinearities in the prediction model (for a detailed proof see [Hay01] – appendix A).

4.2.2 Modelica Estimation Framework Integration

The goal of the estimation framework is to provide an automated workbench for nonlinear estimation based on the model equations of a Modelica model. In Table 4.3 all needed evaluations of the prediction model (compare eq. (4.1) & eq. (4.2)) for state estimation applications are summarized. A toolchain has to provide these model evaluations. In the right column the name of the Modelica function is listed to trigger the corresponding evaluation in the toolchain proposed by this work, for details see the following explanations.

Table 4.3: Model evaluations by means of the FMI for nonlinear state estimation

Required model evaluation		**Modelica function**
Integration between two sample points:	$\boldsymbol{f}_{k\|k-1} = \boldsymbol{x}_{k-1} + \int_{t_{k-1}}^{t_k} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}_{k-1})\, \mathrm{d}t$	integrator
Derivative evaluation:	$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$	f
Output evaluation:	$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x})$	h
Optional model evaluations		
(if not provided by tool, computed numerically by difference quotients – eq. (4.4))		
State Jacobian matrix:	$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{u})$	fx
Output Jacobian matrix:	$\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}(\boldsymbol{x})$	hx

In the following a novel and automated procedure is introduced for incorporating Modelica based models in nonlinear observer algorithms. The aim is to start from a given (continuous, usually nonlinear) Modelica model and automatically deduce a nonlinear observer for this model in form of a sampled data system. This task cannot be performed directly, because Modelica has no means to discretize a continuous model and to solve this discretized model with a user-defined method (that means integration and model update of the next state according to the observer equations). Note that it is insufficient to simply integrate the nonlinear models from the last to the new sample instant (which could be achieved by using the “mapping” annotation introduced in Modelica 3.1 [Mod17]). Instead, the extended Kalman filter algorithm (compare Table 4.1) requires linearizing the model around $\hat{\boldsymbol{x}}_{k-1}^{+}$ for the state Jacobian matrix and around $\hat{\boldsymbol{x}}_{k}^{-}$ for the output Jacobian matrix. On the contrary, the unscented Kalman filter (compare Table 4.2) requires integrating the model several times with disturbed states from the last to the new sample instant. Therefore, the basic approach is to export the Modelica model in the FMI-format, import it again in Modelica and during import, call the FMI-functions in such a way that the model is discretized and utilized in a nonlinear observer algorithm.

The goal of the FMI technology is to describe input/output blocks of dynamic systems defined by differential, algebraic and discrete equations and to provide an interface to evaluate these equations as needed in different simulation environments, as well as in embedded control systems, with explicit or implicit integrators and fixed or variable step size. The FMI interface consists of a small set of standardized “C-functions” to evaluate the model equations and a XML-file that contains all information that is not needed during execution, such as the variable definitions. Every variable has a handle (a 32 bit Integer) that is used to identify the variable in the C-function calls. The source and/or object code of the C-functions, as well as the XML-file and optionally other files, are stored in a zip-file with the extension “.fmu” for “Functional Mockup Unit”.

Utilizing an FMU in an Estimation Algorithm

In Figure 4.2 the extended FMU 2.0 co-simulation interface embedded in the estimation framework is shown (in the style of [Mod13] – Figure 10). It provides the necessary interfaces for the discrete estimation algorithms to calculate the quantities described in Table 4.3. All non-standard FMU interfaces are indicated by dotted lines. Note that currently this is not standardized but available in a prototype extension for the Modelica Simulator Dymola.

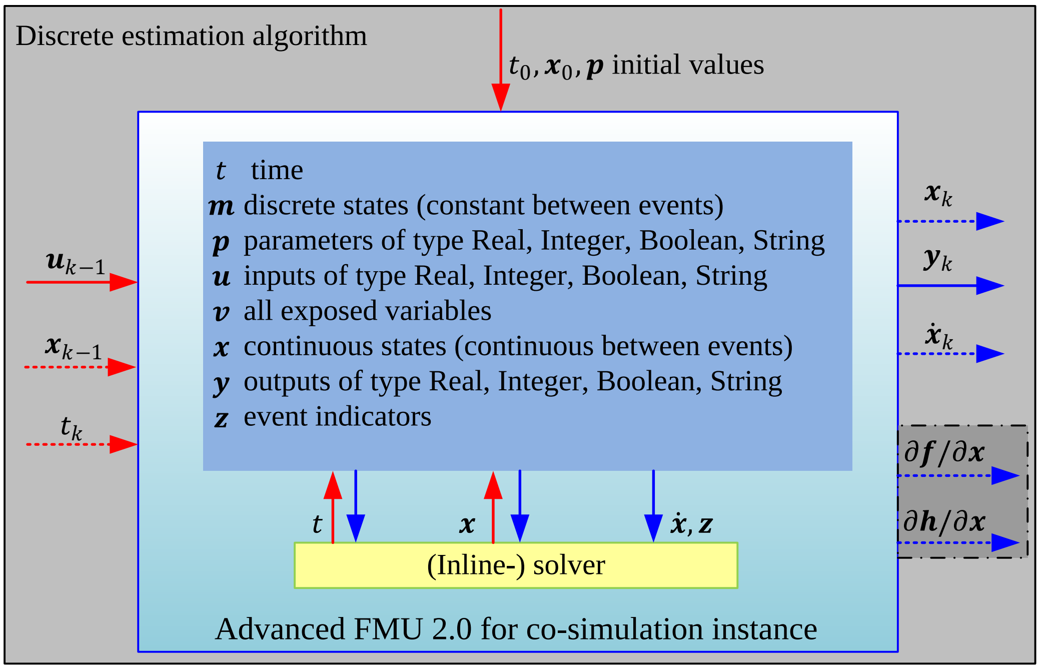

Discrete estimation algorithm
$t_0, \boldsymbol{x}_0, \boldsymbol{p}$ initial values
t time
$\boldsymbol{m}$ discrete states (constant between events)
$\boldsymbol{p}$ parameters of type Real, Integer, Boolean, String
$\boldsymbol{u}$ inputs of type Real, Integer, Boolean, String
$\boldsymbol{v}$ all exposed variables
$\boldsymbol{x}$ continuous states (continuous between events)
$\boldsymbol{y}$ outputs of type Real, Integer, Boolean, String
$\boldsymbol{z}$ event indicators
$\boldsymbol{u}_{k-1}$
$\boldsymbol{x}_{k-1}$
t_k
$\boldsymbol{x}_k$
$\boldsymbol{y}_k$
$\dot{\boldsymbol{x}}_k$
$\partial \boldsymbol{f}/\partial \boldsymbol{x}$
$\partial \boldsymbol{h}/\partial \boldsymbol{x}$
t
$\boldsymbol{x}$
$\dot{\boldsymbol{x}}, \boldsymbol{z}$
(Inline-) solver
Advanced FMU 2.0 for co-simulation instance

Figure 4.2: The extended FMI 2.0 for model co-simulation

Besides the already explained variables $t, \boldsymbol{u}, \boldsymbol{x}, \boldsymbol{y}$ and the set of model parameters $\boldsymbol{p}$ (e.g. the mass of a vehicle) the FMU has the internal variables $\boldsymbol{m}, \boldsymbol{v}, \boldsymbol{z}$, which are now explained briefly. The discrete states $\boldsymbol{m}$ are discrete-time variables with two values: the value of the variable from the previous event instant, and the value of the variable at the actual event instant [Mod13].

Since this work focuses on continuous-time formulated prediction models, these are not used in the prediction model FMUs. The vector $\boldsymbol{v}$ denotes all model variables, i.e. states or sub-system variables, which are defined in the element "<ModelVariables>" of the corresponding XML description. The event indicator vector $\boldsymbol{z}$ contains internal variables such as state events which give the (inline-) solver the information that an event (e.g. the fulfillment of another branch in a "if-else" statement – see for instance the vehicle's standstill condition in Chapter 5.2.2) occurred between the last model evaluation and the current integration step. Internally this is handled by the (inline-) solver to guarantee a correct integration and output of the model variables. More information of this methodology can be found in Chapter 3 of [Mod13].

The overall process of designing a state estimator in Modelica and (cross-) compiling it for embedded targets is illustrated in Figure 4.3. The workflow follows the red arrow and the notation of the dependencies is adopted from UML representation for class diagrams [Mil08].

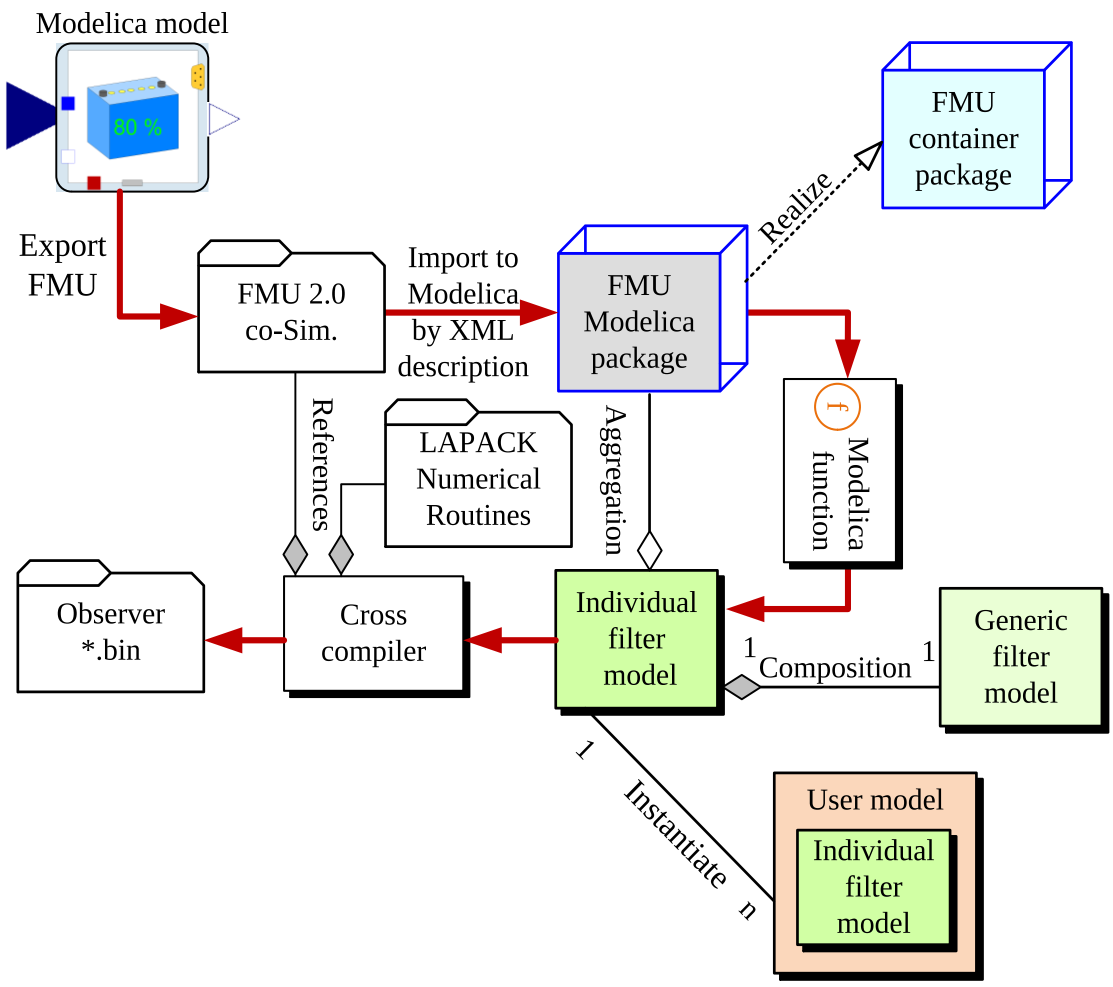


Figure 4.3: Workflow to generate an estimator setup for a (cross-) platform target

The process can be described as follows: an FMU (usually exported from a Modelica model) is imported into the Modelica environment by extending the package "FMU Modelica package". The imported package can be included in an "FMU container package" to collect several FMUs for easy access. For such an FMU package an "Individual (Kalman) filter model" is generated, by a user invoked "Modelica function", which provides variable names on buses and user convenient parameter menus. The algorithmic part of the state estimator is provided by a “Generic filter model” (e.g. an UKF). Finally, the individual filter model can be instantiated in the user’s application model or can be cross compiled for an embedded target platform using the C-code sources from the FMU and a target dependent cross compiled numerical library of LAPACK [And99], which is used for the matrix calculus operations.

For details on the implementation of the DLR Kalman Filter Library in the Modelica language please refer to Chapter B – Implementation of the DLR Kalman Filter Library in the appendix.

Notes on Detectability and Observability

The designed prediction model is normally nonlinear and in this thesis there is no deeper discussion of the determination of the observability of the system. In practical applications it is mostly sufficient if the model is detectable rather than completely observable, i.e. all not observable modes/states are stable. An analysis of these properties can be performed with the DLR Modelica LinearSystems2 library. As a good rule of thumb it is helpful to make a detectability and observability analysis at the nominal operation points or in a grid of potential operation modes (e.g. the vehicle velocity operation range for the later example in Chapter 5.2).

4.3 Kalman Filter Theory Extension with Inequality Constraints

The so far presented estimation algorithms, based on the Kalman approach, are very well suited for a large bandwidth of observer problems. Besides the stochastic assumption of zero mean, Gaussian white noise processes, there are neither system state constraints nor delays in the data acquisition chain considered. In [Sim10] a good and comprehensive overview of existing methods is given for incorporating (linear) constraints to Kalman filtering for linear and nonlinear systems. For linear systems with linear constraints the most relevant methods [Sim10] are briefly summarized in the following table:

Table 4.4: Overview of linear constraint handling methods for linear Kalman filters

Method	**Approach description**
Model reduction	Substitute the equality constraints $\boldsymbol{D} \cdot \hat{\boldsymbol{x}}_k = \boldsymbol{d}$ in the system equations to preserve an unconstrained filtering problem.
Perfect measurements	Extend the measurement vector with the state equality constraints as perfect measurements with zero measurement noise: $$\begin{bmatrix} \boldsymbol{y}_k \\ \boldsymbol{d} \end{bmatrix} = \begin{bmatrix} \boldsymbol{H}_k \\ \boldsymbol{D} \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} \boldsymbol{v}_k \\ 0 \end{bmatrix}$$
Probability density function (PDF) truncation	Truncate the PDF calculated in the Kalman filter to the edges of the constraint surface. The mean of the new PDF is identical to the constraint estimate.

Estimate projection for equality constraints	Project the a posteriori state estimate on the constraint surface: $$\min_{\boldsymbol{x}} \|\boldsymbol{x} - \hat{\boldsymbol{x}}_k^+\|_{\boldsymbol{W}} \quad \text{s.t. } \boldsymbol{D} \cdot \boldsymbol{x} = \boldsymbol{d}$$ The solution of the projected constraint estimation is: $$\bar{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^+ + \boldsymbol{P}_k^+ \boldsymbol{D}^T (\boldsymbol{D}\boldsymbol{P}_k^+ \boldsymbol{D}^T)^{-1} (\boldsymbol{D}\hat{\boldsymbol{x}}_k^+ - \boldsymbol{d})$$
Estimate projection for inequality constraints	Find a solution for the optimization problem: $$\min_{\boldsymbol{x}} \|\boldsymbol{x} - \hat{\boldsymbol{x}}_k^+\|_{\boldsymbol{W}} \quad \text{s.t. } \boldsymbol{D} \cdot \boldsymbol{x} \leq \boldsymbol{d}$$ It can be solved via a quadratic program (QP).
Gain projection	Incorporate the constraints to the original optimization problem formulation of the standard Kalman filter [Sim06]: $$\min_{\boldsymbol{K}} \operatorname{Tr}\big((\boldsymbol{I} - \boldsymbol{K}\boldsymbol{H})\boldsymbol{P}_k^- (\boldsymbol{I} - \boldsymbol{K}\boldsymbol{H})^T + \boldsymbol{K}\boldsymbol{R}\boldsymbol{K}\big) \quad \text{s.t. } \boldsymbol{D} \cdot \hat{\boldsymbol{x}}_{\boldsymbol{k}}^+ = \boldsymbol{d}$$
System projection	Project the process noise covariance $\boldsymbol{Q}$ on a modified covariance $\widetilde{\boldsymbol{Q}}$ by means of a singular value decomposition of $\boldsymbol{D}$ such that $$\boldsymbol{D}\widetilde{\boldsymbol{Q}}\boldsymbol{D}^T = 0 \text{ holds.}$$
Soft constraints	Do only approximately fulfill the constraints; this can be achieved by e.g. adding measurement noise to perfect measurements or adding an additional regulation term to the Kalman filter.
Moving horizon estimation	Reformulate the general Kalman optimization problem eq. (A.36) to a fixed moving estimation window eq. (A.39) with equality and inequality constraints. It can be solved via a QP as explained in Chapter A.5.

Some of these methods are not only applicable for linear constraints. Especially the estimation projection methods can be used for nonlinear constraints through a first or second order Taylor series expansion around the a priori state estimation $\hat{\boldsymbol{x}}_k^-$. For the approach of perfect measurements constraint handling, this linearization is repeatedly applied to the measurements to smoothly improve the estimation.

This thesis focuses on the class of nonlinear inequality constraints for nonlinear state estimation, since for many estimation tasks the control design engineer needs to incorporate inequality constraints to the states, e.g. the slider position state p of an inverted pendulum is limited to the interval $p \in [p_{\min}, p_{\max}]$. Moreover, the constraints should be formulated (non-)linear by means of the Modelica language within the prediction model or in a separate FMU (see Chapter 5.2 for the detailed practical realization). Additionally, by reason of exchangeability and flexibility, the state constraint handling algorithms should be separable from the particular unconstrained Kalman filter algorithm. I.e. the user can configure the estimation task with constraint handling features and then examine different Kalman filters (e.g. EKF or UKF) without the need to adopt the constraints to the particular estimation algorithm. By these reasons it has been decided to extend the a posteriori estimation projection approach with an inequality constraints algorithm in a computational efficient way to guarantee real-time capability. This is explained in the following subchapters of Chapter 4.3. To the best knowledge of the author new methods for nonlinear estimation with nonlinear inequality constraints are given in the next chapters that are different to the algorithms in literature (see Table 4.5).

Table 4.5: Overview of nonlinear constraint handling methods for nonlinear Kalman filters

Method	**Approach description**
Nonlinear soft constraints	Nonlinear soft constraints by means of perfect measurements [Sim10] are shown in the application example of a battery state estimator in Chapter 5.1.
Nonlinear optimization based estimation projection	Project the states on the constraint surface by solving the restrictive optimization problem $$\min_{\boldsymbol{x}} \lVert \boldsymbol{x} - \hat{\boldsymbol{x}}_k^+ \rVert_{\boldsymbol{W}} \quad \text{s.t. } \boldsymbol{d}(\boldsymbol{x}) = \boldsymbol{0}, \boldsymbol{c}(\boldsymbol{x}) \leq \boldsymbol{0}$$ which can be solved by means of a nonlinear interior point solver or a sequential quadratic program (SQP) [Scn11].
Sigma point projection	This is a special approach for UKF algorithms (compare Chapter A.4). The sigma points are projected on the borders of the constraint region and with these projected points the covariance update is calculated [Kan08]. In [Scn11] additionally the sigma point weights are scaled to improve the constraint covariance confidence. Similar methods are the two-step UKF and the unscented recursive nonlinear dynamic data reconciliation (URNDDR) which performs a MHE with a horizon size of one to the a posteriori sigma points [Sim10].
Nonlinear moving horizon estimation	A nonlinear moving horizon estimator with a nonlinear gradient descent optimization which can handle equality and inequality constraints as well as delayed measurements is formulated in Chapter 4.4. Besides this, various other formulations, like the URNDDR, the constrained UKF [Sim10] or the constrained real-time approach [Bog14] using the ACADO toolbox [Hou10] can be found as examples for nonlinear moving horizon estimation.

The following proposed estimation projection methods are appropriate for nonlinear inequality constraints of the form $\boldsymbol{c}(\boldsymbol{x}) \leq \boldsymbol{0}$, with limits to the distinguished approximation. First, in Chapter 4.3.1 an efficient root-finding algorithm determines a linear scaling factor between the vector distance $\overline{\hat{\boldsymbol{x}}_k^+ \, \hat{\boldsymbol{x}}_{k-1}^+}$ to the point where the constraint gets active $c(\boldsymbol{x}) = 0$. Afterwards, two different methods are proposed, one based on statistical linear regression (compare Chapter A.4) and one with a Lagrange multiplier. These methods determine a constraint estimate that lays closer to the border of the constrained surface $c(\boldsymbol{x}) = 0$.

4.3.1 The Formulation of State Constraints $\boldsymbol{c}(\boldsymbol{x}) \leq \mathbf{0}$

In the most common cases there exist a lot of known constraints to estimated states, e.g. a limitation of the position due to physical stops in a mechanical system. These can be formulated as restrictions that fulfill a set of inequalities $\boldsymbol{c}(\boldsymbol{x}) \leq \mathbf{0}$. The main principal of the proposed algorithms is to change the state vector after the correction step $\widehat{\boldsymbol{x}}_k^+$ of the Kalman filter such that all restrictions are fulfilled. The optimization problem is defined as:

$$\begin{gathered} \widehat{\boldsymbol{x}}_k^0 = \underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{x} - \widehat{\boldsymbol{x}}_k^+\| \\ \text{s.t. } \boldsymbol{c}(\boldsymbol{x}) \leq \mathbf{0} \end{gathered} \tag{4.5}$$

It is assumed that the constraints $\boldsymbol{c}(\boldsymbol{x}) \leq \mathbf{0}$ are neither contradictory nor redundant and only one constraint is active at the same time. In many cases this is valid, since the estimation step size is small. Examples are given in Chapter 5.1.5 and Chapter 5.2.3. In case of large step sizes and/or if multiple constraints are likely to be active, it is possible to handle these situations by an SQP formulation (see Table 4.5) or with a moving horizon estimator formulation as proposed in Chapter 4.4.

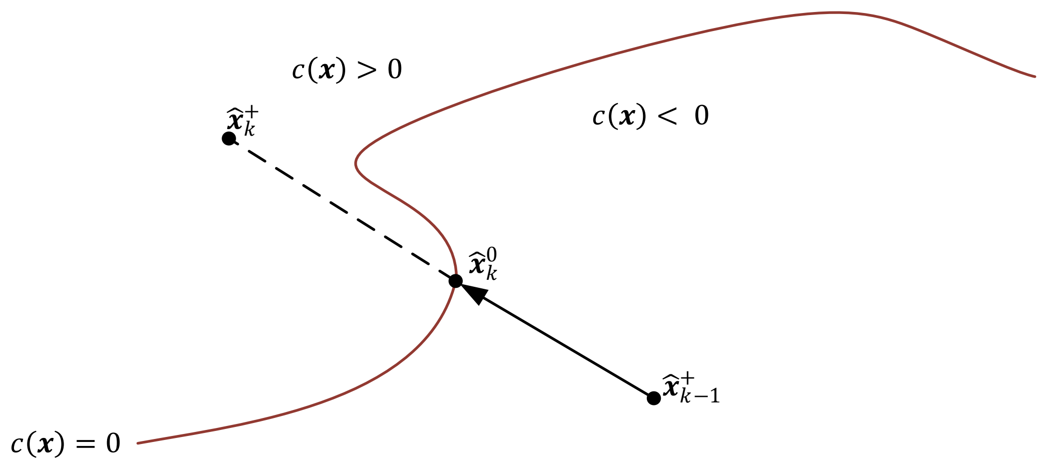

$c(\boldsymbol{x}) > 0$
$\widehat{\boldsymbol{x}}_k^+$
$c(\boldsymbol{x}) < 0$
$\widehat{\boldsymbol{x}}_k^0$
$\widehat{\boldsymbol{x}}_{k-1}^+$
$c(\boldsymbol{x}) = 0$

Figure 4.4: Barrier violation in constraint state estimation

The obvious way to handle the active constraint is to find the point $\widehat{\boldsymbol{x}}_k^0$ in the vector $\overline{\widehat{\boldsymbol{x}}_k^+ \widehat{\boldsymbol{x}}_{k-1}^+}$ where the violated constraint $c(\boldsymbol{x}) > 0$ is no longer fulfilled (see Figure 4.4). This means the root α of $c\big(\boldsymbol{x}(\alpha)\big) = 0$ has to be determined, such that

$$\widehat{\boldsymbol{x}}_k^0 = \widehat{\boldsymbol{x}}_{k-1}^+ + \alpha \cdot (\widehat{\boldsymbol{x}}_k^+ - \widehat{\boldsymbol{x}}_{k-1}^+), \quad \alpha \in [0,1] \tag{4.6}$$

This problem is solved with the derivative free root finding method of Brent [Brn73]. This algorithm is available in the Modelica Standard Library ("Modelica.Math.Nonlinear. solveOneNonlinearEquation"). Once α is known, the Cholesky decomposition $\boldsymbol{CP}_k^0$ of the covariance matrix can be calculated as:

$$\boldsymbol{CP}_k^0 = (1 - \alpha) \cdot \boldsymbol{CP}_k^- + \alpha \cdot \boldsymbol{CP}_k^+ \tag{4.7}$$

The rudimentary way would be to use this solution for the estimation projection, but one can easily see in Figure 4.4 that it is not the closest estimate to the unconstrained solution $\widehat{\boldsymbol{x}}_k^+$. Therefore, below two approaches are discussed that are intended to find a better approximation of the solution.

4.3.2 A Statistical Linear Regression Approach for Unscented Kalman Filters

The first method is based on an advanced system approximation to transfer the nonlinear restricted problem in eq. (4.8) into an equivalent linear formulation. This proposed algorithm makes use of the weighted statistical linear regression (WSLR) method, described in Chapter A.4, applicable for inequality constraints in unscented Kalman filters (UKF). To solve the optimization objective eq. (4.5) with this method two assumptions have to hold. First, a state vector $\widehat{\boldsymbol{x}}_k^0$ can be found such that the active constraint $c(\widehat{\boldsymbol{x}}_k^0) = 0$ is exactly fulfilled (compare eq. (4.6)) and therefore the objectives rewrites as follows:

$$\begin{gathered} \widehat{\boldsymbol{x}}_k^{\mathrm{SLR}} = \underset{x}{\operatorname{argmin}} \| \boldsymbol{x} - \widehat{\boldsymbol{x}}_k^+ \|_{\boldsymbol{W}}^2 \\ \mathrm{s.t.}\ c(\boldsymbol{x}) = 0 \end{gathered} \tag{4.8}$$

Second, it is possible to linearize $c(\boldsymbol{x})$ in an efficient way. The linearization of the equality constraint could be generally performed e.g. via a Taylor series approximation [Sim10]. A more reliable way is to linearize the constraint via the WSLR method:

$$\begin{aligned} y &= c(\boldsymbol{x}) \\ \hat{y} &= \boldsymbol{C}\boldsymbol{x} + b \\ \epsilon &= y - \underbrace{\boldsymbol{C}\boldsymbol{x} + b}_{\hat{y}} \end{aligned} \tag{4.9}$$

The performance gain is graphically analyzed in App. Figure A.4. With the results of the derivation sketched in eq. (A.28) through eq. (A.31) and the use of the Cholesky factors of $\boldsymbol{P}_k^0$, $\boldsymbol{C}$ can be directly calculated:

$$\begin{gathered} \boldsymbol{C} = \boldsymbol{P}_{\boldsymbol{x}\boldsymbol{c}_0} \cdot {\boldsymbol{P}_k^0}^{-1} = \boldsymbol{P}_{\boldsymbol{x}\boldsymbol{c}_0} \cdot C{\boldsymbol{P}_k^0}^{-T} \cdot C{\boldsymbol{P}_k^0}^{-1} \\ \text{with } \boldsymbol{P}_{\boldsymbol{x}\boldsymbol{c}_0} = \sum_{i=0}^{2\cdot n} \boldsymbol{w}_i^{\mathrm{c}} \cdot \left[\boldsymbol{X}_{i,k}^0 - \widehat{\boldsymbol{x}}_k^0\right]\left[c(\boldsymbol{X}_{i,k}^0) - 0\right]^T \\ \text{and } \boldsymbol{X}_k^0 = \left[\widehat{\boldsymbol{x}}_k^0, \widehat{\boldsymbol{x}}_k^0 + \gamma \cdot C\boldsymbol{P}_k^0, \widehat{\boldsymbol{x}}_k^0 - \gamma \cdot C\boldsymbol{P}_k^0\right] \end{gathered} \tag{4.10}$$

In eq. (4.10) γ denotes the weighting of the covariance and can be calculated according to App. Table A.5 With the WSLR linearization method, eq. (4.8) can be approximated by a linear equality constrained least squares problem

$$\begin{gathered}\boldsymbol{W} = \boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{P}_k^{0^{-1}} \rightarrow \boldsymbol{R} = \boldsymbol{CP}_k^{0^{-1}} \\ \hat{\boldsymbol{x}}_k^{\mathrm{SLR}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\| \underbrace{\boldsymbol{R}}_{\boldsymbol{A}}\boldsymbol{x} - \underbrace{\boldsymbol{R}\hat{\boldsymbol{x}}_k^+}_{\boldsymbol{a}} \right\| \\ \text{s.t. } \underbrace{\boldsymbol{C}}_{\boldsymbol{B}} \cdot \boldsymbol{x} = \underbrace{\boldsymbol{C}\,\hat{\boldsymbol{x}}_k^0}_{b}\end{gathered} \tag{4.11}$$

that can be solved for example with DGGLSE [And99]. As the result a new constraint estimate $\hat{\boldsymbol{x}}_k^{\mathrm{SLR}}$ (see Figure 4.5) is obtained that is closer to the unconstrained estimate $\hat{\boldsymbol{x}}_k^+$, than the constrained states $\hat{\boldsymbol{x}}_k^0$ computed with the root finding algorithm presented in Chapter 4.3.1.

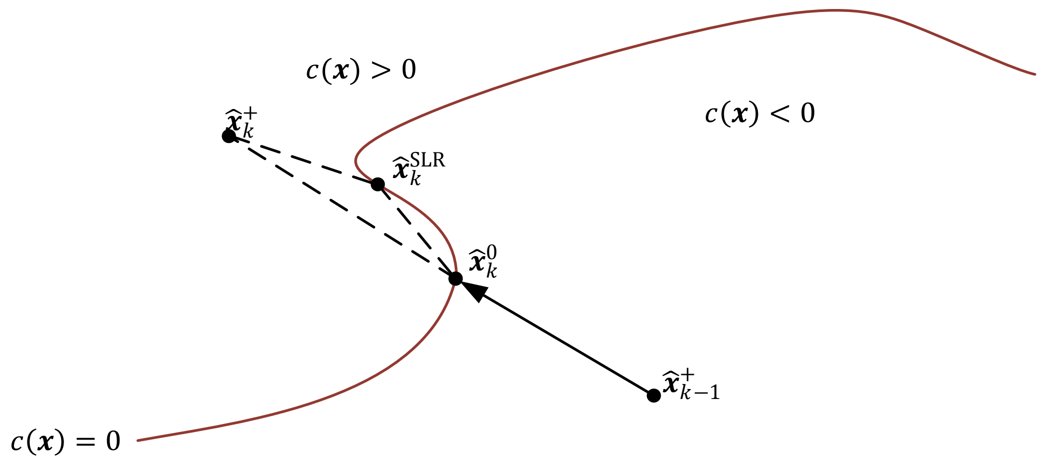

$c(\boldsymbol{x}) > 0$
$c(\boldsymbol{x}) < 0$
$\hat{\boldsymbol{x}}_k^+$
$\hat{\boldsymbol{x}}_k^{\mathrm{SLR}}$
$\hat{\boldsymbol{x}}_k^0$
$\hat{\boldsymbol{x}}_{k-1}^+$
$c(\boldsymbol{x}) = 0$

Figure 4.5: Principle of the SLR approach for state constraint handling

The complete algorithm, which exploits the structure of the given matrices, writes as follows:

Table 4.6: Algorithm for estimation projection based on the SLR technique

1. Find $\alpha \in [0,1]$ such that $c(\hat{\boldsymbol{x}}_k^0) = 0$ with:

$$\begin{gathered}\hat{\boldsymbol{x}}_k^0 = \hat{\boldsymbol{x}}_{k-1}^+ + \alpha \cdot (\hat{\boldsymbol{x}}_k^+ - \hat{\boldsymbol{x}}_{k-1}^+) \\ \boldsymbol{CP}_k^0 = (1-\alpha) \cdot \boldsymbol{CP}_k^- + \alpha \cdot \boldsymbol{CP}_k^+\end{gathered}$$

3. Calculate $\boldsymbol{C}$:

$$\boldsymbol{C} = \boldsymbol{P}_{\boldsymbol{x}c_0} \cdot \boldsymbol{P}_k^{0^{-1}} = \boldsymbol{P}_{\boldsymbol{x}c_0} \cdot \boldsymbol{CP}_k^{0^{-T}} \cdot \boldsymbol{CP}_k^{0^{-1}}$$

$$\text{with } \boldsymbol{P}_{\boldsymbol{x}c_0} = \sum_{i=0}^{2\cdot n} \boldsymbol{w}_i^{\mathrm{c}} \cdot [\boldsymbol{X}_{i,k}^0 - \hat{\boldsymbol{x}}_k^0][c(\boldsymbol{X}_{i,k}^0) - 0]^T$$

4. Least squares minimization $\rightarrow \hat{\boldsymbol{x}}_k^{\mathrm{SLR}}$:

$$\begin{gathered}\boldsymbol{W} = \boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{P}_k^{0^{-1}} \rightarrow \boldsymbol{R} = \boldsymbol{CP}_k^{0^{-1}} \\ \hat{\boldsymbol{x}}_k^{\mathrm{SLR}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{R}\boldsymbol{x} - \boldsymbol{R}\hat{\boldsymbol{x}}_k^+\| \\ \text{s.t. } \boldsymbol{C} \cdot \boldsymbol{x} = \boldsymbol{C}\hat{\boldsymbol{x}}_k^0\end{gathered}$$

In [Scn09] another methodology for handling inequality constraints has been developed by means of the WSLR. The difference to the here discussed approach is, that the root finding is performed by a computational costly bisection method. Additionally, a direct projection method, to eliminate the equality constraints from the optimization objective, is applied.

4.3.3 A General Approach – The Simplified Newton Descent Search

As a third approach a more general method for incorporating inequality constraints is presented. The WSLR based algorithm of Chapter 4.3.2 works fine with the SR-UKF as well as the SR-EKF algorithm, because in these cases the algorithm can directly use the Cholesky factorization of the a priori covariance matrix. For other algorithms, that do not rely on square-root filtering (compare Chapter A.4.2), a computational costly Cholesky decomposition of the covariance matrix would have to be computed in every time instance when a constraint gets active. The here proposed algorithm is based on a simplified Newton descent search (i.e. without considering the second order derivative), that projects the a posteriori estimation on the constraint surface. This opens up a more general use. The optimization objective is formulated as follows:

$$\begin{aligned} \hat{\boldsymbol{x}}_k^{\mathrm{P}} &= \underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{x} - \hat{\boldsymbol{x}}_k^{+}\| \\ &\mathrm{s.t.}\;\; c(\boldsymbol{x}) = 0 \end{aligned} \tag{4.12}$$

The idea behind this algorithm is to perform a descent search along the gradient $\boldsymbol{\nabla} c(\hat{\boldsymbol{x}}_k^{+})$ calculated at the point $\hat{\boldsymbol{x}}_k^{+}$ until the constraint equation $c\left(\hat{\boldsymbol{x}}_k^{\mathrm{P}}\right) \leq 0$ holds again. Graphically this can be interpreted as shown in Figure 4.6

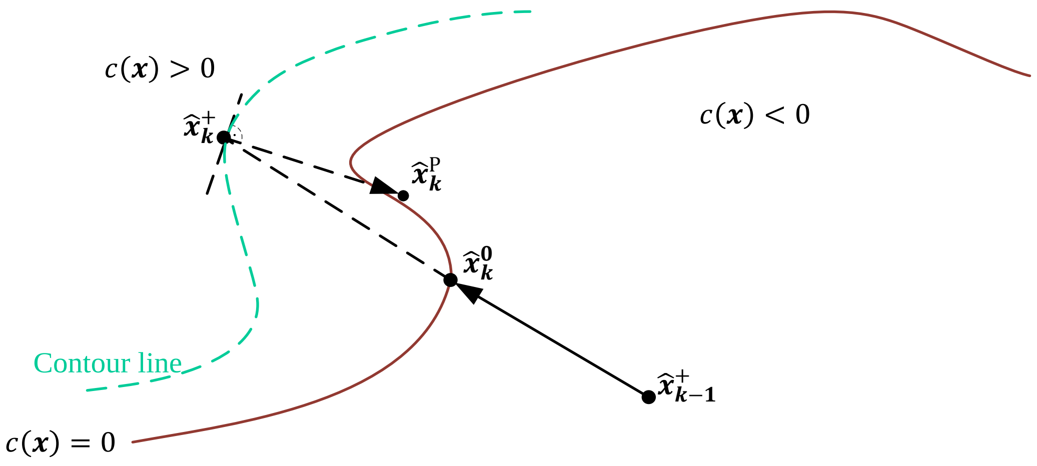

$c(\boldsymbol{x}) > 0$
$c(\boldsymbol{x}) < 0$
$\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{+}$
$\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{\mathrm{P}}$
$\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{\boldsymbol{0}}$
$\hat{\boldsymbol{x}}_{\boldsymbol{k-1}}^{+}$
Contour line
$c(\boldsymbol{x}) = 0$

Figure 4.6: Principle of the simplified Newton descent search for state constraint handling

This optimization task can be formulated with the method of Lagrange multipliers [Boy04]. Eq. (4.13) denotes the above formulated optimization objective in a general description:

$$\min f(\boldsymbol{x}) \quad \mathrm{s.t.}\;\; g(\boldsymbol{x}) = d \tag{4.13}$$

It is assumed that both functions f, g have continuous first partial derivatives around the solution. With the introduction of the Lagrange multiplier μ the optimization task can be reformulated as an unconstrained minimization problem:

$$\min \boldsymbol{\mathcal{L}}(\boldsymbol{x}, \mu) = f(\boldsymbol{x}) - \mu \cdot (g(\boldsymbol{x}) - d) \tag{4.14}$$

Therefore, the optimal solution of this unrestricted minimization problem is denoted as follows:

$$\boldsymbol{\nabla}_{x,\mu} \boldsymbol{\mathcal{L}}(\boldsymbol{x}, \mu) = 0 \tag{4.15}$$

This approach is now applied to the estimation projection optimization problem eq. (4.12) and the searched restricted estimate $\widehat{\boldsymbol{x}}_k^{\mathrm{P}}$ is calculated using the Lagrange multiplier μ and the gradient of the active constraint $c(\boldsymbol{x})$:

$$\begin{aligned} \widehat{\boldsymbol{x}}_k^{P} &= \widehat{\boldsymbol{x}}_k^{+} + \mu \cdot \boldsymbol{\nabla} c(\boldsymbol{x}) \quad \text{s.t.}\ c(\boldsymbol{x}) = 0 \\ c\big(\widehat{\boldsymbol{x}}_k^{+} + \mu \cdot \boldsymbol{\nabla} c(\boldsymbol{x})\big) &= 0 \end{aligned} \tag{4.16}$$

To fulfill eq. (4.15) the approach is formulated with a simplified Newton descent search algorithm by means of a scalar zero search $\mathrm{F}(\mu) = 0$ with respect to the Lagrange variable μ:

$$\begin{aligned} &\boldsymbol{\nabla} c(\boldsymbol{x}) \approx \boldsymbol{\nabla} c(\widehat{\boldsymbol{x}}_k^{+}) \\ &\xrightarrow{\text{yields}} \mathrm{F}(\mu) = c\big(\widehat{\boldsymbol{x}}_k^{+} + \mu \cdot \boldsymbol{\nabla} c(\widehat{\boldsymbol{x}}_k^{+})\big) \overset{!}{=} 0 \\ &\frac{\partial \mathrm{F}}{\partial \mu} = \boldsymbol{\nabla} c(\widehat{\boldsymbol{x}}_k^{+} + \mu \cdot \boldsymbol{\nabla} \boldsymbol{c})^T \cdot \boldsymbol{\nabla} c(\widehat{\boldsymbol{x}}_k^{+}) \\ &\left.\frac{\partial \mathrm{F}}{\partial \mu}\right|_{\mu=0} = \boldsymbol{\nabla} c(\widehat{\boldsymbol{x}}_k^{+})^T \boldsymbol{\nabla} \mathrm{c}(\widehat{\boldsymbol{x}}_k^{+}) \end{aligned} \tag{4.17}$$

This can be implemented as an iterative search algorithm, which determines μ such that the condition $\mathrm{F}(\mu) = 0$ holds within a predefined maximum number of calculation steps. The pseudo code for the algorithm is given in Table 4.7.

Table 4.7: Simplified Newton descent search algorithm for constrained estimation projection

```
Initialization:
        $\mu_o = 0$
        $\alpha = \big(\boldsymbol{\nabla} c(\widehat{\boldsymbol{x}}_k^{+})^T \cdot \boldsymbol{\nabla} c(\widehat{\boldsymbol{x}}_k^{+})\big)^{-1}$
        $k = -1$
While $\left|\alpha \cdot \mathrm{c}\big(\widehat{\boldsymbol{x}}_k^{+} + \mu_{k+1} \cdot \boldsymbol{\nabla} \mathrm{c}(\widehat{\boldsymbol{x}}_k^{+})\big)\right| \geq \epsilon_{\mathrm{Newton}}$:
        $k = k + 1$
        $\Delta\mu_{k+1} = -\alpha \cdot c\big(\widehat{\boldsymbol{x}}_k^{+} + \mu_k \cdot \boldsymbol{\nabla} c(\widehat{\boldsymbol{x}}_k^{+})\big)$
        $\mu_{k+1} = \mu_k + \Delta\mu_{k+1}$
Solution:
        $\widehat{\boldsymbol{x}}_k^{\mathrm{P}} = \widehat{\boldsymbol{x}}_k^{+} + \mu_{k+1} \cdot \boldsymbol{\nabla} c(\widehat{\boldsymbol{x}}_k^{+})$
```

4.4 Real-time Nonlinear Moving Horizon Estimation

In this section the recursive one-step Kalman filter approaches sketched in Chapters 4.2.1, A.3, and A.4 are extended to a nonlinear moving horizon estimator (MHE), which utilizes multiple past measurements to estimate state at the current time instance. In addition to state constraints (cf. Chapter 4.4.1), delayed measurements (cf. Chapter 4.4.2) can be directly incorporated in the problem formulation. As an outlook for ongoing and future observer framework extensions, the connection of moving horizon estimation, and nonlinear model predictive control (NMPC) is discussed in Chapter 6.4.2.

Referring to [Sim10], the approach of moving horizon estimation is the reformulation of the general optimization objective of the Kalman filter theory, also known as the full-information filter. In the nonlinear case the minimization problem can be written as follows:

$$\min_{\boldsymbol{\xi}_k} \parallel \boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0 \parallel^2_{\boldsymbol{I}_0^+} + \sum_{i=1}^{N} \parallel \boldsymbol{y}_i^{\mathrm{m}} - \boldsymbol{h}(\boldsymbol{x}_i) \parallel^2_{\boldsymbol{R}^{-1}} + \sum_{i=0}^{N-1} \parallel \boldsymbol{x}_{i+1} - \boldsymbol{f}(\boldsymbol{x}_i) \parallel^2_{\boldsymbol{Q}^{-1}} \qquad (4.18)$$
$$\boldsymbol{\xi}_k = (\boldsymbol{x}_0^T, \boldsymbol{x}_1^T, \dots, \boldsymbol{x}_N^T)^T$$

The discussed recursive Kalman filter algorithms in Chapter 4.2.1 are special cases of this optimization objective in the case that only the measurements at the current time instance t_k are available. Since the dimension N of the optimization problem would grow tremendously with proceeding time, in MHE theory the time span is limited to a predefined length of previous time instances and shifted in every sample step.

As mentioned above the moving horizon estimator (MHE) is very closely connected to model predictive control (MPC) – this matter of fact will be discussed further in Chapter 6.4.2. Instead of predicting future control inputs, a sliding window with M steps back from the actual time instance t_k, is considered to smooth the state estimation. In every sample step T_s this window is shifted one-step ahead – in Figure 4.7 this is illustrated for one measurement variable y^{m}. At the filter initialization t_0 only one measurement is available and therefore the measurement storage must be filled with $r < M$ steps before the window is starting to move. Afterwards the window length is kept constant and all past M measurements are taken into account (the window with M measurements is colored in green in the subsequent figures). In other words, the MHE can be seen as a real-time calculable approximation of the full-information filter.

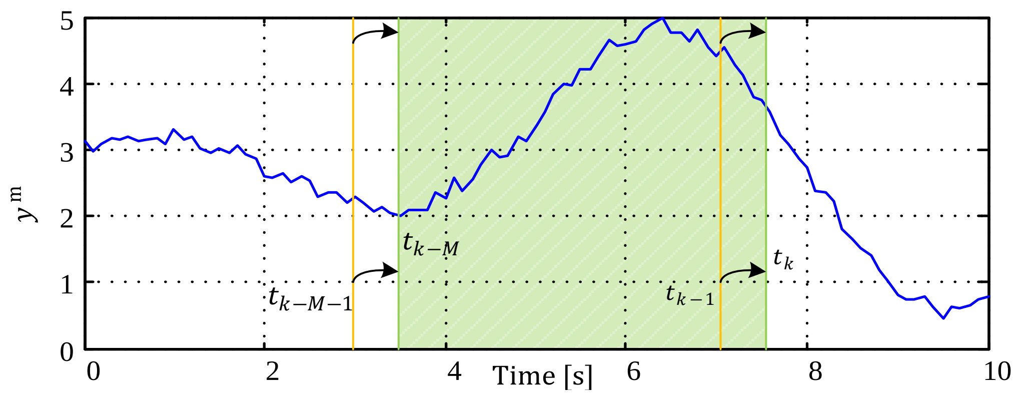


Figure 4.7: Schematic representation of a moving measurement window

In this way the estimate gets more robust against external disturbances, delayed measurements can be incorporated (see Chapter 4.4.2) and also constraints can be imposed directly [Sim10]. The linear case for MHE state estimation and its formulation as a quadratic optimization problem is stated in Chapter A.5.

By means of the proposed estimation framework presented in Chapter 4.2.2 it is also possible to incorporate complex nonlinear Modelica based prediction models to moving horizon estimation.

In the context of toolchains for observer code generation (compare Chapter 4.1) and nonlinear constraint MHE, different research studies were recently published that utilize the ACADO toolbox for embedded [Vuk15] and real-time moving horizon estimation [Bog14], [Fer12]. They make use of the real-time iteration (RTI) scheme originally developed in [Die01] and later transferred to the MHE approach in [Kue11]. The basic strategy is to discretize the estimation problem with a multiple shooting discretization using numerical integration. Then the main idea of the RTI scheme is to use the shifted state variables of the previous optimization run as the new linearization point and perform only one SQP step per sample time [Vuk15].

In this work the problem formulation in eq. (4.19) is extended for nonlinear systems incorporating linear state constraints, tailored to meet real-time application restrictions by means of a nonlinear gradient descent search. Details on the reasons for this approach are given in the following chapter.

$$\begin{aligned} &\min_{\boldsymbol{\xi}_k} \; g\left(\boldsymbol{\xi}_k = \left(\boldsymbol{x}_{k-M}^T, \boldsymbol{x}_{k-M+1}^T, \dots, \boldsymbol{x}_k^T\right)^T\right) \\ &\text{s.t.} \; \boldsymbol{A} \cdot \boldsymbol{\xi}_k = \boldsymbol{b} \\ &\qquad \boldsymbol{C} \cdot \boldsymbol{\xi}_k \leq \boldsymbol{d} \end{aligned} \tag{4.19}$$

$$\begin{aligned} g(\boldsymbol{\xi}_k) = &\parallel \boldsymbol{x}_{k-M} - \hat{\boldsymbol{x}}_{k-M}^{+} \parallel_{\boldsymbol{I}_{k-M}^{+}}^{2} \\ &+ \sum_{i=k-M}^{k} \parallel \boldsymbol{y}_i^{\mathrm{m}} - \boldsymbol{h}(\boldsymbol{x}_i) \parallel_{\boldsymbol{R}^{-1}}^{2} + \sum_{i=k-M+1}^{k} \parallel \boldsymbol{x}_i - \boldsymbol{x}_{\mathrm{int},i} \parallel_{\boldsymbol{Q}^{-1}}^{2} \end{aligned} \tag{4.20}$$

$$\begin{aligned} &\boldsymbol{x}_{\mathrm{int},k-M} = \hat{\boldsymbol{x}}_{k-M}^{+} \\ &\boldsymbol{x}_{\mathrm{int},i} = \boldsymbol{f}_{i|i-1}\left(\boldsymbol{x}_{\mathrm{int},i-1}, \boldsymbol{u}_{i-1}\right) \quad (i = k-M+1, \dots, k) \end{aligned} \tag{4.21}$$

The optimization vector $\boldsymbol{\xi}_k$ is assembled with M subsequent discrete state vectors within the current estimation window. The optimization cost function $g(\boldsymbol{\xi}_k)$ in eq. (4.20) is composed of the following three parts.

The first argument $\|\cdot\|_{\boldsymbol{I}_{k-M}^{+}}$ is the arrival cost, which summarizes all available information prior to the estimation window; this can also be seen as a regulation term on the states at t_{k-M} [Bog14]. It is introduced to guarantee that the oldest estimation $\boldsymbol{x}_{k-M}$ is coincident with the corrected Kalman Filter state estimation $\hat{\boldsymbol{x}}_{k-M}^{+}$ weighted with the information matrix $\boldsymbol{I}_{k-M}^{+}$. Note that in every moving horizon estimation step a Kalman filter step from $k-M-1$ to $k-M$ is performed to fulfill the Kalman state estimation theory – eq. (4.18). The information matrix $\boldsymbol{I}_{k-M}^{+}$ is calculated via the inverse of the covariance matrix$(\boldsymbol{P}_{k-M}^{+})^{-1}$. Since direct matrix inversion should be avoided, due to numerical stability and accuracy, the author proposes to use a SR-UKF or a SR-EKF Kalman filter algorithm that uses square-root decomposition and rank 1 – updates to propagate the covariance matrix in lower triangular form $\boldsymbol{P} = \boldsymbol{L} \cdot \boldsymbol{U}$ (compare

Chapter A.4.2). The inverse of the lower triangular $\boldsymbol{L}$ can be efficiently calculated by the LAPACK routine DTRTRI [And07] and therefore the propagated information matrix results in $\boldsymbol{I}_{k-M}^{+} = \boldsymbol{L}^{-1} \cdot \boldsymbol{L}$.

The second argument $\|\cdot\|_{\boldsymbol{R}^{-1}}$ is the, over the time instances ($k - M$ to M) summarized and with $\boldsymbol{R}^{-1}$ weighted, difference between the available measurements $\boldsymbol{y}_i^{\mathrm{m}}$ and the output equations of the underlying prediction model $\boldsymbol{h}(\boldsymbol{x}_i)$ as function of the states of the current optimization vector $\boldsymbol{\xi}_k$.

Finally, the third argument $\|\cdot\|_{\boldsymbol{Q}^{-1}}$ denotes the, over the time instances ($k - M$ to M) summarized and with $\boldsymbol{Q}^{-1}$ weighted, difference between the optimized states $\boldsymbol{x}_i$ and the open loop integrated prediction model states $\boldsymbol{x}_{\mathrm{int}}$ in eq. (4.21). $\boldsymbol{x}_{\mathrm{int}}$ is calculated only once per optimization step by a simulation using $\hat{\boldsymbol{x}}_{k-M}^{+}$ as start vector and $\boldsymbol{u}_i$ as input vector.

In Figure 4.8 a qualitative graphical interpretation of the optimization problem for a scalar problem with $n_x = n_y = 1$ is shown. The circle points of the quantities denote the particular value at a discrete-time instance, which is evaluated in the objective function eq. (4.20).

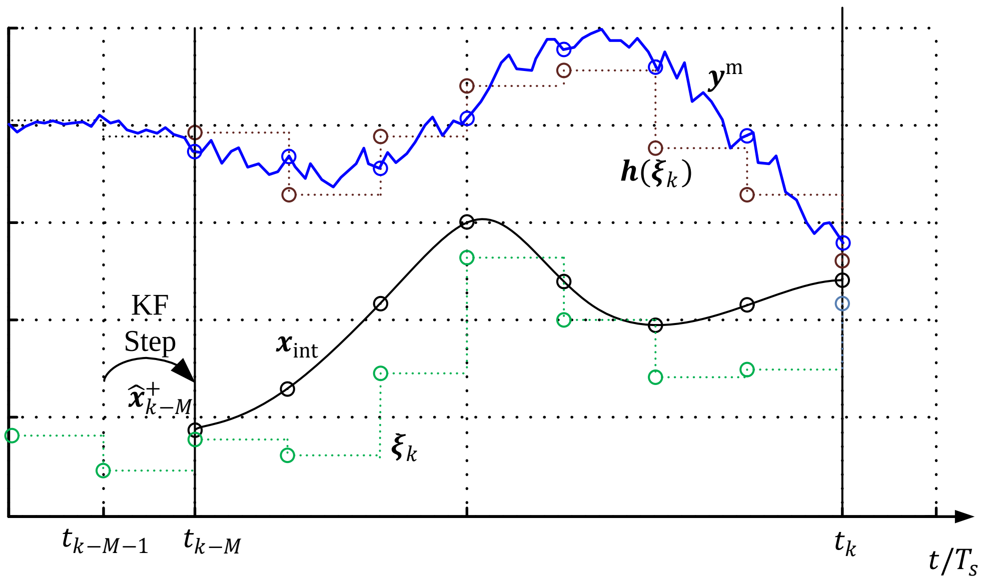

$\boldsymbol{y}^{\mathrm{m}}$
$\boldsymbol{h}(\boldsymbol{\xi}_k)$
KF Step
$\boldsymbol{x}_{\mathrm{int}}$
$\hat{\boldsymbol{x}}_{k-M}^{+}$
$\boldsymbol{\xi}_k$
t_{k-M-1}
t_{k-M}
t_k
t/T_s

Figure 4.8: Graphical interpretation of a moving horizon estimator

The complete algorithm is summarized in Table 4.8. In the first step all past measurements are stored in a first in first out (FIFO) ring buffer. As long as not enough measurements for the complete window M are available, the measurements and the model inputs are appended to the buffer.

Table 4.8: Algorithm for a nonlinear moving horizon estimator

```
1. Set $k = 0$ ($k \in \mathbb{N}^+$) and set $\boldsymbol{x}_k = \boldsymbol{x}_0$
2. Fill ring buffer with measurements and system inputs:
        if $k < M \rightarrow$ append $\boldsymbol{u}_k$ to $\boldsymbol{u}$ and $\boldsymbol{y}_k^{\mathrm{m}}$ to $\boldsymbol{y}^{\mathrm{m}}$
        else $\rightarrow$ left shift one entry of $\boldsymbol{u}$ and $\boldsymbol{y}^{\mathrm{m}}$ and append $\boldsymbol{u}_k$ resp. $\boldsymbol{y}_k^{\mathrm{m}}$
3. Optimize over stored measurement window:
        $\min_{\boldsymbol{\xi}_k} g(\boldsymbol{\xi}_k)$
        $\text{s.t. } \boldsymbol{A} \cdot \boldsymbol{\xi}_k = \boldsymbol{b}$
        $\boldsymbol{G} \cdot \boldsymbol{\xi}_k \leq \boldsymbol{d}$
4. if $k \geq M$ (ring buffer is completely filled)
    a. Propagate $\hat{\boldsymbol{x}}_{k-M-1}^+$ via a Kalman Filter step
        $\hat{\boldsymbol{x}}_{k-M}^+, \boldsymbol{I}_{k-M}^+$
    b. Project states on the constrained area (see Chapter 4.3)
        $\min_{\boldsymbol{x}} \|\boldsymbol{x} - \hat{\boldsymbol{x}}_{k-M}^+\| \text{ s.t. } \boldsymbol{c}(\boldsymbol{x}) \leq 0$
5. Repeat $k = k + 1$
```

4.4.1 A Nonlinear Gradient Descent Optimization Algorithm for MHE

For the solution of the proposed MHE problem formulation, eq. (4.19) to eq. (4.21), a nonlinear gradient descent search algorithm (NG) is chosen, because for this solver only the first derivatives of the system functions are needed, which is an important constraint for the available interfaces of the extended FMI 2.0 co-simulation interface (see Chapter 4.2.2). Furthermore, linear equality and inequality constraints can be incorporated easily and the method can be stopped after every optimization step, still offering a reliable sub-optimal solution. The latter is important if the optimization is not finalized at the next sample instant. The algorithm of the constrained NG is given in Table 4.9, for more details see [Ros60], [Win13]. The here implemented NG algorithm for the observer framework is tailored for the nonlinear moving horizon estimator to meet maximum flexibility in calculation of the gradient and the objective function calculation. I.e. it is possible for the user to modify the calculation in Modelica with replaceable function pointers, without modifying the optimization algorithm itself. This will be shown later in an example in Chapter 5.2.4.

Table 4.9: Nonlinear gradient descent search algorithm for moving horizon estimation

1. Set $j = 0$ ($j \in \mathbb{N}^+$) and define $\boldsymbol{\xi}_k^0 = \left(\boldsymbol{x}_{\text{int},k-M}^T, \dots, \boldsymbol{x}_{\text{int},k}^T\right)^T$
2. Descent direction:
   a. unconstrained case
   $$\boldsymbol{r}_j = -\boldsymbol{\nabla} g\left(\xi_k^j\right)$$
   b. constrained case
   $$\boldsymbol{r}_{\text{pro},j} = -\boldsymbol{P}^\dagger(\mathbf{A}) \cdot \boldsymbol{\nabla} g\left(\xi_k^j\right)$$
   $$\boldsymbol{r}_{\text{f},j} = \boldsymbol{r}_{\text{pro},j} - \left(\boldsymbol{G}_{\text{pa},i}\boldsymbol{r}_{\text{pro},j}\right)\boldsymbol{G}_{\text{pa},i}^T$$
3. Line search to determine the step size:
   $$\eta_j = \underset{0 \le \eta}{\text{argmin}}\; g\left(\xi_k^j + \eta \cdot \boldsymbol{r}_j\right)$$
4. Optimization step:
   $$\xi_k^{j+1} = \xi_k^j + \eta_j \boldsymbol{r}_j$$
5. If stop criterion not reached:
   $$j = j + 1 \text{ and go to step 2;}$$

In step 1 an initial solution $\boldsymbol{\xi}_k^0$ is needed. Well-proven strategies for its calculation is an open loop integration of the prediction model from $\boldsymbol{x}_{k-M}$ to $\boldsymbol{x}_k$ or a left shift of the last optimization vector $\boldsymbol{\xi}_{k-1}$ and appending an open loop integration from $\boldsymbol{x}_{k-1}$ to $\boldsymbol{x}_k$. In the unconstrained case of the second step the gradient $\boldsymbol{\nabla} g\left(\xi_k^j\right)$ of the descent direction can be directly calculated as shown in Table 4.10 (remark, all FMI evaluations shown in Table 4.3 are marked in red).

Table 4.10: Calculation of the MHE gradient by means of FMU evaluations

$$\boldsymbol{R}^* = \left(\boldsymbol{R} \cdot \boldsymbol{R}^T\right)^{-1};\; \boldsymbol{Q}^* = \left(\boldsymbol{Q} \cdot \boldsymbol{Q}^T\right)^{-1}$$
$$\boldsymbol{\nabla} g_{1:n} = \left(\boldsymbol{I}_{k-M}^+ + \left(\boldsymbol{I}_{k-M}^+\right)^T\right)\left(\boldsymbol{x}_{k-M} - \hat{\boldsymbol{x}}_{k-M}^+\right)$$
$$-2\frac{\partial \boldsymbol{h}(\boldsymbol{x}_{k-M})}{\partial \boldsymbol{x}_{k-M}}^T \boldsymbol{R}^*\left(\boldsymbol{y}_{k-M}^m - \boldsymbol{h}(\boldsymbol{x}_{k-M})\right)$$

for $i = k - M + 1, \dots, k$:

$$\boldsymbol{\nabla} g_{1+(i-k+M)\cdot n:(i-k+M+1)\cdot n} = \frac{\partial g(\boldsymbol{\xi}_k)}{\partial \boldsymbol{x}_i}$$
$$= 2\boldsymbol{Q}^*\left(\boldsymbol{x}_i - \boldsymbol{x}_{\text{int},i}\right) - 2\frac{\partial \boldsymbol{h}(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_i}^T \boldsymbol{R}^*\left(\boldsymbol{y}_i^{\text{m}} - \boldsymbol{h}(\boldsymbol{x}_i)\right)$$

If also equality constraints should be incorporated $\boldsymbol{A} \cdot \boldsymbol{\xi} = \boldsymbol{b}$, the descent direction must be projected on these by means of the Moore-Penrose pseudoinverse [Gol13] $\boldsymbol{r}_{\text{pro},j} = \boldsymbol{P}^\dagger(\mathbf{A}) \cdot \boldsymbol{\nabla} g\left(\xi_k^j\right)$. In most of the MHE applications this step can be neglected, i.e. $\boldsymbol{P}^\dagger(\mathbf{A}) = \boldsymbol{I}$, since only inequality constraints are of mayor interest to limit the boundaries of the states. To guarantee that the descent direction does not violate the inequality constraints, a set of possible active straints $\boldsymbol{G}_{\mathrm{pa},i}$ in the null space of the linear constraints must be determined. The descent direction is projected on the active constraints $\left(\boldsymbol{G}_{\mathrm{pa},i}\boldsymbol{r}_{\mathrm{pro},j}\right)\boldsymbol{G}_{\mathrm{pa},i}^{T}$.

In step 3 an iterative free line search via quadratic approximation by a second order Taylor series polynomial is performed (Table 4.11). Unfortunately, the derivatives of $g\left(\boldsymbol{\xi}_k^j\right)$ must be calculated via numerical differences similar to eq. (4.4) since the extended FMU 2.0 co-simulation interface only supports directional derivatives with respect to the system states and inputs, see [Mod13] – Chapter 2.1.9. Note that the necessary determination of $g\left(\boldsymbol{\xi}_k^j + \Delta s\right)$ in the differential quotient causes only evaluations of the algebraic output equations of the FMI (compare eq. (4.20)) since the calculation of the initial guess $\boldsymbol{x}_{\mathrm{int},i}$ is only performed once in this approach.

Table 4.11: Descent step size determination via quadratic approximation

$$g\left(\boldsymbol{\xi}_k^j + \eta \cdot \boldsymbol{r}_j\right) \approx G(\eta) := g\left(\boldsymbol{\xi}_k^j\right) + g'\left(\boldsymbol{\xi}_k^j\right)\eta + \frac{1}{2} g''\left(\boldsymbol{\xi}_k^j\right)\eta^2$$

$$\frac{\partial G(\eta)}{\partial \eta} \overset{!}{=} 0 = g'\left(\boldsymbol{\xi}_k^j\right) + g''\left(\boldsymbol{\xi}_k^j\right)\eta\,,$$

$$\eta_j = -\frac{g'\left(\boldsymbol{\xi}_k^j\right)}{g''\left(\boldsymbol{\xi}_k^j\right)}$$

In step 4 the actual optimization step $\boldsymbol{\xi}_k^{j+1}$ is computed. Finally, in the last step it is checked if the stop criterion is reached. This can be, on the one hand, with a criterion that controls whether the change in the last iteration j is small $\left|g\left(\boldsymbol{\xi}_k^j\right) - g\left(\boldsymbol{\xi}_k^{j+1}\right)\right| < 10 \cdot \epsilon$ and, on the other hand, a real-time constraint that stops the search to guarantee the cycle-time of the real-time system. In this case it is assumed that the initial guess $\boldsymbol{\xi}_k^0$, calculated by the high fidelity Modelica model, has been already a valid suboptimal solution and the iterations of the NG did further improve it (cf. Table 4.9), before the stop $\boldsymbol{\xi}_k^j$.

4.4.2 Theory Extension to Delayed and Multi-rate or Triggered Measurements

State of the Art: Delayed Measurements in Kalman Filters

In many applications there are different kinds of sensors that are connected through a field bus system to the sensor fusion computer, e.g. ROboMObil's central control (see Figure 2.16). In most of the cases these sensors are not synchronized with each other and also have different time delays in the acquisition chain. In [Mer04] a brief and comprehensive overview of relevant methods for time delayed measurement incorporation for recursive Kalman filtering is given and referred to in the following. The simplest method is to neglect the fact, that the measurement at time instance t_k is delayed. An improved approach is to incorporate the delayed measurement with the past states from the time instance t_{k-N}:

$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_{k-N} \cdot (\boldsymbol{y}_{k-N}^{\mathrm{m}} - \boldsymbol{h}_{k-N}(\hat{\boldsymbol{x}}_{k-N}^-)) \tag{4.22}$$

With this formulation still the wrong output equation is fused with the current state prediction. Other approaches, e.g. Alexender's method [Ale91], are of such high complexity that they are comparable to a complete recalculation of the Kalman filter over the horizon from t_{k-N} to t_k [Mer04]. For linear systems (see Chapter A.3.2) a more efficient method is Larsen's method [Lar98], introducing an efficiently calculated correction term $\delta\hat{\boldsymbol{x}}_k^+$ to the current observation fusion:

$$\delta\hat{\boldsymbol{x}}_k^+ = \mathbf{M}_* \cdot \boldsymbol{K}_{k-N} \cdot (\boldsymbol{y}_{k-N}^{\mathrm{m}} - \boldsymbol{H}_{k-N} \cdot \hat{\boldsymbol{x}}_{k-N}^-) \tag{4.23}$$

$$\mathbf{M}_* = \prod_{i=0}^{N-1} (\boldsymbol{I} - \boldsymbol{K}_{k-i}^T \cdot \boldsymbol{H}_{k-i}) \cdot \boldsymbol{F}_{k-i-1} \tag{4.24}$$

In [Mer04] time delayed sensor fusioning for sigma point Kalman filters (SPKF) is presented. In this approach the system state is augmented $(\cdot)^{(\mathrm{a})}$ and a cross correlation of the lagged measurement with the covariance at t_{k-N} and with the covariance of the actual time instance t_k is performed. The augmented state and measurement vectors are stated as follows:

$$\begin{aligned} \hat{\boldsymbol{x}}_k^{(\mathrm{a})-} &= \begin{bmatrix} \hat{\boldsymbol{x}}_k^- \\ \hat{\boldsymbol{x}}_{k-N}^- \end{bmatrix} \\ \tilde{\boldsymbol{y}}_k^{(\mathrm{a})} &= \begin{bmatrix} \boldsymbol{y}_k - \hat{\boldsymbol{y}}_k^- \\ \boldsymbol{y}_{k-N} - \hat{\boldsymbol{y}}_{k-N}^- \end{bmatrix} \end{aligned} \tag{4.25}$$

In comparison to the UKF algorithm in App. Table A.6 the calculation of the a priori covariance $\boldsymbol{P}_k^-$ is augmented to incorporate the delayed measurements in the filter prediction step:

$$\boldsymbol{P}_k^{(\mathrm{a})-} = \begin{bmatrix} \boldsymbol{P}_{x_k}^- & \boldsymbol{P}_{x_k x_{k-N}}^- \\ \boldsymbol{P}_{x_{k-N} x_k}^- & \boldsymbol{P}_{x_{k-N}}^- \end{bmatrix} \tag{4.26}$$

In a similar way the correction step of the unscented Kalman filter is modified:

$$\hat{\boldsymbol{x}}_k^{(\mathrm{a})} = \hat{\boldsymbol{x}}_k^{(\mathrm{a})-} + \boldsymbol{K}_{k-N}^{(\mathrm{a})} \cdot \tilde{\boldsymbol{y}}_k^{(\mathrm{a})} \tag{4.27}$$

The augmented Kalman gain $\boldsymbol{K}_{k-N}^{(\mathrm{a})}$ in eq. (4.28) is calculated with the cross correlations $\boldsymbol{P}_{x_k^{(\mathrm{a})} y_{k-N}}$ and $\boldsymbol{P}_{y_{k-N}}^{-1}$. These are derived in the same way as in the original UKF algorithm via the propagation and weighting of the sigma points with the corresponding values at the time instances t_{k-N} and t_k. When the delayed measurement gets available, the Kalman gain is calculated. The pre-multiplied matrix $\boldsymbol{M}$ controls the inclusion of the delayed measurement when it gets available to the observer in the current time instance.

$$\boldsymbol{K}_{k-N}^{(\mathrm{a})} = \boldsymbol{M} \cdot \boldsymbol{P}_{\boldsymbol{x}_k^{(\mathrm{a})}\boldsymbol{y}_{k-N}} \cdot \boldsymbol{P}_{\boldsymbol{y}_{k-N}}^{-1} = \begin{bmatrix} \boldsymbol{P}_{\boldsymbol{x}_k\tilde{\boldsymbol{y}}_{k-N}} & \boldsymbol{P}_{\tilde{\boldsymbol{y}}_{k-N}}^{-1} \\ \boldsymbol{P}_{\boldsymbol{x}_{k-N}\tilde{\boldsymbol{y}}_{k-N}} & \boldsymbol{P}_{\tilde{\boldsymbol{y}}_{k-N}}^{-1} \end{bmatrix}$$

$$\boldsymbol{M} = \begin{cases} \begin{bmatrix} I & I \\ I & I \end{bmatrix}, \hat{\boldsymbol{y}}_{k-N} = \text{active} \\ \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}, \hat{\boldsymbol{y}}_{k-N} = \text{inactive} \end{cases} \tag{4.28}$$

This method is useful in practical applications when a single sensor information is delayed in an estimation task, but gets computational very demanding if multiple sensors are delayed. Nevertheless, the proposed constraint handling algorithms in Chapter 4.3 can be used as well for the augmented case; however the scaling of the a posteriori covariance matrix can be only performed to the non-lagged part in eq. (4.7).

In the next section, the advantages of the moving horizon estimation technique incorporating reliable handling of delayed or multi-rate measurements are explained.

Multi-rate or Triggered Measurements in MHE

In many real-time control applications the control engineer has to cope with so called multi-rate systems which leads to the fact that signals are only available in an integer multiple to the base real-time clock. In case of ROboMObil this base clock is chosen to be 4 ms (compare illustration in App. Figure C.3). Moreover, it can be the case that some sensor information are only available at certain time instances t_k invoked by an external trigger signal, as it is the case for a GPS low performance receiver, used in a navigation system.

To handle these effects in moving horizon estimation (time-delayed and multi-rate or triggered measurements) the output eq. (4.18) needs to be modified, depending on the scheduling of the available sensor information. In [Val11] an approach for measurement delays compensation in a moving horizon estimator is proposed. It relies on linearization, similar to the linear MHE in Chapter A.5, but without considering the arrival cost propagation of the Kalman filter as it is formulated in eq. (4.20). For the nonlinear MHE approach of Chapter 4.4 it is proposed, to keep the lagged sensor data in a ring buffer, to interconnect them with past measurements of non-lagged sensors, and to index the active sensors at the particular time instance. Comparing this with the formulation of eq. (4.20), all expressions that are connected with the measured output (the middle part) have to be calculated separately in each time step to incorporate the changing number of available sensor information at the dedicated time step.

$$\begin{aligned} g(\boldsymbol{\xi}_k) &= \cdots + \sum_{i=k-M}^{k} \| \boldsymbol{y}_{i,\mathrm{a}}^{\mathrm{m}} - \boldsymbol{h}_{\mathrm{a}}(\boldsymbol{x}_i) \|_{\boldsymbol{R}_k^{-1}}^2 + \cdots \\ \bar{\boldsymbol{\sigma}}_{r_\mathrm{a}} &= \{\boldsymbol{\sigma}_{r_i} | i \in \boldsymbol{y}_k \} \\ \boldsymbol{R}_k &= \mathrm{diag}(\bar{\boldsymbol{\sigma}}_{r_\mathrm{a}}) \\ \boldsymbol{h}_\mathrm{a} &= \{\boldsymbol{h}_i | i \in \boldsymbol{y}_k\} \\ \boldsymbol{y}_\mathrm{a}^\mathrm{m} &= \begin{bmatrix} y_{k-M,1}^\mathrm{m} & \cdots & y_{k,1}^\mathrm{m} \\ \vdots & \cdots & \vdots \\ y_{k-M,n_y}^\mathrm{m} & \cdots & y_{k,n_y}^\mathrm{m} \end{bmatrix} \end{aligned} \tag{4.29}$$

In eq. (4.29) all active measurements are denoted with an additional subscript $(\cdot)_\mathrm{a}$ for active sensors. In the construction of the active measurement matrix $\boldsymbol{y}_\mathrm{a}^\mathrm{m}$ the time-delays of the single sensors are already considered. To cope with varying dimensions in the optimization problem the implementation could utilize the vector indexing feature in the Modelica language. The same technique is used in the Kalman filter propagation (Step 4 in Table 4.8) for the reduced indexed measurements. This procedure is valid according to the Kalman theory and can be seen in analogy to sequential Kalman filtering (see [Sim06] – Chapter 6.1). In these implementations the matrix vector notation of the Kalman filter algorithm (compare Chapter A.3.3) is replaced by sequentially solving a scalar problem for each measurement. Therefore, more sensor information can improve the estimation in the sense of minimizing the covariance $\boldsymbol{P}_{k-M}^{+}$. In the case that less sensor information is available, larger values of $\det(\boldsymbol{P}_{k-M}^{+})$ occur, but the validity of the Kalman theory still holds. In the same manner, also the dimension of the available measurements at t_{k-M} for the calculation of the information matrix $\boldsymbol{I}_{k-M}^{+}$ and the initial guess $\hat{\boldsymbol{x}}_{k-M}^{+}$ needs to be considered in the Kalman step. In Chapter 5.2, a practical example of this approach is shown.

5. Design and Evaluation of Model Based Observers

In this chapter two state estimation applications are discussed. In the first example (Chapter 5.1) a hybrid table data and equation based prediction model approach for a battery state observer is developed. It benefits from automatic discretization with higher order solvers and symbolic transformation by means of the Modelica simulator, as well as from the derivative-free unscented Kalman filter (Chapter A.4) with inequality state constraints (Chapter 4.3). The estimated state of charge (SOC) is used within the energy manager control scheme to limit excessive power demands of the control allocator and to prevent running out-of-capacity during the journey (cf. Figure 3.21). The second application is a vehicle state observer with focus on the estimation of the quantities $\boldsymbol{p}^{\mathrm{I}}_{\mathrm{C_{act}}}, \psi^{\mathrm{I}}_{\mathrm{C_{act}}}, \boldsymbol{v}^{\mathrm{C}}_{\mathrm{act}}, \dot{\psi}^{\mathrm{C}}_{\mathrm{act}}$, which are necessary for the path following control of the EM (cf. Figure 3.21). The synthesis of the vehicle's observer model is a trade-off between modelling complexity and performance. Through the efficient event handling mechanisms of the Modelica compiler it is possible to cope with the vehicle's standstill case, which generally causes a division by zero error. Finally, an extended moving horizon state estimation algorithm enables the incorporation of delayed low bandwidth GPS measurements (cf. Chapter 4.4.2) and the possibility to limit the vehicle position change through the knowledge of the road boundaries.

5.1 A Constrained Nonlinear Battery Observer Application

This subsection is an extended version of the publications [Bre11b] and [Bre11c] that addresses the modeling of a lithium-ion cell for online monitoring and offline benchmarking purposes. It combines physical modeling in an equivalent electric circuit representation with grid tables of cell characteristic information from laboratory tests. The model is fully parameterized and validated with the cell type used in the high voltage (HV) battery pack (cf. Figure 2.12) of ROboMObil.

The development of reasonable, high performance and high capacity secondary chemical energy storages – mainly rechargeable lithium-ion cells – is one of the main tasks for today's automotive industry. In addition to the research of new cell chemistries for higher power density and durability, it is also necessary to develop new embedded systems and advanced algorithms for battery management systems. The aim of these systems is to give a good estimation of actual and future power availability and health monitoring. This requirement is very complex due to the nonlinear behavior, especially in the case of high performance lithium-ion cells. Currently, no direct measurement method is available to determine the cell characteristic parameters and states without destruction of the cell.

All figures with cell characteristic trajectories are measured in conditioned test bench experiments with the "Li-Tec HEI40" cell type used in the ROboMObil HV battery pack. After a brief summary of the state of the art models for lithium-ion cells in Chapter 5.1.1, a hybrid grid table and physical equation mixed estimation model is derived and its quantities and parameterization are explained in Chapter 5.1.2.

5.1.1 State of the Art – Lithium-Ion Cell Modelling

To model the electric behavior of a cell, in literature often an equivalent electric circuit is used. As an example for this type of modeling, the approach from [Boe08] is presented: the cell behavior is separated into three time domains. The impedance determines the short-time range while the long-term behavior is incorporated by a voltage source. Finally, the transitional behavior is captured by a number of exponential functions which are overlaid.

Another possibility for cell modeling is based upon impedance spectroscopy measurements. An example for this class of models can be found in [Sti08] with its enhanced equivalent circuit. The level of detail is significantly higher compared to [Boe08] due to the continuous formulation from low to high frequency effects. The following cell characteristics are considered: ohmic resistances, parasitic inductances, charge transfer and double layer capacity, diffusion processes and formation of solid electrical interface.

For purposes of cell design, the Comsol “Batteries and Fuel Cells Module” [Com17] is an advanced modeling method. The model accuracy is sufficient to simulate the concentration of the electrolytes and therefore it enables battery engineers to test different combinations of materials and dimensions to optimize the cell behavior.

The aforementioned offline models are not applicable to embedded control systems due to their complex modeling approaches and long simulation times. Available battery management systems (BMS), e.g. the system from Actia I+ME [Act17] that is installed in ROMO, use a predetermined cell characteristic table and a current counting method without considering the transient behavior of the cell.
Another more advanced approach is the enhanced self-correcting model (ESC) by Plett proposed in [Ple04], [Ple04b], [Ple04c] and [Ple04d]. The cell is considered as a system with cell current as input and the terminal voltage as the output variable. The SOC variable l is included in the state vector and can therefore be estimated by means of an EKF algorithm. The basis of the model is the open circuit voltage (OCV) U_{OCV} and the ohmic loss. This is represented in the equivalent circuit in Figure 5.1.

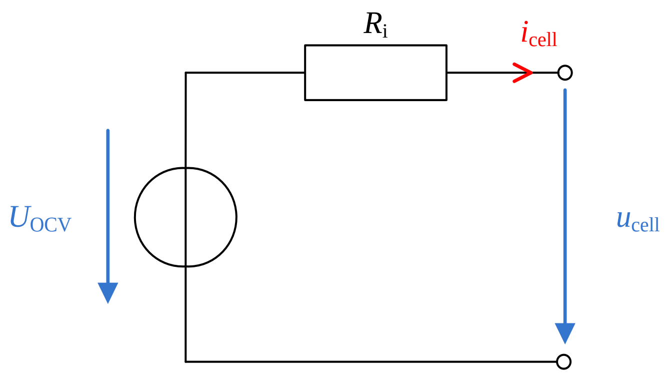


Figure 5.1: Simplified cell equivalent circuit representation of the ESC model

Also hysteresis effects are taken into account. The remaining cell dynamics are described by means of a current filter. These relations result in the following output equation:

$$u_{\mathrm{cell}} = U_{\mathrm{OCV}}(l) + h - R_{\mathrm{i}} \cdot i_{\mathrm{cell}} + f_{\mathrm{f}} \tag{5.1}$$

Here h denotes the hysteresis voltage and f_{f} represents the influence of the current filter. These factors are explained in detail in Chapter 5.1.2.

The implementation of the ESC model shows significant optimization potential. There are several superior discretization methods other than the applied simple explicit Euler1 integration. This helps in simulation stability, speed, and accuracy for online and offline purposes. The formulation of the current filter also seems unnecessary complex. It is formulated as an infinite impulse response (IIR) low pass filter as follows: $\mathrm{LowPass} = 1 - \mathrm{HighPass}$.

Moreover, the computational costly online parameterization of the filter by use of a dual estimation approach (see [Ple04d]) can be done more efficiently offline. In this way the system order can be reduced and therefore the online performance increases. First implementations of the ESC model have shown problems with the determination of the SOC. In addition, the effects of current and temperature on the actual cell capacity are not considered in the ESC model. These effects can have an enormous influence on the calculation of the SOC. This variation is diagramed in Figure 5.2. Especially at low temperatures and high currents the loss of available cell capacity is amplified.

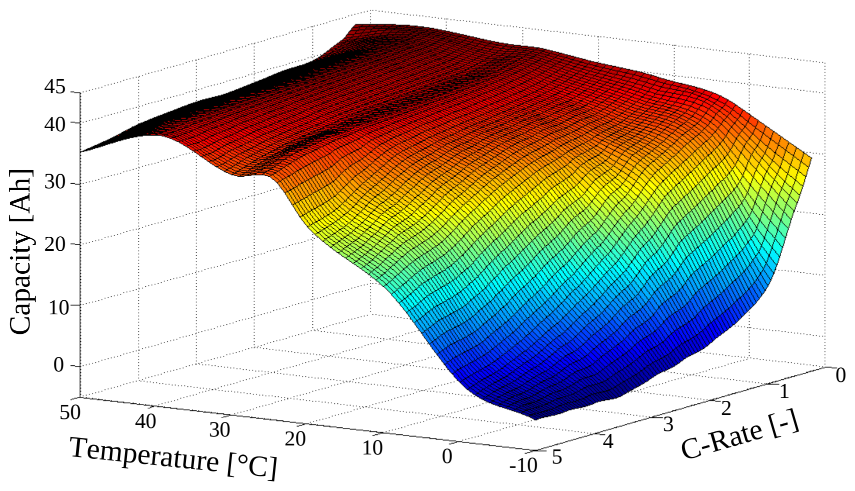


Figure 5.2: Cell capacity in dependency of temperature and current ($C_{\mathrm{Rate}} = C_{\mathrm{N}}/I$)

5.1.2 Proposal of an Equations and Grid Table Combined Cell Model

In the following section a modified and enhanced version of the ESC model is developed, here called the modified enhanced self-correcting model (MESC). In contrast to the original approach the prediction model of the single lithium-ion cell is formulated in acausal continuous-time representation using Modelica. This mathematical description benefits from the automated discretization of the model, by means of a Modelica compiler. The generated prediction FMU can incorporate higher order real-time capable integrators e.g. a Runge-Kutta 4 method and therefore the accuracy as well as stability for a larger sample time T_s can be significantly improved.

The mathematical description of the MESC model in nonlinear state-space representation is formulated as follows:

$$\begin{bmatrix} \dot{l} \\ \dot{h} \\ \\ \dot{\boldsymbol{f}}_\mathbf{f} \\ \\ \end{bmatrix} = \begin{bmatrix} -\frac{\eta_\mathrm{Ah} \cdot k_\mathrm{i}}{C_\mathrm{N}} \cdot i_\mathrm{cell} \\ \left|\frac{\gamma \cdot \eta_\mathrm{Ah} \cdot k_\mathrm{i}}{C_\mathrm{N}} \cdot i_\mathrm{cell}\right| \cdot (M - h) \\ -\omega \cdot f_\mathrm{f1} + \omega \cdot i_\mathrm{cell} \\ \omega \cdot f_\mathrm{f1} - \omega \cdot f_\mathrm{f2} \\ \omega \cdot f_\mathrm{f2} - \omega \cdot f_\mathrm{f3} \\ \omega \cdot f_\mathrm{f3} - \omega \cdot f_\mathrm{f4} \end{bmatrix} \tag{5.2}$$

The differential equation for the SOC l depends on the cell current i_cell, the nominal cell capacity C_N, the coulombic efficiency η_Ah and a correction factor k_i. This factor takes the variation of the cell capacity into account due to the (dis)charging rate and the temperature, see Figure 5.2. Following [Gra02] the correction factor is determined by:

$$k_i = \begin{cases} c_\mathrm{i} \cdot i_\mathrm{cell} + k_0, & \forall\, i_\mathrm{cell} > 0 \text{ (chrg.)} \\ e^{c_\mathrm{i} \cdot i_\mathrm{cell}} + (k_0 - 1), & \forall\, i_\mathrm{cell} < 0 \text{ (dischrg.)} \end{cases} \tag{5.3}$$

Wherein c_i is a positive constant leading into a straight line for positive cell current, which intersects the ordinate at k_0. For a negative cell current the correction factor is described by an exponential function. The parameters c_i and k_0 are determined from capacity tests replacing the simple straight line with more accurate look-up tables.

The hysteresis voltage h (presented in the second differential equation in eq. (5.2)) is described by a more complex equation considering the additional factors M (polarization voltage) and γ (time constant). It describes the dynamic influence of charging and discharging of the cell as depicted in Figure 5.4. Herein M is half of the difference between the charge (blue) and discharge (red) line in dependency of the SOC, exemplified for $l = 0.2$. Especially for lithium-ion cell types with a very flat SOC/OCV curve characteristic (e.g. $\mathrm{LiFePO_4}$ cell chemistry [Wik17b]) neglecting this dynamics may lead to a major model error. The remaining four differential equations describe an optimized fourth order critical damping current filter with only one remaining parameter ω and its four states $\boldsymbol{f}_\mathbf{f}$

$$u_\mathrm{cell} = U_\mathrm{OCV}(l) + h - R_\mathrm{i} \cdot f_\mathrm{f4} \tag{5.4}$$

The output eq. (5.4) is similar to the original ESC model's output equation but aggregates the influence of the cell current (ohmic loss and current filter) into one summand. The internal resistance $R_\mathrm{i}(l, i_\mathrm{cell}, T)$ of the cell is one of the most important descriptive variables. It depends on the SOC, the cell current and the temperature resulting in a three dimensional look-up table. This relationship is visualized for room temperature ($T = 25\,°\mathrm{C}$) in Figure 5.3. Considering eq. (5.4) it is obvious that the resulting cell voltage u_cell varies highly in case of a low SOC and high current flow due to an internal resistance increase of the cell. In such cases the cell is in a demanding situation and can be damaged irreversibly, since the higher resistance causes intense internal cell heating.

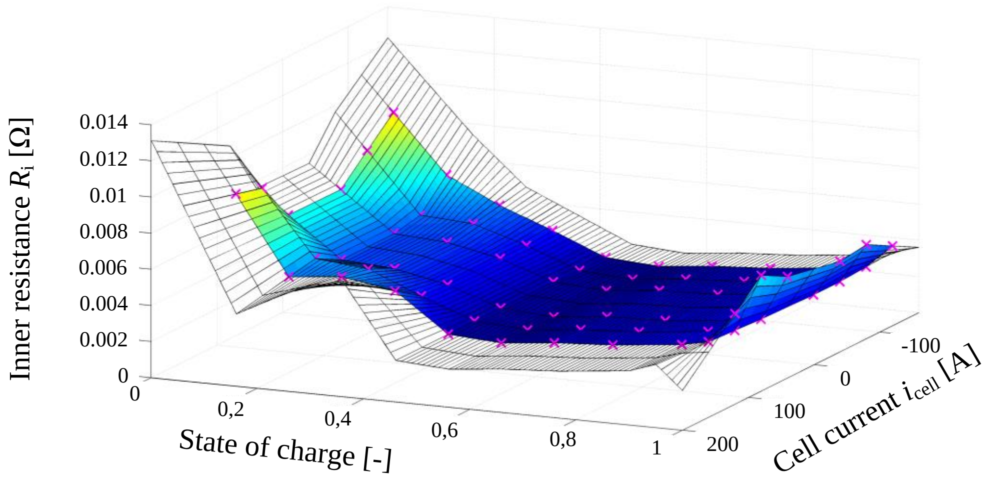


Figure 5.3: Grid table of internal cell resistance for $T = 25$ °C

Another important cell variable is the open circuit voltage (OCV) U_{OCV}, whose characteristic curve incorporates the relationship between SOC and OCV, as shown in Figure 5.4. This figure also illustrates the hysteresis effects during charging and discharging of the cell. The blue and red curve are measured in a cell test bench experiment with very low currents applied to the cell terminals. This minimizes excitation of the cell dynamics so that the cell terminal voltage can be considered unloaded. In addition, the influence of the internal resistance is eliminated during the data analysis. The polarization voltage M is defined as half of the difference between the two curves and therefore also depends on the OCV.

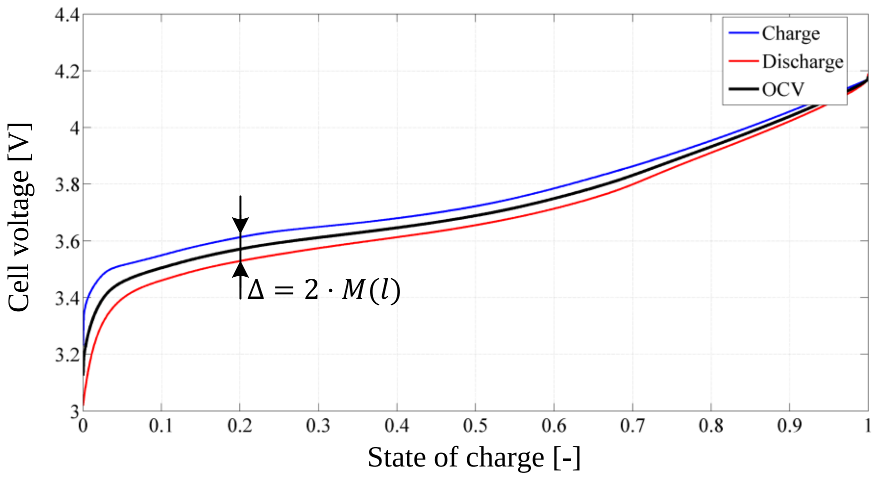


Figure 5.4: Characteristics of the cell open circuit voltage and hysteresis at $T = -10$ °C

Tests have shown that the polarization voltage depends on the actual cell temperature, whereas the highest effect of the considered lithium-ion cell type is for temperatures below 0 °C. This relation is presented graphically in Figure 5.5.

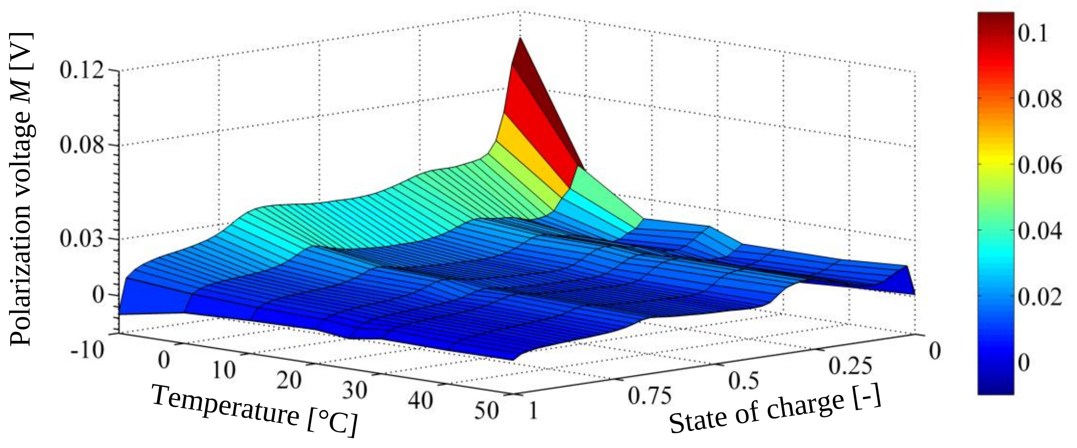


Figure 5.5: Cell polarization voltage M in dependency of temperature and state of charge

The discussed model is implemented in Modelica and included in the latest version of DLR's PowerTrain Library [Tob07]. It is appropriate for offline simulations to optimize energy management strategies of the vehicle controllers and implemented for state of charge estimation in the central control unit of ROboMObil using a rapid prototyping environment.

Cell Model Parameter Derivation and Validation

In the context of the ROboMObil project it was possible to obtain a high performance cell from Li-Tec industries "Li-Tec HEi40". It has a nominal capacity of 40 Ah and with its security features it is fully capable for series production. All cell measurements for parameterization, testing, and validations were done with a Vötsch "VT4011" environment simulator [Vöt17] and a BaSyTec cell testing system [Bas17]. In Figure 5.6 the Modelica implementation of the MESC model is shown. The stationary grid table data SOC/OCV, SOC correction, Polar Voltage, and R_i have been derived offline with this experiment setup.

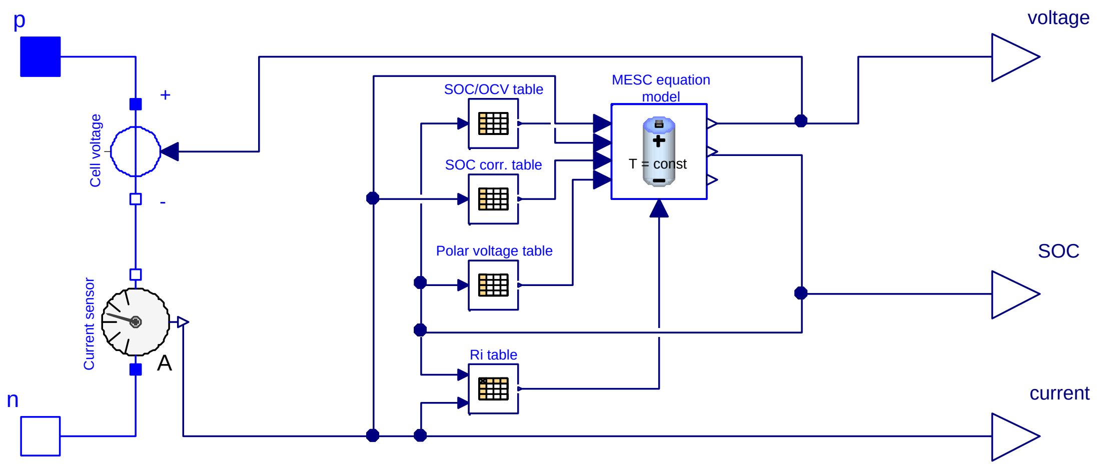


Figure 5.6: Modelica model of the hybrid MESC based on grid tables and equations

The free model parameters in eq. (5.2), by name γ (hysteresis voltage change rate) and $\boldsymbol{f}_{\mathrm{f}_{1..4}}$ (input current filter parameters) were determined by optimization with DLR MOPS (see [Joo08] for more information) on the Linux cluster of the DLR SR institute. The detailed procedure and the necessary test cycles are explained in [Wie10].

5.1.3 Kalman Filter Setup – Perfect Measurements and Power Availability

The MESC model has one input, the cell current, and one output, the cell voltage. These two quantities can be measured directly with high accuracy even in embedded systems. The third quantity is the cell temperature. It is determined by the use of a thermocouple sensor on the cell surface. To achieve a better estimation performance, a second measurement equation is implemented. The main idea is to take constraints into account with a recursive Kalman filter. For this purpose, an additional fictitious measurement is introduced. It can be weighted through the tuning of the output covariance matrix of the Kalman filter. This method is known as perfect measurement in the literature (compare Table 4.5). In this way the output equation of the MESC model is extended to:

$$\boldsymbol{y} = \begin{bmatrix} u_{\mathrm{cell}} \\ l \end{bmatrix} \tag{5.5}$$

The first equation is identical to eq. (5.4) and the second one can be derived as follows:

$$\begin{aligned} u_{\mathrm{cell}} &\approx U_{\mathrm{OCV}}(l) - R_{\mathrm{i}} \cdot i_{\mathrm{cell}} \\ \Rightarrow U_{\mathrm{OCV}}(l) &\approx u_{\mathrm{cell}} + R_{\mathrm{i}} \cdot i_{\mathrm{cell}} \\ \Rightarrow\ l \approx l_{\mathrm{meas}} &= U_{\mathrm{OCV}}^{-1}(u_{\mathrm{cell}} + R_{\mathrm{i}} \cdot i_{\mathrm{cell}}) \end{aligned} \tag{5.6}$$

The measured SOC is calculated through the inverse of the OCV U_{OCV}^{-1} look-up table in combination with a low pass filter to prevent peaks in the perfect measurement output due to rapid changes of the input current. This extension allows the Kalman filter to adjust the SOC directly and therefore to enforce a physically correct estimation.

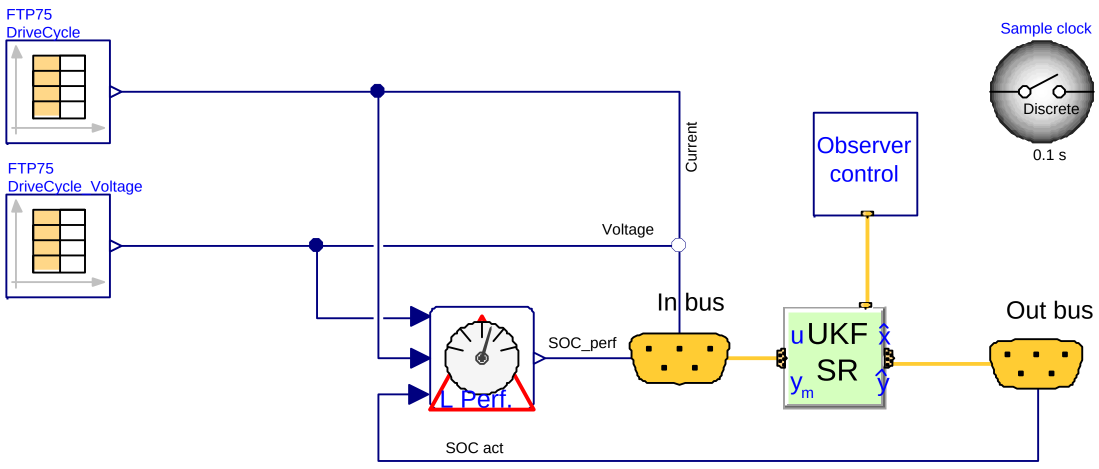


Figure 5.7: Experimental cell observer setup in Modelica

SOC Dependent Power Availability Estimation

Besides the knowledge of the current state of charge it is necessary to know the battery pack power availability for the energy manager (compare control scheme in Figure 3.21). With this information it is possible to limit the power demands of the traction motors, e.g. in case the vehicle is on a descending road and the traction battery is fully charged, it is necessary to reconfigure the actuator demands to use the mechanical brakes or to use the steering to command a slight plow configuration. Also in case of a close-to-minimum discharged battery the longitudinal acceleration demand needs to be reduced or if necessary the vehicle needs to be traced back to standstill. Another scenario is when the battery's SOC is tending to zero. In this case high current demands from the drivetrain may lead to on-bordnet voltage dropdowns caused by the cell SOC/OCV curve decline close to the boundary area (compare Figure 5.4) and the increasing inner resistance (Figure 5.3 upper left corner) correlation $U_{\mathrm{OCV}}(l) \approx u_{\mathrm{cell}} + R_{\mathrm{i}} \cdot i_{\mathrm{cell}}$ in eq. (5.6). Assuming the battery pack is well balanced and the cell health is equalized, the maximum and minimum power availability can be determined by the following formulas:

$$\begin{aligned} P_{\max} &= \min\left(\frac{l - l_{\min}}{\frac{\eta_{\mathrm{Ah}} \cdot \Delta t}{C_{\mathrm{N}}}}; i_{\mathrm{cell}}^{\max} \right) \cdot u_{\mathrm{cell}} \\ P_{\min} &= \max\left(\frac{l - l_{\max}}{\frac{\eta_{\mathrm{Ah}} \cdot \Delta t}{C_{\mathrm{N}}}}; i_{\mathrm{cell}}^{\min} \right) \cdot u_{\mathrm{cell}} \end{aligned} \tag{5.7}$$

Both parameters $i_{\mathrm{cell}}^{\max}$ and $i_{\mathrm{cell}}^{\min}$ denote the technical limitation of the cell type given in the technical datasheet of the corresponding cell manufacture. With this information it is possible to impose a SOC state dependent – besides the technical limitations of the traction motors – control allocation constraint – compare control scheme in Figure 3.21.

$$P_{k,\max} \geq \sum_{i=1}^{4} \omega_k^{\mathrm{W}_i,\mathrm{TM}} \cdot \left(\tau_k^{\mathrm{W}_i,\mathrm{TM}} + \Delta\,\tau_{k+1}^{\mathrm{W}_i,\mathrm{TM}}\right) \cdot \frac{1}{\eta_{\mathrm{el}}^{i}} \geq P_{k,\min} \tag{5.8}$$

The lower index k denotes the current time instance of the control allocation step. Moreover, the current SOC l_k can be used in the velocity profile generation (Chapter 3.3.1) to determine whether or not the cell capacity is sufficient to fulfill the generated velocity profile. If not, the maximum allowed acceleration and vehicle velocity needs to be reduced iteratively.

5.1.4 Parameterization and Model Validation

In this chapter the experimental results with the unconstrained battery observer are discussed. For the generation of the experiment data a FTP-75 driving cycle is used as driving demand for ROboMObil's longitudinal dynamics powertrain model which is based on the DLR PowerTrain Library [Tob07]. This model and its resulting power demand calculation have been validated on the DLR roller test rig in Stuttgart with the whole battery pack (cf. Figure 2.14). With this multiphysical model it is possible to convert the calculated electric power demand of the actuators into the current demand of one cell. In a next step this current demand (Figure 5.8) is used as stimuli data for a single cell test bench in a climate regulated chamber (Chapter 5.1.2).

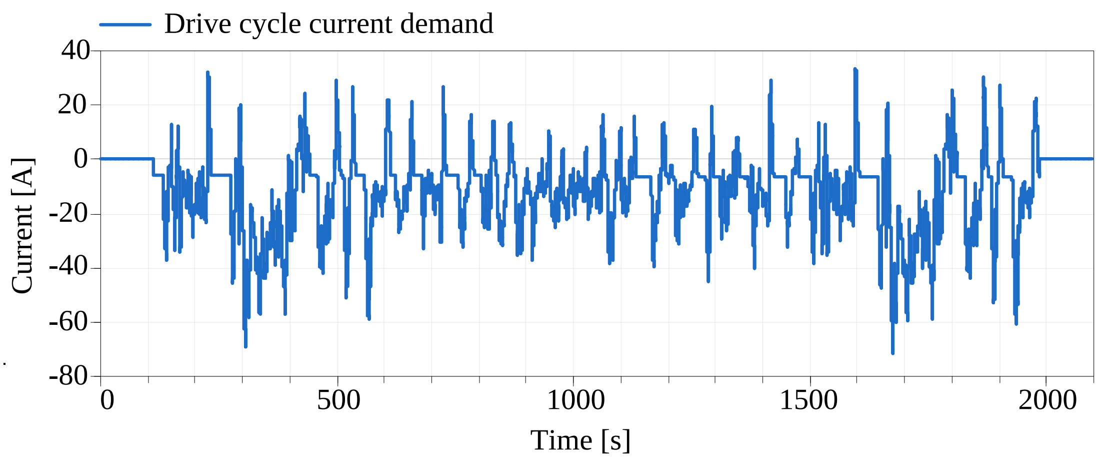


Figure 5.8: Calculated cell current demand based on FTP-75 drive cycle

The voltage at the cell terminals, the surface temperature and the effective current flow are recorded during this test. Finally, they are used as input and measurement data for the nonlinear battery observer experiment setup, shown in Figure 5.7.

In Figure 5.9 the experimental and the corrected results of the proposed FMI model based real-time observer are presented. The red curve shows the SOC characteristic calculated via the perfect measurement.

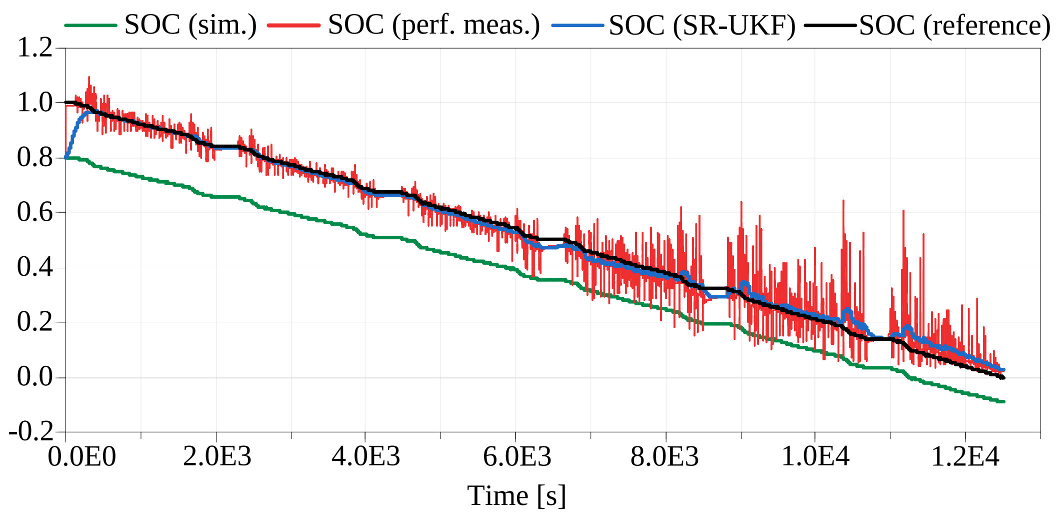


Figure 5.9: Unconstrained observer based state of charge estimation

It is very erratic and noisy, in spite of signal pre-filtering. This characteristic is qualitatively correct, especially in comparison to the green curve, which represents the output of a pure model simulation without observer correction. The pure simulation yields a SOC that is less than zero at the end of the simulation which is physically impossible (cf. Figure 5.9, bottom right). In a real world automotive application, this would cause the SOC display to show incorrect information. In this case it would not be possible to drive on, although the battery is not yet exhaust-ed. Through the nonlinear estimation algorithm, a better and smoother estimation of the SOC can be achieved that converges to zero at the end (blue curve). These observations are quantified by normalized root mean square error criteria (FIT – eq. (C.7)) in the following table.

Table 5.1: State of charge estimation assessment by means of the FIT criterion

SOC reference compared to	FIT
SOC SR-UKF	90.3649 %
SOC perf. measurement	83.3738 %
SOC simulation	47.5859 %

Due to the efficient code for the prediction model provided by the extended FMI 2.0 co-simulation interface, this estimator runs with a real-time factor greater than 100 on standard desktop systems (Intel i7-4600U 2,7 GHz, 8 GB Ram, SSD). Thereby, it was possible to implement the observer on an embedded rapid prototyping controller of ROboMObil.

5.1.5 Extension to Inequality Constraint SOC Estimation

In this subsection the application of the proposed constraint handling mechanisms (see Chapter 4.3) for an one-step Kalman filter estimation algorithm is exemplified. Both approaches, the SLR based (Chapter 4.3.2) as well as the simplified Newton descent search (Chapter 4.3.3), can be used here since a SR-UKF filter has been selected as the most appropriate method, which is stable in case of this estimation task. A common problem in battery state of charge estimation is the limitation of the SOC to realistic values at least between zero and one. Misinterpreted values of the perfect measurement eq. (5.6) exceed the above mentioned boundary values (compare Figure 5.10 – $t = [0\ 10^3]$ red curve) and may cause wrong values of the available charge and therefore also of the power allocation budget. A simplified approach to handle this is to limit the output value of the estimation, but this causes wrong values for future SOC estimate since the state tends more and more in the inadmissible region. In Figure 5.10 the influence of the proposed constraining methods is shown.

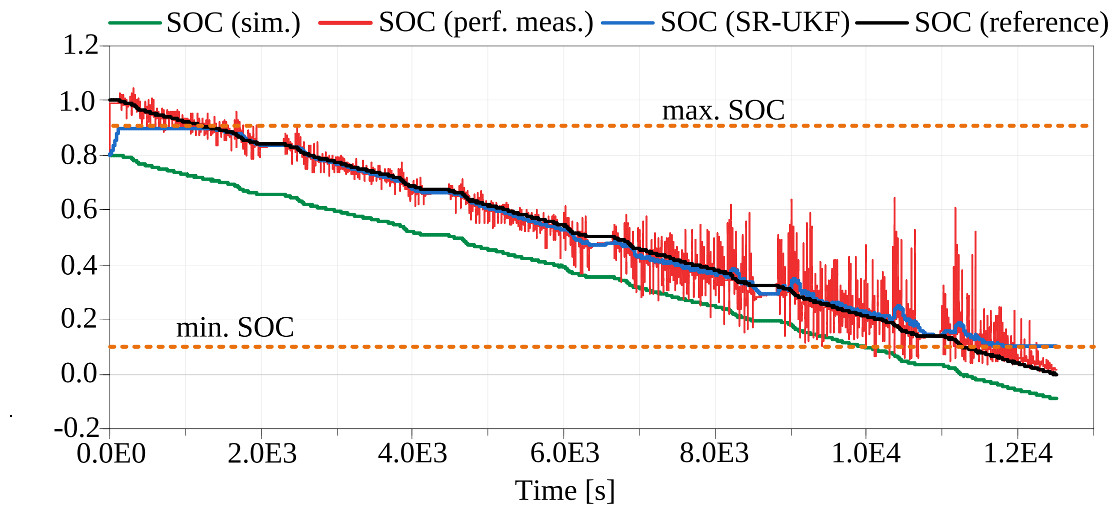


Figure 5.10: Constrained observer based state of charge estimation

Note that here the SOC is limited to [0.1 0.9] due to the lack of missing experimental data, where the aforementioned limits are exceeded. In addition it is worth mentioning that also values, e.g. the cell voltage, are affected by the constraining of the battery SOC value. This is especially important when the correct cell power availability eq. (5.7) should be calculated.

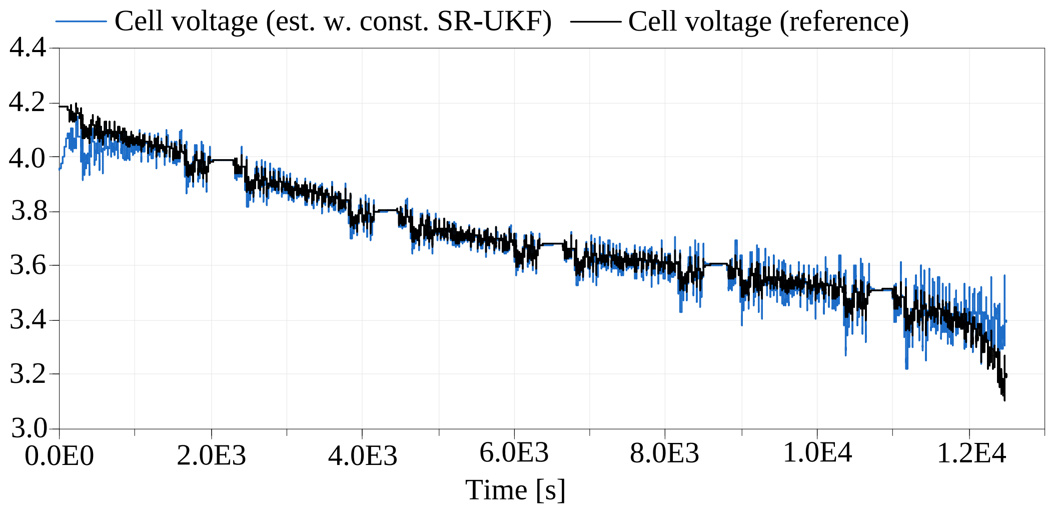


Figure 5.11: Constraint influence to cell voltage estimation

Both state constraining methods have shown similar results in their temporal characteristics, the main difference could be found in the mean integration time $\bar{t}_{\mathrm{sim}}$, where the SLR methods outperforms the more general simplified Newton descent search approach. The experiment with a simulation period of $1.2 \cdot 10^4$ s has been executed on a 64-bit Windows based system (Intel i7-4600U 2,7 GHz, 8 GB Ram, SSD).

Table 5.2: Mean simulation time comparison of the constrained battery observer setups

Setup	Mean integration time $\bar{t}_{\mathrm{sim}}$
SR-UKF unconstrained	49,00 s
SR-UKF w. simplified Newton descent search	71,20 s
SR-UKF w. SLR constraints	62,30 s

The results and benefits of the nonlinear constrained cell observer can be concluded as follows:

- A continuous-time Modelica semi-physical cell model with state dependent characteristic mapping correlation has been automatically symbolically manipulated and discretized via Dymola and imported into the model based observer framework via the extended FMU 2.0 co-simulation interface.
- This gives benefits in comparison to [Ple04], [Ple04b], [Ple04c], [Ple04d] in means of error-prone manual discretization, the capability to make use of higher order discretization methods as well as in prediction model integrated efficient table interpolation.
- The cell prediction model parameters and characteristics were derived from real world test bench experiments matching the cell type used in ROMO for the high voltage system.

- An SR-UKF observer extended with two methods for state constraint handling has been designed and validated with experimental data from cell test bench investigations.

5.2 A Vehicle Position Observer for Path Following Control

The second application is a vehicle state observer for the vehicle dynamics quantities $\boldsymbol{p}^{\mathrm{I}}_{\mathrm{C_{act}}}, \psi^{\mathrm{I}}_{\mathrm{C_{act}}}, \boldsymbol{v}^{\mathrm{C}}_{\mathrm{act}}, \dot{\psi}^{\mathrm{C}}_{\mathrm{act}}$ which are necessary for the path following control of the proposed spatial energy management (cf. Figure 3.21). In Chapter 5.2.1 the motivation for the usage of an MHE algorithm with delayed global positioning system (GPS) measurements is given as well as the experimental setup is motivated. The observer synthesis vehicle model is explained in Chapter 5.2.2 and the details of the mathematical description and parameter fitting matching the characteristics of ROMO is given in Chapter C.1. This extended single track model (ESTM) represents a tradeoff between complexity and accuracy for representing normal driving conditions. Through the event handling mechanisms of Modelica it is possible to extend the model with vehicle standstill functionalities. An extended moving horizon state estimation algorithm (Chapter 5.2.4) enables the incorporation of the delayed low bandwidth GPS measurements (Chapter 4.4.2) and the possibility to limit the vehicle position change through the knowledge of the edges of the roadway (Chapter 5.2.3). Finally, different enhanced optimization objective functions, interconnecting the discrete optimization variables, and a heuristic “anti-freezing” feature are assessed for their effectiveness in a comprehensive observer study (Chapter 5.2.5.).

5.2.1 Motivation and Experimental Setup

In Chapter 2.3.3, ROboMObil’s sensors and actuators were introduced to show the signal runtime through the different measuring feeders and local network busses. Chapter C.3 analyzes in detail the most important vehicle dynamics sensors of ROboMObil, focusing on their noise behavior and relative measurement delays. The outcome is visualized as a scheduling scheme in App. Figure C.3. The longest time-delay of $24\ \mathrm{ms}$ is caused by the Correvit optical velocity-over-ground sensor followed by the OxTS inertial measurement platform. The quantities of the wheel robots are only delayed with one cycle step of $4\ \mathrm{ms}$. In a realistic application with a consumer quality GPS sensor a latency between $50\ \mathrm{ms} - 1000\ \mathrm{ms}$ [Mer04] is usually considered. This implies a drastic delay in comparison to other in-vehicle sensor systems as mentioned above. This fact motivates the here discussed observer example with highly delayed measurements in a vehicle position estimation application.

For the experimental evaluation of the constrained state estimation of ROboMObil, several test campaigns were carried out at the ADAC vehicle testing ground in Kempten. Here, tests of an advanced version of the interactive vehicle path following control (PFC) [Rit15] have been performed. In the top left of Figure 5.12 a portrait of the testing track is shown. The green line depicts the preplanned vehicle path, while the road boundaries are delimited by the dotted-orange lines.

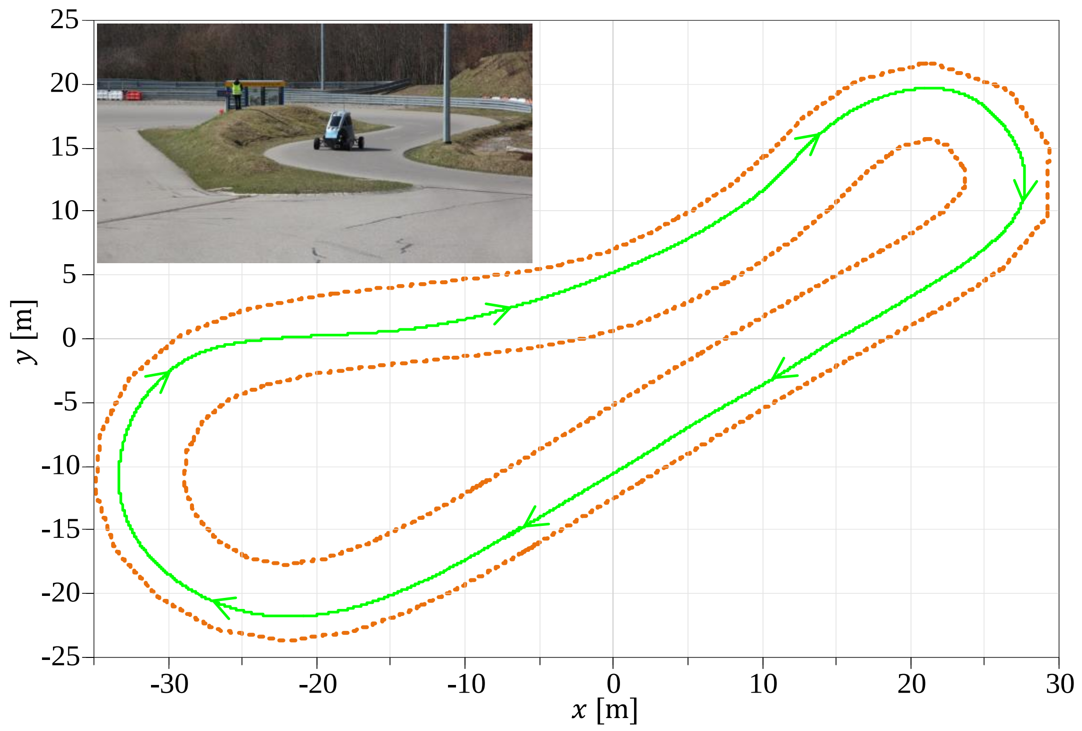
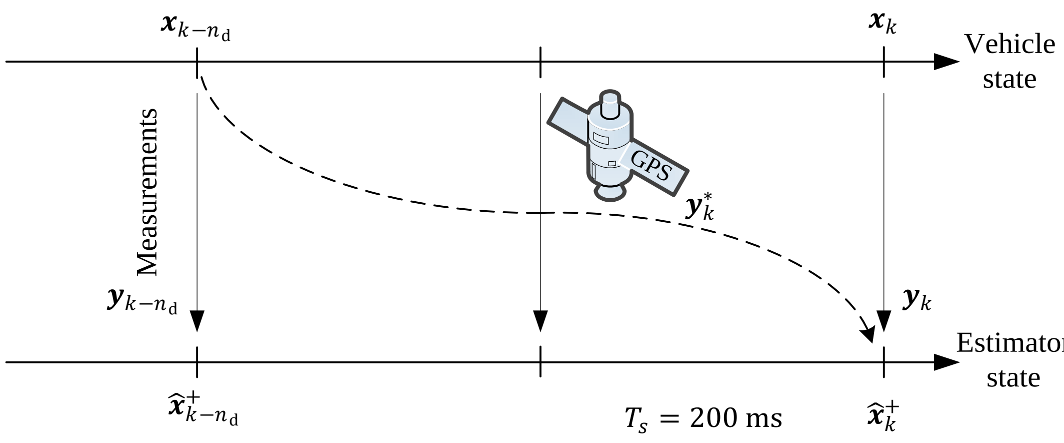


Figure 5.12: PFC experiment with ROMO at ADAC's test facilities in Kempten

The ground-truth data was gathered with help of a differential GPS to guarantee high fidelity vehicle state measurements for the experimental validation. In Figure 5.13 (adopted from [Mer04b]) the situation of the here proposed MHE estimator with delayed measurements is depicted. At the current time instance t_k the estimator receives measurements of the current vehicle yaw rate state $y_k = \dot{\psi}^{\mathrm{C}}_{\mathrm{IMU}}$ and the delayed measurements $\boldsymbol{y}^*_k = \boldsymbol{p}^{\mathrm{I}}_{\mathrm{C_{GPS}}}$ which is delayed for n_{d} samples and thus belong to the past vehicle state $\boldsymbol{x}_{k-n_{\mathrm{d}}}$. Later, it will be shown that the incorporation of these delayed measurements in a conventional one-step estimation approach (i.e. a Kalman filter) leads to poor performance or even instability.



$\boldsymbol{x}_{k-n_{\mathrm{d}}}$
$\boldsymbol{x}_k$
Vehicle state
Measurements
GPS
$\boldsymbol{y}^*_k$
$\boldsymbol{y}_{k-n_{\mathrm{d}}}$
$\boldsymbol{y}_k$
Estimator state
$\hat{\boldsymbol{x}}^+_{k-n_{\mathrm{d}}}$
$T_s = 200$ ms
$\hat{\boldsymbol{x}}^+_k$

Figure 5.13: Delayed GPS measurements incorporation in vehicle position estimation

5.2.2 The Extended Single Track Model

Since the scope of the proposed observer (as implied in the introduction of this chapter) lies on the observation of the vehicle states $\boldsymbol{p}^{\mathrm{I}}_{\mathrm{C_{act}}}, \psi^{\mathrm{I}}_{\mathrm{C_{act}}}, \boldsymbol{v}^{\mathrm{C}}_{\mathrm{act}}, \dot{\psi}^{\mathrm{C}}_{\mathrm{act}}$ for the path following controller of the proposed EM (cf. Figure 3.21) and the vehicle is operated within limited lateral acceleration ($a^{\mathrm{C}}_{y} \leq 5\ \mathrm{m/s^2}$ – compare simulations in Chapter 3.7), an observer synthesis model reproduction depth has been chosen that meets the requirements of computational efficiency and fidelity. It has been decided to use an extended single track model (ESTM) that incorporates vehicle standstill functionalities, rolling resistance, drag forces, and Pacejka's Magic Formula [Pac12] lateral characteristics of the tires.

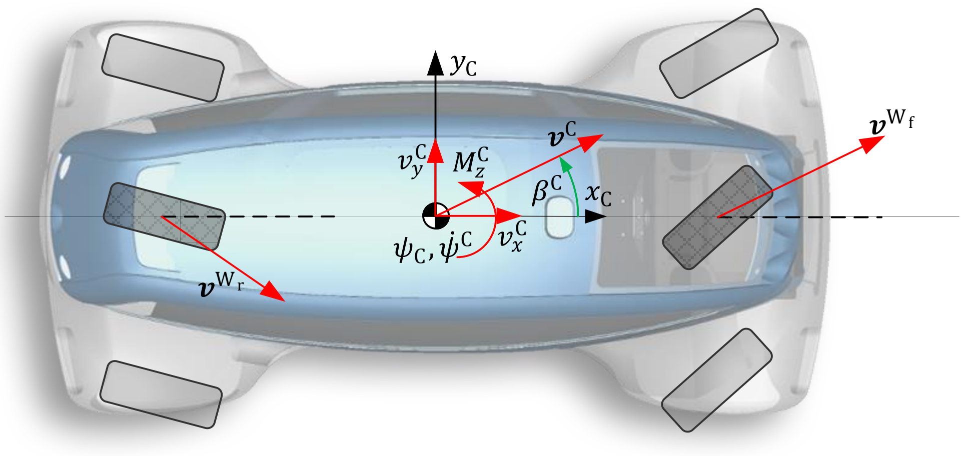

y_{C}
$\boldsymbol{v}^{\mathrm{W_f}}$
$\boldsymbol{v}^{\mathrm{C}}$
v^{C}_{y}
M^{C}_{z}
β^{C}
x_{C}
v^{C}_{x}
$\psi_{\mathrm{C}}, \dot{\psi}^{\mathrm{C}}$
$\boldsymbol{v}^{\mathrm{W_r}}$

Figure 5.14: Quantities of the extended single track model

The details of the extended single track model equation derivation are given in Chapter C.1. It is worth mentioning that with the Modelica modeling technology it was easily possible to integrate "if-else" constructs which are efficiently processed by the event handling features of the compiler and besides experimenting with varying discretization methods and sample rates since the model is in continuous-time formulation. The vehicle state vector is denoted as follows and graphically exemplified in Figure 5.14:

$$\boldsymbol{x}^{\mathrm{C}} = \{\beta^{\mathrm{C}}, v^{\mathrm{C}}, \dot{\psi}^{\mathrm{C}}, \psi_{\mathrm{C}}, x_{\mathrm{C}}, y_{\mathrm{C}}\} \tag{5.9}$$

Preliminary observer case studies showed that with the here chosen modeling approach in combination with a Runge-Kutta 4 integrator the vehicle position estimator can be run with a cycle time of $T_s = 200$ ms. Moreover, the usage of the vehicle yaw rate $\dot{\psi}^{\mathrm{C}}$ can actively contribute to improve the measurement, whereas the incorporation of the vehicle lateral acceleration a^{C}_{y} downgrades the observer performance. The output equation $a^{\mathrm{C}}_{y} = v^{\mathrm{C}} \cdot (\dot{\psi}^{\mathrm{C}} + \dot{\beta}^{\mathrm{C}})$ shows clearly, that the lateral acceleration is algebraically cross-coupled to the vehicle yaw rate $\dot{\psi}^{\mathrm{C}}$. In fact, this causes the aforementioned observations since it is, by the best knowledge to the author, impossible to find a covariance configuration – even by optimization – which makes reasonable use of both sensor information and overcomes the negative effects of e.g. minimal jittering relative delays between both signals.

In Figure 5.15 different simulation results of the same vehicle position estimator are given. Again the road boundaries are marked in dotted-orange color. The vehicle completed three rounds through the circuit. The red trajectory denotes the open loop result – the ESTM model is simulated with the actual system inputs $\boldsymbol{u}$ – and the position is strongly drifting away from the planned trajectory, caused by the evolving position integration error (compare equations for $\mathrm{d}x_\mathrm{C}/\mathrm{d}t\,, \mathrm{d}y_\mathrm{C}/\mathrm{d}t$ – in eq. (C.1)).

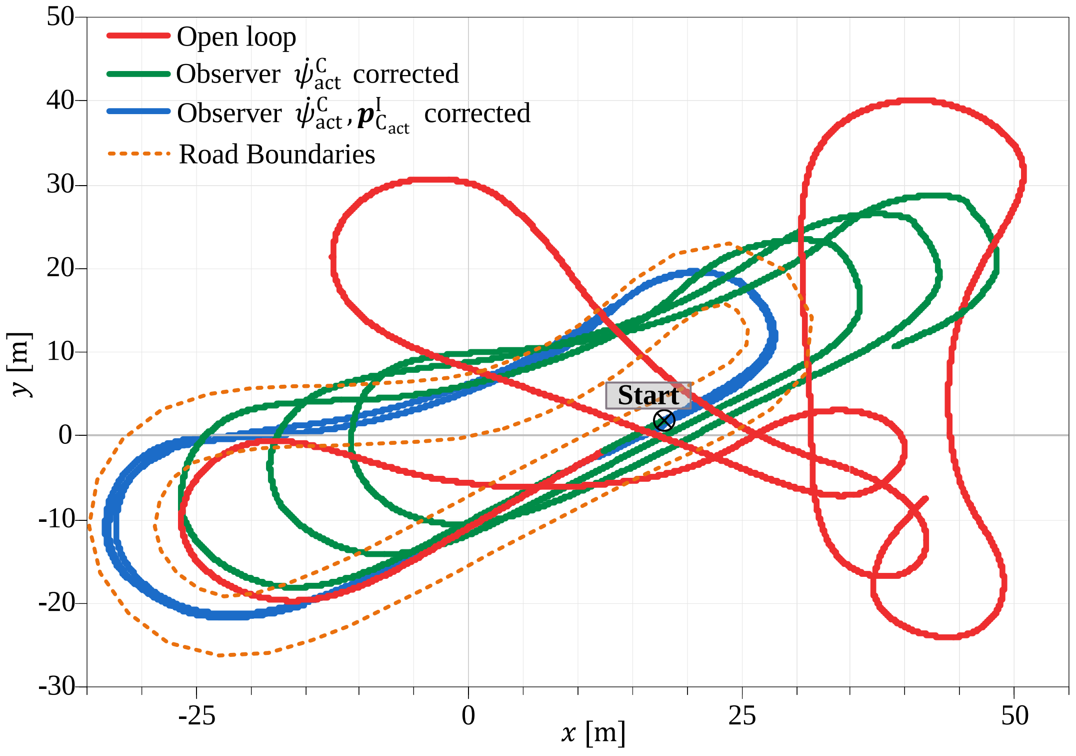


Figure 5.15: Comparison of different measurement incorporations

The green and the blue curves denote the results with a single step Kalman filter under the assumption that no delays are in the measurements.

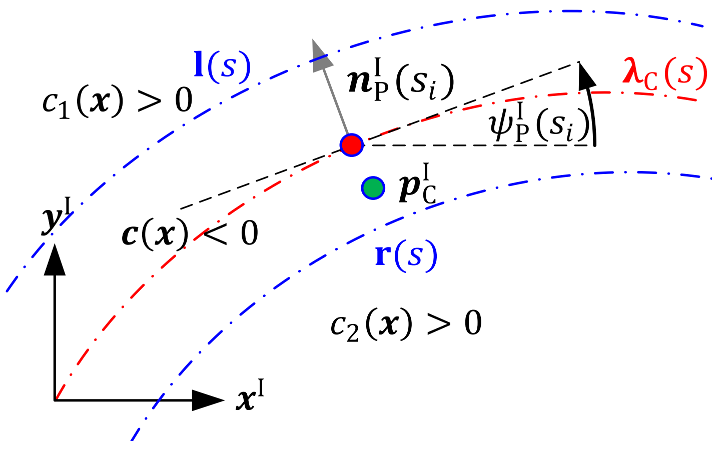
The best observation results could be achieved by the use of an SR-EKF algorithm (see Chapter A.3.3), whereas the SR-UKF (see Chapter A.4.2) was less robust. The reason is the sigma point propagation through the Pacejka tire model, which evidently leads to a wrong state propagation caused by its high sensitivity around the actual state estimation point.

5.2.3 Road Boundaries Constraint Formulation and Evaluation

In this section a methodology is described for incorporating in advance-known street boundaries (e.g. from digital maps in combination with vision sensors) in the vehicle position estimation. It is assumed that the initial vehicle position $\boldsymbol{p}^\mathrm{I}_\mathrm{C}$ is sufficiently precisely known and the corresponding path parameter s^* (compare eq. (3.18)) can be determined by means of the path interpolation from Chapter 3.4.1. The parametric path $\boldsymbol{\lambda}_\mathrm{cstr}(s)$ contains the following quantities for the calculation of the road boundaries constraints: the position $x^\mathrm{I}_\mathrm{P}(s)$ and $y^\mathrm{I}_\mathrm{P}(s)$ of the road middle lane, the positions of the left $(l_x^{\mathrm{I}}(s), l_y^{\mathrm{I}}(s))$ and the right $(r_x^{\mathrm{I}}(s), r_y^{\mathrm{I}}(s))$ border, the corresponding path orientation $\psi_{\mathrm{P}}^{\mathrm{I}}(s)$ and its curvature $\kappa_{\mathrm{P}}(s)$.

The extension of the vehicle prediction model (cf. eq. (C.1)) with the roadway boundaries interpolation yields an extra state $\dot{s}$ (eq. (3.21) and eq. (3.22)), which is a non-physical state that belongs to the estimation task. Experimental tests considering $\dot{s}$ as an estimated state showed, that this configuration leads to unsatisfactory results in the overall observer performance. Therefore, this state is separated from the state correction step in the Kalman algorithms (cf. Chapter A.3.3.). Further optimization regarding computational performance is given in the later Chapter 5.2.4.

In Figure 5.16 the calculation of the roadway border constraints is graphically shown. By means of the above described algorithm a path parameter s_i can be found for which the longitudinal derivation error of the vehicle actual position $\boldsymbol{p}_{\mathrm{C}}^{\mathrm{I}}$ tends to zero.


$c_1(\boldsymbol{x}) > 0$
$\mathbf{l}(s)$
$\boldsymbol{n}_{\mathrm{P}}^{\mathrm{I}}(s_i)$
$\boldsymbol{\lambda}_{\mathrm{C}}(s)$
$\psi_{\mathrm{P}}^{\mathrm{I}}(s_i)$
$\boldsymbol{p}_{\mathrm{C}}^{\mathrm{I}}$
$\boldsymbol{y}^{\mathrm{I}}$
$\boldsymbol{c}(\boldsymbol{x}) < 0$
$\mathbf{r}(s)$
$c_2(\boldsymbol{x}) > 0$
$\boldsymbol{x}^{\mathrm{I}}$
Figure 5.16: Graphical analysis of the street boundary calculation

By means of the path normal vector $\boldsymbol{n}_{\mathrm{P}}^{\mathrm{I}}(s_i)$, represented in the inertial coordinate system (eq. (5.10)), an inequality functional $\boldsymbol{c}(\boldsymbol{x}) \leq 0$ is calculated that penalizes positions outside of the roadway borders (eq. (5.11)):

$$\boldsymbol{n}_{\mathrm{P}}^{\mathrm{I}}(s_i) = \left\{-\sin\left(\lambda_\psi(s_i)\right), \cos\left(\lambda_\psi(s_i)\right)\right\} \tag{5.10}$$

$$\boldsymbol{c}(\boldsymbol{x}) = \begin{bmatrix} -l_x^{\mathrm{I}}(s_i) + x_{\mathrm{C_{act}}}^{\mathrm{I}} & -l_y^{\mathrm{I}}(s_i) + y_{\mathrm{C_{act}}}^{\mathrm{I}} \\ r_x^{\mathrm{I}}(s_i) - x_{\mathrm{C_{act}}}^{\mathrm{I}} & r_y^{\mathrm{I}}(s_i) - y_{\mathrm{C_{act}}}^{\mathrm{I}} \end{bmatrix} \cdot \boldsymbol{n}_{\mathrm{P}}^{\mathrm{I}}(s_i) \tag{5.11}$$

This nonlinear inequality function $\boldsymbol{c}(\boldsymbol{x})$ can be handled directly by the proposed constraining algorithms in Chapter 4.3. For the moving horizon estimation algorithm in Chapter 4.4 it is necessary to linearize $\boldsymbol{c}(\boldsymbol{x})$ at all time instances t_k where it is likely that a constraint may be violated by the estimator $\boldsymbol{c}(\boldsymbol{x}_k) > -\epsilon$:

$$\boldsymbol{c}(\boldsymbol{x}) \cong \boldsymbol{c}(\boldsymbol{x}_k) + \left.\frac{\partial \boldsymbol{c}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_k} \cdot (\boldsymbol{x} - \boldsymbol{x}_k) \tag{5.12}$$

By rearranging eq. (5.12) the linearized inequality description in eq. (4.19) can be formulated:

$$\boldsymbol{C}_k \cdot \boldsymbol{x} \leq \boldsymbol{d}_k \xrightarrow{\text{yields}} \underbrace{\frac{\partial \boldsymbol{c}}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}_k}}_{\boldsymbol{C}_k} \cdot \boldsymbol{x} \leq \underbrace{\frac{\partial \boldsymbol{c}}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}_k} \cdot \boldsymbol{x}_k - \boldsymbol{c}(\boldsymbol{x}_k)}_{\boldsymbol{d}_k} \tag{5.13}$$

Figure 5.17 shows an example for the calculation of the nonlinear constraint function. In the left plot a street is marked with the left $\boldsymbol{l}(s)$ and the right $\boldsymbol{r}(s)$ street boundaries while the car (red line with direction arrows) crosses the right boundary in the hairpin curve. This leads to a violation of the right border constraint condition $c_1(\boldsymbol{x}) > 0, \forall\, t \in [19.6\ 22.1] \vee [29.4\ 30]$ as it can be seen in the right plot of Figure 5.17.

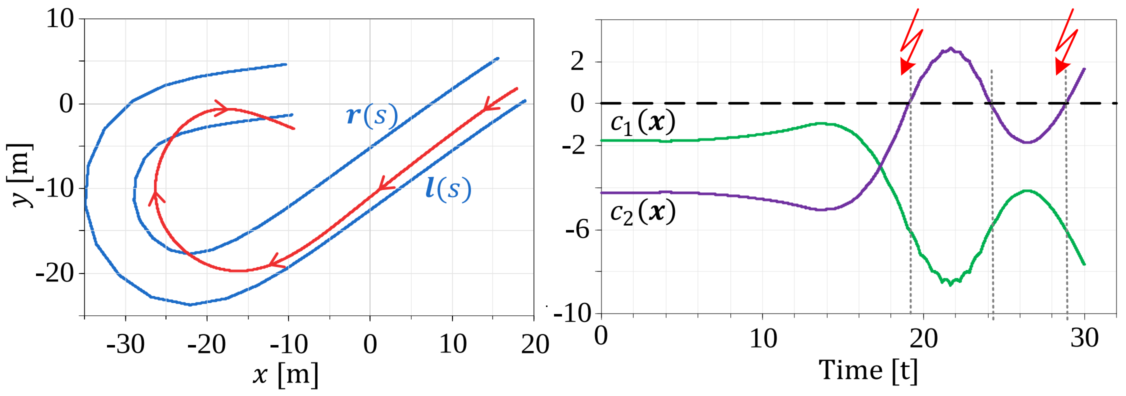


Figure 5.17: Example path with boundary violation (left) and constraints evaluation (right)

Assuming that only the vehicle yaw rate $\dot{\psi}^{\mathrm{C}}_{\mathrm{act}}$ is available to the Kalman filter, the position estimate would drift away like shown in Figure 5.15 (dark green line). Making use of the here proposed boundary estimation approach in combination with the inequality handling feature from Chapter 4.3, yields a bounded and valid result (light green line):

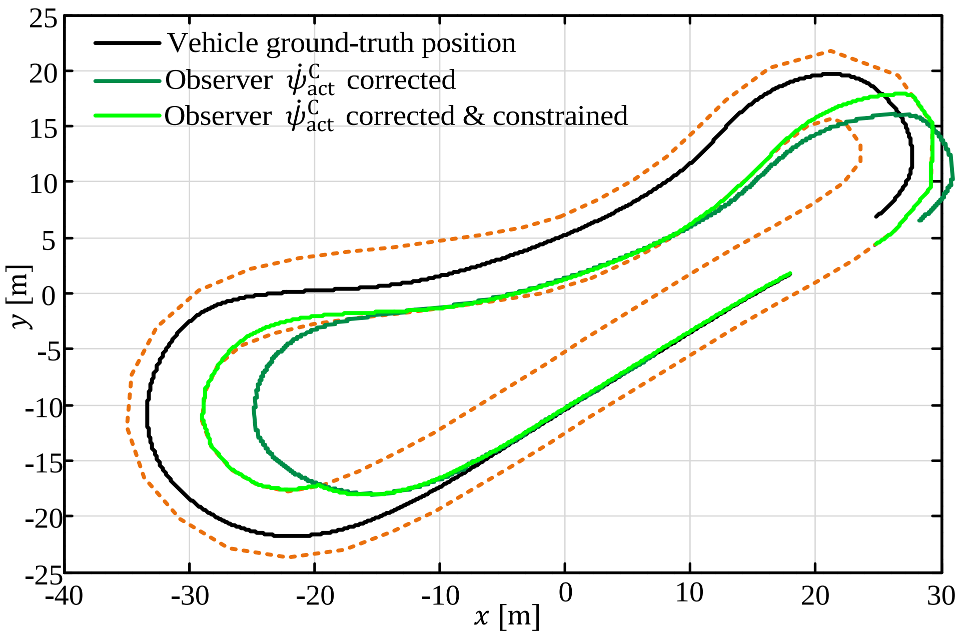


Figure 5.18: Observer behavior with and without taking the path constraints into account

5.2.4 Moving Horizon Estimation Algorithm Extensions

For the here analyzed ESTM MHE observer different extensions are discussed, in comparison to the nominal MHE algorithm formulation in Chapter 4.4. First, a computationally reliable method is introduced for the calculation of the prediction model and the observer constraints, by means of two multi-rate extended FMUs 2.0 for co-simulation. Second, advanced methods for the coupling of discrete optimization variables are proposed. Third, a heuristic method is discussed to prevent optimization freezing through intelligent recalculation of the reference trajectory $\boldsymbol{x}_{\text{int}}$ in segments, where no measurements are available.

Constraint Evaluation with a Multi-Rate FMU Model Splitting Concept

First implementations of the constrained observer were based on a single prediction FMU that combined the ESTM prediction model (Chapter 5.2.2) as well as the boundary constraint evaluation (Chapter 5.2.3). To separate the estimated states from the state of the constraint evaluation (the path parameter s) Modelica's logical vector indexing feature has been used. However, a simulation experiment analysis showed that in this configuration it is necessary to run the whole estimator with a fast sampling rate of $T_s = 4$ ms. This is due to the fast dynamics of the control loop to determine the current path parameter s in the constraint calculation module (cf. Chapter 5.2.3). To overcome this issue the ESTM and the boundary constraint (BC) model were split into two separate FMUs with different sample times ($T_s^{\text{ESTM}} = 200$ ms, $T_s^{\text{BC}} = 4$ ms). In the style of Figure 4.2, the connection of the multi-rate FMUs and the estimation algorithm is sketched in Figure 5.19. Through model splitting, the states of the ESTM FMU ($\boldsymbol{x} = \{\beta^{\text{C}}, v^{\text{C}}, \dot{\psi}^{\text{C}}, \psi_{\text{C}}, x_{\text{C}}, y_{\text{C}}\}$) are the inputs of the boundary constraints FMU, which only has one state, the corresponding path parameter s. The sample time of the constrained FMU is 50 times higher than the one of the ESTM, to guarantee numerical stability. $\boldsymbol{Z}$ denotes the permutation matrix between the inputs of the BC FMU and the states of the ESTM FMU.

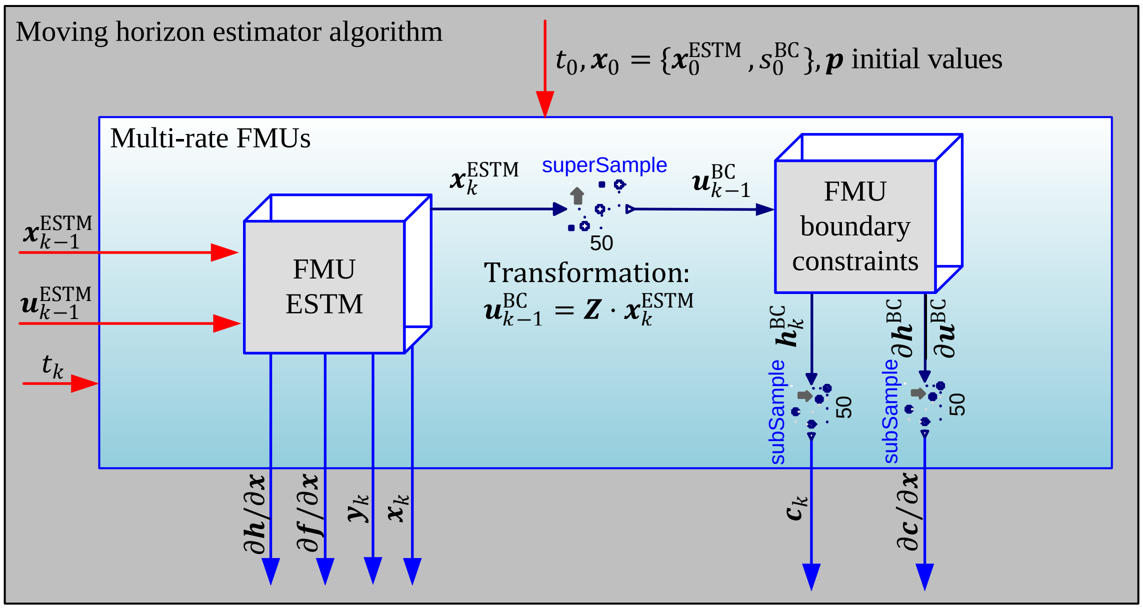

Moving horizon estimator algorithm
$t_0, \boldsymbol{x}_0 = \{\boldsymbol{x}_0^{\text{ESTM}}, s_0^{\text{BC}}\}, \boldsymbol{p}$ initial values
Multi-rate FMUs
$\boldsymbol{x}_{k-1}^{\text{ESTM}}$
$\boldsymbol{u}_{k-1}^{\text{ESTM}}$
t_k
FMU ESTM
$\boldsymbol{x}_k^{\text{ESTM}}$
superSample
50
$\boldsymbol{u}_{k-1}^{\text{BC}}$
FMU boundary constraints
Transformation:
$\boldsymbol{u}_{k-1}^{\text{BC}} = \boldsymbol{Z} \cdot \boldsymbol{x}_k^{\text{ESTM}}$
$\boldsymbol{h}_k^{\text{BC}}$
$\partial \boldsymbol{h}^{\text{BC}} / \partial \boldsymbol{u}^{\text{BC}}$
subSample
50
subSample
50
$\partial \boldsymbol{h}/\partial \boldsymbol{x}$
$\partial \boldsymbol{f}/\partial \boldsymbol{x}$
$\boldsymbol{y}_k$
$\boldsymbol{x}_k$
$\boldsymbol{c}_k$
$\partial \boldsymbol{c}/\partial \boldsymbol{x}$

Figure 5.19: Two encapsulated multi-rate FMUs as one prediction model

With this implementation all the necessary quantities for the constrained MHE are nested within the multi-rate FMU block, whose interfaces to the outside (denoted with blue arrows) are the same as if no inner model separation would have been performed. This gives a large benefit in the matter of computational effort, not only caused by the larger integration step, but it also enables the possibility to calculate the constraint only if it is necessary for the estimation algorithm. This benefit is shown in the following simplified flow diagram of the here proposed extended MHE algorithm (see Figure 5.20). By means of a forward integration in step 1 from t_{k-M} to t_k it is checked whether any constraint may be potentially activated c^a and if so the constraints are linearized along the open loop state trajectory. In step 2 the nonlinear gradient algorithm performs the optimization over the estimation window incorporating the linearized constraints if necessary. In the last step 3 the Kalman filter is updated with consideration of the system constraints to guarantee that the initial state of the moving window in the next iteration step lies within the feasible region.

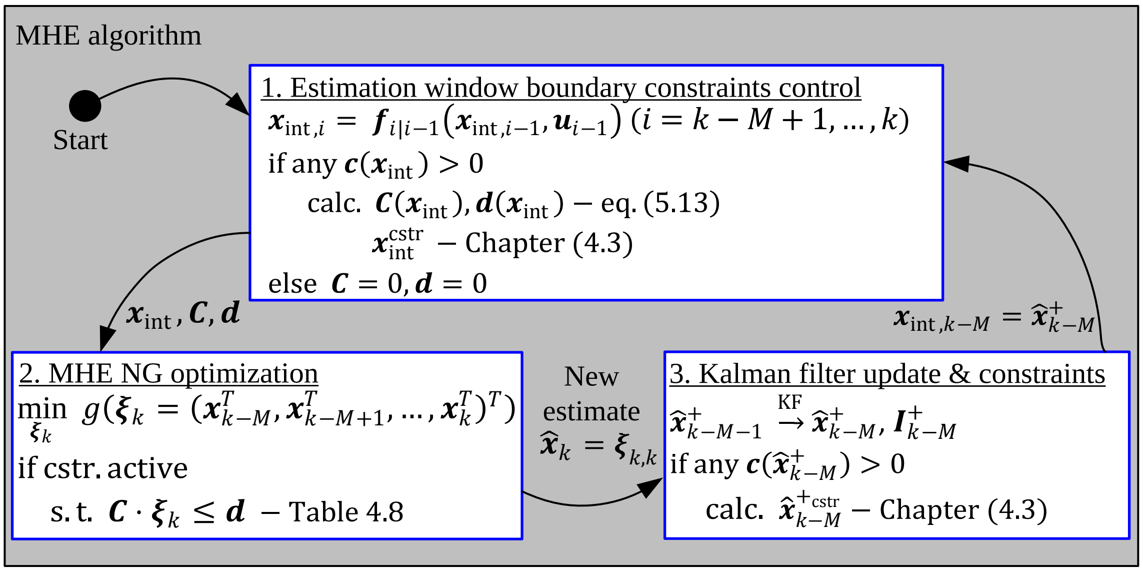

MHE algorithm
Start
1. Estimation window boundary constraints control
$\boldsymbol{x}_{\text{int},i} = \boldsymbol{f}_{i|i-1}(\boldsymbol{x}_{\text{int},i-1}, \boldsymbol{u}_{i-1})\ (i = k-M+1, \dots, k)$
if any $\boldsymbol{c}(\boldsymbol{x}_{\text{int}}) > 0$
calc. $\boldsymbol{C}(\boldsymbol{x}_{\text{int}}), \boldsymbol{d}(\boldsymbol{x}_{\text{int}})$ – eq. (5.13)
$\boldsymbol{x}_{\text{int}}^{\text{cstr}}$ – Chapter (4.3)
else $\boldsymbol{C} = 0, \boldsymbol{d} = 0$
$\boldsymbol{x}_{\text{int}}, \boldsymbol{C}, \boldsymbol{d}$
$\boldsymbol{x}_{\text{int},k-M} = \hat{\boldsymbol{x}}_{k-M}^{+}$
2. MHE NG optimization
$\min_{\boldsymbol{\xi}_k} g(\boldsymbol{\xi}_k = (\boldsymbol{x}_{k-M}^T, \boldsymbol{x}_{k-M+1}^T, \dots, \boldsymbol{x}_k^T)^T)$
if cstr. active
s.t. $\boldsymbol{C} \cdot \boldsymbol{\xi}_k \leq \boldsymbol{d}$ – Table 4.8
New estimate
$\hat{\boldsymbol{x}}_k = \boldsymbol{\xi}_{k,k}$
3. Kalman filter update & constraints
$\hat{\boldsymbol{x}}_{k-M-1}^{+} \xrightarrow{\text{KF}} \hat{\boldsymbol{x}}_{k-M}^{+}, \boldsymbol{I}_{k-M}^{+}$
if any $\boldsymbol{c}(\hat{\boldsymbol{x}}_{k-M}^{+}) > 0$
calc. $\hat{\boldsymbol{x}}_{k-M}^{+\text{cstr}}$ – Chapter (4.3)

Figure 5.20: Flowchart of the extended moving horizon estimator

Qualitatively the incorporation of the constraints is depicted in Figure 5.21. The feasible region of the constraint is limited through the function c_1 and c_2 (orange-dotted). In this example the state propagation of the Kalman filter (step 3) causes a violation of c_2 and therefore the a posteriori propagated state must be constrained by the method proposed in Chapter 4.3.3. The boundary constraints control (step 1) starts now with the corrected a posteriori estimate $\hat{\boldsymbol{x}}_{k-M}^{+\text{cstr}}$ and detects a constraint violation between the third and fourth sample point. This is only possible since the BC model is integrated fifty times between every ESTM evaluation and correction step. To guarantee that the initial solution of the NG solver lies in the feasible region the $\boldsymbol{x}_{\text{int}}^{\text{cstr}}$ is limited via the simplified Newton descent search given in Table 4.7.

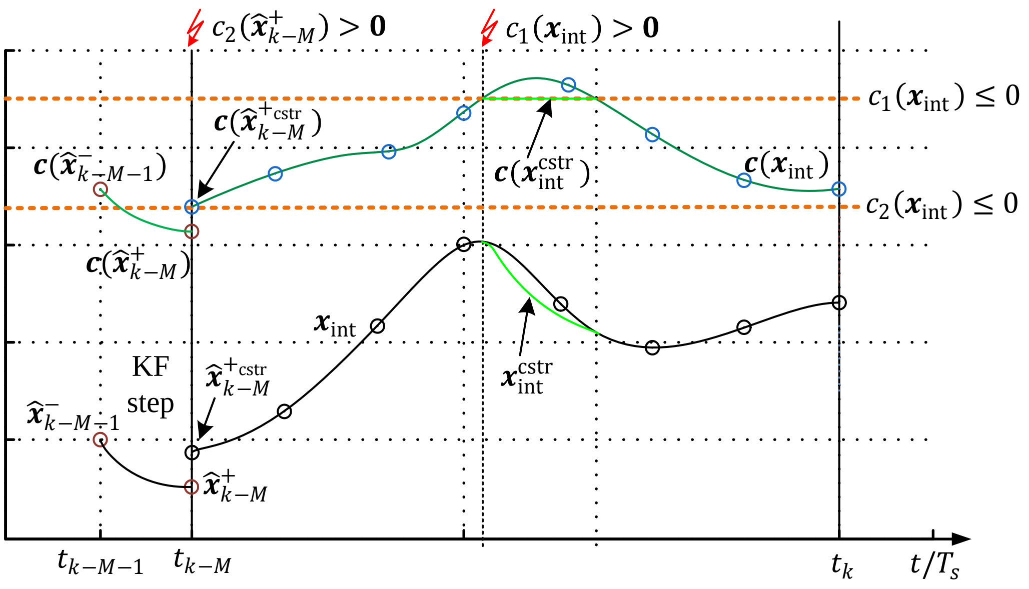

$c_2(\hat{\boldsymbol{x}}_{k-M}^{+}) > \mathbf{0}$
$c_1(\boldsymbol{x}_{\text{int}}) > \mathbf{0}$
$c_1(\boldsymbol{x}_{\text{int}}) \leq 0$
$c_2(\boldsymbol{x}_{\text{int}}) \leq 0$
$\boldsymbol{c}(\hat{\boldsymbol{x}}_{k-M}^{+\text{cstr}})$
$\boldsymbol{c}(\hat{\boldsymbol{x}}_{k-M-1}^{-})$
$\boldsymbol{c}(\boldsymbol{x}_{\text{int}}^{\text{cstr}})$
$\boldsymbol{c}(\boldsymbol{x}_{\text{int}})$
$\boldsymbol{c}(\hat{\boldsymbol{x}}_{k-M}^{+})$
$\boldsymbol{x}_{\text{int}}$
KF step
$\hat{\boldsymbol{x}}_{k-M}^{+\text{cstr}}$
$\boldsymbol{x}_{\text{int}}^{\text{cstr}}$
$\hat{\boldsymbol{x}}_{k-M-1}^{-}$
$\hat{\boldsymbol{x}}_{k-M}^{+}$
t_{k-M-1}
t_{k-M}
t_k
t/T_s

Figure 5.21: Constraint violation detection and handling within a moving horizon window

For the sake of completeness the whole extended MHE algorithm with references is summarized in the following table:

Table 5.3: A MHE algorithm for the ESTM with multi-rate and constraint incorporation

```
1. Set $k = 0$ ($k \in \mathbb{N}^{+}$) and set $\boldsymbol{x}_k = \boldsymbol{x}_0$
2. Fill the ring buffer with measurements and system inputs:
        if $k < M$ → append $\boldsymbol{u}_k$ to $\boldsymbol{u}$ and $\boldsymbol{y}^{\mathrm{m}}_k$ to $\boldsymbol{y}^{\mathrm{m}}$
        else → left shift on entry of $\boldsymbol{u}$ and $\boldsymbol{y}^{\mathrm{m}}$ and append $\boldsymbol{u}_k$ resp. $\boldsymbol{y}^{\mathrm{m}}_k$
3. Perform boundary constraint control in estimation window:
        a. Integrate ESTM within horizon & evaluate constraints:
           $\boldsymbol{x}_{\mathrm{int},k-M} = \hat{\boldsymbol{x}}^{+}_{k-M}$
           $\boldsymbol{x}^{\mathrm{ESTM}}_{\mathrm{int},i} = \boldsymbol{f}^{\mathrm{ESTM}}_{i|i-1}\left(\boldsymbol{x}^{\mathrm{ESTM}}_{\mathrm{int},i-1}, \boldsymbol{u}_{i-1}\right)$  $(i = k - M + 1, \dots, k)$
           for j = 1: upSample
              $\boldsymbol{x}^{\mathrm{BC}}_{\mathrm{int},j} = \boldsymbol{f}^{\mathrm{BC}}_{j|j-1}\left(\boldsymbol{x}^{\mathrm{BC}}_{\mathrm{int},i-1}, \boldsymbol{Z} \cdot \boldsymbol{x}^{\mathrm{ESTM}}_{\mathrm{int},i}\right)$
           $\boldsymbol{c}_i = \boldsymbol{h}(\boldsymbol{x}^{\mathrm{BC}}_{\mathrm{int},j})$
        b. Probe constraint fulfillment:
           if any $\boldsymbol{c}_i > 0$ – (cf. Figure 5.21)
              Linearize constraints for NG optimizer:
              $\boldsymbol{C}_i = \left.\frac{\partial \boldsymbol{c}}{\partial \boldsymbol{x}^{\mathrm{ESTM}}}\right|_{\boldsymbol{x}^{\mathrm{ESTM}}_{\mathrm{int},i}} = \boldsymbol{Z}^{-1} \cdot \partial \boldsymbol{h}^{\mathrm{BC}}_j / \partial \boldsymbol{u}^{\mathrm{BC}}_j$
              $\boldsymbol{d}_i = \boldsymbol{C}_i \cdot \boldsymbol{x}^{\mathrm{ESTM}}_{\mathrm{int},i} - \boldsymbol{c}_i$
              Project initial NG guess to feasible region:
              $\min_{\boldsymbol{x}} \left\| \boldsymbol{x} - \boldsymbol{x}^{\mathrm{ESTM}}_{\mathrm{int}} \right\|$ s.t. $\boldsymbol{c}_i \leq 0$ – Table 4.7
           else $\boldsymbol{C} = 0, \boldsymbol{d} = 0$
4. Optimize over stored measurements with NG algorithm:
        $\min_{\boldsymbol{\xi}_k} g(\boldsymbol{\xi}_k)$
        if cstr. active
           s.t. $\boldsymbol{C} \cdot \boldsymbol{\xi}_k \leq \boldsymbol{d}$
5. if $k \geq M$ (ring buffer is completely filled)
        a. Propagate $\hat{\boldsymbol{x}}^{-}_{k-M}$ via a Kalman filter step:
           $\hat{\boldsymbol{x}}^{+}_{k-M}, \boldsymbol{I}^{+}_{k-M}$
        b. Project states on the constrained area (cf. Chapter 4.3):
           $\min_{\boldsymbol{x}} \| \boldsymbol{x} - \hat{\boldsymbol{x}}^{+}_{k-M} \|$
6. Repeat $k = k + 1$
```

Multiple Shooting Inspired Optimization Objective Extension

In the original MHE problem formulation (eq. (4.19) to eq. (4.21)) the discrete system states within the moving window $\boldsymbol{\xi}_k = \left(\boldsymbol{x}_{k-M}^T, \boldsymbol{x}_{k-M+1}^T, \dots, \boldsymbol{x}_k^T\right)^T$ are not coupled with each other between the sample points t_k. This implies that the optimizer algorithm does not have any information about the dynamic behavior of the ESTM prediction model between the sample points within the estimation window. In the case of the ESTM, with a large sample time $T_s = 200$ ms, this may lead to a physically unfeasible set $\boldsymbol{\xi}_k$ which however minimizes the optimization criteria. A consideration to overcome this weak point is the introduction of coupling penalty terms between the time instances in the minimization criterion in eq. (4.20) and eq. (4.21). Figure 5.22 exemplifies the here developed approach: the initial open loop integration from time instance t_{k-M} to the current time instance t_k is denoted as $\boldsymbol{x}_{\text{int}}^0$. The set of optimized state vectors $\boldsymbol{\xi}_k^j$ in the j-th NG descent step (compare algorithm in Table 4.9) is marked with green circles.

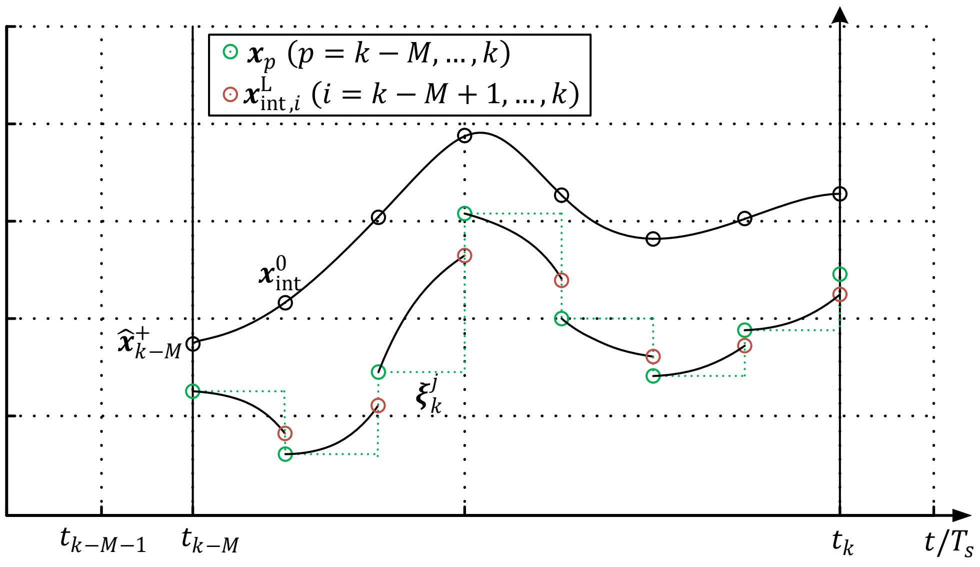

○ $\boldsymbol{x}_p$ $(p = k - M, \dots, k)$
○ $\boldsymbol{x}_{\text{int},i}^{\text{L}}$ $(i = k - M + 1, \dots, k)$
$\boldsymbol{x}_{\text{int}}^0$
$\hat{\boldsymbol{x}}_{k-M}^+$
$\boldsymbol{\xi}_k^j$
t_{k-M-1}
t_{k-M}
t_k
t/T_s

Figure 5.22: Coupling of discrete optimization variables in the moving horizon window

The temporal evolution from these discrete states by means of the FMU yields a set of system states $\boldsymbol{x}_{\text{int},i}^{\text{L}}$ denoted with a red circle:

$$\boldsymbol{x}_{\text{int},i}^{\text{L}} = \boldsymbol{f}_{i|i-1}(\boldsymbol{x}_{i-1}, \boldsymbol{u}_{i-1}), \quad (i = k - M, \dots, k) \tag{5.14}$$

In the depicted qualitative example (cf. Figure 5.22) one can see, that through the evolution of the optimization process a displacement $\boldsymbol{x}_{\text{int},i}^{\text{L}} \neq \boldsymbol{x}_p$ is caused in the j-th iteration step of the NG algorithm (see Table 4.9) . To minimize this gap, the MHE optimization objective is changed to:

$$\begin{aligned} g(\boldsymbol{\xi}_k) = & \parallel \boldsymbol{x}_{k-M} - \hat{\boldsymbol{x}}_{k-M}^{+} \parallel_{\boldsymbol{I}_{k-M}^{+}}^{2} \\ & + \sum_{i=k-M}^{k} \parallel \boldsymbol{y}_i^{\mathrm{m}} - \boldsymbol{h}(\boldsymbol{x}_i) \parallel_{\boldsymbol{R}^{-1}}^{2} + \sum_{i=k-M+1}^{k} \parallel \boldsymbol{x}_i - \boldsymbol{x}_{\mathrm{int},i} \parallel_{\boldsymbol{Q}^{-1}}^{2} \\ & + \underbrace{\sum_{i=k-M}^{k} \parallel \boldsymbol{x}_i - \boldsymbol{x}_{\mathrm{int},i}^{\mathrm{L}} \parallel_{\boldsymbol{Q}_{\mathrm{MS}}}^{2}}_{\text{Additional penalty}} \end{aligned} \tag{5.15}$$

In comparison to the original formulation, the quality functional is extended with an additional weighted least squares expression to enforce a stronger coupling of the piecewise integration $\boldsymbol{x}_{\mathrm{int},i}^{\mathrm{L}}$ and the optimized stated vector $\boldsymbol{x}_i$ by means of the user tunable weighting matrix $\boldsymbol{Q}_{\mathrm{MS}}$.

Besides performing the integration $\boldsymbol{x}_{\mathrm{int},i}^{\mathrm{L}}$ by means of the FMU (eq. (5.14)) it is necessary to approximate the integration rule for the gradient $g(\boldsymbol{\xi}_k)$ calculation. It is proposed that, it is more important to generate a good approximated descent direction for the optimizer than the exact reproduction of the integration method $\boldsymbol{x}_{\mathrm{int},i}^{\mathrm{L}}$ used in the FMU.

In the simplest case this is achieved by the consideration of the directional derivative at the past instance (later called V1). In the second version (V2) the integrator is approximated as an Euler 1 integration rule. The last optimization variable coupling approximation is formulated by a trapezoid integration rule (V3):

$$\begin{aligned} \mathrm{V2} \rightarrow \boldsymbol{x}_{\mathrm{E1},i}^{\mathrm{L}} &= \boldsymbol{f}_{i|i-1}(\boldsymbol{x}_{i-1}, \boldsymbol{u}_{i-1}) \approx \boldsymbol{x}_{i-1} + \boldsymbol{f}_{i-1} \cdot T_s \\ \mathrm{V3} \rightarrow \boldsymbol{x}_{\mathrm{Tr},i}^{\mathrm{L}} &= \boldsymbol{f}_{i|i-1}(\boldsymbol{x}_{i-1}, \boldsymbol{u}_{i-1}) \approx \boldsymbol{x}_{i-1} + \frac{1}{2} \cdot (\boldsymbol{f}_{i-1} + \boldsymbol{f}_i) \cdot T_s \end{aligned} \tag{5.16}$$

With these three versions the complete extended MHE gradient computation is given in Table 5.4. The new additive terms are marked by "V# → " and are exchanged dependent on which approximation should be used for the particular observer setup. Comparing this gradient calculation to the original formulation in Table 4.10, the main difference is the additional “for loop” with the index j in which the operator $+=$ denotes that all values are additively added to the existing entries from the earlier loop.

Table 5.4: Multiple shooting extension for the MHE gradient calculation

$$\boldsymbol{R}^* = (\boldsymbol{R} \cdot \boldsymbol{R})^{-1}\ ;\ \boldsymbol{Q}^* = (\boldsymbol{Q} \cdot \boldsymbol{Q})^{-1}$$

$$\boldsymbol{\nabla} g_{1:n} = (\boldsymbol{I}_{k-M}^{+} + (\boldsymbol{I}_{k-M}^{+})^T)(\boldsymbol{x}_{k-M} - \hat{\boldsymbol{x}}_{k-M}^{+}) - 2\frac{\partial \boldsymbol{h}(\boldsymbol{x}_{k-M})}{\partial \boldsymbol{x}_{k-M}}^T \boldsymbol{R}^*\left(\boldsymbol{y}_{k-M}^m - \boldsymbol{h}(\boldsymbol{x}_{k-M})\right)$$

for $i = k - M + 1, \dots, k$:

$$\boldsymbol{\nabla} g_{1+(i-k+M)\cdot n:(i-k+M+1)\cdot n} = \frac{\partial \boldsymbol{g}(\boldsymbol{\xi}_k)}{\partial \boldsymbol{x}_i} = 2\boldsymbol{Q}^*\left(\boldsymbol{x}_i - \boldsymbol{x}_{\mathrm{int},i}\right) - 2\frac{\partial \boldsymbol{h}(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_i}^T \boldsymbol{R}^*\left(\boldsymbol{y}_i^{\mathrm{m}} - \boldsymbol{h}(\boldsymbol{x}_i)\right)$$

$$\mathrm{V3} \rightarrow += -\underbrace{\left(\frac{1}{2} \cdot T_s \cdot \frac{\partial \boldsymbol{f}_i}{\partial \boldsymbol{x}_i}\right)^T}_{\frac{\partial x_{\mathrm{Tr},i}}{\partial x_i}} \cdot 2 \cdot \boldsymbol{Q}_{\mathrm{MS}} \cdot \left(\boldsymbol{x}_i - \boldsymbol{x}_{\mathrm{int},i}^{\mathrm{L}}\right)$$

for $j = k - M, \dots, k - 1$:

$$\boldsymbol{\nabla} g_{1+(j-k+M)\cdot n:(j-k+M+1)\cdot n} += \frac{\partial \boldsymbol{g}(\boldsymbol{\xi}_k)}{\partial \boldsymbol{x}_j}$$

$$\mathrm{V1} \rightarrow += -\frac{\partial \boldsymbol{f}_j}{\partial \boldsymbol{x}_j} \cdot 2 \cdot \boldsymbol{Q}_{\mathrm{MS}} \cdot \left(\boldsymbol{x}_{j+1} - \boldsymbol{x}_{\mathrm{int},j+1}^{\mathrm{L}}\right)$$

$$\mathrm{V2} \rightarrow += -\underbrace{\left(\frac{1}{2} \cdot T_s \cdot \frac{\partial \boldsymbol{f}_j}{\partial \boldsymbol{x}_j}\right)^T}_{\frac{\partial x_{E1,j+1}}{\partial x_j}} \cdot 2 \cdot \boldsymbol{Q}_{\mathrm{MS}} \cdot \left(\boldsymbol{x}_{j+1} - \boldsymbol{x}_{\mathrm{int},j+1}^{\mathrm{L}}\right)$$

$$\mathrm{V3} \rightarrow += -\underbrace{\left(\boldsymbol{E} + \frac{1}{2} \cdot T_s \cdot \frac{\partial \boldsymbol{f}_j}{\partial \boldsymbol{x}_j}\right)^T}_{\frac{\partial x_{\mathrm{Tr},j+1}}{\partial x_j}} \cdot 2 \cdot \boldsymbol{Q}_{\mathrm{MS}} \cdot \left(\boldsymbol{x}_{j+1} - \boldsymbol{x}_{\mathrm{int},j+1}^{\mathrm{L}}\right)$$

Adaptive Initial Reference Refreshing for Delayed Measurements

In Chapter 4.4.2 a theory extension to MHE is given that enables, the assignment of measurements to their particular time instance by intelligent measurement storage and temporal activation indexing. In Figure 5.23 a measurement signal $\boldsymbol{y}_{\mathrm{a}}^{\mathrm{m}}$ is schematically sketched, which is only available at time instances highlighted with a yellow flash. For example between time instances t_{k-M+1} and t_{k-M+3} no new measurement information is available. Different algorithm experiments have shown that this may force the NG optimizer to tune the variable $\boldsymbol{\xi}_{k,3}$ towards to the initial guess of the open loop state trajectory $\boldsymbol{x}_{\mathrm{int}}$. To overcome this very conservative solution, a heuristic method is introduced in eq. (5.17) to refresh $\boldsymbol{x}_{\mathrm{int}}^{j+1}$ after step 4 in algorithm Table 4.9.

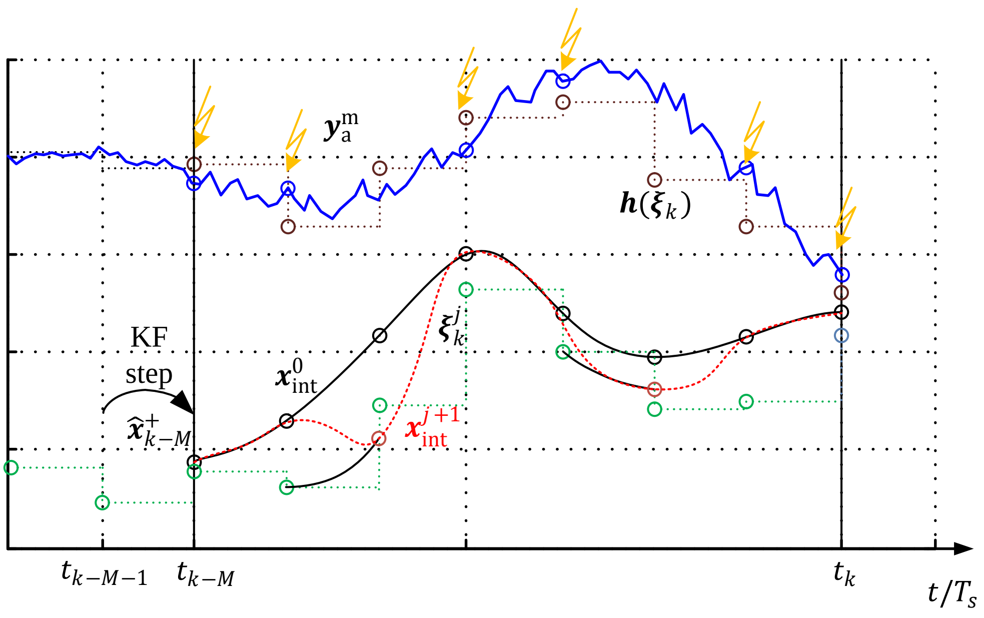


Figure 5.23: Moving horizon window with fragmentary measurements

It determines gaps in the logical vector indexing matrix of the active measurements $y_{\mathrm{index_a}}^{\mathrm{m}}$ and integrates from the last time instance where all measurements are available:

$$x_{\mathrm{int},i}^{j+1} = \begin{cases} x_{\mathrm{int},i}^{0}, \forall \{i|(i \in y_{\mathrm{a}}^{\mathrm{m}}) \wedge (i+1 \in y_{\mathrm{a}}^{\mathrm{m}})\} \\ f_{i|i-1}(\xi_{k,i-1}^{j}, u_{i-1}), \forall \{i|(i \in y_{\mathrm{a}}^{\mathrm{m}}) \wedge (i+1 \notin y_{\mathrm{a}}^{\mathrm{m}})\} \\ f_{i|i-1}(x_{\mathrm{int},i-1}^{j+1}, u_{i-1}), \forall \{i|(i \notin y_{\mathrm{a}}^{\mathrm{m}}) \wedge (i+1 \notin y_{\mathrm{a}}^{\mathrm{m}})\} \end{cases}, (i = k-M, \dots, k) \quad (5.17)$$

5.2.5 Experimental Evaluation of the ESTM MHE Observer

In this last subchapter the complete MHE ESTM algorithm with its extensions to couple the optimization variables (cf. Chapter 5.2.4), refreshment of the initial guess and efficient road boundary constraint handling is demonstrated (compare Figure 5.18). In Table 5.5 the outcome of a comprehensive simulation case study is summarized. The window length has been set to $M = 4$ and the GPS time delay varies between $n_{\mathrm{d}} = 0 \ldots 3$ steps. All parameters for the respective observer configuration have been optimized with the DLR MOPS optimizer framework [Joo08] to guarantee comparable results.

The results in Table 5.5 are sorted with respect to the number of delayed intervals n_{d}. Results that differ tremendously good or bad in a group of the same number of delays are highlighted in green or respectively in red. The first column denotes the used observer setup in which the acronyms for SS = single shooting (standard formulation), MS = multiple shooting, fix = constant initial guess, and up = anti optimization freezing are used. The second column lists the number of delayed samples n_{d} of the GPS signal. The third column gives the mean simulation time $\bar{t}_{\mathrm{sim}}$ of the experiment executed on a standard 64-bit Windows based system (Intel i7-4600U 2,7 GHz, 8 GB Ram, SSD) and can be interpreted as a measure for the increase of computational complexity in comparison to the improvement of the estimation. In the fourth to ninth column the percentage goodness of fit (see Chapter C.5) in comparison to the reference measures of the experiment (cf. Chapter 5.2.1) is summarized. In the last column the square-root maximum distance of the estimation in comparison to the true vehicle position is given.

The first two rows in Table 5.5 give a reference of the ESTM without any observer correction but in the second row with the roadway constraint incorporation. The next two rows are the first results using an observer which only incorporates the measured vehicle yaw rate $\dot{\psi}^{\mathrm{C}}_{\mathrm{act}}$. Both still have a large positional deviation albeit the estimation of the vehicle yaw rate and angle as well as the side slip angle are improved in comparison to the open loop tests. The next section highlights three versions of the ESTM observer with all measurements $\dot{\psi}^{\mathrm{C}}_{\mathrm{act}}$, $\boldsymbol{p}^{\mathrm{I}}_{\mathrm{C_{act}}}$ available and not delayed. The best results can be achieved with the multiple shooting objective V1. The following two experiments incorporate a delay of one sample step. Unfortunately, the SR-EKF algorithm could not be stabilized to give a feasible estimate for all measures; especially the side slip angle β^{C} is heavily oscillating (cf. Figure 5.24 – bottom left). Here, the first time the delayed measurement compensation in the MHE formulation can show its advantage in the single shooting as well as in the multiple shooting objective formulations.

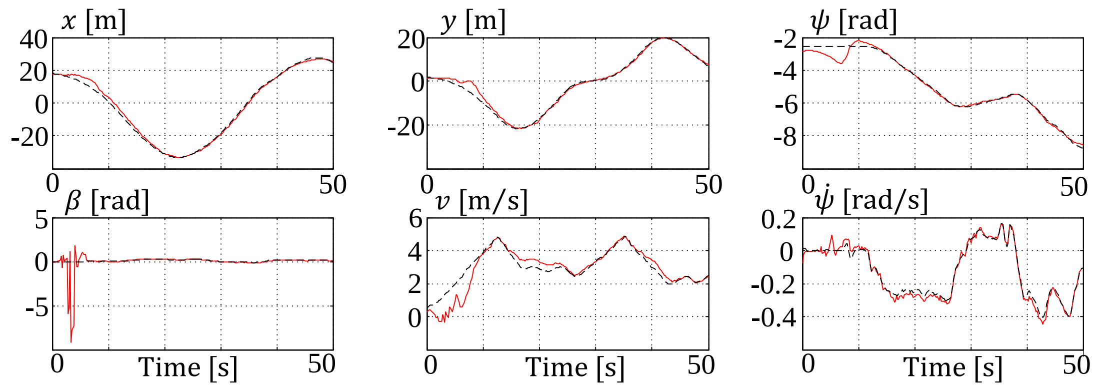


Figure 5.24: ESTM SR-EKF ($n_{\mathrm{d}} = 1$) setup – state estimations (red) vs. reference (black)

The section with $n_{\mathrm{d}} = 2$ is the largest section which correlates to a delay of 400 ms. All the modifications introduced in Chapter 5.2.4 are tested here for their performance. Even though all results are very close to each other, the multiple shooting V1 with reference updating gives the best performance in positioning accuracy and computational reliability (compare state plots in Figure 5.25), only beaten with respect to the mean squared root distance of the V2 version.

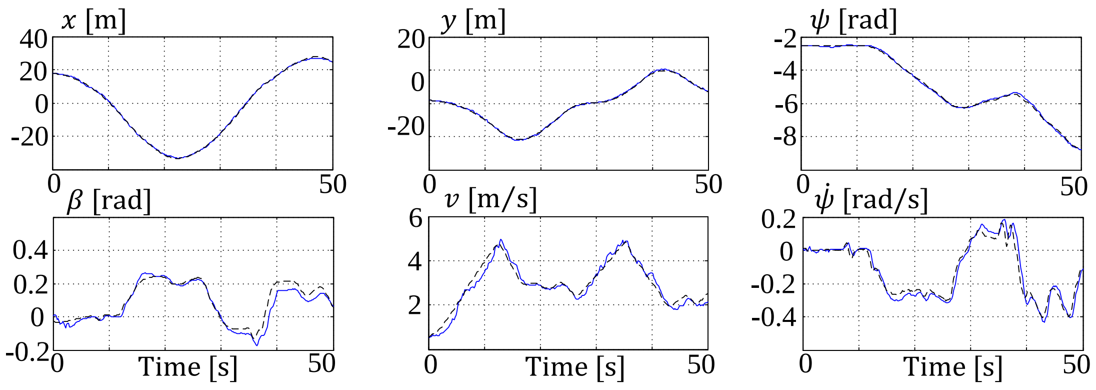


Figure 5.25: ESTM MHE MS up V1 ($n_{\mathrm{d}} = 2$) state estimations (blue) vs. reference (black)

In the last two rows of the simulation study two configurations with $n_\mathrm{d} = 3$ are assessed. The single shooting (see Figure 5.26) as well as the multiple shooting do benefit from the proposed update mechanism, even if the GPS signal is delayed with 600 ms the results are still reliable.

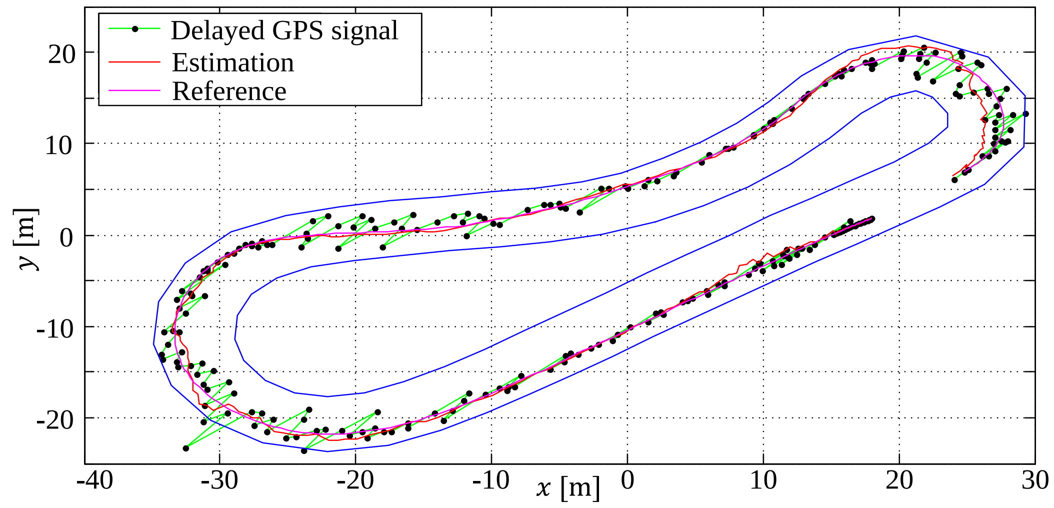


Figure 5.26: ESTM MHE SS up ($n_\mathrm{d} = 3$) – x-y position estimations results

In total it can be stated, that the time delay incorporation in a MHE is very effective and gives good stability especially when the sample steps are large. The competitive SR-EKF algorithm failed already with a delay of $n_\mathrm{d} = 1$ although its computational time is up to 20 times lower. The influence of the extension of the objective function with multiple shooting penalties needs to be analyzed from case to case. Whereas the anti-freezing feature in the optimizer can be seen as a good improvement to the solution at time instance t_k (compare Figure 5.27). Future investigations will be performed with other than the NG solver methods, as recently proposed in [Kou16], that are capable to handle nonlinear (in-)equality constraints in real-time. With this extension it is likely that the multiple shooting approaches might be even more effective.

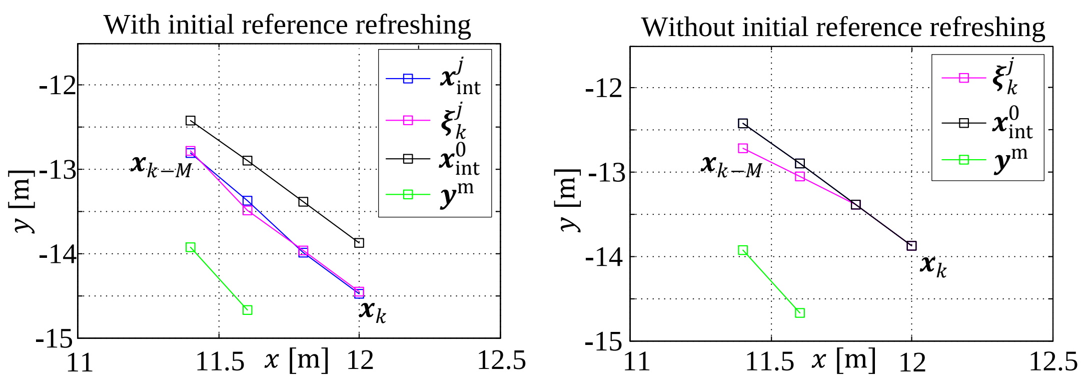


Figure 5.27: Effectiveness of the anti-freezing heuristic within the optimization window

Table 5.5: Comparison of the vehicle position observer performance indicators

Algorithm	n_{d}	$\bar{t}_{\mathrm{sim}}$	$x_{\mathrm{C_{Fit}}}$	$y_{\mathrm{C_{Fit}}}$	$\psi_{\mathrm{C_{Fit}}}$	$\dot{\psi}^{\mathrm{C}}_{\mathrm{Fit}}$	$v^{\mathrm{C}}_{\mathrm{Fit}}$	$\beta^{\mathrm{C}}_{\mathrm{Fit}}$	$\overline{\sqrt{r}}$
Open Loop	-	1.6 s	75.55 %	62.21 %	96.47 %	80.21 %	77.72 %	81.03 %	21.26 m
Open Loop Cstr.	-	1.6 s	87.75 %	87.57 %	96.47 %	80.21 %	77.72 %	81.05 %	8.20 m
SR-EKF $\dot{\psi}^{\mathrm{C}}_{\mathrm{act}}$	0	1.1 s	78.94 %	95.26 %	99.18 %	95.46 %	77.30 %	87.40 %	11.29 m
SR-EKF $\dot{\psi}^{\mathrm{C}}_{\mathrm{act}}$ Cstr.	0	1.2 s	88.21 %	89.26 %	99.17 %	95.47 %	77.31 %	87.40 %	8.27 m
SR-EKF	0	1.6 s	97.97 %	97.26 %	99.21 %	93.33 %	82.42 %	85.26 %	2.26 m
MHE SS	0	16.8 s	97.30 %	96.56 %	99.47 %	95.50 %	83.10 %	81.37 %	2.62 m
MHE MS V1	0	11.6 s	98.56 %	97.82 %	99.36 %	86.90 %	86.42 %	80.46 %	1.42 m
MHE SS up	1	19.7 s	98.29 %	97.80 %	99.45 %	98.35 %	84.63 %	80.94 %	2.05 m
MHE MS up V1	1	23.4 s	98.36 %	97.86 %	99.47 %	86.81 %	85.32 %	81.04 %	2.02 m
MHE SS fix	2	19.4 s	98.49 %	97.85 %	99.25 %	98.93 %	85.99 %	80.10 %	1.49 m
MHE SS up	2	18.8 s	98.70 %	98.13 %	99.25 %	98.22 %	85.99 %	80.36 %	1.31 m
MHE MS up V1	2	15.4 s	98.84 %	98.23 %	99.36 %	86.92 %	86.42 %	80.46 %	1.14 m
MHE MS fix V2	2	16.2 s	98.50 %	98.13 %	99.27 %	86.25 %	86.75 %	80.07 %	1.42 m
MHE MS up V2	2	27.5 s	98.72 %	98.23 %	99.27 %	86.86 %	86.75 %	80.07 %	1.11 m
MHE MS fix V3	2	16.8 s	98.51 %	98.07 %	99.29 %	86.51 %	86.69 %	80.33 %	1.45 m
MHE MS up V3	2	34.2 s	98.70 %	98.30 %	99.29 %	86.66 %	86.69 %	80.35 %	1.20 m
MHE SS up	3	14.2 s	97.79 %	97.13 %	98.95 %	97.28 %	86.85 %	79.66 %	1.96 m
MHE MS up V1	3	19.9 s	97.75 %	97.10 %	99.11 %	86.98 %	87.30 %	80.19 %	2.15 m

The outcome of the ESTM MHE observer can be summarized as follows:

- A continuous-time Modelica vehicle model with event handling for vehicle standstill could be derived (see estimation experiment in Figure 5.28 starting from standstill and coming back to standstill for about 20 s) and automatically discretized via Dymola, the extended FMU 2.0 for co-simulation technology and the model based observer framework.

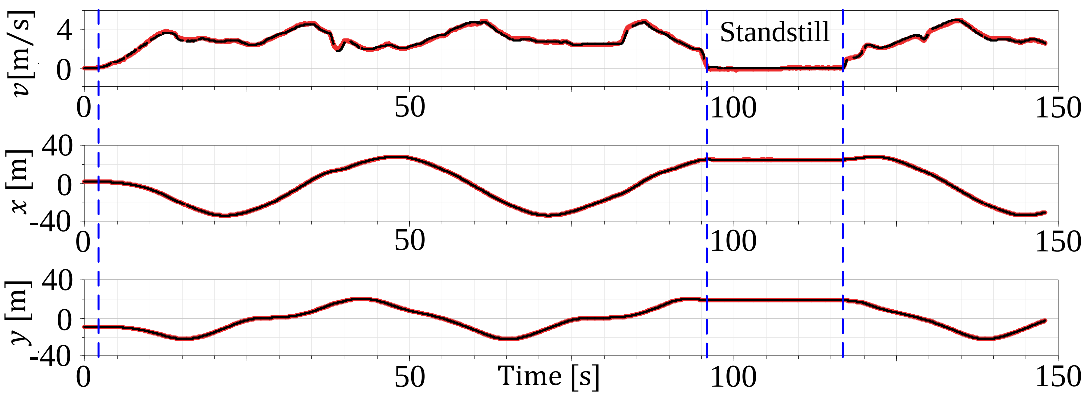


Figure 5.28: Experiment with vehicle standstill – estimation (red) vs. reference (black)

- The derivation of the roadway limit constraint has been designed by extending the principle of the path interpolation introduced in Chapter 3.4.1
- The nominal MHE algorithm of Chapter 4.4 was augmented with a multiple shooting formulation in the objective function, a heuristic optimization freezing prevention, a multi-rate model and a constraint calculation splitting methodology.
- For the experimental investigations real test data from ROMO was selected and additionally the GPS position measures were delayed for the experimental setup.
- All together with the technique for delayed measurements in MHE application (Chapter 4.4.2) a comprehensive simulative assessment with the different objective configuration and anti-freezing features as well as varying delays in comparison to a standard Kalman filter have been given.

- The proposed estimation approach could achieve a position $x_\mathrm{C}, y_\mathrm{C}$ estimate fit of about 98 % and by mean of the delay compensation technique this value is, even with a delay of $n_\mathrm{d} = 3$, only reduced to about 97 %. The estimate of the vehicle velocity is even 4 % improved by the use of the MHE technique in comparison to the EKF. The yaw angle estimate ψ_C quality is for all configurations very high, whereas the yaw rate $\dot{\psi}^\mathrm{C}$ and side slip angle β^C do vary about 5 % but still have reasonable estimates. Finally, the mean square-root distance from the reference could be improved to about 1 m in the $n_\mathrm{d} = 2$ MHE configuration in comparison to the reference SR-EKF filter.

6. Summary

In this last chapter of the doctoral thesis, the outcome of the investigations is discussed, potential and planned developments of the achievements are addressed and the contributions are summarized.

6.1 Conclusion and Discussion

In this thesis a novel, clean sheet designed, robotic electric vehicle development – the DLR ROboMObil – has been shown. Its complete x-by-wire technology with no direct mechanical connection between the driver wish and the actuator control realization in the wheel robots enables a completely new conceptual design of an energy management framework. In comparison to the state of the art, it focuses on the spatial movement and its optimization, empowered by the disruptive ROboMObil concept. From the control design in the different levels of the motion demand abstraction, a control strategy has been successfully designed to minimize the energy consumption (which is crucial for electric vehicles) for executing the driving task. During its development the focus has always been on a later real-time capable in-vehicle realization. Finally, the effectiveness could be assessed via a high-fidelity Modelica full vehicle and mechatronic drivetrain model of ROboMObil.
The approach showed its potential for future investigations on efficient real-time capable trajectory optimization algorithms.

In the second part of the work, a novel approach for automatically generated model based observers has been introduced. It makes use of the extended FMI 2.0 for co-simulation technology to enable a reliable and efficient modeling of the prediction models in continuous-time without the need to care about error-prone manual discretization or system event handling. The framework is designed as flexible as possible so new estimation algorithms can be integrated (i.e. independent from the estimation task) to increase maximal reusability. Besides state of the art state estimation algorithms, different extensions to constraint handling, and real-time capable nonlinear moving horizon estimation have been developed in this thesis. They were applied in two ROboMObil applications: a battery state of charge and a vehicle position estimator. Moreover, the framework was further extended to multi-rate systems and in simulation studies successfully tested with real world measurements.

6.2 Contributions

The contribution and achievements of this doctoral thesis project are summarized below:

- Systematic development of the robotic electric vehicle ROboMObil, by use of model based simulation techniques with the Modelica Language, awarded with the eCarTec 2012 and ESNC 2011 award:
    - Design and optimization of the wheel robot kinematics and suspension design, according to the requirements of a by-wire driven in-wheel motor vehicle with focus on reliability, stability and self-stabilizing mechanic features.
    - Derivation of a hierarchical control mechanism according to the electrical and information technical framework of ROMO.
- Concept of a model based energy manager framework:
    - Design derived from a management pyramid with three levels of abstraction in transformation from the planned route down to actuator commands in the wheel robots.
    - Development of a real-time capable path planning module that reduces the path curvature by means of an efficient nonlinear gradient optimization followed by a vehicle velocity profile generation bounded to the vehicle’s physical limits.
    - Realization of a parametric path description through a path tracking controller.
    - Distribution of the planar movement demand to the actuators by means of a nonlinear control allocator whose minimization objective is the reduction of control energy.
    - Simulative evaluation and assessment in different simulation scenarios.
- Design of a Kalman filter estimator framework with constraints, time delay and MHE:
    - Implementation tailored for embedded systems using FMI technology to incorporate complex Modelica multiphysical models in state estimation problems.
    - Derivation of two Kalman Filter theory extensions to incorporate inequality constraints in one-step recursive state estimation algorithms.
    - Development of a real-time capable moving horizon estimation formulation using a nonlinear gradient descent search algorithm.
    - Introduction of time delayed measurements to moving horizon formulation.
    - Practical application of the framework in a constraint battery estimator using square-root unscented Kalman filter techniques in combination with perfect measurements, as well as a
    - Vehicle position estimator with a multi-rate FMU approach for reliable constraint evaluation combined with delayed GPS measurements compensation in a moving horizon estimator with extended descent search update algorithm and objective functions.

6.3 Outlook: Extensions to the Proposed EM Framework

As aforementioned, one improvement aspect is the further development of the trajectory optimization. A promising approach for the optimal velocity profile generation using dynamic programming is published in [Win17]. Besides improvements in the path tracking controller, there is potential in the level of energy optimal control allocation. As in the simulative assessment of Chapter 3.7.4 already analyzed, the one-step control allocation approach may lead to jittering in the actuator demands. This behavior could be improved by an MPC formulation (eq. (6.3)) that makes use of the knowledge of the future demanded vehicle trajectory:

$$\begin{gathered}\min_{\boldsymbol{u}}\left(J_{\text{criteria}}(\Delta\boldsymbol{u}) + \gamma \parallel \boldsymbol{W}_{\nu}\left(\boldsymbol{B}\left(\boldsymbol{u}^{\text{W}}, \boldsymbol{\nu}^{\text{C}}\right)\Delta\boldsymbol{u} - \Delta\boldsymbol{\nu}\right) \parallel\right) \\ \Delta\underline{\mathbf{u}} \leq \Delta\mathbf{u} \leq \Delta\overline{\mathbf{u}}\end{gathered} \tag{6.1}$$

For a real-time application the proposed nonlinear moving horizon estimator, utilizing the nonlinear gradient descent search algorithm in Chapter 6.4.2, could be an appropriate solution.

6.4 Outlook: Extensions to the FMI Based Estimation Framework

In the ITEA 3 project EMPHYSIS, started in October 2017, it is planned to further develop the FMI technology towards the requirements of embedded systems. This leads to a major increase of the technology readiness level for the here developed FMI based estimation framework, since it can be assumed that the new standard will be computationally more efficient and with a smaller memory footprint. Recently, research at DLR institute SR started to reimplement numerical routines, that are used within the estimation algorithms and are classically taken from the well proofed FORTRAN based matrix computation library Lapack [And99]. The aim is to bring them in standard C-code capable of being certified according to automotive standards for embedded microcontrollers such as ISO 26262 or Misra-C. A second development direction is the newly developed language Modia [Elm17], which can be seen as a potential candidate for Modelica 4.0. It is based on the Julia scientific programming language providing powerful features such as multiple dispatch and meta-programming [Bez17] which enables completely new possibilities in processing the equation manipulated models in discrete-time estimation algorithms. So it is planned, together with the Modelica synchronous features, to bring a model directly into an observer, without having the need to export the prediction in advance to an extended FMU 2.0 for co-simulation and then reimport it.

Besides these technical implementation aspects, there are also additional features planned for upcoming releases of the DLR Kalman Filter Library: FMI based parameter estimation and nonlinear model predictive control as explained in the following subchapters.

6.4.1 Parameter Estimation Using Kalman Filter Techniques and FMI

Beside the estimation of the system state it is also possible to estimate (slow) time variant system parameters. One can imagine a use case e.g. the wearing of the sliding mechanism in a vehicle's electric window, where it is necessary to calculate the correct maximum motor current for the anti-clamping protection algorithm. The fundamental idea of this technique is the extension of the estimation state vector by introducing the parameter vector $\boldsymbol{w}$ (see eq. (A.17)), this yielding the following system description:

$$\begin{gathered} \begin{bmatrix} \dot{\boldsymbol{x}} \\ \dot{\boldsymbol{w}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{u}) \\ \boldsymbol{0} \end{bmatrix}, \\ \boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{w}), \\ t \in \mathbb{R}, \boldsymbol{u}(t) \in \mathbb{R}^{n_u}, \boldsymbol{x}(t) \in \mathbb{R}^{n_x}, \boldsymbol{w}(t) \in \mathbb{R}^{n_w}, \boldsymbol{y}(t) \in \mathbb{R}^{n_y} \end{gathered} \tag{6.2}$$

With the later introduction of the variability "tunable" for scalar variables in the FMI 2.0 standard, i.e. parameters, these can be made tunable during the simulation. Unfortunately the special version of the Modelica simulator Dymola with the extended FMU 2.0 for co-simulation necessary for the state manipulation – which was available to the author – is based on the FMI 2.0 RC1 release, which did not have these features [Mod13].

Moreover, it is worth mentioning that the parameter estimation algorithm can be connected with the constraint handling algorithms introduced in Chapter 4.3. In this way the bounds of possible parameter values can be set $\boldsymbol{w}_k^{\text{low}} \leq \hat{\boldsymbol{w}}_k \leq \boldsymbol{w}_k^{\text{up}}$ as well as the dependency between individual parameters can be bounded $\left|\alpha \cdot \boldsymbol{w}_k^i\right| - \left|\beta \cdot \boldsymbol{w}_k^i\right| \leq \epsilon$.

6.4.2 MHE Algorithm Reformulation to Nonlinear Model Predictive Control

In this section the close interaction between moving horizon estimation and nonlinear model predictive control (NMPC) is sketched. It is shown that also for this type of model based control problems the proposed Modelica and FMI based framework (see Chapter 4.2) is appropriate.

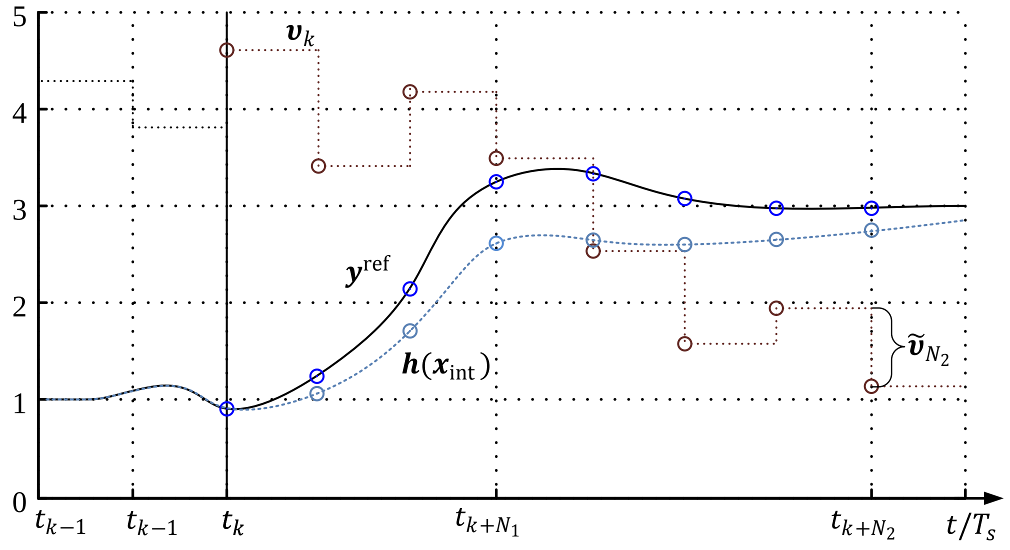

5
4
3
2
1
0
$\boldsymbol{v}_k$
$\boldsymbol{y}^{\text{ref}}$
$\boldsymbol{h}(\boldsymbol{x}_{\text{int}})$
$\tilde{\boldsymbol{v}}_{N_2}$
t_{k-1}
t_{k-1}
t_k
t_{k+N_1}
t_{k+N_2}
t/T_s

Figure 6.1: Schematic diagram of nonlinear model predictive control

By shifting the time horizon to the future, the proposed MHE algorithm (compare Figure 4.8) can be reformulated to a real-time capable nonlinear model predictive control (NMPC) objective that can cope with constraints in its optimization variables $\boldsymbol{v}_k$ (i.e. the discrete model inputs over the horizon) – cf. Figure 6.1.

A finite horizon input constrained NMPC without terminal cost but constraints making use from the solver structure of the NG optimizer (Chapter 4.4.1) is formulized in eq. (6.3). This setup is only valid for dissipative systems and the necessary horizon sizes must be chosen carefully to guarantee closed loop stability [Gru17]. The cost function eq. (6.4) is only dependent on the set of chosen input variables $\boldsymbol{v}_k$ and the state estimate at the current time instance $\boldsymbol{x}_k$.

$$\begin{aligned} \min_{\boldsymbol{v}_k} \; & c\left(\boldsymbol{x}_k, \boldsymbol{v}_k = \left(\boldsymbol{u}_k^T, \boldsymbol{u}_{k+1}^T, \dots, \boldsymbol{u}_{k+N_2}^T\right)^T\right) \\ \text{s.t. } & \boldsymbol{A} \cdot \boldsymbol{v}_k = \boldsymbol{b} \\ & \boldsymbol{C} \cdot \boldsymbol{v}_k \leq \boldsymbol{d} \end{aligned} \tag{6.3}$$

$$\begin{aligned} c(\boldsymbol{x}_k, \boldsymbol{v}_k) = & \sum_{i=N_1}^{N_2} \parallel \boldsymbol{y}_i^{\text{ref}} - \boldsymbol{h}\left(\boldsymbol{x}_{\text{int},i}\right) \parallel_{\boldsymbol{Q}}^2 + \sum_{i=0}^{N_2} \parallel \widetilde{\boldsymbol{v}}_{\text{i}} \parallel_{\boldsymbol{R}}^2 \\ & \widetilde{\boldsymbol{v}}_{\text{i}} = \boldsymbol{v}_{i-1} - \boldsymbol{v}_i \end{aligned} \tag{6.4}$$

$$\begin{aligned} \boldsymbol{x}_{\text{int},k-1} &= \boldsymbol{x}_k \\ \boldsymbol{x}_{\text{int},i} &= \boldsymbol{f}_{i|i-1}\left(\boldsymbol{x}_{\text{int},i-1}, \boldsymbol{v}_{i-1}\right) \end{aligned} \tag{6.5}$$

The first term of the cost functions penalizes the difference between the desired trajectory of the plant $\boldsymbol{y}^{\text{ref}}$and the model output function $\boldsymbol{h}(\boldsymbol{x}_{\text{int}})$, which is calculated by a forward model integration (here again the FMI interface is used) with the set of $\boldsymbol{v}_k$ determined by the optimizer. The second term penalizes too steep gradients of system input variables $\boldsymbol{v}_k$ sets to meet the actuator limitations.

Recently, the proposed methodology of incorporating FMUs in discrete-time algorithms was extended to be used with the DLR Optimization Library [Pfe12] for multi-criteria single shooting nonlinear model predictive control applications [See16].

Bibliography

[Act17] Actia I+ME GmbH. (n.d.). *Batterie Management Systeme*. Retrieved 05 05, 2017, from http://www.ime-actia.de/index.php/de/loesungen-fuer-fahrzeughersteller/loesungen-fuer-pkw/batterie-management-systeme

[Ale91] Alexander, H. L. (1991). State estimation for distributed systems with sensing delay. *Proc. SPIE. 1470, Data Structures and Target Classication*, pp. 103-111. Orlando, FL: SPIE.

[And99] Anderson, E., Bai, Z., Bischof, C., Blackford, L. S., Demmel, J., Dongarra, J. J., et al. (1999). *LAPACK Users' guide* (third ed.). Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

[And07] Andreasson, J. (2007). *On Generic Road Vehicle Motion Modelling and Control.* Ph.D. dissertation, KTH Royal Institute of Technology, Vehicle Dynamics, Stockholm.

[Ara07] Arasaratnam, I., Haykin, S., & Elliott, R. (2007). Discrete-Time Nonlinear Filtering Algorithms Using Gauss Hermite Quadrature. *Proceedings of the IEEE , 95* (5), 953-977.

[Arn12] Arnold, N., Henning, H., Kutschera, I., & Bargende, M. (2012). Cylinder charge based injection control. *12. Internationales Stuttgarter Symposium Automobil- und Motorentechnik.* Stuttgart, Germany: ATZlive.

[Aso08] Asogawa, K. (2008). *Patent No. EP1902926 - Variable Wheel Positioning in a Vehicle.* Europe.

[Bas17] BaSyTec. (2017). *CTS - Cell Test System*. Retrieved 05 18, 2017, from http://www.basytest.de/prospekte/CTS%202015_01.pdf

[Bec02] Beck, R. E. (2002). *Application of Control Allocation Methods to Linear Systems with Four or More Objectives.* Ph.D. dissertation, Virginia Polytechnic Institute and State University, Aerospace and Ocean Engineering, Blacksburg, USA.

[Bez17] Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A Fresh Approach to Numerical Computing. *SIAM Review , 59* (1), 65-98.

[Boe13] Böck, S., Brembeck, J., & Parzhuber, O. (2013). *Sensor-Analyse & -Modellierung für die zeitliche Synchronisierung der Sensorfusion im Forschungsfahrzeug ROboMObil.* Bachelor's thesis, Hochschule München, Munich, Germany.

[Bog14] Boegli, M. (2014). *Real-Time Moving Horizon Estimation for Advanced Motion Control - Application to Friction State and Parameter Estimation.* Ph.D. dissertation, KU Leuven, Faculty of Engineering Science, Leuven, Belgium.

[Boe08] Böhm, K. A. (2008). *Charakterisierung und Modellierung von elektrischen Energiespeichern für das Kfz.* Aachen: Shaker Media Verlag.

[Bon14] Bonvini, M., Wetter, M., & Sohn, M. D. (2014). An FMI-based Framework for State and Parameter Estimation. In H. Tummescheit, & K.-E. Arzen (Ed.), *Proceedings of 10th International Modelica Conference. 96*, pp. 647-656. Lund, Sweden: Linköping University Electronic Press.

[Boy04] Boyd, S., & Vandenberghe, L. (2004). *Convex Optimization.* New York, NY, USA: Cambridge University Press.

[Bra08] Braghin, F., Cheli, F., Melzi, S., & Sabbioni, E. (2008). Race Driver Model. (C. M. Soares, M. Bendsoe, K. Choi, & J. Herskovits, Eds.) *Computers and Structures , 86* (13-14), 1503-1516.

[Bre13] Brembeck, J. (2013). *Method to extend models for system design to models for system operation.* Mid-term report ITEA2 - MODRIO, DLR e.V., Weßling.

[Bre12] Brembeck, J., & Ritzer, P. (2012). Energy optimal control of an over actuated Robotic Electric Vehicle using enhanced control allocation approaches. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 322-327). Alcala de Henares, Spain: IVS IEEE.

[Bre11b] Brembeck, J., & Wielgos, S. (2011). A real time capable battery model for electric mobility applications using optimal estimation methods. *Proceedings of 8th International Modelica Conference* (pp. 398-405). Dresden, Germany: Linköping Electronic Conference Proceedings.

[Bre14] Brembeck, J., & Winter, C. (2014). Real-Time Capable Path Planning for Energy Management Systems in Future Vehicle Architectures. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 599-604). Dearborn, MI, USA: IVS IEEE.

[Bre11] Brembeck, J., Ho, L. M., Schaub, A., Satzger, C., Tobolar, J., Bals, J., et al. (2011). ROMO - The Robotic Electric Vehicle. *22nd IAVSD International Symposium on Dynamics of Vehicle on Roads and Tracks.* Manchester Metropolitan University: IAVSD.

[Bre11c] Brembeck, J., Otter, M., & Zimmer, D. (2011). Nonlinear Observers based on the Functional Mockup Interface with Applications to Electric Vehicles. *Proceedings of 8th International Modelica Conference* (pp. 474-483). Dresden, Germany: Linköping Electronic Conference Proceedings.

[Bre14b] Brembeck, J., Pfeiffer, A., Fleps-Dezasse, M., Otter, M., Wernersson, K., & Elmqvist, H. (2014). Nonlinear State Estimation with an Extended FMI 2.0 Co-Simulation Interface. In H. Tummescheit, & K.-E. Arzen (Ed.), *Proceedings of 10th International Modelica Conference. 96*, pp. 53-62. Lund: Linköping University Electronic Press.

[Brm15] Brembo S.p.A. (2015, 09 22). *Brembo präsentiert auf der IAA sein erstes In-Wheel projekt*. Retrieved 05 03, 2017, from http://www.brembo.com/de/company/news/in-wheel

[Brn73] Brent, R. P. (1973). *Algorithms for Minimization Without Derivatives.* Englewood Cliffs, New Jersery, United States: Prentice-Hall, Inc.

[Bry06] Bryant, E. (2006, 08 10). (Autoblog, Editor) Retrieved 04 21, 2017, from Siemens VDO announces eCorner motor-in-hub concept: http://www.autoblog.com/2006/08/10/siemens-vdo-announces-ecorner-motor-in-hub-concept/

[Buc08] Büchner, S. (2008). *Energiemanagement-Strategien für elektrische Energiebordnetze in Kraftfahrzeugen.* Technische Universität Dresden. Göttingen: Cuvillier Verlag.

[Bue98] Bünte, T. (1998). *Beiträge zur robusten Lenkregelung von Personenkraftwagen.* RWTH Aachen. Düsseldorf: VDI-Verlag.

[Bue11b] Bünte, T., & Chrisofakis, E. (2011). A Driver Model for Virtual Drivetrain Endurance Testing. *Proceedings of 8th International Modelica Conference* (pp. 180-188). Dresden, Germany: Linköping Electronic Conference Proceedings.

[Bus04] Buss, S. R., & Kim, J.-S. (2004, April). Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares methods. *IEEE Journal of Robotics and Automation* .

[Bus05] Buss, S. R., & Kim, J.-S. (2005). Selectively Damped Least Squares for Inverse Kinematics. *Journal of Graphics Tools , 10* (3), 37-49.

[Car10] Cardamone, L., Loiacono, D., Lanzi, P. L., & Bardelli, A. P. (2010). Searching for the optimal racing line using genetic algorithms. *Proceedings of the Conference on Computational Intelligence and Games* (pp. 388-394). Copenhagen, Denmark: IEEE.

[Com17] COMSOL. (n.d.). Retrieved 04 21, 2017, from Batteries and Fuel Cells Module: https://www.comsol.com/batteries-and-fuel-cells-module

[Dan10] Daniel, J., Birouche, A., Lauffenburger, J. P., & Basset, M. (2010). Energy constrained trajectory generation for ADAS. *Proceedings of the Intelligent Vehicles Symposium* (pp. 244-249). San Diego, USA: IVS IEEE.

[Dan11] Daniel, J., Birouche, A., Lauffenburger, J.-P., & Basset, M. (2011). Navigation-based constrained trajectory generation for advanced driver assistance systems. *International Journal of Vehicle Autonomous Systems , 9* (3-4), 269-296.

[Die01] Diehl, M. (2001). *Real-Time Optimization for Large Scale Nonlinear Processes.* Ph.D. dissertation, Universität Heidelberg, Fakultät für Mathematik und Informatik, Heidelberg, Germany.

[Dlr09] DLR e.V. (2009). *Machbarkeitsstudie zur Entwicklung eines eCorners.* Project report, Robotics and Mechatronics Center, Wessling, Germany.

[Don79] Dongarra, J. J., Moler, C. B., Bunch, J. R., & Stewart, G. W. (1979). *LINPACK User's Guide.* Philadelphia, PA: SIAM.

[Elm95] Elmqvist, H., Cellier, F., & Otter, M. (1995). Inline Integration: A New Mixed Symbolic/Numeric Approach for Solving Differential-Algebraic Equation Systems. In M. Snorek (Ed.), *European Simulation Multiconference*, (pp. 23-34). Prague, Czech Republic.

[Elm17] Elmqvist, H., Henningsson, T., & Otter, M. (2017). Innovations for Future Modelica. In J. Kofránek, & F. Casella (Ed.), *Proceedings of 12th International Modelica Conference* (pp. 693-702). Prague, Czech Republic: Linköping University Electronic Press.

[Eng10] Engst, C., Brembeck, J., & Kennel, R. (2010). *Object-Oriented Modelling and Real-Time Simulation of an Electric Vehicle in Modelica.* Master's thesis, Technische Universität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.

[Ens08] Engstle, A. (2008). *Energiemanagment in Hybridfahrzeugen.* Fakultät für Elektrotechnik und Informationstechnik der TU München, Energiewandlungstechnik. Aachen: Shaker.

[Enn13] Ennifar, H., Brembeck, J., Otter, M., & Kennel, R. (2013). *Integration of a Real-Time Battery Model in a RCP-System and its Implementation on a Research Platform.* Bachelor's thesis, Technische Unvierstität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.

[Fer12] Ferreau, H. J., Kraus, T., Vukov, M., Saeys, W., & Diehl, M. (2012). High-speed moving horizon estimation based on automatic code generation. *Proceedings of the 51st Annual Conference on Decision and Control (CDC)* (pp. 687-692). Maui, USA: IEEE.

[Fle13] Fleps-Dezasse, M., & Brembeck, J. (2013). Model based vertical dynamics estimation with Modelica and FMI. In T. Kawabe (Ed.), *Proceedings of the 7th IFAC Symposium on Advances in Automotive Control. 7,* pp. 341-346. Tokyo, Japan: Elsevier.

[Fro08] Fröschl, J., Knobel, C., Pröbstle, H., & Sirch, O. (2008). Energiebordnetz und elektrisches Energiemanagement der Zukunft. *ELKS 2008 - Elektrische Leistungsbordnetze und Komponenten von Straßenfahrzeugen* (pp. 111-124). Braunschweig: ITS Automotive Nord.

[Fun12] Funke, J., Theodosis, P., Hindiyeh, R., Stanek, G., Kritatakirana, K., Gerdes, C., et al. (2012). Up to the limits: Autonomous Audi TTS. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 541-547). Alcalá de Henares, Spain: IVS IEEE.

[Gas08] Gashi, R., & Laurent, D. (2008). *Patent No. US2008/0100020 - Vehicle Ground Connection Comprising A Wheel And A Suspension Integrated Therein.*

[Gol13] Golub, G. H., & Van Loan, C. F. (2013). *Matrix Computations* (4th ed.). Baltimore, USA: Johns Hopkins University Press.

[Gra02] Graaf, R. (2010). *Simulation hybrider Antriebskonzepte mit Kurzzeitspeicher für Kraftfahrzeuge.* Ph.D. dissertation, RWTH Aachen, Institut für Kraftfahrzeuge, Aachen, Germany.

[Gre15] Grewal, M. S., & Andrews, A. P. (2015). *Kalman filtering: theory and practice using MATLAB* (4. ed.). Hoboken, New Jersey: John Wiley & Sons Inc.

[Gru17] Grüne, L., & Pannek, J. (2017). *Nonlinear Model Predictive Control - Theory and Algorithms* (2. ed.). London, England: Springer.

[Gug08] Guglielmino, E., Sireteanu, T., Stammers, C. W., Ghita, G., & Giuclea, M. (2008). *Semi-active suspension control - Improved vehicle ride and road friendliness* (1. ed.). London, England: Springer.

[Hae03] Härkegard, O. (2003). *Backstepping and Control Allocation with Applications to Flight Control.* Ph.D. dissertation, Linköping University, Department of Electrical Engineering, Linköping, Sweden.

[Har11] Hartikainen, J., Solin, A., & Särkkä, S. (2011, August 16). *Optimal filtering with Kalman filters and smoothers - A Manual for Matlab toolbox EKF/UKF.* Retrieved 04 22, 2017, from http://www.lce.hut.fi/research/mm/ekfukf/

[Har10] Hartweg, S. (2010). *Einfluss elastischer Strukturen auf Fahrdynamikregelysteme.* Master's thesis, RWTH Aachen, Institut für Kraftfahrzeuge (IKA), Aachen, Germany.

[Hay01] Haykin, S. (2001). *Kalman Filtering and Neural Networks* (1. ed.). New York, USA: Wiley-Interscience.

[Hec15] Heckmann, A., & Schneider, S. (2015). *Brakes and Observers - use cases from the railway field.* DLR e.V. and Knorr Bremse. ITEA2 - MODRIO.

[Hel14] Hellerer, M., Bellmann, T., & Schlegel, F. (2014). The DLR Visualization Library - Recent development and applications. *Proceedings of 10th International Modelica Conference* (pp. 899-911). Lund, Sweden: Linköping University Electronic Press.

[Hin05] Hindmarsh, A. C., Brown, P. N., Grant, K. E., Lee, S. L., Serban, R., & Shumaker, D. E. (2005). SUNDIALS: Suite of Nonlinear and Differential/Algebraic. *ACM Transactions on Mathematical Software 31(3)* , pp. 363-369.

[Hir08] Hirschmüller, H. (2008). Stereo Processing by Semiglobal Matching and Mutual Information. *IEEE Transactions on Pattern Analysis and Machine Intelligence , 30* (2), 328-341.

[Ho16] Ho, L. M., & Bünte, T. (2016). Sensor Fault Detection and Isolation using a Non-linear Planar Kinematic Model of Vehicle Dynamics. *Proceedings of the IEEE Multi-Conference on Systems and Control (CCA)* (pp. 297-304). Buenos Aires, Argentine: IEEE.

[Hor83] Horn, B. K. (1983). The Curve of Least Energy. *ACM Transactions on Mathematical Software , 9* (4), 441-460.

[Hou10] Houska, B., Ferreau, H. J., & Diehl, M. (2010). ACADO toolkit - An open-source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods , 32* (3), 298-312.

[Iso88] International Organization for Standardization. (1988). ISO 7401. *Road vehicles: Lateral transient response test methods* . International Organization for Standardization.

[Ite12] ITEA. (2012, September). *11004 Modrio*. Retrieved 04 23, 2017, from https://itea3.org/project/modrio.html

[Jrc13] Joint Research Center - Umweltbundesamt. (2013, 09). *Treibgasemissionen verschiedener Kraftstoffe und Antriebsarten.* Retrieved 04 22, 2017, from http://www.goingelectric.de/forum/resources/co2-bilanz/9191

[Jon07] Jonasson, M. (2007). *Aspects of Autonomous Corner Modules as an Enabler for New Vehicle Chassis Solutions.* Licentiate thesis, KTH Royal Institute of Technology, Stockholm.

[Jon09] Jonasson, M. (2009). *Exploiting individual wheel actuators to enhance vehicle dynamics and safety in electric vehicles.* Ph.D. dissertation, KTH Royal Institute of Technology, Stockholm, Sweden.

[Joo08] Joos, H.-D., Bals, J., Looye, G., Schnepper, K., & Varga, A. (2008). MOPS: Eine integrierte optimierungsbasierte Entwurfsumgebung für mehrzielige, parametrische Analyse und Synthese. DGLR Workshop Systemidentifizierug, Parameterschätzung und Optimierung.

[Jul04] Julier, S. J., & Uhlmann, J. K. (2004). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE , 92* (3), 401-422.

[Kam11] Kamil Çagatay Bayindir, M. A. (2011). A comprehensive overview of hybrid electric vehicle: Powertrain configurations, powertrain control techniques and electronic control units. *Energy Conversion and Management , 52* (2), 1305-1313.

[Kan08] Kandepu, R., Imsland, L., & Foss, B. (2008). Constrained state estimation using the Unscented Kalman Filter. *Proceedings of the 16th Mediterranean Conference on Control and Automation* (pp. 1453-1458). Ajaccio, France: IEEE.

[Kno08] Knobel, C. (2008). *Optimal Control Allocation for Road Vehicle Dynamics using Wheel Steer Angles, Brake/Drive Torques, Wheel Loads and Camber Angles.* Ph.D. dissertation, Technischen Universität München, Maschinenwesen, Munich, Germany.

[Kno16] Knobel, C., Pöltenstein, A., & Knoller, S. (2016). Vorausschauende EfficientDynamics-Funktionen. *Automobiltechnische Zeitschrift , 04* (118), 16-21.

[Koc10] Koch, G., Kloiber, T., Pellegrini, E., & Lohmann, B. (2010). A nonlinear estimator concept for active vehicle suspension control. *Proceedings of the American Control Conf. (ACC)* (pp. 4576-4581). Baltimore, USA: IEEE.

[Kou16] Kouzoupis, D., Quirynen, R., Girrbach, F., & Diehl, M. (2016). An efficient SQP algorithm for Moving Horizon Estimation with Huber penalties and multi-rate measurements. *Proceedings of the Conference on Control Applications (CCA)* (pp. 1482-1487). Buenos Aires, Argentina: IEEE.

[Kue11] Kühl, P., Diehl, M., Kraus, T., Schlöder, J. P., & Bock, H. G. (2011). A real-time algorithm for moving horizon state and parameter estimation. *Journal of Computers & Chemical Engineering , 35* (1), 71-83.

[Lai07] Laine, L., & Fredriksson, J. (2007). Coordination of Vehicle Motion and Energy Management Control Systems for Wheel Motor Driven Vehicles. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 773-780). Istanbul, Turkey: IVS IEEE.

[Lar98] Larsen, T. D., Andersen, N. A., Ravn, O., & Poulsen, N. K. (1998). Incorporation of time delayed measurements in a discrete-time Kalman filter. *Proceedings of the 37th IEEE Conference on Decision and Control. 4*, pp. 3972-3977. Tampa, Florida, USA: IEEE.

[Lju98] Ljung, L. (1998). *System Identification: Theory for the User* (2nd ed.). Linköping University, Sweden: Prentice Hall PTR.

[Ma12] Ma, L., Yang, J., & Zhang, M. (2012). A Two-level Path Planning Method for On-road Autonomous Driving. *International Conference on Intelligent System Design and Engineering Application (ISDEA)* (pp. 661-664). Sanya, China: IEEE.

[Mer04b] Merwe, R. v. (2004). *Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models.* Ph.D. dissertation, Oregon Health & Science University, Department of Electrical and Computer Engineering, Oregon, USA.

[Mer01] Merwe, R. v., & Wan, E. (2001). The square-root unscented Kalman filter for state and parameter-estimation. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing. 6,* pp. 3461-3464. Salt Lake City, Utah, USA: IEEE.

[Mer04] Merwe, R. v., Wan, E. A., & Julier, S. (2004). Sigma-Point Kalman Filters for Nonlinear Estimation and Sensor-Fusion: Applications to Integrated Navigation. *Proceedings of the AIAA Guidance, Navigation & Control Conference* (pp. 16-19). Providence, USA: AIAA.

[Mic08] Michaud, S., Gibbesch, A., Thueer, T., Krebs, A., Lee, C., Despont, B., et al. (2008). Development of the ExoMars Chassis and Locomotion Subsystem. *Proceedings of the 9th International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSairas).* Los Angeles, USA: Eidgenössische Technische Hochschule Zürich, Autonomous Systems Lab .

[Mil08] Miles, R., & Hamilton, K. (2008). *Learning UML 2.0 - A Pragmatic Introduction to UML.* Sebastopol, USA: O'Reilly Media.

[Mit04] Mitschke, M., & Wallentowitz, H. (2004). *Dynamik der Kraftfahrzeuge* (4. ed.). Berlin: Springer.

[Mod13] Modelica Association. (2013, 10 18). *Functional Mockup Interface for Model Exchange and Co-Simulation 2.0 RC1*. Retrieved 11 30, 2013, from https://www.fmi-standard.org/downloads#version2

[Mod10] Modelica Association. (2010, January). *Functional Mock-up Interface for Model Exchange,Version 1.0. MODELISAR (07006)*. Retrieved 12 04, 2017, from http://fmi-standard.org/downloads/

[Mod17] Modelica Association. (2017). *Modelica*. Retrieved 04 23, 2017, from http://www.modelica.org

[Mab17] Modelon AB. (2017, March). *PyFMI 2.4*. Retrieved 04 2017, from https://pypi.python.org/pypi/PyFMI

[Mor08] Morin, P., & Claude, S. (2008). Motion Control of Wheeled Mobile Robots. In B. Siciliano, O. Khatib, & K. Siciliano (Ed.), *Handbook of Robotics* (pp. 799-826). Springer.

[Nor04] Nørgaard, M., Poulsen, N. K., & Ravn, O. (1998). *Advances in Derivative-Free State Estimation for Nonlinear Systems.* Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby: Informatics and Mathematical Modelling, Technical University of Denmark, DTU.

[Ong98] Ong, C.-M. (1998). *Dynamic Simulation of Electric Machinery* (1. ed.). Indiana, UAS: Prentice Hall.

[Ott12] Otter, M., Thiele, B., & Elmquvist, H. (2012). A Library for Synchronous Control Systems in Modelica. *Proceedings of the 9th International Modelica Conference* (pp. 27-36). Munich, Germany: Linköping Electronic Conference Proceedings.

[Pac12] Pacejka, H. (2012). *Tire and Vehicle Dynamics* (3. ed.). Netherlands: Elsevier Ltd.

[Pad16] Paden, B., Čáp, M., Yong, S. Z., Yershov, D. S., & Frazzoli, E. (2016). A Survey of Motion Planning and Control Techniques for Self-driving Urban Vehicles. *The Computing Research Repository (CoRR) , abs/1604.07446*.

[Pfe12] Pfeiffer, A. (2012). Optimization Library for Interactive Multi-Criteria Optimization Tasks. *Proceedings of the 9th International Modelica Conference* (pp. 669-679). Munich, Germany: Linköping Electronic Conference Proceedings.

[Ple04b] Plett, G. L. (2004). Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs: Part 1. Background. *Journal of Power Sources , 134* (2), 252-261.

[Ple04c] Plett, G. L. (2004). Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs: Part 2. Modeling and identification. *Journal of Power Sources , 134* (2), 262-276.

[Ple04d] Plett, G. L. (2004). Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs: Part 3. State and parameter estimation. *Journal of Power Sources , 134* (2), 277-292.

[Ple04] Plett, G. L. (2004). High-performance battery-pack power estimation using a dynamic cell model. (IEEE, Ed.) *IEEE Transactions on Vehicular Technology , 53* (5), 1586-1593.

[Poh05] Pohl, S., & Gräf, M. (2005). Dynamic Simulation of a Free-Piston Linear Alternator in Modelica. *Proceedings of the 4th International Modelica Conference* (pp. 393-399). Hamburg, Germany: The Modelica Association.

[Ril01] Rill, G. (2001). *Road Vehicle Dynamics: Fundamentals and Modeling.* (G. Rill, Ed.) Regensburg, Germany: CRC Press.

[Rit16] Ritzer, P., Panzirsch, M., & Brembeck, J. (2016). Robotic Motion - Interactive motion simulation of vehicle dynamics. (B. Schäfers-Maiwald, Ed.) *dSPACE Magazin , 01*, 52-57.

[Rit15] Ritzer, P., Winter, C., & Brembeck, J. (2015). Advanced Path Following Control of an Overactuated Robotic Vehicle. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 1120-1125). Seoul, South Korea: IVS IEEE.

[Rit16b] Ritzer, P., Winter, C., & Brembeck, J. (2016). Experimental Validation of Geometric Path Following Control with Demand Supervision on an Over-Actuated Robotic Vehicle. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 539-545). Gothenborg, Sweden: IVS IEEE.

[Ros07] Rosario, L., & Luk, P. (2007). Applying Management Methodology to Electric Vehicles with Multiple Energy Storage Systems. *Proccedings of the 2007 International Conference on Machine Learning and Cybernetics. 7*, pp. 4223-4230. Hong Kong, China: IEEE.

[Ros60] Rosen, J. (1960). The Gradient Projection Method for Nonlinear Programming. Part I. Linear Constraints. *Journal of the Society for Industrial and Applied Mathematics , 8* (1), 181-217.

[Sat16] Satzger, C., de Castro, R., Knoblach, A., & Brembeck, J. (2016). Design and Validation of an MPC-based Torque Blending and Wheel Slip Control Strategy. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 514-520). Gothenburg, Sweden: IVS IEEE.

[Sav10] Savaresi, S., Poussot-Vassal, C., Spelta, C., Sename, O., & Dugard, L. (2010). *Semi-Active Suspension Control Design for Vehicles* (1. ed.). Oxford, UK: Elsevier Science.

[Sch14] Schaeffler Technologies AG & Co. KG . (2014). *E-Wheel Drive Radnabenantrieb*. Retrieved 05 02, 2017, from http://m.schaeffler.de/content.mobile.products/de/products/automotive/e_mobility/e_wheel_drive/e_wheel_drive_info.html

[Sca16] Schaub, A., Baumgartner, D., & Burschka, D. (2016). Reactive Obstacle Avoidance for Highly Maneuverable Vehicles Based on a Two-Stage Optical Flow Clustering. *IEEE Transactions on Intelligent Transportation Systems , PP* (99), 1-16.

[Sca11] Schaub, A., Brembeck, J., Burschka, D., & Hirzinger, G. (2011). Robotic Electric Vehicle with Camera-based Autonomy Approach. *ATZelektronik , 2* (2), 10-16.

[Sch05] Schittkowski, K. (2005, July). *QL: A Fortran Code for Convex Quadratic Programming - User's Guide, Version 2.11 -*. Retrieved 04 24, 2017, from http://www.ai7.uni-bayreuth.de/ql_rep.htm

[Scl15] Schlabe, D. (2015). *Modellbasierte Entwicklung von Energiemanagement-Methoden für Flugzeug-Energiesysteme.* Technische Universität Dresden, Fakultät Elektrotechnik und Informationstechnik. Dresden, Germany: VDI Verlag.

[Scm07] Schmitt, P. (2007). *Just build it! : a fully functional concept vehicle using robotic wheels.* Master's thesis, Massachusetts Institute of Technology, School of Architecture and Planning, Program in Media Arts and Sciences, Massachusetts, USA.

[Scn11] Schneider, P. (2011). *Sigma-Punkt Kalman-Filter mit Ungleichungsnebenbedingungen.* Universität des Saarlandes, Fakultät 7 - Physik und Mecahtronik. Saarbrücken: Logos Verlag Berlin GmbH.

[Scn09] Schneider, P., & Janocha, H. (2009). Sigma-Punkt-Kalman-Filter mit Zustandsnebenbedingungen (Sigma-Point Kalman Filter with State Constraints). (G. Bretthauer, Ed.) *Automatisierungstechnik , 57* (4), 169-176.

[Scu15] Schuller, J., & Buhlmann, M. (2015). Hochintegrationssteuergerät für Fahrwerkregelfunktionen. *ATZ - Automobiltechnische Zeitschrift , 117* (10), 36-41.

[See16] Seefried, A., & Pfeiffer, A. (2016). Rapid-Prototyping of NMPC in Modelica. *Proceedings of the 7th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools* (pp. 41-44). Milano, Italy: ACM Digital Library.

[Seg07] Seeger, M. (2007). *Low rank updates for the Cholesky decomposition.* Technical report, University of California at Berkeley, Department of EECS, Berkeley, USA.

[Sim10] Simon, D. (2010). Kalman filtering with state constraints: a survey of linear and nonlinear algorithms. *IET Control Theory & Applications , 4* (8), 1303-1318.

[Sim06] Simon, D. (2006). *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches* (1. ed.). Cleveland, USA: Wiley & Sons.

[Sti08] Stiftl, J. (2008). *Modellierung und Bewertung von Fahrzeugen mit seriellem Plug-In Hybridantrieb.* Diploma thesis, Hochschule Karlsruhe – Technik und Wirtschaft, Karlsruhe, Germany.

[The07] Theuerkauf, H., & Schmidt, M. (2007). Ein neues Energiemanagement-Konzept für das elektrische Bordnetz. *ATZ - Automobiltechnische Zeitschrift , 109* (01), 10-15.

[Thu05] Thummel, M., Looye, G., Kurze, M., Otter, M., & Bals, J. (2005). Nonlinear Inverse Models for Control. *Proceedings of the 4th International Modelica Conference,* (pp. 267-279). Hamburg, Germany: The Modelica Association.

[Tob16] Tobolar, J., de Castro, R. P., Bleck, U., Satzger, C., Brembeck, J., & Hirano, Y. (2016). Comparative evaluation of energy efficiency of electrical vehicle powertrain configurations. *Proceedings of the 24th Symposium of the International Association for Vehicle System Dynamics (IAVSD 2015). 1*, pp. 691-700. Graz, Austria: CRC Press.

[Tob07] Tobolar, J., Otter, M., & Bünte, T. (2007). Modelling of Vehicle Powertrains with the Modelica PowerTrain Library. In A. Laschet, *Systemanalyse in der Kfz-Antriebstechnik IV* (pp. 204-216). Augsburg: Haus der Technik.

[Toe08] Töpler, F., Anthony, P., Langhammer, S. a., & Köhle, S. (2008). Hybridbetriebsstrategien mit elektronischem Horizont:. *Proceedings of the 17. Aachener Kolloquium: Fahrzeug- und Motorentechnik* . Aachen, Germany: fka.

[Trz08] Trzesniowski, M. (2008). *Rennwagentechnik: Grundlagen, Konstruktion, Komponenten, Systeme* (1. ed.). Wiesbaden, Germany: Vieweg+Teubner Verlag.

[Val11] Valencia, F., Lopez, J., Marquez, A., & Espinosa, J. (2011). Moving horizon estimator for measurement delay compensation in model predictive control schemes. *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)* (pp. 6678-6683). Orlanda, USA: IEEE.

[Vir17] VIRES Simulationstechnologie GmbH. (2017). *OpenDRIVE - Format specification*. Retrieved 04 10, 2017, from http://www.opendrive.org/docs/OpenDRIVEFormatSpecRev1.4H.pdf

[Vir17b] VIRES Simulationstechnologie GmbH. (2017). *VIRES Virtual Test Drive*. Retrieved 04 10, 2017, from https://www.vires.com/products.html

[Vöt17] Vötsch. (2017). *Temperature test chambers* . Retrieved 05 18, 2017, from http://www.v-it.com/sixcms/media.php/2335/ VIT_Laboratory%20Temperature%20test%20chambers%5B1%5D.pdf

[Vuk15] Vukov, M. (2014). *Embedded Model Predictive Control and Moving Horizon Estimation for Mechatronics Applications.* KU Leuven, Faculty of Engineering Science. Leuven, Belgium: KU Leuven – Faculty of Engineering Science.

[Wie10] Wielgos, S., Brembeck, J., Otter, M., & Kennel, R. (2010). *Development of an Energy Management System for Electric Vehicles Design and System Simulation.* Master's thesis, Technische Universität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.

[Wik17] WIKIPEDIA. (2017). *Better Place - Insolvency in 2013*. Retrieved 04 21, 2017, from https://de.wikipedia.org/wiki/Better_Place

[Wik17b] WIKIPEDIDA. (2017). *Lithium-Eisenphosphat-Akkumulator*. Retrieved 05 17, 2017, from https://de.wikipedia.org/wiki/Lithium-Eisenphosphat-Akkumulator

[Wid08] Wilde, A., Schneider, J., & Herzog, H.-G. (2008). Fahrstil- und fahrsituationsabhängige Ladestrategie bei Hybridfahrzeugen. *ATZ - Automobiltechnische Zeitschrift , 05*, 412–421.

[Wid09] Wilde, D. K. (2009). Computing clothoid segments for trajectory generation. *Proccedings of the International Conference on Intelligent Robots and Systems* (pp. 2440-2445). St. Louis, USA: IEEE.

[Wil94] Williams, R. (1994). Electronically controlled automotive suspensions. *Computing Control Engineering Journal , 5* (3), 143-148.

[Win17] Winter, C., & de Castro, D. R. (2017). Optimal Velocity Profile Generation for Semi-Autonomous Vehicles. *VDI Berichte. 2292*, pp. 129-141. Berlin, Germany: VDI Wissensforum GmbH.

[Win13] Winter, C., Brembeck, J., & Kennel, R. (2013). *Online Energy Optimal Path Planner for Advanced Electric Vehicles.* Master's Thesis, Technische Universität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.

[Won03] Won, J. S., & Langari, R. (2003). Intelligent energy management agent for a parallel hybrid vehicle. *Proceedings of the 2003 American Control Conference. 6*, pp. 2560-2565. Denver, USA: IEEE.

Appendix

A Discrete-Time State Estimation Theory

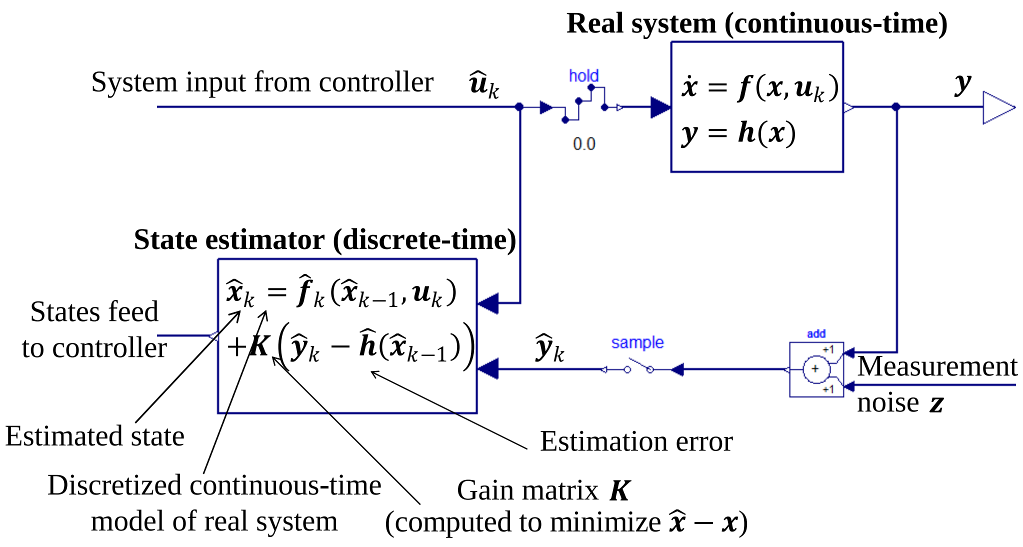
For many of today's control tasks like model predictive or state feedback control, the knowledge of the actual full state vector of the controller's synthesis model is necessary. In most of the control applications not all states can be measured directly via sensors. This might be reducible to the unavailability (e.g. no sensor method for direct state of charge measurement of a lithium-ion cell) or cost considerations (e.g. force-momentum sensors in each axis of an industrial robot). Therefore, the missing states must be reconstructed from the available measurement (e.g. motor current and angular velocity of a robot axis) in combination with the knowledge of the system dynamics of the controlled plant. Since nowadays nearly all controllers are executed on real-time capable microcontrollers, the estimation algorithms must be performed in discrete-time. Thus the next chapters focus on methods for discrete-time state estimation; the continuous-time theory of state estimation is not considered here. Moreover, the outline focuses on Kalman filter based algorithms which are widely used in applications and have shown good results for causal systems in the last decades e.g. [Sim06].

A.1 Content of this Appendix Chapter

Beginning from the principal need of state estimation and the sketch of the configuration on real world microcontrollers in Chapter A.2, the appendix is organized as follows: In Chapter A.3 an introduction to recursive estimation techniques is given. Recursive means, in every time step only the information from the last one is taken into account for the estimation algorithm. This technique is frequently used due to limited memory and computing capacity of most common microcontrollers. Here the connection between (recursive) weighted least squares estimation of a constant quantity and the (linear) Kalman filter technique for state estimation is explained. In the later chapters this interconnection becomes very interesting for more complex methods like the moving horizon estimation (see Chapter A.5). This linear Kalman filter is then extended for nonlinear applications with two different types of approaches: on the one hand algorithms that need Jacobians and on the other hand derivative-free methods are explained.

A.2 Basic Structure of an Estimation Setup

In App. Figure A.1 a signal flow diagram in Modelica style is sketched that shows the configuration of a state estimator in a discrete-time system (e.g. on a microcontroller). The controller's system input $\boldsymbol{u}_k$ is commanded to the real system (by help of a digital to analog (D/A) converter) and also directed to the inputs of the prediction model of the state estimator. Note that it is always assumed that the inputs have no direct feed through to the outputs; this is a valid premise for real physical systems and controlled plants. Due to the input signal and the system dynamics an output $\boldsymbol{y}$ of the real system can be measured that may be disturbed by a measurement noise $\boldsymbol{z}$. To estimate the states of the real system, the error between measurement and prediction needs to be minimized. This is coped via a calculation of the gain $\boldsymbol{K}$ in the estimator algorithm. It is determined in a way that the error between $\hat{\boldsymbol{x}} - \boldsymbol{x}$ is minimized, under statistical assumption consideration of the prediction model and the measured values.


Real system (continuous-time)
System input from controller $\hat{\boldsymbol{u}}_k$
hold
0.0
$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}_k)$
$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x})$
$\boldsymbol{y}$
State estimator (discrete-time)
States feed to controller
$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{f}}_k(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k) + \boldsymbol{K}\left(\hat{\boldsymbol{y}}_k - \hat{\boldsymbol{h}}(\hat{\boldsymbol{x}}_{k-1})\right)$
$\hat{\boldsymbol{y}}_k$
sample
add
+1
+1
Measurement noise $\boldsymbol{z}$
Estimated state
Estimation error
Discretized continuous-time model of real system
Gain matrix $\boldsymbol{K}$ (computed to minimize $\hat{\boldsymbol{x}} - \boldsymbol{x}$)

App. Figure A.1: State estimator in a discrete-time environment

A.3 Recursive State Estimation Algorithms

In this chapter, the principle ideas of recursive state estimation are summarized and its (historical) development leading to the Kalman filter is outlined. Background information, alternative formulations, and recent developments are provided in the standard textbook [Sim06] that is also the starting point for the following explanations. Parts of the Chapter A.3.1 to Chapter A.3.3 have also been published in [Bre11c] and [Bre13].

A.3.1 Weighted Least Squares Estimation

First, an estimation of a constant quantity based on several noisy measurements is considered. This weighted least squares estimation (WLS) problem is well-known in system identification tasks (see, e.g. [Lju98]). Through the weighted formulation, the control engineer can assign different levels of confidence to certain measurements (or observations). This feature is crucial for the tuning of Kalman filters. The corresponding minimization problem is formulated as follows:

$$\begin{bmatrix} y_1^{\mathrm{m}} \\ \vdots \\ y_k^{\mathrm{m}} \end{bmatrix} = \begin{bmatrix} H_{11} & \cdots & H_{1n} \\ \vdots & \ddots & \vdots \\ H_{kn} & \cdots & H_{kn} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} v_1 \\ \vdots \\ v_k \end{bmatrix}$$
$$E\left(v_i^2\right) = \sigma_i^2 \quad (i = 1, \ldots, k)$$
$$\boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{y} \in \mathbb{R}^k, \boldsymbol{v} \in \mathbb{R}^k \tag{A.1}$$

The unknown vector $\boldsymbol{x}$ is constant and consists of n elements, $\boldsymbol{y}^{\mathrm{m}}$ is a k-element noisy measurement vector and usually $k \gg n$. Each element y_i^{m} of $\boldsymbol{y}^{\mathrm{m}}$ is a linear combination ($\boldsymbol{H}_{k*}$) with the unknown vector $\boldsymbol{x}$ and the variance of the measurement noise of the i-th measurement v_i. The noise of each measurement is zero-mean and independent from each other, therefore the measurement covariance matrix is

$$\boldsymbol{R} = \mathrm{E}(\boldsymbol{v}\boldsymbol{v}^T) = \mathrm{diag}(\boldsymbol{\sigma}_1^2, \dots, \boldsymbol{\sigma}_k^2) \tag{A.2}$$

The residual

$$\boldsymbol{\epsilon_y} = \underbrace{(\boldsymbol{Hx} + \boldsymbol{v})}_{=\boldsymbol{y}^{\mathrm{m}}} - \underbrace{\boldsymbol{H\hat{x}}}_{\hat{\boldsymbol{y}}} \tag{A.3}$$

is the difference of all measured values $\boldsymbol{y}^{\mathrm{m}}$ to the (unknown) $\boldsymbol{x}$-vector minus the estimated vector $\hat{\boldsymbol{y}}$ that is computed from the estimated vector $\hat{\boldsymbol{x}}$. The goal is to compute the estimated vector $\hat{\boldsymbol{x}}$ such that the weighted residual is as small as possible, i.e. to minimize the cost function J:

$$J = \frac{\epsilon_{y1}^2}{\sigma_1^2} + \dots + \frac{\epsilon_{yk}^2}{\sigma_k^2} \tag{A.4}$$

To minimize J, it is useful to compute the partial derivative with respect to the estimated $\hat{\boldsymbol{x}}$-vector which is afterwards set equal to zero. In this way, an optimal solution for $\hat{\boldsymbol{x}}$ can be calculated:

$$\begin{aligned} \frac{\partial J}{\partial \hat{\boldsymbol{x}}} &= 2 \cdot (-(\boldsymbol{y}^{\mathrm{m}})^T \boldsymbol{R}^{-1}\boldsymbol{H} + \hat{\boldsymbol{x}}^T \boldsymbol{H}^T \boldsymbol{R}^{-1} \boldsymbol{H}) = 0 \\ \hat{\boldsymbol{x}} &= (\boldsymbol{H}^T \boldsymbol{R}^{-1} \boldsymbol{H})^{-1} \boldsymbol{H}^T \boldsymbol{R}^{-1} \boldsymbol{y}^{\mathrm{m}} \end{aligned} \tag{A.5}$$

Eq. (A.5) requires that $\boldsymbol{R}$ is non-singular and $\boldsymbol{H}$ has full rank. This is the “textbook” version of the algorithm. It is inefficient and numerically not reliable. Alternatively, eq. (A.4) can be formulated as:

$$J = \begin{bmatrix} \frac{\epsilon_{y1}}{\sigma_1} & \cdots & \frac{\epsilon_{yk}}{\sigma_k} \end{bmatrix} \cdot \begin{bmatrix} \frac{\epsilon_{y1}}{\sigma_1} \\ \vdots \\ \frac{\epsilon_{yk}}{\sigma_k} \end{bmatrix} \tag{A.6}$$

To solve it more efficiently, the problem is transformed to a standard linear least squares problem that minimizes the 2-norm of the weighted residue vector:

$$\begin{aligned} &\min_{\hat{\boldsymbol{x}}} \left\| \begin{bmatrix} \frac{\epsilon_{y1}}{\sigma_1} & \cdots & \frac{\epsilon_{yk}}{\sigma_k} \end{bmatrix} \right\|^2 \\ &= \min_{\hat{\boldsymbol{x}}} \|\boldsymbol{W}(\boldsymbol{y} - \boldsymbol{H}\hat{\boldsymbol{x}})\|^2 \\ &= \min_{\hat{\boldsymbol{x}}} \|\boldsymbol{WH}\hat{\boldsymbol{x}} - \boldsymbol{W}\boldsymbol{y}^{\mathrm{m}}\|^2 \\ &= \min_{\hat{\boldsymbol{x}}} \|\boldsymbol{A}\hat{\boldsymbol{x}} - \boldsymbol{b}\|^2 \\ &\boldsymbol{W} = \mathrm{diag}(1/\sigma_1, \dots, 1/\sigma_k) \end{aligned} \tag{A.7}$$

This minimization problem has a unique solution, if $\boldsymbol{A}=\boldsymbol{WH}$ has full rank. If $\mathbf{A}$ is rank deficient, an infinite number of $\widehat{\boldsymbol{x}}$ solutions exists. The usual approach is to select from the infinite number of solutions the unique one that additionally minimizes the norm of the solution vector: $\min\|\widehat{\boldsymbol{x}}\|^2$. Given $\boldsymbol{A}=\boldsymbol{WH}$ and $\boldsymbol{b}=\boldsymbol{W}\boldsymbol{y}^{\mathrm{m}}$, this solution vector can be computed with a least squares solver like the LAPACK function DGELSX [And99]. This function uses a QR-decomposition of $\boldsymbol{A}$ with column pivoting together with right multiplication of an orthogonal matrix $\boldsymbol{Z}$ to arrive at:

$$\min_{\widehat{\boldsymbol{x}}} \left\| [\boldsymbol{Q}_1 \quad \boldsymbol{Q}_2] \begin{bmatrix} \boldsymbol{U} & 0 \\ 0 & 0 \end{bmatrix} \boldsymbol{ZP}\widehat{\boldsymbol{x}} - \boldsymbol{b} \right\|^2 \tag{A.8}$$

where $\boldsymbol{Q}$ and $\boldsymbol{Z}$ are orthogonal matrices, $\boldsymbol{P}$ is a permutation matrix, $\boldsymbol{U}$ is a regular, upper triangular matrix and the dimension of the quadratic matrix $\boldsymbol{U}$ is identical to the rank of $\boldsymbol{A}$. Since the norm of a vector is invariant against orthogonal transformations, this equation can be transformed to:

$$\min_{\widehat{\boldsymbol{x}}} \left\| \begin{bmatrix} \boldsymbol{U} & 0 \\ 0 & 0 \end{bmatrix} \boldsymbol{ZP}\widehat{\boldsymbol{x}} - \begin{bmatrix} \boldsymbol{Q}_1^T\boldsymbol{b} \\ \boldsymbol{Q}_2^T\boldsymbol{b} \end{bmatrix} \right\|^2 \tag{A.9}$$

This is equivalent to

$$\min_{\widehat{\boldsymbol{x}}} \left\| \begin{bmatrix} \boldsymbol{U} \\ 0 \end{bmatrix} \overline{\widehat{\boldsymbol{x}}}_1 - \begin{bmatrix} \boldsymbol{Q}_1^T\boldsymbol{b} \\ \boldsymbol{Q}_2^T\boldsymbol{b} \end{bmatrix} \right\|, \overline{\widehat{\boldsymbol{x}}} = \boldsymbol{ZP}\widehat{\boldsymbol{x}}^2 \tag{A.10}$$

from which the solution can be directly computed as (taking into account $\boldsymbol{b}=\boldsymbol{W}\boldsymbol{y}^{\mathrm{m}}$):

$$\widehat{\boldsymbol{x}} = \boldsymbol{P}\boldsymbol{Z}^T\boldsymbol{U}^{-1}\boldsymbol{Q}_1^T\boldsymbol{W}\boldsymbol{y}^{\mathrm{m}} \tag{A.11}$$

In the following, only "textbook-style" versions of algorithms will be shown, such as eq. (A.5). Their implementation is performed in an efficient and numerically reliable way, such as eq. (A.11), where matrices $\boldsymbol{R}$ and $\boldsymbol{H}$ can be rank deficient. In the sketched approach, both eq. (A.5) and eq. (A.11), can be used for offline estimation with a predetermined number of measurements k. In real-time applications, new measurements arrive at each sample period to improve the estimation. Using eq. (A.11) would require a complete recalculation with $O(k^3)$-flops. One approach could be to use a moving horizon and dismiss older measurements (still computational extensive – discussed in Chapter A.5). Another option is to recursively reformulate the problem into a form that is updated at every sample instant with the new measurements. It follows that a linear recursive estimator can be written in the subsequent representation:

$$\begin{aligned} \boldsymbol{y}_k &= \boldsymbol{H}_k\boldsymbol{x} + \boldsymbol{v}_k \\ \widehat{\boldsymbol{x}}_k &= \widehat{\boldsymbol{x}}_{k-1} + \boldsymbol{K}_k \cdot (\boldsymbol{y}_k^{\mathrm{m}} - \boldsymbol{H}_k\widehat{\boldsymbol{x}}_{k-1}) \end{aligned} \tag{A.12}$$

The estimate $\widehat{\boldsymbol{x}}_k$ is computed based on the estimation from the last time step $\widehat{\boldsymbol{x}}_{k-1}$ and the information from the new measurement $\boldsymbol{y}_k^{\mathrm{m}}$. $\boldsymbol{K}_k$ is the estimator gain vector that weights the correction term $\boldsymbol{y}_k^{\mathrm{m}} - \boldsymbol{H}_k\widehat{\boldsymbol{x}}_{k-1}$. Hence, the optimal gain $\boldsymbol{K}_k$ needs to be computed in a recursive way. To achieve this, it is necessary to formulate another cost function that minimizes the covariance in a recursive way.

$$\frac{\partial J_k}{\partial \boldsymbol{K}_k} = \frac{\partial \mathrm{Tr}\boldsymbol{P}_k}{\partial \boldsymbol{K}_k} = 0 \tag{A.13}$$

$$\begin{aligned}\boldsymbol{P}_k &= (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_k)\boldsymbol{P}_{k-1}(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_K)^T \\ &+\boldsymbol{K}_k\boldsymbol{R}_k\boldsymbol{K}_k^T\end{aligned} \tag{A.14}$$

$$\boldsymbol{K}_k = \boldsymbol{P}_{k-1}\boldsymbol{H}_k^T \cdot \left(\boldsymbol{H}_k\boldsymbol{P}_{k-1}\boldsymbol{H}_k^T + \boldsymbol{R}_k\right)^{-1} \tag{A.15}$$

This results in a recursive formula to update the estimation in every sample of the unknown, but constant, vector $\boldsymbol{x}$ with the latest measurements, based only on the estimation from the last sample. App. Table A.1 summarizes the whole algorithm. The operand $\mathrm{E}(\cdot)$ calculates the expectation value of a random variable [Sim06].

App. Table A.1: The recursive weighted least squares algorithm

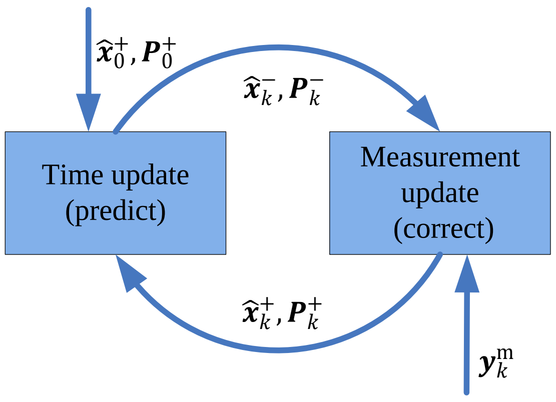
```
Initialization:
        $\hat{\boldsymbol{x}}_0 = \mathrm{E}(\boldsymbol{x})$
        $\boldsymbol{P}_0 = \mathrm{E}((\boldsymbol{x} - \hat{\boldsymbol{x}}_0)(\boldsymbol{x} - \hat{\boldsymbol{x}}_0)^T)$
For $k = 1,2,\ldots\ (k \in \mathbb{N}^+)$:
    $\boldsymbol{K}_k = \boldsymbol{P}_{k-1}\boldsymbol{H}_k^T \cdot \left(\boldsymbol{H}_k\boldsymbol{P}_{k-1}\boldsymbol{H}_k^T + \boldsymbol{R}_k\right)^{-1}$
    $\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_{k-1} + \boldsymbol{K}_k \cdot (\boldsymbol{y}_k^{\mathrm{m}} - \boldsymbol{H}_k\hat{\boldsymbol{x}}_{k-1})$
    $\boldsymbol{P}_k = (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_k)\boldsymbol{P}_{k-1}(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_K)^T + \boldsymbol{K}_k\boldsymbol{R}_k\boldsymbol{K}_k^T$
```

A.3.2 Recursive Linear Kalman Filter

For many real-time control problems, it is more interesting to estimate the system states rather than the plant's parameters. Therefore, the linear Kalman filter was developed in the 1960's [Sim06]. It enables the efficient estimation of system states of a linear discrete-time model in a recursive way. The fundamental assumption of this approach is that the system $\mathrm{E}\left(\boldsymbol{w}_k\boldsymbol{w}_k^T\right) = \boldsymbol{Q}_k\delta_{k-j}$ and the output $\mathrm{E}(\boldsymbol{v}_k\boldsymbol{v}_k^T) = \boldsymbol{R}_k\delta_{k-j}$ equations are disturbed by Gaussian white noise. Both of these noise processes are regarded to be uncorrelated with zero mean $\mathrm{E}\left(\boldsymbol{w}_k\boldsymbol{v}_k^T\right) = 0$. This results in the following equations:

$$\begin{aligned}\boldsymbol{x}_k &= \boldsymbol{F}_{k-1}\,\boldsymbol{x}_{k-1} + \boldsymbol{G}_{k-1}\boldsymbol{u}_{k-1} + \boldsymbol{w}_{k-1} \\ \boldsymbol{y}_k &= \boldsymbol{H}_k\boldsymbol{x}_k + \boldsymbol{v}_k\end{aligned} \tag{A.16}$$

Here $\boldsymbol{F}_{k-1}$ is the state transition matrix, $\boldsymbol{G}_{k-1}$ denotes the input coupling and $\boldsymbol{w}_{k-1}$ the process Gaussian white noise; $\boldsymbol{H}_k$ is the system output matrix and $\boldsymbol{v}_k$ the corresponding noise. At this point, the principle of every Kalman filter derivation (compare [Sim06]) is introduced in App. Figure A.2. Subsequent to the filter initialization, the first step in every sample step is the a priori estimation of the mean (system states) and the covariance (a scale for the confidence in them). This is called the prediction step and all of the equations that are related with it contain a "–" in the superscript.


$\hat{\boldsymbol{x}}_0^+, \boldsymbol{P}_0^+$
$\hat{\boldsymbol{x}}_k^-, \boldsymbol{P}_k^-$
Time update (predict)
Measurement update (correct)
$\hat{\boldsymbol{x}}_k^+, \boldsymbol{P}_k^+$
$\boldsymbol{y}_k^{\mathrm{m}}$

App. Figure A.2: Principle of Kalman filter based estimation, $\boldsymbol{y}_k^{\mathrm{m}}$ denotes the vector of measured outputs

This step forms the basis for the calculation of the optimal Kalman gain that is used to correct the estimated state vector with the information from the actual measurements $\boldsymbol{y}_k^{\mathrm{m}}$. Finally, the covariance matrix is updated. This is called the correction step and all of the equations that are related with it contain a " $+$ " in the superscript. In the next sample, these values are used to restart at the subsequent prediction step. The algorithm can be formulated as follows:

App. Table A.2: The linear discrete Kalman filter algorithm

Initialization:

$$\hat{\boldsymbol{x}}_0^+ = \mathrm{E}(\boldsymbol{x}_0)$$
$$\boldsymbol{P}_0^+ = \mathrm{E}\big((\boldsymbol{x} - \hat{\boldsymbol{x}}_0^+)(\boldsymbol{x} - \hat{\boldsymbol{x}}_0^+)^T\big)$$

for $k = 1,2, \dots$ $(k \in \mathbb{N}^+)$:

$$\hat{\boldsymbol{x}}_k^- = \boldsymbol{F}_{k-1}\hat{\boldsymbol{x}}_{k-1}^+ + \boldsymbol{G}_{k-1}\boldsymbol{u}_{k-1}$$
$$\boldsymbol{P}_k^- = \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}^+\boldsymbol{F}_{k-1}^T + \boldsymbol{Q}$$
$$\boldsymbol{K}_k = \boldsymbol{P}_k^-\boldsymbol{H}_k^T \cdot \big(\boldsymbol{H}_k\boldsymbol{P}_k^-\boldsymbol{H}_k^T + \boldsymbol{R}\big)^{-1}$$
$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k \cdot (\boldsymbol{y}_k^{\mathrm{m}} - \boldsymbol{H}_k\boldsymbol{x}_k)$$
$$\boldsymbol{P}_k^+ = (\boldsymbol{I} - \boldsymbol{K}_K \cdot \boldsymbol{H}_k) \cdot \boldsymbol{P}_k^-$$

To determine the connection between the Kalman filter and the recursive weighted least squares parameter estimation, one should have a closer look at App. Table A.2. The matrix $\boldsymbol{Q}(\boldsymbol{w}\boldsymbol{w}^T) = \mathrm{diag}(\sigma_{w1}^2, \dots, \sigma_{wn}^2)$ represents the covariance of the system states ($\boldsymbol{w}$ denotes the variance of the system states). Its entries represent the confidence in the a priori estimation and can be tuned by the application engineer. Large values represent high uncertainty (probably due to an imprecise model), whereas small values indicate a good trust. The second tuning matrix $\boldsymbol{R}$ represents the confidence in the actual measurements. Its effect resembles the first estimation problem (eq. (A.1) to eq. (A.5)). Moreover, it can be shown that if $\boldsymbol{x}_k$ is a constant vector then it follows that $\boldsymbol{F}_k = \boldsymbol{I}$, $\boldsymbol{Q}_k = 0$ and $\boldsymbol{u}_k = 0$.

In this case, the linear discrete Kalman filter algorithm (cf. App. Table A.2) reduces to the recursive weighted least squares algorithm (cf. App. Table A.1). This property is often exploited in the formulation of parameter estimation problems using Kalman filter algorithms (for more details see e.g. [Hay01] or Chapter 6.4.1).

A.3.3 Nonlinear Extended Kalman Filter

This section is nearly identical to Chapter 4.2.1. It is assumed that the plant model to be used in state estimation is naturally described as a nonlinear continuous-time state-space system:

$$\begin{gathered}\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}),\\ \boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}),\\ t \in \mathbb{R}, \boldsymbol{u}(t) \in \mathbb{R}^{n_u}, \boldsymbol{x}(t) \in \mathbb{R}^{n_x}, \boldsymbol{y}(t) \in \mathbb{R}^{n_y}\end{gathered} \tag{A.17}$$

where t is time, $\boldsymbol{u}(t)$ is the vector of inputs, $\boldsymbol{x}(t)$ is the vector of states and $\boldsymbol{y}(t)$ is the vector of outputs. The model eq. (A.17) cannot be utilized directly in a sampled data system. Instead, a discrete-time representation is necessary in a discrete-time state estimator. Therefore, the discrete-time version of eq. (A.17) with additive Gaussian noise is used in the sequel:

$$\begin{aligned}\boldsymbol{x}_k &= \boldsymbol{f}_{k|k-1}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) + \boldsymbol{w}_{k-1},\\ \boldsymbol{y}_k &= \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{v}_k,\\ &\boldsymbol{w}_k \sim N(0, \boldsymbol{Q}_k),\\ &\boldsymbol{v}_k \sim N(0, \boldsymbol{R}_k).\end{aligned} \tag{A.18}$$

Here t_k is the k-th sample time instant of a periodically sampled data system, $\boldsymbol{u}_k = \boldsymbol{u}(t_k)$, $\boldsymbol{x}_k = \boldsymbol{x}(t_k)$, $\boldsymbol{y}_k = \boldsymbol{y}(t_k)$, $\boldsymbol{w}_k$, $\boldsymbol{v}_k$ are Gaussian noise, and

$$\boldsymbol{f}_{k|k-1} = \boldsymbol{x}_{k-1} + \int_{t_{k-1}}^{t_k} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}_{k-1})\, \mathrm{d}t. \tag{A.19}$$

is the time integration of the prediction model between two time instances. The extended Kalman filter (EKF) algorithm is very similar to the linear Kalman filter presented before. To handle the nonlinearity, the system is linearized around the last estimation point using a Taylor series truncated after the first term. This can be performed for example numerically by use of a forward difference quotient (cf. eq. (A.20)).

App. Table A.3: The extended Kalman filter algorithm

Initialization:

$$\hat{\boldsymbol{x}}_0^+ = \mathrm{E}(\boldsymbol{x}_0)$$
$$\boldsymbol{P}_0^+ = \mathrm{E}\big((\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)^T\big)$$

for $k = 1,2,\dots$ $(k \in \mathbb{N}^+)$:

Predict:
$$\hat{\boldsymbol{x}}_k^- = \boldsymbol{f}_{k|k-1}(\hat{\boldsymbol{x}}_{k-1}^+, \boldsymbol{u}_{k-1})$$
$$\boldsymbol{P}_k^- = \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1}^+\boldsymbol{F}_{k-1}^T + \boldsymbol{Q}$$
$$\text{with } \boldsymbol{F}_{k-1} = e^{\left(\left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}_{k-1}^+} \cdot T_s\right)}$$

Correct:
$$\boldsymbol{K}_k = \boldsymbol{P}_k^-\boldsymbol{H}_k^T \cdot \left(\boldsymbol{H}_k\boldsymbol{P}_k^-\boldsymbol{H}_k^T + \boldsymbol{R}\right)^{-1}$$
$$\text{with } \boldsymbol{H}_k = \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}_k^-}$$
$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k \cdot (\boldsymbol{y}_k^{\mathrm{m}} - \boldsymbol{h}(\hat{\boldsymbol{x}}_k^-))$$
$$\boldsymbol{P}_k^+ = (\boldsymbol{I} - \boldsymbol{K}_K \cdot \boldsymbol{H}_k) \cdot \boldsymbol{P}_k^-$$

The calculation of $\hat{\boldsymbol{x}}_k^-$ is performed by integrating model eq. (A.17) from t_{k-1} to t_k by means of eq. (A.19). $\boldsymbol{F}_{k-1}$ is the state-transitions matrix of $\boldsymbol{f}$ with respect to $\boldsymbol{x}$ at $\hat{\boldsymbol{x}}_{k-1}^+$ and $\boldsymbol{H}_k$ is the partial derivative matrix of $\boldsymbol{h}$ with respect to $\boldsymbol{x}$ at $\hat{\boldsymbol{x}}_k^-$. The Jacobians $\boldsymbol{J}_{k-1}$ and $\boldsymbol{H}_k$ must either be provided directly, or they can be determined numerically, for example with a forward difference quotient (note that ϵ denotes the machine precision of the particular computer architecture and n_x the number of system states):

for $i = 1,2,\dots,n_x$; $\Delta \cong \sqrt{\epsilon}$

$$(\boldsymbol{J}_{k-1})_i = \frac{\boldsymbol{f}(\hat{\boldsymbol{x}}_{k-1} + \Delta\, \boldsymbol{e}_i, \boldsymbol{u}_{k-1}) - \boldsymbol{f}(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_{k-1})}{\Delta} \tag{A.20}$$

The sketched EKF algorithm in App. Table A.3 has several practical difficulties when it should be implemented on microcontrollers with a limited word length of the data types (i.e. a double float). So the numerical approximation of the Jacobian matrix can cause inaccuracies due to cancelling effects and therefore a wrong approximation of the nonlinearity in the surrounding of the working point in the actual step might occur. Another problem is the calculation of the propagated covariance matrix via the difference $\mathbf{P}_\mathrm{k}^+ = \mathbf{P}_\mathrm{k}^- - \mathbf{P}_\mathrm{k}^-\mathbf{K}_\mathrm{k}\mathbf{H}_\mathrm{k}$ (compare App. Table A.3). The so called divergence phenomenon [Hay01] may cause the loss of positive definiteness of the propagated matrix $\boldsymbol{P}_k^+$ (a necessary condition for a covariance matrix). To cope with this problem there are different approaches that enhance the numeric accuracy of the standard EKF algorithm. The most common one is the square-root (SR) filtering technique by means of a Cholesky or U-D decomposition [Gol13].

In SR filtering algorithms the square-root of the covariance matrix

$$\boldsymbol{P}_k = \boldsymbol{S}_k^T \cdot \boldsymbol{S}_k \tag{A.21}$$

is propagated in the recursive formulation. Defining the condition number of a matrix as

$$\kappa(\boldsymbol{P}_k) = \frac{\sigma_{\max}(\boldsymbol{P}_k)}{\sigma_{\min}(\boldsymbol{P}_k)} \geq 1 \tag{A.22}$$

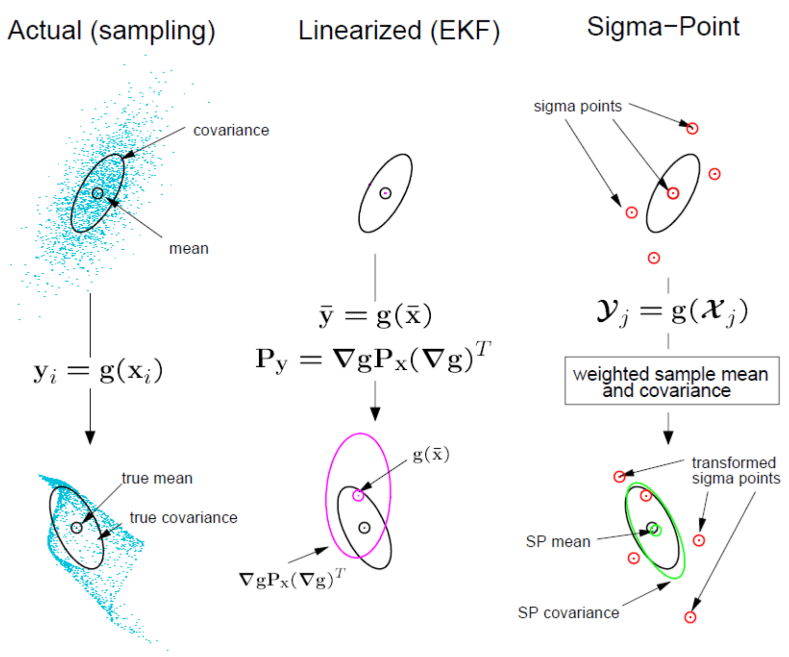
wherein $\sigma_{\max/\min}$ are the largest and smallest singular value, it can be shown that the condition number of the square-root $\mathbf{S}_\mathrm{k}$ is the square-root of the condition number of matrix $\mathbf{P}_\mathrm{k}$ (see [Gre15] for more details):

$$\begin{gathered}\sigma^2(\boldsymbol{P}_k) = [\sigma^2(\boldsymbol{S}_k)]^2 \\ \frac{\sigma_{\max}(\boldsymbol{P}_k)}{\sigma_{\min}(\boldsymbol{P}_k)} = \frac{\sigma_{\max}^2(\boldsymbol{S}_k)}{\sigma_{\min}^2(\boldsymbol{S}_k)} \\ \kappa(\boldsymbol{P}_k) = \kappa^2(\boldsymbol{S}_k)\end{gathered} \tag{A.23}$$

At this point no further detail on the implementation is given. The most common publications regarding the numerically efficient and reliable implementation of the different approaches are given in [Gre15]. Especially the SR implementation is important for complex state estimation algorithms as proposed in Chapter A.4.2.

A.4 Recursive Nonlinear Derivative-Free Estimation Methods

In this chapter estimation algorithms are examined that do not need the calculation of the Jacobians of the nonlinear prediction model. Besides the unscented transformation (UT) based algorithm, the central difference filter (e.g. [Nor04]) and the Gaussian quadrature Kalman filter [Ara07] exist. It has been shown that both methods are strongly connected with the UT [Hay01] and are therefore not considered here separately. The so called sigma point transformation (SPT) is based on the idea that it is easier to approximate a Gaussian distribution than it is to approximate an arbitrary nonlinear function or transformation (see [Jul04], [Sim06]). In other words it is difficult to find a nonlinear transformation of a probability density function (PDF), but it is easy to perform a nonlinear transformation of a vector. Therefore, this algorithm selects a set of points around the actual operation point in a heuristic manner and then transforms them through the underlying nonlinear function. With the help of weighting factors and a transformed set of points in the state-space a sampled PDF can be approximated [Sim10]. In this way, a more appropriate solution, in comparison to the EKF, of the estimate mean and the covariance can be calculated. In general, this approximation (for Gaussian assumption) is accurate to the third term of the Taylor series expansion. For a non-Gaussian assumption it is still valid to the second order [Mer01]. For a proof, see for example [Hay01] – appendix A. One can easily see the advantage of UT in App. Figure A.3 where a comparison between a Monte Carlo simulation, the unscented approach, and the extended Kalman filter is shown. It is quite clear that the truncation of the Taylor series expansion after the first term (EKF) leads to the most inaccurate results.



App. Figure A.3: Advantages of the unscented transformation [Mer04]

In the following more details are given regarding the principles of the UT and its implicit context with the weighted statistical linear regression (WSLR) method [Mer04b]. The aim of the WSLR is to find a linear regression

$$\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x}) \approx \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} \tag{A.24}$$

for the nonlinear function $\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x})$ through the evaluation of it in N points $(\boldsymbol{\chi}_i, i = 1..N)$. The set of $\boldsymbol{\chi}_i$ is chosen such that it can represent the statistical properties of $\boldsymbol{x}$ like mean $\bar{\boldsymbol{x}}$ and covariance $\boldsymbol{P}_{\boldsymbol{x}}$:

$$\begin{gathered} \bar{\boldsymbol{x}} = \Sigma_{i=1}^{N} \boldsymbol{w}_i \boldsymbol{\chi}_i \\ \boldsymbol{P}_{\boldsymbol{x}} = \Sigma_{i=1}^{N} \boldsymbol{w}_i (\boldsymbol{\chi}_i - \bar{\boldsymbol{x}})(\boldsymbol{\chi}_i - \bar{\boldsymbol{x}})^T \\ \Sigma_{i=1}^{N} \boldsymbol{w}_i = 1 \text{ (regression weights)} \end{gathered} \tag{A.25}$$

The optimization objective is to find a solution for eq. (A.24),

$$\boldsymbol{A}, \boldsymbol{b} = \min \left(\underbrace{\mathrm{E}(\boldsymbol{\epsilon}^T \boldsymbol{W} \boldsymbol{\epsilon})}_{J} \right) \approx \min \left(\Sigma_{i=1}^{N} \boldsymbol{w}_i \boldsymbol{\epsilon}_i^T \boldsymbol{\epsilon}_i \right) \tag{A.26}$$

$$\boldsymbol{\epsilon}_i = \boldsymbol{g}(\boldsymbol{\chi}_i) - (\boldsymbol{A}\boldsymbol{\chi}_i + \boldsymbol{b}) \tag{A.27}$$

where $\boldsymbol{\epsilon}_i$ denotes the point-wise linearization error and it is assumed that the selected regression points $\boldsymbol{\chi}_i$ gather the prior mean and covariance of $\boldsymbol{x}$. According to [Mer04b], $\boldsymbol{A}, \boldsymbol{b}$ can be derived through statistical linearization. For calculating $\boldsymbol{\epsilon}$, the optimization objective J in eq. (A.26) is substituted with the expression in eq. (A.27):

$$J = \mathrm{E}\Big((\boldsymbol{g}(\boldsymbol{\chi}_i) - (\boldsymbol{A}\boldsymbol{\chi}_i + \boldsymbol{b}))^T \boldsymbol{W} (\boldsymbol{g}(\boldsymbol{\chi}_i) - (\boldsymbol{A}\boldsymbol{\chi}_i + \boldsymbol{b})) \Big) \tag{A.28}$$

To derive the unknown vector $\boldsymbol{b}$ the partial derivative of J with respect to $\boldsymbol{b}$ needs to be calculated and set equal to zero:

$$\begin{aligned} \frac{\partial J}{\partial \boldsymbol{b}} &= -2 \cdot \mathrm{E}\big(\boldsymbol{W}(\boldsymbol{g}(\boldsymbol{\chi}_i) - \boldsymbol{A}\boldsymbol{\chi}_i - \boldsymbol{b})\big) \overset{!}{=} 0 \\ \Rightarrow \boldsymbol{b} &= \mathrm{E}\big(\boldsymbol{g}(\boldsymbol{\chi}_i)\big) - \boldsymbol{A}\mathrm{E}(\boldsymbol{\chi}_i) = \\ &= \bar{\boldsymbol{y}} - \boldsymbol{A}\bar{\boldsymbol{x}} \end{aligned} \tag{A.29}$$

In a second step $\boldsymbol{A}$ can be derived by substituting $\boldsymbol{b}$ in eq. (A.29), afterwards the partial derivative is calculated with respect to $\boldsymbol{A}$ which is set equal to zero:

$$\begin{aligned} &J = \boldsymbol{\epsilon}^T \boldsymbol{W} \boldsymbol{\epsilon}, \;\; \boldsymbol{\epsilon}_i = \boldsymbol{g}(\boldsymbol{\chi}_i) - (\boldsymbol{A}\boldsymbol{\chi}_i + \bar{\boldsymbol{y}} - \boldsymbol{A}\bar{\boldsymbol{x}}) = \boldsymbol{g}(\boldsymbol{\chi}_i) - \boldsymbol{A}(\boldsymbol{\chi}_i - \bar{\boldsymbol{x}}) - \bar{\boldsymbol{y}} \\ &\frac{\partial J}{\partial \boldsymbol{A}} = -2 \cdot \mathrm{E}\big(\boldsymbol{W}(\boldsymbol{A}\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}^T + \tilde{\boldsymbol{y}}\tilde{\boldsymbol{x}}^T)\big) \overset{!}{=} 0, \;\; \tilde{\boldsymbol{x}} = \boldsymbol{\chi}_i - \bar{\boldsymbol{x}}\,, \tilde{\boldsymbol{y}} = \boldsymbol{g}(\boldsymbol{\chi}_i) - \bar{\boldsymbol{y}} = \boldsymbol{\gamma}_i - \bar{\boldsymbol{y}} \\ &\qquad\qquad \Rightarrow \boldsymbol{A} = \mathrm{E}(\tilde{\boldsymbol{y}}\tilde{\boldsymbol{x}}^T) \cdot \mathrm{E}(\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}^T)^{-1} \\ &\qquad\qquad\quad\;\; = \mathrm{E}(\tilde{\boldsymbol{x}}\tilde{\boldsymbol{y}}^T)^T\, \mathrm{E}(\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}^T)^{-1} \\ &\qquad\qquad\quad\;\; = \boldsymbol{P}_{xy}^T \boldsymbol{P}_x^T \end{aligned} \tag{A.30}$$

To calculate the unknown a posteriori Gaussian statistics – the covariance $\boldsymbol{P}_{xy}, \boldsymbol{P}_y$ and the mean value $\bar{\boldsymbol{y}}$ – the following approximations of the propagated regression point $\boldsymbol{\gamma}_i = \boldsymbol{g}(\boldsymbol{\chi}_i)$ are assumed (compare eq. (A.25)):

$$\begin{aligned} \bar{\boldsymbol{y}} &\approx \hat{\boldsymbol{y}} = \Sigma_{i=1}^N w_i \boldsymbol{\gamma}_i \\ \boldsymbol{P}_y &\approx \widehat{\boldsymbol{P}}_y = \Sigma_{i=1}^N w_i (\boldsymbol{\gamma}_i - \hat{\boldsymbol{y}})(\boldsymbol{\gamma}_i - \hat{\boldsymbol{y}})^T \\ \boldsymbol{P}_{xy} &\approx \widehat{\boldsymbol{P}}_{xy} = \Sigma_{i=1}^N w_i (\boldsymbol{\chi}_i - \bar{\boldsymbol{x}})(\boldsymbol{\gamma}_i - \hat{\boldsymbol{y}})^T \\ \boldsymbol{P}_\epsilon &\approx \Sigma_{i=1}^N w_i \boldsymbol{\epsilon}_i^T \boldsymbol{\epsilon}_i = \widehat{\boldsymbol{P}}_y - \boldsymbol{A}\boldsymbol{P}_x\boldsymbol{A}^T \end{aligned} \tag{A.31}$$

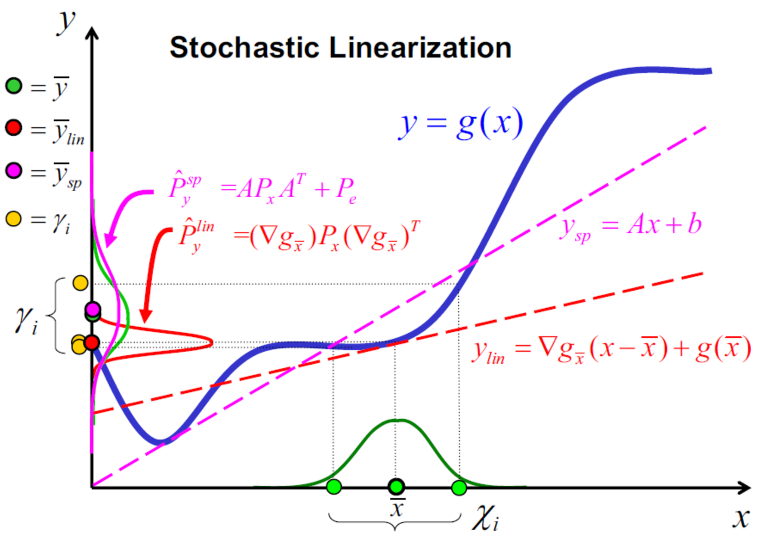
Where $\boldsymbol{\epsilon}_i$ is defined as the point-wise linearization error:

$$\boldsymbol{\epsilon}_i = \boldsymbol{g}(\boldsymbol{\chi}_i) - (\boldsymbol{A}\boldsymbol{\chi}_i + \boldsymbol{b}) \tag{A.32}$$

By the help of these assumptions the statistics of $\boldsymbol{y}$ can be expressed as follows:

$$\begin{aligned} \hat{\boldsymbol{y}}^{\mathrm{SLR}} &= \boldsymbol{A}\bar{\boldsymbol{x}} + \boldsymbol{b} \\ \widehat{\boldsymbol{P}}_y^{\mathrm{SLR}} &= \boldsymbol{A}\boldsymbol{P}_x\boldsymbol{A}^T + \boldsymbol{P}_\epsilon \end{aligned} \tag{A.33}$$

The following graphic repeats the above sketch for the interconnection of statistical properties by means of a simple example.


Stochastic Linearization
y
● $= \bar{y}$
● $= \bar{y}_{lin}$
● $= \bar{y}_{sp}$
● $= \gamma_i$
$y = g(x)$
$\hat{P}_y^{sp} = AP_xA^T + P_e$
$\hat{P}_y^{lin} = (\nabla g_{\bar{x}})P_x(\nabla g_{\bar{x}})^T$
$y_{sp} = Ax + b$
γ_i
$y_{lin} = \nabla g_{\bar{x}}(x - \bar{x}) + g(\bar{x})$
$\bar{x}$
χ_i
x

App. Figure A.4. WSLR vs. first order Taylor approximation [Mer04b]

In App. Figure A.4 a one dimensional Gaussian random variable (GRV) x with a certain distribution (green line) and a mean $\bar{x}$ is transformed through a nonlinear function $y = g(x)$ (blue line) in order to reproduce the a posteriori random variable y. The green line on the y-axis indicates the true mean and covariance of the distribution after the transformation. Considering the two different approximations of the transformation (lin = first order Taylor approximation (red), sp = sigma point transformation (magenta)) one can see instantaneously the difference in the accuracy of the approaches. The linearization $y_{\text{lin}} = \nabla g_{\bar{x}}(x - \bar{x}) + g(\bar{x})$ of the nonlinear function results in a strongly biased mean (red dotted) and a totally different distribution of P_y (red line). On the one hand, this is reasoned to the fact that the WSLR calculates an approximate expected Jacobian whereas the EKF simply calculates the Jacobian at the prior mean $\bar{x}$. On the other hand, the WSLR considers a non-zero bias b, whereas the first order Taylor series approximation of the EKF Jacobian drops this term [Mer04b].

A.4.1 Unscented Kalman Filter Algorithm

In order to achieve higher accuracy, the unscented Kalman filter (UKF) calculates the mean and the covariance from a set of disturbed state vectors, by use of the unscented transformation of the sigma points (SP) (for more details see Chapter A.4). Thereby, the Jacobians of $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{h}(\boldsymbol{x})$ are not needed for the approximation of the nonlinear prediction model. The structure of the equation set, containing prediction and update step, is similar to the one of the EKF (App. Table A.3). However, the calculation of the covariance matrix requires $2n + 1$ operations to integrate the nonlinear system from the last to the actual time instant ($\boldsymbol{X}_{(k|k-1)}$ calculation step in App. Table A.6) and is therefore computationally costly. The symmetry of all the involved matrices is fully exploited to reduce computational costs.

All square-root matrix operations are done by the use of a Cholesky decomposition [And99]. This decomposition is possible due to the fact that the covariance matrix is symmetric positive definite. For all calculus with the decomposed matrix only the lower triangular part of the solution is used. $\boldsymbol{X}_{(k|k-1)}$ denotes the propagation from time step $k-1$ to k through the continuous nonlinear model by the application of a discrete integration method (e.g. Runge-Kutta 4). In the following App. Table A.4 and App. Table A.5 a list of the tuning parameters with typical default values for the UKF is given. The algorithm itself with additive Gaussian white noise $\boldsymbol{Q}, \boldsymbol{R}$ is sketched in the pseudo code in App. Table A.6.

App. Table A.4: List of UKF parameters

Parameter	**Description**
n	Number of states
α	Spread of sigma points around the actual mean (e.g. $10^{-4} \leq \alpha \leq 1$)
κ	Scaling kurtosis of sigma points (Default = 0, or $3-n$)
β	Characteristic of the distribution of noise process (For Gaussian 2 is optimal)

App. Table A.5: List of pre-calculated UKF weightings

Weight	**Description**
$\lambda = \alpha^2 \cdot (n+\kappa) - n$	Scaling parameter
$a = \alpha^2 \cdot (n+\kappa)$	Denominator argument of weightings
$w_0^{\mathrm{m}} = \dfrac{\lambda}{a}$	Weighting of unmodified mean prediction
$w_0^{\mathrm{c}} = \dfrac{\lambda}{\alpha + 1 - \alpha^2 + \beta}$	Weighting of unmodified output mean prediction
$w_i^{\mathrm{m}} = w_i^{\mathrm{c}} = \dfrac{1}{2 \cdot a}\,; i = 1, \dots, 2n$	Weighting of sigma points of states and outputs
$\gamma = \sqrt{\alpha^2 \cdot (n+\kappa)}$	Weighting of square-root of covariance from $k-1$

App. Table A.6: The unscented Kalman filter algorithm

Initialization:

$$\hat{\boldsymbol{x}}_0^+ = \mathrm{E}(\boldsymbol{x}_0)$$

$$\boldsymbol{P}_0^+ = \mathrm{E}\big((\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)^T\big)$$

for $k = 1,2,\ldots$ $(k \in \mathbb{N}^+)$:

Predict:

$$\boldsymbol{X}_{k-1} = \left[\hat{\boldsymbol{x}}_{k-1}, \hat{\boldsymbol{x}}_{k-1} + \gamma \cdot \sqrt{\boldsymbol{P}_{k-1}^+}, \hat{\boldsymbol{x}}_{k-1} - \gamma \cdot \sqrt{\boldsymbol{P}_{k-1}^+}\right]$$

$$\boldsymbol{X}_{k|k-1} = \boldsymbol{f}_{k|k-1}(\boldsymbol{X}_{k-1}, \boldsymbol{u}_{k-1})$$

$$\hat{\boldsymbol{x}}_k^- = \sum_{i=0}^{2\cdot n} w_i^{\mathrm{m}} \cdot \boldsymbol{X}_{i,k|k-1}$$

$$\boldsymbol{P}_k^- = \sum_{i=0}^{2\cdot n} w_i^{\mathrm{c}} \cdot \left[\boldsymbol{X}_{i,k|k-1} - \hat{\boldsymbol{x}}_k^-\right]\left[\boldsymbol{X}_{i,k|k-1} - \hat{\boldsymbol{x}}_k^-\right]^T + \boldsymbol{Q}$$

$$\boldsymbol{X'}_k = \left[\hat{\boldsymbol{x}}_k^-, \hat{\boldsymbol{x}}_k^- + \gamma \cdot \sqrt{\boldsymbol{P}_k^-}, \hat{\boldsymbol{x}}_k^- - \gamma \cdot \sqrt{\boldsymbol{P}_k^-}\right]$$

$$\boldsymbol{Y}_k = \boldsymbol{h}(\boldsymbol{X'}_k)$$

$$\hat{\boldsymbol{y}}_k^- = \sum_{i=0}^{2\cdot n} w_i^{\mathrm{m}} \cdot \boldsymbol{Y}_{i,k}$$

Correct:

$$\boldsymbol{P}_{y_k} = \sum_{i=0}^{2\cdot n} w_i^{\mathrm{c}} \cdot \left[\boldsymbol{Y}_{i,k} - \hat{\boldsymbol{y}}_k^-\right]\left[\boldsymbol{Y}_{i,k} - \hat{\boldsymbol{y}}_k^-\right]^T + \boldsymbol{R}$$

$$\boldsymbol{P}_{x_k y_k} = \sum_{i=0}^{2\cdot n} w_i^{\mathrm{c}} \cdot \left[\boldsymbol{X}_{i,k|k-1} - \hat{\boldsymbol{x}}_k^-\right]\left[\boldsymbol{Y}_{i,k} - \hat{\boldsymbol{y}}_k^-\right]^T$$

$$\boldsymbol{K}_k = \boldsymbol{P}_{x_k y_k} \cdot \boldsymbol{P}_{y_k}^{-1}$$

$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k \cdot (\boldsymbol{y}_k^{\mathrm{m}} - \hat{\boldsymbol{y}}_k^-)$$

$$\boldsymbol{P}_k^+ = \boldsymbol{P}_k^- - \boldsymbol{K}_k \cdot \boldsymbol{P}_{y_k} \cdot \boldsymbol{K}_k^T$$

At this point the UKF algorithm (see App. Table A.6) is compared with the WSLR method, to show up its implicit context to the WSLR as mentioned in the last section. The calculation of the covariances $\boldsymbol{P}_{\boldsymbol{k}}^-, \boldsymbol{P}_{y_k}, \boldsymbol{P}_{x_k y_k}$ is performed in a similar way, besides adding additive uncertainties $\boldsymbol{Q}, \boldsymbol{R}$, like in eq. (A.25) and eq. (A.31). Another interesting point is the calculation of the Kalman gain $\boldsymbol{K}_k = \boldsymbol{P}_{x_k y_k} \cdot \boldsymbol{P}_{y_k}^{-1}$. It optimally propagates the new information of the measurement reversely in the state-space. Comparing it with the WSLR gain $\boldsymbol{A}$ (eq. (A.30)), this can be interpreted as propagating the innovation information downwards (from y-space to the x-space) using a statistically linearized inverse observation function [Mer04b].

A.4.2 Square-Root Unscented Kalman Filter Algorithm

The equations of the square-root unscented Kalman filter (SR-UKF) are identical to the UKF, with the distinction, that the postitive defintites of the covariance matrices are guaranteed to be compututionaly efficient. In the SR-UKF implementation, the Cholesky factors $\boldsymbol{CP}_k^-, \boldsymbol{CP}_k^+$ are propagated directly and the refactorization of the covariance matrices is avoided through rank-one updates of the Cholesky matrix [Mer01]. In this way, it is ensured that the Cholesky factor is always regular, which retains numerical robustnes of the algorithm. Moreover, the gain matrix $\boldsymbol{K}_k$ is determined as the solution of the linear equation system

$$\boldsymbol{K}_k \cdot \boldsymbol{P}_{\boldsymbol{y}_k} = \boldsymbol{P}_{\boldsymbol{x}_k \boldsymbol{y}_k} \tag{A.34}$$

that can be more efficiently solved by utilizing the Cholesky factorization:

$$\boldsymbol{K}_k = \boldsymbol{P}_{\boldsymbol{x}_k \boldsymbol{y}_k} \cdot \left(\boldsymbol{CP}'_{\boldsymbol{y}_k} \boldsymbol{CP}_{\boldsymbol{y}_k}\right)^{-1} \tag{A.35}$$

In App. Table A.7 the algorithm of the SR-UKF is given and the most important numerical operators are explained in the following section. LQ denotes the transpose of the QR-decomposition in order to be consistent with the other lower triangular operations (e.g. Cholesky decomposition). The matrix $\boldsymbol{Q}$ is not computed directly to minimize the number of necessary operations and only $\boldsymbol{R}$ is calculated ($\boldsymbol{A} = \boldsymbol{QR} \in \mathbb{R}^{\mathrm{m} \times \mathrm{n}}$). The operation $\mathrm{cholupd}(\boldsymbol{CP}, v, \pm 1)$ denotes a rank one update of a Cholesky factorized matrix $\mathbf{A} = \mathbf{L}^{\mathrm{T}} \cdot \mathbf{L}$, see [Seg07], [Don79]. It computes the lower triangular matrix $\boldsymbol{L}^* = \boldsymbol{L} \pm \boldsymbol{v}\boldsymbol{v}^T$ that ensures the positive definiteness of matrix $\boldsymbol{A}^* = \boldsymbol{L}^{T*} \cdot \boldsymbol{L}^*$ and makes the SR-UKF algorithm more robust against numerical instability than the original UKF algorithm (see App. Table A.6). The parameters and pre-calculated weights are the same as in the case of the UKF algorithm (see App. Table A.4 & App. Table A.5).

App. Table A.7: The square-root unscented Kalman filter algorithm

Initialization:

$$\hat{\boldsymbol{x}}_0^+ = \mathrm{E}(\boldsymbol{x}_0)$$

$$\boldsymbol{CP}_0^+ = \mathrm{chol}\left(\mathrm{E}\big((\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)^T\big)\right)$$

for $k = 1,2,\ldots\ (k \in \mathbb{N}^+)$:

Predict:

$$\boldsymbol{X}_{k-1} = [\hat{\boldsymbol{x}}_{k-1}, \hat{\boldsymbol{x}}_{k-1} + \gamma \cdot \boldsymbol{CP}_{k-1}^+, \hat{\boldsymbol{x}}_{k-1} - \gamma \cdot \boldsymbol{CP}_{k-1}^+]$$

$$\boldsymbol{X}_{k|k-1} = \boldsymbol{f}_{k|k-1}(\boldsymbol{X}_{k-1}, \boldsymbol{u}_{k-1})$$

$$\hat{\boldsymbol{x}}_k^- = \sum_{i=0}^{2\cdot n} w_i^{\mathrm{m}} \cdot \boldsymbol{X}_{i,k|k-1}$$

$$\boldsymbol{CP}_k^- = \mathrm{LQ}\left(\left[\sqrt{w_1^{\mathrm{c}}} \cdot (\boldsymbol{X}_{1:2\cdot n,k|k-1} - \hat{\boldsymbol{x}}_k^-), \sqrt{\boldsymbol{Q}}\,\right]\right)$$

$$\boldsymbol{CP}_k^- = \mathrm{cholupd}\big(\boldsymbol{CP}_k^-, \boldsymbol{X}_{k|k-1} - \hat{\boldsymbol{x}}_k^-, \boldsymbol{w}_0^{\mathrm{c}}\big)$$

$$\boldsymbol{X'}_k = [\hat{\boldsymbol{x}}_k^-, \hat{\boldsymbol{x}}_k^- + \gamma \cdot \boldsymbol{CP}_k^-, \hat{\boldsymbol{x}}_k^- - \gamma \cdot \boldsymbol{CP}_k^-]$$

$$\boldsymbol{Y}_k = \boldsymbol{h}(\boldsymbol{X'}_k)$$

$$\hat{\boldsymbol{y}}_k^- = \sum_{i=0}^{2\cdot n} w_i^{\mathrm{m}} \cdot \boldsymbol{Y}_{i,k}$$

Correct:

$$\boldsymbol{CP}_{y_k} = \mathrm{LQ}\left(\left[\sqrt{w_1^{\mathrm{c}}} \cdot (\boldsymbol{Y}_{1:2\cdot n,k} - \hat{\boldsymbol{y}}_k^-), \sqrt{\boldsymbol{R}}\,\right]\right)$$

$$\boldsymbol{CP}_{y_k} = \mathrm{cholupd}\big(\boldsymbol{S}_{y_k}, \boldsymbol{Y}_{0,k} - \hat{\boldsymbol{y}}_k^-, \boldsymbol{w}_0^{\mathrm{c}}\big)$$

$$\boldsymbol{P}_{x_k y_k} = \sum_{i=0}^{2\cdot n} w_i^{\mathrm{c}} \cdot \left[\boldsymbol{X}_{i,k|k-1} - \hat{\boldsymbol{x}}_k^-\right]\left[\boldsymbol{Y}_{i,k} - \hat{\boldsymbol{y}}_k^-\right]^T$$

$$\boldsymbol{K}_k = \boldsymbol{P}_{x_k y_k} \cdot \left(\boldsymbol{CP}'_{y_k} \boldsymbol{CP}_{y_k}\right)^{-1}$$

$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k \cdot (\boldsymbol{y}_k^{\mathrm{m}} - \hat{\boldsymbol{y}}_k^-)$$

$$\boldsymbol{U} = \boldsymbol{K}_k \cdot \boldsymbol{CP}_{y_k}$$

$$\boldsymbol{CP}_k^+ = \mathrm{cholupd}(\boldsymbol{CP}_k^-, \boldsymbol{U}, -1)$$

A.5 Linear Moving Horizon Estimation

This special estimation algorithm makes use of several past measurements in a static moving window to improve the state estimate at the current time instance t_k. Referring to [Sim10], the idea of the moving horizon estimator (MHE) is the reformulation of the fundamental optimization objective of the Kalman filter. For the general nonlinear case the minimization problem can be defined as:

$$\min_{\xi_k} \parallel \boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0 \parallel^2_{\boldsymbol{I}_0^+} + \sum_{i=1}^{N} \parallel \boldsymbol{y}_i^{\mathrm{m}} - \boldsymbol{h}(\boldsymbol{x}_i) \parallel^2_{\boldsymbol{R}^{-1}} + \sum_{k=0}^{N-1} \parallel \boldsymbol{x}_{i+1} - \boldsymbol{f}(\boldsymbol{x}_i) \parallel^2_{\boldsymbol{Q}^{-1}}$$
$$\boldsymbol{\xi}_k = (\boldsymbol{x}_0^T, \boldsymbol{x}_1^T, \dots, \boldsymbol{x}_N^T)^T \tag{A.36}$$

$$\text{s.t. } \mathbf{g}(\boldsymbol{\xi}_k) = 0 \tag{A.37}$$

$$\mathbf{c}(\boldsymbol{\xi}_k) \leq 0 \tag{A.38}$$

The two additional equations (A.37) and (A.38) incorporate constraints to the states at the time instances $k = 1, \dots, N$. Since the number of measurements would increase with every time step k, also the dimension of the optimization problem would grow tremendously and by this the necessary calculation time. In order to reduce it, a moving window (with M steps) is defined that gives a good tradeoff between accuracy and efficiency:

$$\min_{\xi_k} \parallel \boldsymbol{x}_{k-M} - \hat{\boldsymbol{x}}_{k-M}^+ \parallel^2_{\boldsymbol{I}_{k-M}^+} + \sum_{i=k-M}^{k} \parallel \boldsymbol{y}_i^{\mathrm{m}} - \boldsymbol{h}(\boldsymbol{x}_i) \parallel^2_{\boldsymbol{R}^{-1}}$$
$$+ \sum_{i=k-M+1}^{k} \parallel \boldsymbol{x}_i - \boldsymbol{f}(\boldsymbol{x}_{i-1}) \parallel^2_{\boldsymbol{Q}^{-1}} \tag{A.39}$$

The dimension of the optimization problem is thereby reduced to $(N - M + 1)$. For the initialization of the MHE filter the information matrix $\boldsymbol{I}_{k-M}^+$, which is the inverse of the a posteriori covariance matrix $\boldsymbol{P}_{k-M}^+$, needs to be calculated e.g. by the use of the standard EKF algorithm (compare App. Table A.3), wherein $\boldsymbol{I}_{k-M}^+ = (\boldsymbol{P}_{k-M}^+)^{-1}$.

In the following, the solution of the optimization problem eq. (A.39) with respect to eq. (A.37) and eq. (A.38) for a linear prediction model is sketched. The aim is to have a representation that can be solved by a QP solver (such as QL [Sch05]). For this, it is necessary to bring eq. (A.40) to the QP standard form:

$$\boldsymbol{\xi}_k^{\,*} = \underset{\xi_k}{\operatorname{argmin}} \left(\frac{1}{2} \boldsymbol{\xi}_k^{\,T} \boldsymbol{G} \boldsymbol{\xi}_k + \boldsymbol{c}^T \boldsymbol{\xi}_k \right)$$
$$\text{s.t. } \boldsymbol{\xi}_{k_{\min}} \leq \boldsymbol{\xi}_k \leq \boldsymbol{\xi}_{k_{\max}} \tag{A.40}$$
$$\boldsymbol{A} \cdot \boldsymbol{\xi}_k = \boldsymbol{b}\,;\; \boldsymbol{L} \cdot \boldsymbol{\xi}_k \geq \boldsymbol{k}$$

Starting with the first term of the minimization problem in eq. (A.39) the following expression can be derived:

$$\begin{aligned}\parallel \boldsymbol{x}_{k-M} - \widehat{\boldsymbol{x}}^{+}_{k-M} \parallel^{2}_{\boldsymbol{I}^{+}_{k-M}} &= \parallel \boldsymbol{I}^{+}_{k-M} \cdot (\boldsymbol{x}_{k-M} - \widehat{\boldsymbol{x}}_{k-M}) \parallel_{2} \\ &= (\boldsymbol{x}_{k-M} - \widehat{\boldsymbol{x}}^{+}_{k-M})^{T} \boldsymbol{I}^{T}_{k-M} \boldsymbol{I}_{k-M} (\boldsymbol{x}_{k-M} - \boldsymbol{x}^{+}_{k-M}) \\ &= \boldsymbol{x}^{T}_{k} \cdot \boldsymbol{I}_{k-M} \cdot \boldsymbol{x}_{k-M} - 2\boldsymbol{x}^{+T}_{k-M} \boldsymbol{I}_{k-M} \cdot \boldsymbol{x}_{k-M}\end{aligned} \tag{A.41}$$

The second term is expanded as follows:

$$\begin{aligned}\parallel \boldsymbol{y}^{\mathrm{m}}_{i} - \boldsymbol{h}(\boldsymbol{x}_{i}) \parallel^{2}_{\boldsymbol{R}^{-1}} &= \parallel \boldsymbol{R}^{-1} \cdot \left(\boldsymbol{y}^{\mathrm{m}}_{i} - \boldsymbol{h}(\boldsymbol{x}_{i})\right) \parallel_{2} \\ &= \left(\boldsymbol{y}^{\mathrm{m}}_{i} - \boldsymbol{h}(\boldsymbol{x}_{i})\right)^{T} \boldsymbol{R}^{-T} \boldsymbol{R}^{-1} \left(\boldsymbol{y}^{\mathrm{m}}_{i} - \boldsymbol{h}(\boldsymbol{x}_{i})\right) \\ &= \boldsymbol{h}^{T}(\boldsymbol{x}_{i}) \cdot \boldsymbol{R}^{-T} \boldsymbol{R}^{-1} \cdot \boldsymbol{h}(\boldsymbol{x}_{i}) - 2\boldsymbol{h}^{T}(\boldsymbol{x}_{i}) \boldsymbol{R}^{-T} \boldsymbol{R}^{-1} \cdot \boldsymbol{y}_{i} \\ \boldsymbol{H}_{i} &= \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right|_{x_{i}}, \quad \boldsymbol{h}(\boldsymbol{x}_{i}) = \boldsymbol{H}_{i} \cdot \boldsymbol{x}_{i} \\ \boldsymbol{R}_{\mathrm{inv}} &= \boldsymbol{R}^{-T} \boldsymbol{R}^{-1} \\ &= \boldsymbol{x}^{T}_{i} \cdot \boldsymbol{H}^{T}_{i} \boldsymbol{R}_{\mathrm{inv}} \boldsymbol{H}_{i} \cdot \boldsymbol{x}_{i} - \boldsymbol{x}^{T}_{i} \cdot 2\boldsymbol{H}^{T} \boldsymbol{R}_{\mathrm{inv}} \cdot \mathbf{y}^{\mathrm{m}}_{i} \\ &= \boldsymbol{x}^{T}_{i} \cdot \boldsymbol{H}^{T}_{i} \boldsymbol{R}_{\mathrm{inv}} \boldsymbol{H}_{i} \cdot \boldsymbol{x}_{i} - 2\, \boldsymbol{y}^{T}_{i} \boldsymbol{R}^{T}_{\mathrm{inv}} \boldsymbol{H} \cdot \boldsymbol{x}_{i}\end{aligned} \tag{A.42}$$

Finally, the third part of the optimization objective can be reformulated in this way:

$$\begin{aligned}\parallel \boldsymbol{x}_{i} - \boldsymbol{f}(\boldsymbol{x}_{i}) \parallel^{2}_{\boldsymbol{Q}^{-1}} &= \left(\boldsymbol{x}_{i} - \boldsymbol{f}(\boldsymbol{x}_{i-1})\right)^{T} \boldsymbol{Q}^{-T} \boldsymbol{Q}^{-1} \left(\boldsymbol{x}_{i} - \boldsymbol{f}(\boldsymbol{x}_{i-1})\right) \\ &= \boldsymbol{f}^{T}(\boldsymbol{x}_{i-1}) \cdot \boldsymbol{Q}^{-T} \boldsymbol{Q}^{-1} \cdot \boldsymbol{f}(\boldsymbol{x}_{i-1}) - \boldsymbol{f}^{T}(\boldsymbol{x}_{i-1}) \cdot \boldsymbol{Q}^{-T} \boldsymbol{Q}^{-1} \cdot \boldsymbol{x}_{i} \\ &\quad - \boldsymbol{x}_{i} \boldsymbol{Q}^{-T} \boldsymbol{Q}^{-1} \boldsymbol{f}(\boldsymbol{x}_{i-1}) \\ \boldsymbol{F}_{i-1} &= \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{x_{i-1}}, \boldsymbol{\Phi}_{i-1} = \exp(\boldsymbol{F}_{i-1} \cdot T_{s}), \boldsymbol{f}(\boldsymbol{x}_{i-1}) = \boldsymbol{\Phi}_{i-1} \cdot \boldsymbol{x}_{i-1} \\ \boldsymbol{Q}_{\mathrm{inv}} &= \boldsymbol{Q}^{-T} \boldsymbol{Q}^{-1} \\ &= \boldsymbol{x}^{T}_{i-1} \cdot \boldsymbol{\Phi}^{T}_{i-1} \boldsymbol{Q}_{\mathrm{inv}} \boldsymbol{\Phi}_{i-1} \cdot \boldsymbol{x}_{i-1} \\ &\quad - \boldsymbol{x}^{T}_{i-1} \cdot \boldsymbol{\Phi}^{T}_{i-1} \boldsymbol{Q}_{\mathrm{inv}} \cdot \boldsymbol{x}_{i} - \boldsymbol{x}^{T}_{i-1} \cdot \boldsymbol{Q}_{\mathrm{inv}} \boldsymbol{\Phi}_{i-1} \cdot \boldsymbol{x}_{i-1}\end{aligned} \tag{A.43}$$

To achieve the QP interface eq. (A.40), eq. (A.41), eq. (A.42), and eq. (A.43) are rearranged in a matrix formulation $\boldsymbol{G}$ and vector $\boldsymbol{c}^{T}$ formulation:

$$\boldsymbol{\xi}_{k} = \begin{pmatrix} \boldsymbol{x}_{k-M} \\ \boldsymbol{x}_{k-M+1} \\ \vdots \\ \boldsymbol{x}_{k} \end{pmatrix} ; \ \boldsymbol{\Phi}^{QQ}_{i} = \boldsymbol{\Phi}^{T}_{i} \boldsymbol{Q}_{inv} \boldsymbol{\Phi}_{i} ; \ \boldsymbol{H}^{RR}_{i} = \boldsymbol{H}^{T}_{i} \boldsymbol{R}_{inv} \boldsymbol{H}_{i}$$

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{I}_{k-M} + \boldsymbol{\Phi}^{\mathrm{QQ}}_{k-M} & -\boldsymbol{\Phi}^{T}_{k-M} \boldsymbol{Q}_{\mathrm{inv}} & 0 & \cdots & 0 & 0 \\ -\boldsymbol{Q}_{\mathrm{inv}} \boldsymbol{\Phi}_{k-M} & \boldsymbol{H}^{RR}_{k-M} + \boldsymbol{\Phi}^{QQ}_{k-M+1} & -\boldsymbol{\Phi}^{T}_{k-M+1} \boldsymbol{Q}_{\mathrm{inv}} & 0 & \vdots & 0 \\ 0 & -\boldsymbol{Q}_{\mathrm{inv}} \boldsymbol{\Phi}_{k-M+1} & \boldsymbol{H}^{RR}_{k-M+1} + \boldsymbol{\Phi}^{QQ}_{k-M+2} & \ddots & 0 & \vdots \\ \vdots & 0 & \ddots & \ddots & -\boldsymbol{\Phi}^{T}_{k-1} \boldsymbol{Q}_{\mathrm{inv}} & 0 \\ 0 & \vdots & 0 & -\boldsymbol{Q}_{\mathrm{inv}} \boldsymbol{\Phi}_{k-1} & \boldsymbol{H}^{RR}_{k-1} + \boldsymbol{\Phi}^{QQ}_{k-1} & -\boldsymbol{\Phi}^{T}_{k-1} \boldsymbol{Q}_{\mathrm{inv}} \\ 0 & 0 & \cdots & 0 & -\boldsymbol{Q}_{\mathrm{inv}} \boldsymbol{\Phi}_{k-1} & \boldsymbol{H}^{RR}_{k} \end{bmatrix} \tag{A.44}$$

$$\boldsymbol{c}^{T} = (-2\boldsymbol{x}^{T}_{k-M} \boldsymbol{I}_{k-M} \quad -2\boldsymbol{H}^{T} \boldsymbol{R}_{\mathrm{inv}} \boldsymbol{y}_{k-M+1} \quad \cdots \quad -2\boldsymbol{H}^{T} \boldsymbol{R}_{\mathrm{inv}} \boldsymbol{y}_{k})$$

A.6 Kalman Filter based Parameter Estimation

In this brief chapter a description of the most important parameter estimation approach is given.

A.6.1 SR-UKF Parameter Estimation

The SR-UKF parameter estimation algorithm is a computationally efficient version $\mathrm{O}(ML^2)$ and can be found in [Mer01]. It makes use of an exponential weighting of past data. The forgetting factor $\gamma \approx 0.9995$ is similar to the WSLR approach (see Chapter A.3.1) and therefore avoids the costly $\mathrm{O}(L^3)$ QR and Cholesky operations which are necessary in the state estimation filter algorithm [Mer01]. The algorithm is given in App. Table A.8.

App. Table A.8: The square-root unscented Kalman filter parameter estimation algorithm

```
Initialization:
        $\widehat{\boldsymbol{w}}_0^+ = \mathrm{E}(\boldsymbol{w}_0)$
        $\boldsymbol{CP}_{w_0}^+ = \mathrm{chol}\left(\mathrm{E}\big((\boldsymbol{w}_0 - \widehat{\boldsymbol{w}}_0^+)(\boldsymbol{w}_0 - \widehat{\boldsymbol{w}}_0^+)^T\big)\right)$
for $k = 1,2,\dots$ $(k \in \mathbb{N}^+)$:
    Predict:
        $\widehat{\boldsymbol{w}}_k^- = \widehat{\boldsymbol{w}}_{k-1}^+$
        $\boldsymbol{CP}_{\boldsymbol{w}_k}^+ = \gamma^{-1/2} \cdot \boldsymbol{CP}_{w_k}^+$
        $\boldsymbol{W}_{k-1} = [\widehat{\boldsymbol{w}}_k, \widehat{\boldsymbol{w}}_k + \gamma \cdot \boldsymbol{CP}_{\boldsymbol{w}_k}^+, \widehat{\boldsymbol{w}}_k - \gamma \cdot \boldsymbol{CP}_{\boldsymbol{w}_k}^+]$
        $\boldsymbol{D}_{k|k-1} = \boldsymbol{h}\left(\boldsymbol{f}_{k|k-1}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}, \boldsymbol{W}_{k-1})\right)$
        $\widehat{\boldsymbol{d}}_k^- = \sum_{i=0}^{2\cdot n} \boldsymbol{w}_i^{\mathrm{m}} \cdot \boldsymbol{D}_{i,k|k-1}$
    Correct:
        $\boldsymbol{CP}_{\boldsymbol{d}_k} = \mathrm{LQ}\left(\left[\sqrt{\boldsymbol{w}_1^{\mathrm{c}}} \cdot (\boldsymbol{D}_{1:2\cdot n,k} - \widehat{\mathbf{w}}_k^-), \sqrt{\boldsymbol{R}_w}\,\right]\right)$
        $\boldsymbol{CP}_{\boldsymbol{d}_k} = \mathrm{cholupd}\big(\boldsymbol{CP}_{\boldsymbol{d}_k}, \boldsymbol{D}_{0,k} - \hat{\mathbf{d}}_k^-, \boldsymbol{w}_0^{\mathrm{c}}\big)$
        $\boldsymbol{P}_{\boldsymbol{w}_k\boldsymbol{d}_k} = \sum_{i=0}^{2\cdot n} \boldsymbol{w}_i^{\mathrm{c}} \cdot \big[\boldsymbol{W}_{i,k|k-1} - \widehat{\boldsymbol{w}}_k^-\big]\big[\boldsymbol{D}_{i,k|k-1} - \hat{\mathbf{d}}_k^-\big]^T$
        $\boldsymbol{K}_k = \boldsymbol{P}_{\boldsymbol{w}_k\boldsymbol{d}_k} \cdot \big(\boldsymbol{CP}_{\boldsymbol{d}_k}' \cdot \boldsymbol{CP}_{\boldsymbol{d}_k}\big)^{-1}$
        $\widehat{\boldsymbol{w}}_k^+ = \widehat{\boldsymbol{w}}_k^- + \boldsymbol{K}_k \cdot (\boldsymbol{y}_k^{\mathrm{m}} - \widehat{\boldsymbol{d}}_k^-)$
        $\boldsymbol{U} = \boldsymbol{K}_k \cdot \boldsymbol{CP}_{\boldsymbol{d}_k}$
        $\boldsymbol{CP}_{\boldsymbol{w}_k}^+ = \mathrm{cholupd}\big(\boldsymbol{CP}_{\boldsymbol{w}_k}^-, \boldsymbol{U}, -1\big)$
```

B Implementation of the DLR Kalman Filter Library

This chapter is an extended version of [Bre11c] and [Bre14b]. The implementation is based on an extended FMI 2.0 RC1 implementation [Mod13] an international standard for the exchange of models between different simulation tools e.g. multi-body or electric circuit simulation. Through the standardized interface consisting of C-code and/or binary model files and the model description in a defined XML scheme it is possible to use these models also for extended applications, besides simulation, as it is done in the DLR Kalman Filter Library.

B.1 FMI for State Estimation with Dymola/Modelica

The standard FMI co-simulation interface allows integrating eq. (A.17) from sample instant t_{k-1} to t_k with function `fmiDoStep(..)` and therefore computing eq. (A.18). In standard co-simulation the continuous-time states of a model are hidden in the co-simulation slave (see Chapter 4 in [Mod13]). However, for state estimation the states need to be explicit and it must be possible to reset the states at sample instants, see Chapter 4.2.2. In order to achieve this, Dymola 2014 FD01 and later versions which have support for FMI 2.0 co-simulation according to [Mod13], have been extended with the needed features. Especially,

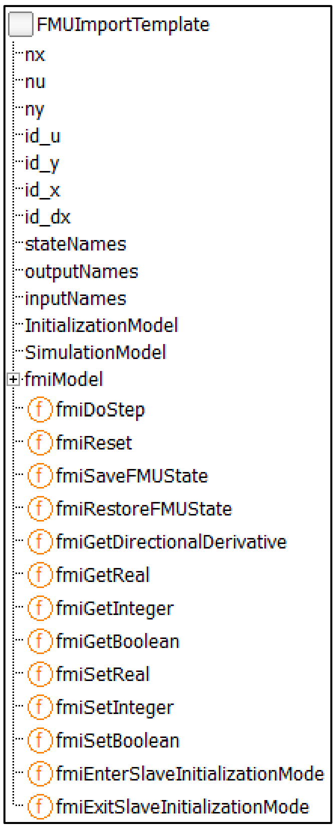
- the continuous-time states are reported in the `modelDescription.xml` file under element `ModelStructure`,
- it is possible to explicitly set the continuous-time states with `fmiSetReal(..)` before `fmiDoStep(..)` is called,
- it is possible to inquire the actual values of all variables with `fmiGetReal(..)` after `fmiSetReal(..)` was called, without an `fmiDoStep(..)` in between,
- when importing an FMU for co-simulation into Modelica, Dymola optionally generates the Modelica code according to the `FMUImportTemplate` package shown in the next section. This package serves as interface to access the needed FMI functionality from a Modelica model or function.

B.2 The FMU Import Template Package

Importing an FMU means to generate a package that contains all the functionality needed to simulate the FMU or use it in a state estimator. For this the template package `FMUImportTemplate` is provided, see the code excerpt and the figure below. The imported FMU extends from the `FMUImportTemplate` and redeclares all elements.

```
partial package FMUImportTemplate
  constant Integer nx=1;
  …
  constant Integer id_x[nx];
  …
  constant String stateNames[nx];
  …
  replaceable model SimulationModel
  end SimulationModel;

  replaceable model InitializationModel
```

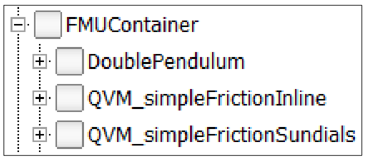
```
      fmiModel fmi;
      parameter Real fmiInitOk(fixed=false);
    end InitializationModel;


    replaceable partial class fmiModel
      extends ExternalObject;
      function constructor
        …
      end constructor;
    …
    end fmiModel;

    replaceable function fmiDoStep
        input fmiModel fmi;
        …
    end fmiDoStep;
    …
  end FMUImportTemplate;
```



```
FMUImportTemplate
  nx
  nu
  ny
  id_u
  id_y
  id_x
  id_dx
  stateNames
  outputNames
  inputNames
  InitializationModel
  SimulationModel
  fmiModel
  fmiDoStep
  fmiReset
  fmiSaveFMUState
  fmiRestoreFMUState
  fmiGetDirectionalDerivative
  fmiGetReal
  fmiGetInteger
  fmiGetBoolean
  fmiSetReal
  fmiSetInteger
  fmiSetBoolean
  fmiEnterSlaveInitializationMode
  fmiExitSlaveInitializationMode
```

Important dimensions of the FMU such as the number of continuous states `nx`, inputs `nu` and outputs `ny` are set in the imported FMU package. Moreover, the FMI references are available by the vectors `id_x`, `id_dx`, `id_u` and `id_y` for state, state derivative, input and output variables. It is also important to get variable names for states, inputs and outputs. Otherwise the order of the components in the vectors `x`, `u`, `y` would be only visible by user-unfriendly reference values instead of variable names. The names are used in the parameter GUIs of the filter model in the next sub-section and in the input and output bus of a filter model.

The imported FMU package contains two models: `SimulationModel` and `InitializationModel`. The model `SimulationModel` is a fully operating Modelica model (with inputs and outputs) that wraps the extended FMU 2.0 for co-simulation whereas in `InitializationModel` only the FMU is instantiated by the external object `fmiModel` and the FMI initialization phase is executed. The `InitializationModel` is used in a Kalman filter model; the `SimulationModel` is contained for completeness to use the imported FMU package also for other applications like a "real" FMU for co-simulation in the Modelica simulation environment.



```
FMUContainer
  DoublePendulum
  QVM_simpleFrictionInline
  QVM_simpleFrictionSundials
```

The FMU package provides interface functions to all (or at least most) of the functions defined in the FMI co-simulation standard 2.0. For a user-convenient handling of the FMU import process, it is desirable to import an FMU as a sub-package into an existing Modelica package. The default package is the package FMUContainer that hosts several imported FMUs, see figure on the right side.

B.3 Model Functions for State Estimators

The state estimator algorithms are implemented with Modelica functions that provide the needed model evaluations. In partial package `BaseFunctions` the interfaces of these functions are defined and in package `SystemFunctions` the function prototypes are collected. The latter are replaceable functions that provide the needed functionality of Table 4.3 (the right column of this table lists the name of the function).

For example, partial function `fBase` is defined as:

```
partial function fBase "Base class of the
          state equation dx/dt = f(x,u,t)"
  input Integer nx "Number of states";
  input Integer nu "Number of inputs";
  input Real x[nx] "States";
  input Real u[nu] "Inputs";
  input Modelica.SIunits.Time t "Time";
  output Real dxdt[nx] "Derivatives";

end fBase;
```

The dimensions `nx, nu` are conceptually not necessary, because the dimensions could be determined by the size of the vectors `x` and `u`. The function prototypes are collected in package `SystemFunctions`:

```
partial package SystemFunctions
  replaceable function f
    extends fBase;
  end f;
  …
  replaceable function integrator
     extends integratorBase;
  end integrator;

end SystemFunctions;
```

For a particular model, an implementation of the `SystemFunctions` functions has to be provided. For FMUs, this is performed with the generic package `FMISystemFunctions`. The implementation is based on the `FMUImportTemplate` package and holds therefore for every FMU that extends from this template package.

```
package FMISystemFunctions
  extends SystemFunctions;
  replaceable package FMU
           constrainedby FMUImportTemplate;
  redeclare function extends f
   input FMU.fmiModel fmi;
  algorithm
    FMU.fmiSetReal(fmi, FMU.id_u, u);
    FMU.fmiSetReal(fmi, FMU.id_x, x);
    dxdt := FMU.fmiGetReal(fmi, FMU.id_dx);
  end f;…
  redeclare function extends integrator
    input FMU.fmiModel fmi;
  algorithm
    FMU.fmiSaveFMUState(fmi);
    FMU.fmiSetReal(fmi, FMU.id_u, u);
    FMU.fmiSetReal(fmi, FMU.id_x, x);
    FMU.fmiDoStep(fmi, t, dt, 0);
    xNew := FMU.fmiGetReal(fmi, FMU.id_x);
    FMU.fmiRestoreFMUState(fmi);
  end integrator;

end FMISystemFunctions;
```

The system functions `f`, `h`, `integrator` can be directly implemented with functions provided in `FMUImportTemplate`. The Jacobians `fx` and `hx` are implemented by computing them numerically with finite difference quotients or directional function `fmiGetDirectionalDerivatives` (if the tool supports this for co-simulation), then this function can be directly called and will provide a more efficient and reliable evaluation of the Jacobians.

Dymola supports two techniques for the FMI function `fmiDoStep`. Either the Sundials solvers [Hin05] are used (that are integrators with variable step size and error control) to numerically integrate the model equations, or inline integration [Elm95] is applied, which means fixed step solvers are embedded in the model equations. The DLR Kalman Filter Library works with both techniques. For real-time applications, fixed-step methods have to be used and therefore a Kalman filter will usually utilize inline integration.

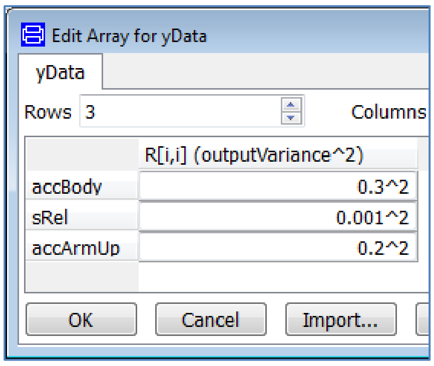
The functions `fmiSave/RestoreFMUState` in the above code fragments are auxiliary functions that call the FMI functions `fmiGet/Set/FreeFMUstate` to enable several calls of `fmiDoStep` starting at the same time instant, as needed, for example, for the UKF algorithm (compare Table 4.2).

B.4 Tailored Kalman Filter Models in Modelica

Based on the imported FMU package an individual Kalman filter model has to be generated. In the DLR Kalman Filter Library this can be performed automatically by use of a Modelica/Dymola scripting function. The idea is to define an input bus `InBus` and an output bus `OutBus` for exchanging variables between the filter model and higher level models. The names of the bus variables correspond to the variable names of the imported FMU – only ".", ",", "[", "]" and " " are replaced by "_" due to the Modelica syntax. The bus definitions for the use case example in Chapter B.5.1 are listed below:

```
encapsulated expandable connector InBus

  import Modelica;
  extends Modelica.Icons.SignalBus;
  // Model Inputs
  Real u;
  // Measured Model Outputs
  Real accBody;
  Real sRel;
  Real accArmUp;

end InBus;
encapsulated expandable connector OutBus
  import Modelica;
  extends Modelica.Icons.SignalBus;
  // Estimated Model States
  Real mass_wheel_s;
  Real mass_wheel_v;
  Real mass_body_s;
  Real mass_body_v;
  Real …FirstOrderShapingFilter_s;
  // Estimated Model Outputs
  Real accBody;
  Real sRel;
  Real accArmUp;
end OutBus;
```

The advantage of this approach is that not vectors of anonymous variables are defined, but bus variables with meaningful names are tailored to each individual FMU. The main state estimation algorithms are implemented in sub-functions and in a partial filter model, e.g. for an UKF (see Chapter A.4.1). This model defines several variables and parameters for the filter algorithm that is called at each sample point of a sampled integration time interval. In the filter model also an instance of `InitializationModel` of the imported FMU package is included. Together with the package `FMISystemFunction` all necessary parts are put together to run FMI based Kalman filter algorithms within a Modelica model.



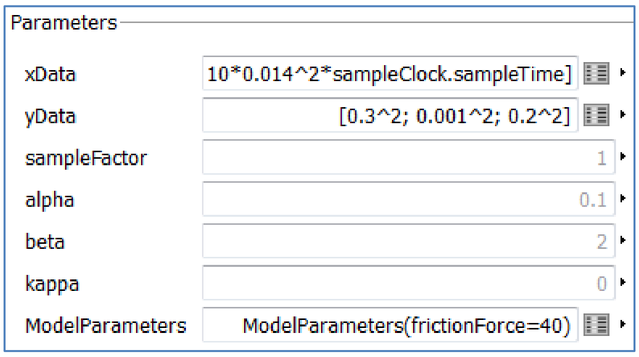
	R[i,i] (outputVariance^2)
accBody	0.3^2
sRel	0.001^2
accArmUp	0.2^2

App. Figure B.1: Parameter menu for output variances with names of output variables

A further improvement of the user interface compared to the first version proposed in [Bre11c] are the filter parameters like state and output variances that are shown in lists with names of the respective variables – instead of indices of vectors, see App. Figure B.1. Basically, a matrix is defined and then via Dymola specific annotations row and column headings can be added to the parameter menu. For example the menu in App. Figure B.2 is defined in the following way:

```
parameter Real yData[FMUPackage.ny,1]
  annotation(Dialog(
    __Dymola_columnHeadings =
      {"R[i,i] (outputVariance^2)"},
    __Dymola_rowHeadings =
      {"accBody", "sRel", "accArmUp"}));
```

In the parameter menu of the filter in App. Figure B.2 the user can press the button on the right side of `yData` to get the menu of App. Figure B.1. Also the model parameters of the FMU may be modified by clicking on the button on the right side of `ModelParameters`.



xData	10*0.014^2*sampleClock.sampleTime]
yData	[0.3^2; 0.001^2; 0.2^2]
sampleFactor	1
alpha	0.1
beta	2
kappa	0
ModelParameters	ModelParameters(frictionForce=40)

App. Figure B.2: Menu of a SR-UKF Kalman Filter model

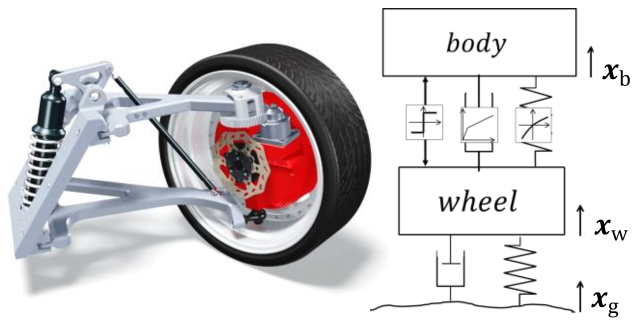
B.5 Additional Observer Examples

In this chapter state estimator examples are discussed that make use of the DLR Kalman Filter Library and which are not directly connected to the work in this doctoral thesis.

B.5.1 Vehicle Vertical Dynamics State Estimator

As an example application an advanced state estimator for the vertical dynamics of ROboMObil by means of the state estimator framework is developed. A more detailed version of this application case is available in [Fle13].

In contrast to fully active suspension systems, semi-active dampers need far less energy [Wil94]. Therefore, a diversity of control strategies for semi-active dampers is presented in literature. An overview about these control strategies can be found in [Sav10] and [Gug08]. As most of these strategies need state feedback, but not all states can be measured, state estimation becomes an important topic during the design of semi-active suspension systems $\boldsymbol{x}_{\mathrm{g}}$.

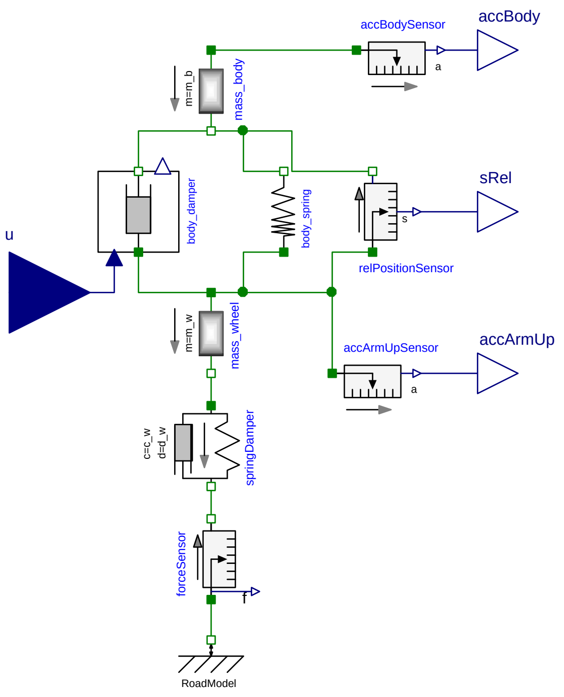

body
$\boldsymbol{x}_{\mathrm{b}}$
wheel
$\boldsymbol{x}_{\mathrm{w}}$
$\boldsymbol{x}_{\mathrm{g}}$

App. Figure B.3: Left: ROMO wheel robot, right: nonlinear two mass system

The DLR Kalman Filter Library offers an easy to use framework for the development of state estimators for nonlinear systems like this semi-active suspension system. Especially, a square-root based UKF implementation (cf. Chapter A.4.2), named SR-UKF, is well suited for this highly nonlinear system, because of its higher order linearization accuracy of mean and covariance. Moreover, the nonlinear parts can be easily taken into account and in comparison to an EKF algorithm (cf. Chapter A.3.3) no derivatives and Jacobians are needed.

The Nonlinear Quarter Vehicle Model

The suspension system of one wheel robot is modeled as a nonlinear two mass system (see App. Figure B.3 – right) as described in [Fle13]. The corresponding implementation in Modelica is shown in App. Figure B.4. The model consists of the two masses “mass_body” and “mass_wheel”, a linear spring damper component, which approximates the wheel behavior, a road model as explained in [Mit04] or [Koc10] and the “body_spring” and “body_damper”.



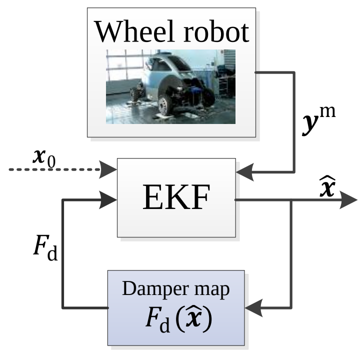
App. Figure B.4: Nonlinear two mass system of a wheel robot in Modelica

As the motion of these two components are connected to the wheel and body motion by a push rod-rocker kinematic (compare App. Figure B.3 – left), the motion and the force of these components is scaled by a transmission ratio. Details on the nonlinear characteristics of the "body_damper" are shown in [Fle13].

The third nonlinearity of the two mass system, besides the transmission ratio and the damper characteristic, is the friction of the suspension system. It covers the friction of the damper and of all joints of the suspension system. For the state estimation the friction force F_f is modeled without stiction by a smooth tanh-switching function:

$$F_\mathrm{f} = F_\mathrm{f,const} \cdot \tanh(v_\mathrm{d}/\epsilon_\mathrm{f}). \tag{B.1}$$

Here $F_\mathrm{f,const}$ represents a constant sliding friction. The direction of the friction force is determined according to the current velocity difference v_d between body and wheel. The parameter ϵ_f is used to define the transitional behavior of the tanh-function.



App. Figure B.5: State of the art EKF based vertical dynamics estimator

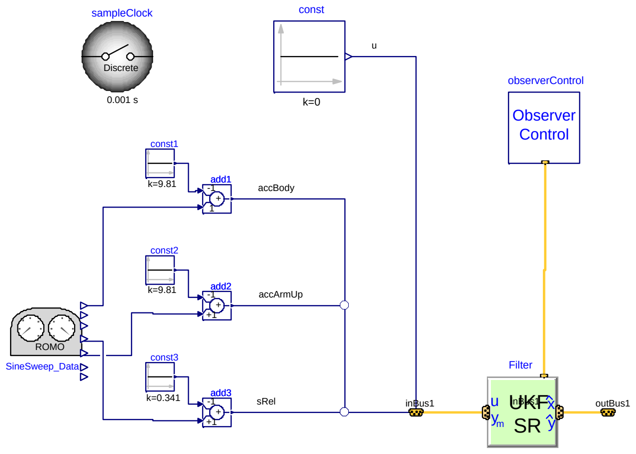
In [Koc10] the proposed Extended Kalman Filter is extended by an extra fictitious input to handle the damper mapping, shown in App. Figure B.5. By using the FMI based DLR Kalman Filter Library the data table and its interpolation could be fully integrated into the prediction model (App. Figure B.7).

Experimental Setup and Results

The nonlinear two mass system described in the preceding section is integrated in an SR-UKF state estimator using the DLR Kalman Filter Library including the extended FMI 2.0 for co-simulation interface and inline integration as described in Chapter 4.2. Subsequently the resulting estimators are applied to measurement data recorded with ROboMObil on a four-post test rig (App. Figure B.6).

App. Figure B.6: ROMO on a four-post test rig

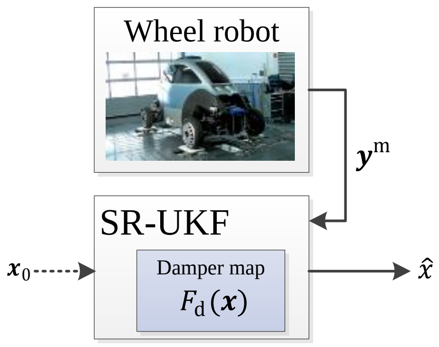
App. Figure B.7 shows the Modelica model of the SR-UKF vehicle vertical dynamics estimator. On the left-hand side the measurement data is read by a “CombiTimeTable” and on the right-hand side the estimator, called “Filter”, and its corresponding settings block “observerControl” can be found. The estimator uses three measurement inputs: the acceleration of the body above the wheel, the wheel acceleration and the damper deflection.



App. Figure B.7: SR-UKF vertical dynamics estimator in Modelica

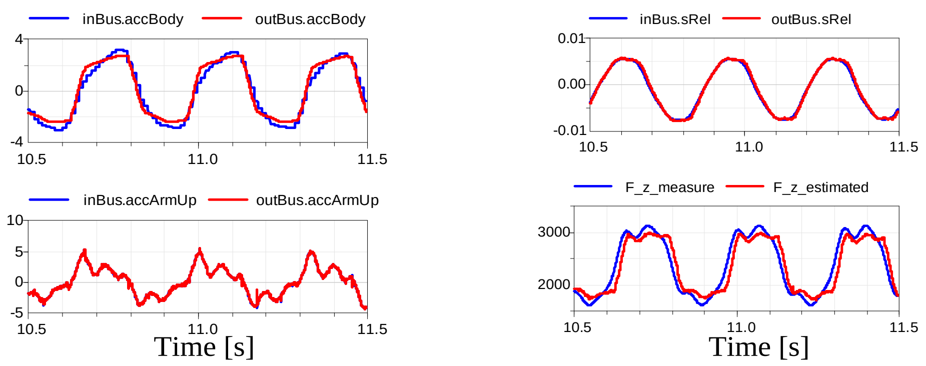
The parameters of the estimators, as well as the system covariances, are tuned according to the optimization procedure presented in [Fle13]. The measurement covariances are set according to the sensor noise. The experimental setup is shown in App. Figure B.8.

The performance of the estimator, subject to a sine sweep excitation, is shown in App. Figure B.9 by comparing the measured and the estimated tire contact forces in the last plot. Please notice that, the tire force $F_{z_{\mathrm{measure}}}$ is only available on the four-post test rig (App. Figure B.6). It is used for validating the estimator performance and not as a measurement output $\boldsymbol{y}^{\mathrm{m}}$ (compare experimental setup in App. Figure B.7). It can be seen that the estimator reproduces the measurements and the tire contact force with a good accuracy.


Wheel robot
$\boldsymbol{y}^{\mathrm{m}}$
SR-UKF
$\boldsymbol{x}_0$
Damper map
$F_{\mathrm{d}}(\boldsymbol{x})$
$\hat{x}$

App. Figure B.8: Proposed vertical dynamics estimator setup

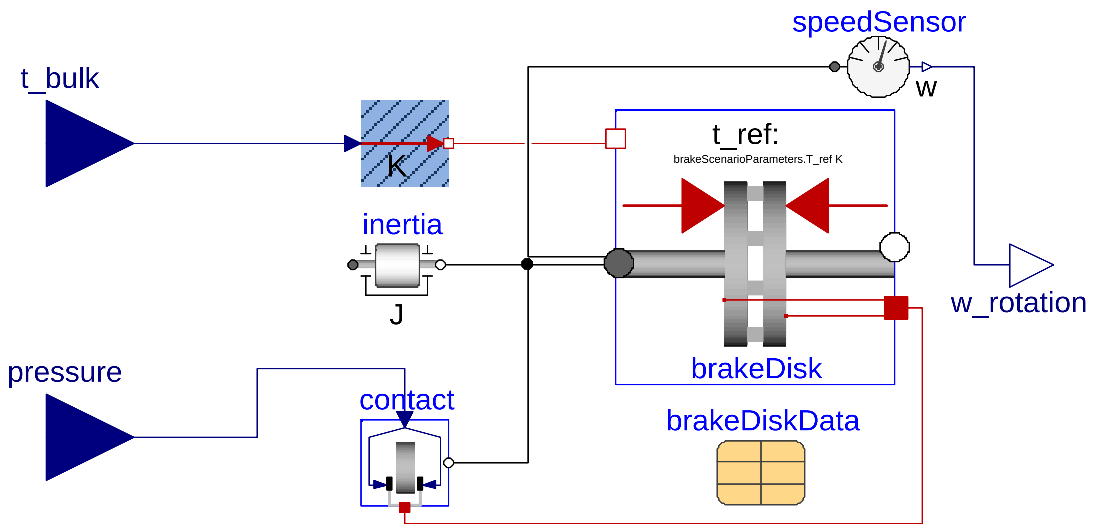
As the body acceleration sensor has the largest noise level, the weighting of its measurements “accBody” was chosen in such a way that the estimator relies more on the damper deflection “sRel” and the wheel acceleration “accArmUp”.



App. Figure B.9: Comparison results to measurements – “sine sweep” excitation

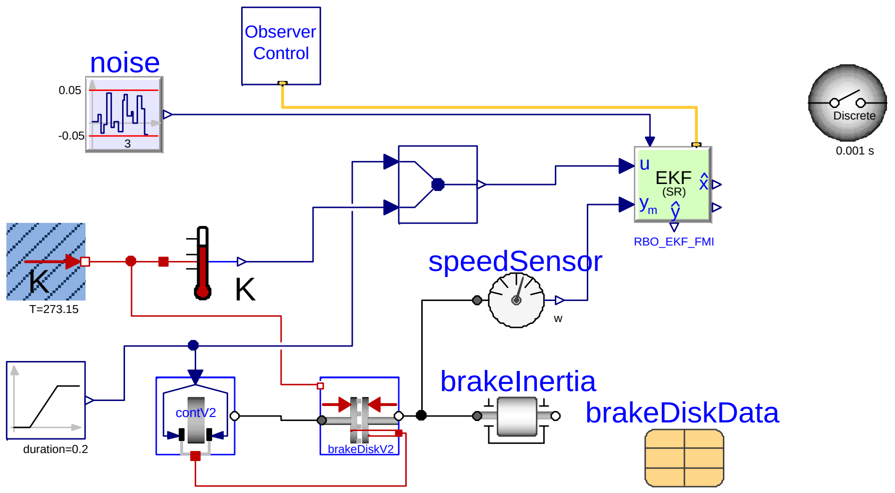
B.5.2 Railroad Disk Brake State Estimator

Besides the already discussed observer example, the author contributed to the modelling and setup of a railroad disk brake observer [Hec15], which was part of the Modrio project [Ite12]. In App. Figure B.10 the prediction model for the brake disk observer is shown. Here, an axial temperature field of the disk and the friction contacts are modeled in a nonlinear way [Hec15]. With real world experiments on the dynamometer rig of Knorr-Bremse it has been possible to identify the model parameters using the DLR Optimization Library [Pfe12].



App. Figure B.10: Railroad brake disk prediction model

An experimental brake disk observer setup using a square-root extended Kalman filter (according to Chapter A.4.2) is depicted in App. Figure B.11. In this configuration the measurements for the observer were generated by a second brake disk model and disturbed through random Gaussian noise to simulate the behavior of real measuring feeders.



App. Figure B.11: Railroad brake observer setup

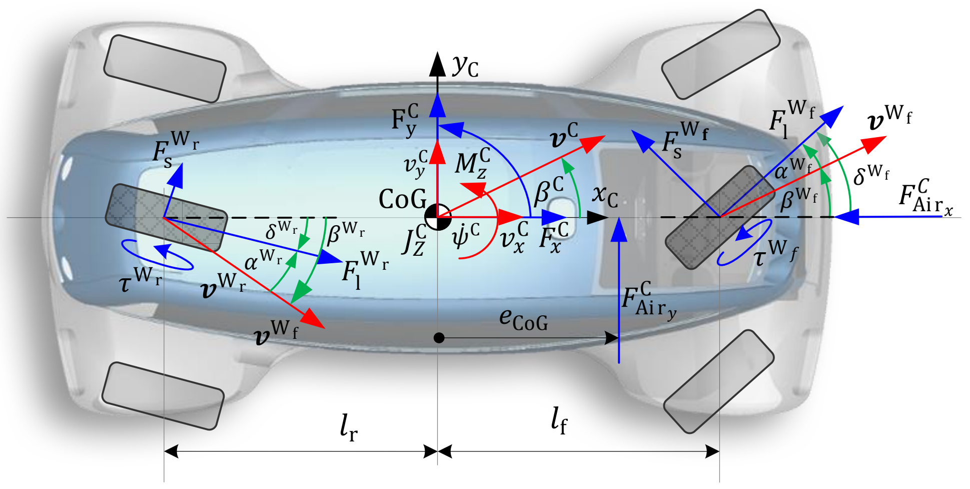
The observer’s task is to estimate the unknown actual disk temperature and thereby the friction coefficient between the brake pads and the disk. This measure is necessary to optimize the brake control of modern train systems with the purpose of exact automated braking to standstill in a rail way station or to minimize the braking distance. In [Hec15] comprehensive information about the detailed modeling approach, the parameters and the optimization procedure are given. Since simulative studies with test bench data showed good results, this approach is currently further investigated for the possibility of online application capability.

C Miscellaneous

In this chapter different topics are collected which were not suitable for the main part of this thesis.

C.1 The Extended Single Track Observer Synthesis Model

In this section the extended single track model (ESTM) used for the observer synthesis in Chapter 5.2 is derived. To reduce the model complexity both the front and rear axle are reduced to one single wheel each.



App. Figure C.1: Vehicle dynamics quantities of the extended single track model

C.1.1 The Nonlinear Single Track Model Equations

A good description of the equations of motion is given in [Bue98] appendix A.1 resulting in the following set of ordinary differential equations:

$$\begin{aligned}
\frac{\mathrm{d}\beta^C}{\mathrm{d}t} &= \frac{-\sin(\beta^C)\,F_x^C + \cos(\beta^C)\,F_y^C}{m \cdot v_{\mathrm{mod}}^C} - \dot{\psi}^C \\
\frac{\mathrm{d}v^C}{\mathrm{d}t} &= \frac{\cos(\beta^C)F_x^C + \sin(\beta^C)\,F_y^C}{m} \\
\frac{\mathrm{d}\dot{\psi}^C}{\mathrm{d}t} &= \frac{M_z^C}{J_Z^C} \\
\frac{\mathrm{d}\psi_C}{\mathrm{d}t} &= \dot{\psi}^C \\
\frac{\mathrm{d}x_C}{\mathrm{d}t} &= v^C \cdot \cos(\psi_C + \beta^C) \\
\frac{\mathrm{d}y_C}{\mathrm{d}t} &= v^C \cdot \sin(\psi_C + \beta^C)
\end{aligned} \tag{C.1}$$

The vehicle side slip β^C is the angle between the vehicle origin coordinate system and the actual vehicle velocity vector $\boldsymbol{v}^C$. The yaw angle ψ_C and yaw angle rate $\dot{\psi}^C$ describe the orientation with respect to a fixed inertial system, in which also the vehicle positions $x_\mathrm{C}, y_\mathrm{C}$ are expressed. Additional quantities are the vehicle mass m, the yaw moment around the center of gravity M_Z^C, as well as the corresponding vehicle yaw inertia J_Z^C. To prevent division by zero the denominator of the vehicle side slip state $\dot{\beta}^\mathrm{C}$ is calculated by the use of eq. (C.2) which limits the minimum velocity to $v_\mathrm{min}^\mathrm{C}$.

$$v_\mathrm{mod}^\mathrm{C} = \frac{\sqrt{v^\mathrm{C} \cdot v^\mathrm{C} + 4 \cdot v_\mathrm{min}^\mathrm{C} \cdot v_\mathrm{min}^\mathrm{C}}}{2} \tag{C.2}$$

To enhance the model fidelity the STM is not linearized in contrast to [Bue98]. The forces (marked blue in App. Figure C.1) to the vehicle body center of gravity are calculated by considering the geometric dependencies to the instantaneous steering angles $\delta^{(\cdot)}$:

$$\begin{aligned}
F_x^\mathrm{C} &= -\sin\left(\delta^{\mathrm{W_f}}\right) F_\mathrm{s}^{\mathrm{W_f}} - \sin\left(\delta^{\mathrm{W_r}}\right) F_\mathrm{s}^{\mathrm{W_r}} \\
&\qquad + \cos\left(\delta^{\mathrm{W_f}}\right) F_\mathrm{l}^{\mathrm{W_f}} + \cos\left(\delta^{\mathrm{W_r}}\right) F_\mathrm{l}^{\mathrm{W_r}} + F_{\mathrm{Air}_x}^\mathrm{C} \\
F_y^\mathrm{C} &= \cos\left(\delta^{\mathrm{W_f}}\right) F_\mathrm{s}^{\mathrm{W_f}} + \cos\left(\delta^{\mathrm{W_r}}\right) F_\mathrm{s}^{\mathrm{W_r}} \\
&\qquad + \sin\left(\delta^{\mathrm{W_f}}\right) F_\mathrm{l}^{\mathrm{W_f}} + \sin\left(\delta^{\mathrm{W_r}}\right) F_\mathrm{l}^{\mathrm{W_r}} + F_{\mathrm{Air}_y}^\mathrm{C} \\
M_Z^\mathrm{C} &= l_\mathrm{f} \cdot \cos\left(\delta^{\mathrm{W_f}}\right) F_\mathrm{s}^{\mathrm{W_f}} \\
&\quad - l_\mathrm{r} \cdot \cos\left(\delta^{\mathrm{W_r}}\right) F_\mathrm{s}^{\mathrm{W_r}} + l_\mathrm{f} \cdot \sin\left(\delta^{\mathrm{W_f}}\right) F_\mathrm{l}^{\mathrm{W_f}} \\
&\quad - l_\mathrm{r} \cdot \sin\left(\delta^{\mathrm{W_r}}\right) F_\mathrm{l}^{\mathrm{W_r}} + e_\mathrm{CoG} \cdot F_{\mathrm{Air}_y}^\mathrm{C} \\
F_{\mathrm{Air}_y}^\mathrm{C} &= 0.5 \cdot c_{\mathrm{w}_y} \cdot \rho \cdot A_y \cdot {v_{\mathrm{Air}_y}^\mathrm{C}}^2
\end{aligned} \tag{C.3}$$

In this set of equations $F_\mathrm{s}^{(\cdot)}$ denotes the wheel side forces and $F_\mathrm{l}^{(\cdot)}$ the corresponding longitudinal forces. The distances from the CoG to the rear and front wheels are given by $l_{(\cdot)}$, and the external longitudinal $F_{\mathrm{Air}_x}^\mathrm{C}$ and lateral $F_{\mathrm{Air}_y}^\mathrm{C}$ air drag forces are applied to the vehicle body. The lateral forces of the tire are calculated by trigonometric functions based on Pacejka's tire model [Pac12] whose input variable is the wheel's side slip angle $\alpha^{\mathrm{W}_{(\cdot)}}$:

$$\begin{aligned}
F_\mathrm{s}^{\mathrm{W_f}} &= D \cdot \sin\left(C \cdot \operatorname{atan}\left(B \cdot \alpha^{\mathrm{W_f}} - E \cdot \left(B \cdot \alpha^{\mathrm{W_f}} - \operatorname{atan}\left(B \cdot \alpha^{\mathrm{W_f}}\right)\right)\right)\right) \\
F_\mathrm{s}^{\mathrm{W_r}} &= D \cdot \sin\left(C \cdot \operatorname{atan}\left(B \cdot \alpha^{\mathrm{W_r}} - E \cdot \left(B \cdot \alpha^{\mathrm{W_r}} - \operatorname{atan}\left(B \cdot \alpha^{\mathrm{W_r}}\right)\right)\right)\right) \\
\alpha^{\mathrm{W_f}} &= \left(\delta^{\mathrm{W_f}}\right) - \operatorname{atan}\left(\frac{v_\mathrm{mod}^\mathrm{C} \cdot \sin\left(\beta^\mathrm{C}\right) + l_\mathrm{f} \cdot \dot{\psi}^\mathrm{C}}{v_\mathrm{mod}^\mathrm{C}} \cdot \cos\left(\beta^\mathrm{C}\right)\right) \\
\alpha^{\mathrm{W_r}} &= \left(\delta^{\mathrm{W_r}}\right) - \operatorname{atan}\left(\frac{v_\mathrm{mod}^\mathrm{C} \cdot \sin\left(\beta^\mathrm{C}\right) - l_\mathrm{r} \cdot \dot{\psi}^\mathrm{C}}{v_\mathrm{mod}^\mathrm{C}} \cdot \cos\left(\beta^\mathrm{C}\right)\right)
\end{aligned} \tag{C.4}$$

Herein the factor B denotes the tire's stiffness, C is an aspect ratio for the calculated force, D the maximum value of the curve, and E is the modulus of bending rupture. For the longitudinal tire dynamics it has been decided to use a simplified modeling without the consideration of the actual wheel slip, since experiments in combination with observer algorithms yielded to poor as well as unstable results and the vehicle's acceleration range of the here considered maneuvers is limited:

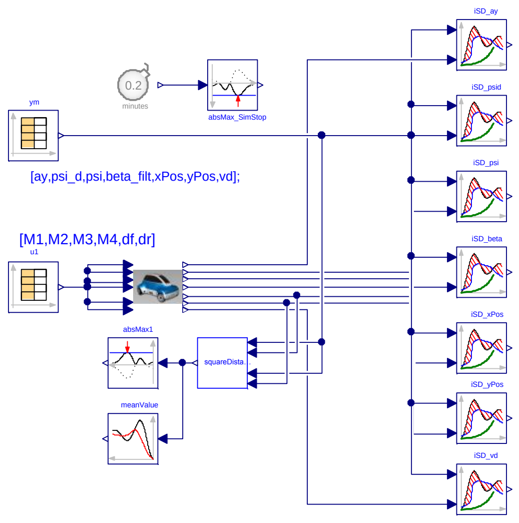
$$\begin{aligned}
F_{\mathrm{L}}^{\mathrm{W_f}} &= \frac{\tau^{\mathrm{W_1}}}{R} + \frac{\tau^{\mathrm{W_2}}}{R} - f_{\mathrm{rv}} \cdot \left(\frac{m \cdot l_{\mathrm{r}} \cdot g}{l_{\mathrm{f}} + l_{\mathrm{r}}}\right) \\
F_{\mathrm{L}}^{\mathrm{W_r}} &= \frac{\tau^{\mathrm{W_3}}}{R} + \frac{\tau^{\mathrm{W_4}}}{R} - f_{\mathrm{rv}} \cdot \left(\frac{m \cdot l_{\mathrm{f}} \cdot g}{l_{\mathrm{f}} + l_{\mathrm{r}}}\right)
\end{aligned} \tag{C.5}$$

Finally, the speed dependent rolling resistance f_{rv} and the longitudinal air drag $F_{\mathrm{Air}_x}^{\mathrm{C}}$ are calculated according to eq. (C.4). By use of the "if-statement" it is prevented that the vehicle rolls backwards in case of standstill on a planar surface and no braking torque is applied:

$$\begin{aligned}
&\text{if } v^{\mathrm{C}} > v_{\mathrm{min}}^{\mathrm{C}} \\
&\quad f_{\mathrm{rv}} = f_{\mathrm{R0}} + \frac{f_{\mathrm{R1}} \cdot v_{\mathrm{mod}}^{\mathrm{C}}}{100} + f_{\mathrm{R4}} \cdot \left(\frac{v_{\mathrm{mod}}^{\mathrm{C}}}{100}\right)^4 \\
&\quad F_{\mathrm{Air}_x}^{\mathrm{C}} = {}^{1}\!/_{2} \cdot c_{\mathrm{w}_x} \cdot \rho \cdot A_x \cdot {v_{\mathrm{Air}_x}^{\mathrm{C}}}^2 \\
&\text{else} \\
&\quad f_{\mathrm{rv}} = 0 \\
&\quad F_{\mathrm{Air}_x}^{\mathrm{C}} = 0 \\
&\text{end}
\end{aligned} \tag{C.6}$$

C.1.2 ROboMObil's Single Track Model Parameter Identification

In the following the procedure of the STM parameter identification by means of a nonlinear model based optimization is described. The derived model is implemented in Modelica and later exported as an FMU for the observer, described in Chapter 4. For the optimization process the DLR Optimization Library [Pfe12] has been utilized. As a tuning algorithm the pattern search method was chosen. For training and validation data real world experiments on vehicle test tracks with ROboMObil have been used.



App. Figure C.2: ROMO's ESTM optimization setup

In App. Figure C.2 the ESTM parameter optimization setup is shown. On the left side input demand data $\boldsymbol{u} = \{\tau^{\mathrm{W}_1}, \tau^{\mathrm{W}_2}, \tau^{\mathrm{W}_3}, \tau^{\mathrm{W}_4}, \delta^{\mathrm{W}_\mathrm{f}}, \delta^{\mathrm{W}_\mathrm{r}}\}$ to ROMO, recorded during real world experiments, are commanded to the ESTM of ROMO. The integral squared deviation between the model outputs and the measured data of ROMO (compare Figure 2.16) and the squared position distance are taken as the minimization objectives. For identification, driving maneuvers e.g. sine sweep steering, sinusoidal steering or free driving were used. For validation, other data sets were utilized to guarantee the reliability of the optimization. The outcome of this procedure is given in the following table:

App. Table C.1: ROMO'S ESTM optimized parameters

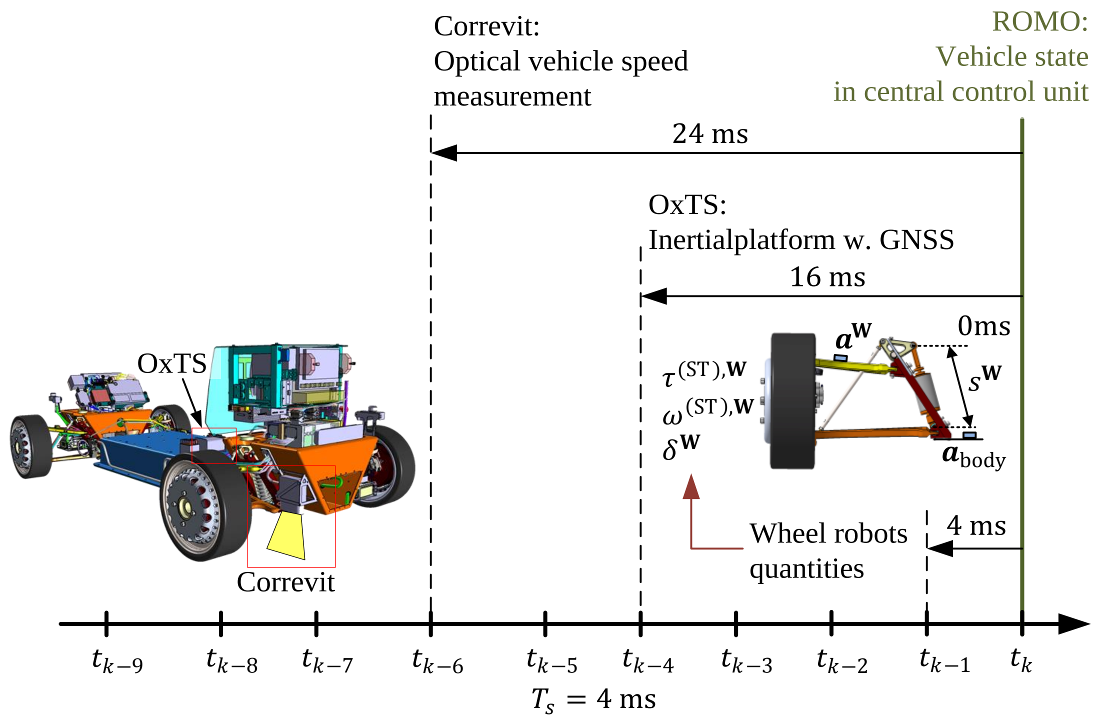
Parameter	Value	Description
f_{R0}	0.009	First parameter of roll resistance
f_{R1}	0.2811588	Second parameter of roll resistance
f_{R4}	0.44906	Fourth parameter of roll resistance
B	5.1088547	Parameter of Pacjeka's magic formula
C	2.0280	Parameter of Pacjeka's magic formula
D	724.70	Parameter of Pacjeka's magic formula
E	0.8903703	Parameter of Pacjeka's magic formula
c_{w_x}	0.3	Longitudinal air drag coefficient
ρ	$1.249512\ [\mathrm{N/m^2}]$	Air density
A_x	$1.2323485\ [\mathrm{m^2}]$	Effective flow surface front
m	1013 [kg]	ROboMObil mass
l_{f}	1.218 [m]	CoG to front wheel
l_{r}	1.182 [m]	CoG to rear wheel
R	0.3722 [m]	Wheel radius
J_Z^{C}	1130 [kgm^2]	Chassis moment of yaw inertia

C.2 ROboMObil's Multiphysical Modelica Component Models

In this appendix chapter the most relevant components for ROMO's multiphysical Modelica model are summarized. The focus of their development has been a real-time capable abstraction depth that enables the usage on a HIL environment [Rit16] or in complex optimization setups where a fast simulation execution is necessary. In this way, it was decided to model the electric components as quasi-stationary models with basic transient effects, which can reproduce the most relevant losses for the EM framework (cf. Chapter 3). The electric traction and electro-mechanic steering motors and the electro hydraulic brake are modeled in a dq-frame approach (see [Eng10] or the manual of DLR PowerTrain Library [Tob07] – PowerTrain.ElectricDrives) which include all the control loops of the real wheel robots, but do overcome the numerical demanding switching of the inverter. The modeled losses are the one in eq. (3.39) already used for the physically cost function of the control allocator (compare Chapter 3.5). The steering mechanism is depicted in Figure 3.16 and includes the additional losses according to eq. (3.40). The battery model applied for the simulative evaluation of the EM is described in Chapter 5.1 in detail and is the same as the prediction model for the battery observer. Besides this model, also a simplified version of a battery model was developed in the bachelor thesis [Enn13] – supervised by the author – for the low voltage backup battery in ROMO's axle modules. Moreover, the models were extended by quasi-stationary thermal models to reproduce the cell heating. Further, electric components are the axle module DC/DC buck converters (see Figure 2.12) which are based on a power balance principle and reproduce the losses due to the inverter based transformation from HV to LV. This model can be found in the DLR PowerTrain Library. ROMO's chassis is modelled as a skate board principle with components from Modelica multibody library and fitted with experimental data from real world experiments [Rit16]. The tires are described by Pacejka's MF-Tire 5.2 extended with rolling resistance and losses [Tob16]. For these two sets of data, the original tires of ROMO a Nankang 165/35 R17 and the L&N Bridgestone 155/55 R18 tire are available. The road to tire contact is realized via a Modelica implementation of the OpenDRIVE [Vir17] interface which is connected to the VIRES virtual world description [Vir17b]. Furthermore, a high fidelity 3D visualization of ROMO and the virtual world (e.g. see Figure 3.29) is carried out by means of the DLR Visualization Library [Hel14].

C.3 Sensor Analysis of ROboMObil

Within in the Bachelor thesis project of [Boe13] the latency and noise behavior of ROMO's sensor system have been systematically analyzed. Please note that this is done by experimental tests and cross correlation since most of automotive suitable sensors are free running ones – that means they do not have an external trigger or synchronization interface and their acquisition system does not offer a unique synchronized time stamp. In App. Figure C.3 an improved version of the outcome is graphically depicted in a scheduling table with respect to the base clock $T_s = 4$ ms of ROMO's central control unit (compare Figure 2.16).


Correvit:
Optical vehicle speed
measurement
ROMO:
Vehicle state
in central control unit
24 ms
OxTS:
Inertialplatform w. GNSS
16 ms
$\boldsymbol{a}^{\mathrm{W}}$
0ms
$\tau^{(\mathrm{ST}),\mathbf{W}}$
$\omega^{(\mathrm{ST}),\mathbf{W}}$
$\delta^{\mathbf{W}}$
$s^{\mathbf{W}}$
$\boldsymbol{a}_{\mathrm{body}}$
OxTS
Wheel robots
quantities
4 ms
Correvit
t_{k-9} t_{k-8} t_{k-7} t_{k-6} t_{k-5} t_{k-4} t_{k-3} t_{k-2} t_{k-1} t_k
$T_s = 4$ ms

App. Figure C.3: Time delay scheduling of ROMO's sensor system

In the following table a list of the sensor information regarding sensor noise σ_s, bias m and latency is summarized. The signals are assumed to be disturbed by additive Gaussian white noise.

App. Table C.2: ROboMObil's vehicle sensors noise, bias and latency

OxTS	**Bias $\boldsymbol{m}$ $[-]$**	**Deviation $\boldsymbol{\sigma_s}$ $[-]$**	**Delay [ms]**
a_x [m/s^2]	$-2.724418224663677\mathrm{e}^{-1}$	$2.769240181700604\mathrm{e}^{-2}$	16
a_y [m/s^2]	$-1.638013449871428\mathrm{e}^{-4}$	$3.138268966012002\mathrm{e}^{-2}$	16
a_z [m/s^2]	$-3.466036054923634\mathrm{e}^{-3}$	$4.848351742705138\mathrm{e}^{-2}$	16
v_x [m/s]	$-1.183700305488312\mathrm{e}^{-3}$	$1.214962752669004\mathrm{e}^{-2}$	16
v_y [m/s]	$4.671748052248540\mathrm{e}^{-3}$	$7.316319010707994\mathrm{e}^{-3}$	16
Correvit			
v_x [m/s]	$3.877289277526296\mathrm{e}^{-4}$	$6.842614450089170\mathrm{e}^{-2}$	24
v_y [m/s]	$-1.942064312033667\mathrm{e}^{-2}$	$6.578393401372411\mathrm{e}^{-2}$	24
$\boldsymbol{a}^{\mathrm{W}}$			
a_x^{W} [m/s^2]	$1.379990943683445\mathrm{e}^{0}$	$4.937188165827279\mathrm{e}^{-1}$	4
a_y^{W} [m/s^2]	$-1.668700471533053\mathrm{e}^{0}$	$4.840309242271431\mathrm{e}^{-1}$	4
a_z^{W} [m/s^2]	$9.064704725918237\mathrm{e}^{0}$	$4.546746182918726\mathrm{e}^{-1}$	4
$\boldsymbol{s}^{\mathrm{W}}$			
s^{W} [mm]	$3.372870728583747\mathrm{e}^{-1}$	$6.585450722185397\mathrm{e}^{-5}$	4
WCU			
ω^{W} [rad/s]	$-4.197454752096141\mathrm{e}^{-3}$	$3.718427530519079\mathrm{e}^{-2}$	4

ω^{ST} [rad/s]	n.a.	n.a.	4
τ^{W} [N/m]	n.a.	n.a.	4
τ^{ST} [N/m]	n.a.	n.a.	4
δ^{W} [rad]	n.a.	n.a.	4

C.4 ROboMObil's Performance Data

In App. Table C.3 the most relevant performance measures of ROboMObil are listed.

App. Table C.3: ROboMObil's performance characteristics

Performance	
Maximum velocity	100 [km/h]
0-80 km/h	~10 [s]
Maximum torque	4 x 160 [Nm]
Maximum brake-torque	4 x 690 (530 mech. + 160 elec.) [Nm]
Power supply	
Battery type	Li-Ion
Number of cells	90s1p
Traction battery nominal voltage (HV)	342 [V]
Nominal capacity	13 [kWh]
On-Board-voltage (LV)	26.8 [V]
Powertrain	
Driving power	$4\ x\ 16$ [kWh]
Recuperation power	$4\ x\ 16$ [kWh]
Braking system	Electro hydraulic by-wire disk brake
Propulsion method	4 in-wheel permanent magnet synchronous machines

In a second table the most relevant electric drive and converter data of ROMO are given:

App. Table C.4: ROboMObil's electric drives characteristics

ROMO traction motor data	**Type: Proprietary**
$P_{\mathrm{inv,const}}$	80.5 [W]
k_{hyst}	0.0832 [Ws/rad]
τ_{fric}	0.1146 [Ws/rad]
k_{eddy}	$7.6151e^{-5}$[Ws^2/rad^2]
k_{inv}	3.8 [W/I]
R_{s}	0.099 [Ohm]
z_{p}	19 [−]
ψ_{PM}	0.0791339 [Wb]

ROMO steering motor data	**Type: Robodrive ILM 70x18**
$P_{\mathrm{inv,const}}$	$\mathrm{n.a.}\,[\mathrm{W}]$
k_{hyst}	$0.011\ [\mathrm{Ws/rad}]$
τ_{fric}	$0.2\ [\mathrm{Ws/rad}]$
k_{eddy}	$4.9e^{-6}[\mathrm{Ws^2/rad^2}]$
k_{inv}	$n.a.\,[\mathrm{W/I}]$
R_{s}	$0.0819\ [\mathrm{Ohm}]$
z_{p}	$10\ [-]$
ψ_{PM}	$0.0059524\ [\mathrm{Wb}]$

C.5 Percentage Goodness of Fit of Two Signals

For the assessment of the consentaneity of two signals with n sample points in the time domain, that is easy to interpret, it is necessary to have a significant quantity. For this reason, a normalized root mean square error function is chosen in eq. (C.7) that gives a percentage value for the goodness of fit of a time series signal to another reference signal. Originally, this has been implemented for the MATLAB ident toolbox [Lju98] for system identification purposes:

$$\mathrm{Fit}\,[\%] = \left(1 - \frac{\sqrt{\sum_{i=1}^{n}\left(\left|(\boldsymbol{y}_i^{\mathrm{m}} - \hat{\boldsymbol{y}}_i\right|^2\right)}}{\sqrt{\sum_{i=1}^{n}\left(\left|\boldsymbol{y}_i^{\mathrm{m}} - \frac{1}{n}\cdot\sum_{i=1}^{n}\boldsymbol{y}_i^{\mathrm{m}}\right|^2\right)}}\right)\cdot 100 \tag{C.7}$$

In eq. (C.7) $\boldsymbol{y}^{\mathrm{m}}$ is the reference signal and $\hat{\boldsymbol{y}}$ the corresponding estimate (e.g. in a Kalman filter application).

D Related Work

D.1 List of Publications as Main Author

The following list includes all peer-reviewed and published articles by the author that are connected to his doctoral thesis project. They are sorted by the date of publication:

- [Bre14b] Brembeck, J., Pfeiffer, A., Fleps-Dezasse, M., Otter, M., Wernersson, K., & Elmqvist, H. (2014). Nonlinear State Estimation with an Extended FMI 2.0 Co-Simulation Interface. In H. Tummescheit, & K.-E. Arzen (Ed.), *Proceedings of 10th International Modelica Conference. 96*, pp. 53-62. Lund: Linköping University Electronic Press.
- [Bre14] Brembeck, J., & Winter, C. (2014). Real-Time Capable Path Planning for Energy Management Systems in Future Vehicle Architectures. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 599-604). Dearborn, MI, USA: IVS IEEE.
- [Bre12] Brembeck, J., & Ritzer, P. (2012). Energy optimal control of an over actuated Robotic Electric Vehicle using enhanced control allocation approaches. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 322-327). Alcala de Henares, Spain: IVS IEEE.
- [Bre11c] Brembeck, J., Otter, M., & Zimmer, D. (2011). Nonlinear Observers based on the Functional Mockup Interface with Applications to Electric Vehicles. *Proceedings of 8th International Modelica Conference* (pp. 474-483). Dresden, Germany: Linköping Electronic Conference Proceedings.
- [Bre11b] Brembeck, J., & Wielgos, S. (2011). A real time capable battery model for electric mobility applications using optimal estimation methods. *Proceedings of 8th International Modelica Conference* (pp. 398-405). Dresden, Germany: Linköping Electronic Conference Proceedings.
- [Bre11] Brembeck, J., Ho, L. M., Schaub, A., Satzger, C., Tobolar, J., Bals, J., et al. (2011). ROMO - The Robotic Electric Vehicle. *22nd IAVSD International Symposium on Dynamics of Vehicle on Roads and Tracks.* Manchester Metropolitan University: IAVSD.

D.2 List of Publications as Co-Author

The following list includes all peer-reviewed and published articles on which the author of this thesis contributed as a co-author and that are connected to his doctoral thesis project. They are sorted by the date of publication:

- [Rit16b] Ritzer, P., Winter, C., & Brembeck, J. (2016). Experimental Validation of Geometric Path Following Control with Demand Supervision on an Over-Actuated Robotic Vehicle. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 539-545). Gothenburg, Sweden: IVS IEEE.
- [Win16] Winter, C., Ritzer, P., & Brembeck, J. (2016). Experimental Investigation of Online Path Planning for Electric Vehicles. *Proceedings of the IEEE International Conference on Intelligent Transportation Systems (ITSC)* (pp. 1403-1409). Rio de Janeiro, Brasil: IEEE.
- [Rit15] Ritzer, P., Winter, C., & Brembeck, J. (2015). Advanced Path Following Control of an Overactuated Robotic Vehicle. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 1120-1125). Seoul, South Korea: IVS IEEE.
- [Bue14] Bünte, T., Ho, L. M., Satzger, C., & Brembeck, J. (2014). Central Vehicle Dynamics Control of the Robotic Research Platform ROboMObil. *ATZelektronik worldwide, 9* (3), 58-64.
- [Fle13] Fleps-Dezasse, M., & Brembeck, J. (2013). Model based vertical dynamics estimation with Modelica and FMI. In T. Kawabe (Ed.), *Proceedings of the 7th IFAC Symposium on Advances in Automotive Control. 7*, pp. 341-346. Tokyo, Japan: Elsevier.
- [Sat13] Satzger, C., Brembeck, J., & Otter, M. (2013). Framework for the Evaluation of Wheel Torque Blending Algorithms. In T. Kawabe (Ed.), *Proceedings of the 7th IFAC Symposium on Advances in Automotive Control. 7*, pp. 347-352. Tokyo, Japan: Elsevier.
- [Bue11] Bünte, T., Brembeck, J., & Ho, L. M. (2011). Human Machine Interface Concept for Interactive Motion Control of a Highly Maneuverable Robotic Vehicle. *Proceedings of the IEEE Intelligent Vehicles Symposium* (pp. 1170-1175). Baden-Baden, Germany: IVS IEEE.

D.3 Articles and Reports

Here all articles and (internal) technical reports are listed that are related to this work:

- [Rit16] Ritzer, P., Panzirsch, M., & Brembeck, J. (2016). Robotic Motion - Interactive motion simulation of vehicle dynamics. (B. Schäfers-Maiwald, Ed.) *dSPACE Magazin, 01*, 52-57.
- [Bre13] Brembeck, J. (2013). *Method to extend models for system design to models for system operation.* Mid-term report ITEA2 - MODRIO, DLR e.V., Weßling.

- [Kre13] Krenn, R., Köppern, J., Bünte, T., Brembeck, J., Gibbesch, A., & Bals, J. (2013). Modellbasierte Regelungsansätze für überaktuierte planetare Rover und robotische Elektromobile. *at - Automatisierungstechnik , 61* (3), 183-194.
- [Sca11] Schaub, A., Brembeck, J., Burschka, D., & Hirzinger, G. (2011). Robotic Electric Vehicle with Camera-based Autonomy Approach. *ATZelektronik , 2* (2), 10-16.

D.4 Supervised Student Theses

This list contains all student theses supervised by the author:

- [Win13] Winter, C., Brembeck, J., & Kennel, R. (2013). *Online Energy Optimal Path Planner for Advanced Electric Vehicles.* Master's Thesis, Technische Universität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.
- [Rit13] Ritzer, P., Brembeck, J., & Kennel, R. (2013). *Model Based Vehicle Dynamics Control for Modern Vehicle Architectures.* Master's thesis, Technische Universität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.
- [Boe13] Böck, S., Brembeck, J., & Parzhuber, O. (2013). *Sensor-Analyse & -Modellierung für die zeitliche Synchronisierung der Sensorfusion im Forschungsfahrzeug ROboMObil.* Bachelor‘s thesis, Hochschule München.
- [Win13b] Winter, C., & Brembeck, J. (2013). *Quadratische Optimierungsprobleme in der energieoptimalen Bahnplanung.* Intership report, Technische Universität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.
- [Enn13] Ennifar, H., Brembeck, J., Otter, M., & Kennel, R. (2013). *Integration of a Real-Time Battery Model in a RCP-System and its Implementation on a Research Platform.* Bachelor's thesis, Technische Unvierstität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.
- [Win12] Winter, C., Brembeck, J., & Kennel, R. (2012). *Aktuelle Algorithmen & Methoden zur energieoptimalen Bahnplanung und Bewertung derer Einsetzbarkeit in Bezug auf das DLR-ROboMObil.* Research practice report, Technische Universität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.
- [Bau11] Baumgartner, D., Brembeck, J., Herzog, G., & Kennel, R. (2011). *Konzeption eines Drive-By-Wire Lenkungsprüfstands mit flexibler Mutlibody-Simulation im ROboMObil-Projekt.* Diploma thesis, Technische Universität München, Fachgebiet Energiewandlungstechnik, Munich, Germany.

- [Rit11] Ritzer, P., Brembeck, J., & Kennel, R. (2011). *Energieoptimale Bewegungssteuerung überaktuierter Elektrofahrzeuge.* Bachelor's thesis, Technische Universität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.
- [Rit11b] Ritzer, P., Brembeck, J., & Kennel, R. (2011). *Energie- und Leistungsmanagement im Elektrofahrzeug.* Research practice report, Technische Universität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.
- [Eng10] Engst, C., Brembeck, J., & Kennel, R. (2010). *Object-Oriented Modelling and Real-Time Simulation of an Electric Vehicle in Modelica.* Master's thesis, Technische Universität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.
- [Wie10] Wielgos, S., Brembeck, J., Otter, M., & Kennel, R. (2010). *Development of an Energy Management System for Electric Vehicles Design and System Simulation.* Master's thesis, Technische Universität München, Lehrstuhl für Elektrische Antriebssysteme und Leistungselektronik, Munich, Germany.