



Ostfalia
University of
Applied Sciences



ASIM Mitteilung AM140

Simulation technischer Systeme

Grundlagen und Methoden in Modellbildung und Simulation

Tagungsband zum

ASIM/GI Fachgruppentreffen STS/GMMS

Wolfenbüttel, 23. und 24. Februar 2012

Herausgeber:

Xiaobo Liu-Henke

Robert Buchta

Florian Quantmeyer

ASIM-Konferenz STS/GMMS 2012

Simulation technischer Systeme

***Grundlagen und Methoden in
Modellbildung und Simulation***

Herausgeber:

Xiaobo Liu-Henke

Robert Buchta

Florian Quantmeyer

Tagungsveranstalter:

**Ostfalia Hochschule für angewandte Wissenschaften,
Fakultät Maschinenbau**

Prof. Dr.-Ing. X. Liu-Henke

Dipl.-Ing. (FH) R. Buchta

F. Quantmeyer, M.Eng.

J. Pinkert

ASIM-Fachgruppe *Simulation technischer Systeme*

Dr.-Ing. H.-T. Mammen, Hella KGaA Hueck & Co.

**ASIM-Fachgruppe Grundlagen und Methoden in
Modellbildung und Simulation**

Prof. Dr.-Ing. T. Pawletta, Hochschule Wismar

unterstützt durch:



Volkswagen AG

ASIM / ARGESIM Publisher, Vienna

ASIM Mitteilung AM140

ISBN 978-3-901608-39-1

© ARGESIM Verlag, Vienna

Alle Rechte vorbehalten, auch das des Nachdruckes, der Wiedergabe (Photokopie, Mikrokopie), der Speicherung in Datenverarbeitungsanlagen und der Übersetzung, auszugsweise oder vollständig.

Die ASIM Mitteilung AM140, die die öffentlichen Vorträge der Tagung enthält, erscheint als nichtredigierter Manuskriptdruck. Die einzelnen Beiträge geben die auf persönlichen Erkenntnissen beruhenden Ansichten und Erfahrungen der jeweiligen Vortragenden bzw. Autoren wieder.

Vorwort

Als moderne Hochschule für Technik, Gesundheits-, Rechts-, Sozial- und Wirtschaftswissenschaften zählt die Ostfalia zu den größten in Niedersachsen. Forschung und Lehre haben an der Ostfalia Hochschule für angewandte Wissenschaften einen hohen Stellenwert und bilden eine untrennbare Einheit. Forschung und Entwicklung mehrten anwendungsorientierte Erkenntnisse, stärken die Attraktivität der Studiengänge, erschließen praxisorientierte Themen für Diplom-, Bachelor- und Masterarbeiten sowie Praxisstellen und stärken das Renommee der Hochschule.

Die diesjährige ASIM/GI-Tagung der Fachgruppen STS und GMMS findet an der Ostfalia statt. Die Ausrichtung der Tagung stellt eine Erweiterung der Forschungsstrukturen der Ostfalia als Zentrum der angewandten Forschung dar und stärkt die Verzahnung von Theorie und Praxis.

Der Teilnehmerkreis der diesjährigen ASIM-Tagung besteht aus Forschern und Entwicklern aus der Industrie, insbesondere aus der Automobil- und -Zuliefererindustrie, sowie Professoren und Wissenschaftlern aus Hochschulen und Universitäten.

Der vorliegende Tagungsband enthält viele interessante Beiträge, für deren Form und Inhalt die Autoren selbst verantwortlich sind. Die Themen reichen von den Grundlagen der Modellbildung und Simulation bis hin zur Anwendung der Methodik in der modellbasierten Entwicklung und Absicherung mittels Model-in-the-Loop-, Software-in-the-Loop- und Hardware-in-the-Loop-Simulation. Durch die regional stark vertretene Automobil- und Zuliefererindustrie ist die Tagung besonders vom Einsatz der Simulationstechnologie im Automobil geprägt. Darüber hinaus gibt es eine Session die sich mit der Verkehrsmodellierung und –simulation befasst.

Die Tagung wird begleitet von insgesamt 10 Firmen, die die verschiedenen F&E-Bereiche mit einer großen Bandbreite repräsentieren – von Softwareentwicklung für die theoretische Modellbildung, Offline-Simulation bis zu HiL-Anwendungen im Echtzeitbereich.

Xiaobo Liu-Henke, Florian Quantmeyer, Robert Buchta

Wolfenbüttel im Februar 2012

Plenarvortrag

<i>M. Brandt</i>	Integrationstest vernetzter elektronischer Steuergerätefunktionen mit HiL-Simulation bei Volkswagen (unveröffentlicht)	
<i>H. P. Bensler</i>	Berechnungsverfahren in der Volkswagen Konzernforschung (unveröffentlicht)	
<i>R. Dölling</i>	Methode zur automatischen Erstellung von elektrischen Verhaltensmodellen aus Simulationsdaten	1

Modellierungssprachen und Standards

<i>A. Mehlhase, T. Beckmann</i>	Python-Framework zur Simulation von Strukturdynamik-Modellen	25
<i>F. Dunke, S. Nickel</i>	Simulation und Algorithmenanalyse: Ein generisches Vorgehensmodell für Online-Optimierungsprobleme mit Lookahead	37
<i>V. Svjatnyj, V. Kushnarenko</i>	Virtuelle Simulationsmodelle und ein Devirtualisierungsvorgang der Erstellung von parallelen Simulatoren für dynamische Netzobjekte mit verteilten Parametern	49
<i>C. Clauß, M. Arnold, T. Schierz, J. Bastian</i>	Master zur Simulatorkopplung via FMI	57

Simulation und Test in der Automobilindustrie

<i>M. Nägeli, D. Lichtenthäler</i>	Eine virtuelle Simulationsumgebung als Prüfplatz für Hochvoltbatterie-Steuergeräte	71
<i>R. Rasche, D. Neumerkel</i>	Decoupling Test Cases from Real and Virtual Test Systems with ASAM HIL API	83
<i>S. Schneider, D. Brechter, A. Janßen, H. Mauch, C. Bohn</i>	Der Einfluss semi-aktiver Dämpfer auf die Betriebslasten eines PKW	97
<i>R. Mustafa, F. Küçükay</i>	Effizientes Thermomanagement – Einsatz der Gesamtfahrzeugsimulation zur Quantifizierung von Kraftstoffverbrauchseinsparungen im PKW	109
<i>A. Holm, Y. Yu, V. Dorsch</i>	Simulation eines Fahrzeugs mit geregelter Hinterradlenkung	121
<i>C. Kaiser, S. Försterling, W. Tegethoff, J. Köhler</i>	Untersuchung von Regelstrategien für die Omnibusklimatisierung mit Hilfe einer Gesamtfahrzeugsimulation	131
<i>R. Buchta, X. Liu-Henke</i>	Regelung der Quer- und Gierbewegung eines Elektrofahrzeugs mittels radnahen Antrieben	143

Modellierung physikalischer Systeme

<i>P. Tusche, T. Zschunke</i>	Dynamische Simulationen in der Entwurfsphase mit Parameteranpassung durch implementierte Teillastrechnungen	155
<i>G. E. Stebner, C. Hartwig</i>	Softwaretool zur Standardisierung des Modellierungsprozesses permanenterregter Synchronmaschinen	167
<i>A. Makarov</i>	Ein schnelles Verfahren zur digitalen Simulation von Regelungssystemen hoher Ordnung	177
<i>Y. Ding</i>	Simulation und Regelung eines auf der Resonanzschwingungstechnologie basierenden Mischungsreaktors	181

Simulation in der Elektronikentwicklung

<i>F. Pramme, G. Bikker, M. Loeffler</i>	Domänenspezifische Konfiguration von Gesamtfahrzeugsimulationen	189
<i>K. Einwich, T. Arndt</i>	SystemC-AMS basierte Modellierung, Simulation und HiL Echtzeitsimulation für Anwendungen der Automobilelektronik	203
<i>F. Quantmeyer, X. Liu-Henke, W. Diehl, S. Bode</i>	Parametrierung von Batteriesimulationsmodellen mithilfe eines geeigneten Prüfstands	209
<i>M. Fakih, K. Grüttner</i>	Virtual Platform in the Loop Simulation for Accurate Timing Analysis of Embedded Software on Multicore Platforms	221

Systemsimulation im Automobilbereich

<i>E. Hessel, J. Haase</i>	Ansätze zur Systemsimulation im Automobilbau	233
<i>H.-T. Mammen, U. Buschmann</i>	Modellbeispiele mechatronischer Systeme	243
<i>A. Schön</i>	Erstellung einer firmenweiten Modellbibliothek – Potentiale, Anforderungen und Herausforderungen	249
<i>S. Petkun, X. Zhao</i>	BroSAnT – an example of the mechatronical way of thinking	255

Verkehrsmodellierung und –simulation

<i>H. F. Wedde, S. Senge</i>	Simulative Erprobung schwarmbasierter Routingalgorithmen im Straßenverkehr	267
<i>J. Lüßmann, C. Dittrich</i>	Netzweite Bewertung kooperativer Verkehrssysteme mittels mikroskopischer Verkehrsflusssimulation am Beispiel des eCoMove Projektes	269

		Seite
<i>O. Ullrich, D. Lückcrath, S. Franz, E. Speckenmeyer</i>	Simulation and optimization of Cologne's tram schedule	279
<i>P. Kuckertz, O. Ullrich, H. Randerath</i>	A simulation based approach on robust airline job pairing	291
Grundlagen und Methoden in Modellbildung und Simulation		
<i>S. Seele, T. Dettmar, R. Herpers, C. Bauckhage, P. Becker</i>	Cognitive Aspects of Traffic Simulations in Virtual Environments	301
<i>I. Bausch-Gall</i>	Standardisierte effiziente Speicherung von Simulationsergebnissen	305
<i>D. Frechen</i>	Simulation 2.0: Simulationsbaukasten und Team-Modellierung	307
<i>E. Gromakov, M. Sotnikova</i>	Training method of oil and gas industry operators on the example of the automaton simulation usage	319
Simulation bei der Systementwicklung		
<i>O. Rösch, M. F. Zäh</i>	Identifikation und Simulation des Nachgiebigkeitsverhaltens eines Fräsroboters	323
<i>P. Sacher, M. Rambke</i>	Beitrag zur Rückfederungsreduzierung hochfester Strukturblechteile	337
<i>J. Dörner, J. Wortberg</i>	Ein kalibrierbares Modell zur Beschreibung des Spritzgießprozesses elektrisch und wärmeleitfähiger Thermoplaste	349
<i>M. Bräunlich</i>	Simulation hydraulischer Bremskraftherzeuger für Schienenfahrzeuge	361

“Methode zur automatischen Erstellung von elektrischen Verhaltensmodellen aus Simulationsdaten,,

Dr.-Ing. Rolando Dölling Robert Bosch GmbH

Abstract

Projekt wurde auf Saber Basis entwickelt. Automatische datenbasierte Verhaltensmodellierung konservativer und nicht-konservativer Systeme innerhalb der simulativen Systemverifikation

Stichwörter

Pin kompatible elektrische Verhaltensmodelle, datenbasierte Modellierung, support vector machines, dynamik Extraktion, Filter, nicht lineare Systeme, Verkürzung der Simulationszeit, Schaltungsbeispiele aus Automotive

Basierend auf den Arbeiten von

Dr.-Ing. Philipp Senger, Fraunhofer-Institute for Algorithms and Scientific Computing

Dr.-Ing. Holger Mielenz, Robert Bosch GmbH

Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.

**BOSCH**

1

mitronic

Was ist mitronic?

- Eine datenbasierte Methodik mit Anwendung innerhalb der simulativen Schaltungsverifikation auf Blockebene zur automatischen elektrischen Verhaltensmodellierung

Wo findet die mitronic Anwendung statt?

- Anwendbar auf :
 - Konservative (*elektrisch: Kirchhoffgesetze können angewendet werden!*) und
 - nicht konservative Systeme
- Automatische Integration auf der selben Simulationsumgebung oder durch ein äquivalentes elektrisches Verhaltensmodell ermöglicht den einfachen Transfer von der Domän_1 auf die Domän_2.

Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.

**BOSCH**

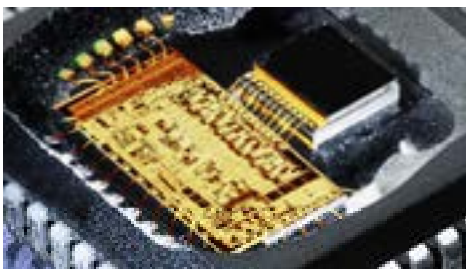
2

Agenda

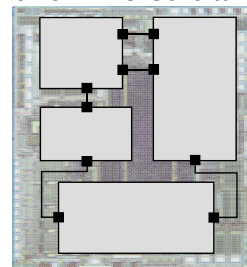
- Motivation
- Datenbasierte Methode zur Verhaltensmodellierung
- Methodik zur Automatisierung
- Anwendungsstudien unter Saber Simulationen
- Zusammenfassung



Einführung



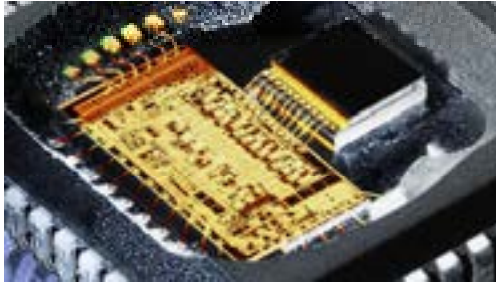
Simulationsmodell
einer A/MS-Schaltung



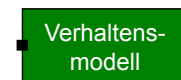
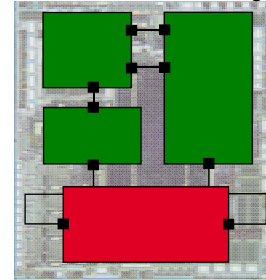
- Sicherheitskritischer Einsatz erfordert erhöhten Verifikationsaufwand bereits beim ASIC-Entwurf
- Die transiente Simulation auf Blockebene ist ein sehr wichtiges Werkzeug zur Überprüfung der richtigen Funktionen
- Analog/Mixed-Signal-Blöcke (A/MS) können aufgrund hoher Frequenzen und komplexer Strukturen lange Simulationszeiten besitzen



Motivation



Simulationsmodell
einer A/MS-Schaltung



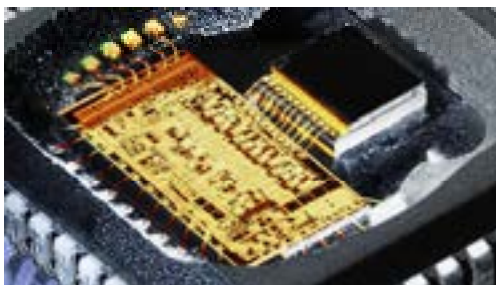
5

Automotive Electronics

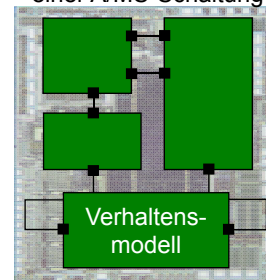
AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.

**BOSCH**

Motivation



Simulationsmodell
einer A/MS-Schaltung



- Beschleunigung der Simulation mittels schnell berechenbarer Verhaltensmodelle mit einfacher interner Struktur
- Modelle approximieren das elektrische Verhalten
- Generierung ist ein manueller Schritt verbunden mit hohem Grad an Schaltungsexpertise seitens des Verifikationsingenieurs

6

Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.

**BOSCH**

Stand der Technik bei der Verhaltensmodellierung

Strukturelle Verhaltensmodellierung

Modellierungsansätze:

[Doboli et al., 2003], [Lüdecke, 2003]

Symbolische Verhaltensmodellierung

Modellierungsansätze:

[Wichmann, 2003], [Halfmann et al., 2007]

Automatisierungsansätze:

[Platte et al., 2007], [Hedrich et al., 2007]

Parametrische Verhaltensmodellierung

Modellierungsansätze:

[Gines et al., 2002], [Jancke et al., 2007]

Automatisierungsansätze:

[Cadence DCM, 2005]

Datenbasierte Verhaltensmodellierung

- nicht konservative Modellierungsansätze:

[Litovskiet al., 2005], [Ceperic, et al. 2009],

[Dölling, 2002-2005]²

[Mielenz, 2008]¹

Automatisierungsansätze:

- Neuer Modellierungsansatz zur konservativen (elektrischen) Modellierung
- Erste Automatisierungsansätze:
- mitronic (Saber user Meeting)

[Dölling, 2009]²

[Senger, 2011]¹

[Dölling, 2011]²

2) Bosch intern, 1) Dissertation

Automotive Electronics

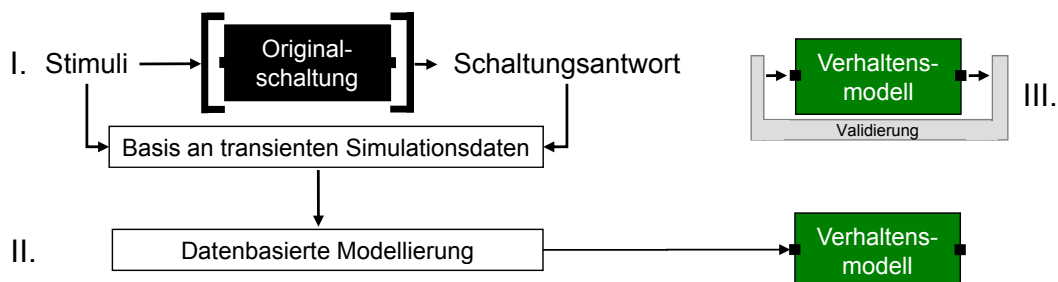
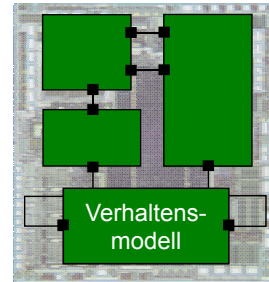
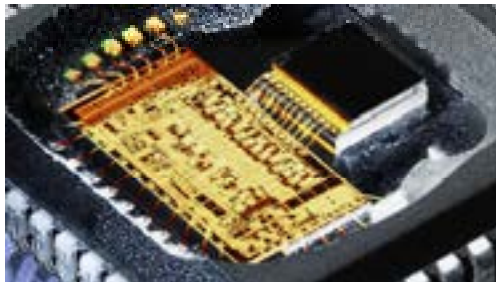
AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

7

Ansatz der datenbasierten Verhaltensmodellierung



Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

8

Ziele

- Automatisierte Erstellung der Verhaltensmodelle unabhängig von der Schaltungstopologie und Komplexität
- Speedup des Verhaltensmodells soll größer Original Schaltung sein ($\rightarrow >10x$)
- Genauigkeit $> 90\%$

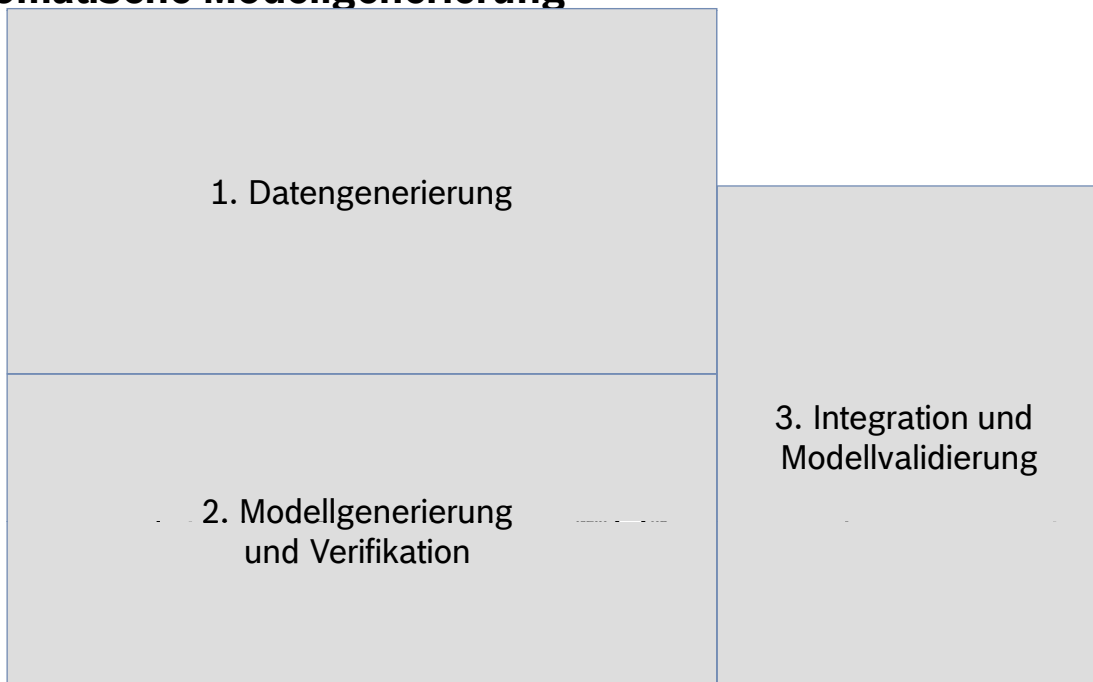


Anforderungen an das Verhaltensmodel

- Automatisierte Erstellung der Verhaltensmodelle unabhängig von der Schaltungstopologie und Komplexität
- Elektrische Eigenschaften des Originalsystems müssen pinkompatibel beschrieben werden
- Abbildung der (nicht-) linearen und dynamischen Übertragungseigenschaften
- Integration von Parametern in die Modellbildung (z.B. Temperatur oder Druck)
- Erweiterbarkeit und Einhaltung von Komplexitätsgrenzen bei Modellen mit höherer Anzahl an Ein- und Ausgängen
- Einstellbares Fehlermaß / abhängig von Speedup



Automatische Modellgenerierung



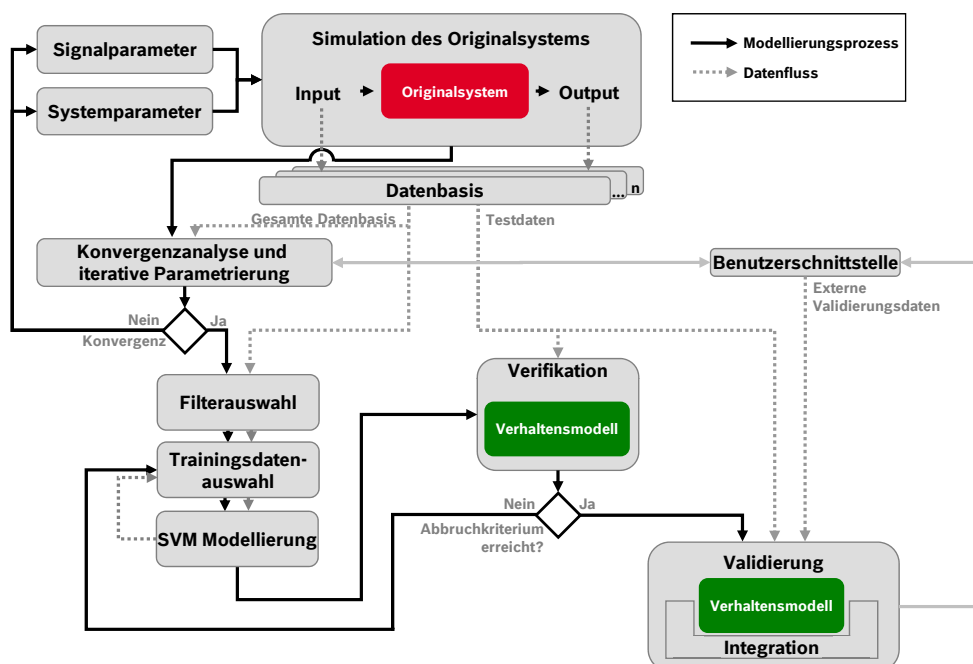
Automotive Electronics

11

ASIM 2012-02-23 © Robert Bosch GmbH 2011. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.


BOSCH

Automatische Modellgenerierung



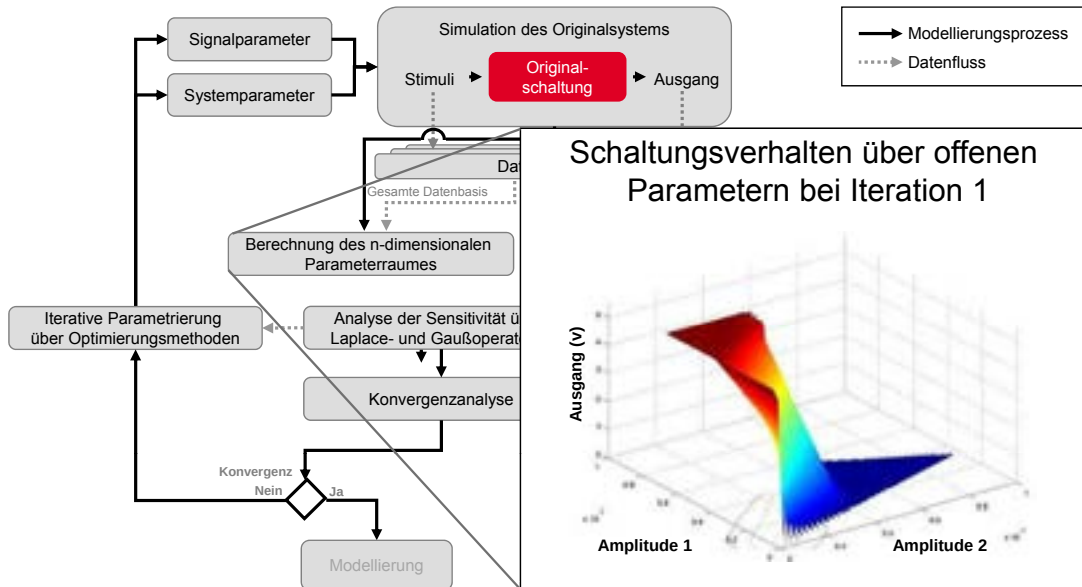
Automotive Electronics

12

ASIM 2012-02-23 © Robert Bosch GmbH 2011. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.


BOSCH

Datengenerierung



13

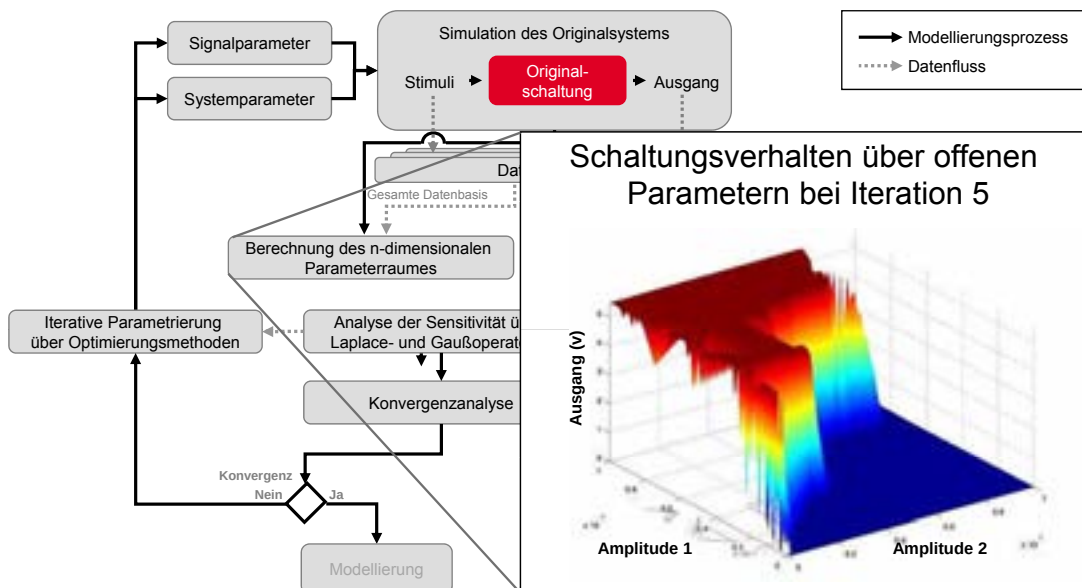
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

Datengenerierung



14

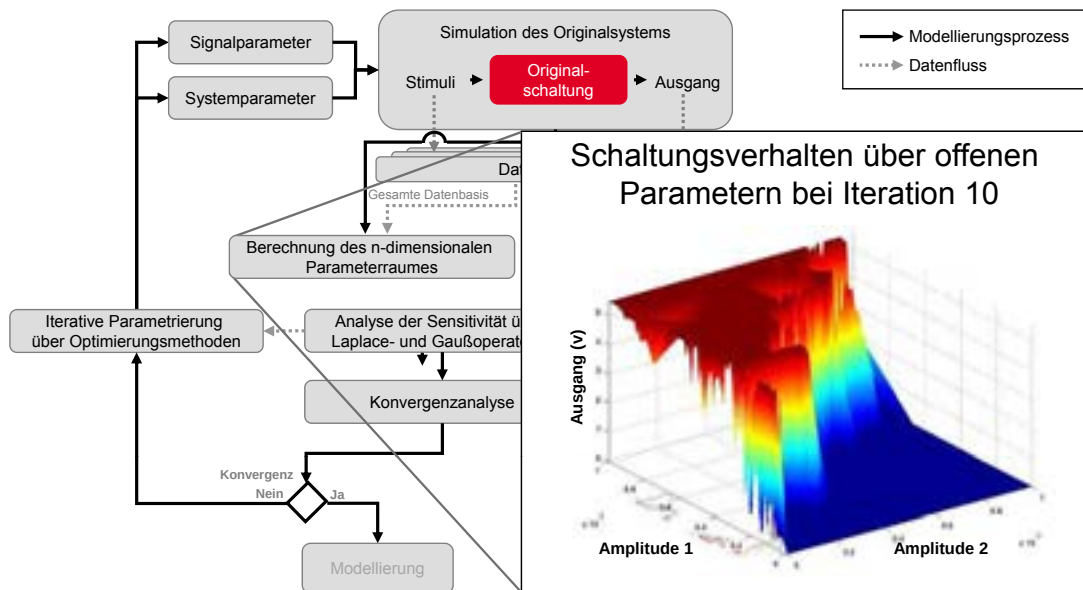
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

Datengenerierung



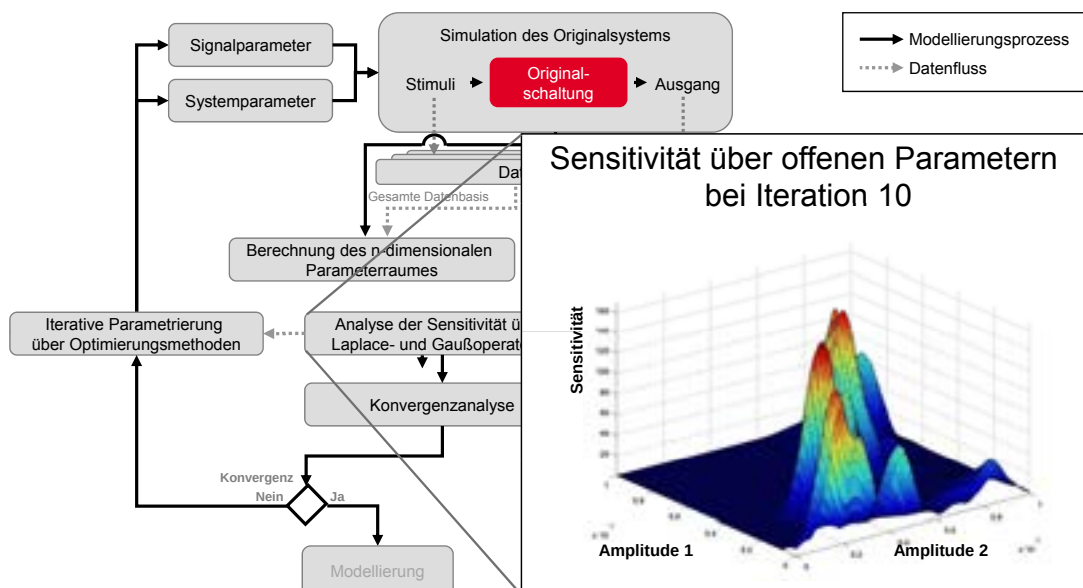
Automotive Electronics

15

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.


BOSCH

Datengenerierung



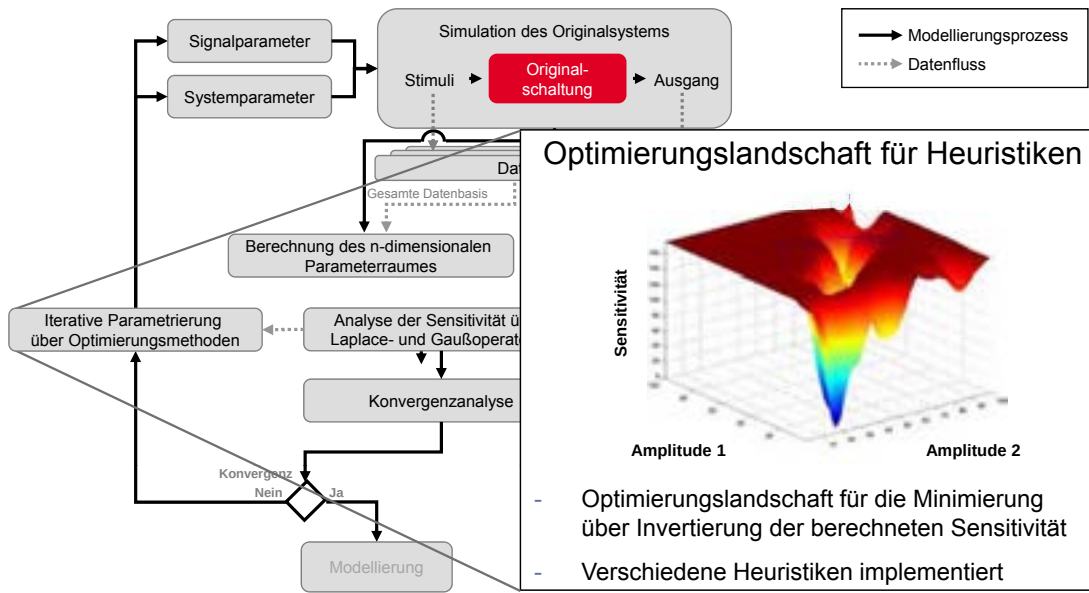
Automotive Electronics

16

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.


BOSCH

Datengenerierung



17

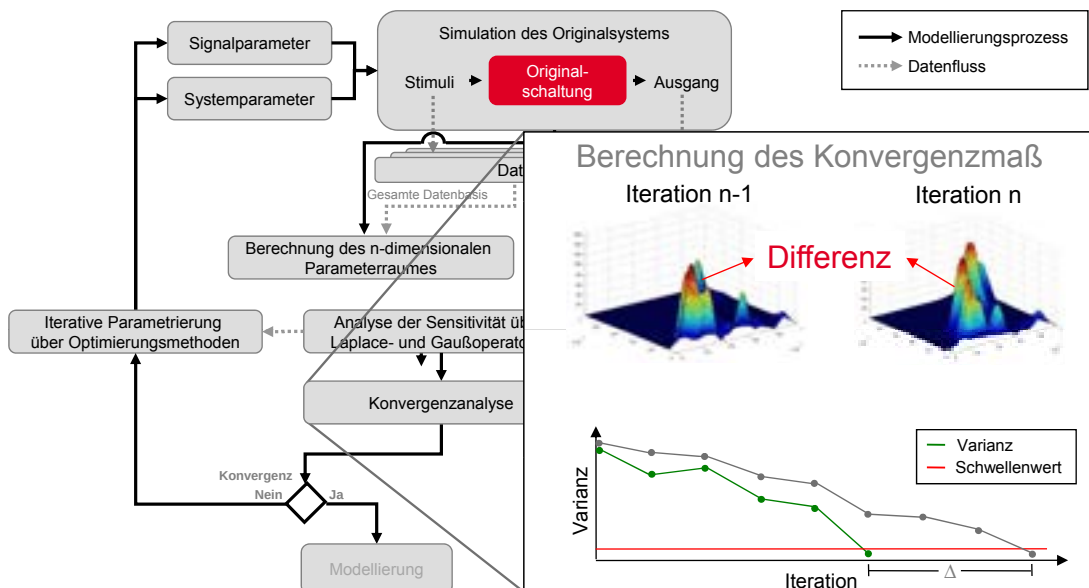
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

Datengenerierung



18

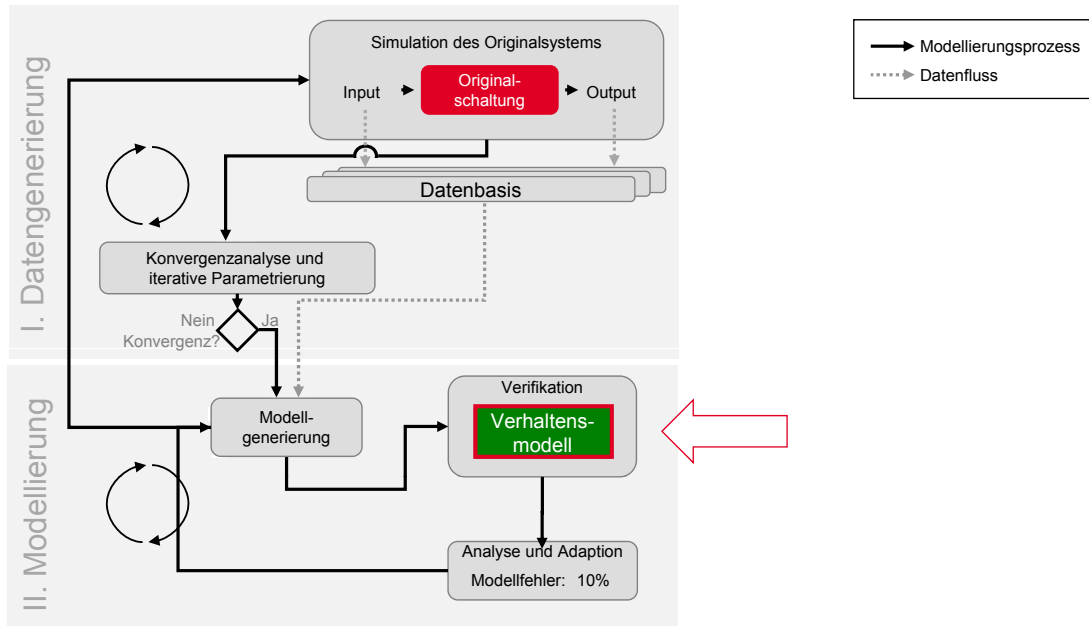
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

Modellierung



19

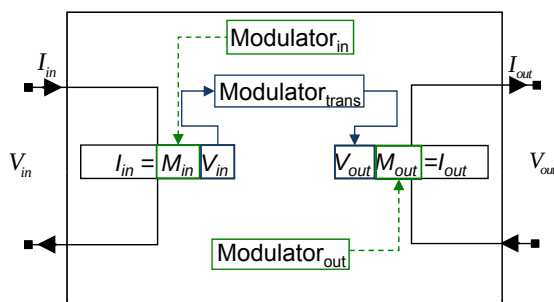
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

Funktionale Modellarchitektur



- Lokale und Transfer-Modulatoren bilden das elektrische Verhalten ab
- Erweiterung auf Schaltungen mit mehreren Ein- bzw. Ausgängen über weitere Modulatoren möglich

20

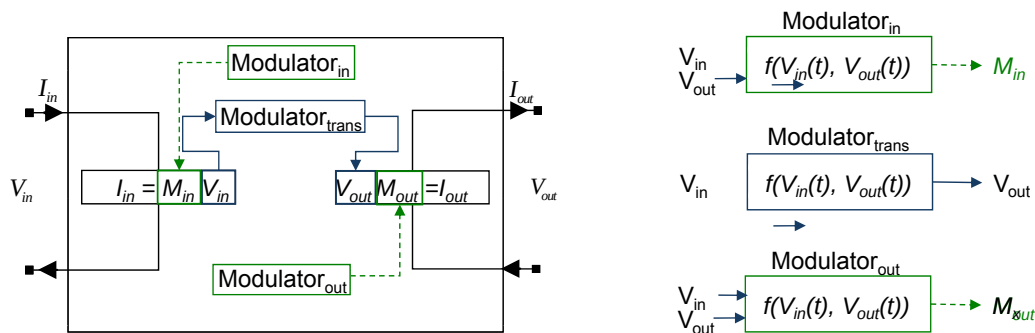
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

Funktionale Modellarchitektur



- Modulatoren sind Funktionen der elektrischen Größen in Abhängigkeit der Zeit t
- Funktionale Zusammenhänge können dynamisch, (nicht-) linear und parameterabhängig sein

21

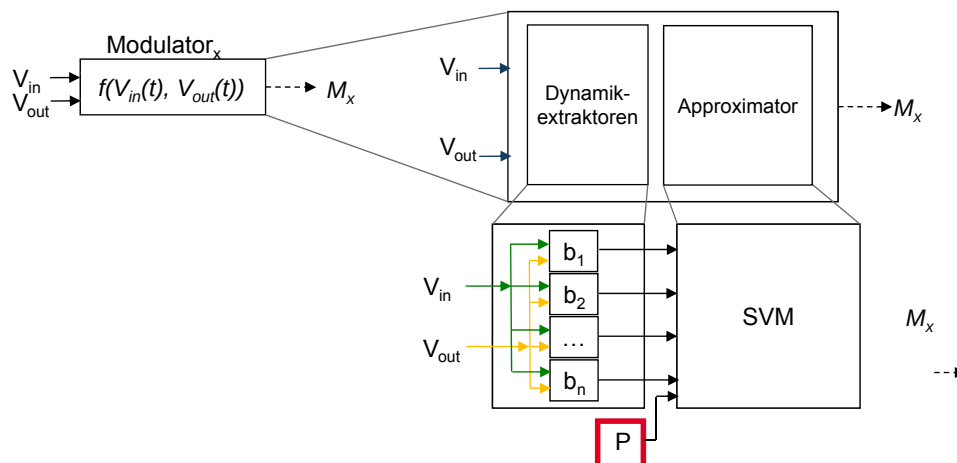
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

Intrinsische Modellarchitektur



- Integration von freien **Parametern P** in den Modellierungsprozess über zusätzliche Eingänge für die Regression

22

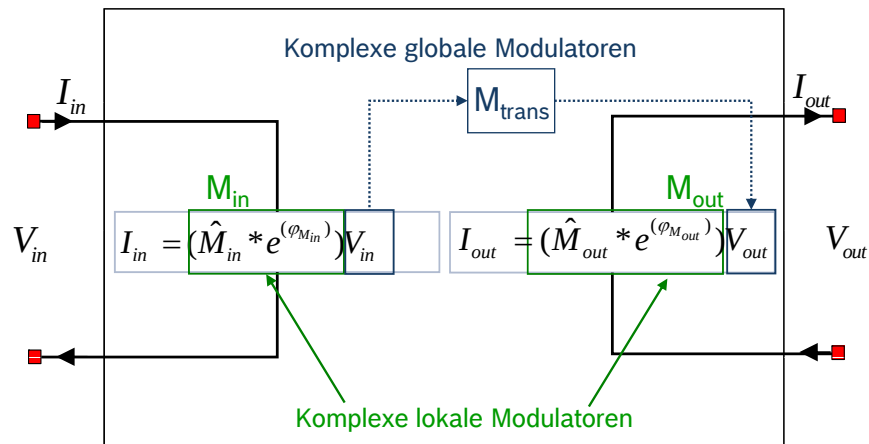
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

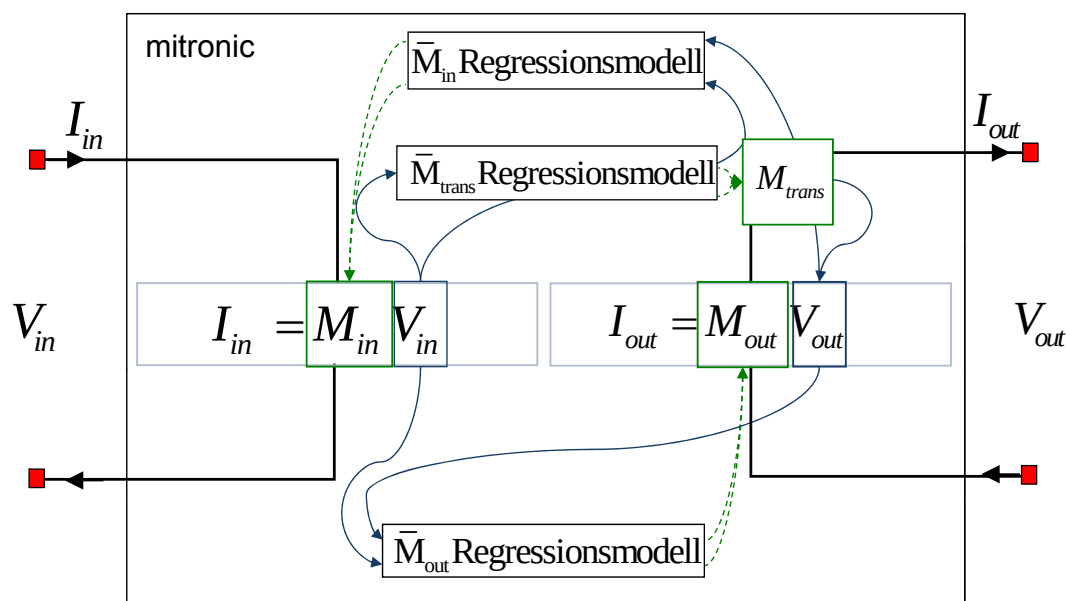
Modellarchitektur – Modellierungsansatz



- Lokale Approximation der elektrischen Beziehungen mit Hilfe der komplexen Modulatoren M_{in} und M_{out}
- Globale Approximation der Abhängigkeiten zwischen den Pins mit Hilfe des M_{trans} Modulators



Modellarchitektur – Modellierungsansatz (II)



- Modulatorabhängigkeiten bilden Abhängigkeiten der elektrischen Größen ab



Modellarchitektur – Modellierungsansatz (III)

$$I = (\hat{M} * e^{(\varphi_M)})V$$

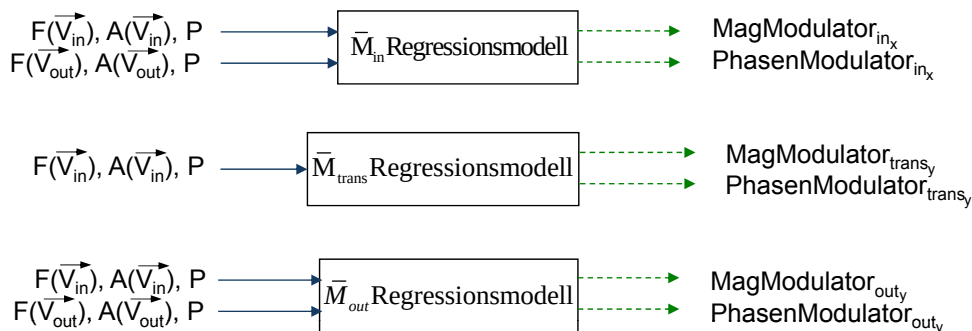
Direkte Integration der realen Anteils
als Amplitudenmodulation in der
Gleichungssektion des Analogsimulators

Direkte Integration der imaginären Anteils
als Phasenmodulation in der
Gleichungssektion des Analogsimulators
über entsprechende Allpassstrukturen

- Komplexe Modulatoren können direkt in der Netzliste des Modells mit dem Analogsimulator gekoppelt werden



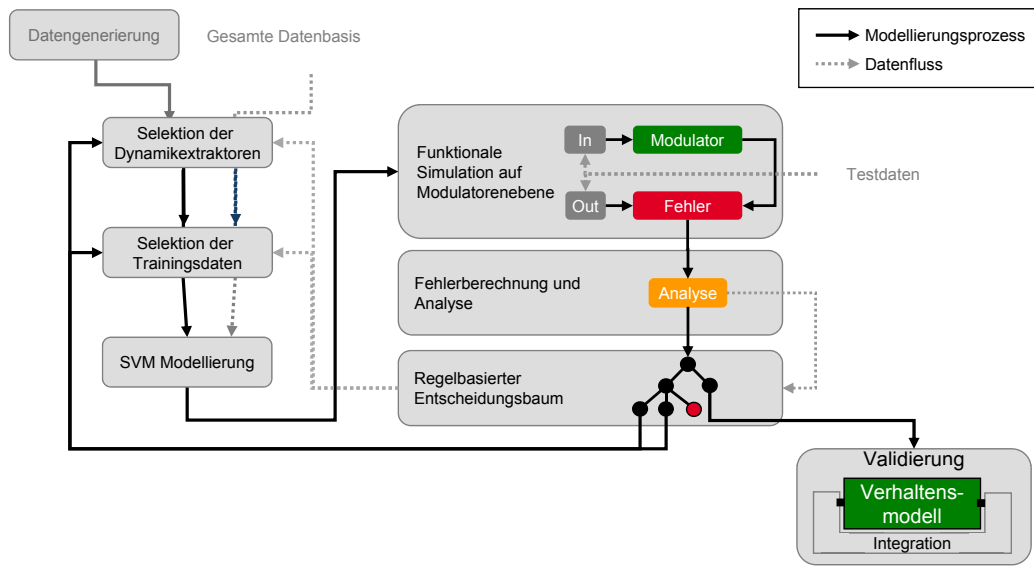
Modellarchitektur – Abhängigkeiten der Submodelle



- Abhängigkeiten für den x-ten Eingang und den y-ten Ausgang
- Integration der Frequenz-, Amplituden und Lastabhängigkeit über weitere Eingabegrößen in die Regression
- Lasttyp ist definiert durch die Schaltungsspezifikation



Modellgenerierung



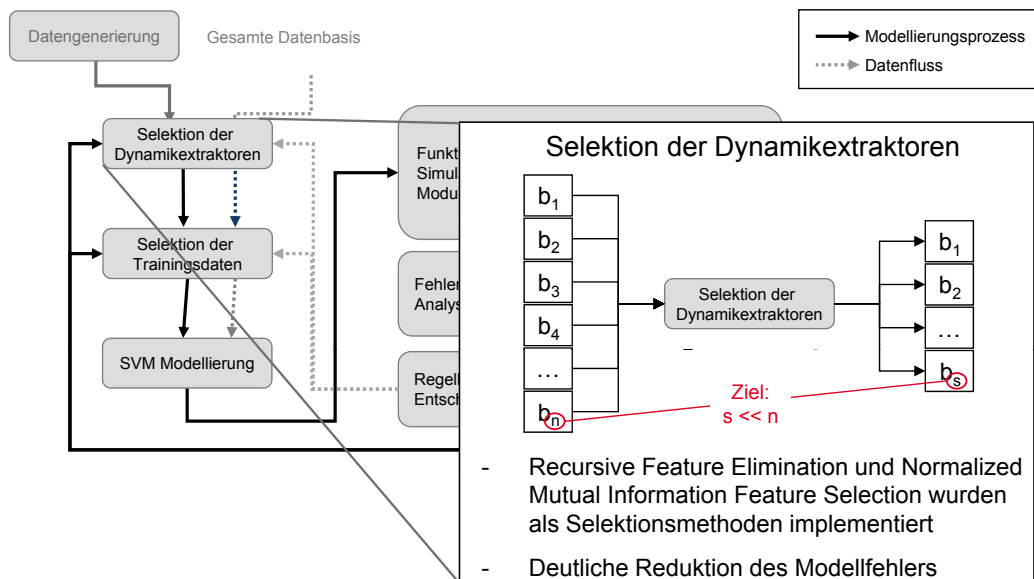
27

Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.


BOSCH

Modellgenerierung



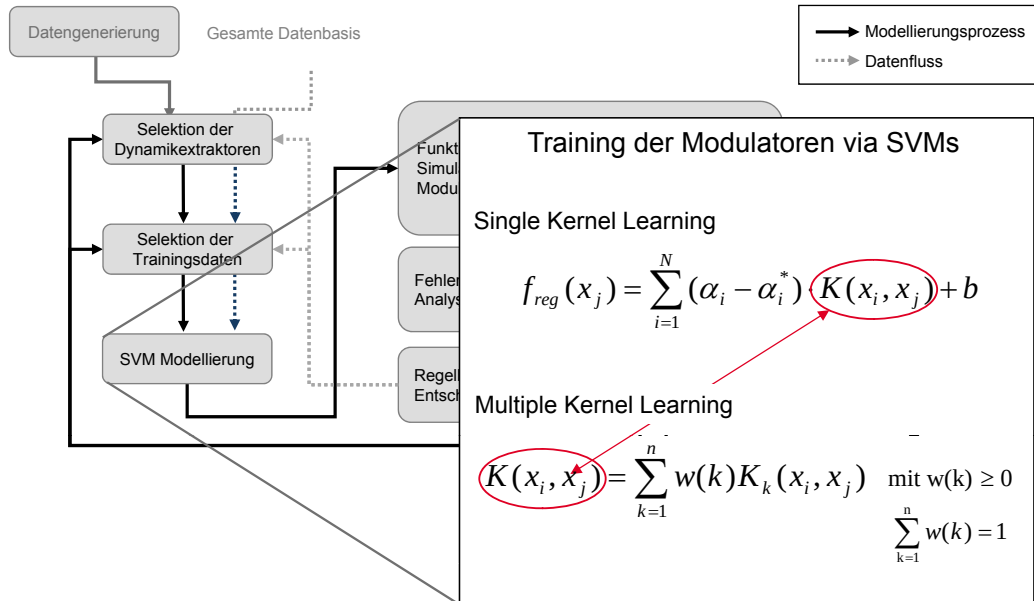
28

Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.


BOSCH

Modellgenerierung



29

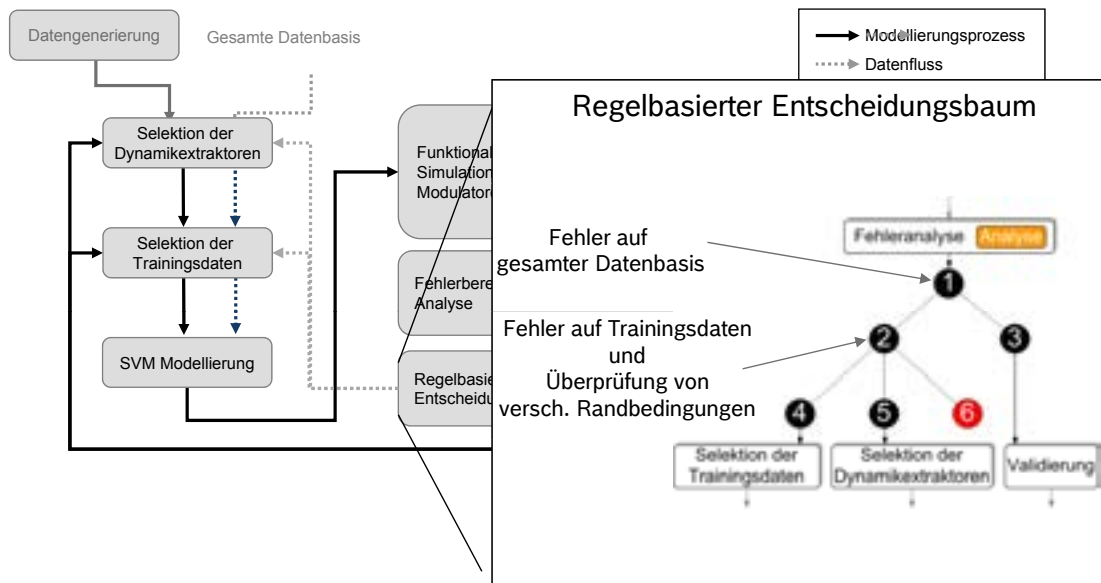
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

Modellgenerierung



30

Automotive Electronics

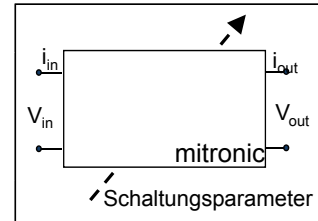
AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

Modellarchitektur – Eigenschaften des resultierenden mitronic Modells

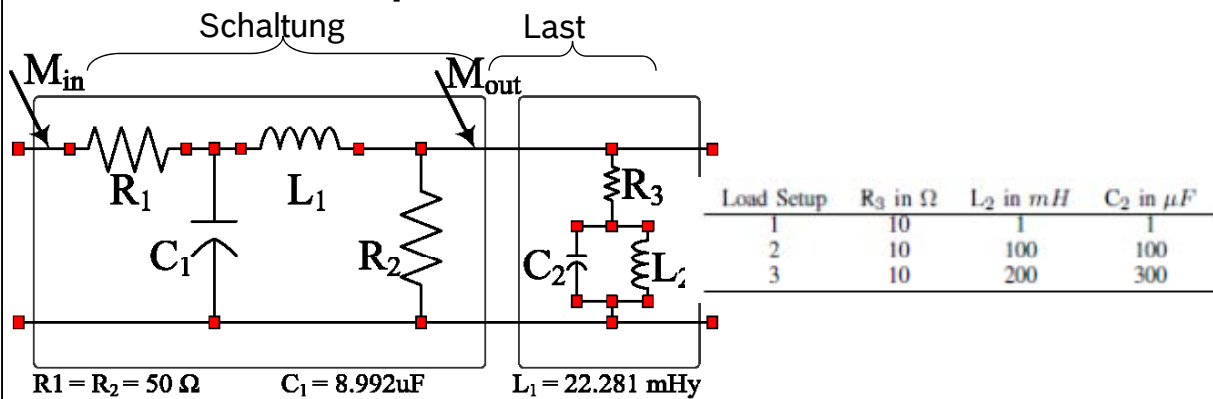
- Inkompatibles *Black Box* Modell
- Elektrische Eigenschaften des Originalsystems werden über das Verhalten an den Ein- und Ausgängen beschrieben
- Automatische Abbildung der Frequenz-, Amplituden- und Lastabhängigkeit
- Integration von Parametern in die Modellbildung (z.B. Temperatur oder Last)
- Lineares Wachstum der Anzahl der Modelle mit der Anzahl der Ein- und Ausgänge
- Weitestgehend automatisierte Erstellung



Anwendungsstudien mit dem Simulator Saber

- Butterworth Tiefpass-Filter
- Ladungspumpe („Charge Pump“)
- Stromquelle
- CAN (Analog/Digital Treiberschaltung)

Butterworth Tiefpass-Filter (1/3)



- Grenzfrequenzen bei 500 bzw. 1500Hz
- Eingangssignal als Chirpsignal mit kontinuierlich ansteigender Frequenz zwischen 1-2500Hz
- Drei verschiedene Lasttypen erfordern mehrere transiente Simulationen zur umfassenden Trainingsdatengewinnung

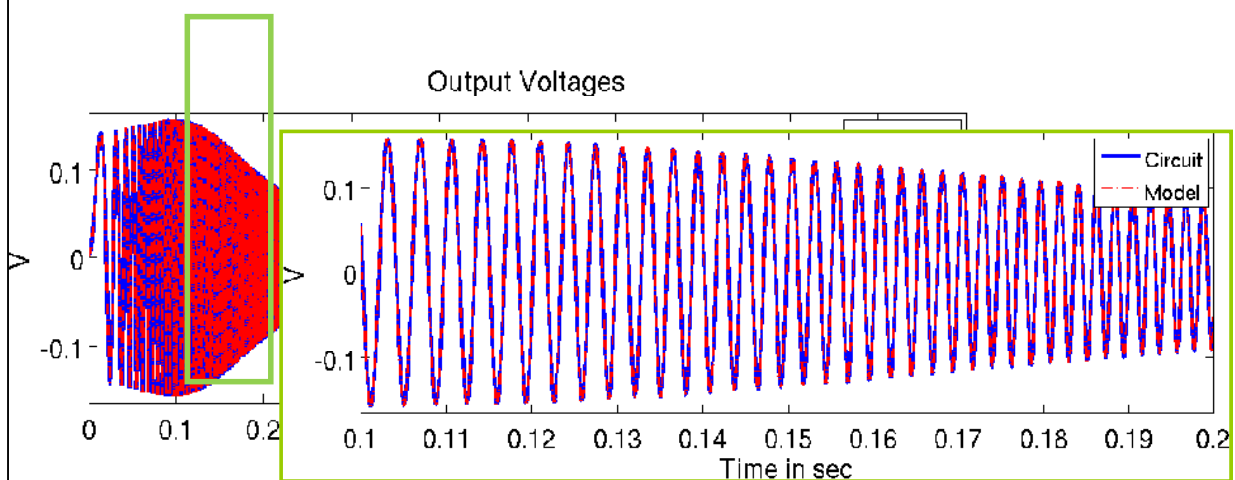
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.


BOSCH

33

Butterworth Tiefpass-Filter (2/3)



- Hohe Übereinstimmung der Spannungen am Schaltungsausgang bei Ausgangslast Setup 1
- Genauigkeit > 95%

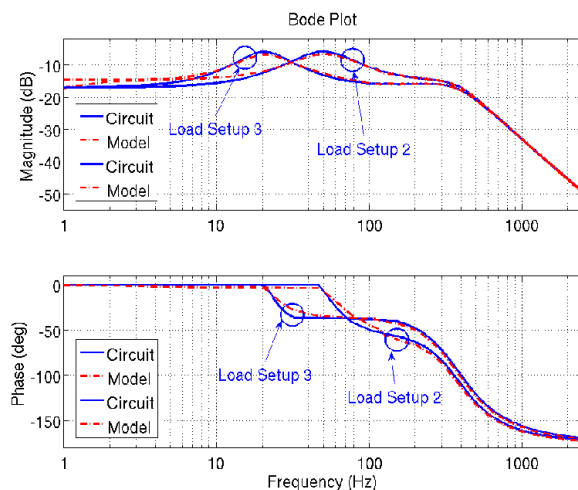
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.


BOSCH

34

Butterworth Tiefpass-Filter (3/3)



- Hohe Übereinstimmung der einzelnen elektrischen Beziehungen zwischen Ein- und Ausgangsspannungen bei Ausgangslast Setup 2 und 3.
- Weitere Modellausgänge sind von ähnlicher Güte im Vergleich zur Originalschaltung

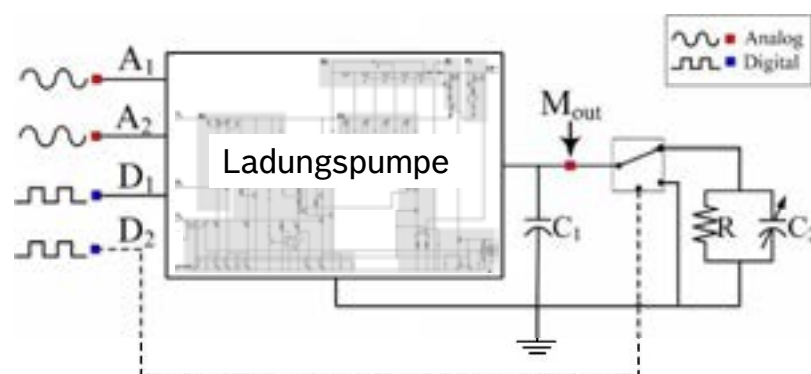
35

Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.


BOSCH

Testbench und Lastsetup der Ladungspumpe (1/2)



Parameter	(A)nalog 1	(A)nalog 2	(D)igital 1	(D)igital 2
Amplitude	5,7-6V	9-12V	0,1	0,1
Frequenz	-	-	3MHz	20KHz

	R	C ₁	C ₂
Lastsetup1	20kΩ	10pF	10pF
Lastsetup 2	20kΩ	10pF	250pF
Lastsetup 3	20kΩ	10pF	500pF

Parameter P

36

Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.

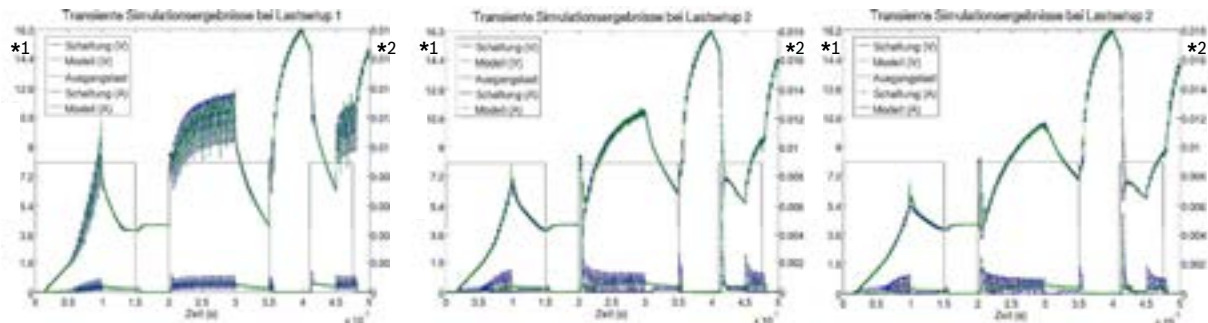

BOSCH

Ladungspumpe: Ergebnisse bei verschiedenen Lastsetups (2/2)

Lastsetup 1

Lastsetup 2

Lastsetup 3



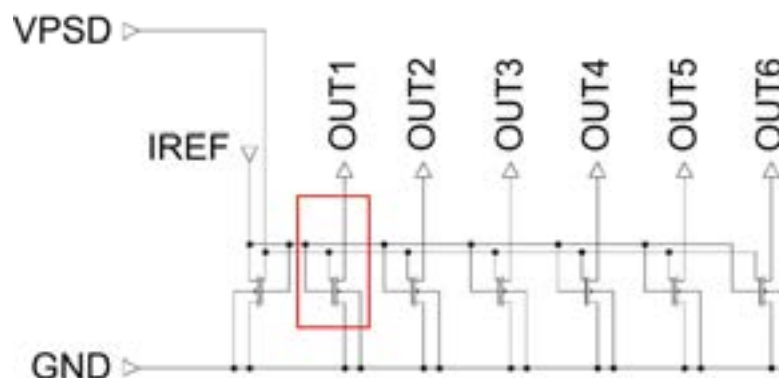
Lastsetup	Ausgangsspannung _{RMSD}	Ausgangsstrom _{RMSD}	Speed-up
1	3,8%	9,4%	37x
2	2,6%	10,4%	31x
3	3,1%	10,7%	34x

*1 Ausgangsspannung (V)

*2 Ausgangsstrom (A)



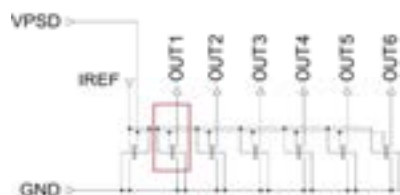
Modellierung Stromquelle



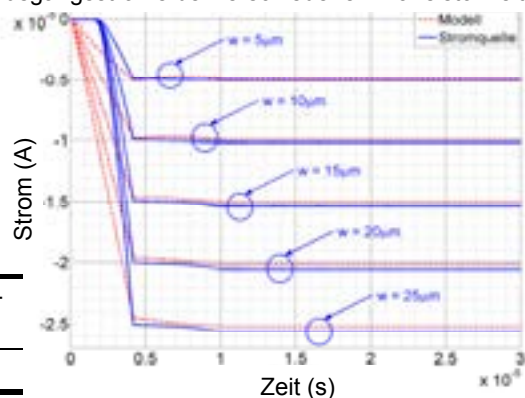
- Stromquelle besteht aus 7 Transistoren
- Integration des Parameters der Transistorweite an Ausgang OUT1 in das Modellierungsverfahren



Modellierung Stromquelle



Ausgangsströme bei verschiedenen Transistorweiten

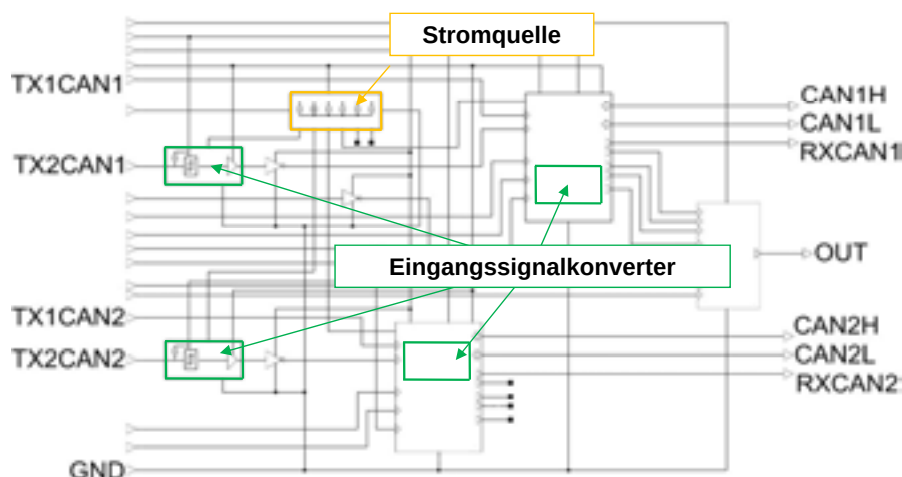


Parameter	VPSD	IREF	Transistor- weite
Amplitude	-1-7V	-30-(-10) μ A	5-25 μ m

Fehler Ausgangsspannung* (%)	Fehler Ausgangsstrom* (%)	Speed-Up*
0,5	1	1x

*Gemittelte Angaben über verschiedene Testszenarien bei einer Transistorweite von 5 μ m

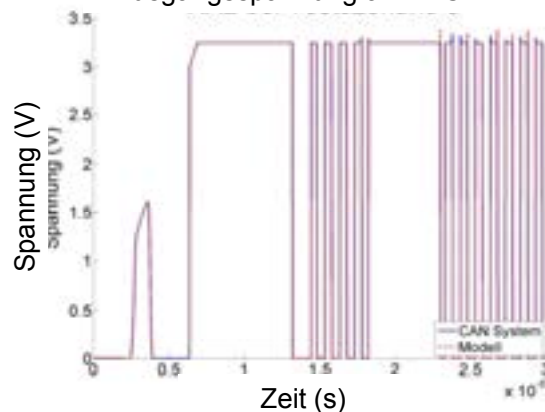
CAN: Integration einzelne Verhaltensmodelle (1/4)



- Gesamtblock besteht aus mehr als 600 aktiven und passiven Bauelemente
- Bewertung der Güte anhand der Ausgangsspannungen des Gesamtblocks

CAN: Ergebnisse der Verhaltensmodellintegration (2/4)

Ausgangsspannung an RXCAN1



- Genauigkeit (85-95%) am Schaltungsausgänge ermöglicht den Einsatz der Modelle innerhalb von Verifikationsaufgaben
- Modelle konnten erfolgreich während der Überprüfung der Verdrahtung eingesetzt werden, mit einem Geschwindigkeitsgewinn von ca. 12x

41

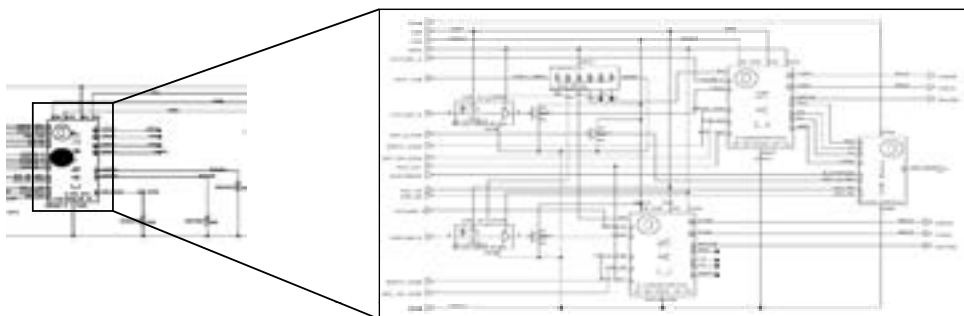
Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

CAN: Modellierung CAN-Gesamt (3/3)



- 18 Eingänge/ 7 Ausgänge / **>600 Bauelemente**
- Komplexe Übertragungsfunktionen
- 85-90% Genauigkeit
- Speed-Up 12x-15x

42

Automotive Electronics

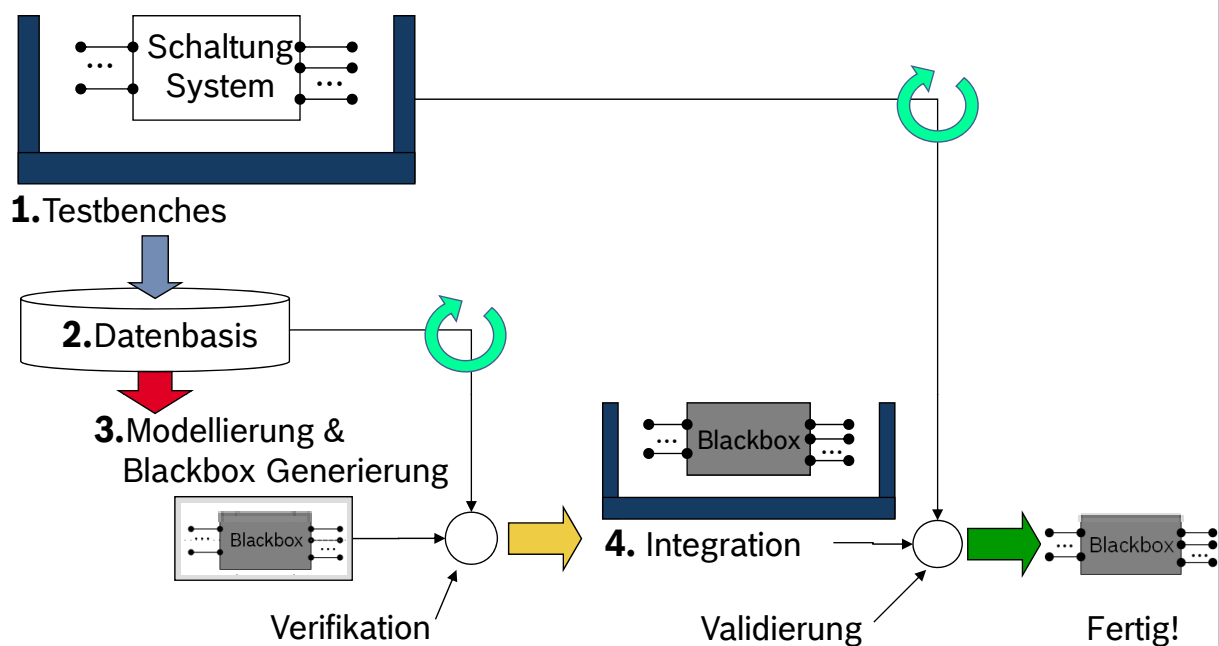
AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



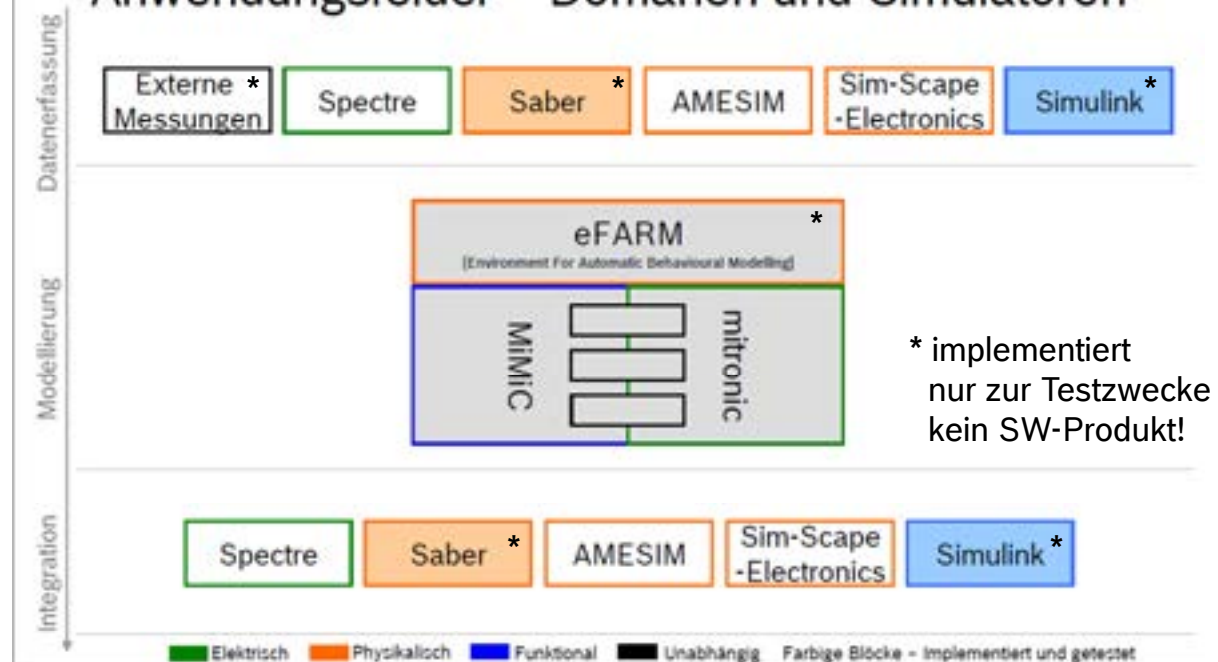
BOSCH

Zusammenfassung Modellierungsergebnisse

Beispiel	Anzahl Bau- elemente	Anzahl Ein- und Ausgangs- pins	Max. Frequenz Ein-/Ausgang (MHz)	Mittlere Genauigkeit (%)	Speed-Up	Modellierungszeit (Std)
Butterworth- Filter	5	1/1	0,2/0,2	98	1x	0,5
Ladungspumpe	120	6/1	3/1	95	Bis zu 37x	2,1
Operations- verstärker	35	5/1	0-1	95	Bis zu 5x	1,7
Stromquelle (Weite des Transistors)	7	3/6	-	95	1x	8,5
Eingangssignalkonverter	20	4/1	3/3 (digital)	95	Bis zu 3x	2,5
CAN (gesamt)	600	18/7	3-5	90	Bis zu 15x	72



Anwendungsfelder – Domänen und Simulatoren



Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

45

Zusammenfassung

Methodik und relevante Innovationsschritte zur vollständigen Automatisierung

- Iteratives Vorgehen bei der Datengenerierung effizient möglich
- Elektrisches Verhalten kann pinweise über Modulatorenstruktur abgebildet werden
- BlackBox Struktur

Anwendungsstudien aus dem Bereich der Automobilindustrie

- Ansatz zeigt auf mehreren Beispielen eine hohe Genauigkeit von über 90%
- Speed-Up (bis zu 37x) ist von der Frequenz, der Schaltungskomplexität und des angestrebten Fehlerniveaus abhängig
- Integration in Gesamtsysteme mit hoher Genauigkeit (98%) möglich
- Portierungen auf anderen Domäne
- Modellierung auch aus Messungen möglich

Automotive Electronics

AE/EIM | 2012-02-23 | © Robert Bosch GmbH 2012. Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.



BOSCH

46

Einige Veröffentlichungen ab 2008:

Dissertation

MIELENZ, Holger: Verhaltensmodellierung in der Kraftfahrzeugtechnik mittels datenbasierter Methoden,
Eberhard-Karls-Universität Tübingen, Dissertation, 2008

SENGER, Philipp ; MIELENZ, Holger ; DÖLLING, Rolando ; ROSENSTIEL, Wolfgang: Efficient Working Area Exploration For Databased-Modeling Of Automotive Applications. In: Proceedings of IADIS Multi Conference on Computer Science and Information Systems IADIS, 2008

XIAO, Shangkun: Adaptive Trainingsdatenerstellung zur automatischen Verhaltensmodellierung von A/MS-Systemen mittels Sensitivitätsanalysen, Fachhochschule Göttingen, Diplomarbeit, Oktober 2009

SENGER, Philipp ; DÖLLING, Rolando ; ROSENSTIEL, Wolfgang: Adaptive Filterselektion innerhalb eines Ansatzes zur automatischen und elektrischen Verhaltensmodellierung. In: VERLAG, VDE (Hrsg.) ; edaCentrum (Veranst.): edaWorkshop 2010 edaCentrum, 2010, S. 15–23

SENGER, Philipp ; XIAO, Shangkun ; DÖLLING, Rolando ; ROSENSTIEL, Wolfgang: Automatic generation of electrical behaviour models by efficient data-based methods. In: ANALOG 2010, ITG/GMM Report, VDE Verlag GmbH, 2010

SENGER, Philipp ; DÖLLING, Rolando ; ROSENSTIEL, Wolfgang: Automated Electrical Behaviour Modelling of Analogue and Mixed Signal Circuits using Data-based Methods. In: Proceeding of IASTED Technology Conferences: Modelling and Simulation; International Association of Science and Technology for Development (IASTED), Actapress, Juli 2010

Dissertation

SENGER, Philipp: Data-based Method for an Automatic Generation of Electrical Behavioural Models in the Automotive Industry
Eberhard-Karls-Universität Tübingen, Dissertation, 2011



„Methode zur automatischen Erstellung von elektrischen
Verhaltensmodellen aus Simulationsdaten,,

Dr.-Ing. Rolando Dölling
Robert Bosch GmbH

Basierend auf den Arbeiten von:

Dr. Philipp Senger, Fraunhofer-Institute for Algorithms and Scientific Computing
Dr. Holger Mielenz, Robert Bosch GmbH

Vielen Dank für Ihre Aufmerksamkeit !



Python-Framework zur Simulation von Strukturdynamik-Modellen

Alexandra Mehlhase, TU Berlin Institut für Softwaretechnik und
Theoretische Informatik
a.mehlhase@tu-berlin.de

Tommy Beckmann, TU Berlin Institut für Softwaretechnik und
Theoretische Informatik
tbeckman@cs.tu-berlin.de

Zusammenfassung

Bei der Modellierung und Simulation von technischen Systemen wird es immer wichtiger detaillierte Modelle zu haben, die möglichst wenig Simulationszeit in Anspruch nehmen. Um dies zu erreichen werden Modelle benötigt, die ihr Gleichungssystem während der Simulation anpassen können. Solche Modelle werden Strukturdynamik-Modelle genannt. Mit diesen Modellen können Verhaltensänderungen und auch Detaillierungsgradänderungen simuliert werden. Im Folgenden wird eine neue objektorientierte Modellierungsebene eingeführt, mit der Strukturdynamik-Modelle modelliert und die Teilmodelle in herkömmlichen Simulationswerkzeugen simuliert werden können. Es wird ein Framework vorgestellt, welches diese Modellierungsebene implementiert und Modellierer bei der Spezifikation von Strukturdynamik-Modellen unterstützt. Das erstellte Modell kann dann mit Hilfe des Frameworks in herkömmlichen Simulationswerkzeugen simuliert werden.

1 Einleitung

In Entwicklungsabteilungen von technischen Systemen gehört die Simulation neuer Systeme zur Tagesordnung. Vor der Erstellung eines Prototyps wird ein (meist mathematisches) Modell erstellt, das das Systemverhalten abbildet. Die Vorteile sind Kosten- und Zeitersparnis sowie erste Abschätzungen, ob ein gedachtes technisches System praktisch realisierbar ist. Zur Modellierung können unterschiedliche Simulationswerkzeuge verwendet werden. Häufig verwendete Werkzeuge sind Matlab in Kombination mit Simulink [8] und Dymola [1]. Beide Werkzeuge bieten gute Möglichkeiten für die Modellierung und Simulation technischer Systeme. Jedoch haben beide Werkzeuge das Defizit, dass für eine Simulation immer nur ein Gleichungssystem gilt. Aus diesem Grund ist es nicht möglich, dass Modellverhalten während der Simulation zu ändern. Ein Flugzeug, das sich am Boden wie ein Fahrzeug verhält, dann in eine Mischung aus Fahr- und Flugzeug übergeht und schließlich zu einem Flugzeug wird,

ist mit beiden Werkzeugen nicht realisierbar. Um solch ein Modell korrekt zu simulieren, müsste es möglich sein das Gleichungssystem während der Simulation auszutauschen.

Durch ein Austauschen des Gleichungssystems könnte ebenfalls der Detaillierungsgrad zur Simulationslaufzeit variiert werden. Diese Maßnahme spart Ressourcen, da das Modell zu jedem Zeitpunkt nur so detailliert wie nötig simuliert werden kann. Modelle, die ihr Gleichungssystem (sowohl Gleichungen als auch Variablen) ändern können, werden als Strukturdynamik-Modelle bezeichnet.

Wir stellen einen Ansatz vor, mit dem die einzelnen Teilmodelle (Modes) eines Strukturdynamik- Modells in den heute gängigen Werkzeugen modelliert und simuliert werden können. Der Wechsel zwischen diesen Modes wird dabei durch eine weitere Modellierungsebene auf Skriptbasis realisiert.

Zunächst wird ein einfaches Strukturdynamik-Modell vorgestellt, welches dann auch für die weiteren Erläuterungen verwendet wird. Daraufhin wird die Grundidee erläutert und analysiert welche Anforderungen die einzelnen Modelle und Werkzeuge erfüllen müssen, um für die Strukturdynamik Anwendung zu finden.

Anhand der Erkenntnisse aus den Grundlagen, wird ein Python-Framework vorgestellt, das es ermöglicht Strukturdynamik-Modelle aus beliebig vielen Modes mit zugehörigen Übergängen zu definieren und zu simulieren. Für jeden Mode eines Strukturdynamik-Modells kann dabei ein individuelles Simulationswerkzeug verwendet werden. Dadurch können die Stärken der einzelnen Werkzeuge genutzt werden.

Mit Hilfe von Beispielen aus verschiedenen Domänen wird das Framework in Bezug auf die Skalierbarkeit und die Anwendbarkeit evaluiert. Zusätzlich wird aufgezeigt, welche Vorteile durch die Modellierung mit Strukturdynamik erzielt werden können.

Es folgt eine Darlegung vom Stand der Technik, in dem weitere Ansätze zur Strukturdynamik vorgestellt werden. Die Zusammenfassung mit einem Ausblick schließt das Papier ab.

2 Grundlagen

Wir betrachten als einleitendes Beispiel ein Modell eines springenden Balls. Der Ball kann am Boden und an einer Wand mit einer elastischen Verformung abprallen (Abbildung 1).

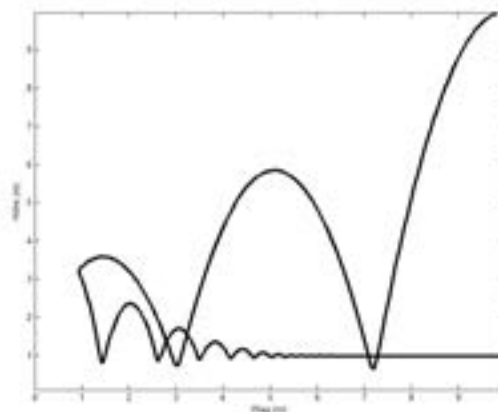


Abbildung 1: Simulationsergebnis eines springenden Balls der vom Boden und links an einer Wand abprallt

Dieses Modell besteht aus drei Modes (siehe Abbildung 2), wobei Transitionen zwischen diesen Modes einen Modewechsel erlauben.

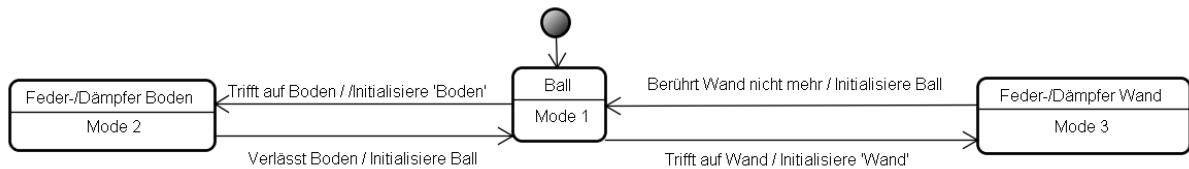


Abbildung 2: Darstellung der drei Modes des springenden Balls

Das Modell startet zunächst als normaler springender Ball, wobei der Ball von rechts geworfen wird. Prallt dieser Ball nun auf dem Boden auf, wird dieser Aufprall durch ein Feder-Masse-Dämpfer Modell (in y Richtung) simuliert. Verlässt der Ball den Boden, geht die Simulation wieder zum fliegenden Ball über. Trifft der Ball auf die Wand, so findet ebenfalls eine elastische Verformung mit Hilfe eines Feder-Masse-Dämpfer Modells (in x Richtung) statt. Auch hier wird wieder auf das ursprüngliche Ball Modell zurückgegriffen, wenn der Ball die Wand nicht mehr berührt.

Simulationswerkzeuge, wie Matlab/Simulink und Dymola, erlauben es zurzeit nicht, solch einen Wechsel zu realisieren. Aus diesem Grund sind neue Methoden notwendig, wobei eine Möglichkeit solch ein Modell zu realisieren im Folgenden vorgestellt wird.

2.1 Grundidee zur Simulation des Modells

Die Grundidee ist, dass die Simulationen der einzelnen Modes mit Hilfe von gebräuchlichen Simulationswerkzeugen stattfindet, damit deren Potenziale genutzt werden können. Da Simulationswerkzeuge, wie Matlab/Simulink und Dymola, es zurzeit nicht erlauben den Übergang von einem Modell in ein anderes Modell während der Simulation zu realisieren, muss jeder Mode in unserem Ansatz ein eigenständig lauffähiges Modell sein. Es wird nun eine neue Modellebene eingeführt, die den Modewechsel vollzieht. Diese zusätzliche Ebene ist durch ein Skript exemplarisch in [3] getestet worden. Dort wurde gezeigt, dass mit Matlab-Skripten Modewechsel in Simulink und auch in Dymola möglich sind.

Abbildung 3 zeigt schematisch den Aufbau eines solchen Skripts, das die Simulation eines Struktur-dynamik-Modells, wie dem springenden Ball, ermöglicht. Dabei werden zunächst alle Modes kompiliert (falls notwendig, wie in Dymola). Darauf folgt die Festlegung des Startmodes (hier Mode 1). Das Skript durchläuft als Nächstes eine Schleife, die so lange ausgeführt wird, bis eine vorgegebene Simulationszeit erreicht ist. Beim ersten Durchlauf der Schleife wird der erste Mode initialisiert und die Simulation des Modes gestartet. Diese Simulation läuft bis eine im Mode definierte Stopp-Bedingung eintritt. In dem Beispiel des springenden Balls wäre dies beispielsweise, wenn der Ball den Boden berührt. Nach dem Terminieren der Simulation liest das Skript die Simulationsdaten ein, um den nächsten Mode zu initialisieren.

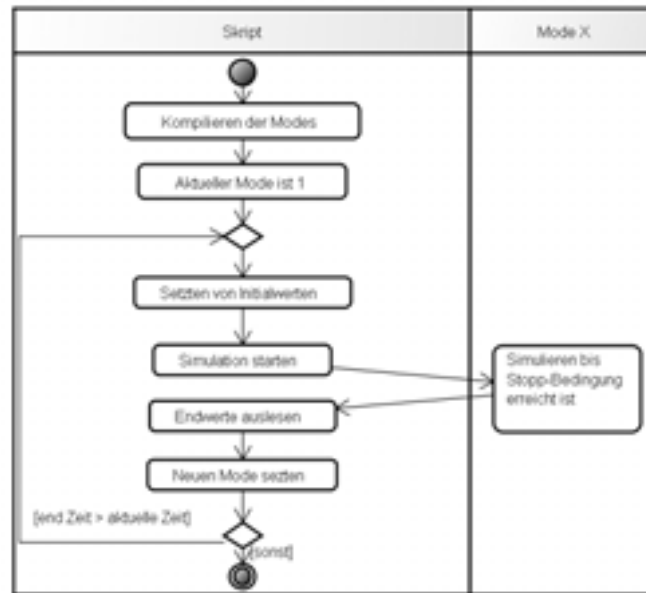


Abbildung 3: Schematische Darstellung des Ablaufs eines Skripts für die Strukturdynamik

2.2 Notwendige Informationen vom Modellierer

Der oben vorgestellte Ansatz zur Simulation von Strukturdynamik-Modellen bedarf der vorherigen Modellierung des Modells. Die einzelnen Modes können in unserem Ansatz in dem jeweilig gewünschten Simulationswerkzeug modelliert werden. Jeder Mode erhält im Gesamtmodell eine eindeutige Identifikationsnummer. Zusätzlich müssen die Übergänge zwischen den Modes beschrieben werden.

Zunächst muss der Modellierer spezifizieren, wann ein Modewechsel stattfinden soll und wohin gewechselt wird. Dies geschieht direkt im Modell des jeweiligen Modes. Der Modellierer gibt eine Stopp-Bedingung an und ebenfalls die Identifikationsnummer des nächsten Modes. In dem Beispiel des springenden Balls mit Boden und Wand hätte der Mode „Ball“ zwei Stoppbedingungen. Die eine tritt ein, wenn der Ball den Boden berührt wobei dann in Mode 2 gewechselt wird und die andere, wenn der Ball die Wand berührt wobei dann in Mode 3 gewechselt wird.

Danach muss spezifiziert werden, wie der neue Mode zu initialisieren ist. Dies geschieht nicht mehr auf der Ebene der Simulationswerkzeuge, sondern auf der höheren, neu eingeführten Ebene.

Der Modellierer muss für jeden Modeübergang angeben welche Daten aus dem vorherigen Mode ausgelesen werden und welche Variablen im neuen Mode damit initialisiert werden. Wird im Ball-Beispiel der Übergang von Mode 1 nach Mode 2 betrachtet, so muss aus dem ersten Mode die Position und Geschwindigkeit des Balls ausgelesen werden. Mit diesen Daten wird der zweite Mode initialisiert. Dabei ist zu beachten, dass die Variablen in den Modes nicht zwangsläufig gleich heißen. In unserem Beispiel besitzt der Ball die Variable „h“, die die Höhe des Ballmittelpunkts beschreibt. Im zweiten Mode des Feder-/Dämpfer-Modells gibt es die Variable „h“ nicht, sondern nur eine Variable „damper.s_rel“, die den

Abstand zum Boden repräsentiert. Der Modellierer muss diese Zusammenhänge erkennen und spezifizieren.

2.3 Notwendige Modelleigenschaften

Zusätzlich zu den Angaben, die der Modellierer machen muss, müssen die Modelle der Modes Anforderungen erfüllen, um als Mode in einem Strukturdynamik-Modell verwendet werden zu können. Jeder Mode muss ein autarkes Gleichungssystem enthalten und somit in einem Simulationswerkzeug simulierbar sein. Hinzu kommt die oben erwähnte Stopp-Bedingung. Diese Zusatzinformation verändert das Modell des Modes und verhindert, dass das Modell noch eigenständig (außerhalb des Strukturdynamik-Modells) sinnvoll verwendet werden kann. Soll das alte Modell unverändert bleiben, bieten objektorientierte Modellierungssprachen, wie Modelica, die Möglichkeit der Generalisierung. So ist es möglich ein normales Ballmodell zu haben und ein Ballmodell, das von dem anderen Modell erbt und zusätzliche Informationen (wie die Stopp-Bedingung) enthält, siehe Abbildung 4.

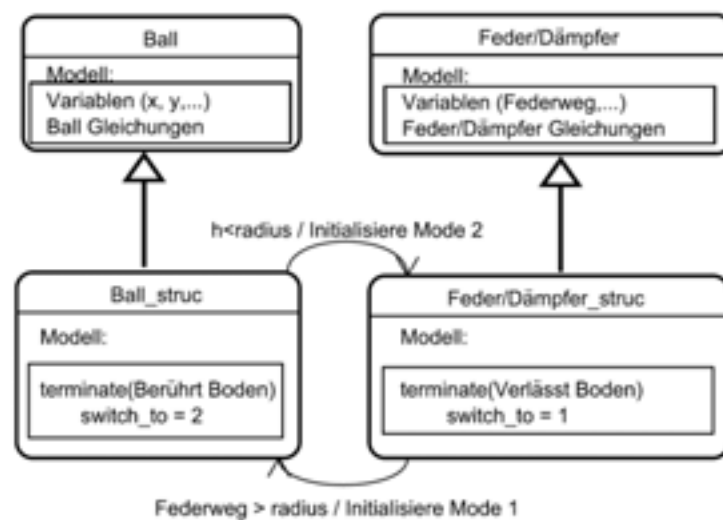


Abbildung 4: Schematische Darstellung der Spezialisierung in einem Strukturdynamik-Modell

Des Weiteren müssen die einzelnen Modes ausreichend Informationen für die Initialisierung des nächsten Modes zur Verfügung stellen. Wäre dies nicht der Fall, so könnte der neue Mode nicht eindeutig initialisiert werden und würde evtl. mit falschen Startbedingungen simuliert. Dies kann zu einer unstetigen Lösung führen, bringt aber auf jeden Fall Fehler in den Simulationsergebnissen mit sich. Daraus folgt, dass nicht alle Modelle als Modes für Strukturdynamik-Modell sinnvoll verwendet werden können.

Weiterhin muss sichergestellt sein, dass die Variablen die zur Initialisierung des Modes notwendig sind, auch verändert werden können. Wird beispielsweise ein Modell aus zwei Behältern betrachtet für das Initial festgelegt ist, dass der zweite Behälter halb so voll ist wie der erste, so kann das Modell von außen (über ein Skript) nicht anders initialisiert werden. Nur durch ein Ändern des Modells wäre dies möglich. Solch ein Modell wäre für die Strukturdynamik, wie sie hier vorgestellt wird, nicht geeignet.

2.4 Notwendige Werkzeugeigenschaften

Auch die verwendeten Simulationswerkzeuge unterliegen Anforderungen, die wir kurz erläutern. Damit die einzelnen Modes in einem bestimmten Werkzeug simuliert werden können, muss das Werkzeug von außen ansteuerbar sein. Dymola ist beispielsweise durch eine DDE Schnittstelle steuerbar, Simulink durch Konsolenbefehle und OpenModelica durch Befehle in der OpenModelica Shell.

Um einen Modewechsel zu vollziehen, müssen die Werkzeuge es erlauben, die Modelle extern zu initialisieren, damit die Modelle nicht verändert werden müssen. In Matlab kann dies über Initialskripte oder Workspace Variablen geschehen, in Dymola und OpenModelica über Initialdateien (dsin.txt).

Um die Effizienz nicht negativ zu beeinflussen, sollte jeder Mode ohne wiederholte Kompilierung simulierbar und initialisierbar sein. In unseren springenden Ball Beispiel wäre es nicht sinnvoll, jedes Mal das Ball-Modell neu zu kompilieren, wenn in diesen Mode gewechselt wird.

3 Framework für die Strukturdynamik

Aus dem vorherigen Abschnitt ist bekannt, wie Strukturdynamik-Modelle simuliert werden können und durch welche Informationen ein solches Modell beschrieben werden kann.

Wir stellen hier nun einen objektorientierten Aufbau vor, in dem die notwendigen Informationen eines Strukturdynamik-Modells enthalten sind. Darauf folgt die Vorstellung einer prototypischen Implementierung eines Frameworks, mit dem Strukturdynamik-Modelle spezifiziert und simuliert werden können.

3.1 Objektorientierter Aufbau

Aus den Informationen aus dem vorherigen Kapitel ist bekannt, dass ein Strukturdynamik-Modell aus mehreren Modes bestehen kann, wobei jeder Mode mit einem bestimmten Simulationswerkzeug erstellt wurde. Zusätzlich ist bekannt, dass aus jedem dieser Modes Transitionen zu anderen Modes führen können und diese Transitionen Informationen über den Modewechsel enthalten. Um diese Zusammenhänge darzustellen, bietet sich ein objektorientierter Aufbau an. Abbildung 5 zeigt die konkrete Umsetzung als Klassenmodell.

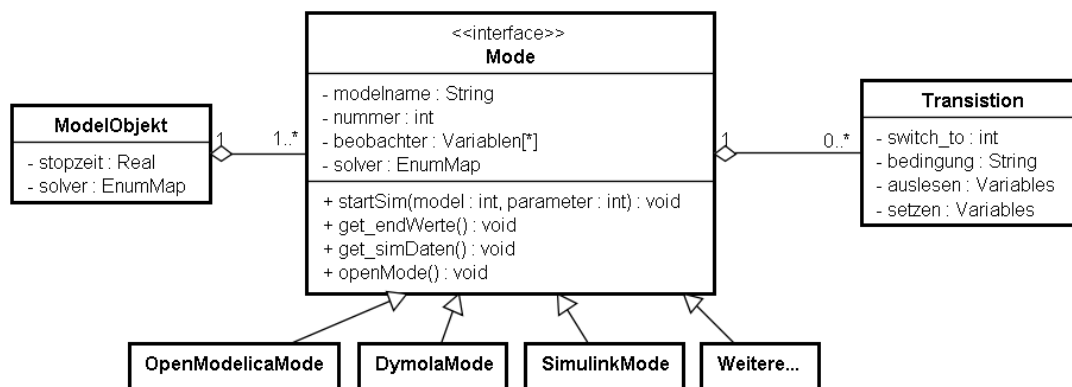


Abbildung 5: Objektorientierter Aufbau eines Strukturdynamik-Modells

Hier gibt es ein *ModelObjekt*, das das komplette Strukturdynamik-Modell beschreibt. In diesem Modell sind die Simulationszeit sowie ein Standardlöser festgelegt. Ein solches Objekt besteht aus mehreren Modes. Jedes dieser Modes enthält Informationen, die für einen Modewechsel erforderlich sind, wie beispielsweise den Namen des mathematischen Modells und die Mode-Identifikationsnummer. Zusätzlich kann ein Beobachter angegeben werden, der festlegt welche Variablen am Ende der Simulation gespeichert werden. Ebenfalls kann ein Löser spezifiziert werden, der statt des Standardlösers aus dem *ModelObjekt* verwendet werden soll.

Darüber hinaus muss jeder Mode mehrere Methoden bereitstellen, wie beispielsweise zum Initialisieren eines Modells und zum Starten einer Simulation. Diese Methoden sind für jeden Mode notwendig, doch sind die Methoden für verschiedene Simulationswerkzeuge unterschiedlich. So funktioniert das Initialisieren von Modes bei Dymola und OpenModelica zwar jedes Mal über Initialdateien, diese haben jedoch nicht das gleiche Format. Aus diesem Grund ist ein Interface eingeführt worden, das die notwendigen Methoden definiert. Für jedes Werkzeug muss eine Klasse, die dieses Interface implementiert, erstellt werden.

Jeder Mode kann Transitionen enthalten, was bedeutet, dass aus diesem Mode in einen anderen gewechselt werden kann. In einer Transition wird die eindeutig Identifikationsnummer des nächsten Modes gespeichert, sowie die Information, wie der neue Mode mit den Daten des alten Modes initialisiert werden muss.

Mit einem solchen Aufbau und dem vorgegebenen Interface, können neue Simulationswerkzeuge einfach hinzugefügt werden. Für die Modellierung von Strukturdynamik-Modellen müssen die Werkzeuge lediglich die Anforderungen aus Abschnitt 2.4 erfüllen.

Wird das Strukturdynamik-Modell des springenden Balls, wie oben beschrieben, aufgebaut, ergibt sich die Objektsicht aus Abbildung 6. Es gibt drei Modes, zwei *Dymola Modes* und ein *OpenModelica Mode*. Die Modes haben keine Solver-einstellungen, daher wird der global definierte Solver verwendet. Der Ball-Mode hat zwei Transitionen, wobei bei einem Modewechsel anhand der eingetretenen Stopp-Bedingung entschieden wird, welche Transition benutzt werden muss. Dafür wird aus dem Mode, die Variable *switch_to* ausgelesen, die dann mit dem *switch_to* aus den Transitionen verglichen wird. Zusätzlich werden in der Transition die Variablen deklariert, die aus dem alten Mode ausgelesen und im neuen Mode gesetzt werden.

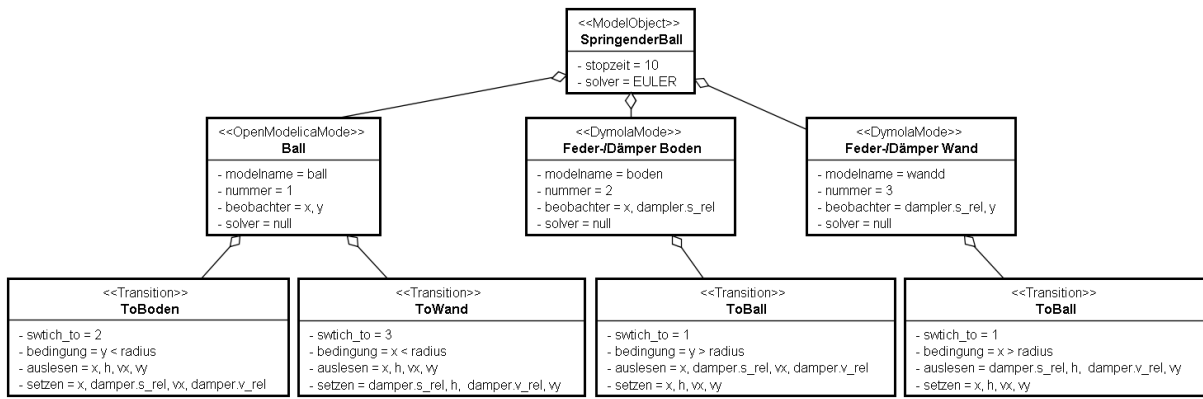


Abbildung 6: Darstellung der drei Modes des springenden Balls

3.2 Prototypische Implementierung

Für die Implementierung des Prototyps wurde die Skriptsprache Python verwendet, da sie frei verfügbar ist und zum anderen sehr weit verbreitet. Dies erleichtert dem Anwender die Erweiterung des Frameworks.

Der in Abschnitt 3.1 vorgestellte Aufbau wurde in Python übernommen. Für jedes implementierte Werkzeug wurde das Mode-Interface implementiert.

Der Modellierer muss sich über den Aufbau des Frameworks jedoch keine Gedanken machen, da ein Template genügt, um das Model zu spezifizieren. Zum Nutzen des Templates genügt das Grundverständnis des Modellaufbaus aus Abbildung 5 sowie die Kenntnis über das eigene mathematische Modell. Das Templates für den springenden Ball ist im Folgenden dargestellt:

```

ALGO = EULER          # Solver
SIMTIME = 20          # Simulationszeit
MODES = ['mechanik.ball_struc', 'mechanik.contact_struc', 'mechanik.contact_wall'] # Modell Name
outputVariablesToSave(['x','h'], ['x', 'damper.s_rel'], ['x', 'h']) # Beobachter [[Mode 1], [Mode 2], [Mode 3]]
transition1_2 = mode.transition(2, ['x','h','vx','vy'], ['x','damper.s_rel','vx','damper.v_rel'])
transition1_3 = mode.transition(3, ['x','h','vx','vy'], [damper.s_rel,'h','damper.v_rel','v'])
transition2_1 = mode.transition(1, ['x','damper.s_rel', 'vx','damper.v_rel'], ['x','h', 'vx','vy'])
transition3_1 = mode.transition(1, [damper.s_rel, 'h','damper.v_rel', 'v'], ['x','h','vx','vy'])
  
```

Code 1: Ausgefülltes Template des springenden Balls

Ist dieses Template ausgefüllt, wird das Strukturmechanik-Modell in den objekt-orientierten Aufbau überführt. Dieses Strukturmechanik-Modell wird dann als Übergabeparameter an eine Methode „switch“ übergeben. Diese Methode funktioniert wie das Skript aus Abbildung 3 nur, dass jetzt auch mehrere Werkzeuge ansteuerbar sind. Je nachdem welcher Mode aktiv ist, wird die für diesen Mode richtige „startSim“ Methode aus der entsprechenden Klasse aufgerufen. Nach jeder Simulation eines Modes werden die zu beobachtenden Variablen in einer Matrix gespeichert, wobei zwei zusätzliche Spalten mit der Simulationszeit und der aktuellen Mode-Identifikationsnummer hinzugefügt werden. Die gesammelten Simulationsdaten werden am Ende der Gesamtsimulation in eine Datei gespeichert. Zurzeit

werden die Daten im Mat-Format gespeichert jedoch sind auch andere Formate wie XLS und HDF5 integrierbar. Abbildung 7 gibt einen schematischen Überblick über den Aufbau des Frameworks.

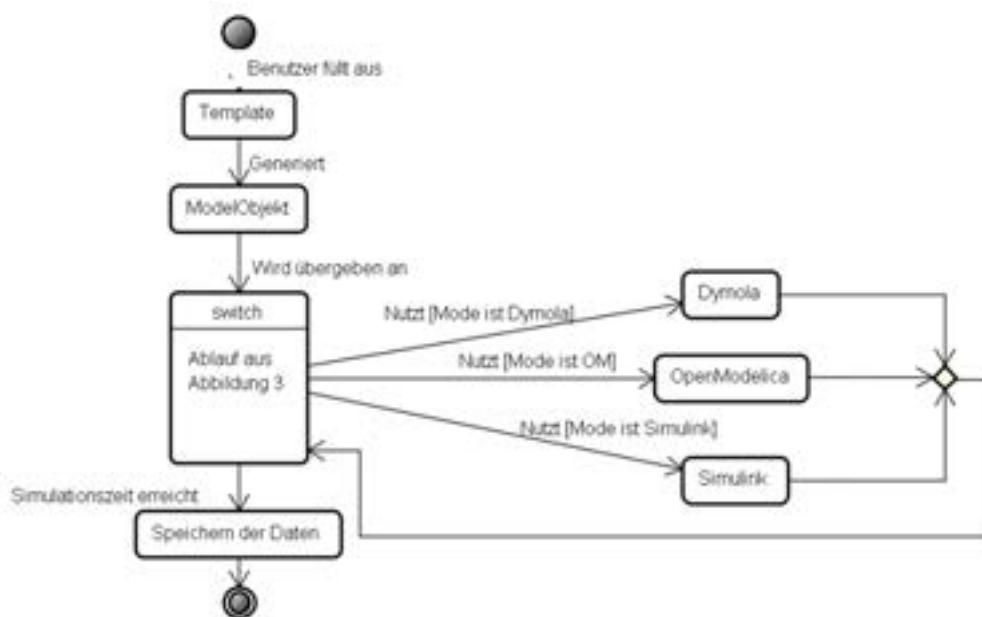


Abbildung 7: Prinzipieller Aufbau des Frameworks

4 Evaluation

Das beschriebene Framework wurde mit unterschiedlichen Strukturdynamik-Modellen evaluiert. Dazu wurden Modelle aus unterschiedlichen Domänen gewählt. Solange es sich um ein mathematisches Modell handelt, das aus Differenzial- und algebraischen Gleichungen bestehen ist das Framework verwendbar. Dabei ist es irrelevant in welchem Werkzeug die einzelnen Modes sind und aus wie vielen Modes und Transitionen sich das Gesamtmodell zusammensetzt. Das gegebene Template führt den Nutzer durch die Beschreibung des Modells und verhindert Fehler in der Modellierung.

Da die Simulationszeit ein sehr wichtiger Faktor bei der Modellierung ist wurde das Framework auf seine Effizienz und seine Skalierbarkeit hin untersucht. Dazu wurde zum einen ein sehr einfaches Modell verwendet mit dem sehr viele Modeübergänge simuliert wurden und zum anderen ein Modell mit vielen Zustandsvariablen, die bei einem Modeübergang alle zu initialisieren sind. Es wurde herausgefunden, dass ein Modeübergang bei vier Zustandsvariablen ca. 0,05 Sekunden in Anspruch nimmt und bei 40.000 Zustandsvariablen ca. 0,7 Sekunden. Dies zeigt, dass die Zeit für einen Modewechsel zwar von den zu initialisierenden Zustandsvariablen abhängt, jedoch sehr gering ist. Das Framework ist demnach auch für große Modelle geeignet. Es sollte jedoch vermieden werden sehr viele Zustandswechsel bei sehr kurzer Simulationszeit vorzunehmen, da dann die benötigte Simulationszeit negativ beeinflusst wird.

Um zu testen, ob durch einen Detaillierungsgradwechsel Simulationszeit gespart werden kann wurde ein Modell eines Dieselmotors verwendet. Dieses Modell liegt in zwei Detaillierungsgraden vor. Dieses Strukturdynamik-Modell wurde so spezifiziert, dass zu jeder Zeit der

möglichst geringe Detaillierungsgrad verwendet wird und nur wenn notwendig das detailliertere Modell. Dabei konnte die Simulationszeit von 19 Sekunden auf 12 Sekunden verringert werden (für 20 Sekunden simulierte Laufzeit). Die Abweichung der Zylindertemperatur und des Zylinderdrucks betrug weniger als ein halbes Prozent. Dies hat gezeigt, dass mit Hilfe von Strukturdynamik-Modellen schneller simuliert werden kann, ohne dabei signifikanten Genauigkeitsverluste zu haben. Natürlich sind diese Vorteile von den verwendeten Modellen abhängig. Ein Detaillierungsgradwechsel ist nur sinnvoll, wenn das weniger detaillierte Modell keine zu großen Fehler produziert und es weniger Zeit zum Simulieren benötigt. Ist dies nicht der Fall, kann mit der Strukturdynamik kein Zeitgewinn erzielt werden. Geht es bei dem Strukturdynamik-Modell um eine Verhaltensänderung und nicht um eine Detaillierungsgradänderung, so ist der Zeitfaktor weniger relevant, da das Modell ohne Strukturdynamik erst gar nicht simulierbar ist. Hier ist es wichtig, dass das Modell die Anforderungen aus Abschnitt 2.3 erfüllt.

5 Stand der Technik

Mit aktuellen Werkzeugen ist die Simulation von Strukturdynamik Modellen mit gewissen Einschränkungen durchaus möglich. Dieses Kapitel gibt einen Überblick über diese Varianten.

An erster Stelle sei MOSILAB [6] genannt. MOSILAB ist ein auf Modelica basierendes Werkzeug, das die Modellierung und Simulation von Strukturdynamik Modellen ermöglicht. Hierbei wird die Modellierungssprache Modelica um Zustandsübergangsdiagramme erweitert, die den Modewechsel beschreiben. Der Nachteil von diesem Werkzeug ist, dass bislang nur Index-0 Modelle unterstützt werden. Ziel ist es, die Erkenntnisse aus unserer Arbeit zu nutzen, um MOSILAB zu verbessern.

Als nächstes sei die Modellierungssprache SOL [10] erwähnt. Diese Sprache unterstützt die Modellierung und Simulation von Strukturdynamik-Modellen. Ist ein Modewechsel notwendig und ändert sich dabei nur eine Teilkomponente, so wird nur der notwendige Teil kausalisiert und nicht das komplette Modell, wie in unserem Ansatz. Da es sich bei der Sprache um eine experimentelle Sprache handelt, ist sie nicht frei zugänglich und hat nicht die gleichen Möglichkeiten, wie kommerzielle Simulationswerkzeuge.

Eine weitere Möglichkeit Strukturdynamik-Modelle zu simulieren ist Keymaera [7]. Hierbei handelt es sich um ein Verifikationswerkzeug für hybride Systeme. Nachteil hierbei ist, die Beschreibungssprache hat keine Ähnlichkeit zu Modelica. Der Entwickler muss sich für die Simulation von Strukturdynamik in eine komplett neue Umgebung einarbeiten.

Auch Hydra [5] stellt eine Möglichkeit zur Simulation von Strukturdynamik-Modellen bereit. Diese Sprache basiert auf funktionaler Programmierung und ist somit nicht für jeden Modellierer leicht erlernbar.

All die genannten Werkzeuge und Sprachen zeigen gute Möglichkeiten für den Umgang mit Strukturdynamik-Modellen. Die wesentlichen Nachteile sind, dass vorhandene Modelle nicht einfach wiederverwendet werden können und eine Einarbeitung in ein neues Werkzeug oder eine neue Sprache notwendig ist.

Eine weitere Möglichkeit ist in [9] beschrieben. Dort wird ein Strukturdynamik-Modell so umgeformt, dass es nur noch ein Modell ist. Dabei werden Variablen umbenannt und die Gleichungen mit „If-then-else“ Konstruktion dargestellt. Dieser Ansatz ist zwar eine Möglichkeit, macht das Gesamtmodell jedoch sehr groß und wird dadurch die Simulationszeit beeinflussen und das Modell unübersichtlicher machen.

In [2] wird ein Ansatz beschrieben, wie in Dymola Strukturdynamik-Modell umgesetzt werden können. Dabei werden die Modelle angepasst, indem mit Booleschen-Werten gearbeitet wird, wodurch ein Teil von Gleichungen mit Null multipliziert wird und damit nicht mehr betrachtet wird. Auch hier muss der Modellierer die Strukturdynamik direkt im Modell beschreiben und das eigentliche Modell verändern.

Das vorgestellte Framework bietet eine einfache Alternative zu den vorgestellten Ansätzen. Der Modellierer ist mit unserem Ansatz in der Lage, seine bekannten Werkzeuge zu nutzen und diese für die Strukturdynamik einzusetzen, ohne sein Modell verändern zu müssen. Der Modellierer hat mit dem neuen Framework erste Evaluierungsmöglichkeiten, ob der Einsatz von Strukturdynamik für ihn sinnvoll ist, oder nicht ohne neue Werkzeuge oder Sprachen lernen zu müssen.

6 Zusammenfassung und Ausblick

Es wurde eine Möglichkeit zur Modellierung und Simulation von Strukturdynamik-Modellen vorgestellt, die mit gängigen Simulationswerkzeugen arbeitet und somit ein Wiederverwenden von vorhandenen Modellen erlaubt. Dazu wurde ein objektorientierter Aufbau entwickelt, der es zulässt Strukturdynamik-Modelle zu beschreiben. Ein Python Framework, das diesen Ansatz implementiert und es ermöglicht verschiedene Simulationswerkzeuge zu verwenden wurde vorgestellt. Mit Hilfe von einfachen Modellen konnte gezeigt werden, dass das Framework die Modewechsel effizient behandeln kann. Es wurde gezeigt, dass die Strukturdynamik es zum einen ermöglicht Simulationszeit bei Detaillierungsgradwechseln zu sparen und zum anderen, dass Modelle erstellt werden können, die ihr Verhalten während der Simulation ändern.

Das vorgestellte Framework gibt Anwendern die Möglichkeit, Strukturdynamik-Modelle einfach und effizient zu erstellen und zu simulieren, ohne sich in ein neues Werkzeug oder eine neue Sprache einzulernen.

Zukünftig soll das Framework auf weitere Werkzeuge erweitert werden und eine Benutzeroberfläche zur Verfügung stellen, um die Nutzung noch weiter zu verbessern. Das Framework soll dazu dienen Untersuchungen zu Strukturdynamik-Modell zu ermöglichen, um mehr über diese Modelle zu lernen. Anwender sollen in der Lage sein zu testen, ob Strukturdynamik für sie Vorteile bringt.

Das Framework soll auf lange Sicht den Nutzer durch eine komplette Methodik leiten, um Strukturdynamik-Modelle sinnvoll und effizient zu erstellen.

Literatur

- [1] Dassault Systems: *URL www.dynasim.se*, 2010
- [2] Elmqvist, H., Cellier, F.E., and Otter, M.: *Object- oriented modeling of hybrid systems*, 2003
- [3] Mehlhase, A.: *Varying the level of detail during simulation*, 21.Symposium Simulationstechnik, ASIM Pub., 2011
- [4] Modelica Association: *Modelica - A Unified Object-Oriented Language for Physical Systems Modeling - Language Specification Version 3.2.*, 2010
- [5] Nilsson, H. and Giorgidze, G., *Exploiting structural dynamism in Functional Hybrid Modelling for simulation of ideal diodes*, Proceedings of the 7th EUROSIM Congress on Modelling and Simulation. Czech Technical University Publishing House, 2012
- [6] Nytsch-Geusen, C., Ernst, T., Nordwig, A., and et al.: *Mosilab: Development of a modelica based generic simulation tool supporting model structural dynamics*, Proceedings of the 4th, International Modelica Conference, TU Hamburg-Harburg, 2008
- [7] Platzer, A., Quesel, J.D.: *Keymaera: A hybrid theorem Prover for hybrid systems*, *IJCAR. VOLUME 5195 OF LNCS*, Springer, 2008
- [8] The Mathworks: *Simulink R7 User's Guide (Release 2010b)*, 2010
- [9] Urquia, A., Dormido, S.: *Object-oriented description of hybrid dynamic systems of variable structure*. Simulation, 2003.
- [10] Zimmer, D.: *Equation-Based Modeling of Variable- Structure Systems*. Ph.D. thesis, Swiss Federal Institute of Technology, 2010

Simulation und Algorithmenanalyse: Ein generisches Vorgehensmodell für Online-Optimierungsprobleme mit Lookahead

Fabian Dunke, Institut für Operations Research,
Karlsruher Institut für Technologie, fabian.dunke@kit.edu
Stefan Nickel, Institut für Operations Research,
Karlsruher Institut für Technologie, stefan.nickel@kit.edu

Zusammenfassung

Im Gegensatz zur klassischen Offline-Optimierung, bei der alle Eingabedaten zu Beginn feststehen, beschäftigt sich die Online-Optimierung mit Situationen, in denen die Eingabedaten sequentiell bekannt werden. Bislang gibt es keinen Modellrahmen zur Untersuchung derartiger Probleme, der eine Berücksichtigung verschiedener Größen für die Informationsvorausschau (Lookahead) eines Algorithmus erlaubt.

Zunächst stellen wir verschiedene Lookaheadarten vor und führen darauf aufbauend ein generisches Vorgehensmodell für Online-Optimierungsprobleme mit Lookahead ein. Dieses ermöglicht eine simulationsbasierte Analyse und Bewertung von Algorithmen in verschiedenen Problemstellungen mit einheitlicher Notation und Vorgehensweise.

In numerischen Experimenten wurde das Vorgehensmodell für Bin Packing und Traveling Salesman Probleme instanziiert und in einer simulationsgestützten Optimierung zur Algorithmenanalyse eingesetzt. Die Ergebnisse zeigen, dass die Auswirkungen einer Informationsvorausschau stark von der betrachteten Problemstellung selbst abhängen.

1 Online-Optimierung mit Lookahead in der Simulation

Die Funktionslogik von Simulationen zur Analyse und Steuerung komplexer Systeme erfordert zu mehreren Zeitpunkten ein dynamisches Entscheiden, um fortfahren zu können. In solchen Fällen muss ein Algorithmus mit Hilfe der aktuell bekannten Eingabedaten eine Entscheidung treffen, die trotz unbekannter Zukunft zu einer möglichst guten Ausgangssituation für den weiteren Simulationslauf führen soll. Da der unter unvollständiger Information arbeitende Algorithmus nach Bekanntgabe neuer Eingabedaten immer wieder aufgerufen wird, bezeichnet man ihn als Online-Algorithmus. Unklar an dieser Begriffsbildung ist, in welchem Ausmaß die Unvollständigkeit der zur Verfügung stehenden Informationen vorliegt. Wir behandeln diese Problematik, indem wir eine formale Beschreibung des Begriffs *Lookahead* geben. Die zugehörigen Probleme bezeichnen wir als *Online-Optimierungsprobleme mit Lookahead*.

Methoden zur Lösung derartiger Probleme zeichnen sich durch sequentielles, dynamisches Entscheiden aus. In einer simulationsbasierten Vorgehensweise werden die Optimierungsalgorithmen von der Simulation iterativ in einer Subroutine aufgerufen und man erhält die hierarchische Verknüpfung in Abbildung 1 (vgl. hierzu [5]):

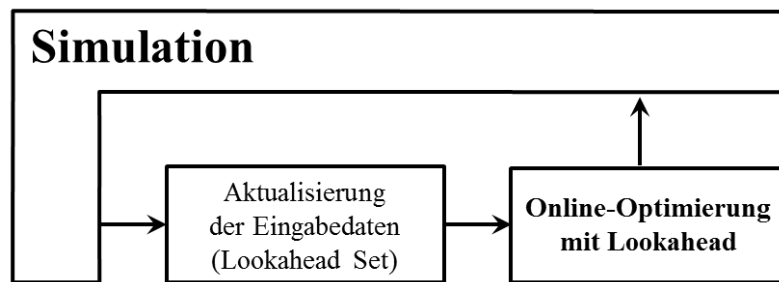


Abbildung 1: Verknüpfung von *Simulation* und *Online-Optimierung mit Lookahead*

Hieraus leiten sich bzgl. einer Analyse von (Optimierungs-) Algorithmen im Rahmen einer Simulation folgende Fragen ab:

- Welche Algorithmen eignen sich zur Online-Optimierung mit Lookahead im Rahmen eines Simulationslaufs bei einem gegebenen Problem?
- Wie hängt die bei einem Simulationslauf durch einen Algorithmus erzielte Güte des Optimierungsergebnisses von der Größe der Informationsvorausschau ab?

2 Grundlagen der Online-Optimierung mit Lookahead

2.1 Optimierungsparadigmen

Die Ausführung von Algorithmen zur Lösung eines Problems unterliegt neben den verfügbaren Rechen- und Speicherressourcen v.a. der zeitlichen Verfügbarkeit der Eingabedaten. Dies legt eine Klassifizierung in drei Optimierungsparadigmen nahe:

- Die *Offline-Optimierung* geht von vollständiger Information vor der Optimierung aus.
- Die *Online-Optimierung* behandelt Probleme, bei denen die Eingabedaten nach und nach bekannt werden und ein Algorithmus die Lösung des Gesamtproblems aus Lösungen der Teilprobleme zusammensetzen muss.
- Bislang nur unsystematisch wurden Probleme untersucht, die sich zwischen Online- und Offline-Problemen befinden. Wir bezeichnen diesen Fall als *Online-Optimierung mit Lookahead* und charakterisieren ihn dadurch, dass in jedem Zeitpunkt ein quantifizierbarer Anteil zukünftiger Eingabedaten zur Verfügung steht.

Beispiel: Bin Packing

Die Gegenstände in Abbildung 2a) mit den angegebenen Größen sind in möglichst wenige Behälter der Größe 1 zu packen. Ist die Menge aller Gegenstände bekannt (Offline-Optimierung), so lässt sich eine Lösung mit sechs Behältern bestimmen (Abbildung 2b)).

Erscheinen die Gegenstände nacheinander (Online-Optimierung) und müssen sie sequentiell ohne Kenntnis zukünftiger Gegenstände jeweils unmittelbar nach Bekanntgabe gepackt werden, so werden mit dem Algorithmus BEST FIT, der einen Gegenstand in den Behälter mit höchstem Füllstand legt, gemäß Abbildung 2c) acht Behälter benötigt. Sind immer die beiden nächsten Gegenstände bekannt (Online-Optimierung mit Lookahead der Größe 2) und packt man den größeren der beiden Gegenstände nach BEST FIT, so erhält man sieben Behälter (Abbildung 2d)).

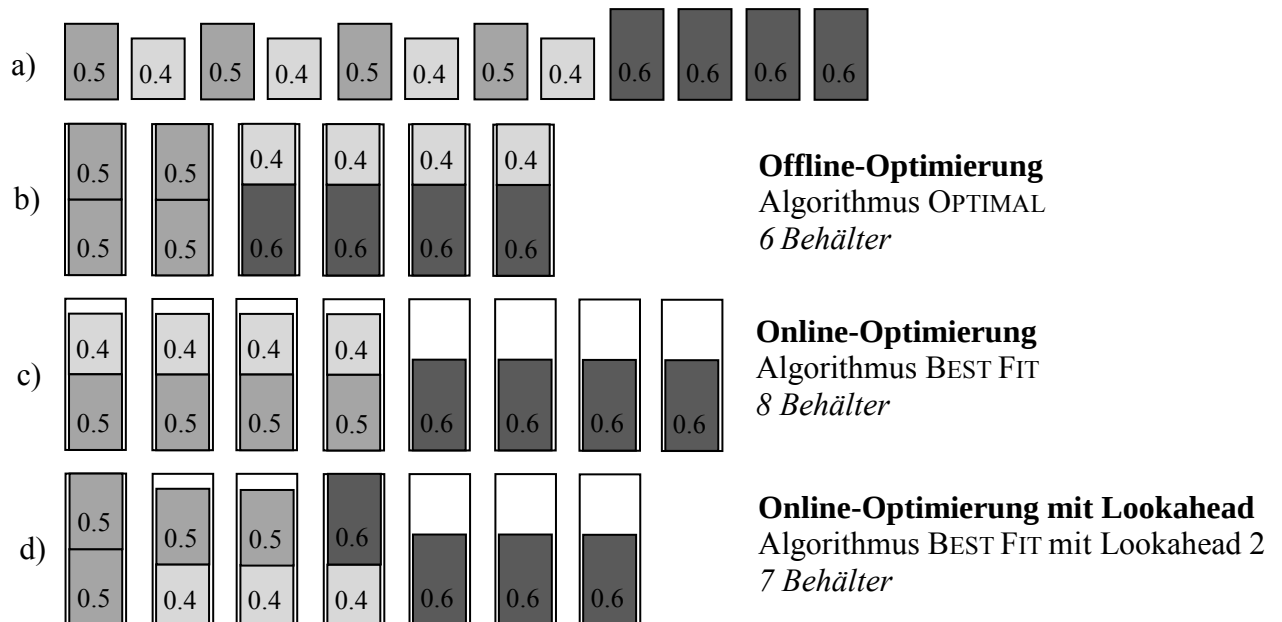


Abbildung 2: Drei Optimierungsparadigmen im Bin Packing Problem

Im weiteren Verlauf seien die Eingabedaten in Form einer Eingabefolge $\sigma = (\sigma_1, \sigma_2, \dots, \sigma_n)$ gegeben. Welcher Teil von σ einem Algorithmus bekannt ist, hängt vom Optimierungsparadigma und der Lookaheadart ab. Die Elemente von σ nennen wir Eingabeelemente. Die Menge der zu einem Zeitpunkt bekannten Eingabeelemente heißt Lookahead Set.

2.2 Lookaheadarten

Eine Vielzahl von Konzepten zur genaueren Beschreibung einer Informationsvorausschau wurde in der Literatur problemspezifisch vorgeschlagen. Die beiden folgenden Definitionen erscheinen intuitiv und sind *nicht* problemspezifisch (siehe [3]):

Request Lookahead: Zu jedem Zeitpunkt kennt der Algorithmus eine fixe Anzahl K an zukünftigen Eingabeelementen. Diese werden in der Form $\sigma_i = (r_i, \pi_i)$ spezifiziert, wobei r_i der eigentlichen Information des Elements entspricht (z.B. Größe eines Gegenstands, Bearbeitungsdauer eines Jobs, frühestmöglicher Besuchszeitpunkt) und π_i die Position in der Eingabefolge angibt. Als Lookahead Set zum Zeitpunkt t erhält man diejenigen K Eingabeelemente mit niedrigster Position, die noch nicht verarbeitet wurden. Beispiele sind Datenblöcke in Computeranwendungen, Materialbedarfe an Fertigungslinien mit fester Stationsanzahl oder Container für eine fixe Produktanzahl bei Pack- und Verladevorgängen.

Time Lookahead: Zu jedem Zeitpunkt t kennt der Algorithmus diejenigen Eingabeelemente, die im Fall ohne Informationsvorausschau bis spätestens $t + D$ bekannt gegeben würden. D wird als Lookaheaddauer bezeichnet. Eingabeelemente werden in der Form $\sigma_i = (r_i, T_i)$ spezifiziert, wobei r_i wie im Fall von Request Lookahead zu verstehen ist und T_i den Bekanntgabezeitpunkt im Fall ohne Informationsvorausschau angibt. Als Lookahead Set zum Zeitpunkt t erhält man diejenigen Eingabeelemente σ_i mit Bekanntgabezeit $T_i \leq t + D$, die noch nicht verarbeitet wurden. Beispiele finden sich beim Routing von Einsatzfahrzeugen, bei der Picklistenerzeugung (Kommissionieren) sowie in der dynamischen Fahrgastinformation.

Die Einteilung in Request und Time Lookahead ist nicht erschöpfend, stellt jedoch geeignete Prototypen für die Definition weiterer Lookaheadarten bereit (vgl. auch [1]).

Neben der Identifikation der in einer Anwendung vorliegenden Art der Informationsvorausschau muss zudem noch eine Angabe über die Berechtigungen für die Bearbeitung der bekannten Eingabeelemente angegeben werden:

- Bei einer *wahlfreien Bearbeitung* dürfen die Eingabeelemente im Lookahead Set in beliebiger Reihenfolge bearbeitet werden. Wahlfreie Bearbeitung profitiert nicht nur von der Information r_i der bekannten Eingabeelemente, sondern auch von Permutationsmöglichkeiten. Ein Beispiel sind kleine Gegenstände in einem Materialpuffer.
- Bei einer *sequentiellen Bearbeitung* müssen die Eingabeelemente im Lookahead Set in der Reihenfolge der Bekanntgabe abgearbeitet werden. Sequentielle Bearbeitung profitiert nur von der Information r_i der im Lookahead Set befindlichen Elemente. Ein Beispiel sind große unhandliche Gegenstände in einem Materialpuffer.

2.3 Bewertungsmaße für Online-Algorithmen mit Lookahead

In der Online-Optimierung hat sich die Kompetitivität als Gütekriterium für Algorithmen etabliert¹ (siehe [4]). Nachfolgend gehen wir von Minimierungsproblemen aus. Ein Online-Algorithmus ALG ist c -kompetitiv, wenn für alle σ gilt, dass $\text{ALG}(\sigma) \leq c \cdot \text{OPT}(\sigma)$, wobei $\text{OPT}(\sigma)$ die minimalen Kosten sind, die ein optimaler *Offline*-Algorithmus für σ benötigt, der alle Informationen zu $t = 0$ besitzt ($c \geq 1$). Als Kompetitivität von ALG bezeichnet man das kleinste c , so dass ALG c -kompetitiv ist. Als weiterer Ansatz zur Algorithmenbewertung ist die bijektive Analyse (siehe [2]) zu erwähnen. Hier vergleicht man die Verteilungsfunktionen der Zielfunktionswerte über alle Eingabefolgen hinweg für unterschiedliche Algorithmen.

Für Online-Algorithmen mit Lookahead gibt es keine eigenen Bewertungsansätze. In der Literatur werden in speziellen Problemstellungen Verbesserungen in der Kompetitivität durch Lookahead nachgewiesen. Diese Art der Analyse ist jedoch wie die kompetitive Analyse selbst mit der Worst-Case-Problematik behaftet und kann zu übermäßig pessimistischen Beurteilungen führen. Dies ist insbesondere aus praktischer Sicht nicht wünschenswert und kann ungünstige Auswahlen von Algorithmen bewirken. Daneben wird versucht, das spezifische Verhalten eines Algorithmus durch eine einzige Kennzahl abzubilden.

¹ Obwohl sich die Optimierungsparadigmen nur durch die Ressource *Information* unterscheiden, hat dies erhebliche Konsequenzen für die Bewertung der Güte von Algorithmen, da im Falle unvollständiger Information kein natürliches Optimalitätskonzept besteht.

Wir stellen deshalb zwei verteilungsbasierte Bewertungsmethoden vor, die sich für eine differenziertere Algorithmenanalyse durch Simulation in praktischen Anwendungen eignen. Zudem ermöglichen sie eine Beurteilung des Nutzens zusätzlicher Informationsvorausschau.

Verteilung des Performance Ratio für Algorithmenmengen

Es seien A und B Mengen von Algorithmen und σ eine Eingabefolge, dann bezeichnen wir

$$c_{\sigma}^{B/A} := \frac{\min_{ALG' \in B} ALG'(\sigma)}{\min_{ALG \in A} ALG(\sigma)}$$

als Performance Ratio von Algorithmenmenge B relativ zu Algorithmenmenge A bzgl. σ .

Der Wert $c_{\sigma}^{B/A}$ gibt an, welches Verhältnis zwischen den besten von B und A erzielbaren Zielfunktionswerten bei σ vorliegt. Beispielsweise kann man A als Menge von Online-Algorithmen ohne Lookahead und B als Menge von Online-Algorithmen mit Lookahead wählen. Liegt eine hinreichend große Menge an Eingabefolgen vor, so kann aus den Werten für $c_{\sigma}^{B/A}$ die zugehörige Verteilungsfunktion $F(c_{\sigma}^{B/A})$ erstellt werden (siehe Abbildung 3a)).

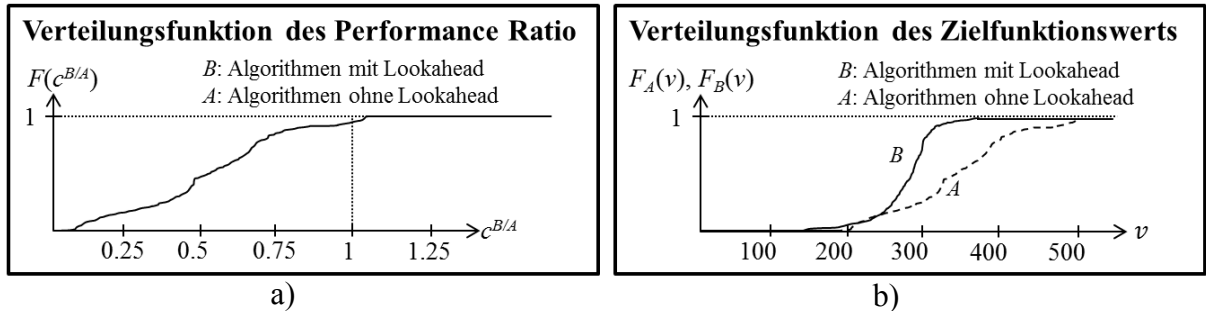


Abbildung 3: Verteilungsfunktionen des a) Performance Ratio und b) Zielfunktionswerts

Verteilung des Zielfunktionswerts für Algorithmenmengen

Sei A eine Algorithmenmenge und σ eine Eingabefolge, dann bezeichnen wir

$$v_{\sigma}^A := \min_{ALG \in A} ALG(\sigma)$$

als Zielfunktionswert der Algorithmenmenge A bzgl. σ . Der Wert v_{σ}^A gibt an, welcher Zielfunktionswert auf σ mit A bestenfalls erreicht wird. Liegt für zwei Algorithmenmengen A und B eine hinreichend große Menge an Eingabefolgen mit Zielfunktionswerten v_{σ}^A und v_{σ}^B vor, so lassen sich die beiden Verteilungsfunktionen $F_A(v)$ und $F_B(v)$ erstellen (siehe Abbildung 3b)).

Die Bewertungsmethoden lassen sich in einer Simulation bei Eingabefolgenerzeugung mittels stochastischer Komponenten für einen effizienten Vergleich von Algorithmen einsetzen.

3 Ein generisches Vorgehensmodell

Gemäß VDI-Richtlinie 3633 (siehe [6]) kann die Simulation als Basis zur Analyse unterschiedlicher Verfahren (Algorithmen) bei einer gegebenen Problemstellung dienen und somit wesentlich zur Auswahl eines passenden Algorithmus beitragen. Wir führen ein generisches Vorgehensmodell ein, das sich zur Modellierung von Lösungsverfahren für Online-Optimierungsprobleme mit Lookahead im Rahmen einer Simulation eignet und mittels generischer Programmierung implementiert sowie für beliebige Probleme instanziiert werden kann.

3.1 Modellierungsbausteine

Aufgrund der ständigen Bekanntgabe neuer Informationen betten wir das System in eine Zeitkomponente ein und bezeichnen die aktuelle Zeit mit t . Liegt keine Informationsvorausschau vor, so wird ein Eingabeelement genau zu dessen Bekanntgabezeitpunkt bekannt. Durch Lookahead wird der Bekanntgabezeitpunkt künstlich vorverlegt². Schließlich verlangen wir, dass ein Eingabeelement genau einmal bearbeitet wird, d.h. eine Aktion erfährt.

Eingabeelement / Eingabefolge: In der Eingabefolge $\sigma = (\sigma_1, \sigma_2, \dots, \sigma_n)$ werden die nacheinander eintreffenden Eingabeelemente festgehalten. Zusätzlich zur Spezifikation der σ_i gemäß der gewählten Lookaheadart assoziieren wir mit jedem σ_i zwei Bearbeitungsvariablen: $p_t(\sigma_i) \in \{\text{unbearbeitet}, \text{in Bearbeitung}, \text{bearbeitet}\}$ gibt den Bearbeitungsstatus von σ_i zum Zeitpunkt t an. $a_t(\sigma_i) = (w, \tau^s, \tau^e)$ gibt die für σ_i geplante Aktion (siehe unten) in Form der Aktionsinformation w , des vorgesehenen Startzeitpunkts τ^s und Endzeitpunkts τ^e an. Solange keine Aktion bestimmt wurde, ist $a_t(\sigma_i) = \text{null}$.

Lookahead Set: Das Lookahead Set L_t beinhaltet alle zum Zeitpunkt t bekannten Eingabeelemente mit Bearbeitungsstatus *unbearbeitet* oder *in Bearbeitung*. Welche Elemente konkret enthalten sind, hängt von der Lookaheadart ab.

Zustand / Zustandsraum: Der Zustandsraum S umfasst die Menge aller Konfigurationen, die das System erreichen kann. Ein Zustand $s \in S$ ist ein Tripel $s = (t_s, L_{t_s}, u_s)$, wobei t_s der Beobachtungszeitpunkt, L_{t_s} das Lookahead Set zu diesem Zeitpunkt und u_s die zusätzliche Zustandsinformation ist. Der Systemzustand ändert sich kontinuierlich. Um das System nur zu diskreten Zeitpunkten inspizieren zu müssen, führen wir Zielfunktionszustände ein.

Zielfunktionszustand / Zielfunktionszustandsraum: Wir extrahieren alle Informationen, die die Entwicklung des Zielfunktionswerts betreffen in den Elementen des Zielfunktionszustandsraums O . Ein Zielfunktionszustand $o \in O$ ist ein Paar $o = (v_o, r_o)$, wobei v_o der Zielfunktionswert und r_o die Zielfunktionszustandsinformation bei Beobachtung von o ist. Entscheidend ist die Annahme, dass sich der Zielfunktionszustand nur durch Beendigung der Bearbeitung eines Eingabeelements und somit nur zu diskreten Zeitpunkten verändern darf.

Aktion / Aktionsraum: Der Aktionsraum A vereinigt die Menge aller Aktionen, die bei der Bearbeitung eines Eingabeelements durchgeführt werden können mit der *null*-Aktion (d.h. noch keine Aktion vorgesehen). Eine Aktion $a \in A$ mit $a \neq \text{null}$ ist ein Tripel $a = (w, \tau^s, \tau^e)$, wobei w der Aktionsinformation und τ^s bzw. τ^e dem Start- bzw. Endzeitpunkt entspricht. Für $t < \tau^s$ gibt a die *geplante* Aktion an und kann durch einen Algorithmus bis τ^s verändert werden.

Übergangsfunktion des Zielfunktionszustands: Die Übergangsfunktion $f: S \times O \rightarrow O$ des Zielfunktionszustands bestimmt ausgehend vom aktuellen Zustand $s \in S$ und Zielfunktionszustand $o \in O$ den Nachfolger des Zielfunktionszustands o , der durch Abarbeitung der zu einem Eingabeelement aus L_{t_s} gehörenden Aktion a entsteht. Somit ändert sich der

² In einem Materialflusssystem erreicht man dies z.B. durch Vorversetzen eines Informationslesegeräts.

Zielfunktionszustand nur nach beendeter Bearbeitung eines Eingabeelements, d.h. zu diskreten Zeitpunkten und eine Auswertung von f muss nur zu diesen Zeitpunkten erfolgen.

Algorithmus: Einen Algorithmus ALG verstehen wir als zielfunktionsorientierte Abfolge von Berechnungen, die der Bestimmung der (geplanten) Aktionen für die einzelnen Eingabeelemente im Lookahead Set dient. Für n im Lookahead Set befindliche Eingabeelemente ist $\text{ALG}: S \times O \rightarrow A^n \setminus \{\text{null}\}^n$ eine Funktion, die in Abhängigkeit des aktuellen Zustands $s \in S$ und Zielfunktionszustands $o \in O$ den n Elementen im Lookahead Set je eine Aktion zuordnet.

3.2 Vorgehensmodell

Das generische Vorgehensmodell für Online-Optimierungsprobleme mit Lookahead beschreibt die Interaktion der vorgestellten Modellierungsbausteine im Zeitverlauf. Initiierend hierfür sind (neben der unbeeinflussbaren Bekanntgabe der Eingabeelemente) die von ALG bestimmten Aktionen für die Eingabeelemente im Lookahead Set. Deshalb muss die Frage beantwortet werden, wann ALG ausgewertet werden soll. Geht man davon aus, dass eine Auswertung bei unverändertem Lookahead Set zu derselben Aktionenmenge für alle Eingabeelemente im Lookahead Set führt, so ist genau dann eine Auswertung vorzunehmen, wenn sich das Lookahead Set ändert. Dies kann entweder durch beendete Bearbeitung eines Eingabeelements oder durch Kenntnisaufnahme über ein neues Eingabeelement geschehen. Das Verfahrensmodell kümmert sich um die *Steuerung* des Verfahrensablaufs und durchläuft iterativ folgende Schritte (vgl. Abbildung 4):

1. Das System befindet sich im Zustand $s \in S$ und im Zielfunktionszustand $o \in O$.
2. Algorithmus ALG bestimmt für alle Eingabeelemente aus dem Lookahead Set L_{t_s} die Aktionen (auch *null*-Aktionen möglich).
3. Veränderung des Lookahead Sets L_{t_s}
 - a. Wird die Bearbeitung eines Eingabeelements beendet, so wird mit der Übergangsfunktion des Zielfunktionszustands f der neue Zielfunktionszustand $o \in O$ bestimmt, der Systemzustand s wird aktualisiert und es wird mit 1. begonnen.
 - b. Wird ein neues Eingabeelement zur Kenntnis genommen, so wird der Systemzustand s aktualisiert und es wird mit 1. begonnen.

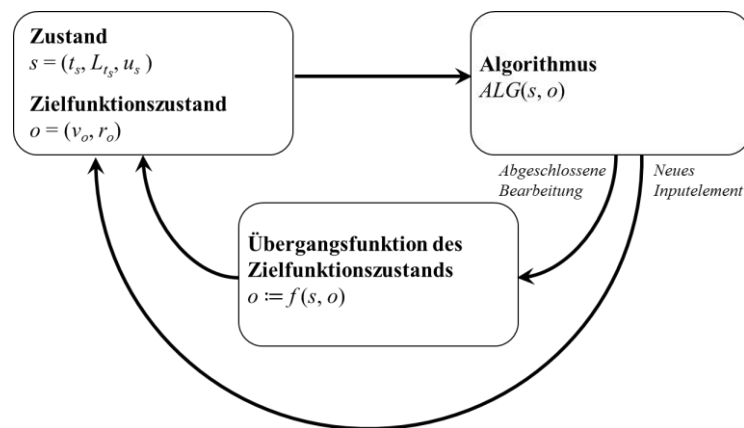


Abbildung 4: Generisches Vorgehensmodell für Online-Optimierungsprobleme mit Lookahead

3.3 Anwendungen

Die nächsten beiden Abschnitte zeigen zwei Optimierungsprobleme für die das generische Vorgehensmodell zu Simulations- und Analysezwecken instanziiert wurde.

3.3.1 Online Bin Packing mit Lookahead

Die Offline-Variante des in Abschnitt 2.1 eingeführten Bin Packing Problems ist NP-schwer³. Dennoch lässt sich der Einsatz adaptierter Methoden zur Lösung des Offline-Problems für die Online- und Request-Lookahead-Version ($K \geq 1$) rechtfertigen, v.a. da aufgrund der durch K gegebenen fixen Problemgrößen polynomialer Aufwand folgt. Bekannte Algorithmen sind:

- FIRST FIT: Ordne die K bekannten Gegenstände nach absteigender Größe und packe falls möglich den größten Gegenstand in den ersten bereits benutzten Behälter, in den er passt; andernfalls lege den Gegenstand in einen neuen Behälter.
- BEST FIT: Ordne die K bekannten Gegenstände nach absteigender Größe und packe falls möglich den größten Gegenstand in den vollsten bereits benutzten Behälter, in den er passt; andernfalls lege den Gegenstand in einen neuen Behälter.
- IP SOLVE: Formuliere für die K bekannten Gegenstände und bereits benutzten Behälter ein ganzzahliges Optimierungsproblem und löse es mit Integer Programming (IP).

Die Instanziierung der Modellierungsbausteine erfolgt ausgehend von Gegenständen als Eingabeelementen. Dabei werden Behälterkonfigurationen im Zielfunktionszustand abgebildet. Eine Aktion gibt den Zielbehälter sowie den Packzeitpunkt eines Gegenstands an. Das Vorgehen eines Algorithmus ergibt sich in Abhängigkeit der bekannten Gegenstände (Lookahead Set) und der Behälterkonfiguration (Zielfunktionszustand). Nach Packen aller Gegenstände liest man am Zielfunktionszustand ab, wie viele Gegenstände benötigt wurden.

3.3.2 Online Traveling Salesman Probleme mit Lookahead

Das Traveling Salesman Problem (TSP) ist Bestandteil vieler Routingprobleme. Es besteht aus n von einem Server zu besuchenden Punkten in einem (metrischen) Raum und einem Ursprung. Ziel ist es, die n Besuchspunkte auf einer möglichst kurzen Route vom Ursprung ausgehend und zu diesem zurückkehrend zu besuchen. Als Informationsvorausschau lässt sich sowohl das Konzept des Request Lookahead als auch das des Time Lookahead einsetzen. Das TSP ist in seiner Offline-Variante NP-schwer⁴. Dennoch bietet sich der Einsatz von an das Online-Problem mit Lookahead angepassten Offline-Lösungsverfahren an:

- NEAREST NEIGHBOR: Der Server besucht als nächstes denjenigen Punkt aus dem Lookahead Set, der am nächsten liegt.
- INSERTION: Der Server besucht als nächstes den ersten Punkt des Hamiltonpfades, der durch sukzessives Einfügen der Punkte aus dem Lookahead Set in eine zum Ursprung führende Tour entsteht.

³ Dies sieht man durch Reduktion des als NP-vollständig bekannten Problems *Partition*.

⁴ Dies sieht man, da es das als NP-vollständig bekannte *Hamiltonpfadproblem* zur Bestimmung eines Weges, der alle Knoten besucht, enthält.

- 2-OPT / 3-OPT: Der Server besucht als nächstes den ersten Punkt des zum Ursprung führenden Hamiltonpfades, der ausgehend von einem Hamiltonpfad der Punkte aus dem Lookahead Set durch Vertauschen von Kantenpaaren bzw. Kantentripeln entsteht.
- SIMULATED ANNEALING: Stochastisches 2-OPT-Verfahren.
- IP SOLVE: Der Server besucht als nächstes den ersten Punkt des zum Ursprung führenden Hamiltonpfades der Punkte aus dem Lookahead Set, der durch Lösung eines ganzzahligen Optimierungsproblems bestimmt wird.

Die Instanziierung der Modellierungsbausteine erfolgt ausgehend von Besuchspunkten als Eingabeelementen. Da das System durch die Serverbewegung dynamisch ist, wird in der zusätzlichen Zustandsinformation die Serverposition notiert. Ein Zielfunktionszustand speichert den zuletzt besuchten Punkt samt Besuchszeitpunkt sowie den erwarteten Rückkehrzeitpunkt zum Ursprung. Eine Aktion gibt den nächsten Besuchspunkt an und spezifiziert Anfahrtsbeginn und -ende. Ausgehend von den bekannten Besuchspunkten (Lookahead Set), der Serverposition (zusätzliche Zustandsinformation) und dem zuletzt besuchten Ort (Zielfunktionszustand) bestimmt ein Algorithmus den nächsten Besuchspunkt.

4 Numerische Ergebnisse

Abschließend betrachten wir den Einsatz des Vorgehensmodells im Bin Packing und Traveling Salesman Problem. Zusammen mit den Bewertungskonzepten lassen sich Aussagen über den Nutzen von Lookahead in unterschiedlichen Problemstellungen gewinnen. Die Experimente wurden mit 1.86 GHz Prozessor und 2 GB Arbeitsspeicher durchgeführt. Algorithmen wurden in C++ implementiert, Optimierungsprobleme mit CPLEX 12.2 gelöst.

4.1 Online Bin Packing mit Request Lookahead

Es wurden je 100 Eingabefolgen mit 25 bzw. 1000 Gegenständen zufälliger Größe in $(0, 1]$ erzeugt und mit den Algorithmen getestet⁵. Einen ersten Eindruck über den Nutzen von zusätzlichem Lookahead gibt die durchschnittlich benötigte Behälteranzahl in Abbildung 5:

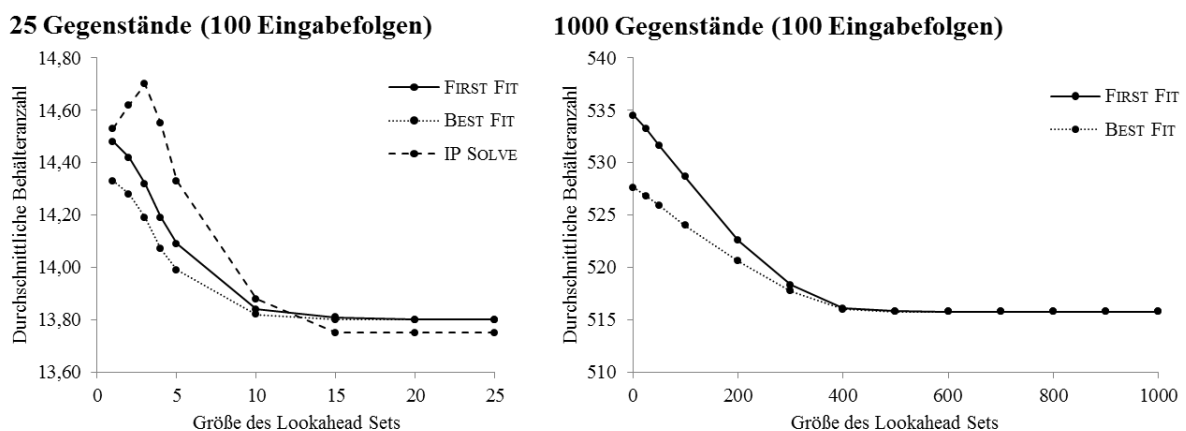


Abbildung 5: Durchschnittliche Behälteranzahl im Bin Packing

⁵ Für 25 Gegenstände können die ganzzahligen Optimierungsprobleme bei IP SOLVE mit CPLEX 12.2 in akzeptabler Zeit gelöst werden. Bereits für 50 Gegenstände kann dies nicht mehr garantiert werden.

Zusätzliche Lookaheadeinheiten führen bei allen Algorithmen tendenziell zu einer geringeren Anzahl benötigter Behälter. Die Verringerung pro Lookaheadeinheit ist jedoch äußerst gering, so dass sich im Vergleich von reiner Online- und Offline-Situation eine Reduktion von weniger als 5% ergibt. Das aufwändige Verfahren IP SOLVE ist erst ab einem Schwellwert für die Lookaheadgröße vorteilhaft. Die „lokale“ Optimalität der Teillösungen hat somit bei kleinem Lookahead und Einsatz von IP SOLVE nicht die Eigenschaft sich auf das „globale“ Ergebnis, das nach Packen des letzten Gegenstands erhalten wird, zu übertragen.

Eine detailliertere Darstellung erhält man durch Verwendung von Verteilungsfunktionen für Algorithmenmengen. Hierzu bezeichnen wir mit ALG_K die Algorithmenmenge mit Lookaheadgröße K . Abbildung 6 zeigt die Verteilungsfunktionen des Performance Ratio für Algorithmenmengen der Lookaheadgrößen $K \in \{25, 50, 100, 200, \dots, 1000\}$ relativ zu reinen Online-Algorithmen ($K = 1$) und des Zielfunktionswerts für dieselben Lookaheadgrößen:

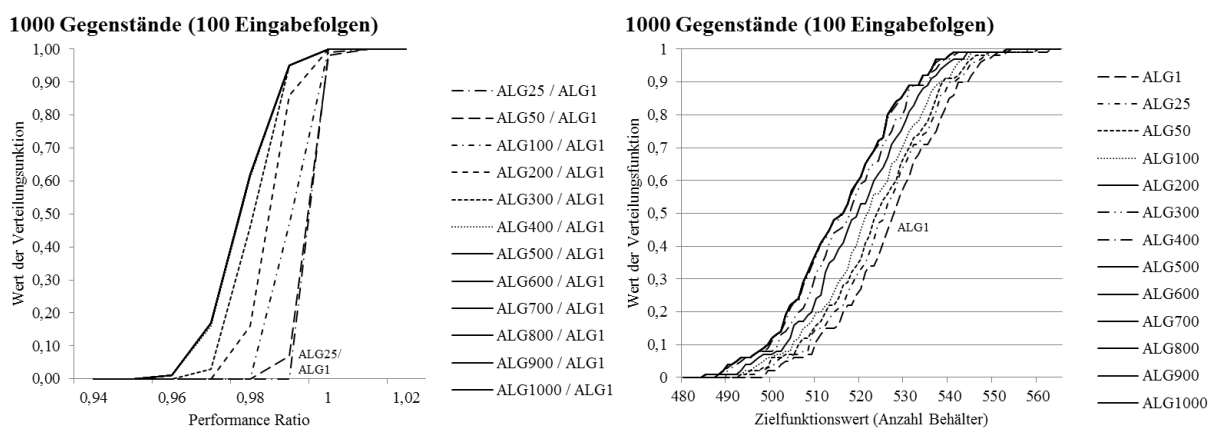


Abbildung 6: Verteilungsfunktion von Performance Ratio / Zielfunktionswert im Bin Packing

Die Verteilungsfunktionen der Algorithmenmengen mit $K \in \{500, 600, \dots, 1000\}$ fallen zusammen. Ein zusätzlicher Nutzen von Informationsvorausschau ist in dieser Größenordnung nicht mehr vorhanden. Für $K < 500$ zeigt sich eine Reihung (Dominanz) der Verteilungsfunktionen, die den positiven Effekt des Lookaheads bestätigt. Da sämtliche Performance Ratios nahe bei 1 liegen, fällt der Lookaheadeffekt jedoch gering aus. Die Graphen sind in einem beschränkten Bereich relativ steil, d.h. die Verteilungen haben eine geringe Varianz. Jede Lookaheadgröße korreliert also mit einem charakteristischen Bereich des Performance Ratios bzw. des Zielfunktionswerts.

Fazit: Im Bin Packing ist eine Reduktion der benötigten Behälteranzahl durch zusätzliche Lookaheadeinheiten gegeben; der Effekt ist vergleichsweise gering und liegt bei weniger als 5%. Exakte Verfahren zur Lösung der Teilprobleme lohnen sich erst bei großem Lookahead.

4.2 Online Traveling Salesman Problem mit Request Lookahead

Vergleichend untersuchen wir das Online TSP mit Request Lookahead. Es wurden je 100 Eingabefolgen mit 25 bzw. 100 zufälligen Besuchspunkten in $[0, 1] \times [0, 1] \subset \mathbb{R}^2$ erzeugt und mit den Algorithmen getestet. Die durchschnittliche Tourlänge bei unterschiedlichen Lookaheadgrößen in Abbildung 7 gibt Aufschluss über mögliche Reduktionen der Tourlänge:

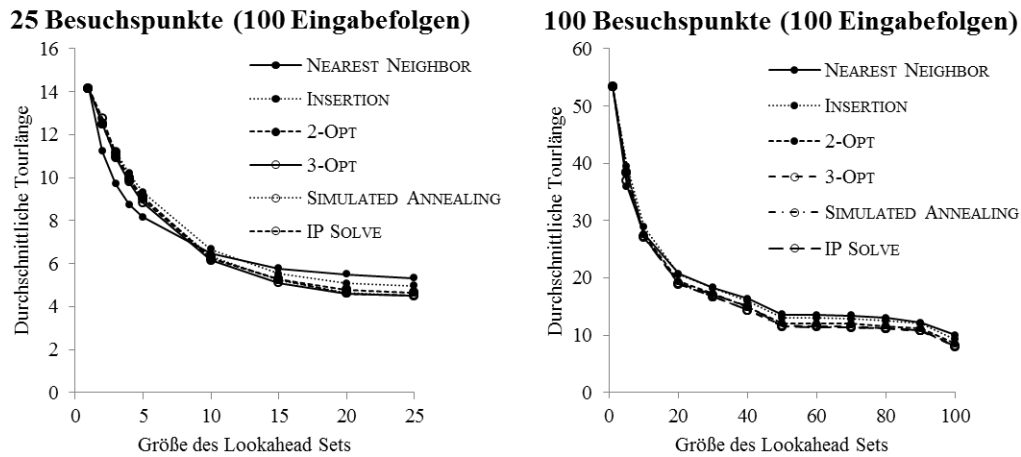


Abbildung 7: Durchschnittliche Tourlänge im Traveling Salesman Problem

Informationsvorausschau bewirkt eine deutliche Verringerung der Tourlänge: So beträgt die Tourlänge bei einem Lookahead von $K = 5$ nur noch ca. 70% der Tourlänge aus dem reinen Online-Fall. Für Lookaheadgrößen nahe der Gesamtanzahl an Besuchspunkten (Offline-Fall) ergeben sich durchschnittliche Tourlängen von ca. 35% (25 Besuchspunkte) bzw. 12% (100 Besuchspunkte) der ursprünglichen Strecke. Der Grenznutzen pro Lookahead-einheit ist monoton fallend. Erstaunlicherweise treten die positiven Effekte bei allen Verfahren in gleichem Maß auf. Entgegen der Erwartung bringen komplexe Verfahren zunächst keine besseren Ergebnisse als einfache Verfahren. Grund ist, dass z.B. bei NEAREST NEIGHBOR eine Stabilität in dem Sinne erzeugt wird, dass Sprünge zwischen Besuchspunkten unterschiedlicher Regionen vermieden werden. Da bei komplexen Verfahren die „lokalen“ Probleme der einzelnen Zeitstufen ohne Rücksicht auf das „globale“ Endergebnis gelöst werden, stellen sich die erhofften Effekte erst bei großem Lookahead ein.

Abbildung 8 zeigt die Verteilungsfunktionen des Performance Ratio für Algorithmenmengen der Lookaheadgrößen $K \in \{5, 10, 20, \dots, 100\}$ relativ zu reinen Online-Algorithmen ($K = 1$) und des Zielfunktionswerts für dieselben Lookaheadgrößen:

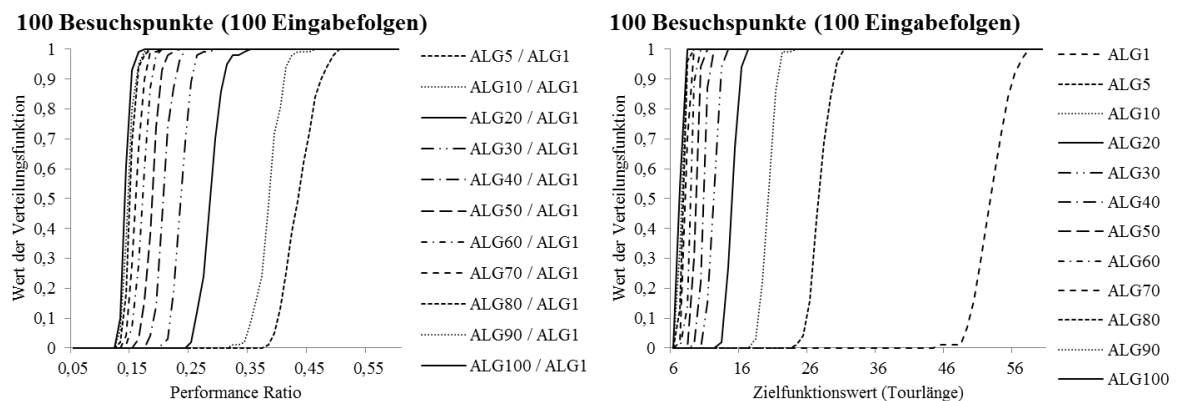


Abbildung 8: Verteilungsfunktion von Performance Ratio / Zielfunktionswert im TSP

Die Schaubilder zeigen die zu erwartende Reihung (Dominanz) der Verteilungsfunktionen. Aufgrund des horizontalen „Abstands“ der Kurven zueinander ist der

Grenznutzen pro Lookaheadeinheit fallend. Der auf einen kleinen Bereich beschränkte steile Verlauf der Verteilungsfunktionen zeigt, dass der entsprechende Nutzenzuwachs charakteristisch für die Optimierungsgüte der Algorithmen bei gegebenem Lookahead ist.

Fazit: Im Vergleich zum Bin Packing ergibt sich beim TSP ein weitaus stärkerer positiver Effekt durch Informationsvorausschau. Exakte Verfahren lohnen sich erneut nur bei größerem Lookahead. Die Ergebnisse beider Problemstellungen verdeutlichen, dass die Sensitivität von Algorithmen bzgl. der Lookaheadgröße in hohem Maß von der Problemstellung selbst abhängt und nicht im Voraus angenommen werden darf.

5 Fazit und Ausblick

Mit dem generischen Vorgehensmodell für Online-Optimierungsprobleme mit Lookahead steht ein Modellierungsframework zur Analyse von Algorithmen in unterschiedlichen Problemstellungen zur Verfügung. Durch verteilungsbasierte Bewertung mit Performance Ratio und Zielfunktionswert lassen sich in Anwendungen praktische Aussagen über typische Verfahrensabläufe und die Eignung bestimmter Algorithmen gewinnen. Diese Ergebnisse können für verschiedene Lookaheadgrößen Aufschluss über den Nutzen der bereitgestellten Informationsvorausschau geben und als Entscheidungshilfe für zu ergreifende Installationsmaßnahmen von Systemen zur Informationsvorausschau dienen.

Eine weitere Untersuchung und mathematische Analyse des Vorgehensmodells selbst ist für die Zukunft wünschenswert. Einen ersten Ansatzpunkt liefern Markov-Ketten, die das zeitliche Verhalten eines diskreten stochastischen Prozesses in ähnlicher Weise nachbilden.

Literatur

- [1] Albers, Susanne: *On the influence of lookahead in competitive paging algorithms*, Algorithmica 18, Springer, 1998
- [2] Angelopoulos, Spyros Angelopoulos und Schweitzer, Pascal: *Paging and List Update under Bijective Analysis*, Proceedings of the 20th ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 2009
- [3] Ausiello, Giorgio et al.: *On-Line Algorithms, Real Time, the Virtue of Laziness, and the Power of Clairvoyance*, Lecture Notes in Computer Science: Theory and Applications of Models of Computation, Springer, 2006
- [4] Borodin, Allan und El-Yaniv, Ran: *Online-Computation and Competitive Analysis*, erste gebundene Ausgabe, Cambridge University Press, 2005
- [5] Lavrov, Alexander und Nickel, Stefan: *Simulation und Optimierung zur Planung von Kommissionierungssystemen*, VDI-Seminar, 2005
- [6] VDI-Richtlinie 3633: *Blatt 1 – Simulation von Logistik-, Materialfluss- und Produktionssystemen – Grundlagen*, VDI-Handbuch Materialfluss und Fördertechnik 8, Beuth, 2008

Virtuelle Simulationsmodelle und ein Devirtualisierungsvorgang der Erstellung von parallelen Simulatoren für dynamische Netzobjekte mit verteilten Parametern

V. Svjatnyj, V. Kushnarenko

Nationale technische Universität Donezk (DonNTU), Fakultät für Computerwissenschaften und Technologien (FCWT), Artemstraße 58, 83000 Donezk, Ukraine, svjatnyj@cs.dgtu.donetsk.ua

1. Einführung

Die technische und verfahrenstechnische Netze sind in mehreren Gegenstandsgebiete die Objekte der Untersuchung, Projektierung, Automatisierung, Überwachung, Qualitätssicherung, optimalen Prozessführung, Sicherheitsanalyse, Prozessvorhersage, Vermeidung der sicherheitskritischen Betriebszustände und der Havarieliquidierung. Die dynamische Netzobjekte mit verteilten Parametern (DNOVP) gehören zu den komplexen (oft sicherheitskritischen) Systemen. Nichtlinearität der prozessbeschreibenden Funktionen, räumliche Verteilung von Prozessparametern, große, im Laufe des Objektbetriebs sich entwickelnde Dimensionen der Netze, ebenso mehrere aktive Elemente mit nichtlinearen stromabhängigen Charakteristiken, wesentliches mehrfaches und hierarchisches Zusammenwirken der regelbaren Parametern sowie gleichzeitige Einflüsse von deterministischen und stochastischen Störungen sind die Hauptmerkmale dieser Komplexität. Als Folge von steigenden Anforderungen an die Qualität, Sicherheit und Wirtschaftlichkeit der DNOVP-Prozesse werden immer genauere und damit auch komplexere Simulationsmodelle benötigt. Bei dieser Entwicklung ergibt sich eine Komplexitätsschwelle für die Implementierung und Anwendung der Simulationsmodelle, die nur durch den Einsatz von parallelen Simulationswerkzeugen mit der vollfunktionellen benutzerfreundlichen Simulationssoftware überwunden werden kann. Deshalb haben die Methoden und Mittel der parallelen Modellierung und Simulation dieser Objektklasse sowohl bei der Projektierung als auch während des Betriebes eine zunehmende theoretische und praktische Bedeutung. Die in dem Beitrag betrachteten Ansätze zur Analyse von virtuellen parallelen DNOVP-Simulationsmodellen und zur Entwicklung von parallelen DNOVP-Simulatoren dienen der effizienten Nutzung der MIMD-Systeme im Rahmen einer anwendungsorientierten parallelen Simulationsumgebung und stehen im Mittelpunkt einer engen Kooperation zwischen der FCWT und dem HLRS auf dem Gebiet von parallelen Simulationstechnik (ParSimTech) [1].

2. DNOVP als Objekte der Modellierung und Simulation

Als DNOVP-Beispiel betrachten wir ein Grubenbewetterungsnetz. Topologisch wird DNOVP als Graph $G(m, n)$ dargestellt und durch Tabelle 1 mit m Zeilen und $s+5$ Spalten kodiert. Hier sind: **QJ** – ein Luftstrom in der **J**-Kante; **AKI** und **EKK** – Anfangs- und Endknoten der **J**-Kante; $I \in (1, 2, \dots, n)$, $J \in (1, 2, \dots, m)$; **PAR(PJ₁, PJ₂, ..., PJ_s)** – Menge von s Parametern **PJ** der Kante; **AEJ** – aktives Element in der **J**-Kante; **VECOMJ** – Kommentar zur technologische Funktion der **J**-Kante.

Tabelle 1 – Kodierung des Graphen

AKI	EKK	QJ	PAR(PJ₁, PJ₂, ..., PJ_s)	AEJ	VECOMJ
------------	------------	-----------	---	------------	---------------

Modell der dynamischen Prozesse in der j -Kante ohne Luftverlusten durch die Wände wird von der Gleichungen

$$\begin{cases} -\frac{\partial P_j}{\partial \xi} = r_j Q_j^2 + \frac{\rho}{F_j} \frac{\partial Q_j}{\partial t} + r_j(\xi_r, t) Q_j^2 \\ -\frac{\partial P_j}{\partial t} = \frac{\rho a^2}{F_j} \frac{\partial Q_j}{\partial \xi}, \end{cases} \quad (1)$$

beschrieben. Hier sind: P_j , Q_j – Druck und Luftstrom der Koordinate ξ entlang, die von AKI- bis zum EKK-Knoten errechnet wird ; r_j – spezifischer aerodynamischer Widerstand; F_j – die Querschnittsfläche der Kante (Luftwegstrecke); ρ – Luftdichte; a – die Schallgeschwindigkeit im Luft; $r_j(\xi_r, t)$ – regelbarer Widerstand; ξ_r – die Ortskoordinate des regelbaren Widerstands (z. B., ein Schieber).

Die Randbedingungen für (1) sind die Druckfunktionen P_{AKI} , P_{EKK} in den Knoten der j -Kante. Es sind drei Kanten- und Knotenarten nach Randbedingungen in DNOVP zu unterscheiden:

- die Kanten, die den inneren n_1 DNO-Knoten inzident sind; hier werden die Druckwerten während des Lösen des DNOVP-Gleichungssystems entsprechend den dynamischen Knotenbedingungen

$$-\frac{\partial P_{wi}}{\partial t} = \frac{\rho a^2}{F_{wi}} \frac{\partial Q_{wi}}{\partial \xi} \quad (2)$$

berechnet; hier sind P_{wi} – Druck im WI-Knoten; Q_{wi} – Gesamtluftstrom durch WI-Knoten; F_{wi} – Querschnittsfläche des Knotenraums;

- die Kanten, die den n_2 Knoten der Ventilatorenanschlüsse inzident sind; hier wird Druck als die Ventilatorcharakteristik vorgegeben

$$P_{wi} = P_{AEJ}(QJ); \quad (3)$$

- die Kanten, die den n_3 Knoten der Atmosphäreanschlüsse inzident sind:

$$P_{wi} = P_{ATM} = const. \quad (4)$$

DNOVP hat insgesamt

$$n = n_1 + n_2 + n_3 \quad (5)$$

Knoten und entsprechend die n Randbedingungen.

Die Anfangsbedingungen sind

$$P_j(\xi, 0), Q_j(\xi, 0) \quad (j = 1, 2, \dots, m). \quad (6)$$

Problemstellung: für DNOVP, dessen Graph wird mit der topologischen Tabelle 1 kodiert und jede Kante wird mit den Gleichungssysteme (1) und Randbedingungen (2), (3), (4) bei den Anfangsbedingungen (6) beschrieben, sollen die parallelen algorithmischen, hardware- und softwaretechnischen Simulationsmitteln entwickelt und implementiert werden, die adequat die dynamischen Prozesse $P_j(\xi, t)$, $Q_j(\xi, t)$ ($j = 1, 2, \dots, m$) mit der Berücksichtigung von definierten Arbeitsbedingungen, Störungen und Regelungen von Luftströmen widerspiegeln.

Die Simulationsmodelle der j -Kante und des DNOVP. Durch Approximation der Gleichungen (1) nach **Linienverfahren** mit der Ortschaftweite $\Delta\xi$ erhalten wir für k -Element der j -Kante das Gleichungssystem:

$$\begin{aligned} -\frac{P_{jk} - P_{j,k+1}}{\Delta\xi_{jk}} &= \frac{\rho}{F_{jk}} \frac{dQ_{jk}}{dt} + r_{jk} Q_{jk} |Q_{jk}| + r_{jk}(\xi_p, t) Q_{jk} |Q_{jk}|, \\ -\frac{Q_{jk} - Q_{j,k+1}}{\Delta\xi_{jk}} &= \frac{F_{jk}}{\rho a^2} \frac{dP_{j,k+1}}{dt}. \end{aligned} \quad (7)$$

Die inneren Randbedingungen von Type (2) werden von Gleichung

$$-\frac{dP_{wi}}{dt} = \frac{\rho a^2}{F_{wi}} \frac{Q_{jk} - \sum_{jwi} (Q_{jwi1} - Q_{jwiMj})}{\Delta\xi_{jk}} \quad (8)$$

approximiert. Hier sind: $P_{wi} = P_{jMj+1}$ – Druck im Endknoten des letzten Elementes $Q_{jk} = Q_{jMj}$ des J -Kantestroms, der in wi -Knote fließt; $jwiMj$ – die Kantenummern aus Menge $j=1, 2, \dots, m$, die dem Knoten wi inzident sind; dabei $jwi1$ ist erstes Element der j -Kante mit den wi als Anfangsknoten (Ausfluß), während $jwiMj$ letztes Element mit den wi als Endknoten (Zufluß) ist. Jede Kante wird bei der Approximation nach M_j Elementen $Q_{j1}, \dots, Q_{jMj}$ zerteilt. Dabei wird die Nummerierung der Druckwerten als $P_{j1}, P_{j2}, \dots, P_{jMj+1}$ erfolgt. Es ist wichtig zu erwähnen, daß j -Kante eine Anfangsknote wi mit dem Druck $P_{wi}=P_{j1}$ und die Endknote $wi+b$ ($b = \text{const}$) mit dem Druck $P_{wi+b} = P_{jMj+1}$ hat.

Für dynamisches Netzobjekt mit verteilten Parametern wird jede Kante nach obige Approximation durch zwei Vektoren Q_j, P_j ($j = 1 \dots m$) präsentiert:

$$Q_j = (Q_{j1}, Q_{j2}, \dots, Q_{jMj})^T \quad (9)$$

ist Luftstrom in j -Kante,

$$P_j = (P_{j1}, P_{j2}, \dots, P_{jMj+1})^T \quad (10)$$

ist Druck in j -Kante. Dabei wird M_j – die Elementenmengen in den Kanten – abhängig von der Kantenlängen l_j bei der gleichen Ortschaftweite $\Delta\xi$ für ganzes DNOVP als

$$M_j = l_j / \Delta\xi \quad (11)$$

berechnet. Ortsdiskretisiertes Modell der j -Kante beinhaltet M_j Gleichungssysteme (7) bei $k=1, 2, \dots, M_j$. Für die numerische Berechnung der Vektorenkomponenten in (9) und (10) werden die Gleichungen (7) zur Form des ortsdiskretisierten Simulationsmodells umgewandelt:

$$\begin{cases} \dot{Q}_{jk} = \alpha_j (P_{jk} - P_{j,k+1}) - \beta_j Q_{jk} |Q_{jk}| - \beta_{rj} Q_{jk} |Q_{jk}| \\ \dot{P}_{j,k+1} = g_j (Q_{jk} - Q_{j,k+1}) \end{cases} \quad (12)$$

Die $\alpha_j, \beta_j, \beta_{rj}, g_j$ sind die von aerodynamischer j -Kanteparametern abhängigen Koeffizienten.

Bei der Entwicklung des Ortsdiskretisierten Simulationsmodells des ganzen Netzobjektes sollen wir m Gleichungssysteme von Type (12) für alle Kanten, d. h. $j=1,2,...,m$, darstellen:

$$\begin{aligned}
 & \begin{cases} \dot{Q}_{1k} = \alpha_1(P_{1k} - P_{1,k+1}) - \beta_1 Q_{1k} |Q_{1k}| - \beta_{r1} Q_{1k} |Q_{1k}| \\ \dot{P}_{1,k+1} = g_1(Q_{1k} - Q_{1,k+1}) \end{cases} \\
 & k=1,2,...,M_1 \\
 & \\
 & \begin{cases} \dot{Q}_{mk} = \alpha_m(P_{mk} - P_{m,k+1}) - \beta_m Q_{mk} |Q_{mk}| - \beta_{rm} Q_{mk} |Q_{mk}| \\ \dot{P}_{m,k+1} = g_m(Q_{mk} - Q_{m,k+1}) \end{cases} \\
 & k=1,2,...,M_m
 \end{aligned} \tag{13}$$

Das Netzobjekt hat $\mathbf{n} = \mathbf{n}_1 + \mathbf{n}_2 + \mathbf{n}_3$ Randbedingungen (2), (8), (3), (4). Entsprechend der Gleichung (8) formulieren wir $\mathbf{n}_1$ Randbedingungen für inneren Knoten des Netzobjektes ($wi=1, 2,...,n_1$):

$$\begin{aligned}
 & \frac{dP_{w1}}{dt} = \frac{\rho a^2}{F_{w1}} \frac{Q_{jk} - \sum_{jw1} (Q_{jw11} - Q_{jw1Mj})}{\Delta \xi_{jk}} \\
 & \\
 & \frac{dP_{wn1}}{dt} = \frac{\rho a^2}{F_{wn1}} \frac{Q_{jk} - \sum_{jwn1} (Q_{jwn11} - Q_{jwn1Mj})}{\Delta \xi_{jk}}
 \end{aligned} \tag{14}$$

Die Ventilatorencharakteristiken bilden nach (3) $\mathbf{n}_2$ weiteren Randbedingungen

$$\begin{aligned}
 & P_{wi} = P_{AEJ}^1(QJ) \\
 & \\
 & P_{wi} = P_{AEJ}^{n_2}(QJ).
 \end{aligned} \tag{15}$$

Dabei ist $wi=n_1+1, n_1+2, ..., n_1+n_2$, wenn die Numerierung der Knoten fortlaufend von inneren Knoten durchgeführt wird. Aktive Elemente (Ventilatoren) sind in (15) von 1 bis $\mathbf{n}_2$ zusätzlich numeriert.

Für $\mathbf{n}_3$ Knoten der Atmosphäreanschlüsse gelten die Randbedingungen nach (4):

$$\begin{aligned}
 & P_{wi} = P_{(n_1+n_2)+1} = P_{ATM}^1 = const, \\
 & \\
 & P_{wi} = P_n = P_{ATM}^{n_3} = const.
 \end{aligned} \tag{16}$$

Bei obiger Nummerierung ist in (16) $wi=(n_1+n_2)+1, ..., (n_1+n_2)+(n_3-1)$, n .

Die Gleichungen (13) bilden zusammen mit den Randbedingungen (14), (15), (16) das ortsdiskretisierte DNOVP-Simulationsmodell:

$$\begin{aligned}
 & \left\{ \begin{aligned} \dot{Q}_{11} &= \alpha_1(P_{11} - P_{12}) - \beta_1 Q_{11}|Q_{11}| - \beta_{r1} Q_{11}|Q_{11}| \\ \dot{P}_{12} &= g_1(Q_{11} - Q_{12}) \end{aligned} \right. \quad \dots \quad \left\{ \begin{aligned} \dot{Q}_{1M1} &= \alpha_1(P_{1M1} - P_{1M1+1}) - \beta_1 Q_{1M1}|Q_{1M1}| - \beta_{r1} Q_{1M1}|Q_{1M1}| \\ \dot{P}_{1M1+1} &= g_1(Q_{1M1} - Q_{1M1+1}) \end{aligned} \right. \\
 & j=1, \quad k=1, \dots, M_1 \\
 & \dots \dots \dots (17) \\
 & \left\{ \begin{aligned} \dot{Q}_{m1} &= \alpha_m(P_{m1} - P_{m,2}) - \beta_m Q_{m1}|Q_{m1}| - \beta_{rm} Q_{m1}|Q_{m1}| \\ \dot{P}_{m,2} &= g_m(Q_{m1} - Q_{m,2}) \end{aligned} \right. \quad \dots \quad \left\{ \begin{aligned} \dot{Q}_{mk} &= \alpha_m(P_{mk} - P_{mk+1}) - \beta_m Q_{mk}|Q_{mk}| - \beta_{rm} Q_{mk}|Q_{mk}| \\ \dot{P}_{mk+1} &= g_m(Q_{mk} - Q_{mk+1}) \end{aligned} \right. \\
 & j=m, \quad k=1, \dots, M_m
 \end{aligned}$$

Wir werden zwischen den Luftstrom(Q)- und Druck(P)-Gleichungen unterscheiden. Die gesamte Menge der zu lösenden Q-Gleichungen in (13) ist

$$N_{QGI} = \sum M_j, \quad 1 \leq J \leq m.$$

Aus (13) und (14) folgt, dass Menge der P-Gleichungen pro Kante $M_j - 2$ ist, weil die zwei Randknotendrucke werden nach (14) berechnet. Deshalb

$$N_{PGI} = \sum (M_j - 2) + n_1, \quad 1 \leq J \leq m.$$

Für die industrienahen DNOVP ($m \geq 1000$, $n \geq 300$, $M_j \geq 50$) geht es um $(N_{QGI} + N_{PGI}) \geq 10^5$, deshalb ist aktuell die rechnergestützte Erstellung der DNOVP-Simulationsmodelle mit Hilfe des Topologieanalysators und des Gleichungsgenerators zu realisieren [2].

3. Die virtuelle parallele DNOVP-Simulationsmodelle

Virtuelles paralleles DNOVP-Simulationsmodell (VPSM) ist eine Abstraktion, die aus dem Gleichungssystem (14)...(17) und der entsprechend einem Parallelisierungsansatz erstellten Struktur von virtuellen MIMD-Prozessen besteht. Virtueller MIMD-Prozess ist ein weitgehend autonomes Programm, das dem Lösungsalgorithmus des Gleichungssystemteils entspricht und über eine Schnittstelle mit den benachbarten Prozessen kommuniziert. Die durch einen Kommunikationsgraphen dargestellte logische Verbindungen zwischen den virtuellen MIMD-Prozessen stellen ein virtuelles Verbindungsnetzwerk (VNW) dar. Die minimale Körnigkeit der virtuellen Prozesse (VP) charakterisiert den Umfang der Prozess-Berechnungsarbeiten und die VP-Menge. Sie ist von der Dekomposition der Topologie und Modellgleichungen sowie der örtlichen DNOVP-Approximation abhängig. Bei der Dekomposition und Approximation entstehen Teilsysteme, deren nicht zerlegbaren Elementen diskretisiert werden. Deshalb wird vorgeschlagen, die minimale Körnigkeit der virtuellen Prozessen entsprechend der bei der Dekomposition und Diskretisierung hergeleiteten Simulationsmodellgleichungen zu definieren. Die mit der minimalen VP-Körnigkeit verbundenen Ansätze zur DNOVP-Parallelisierung führen auf die Parallelitätsebenen der virtuellen parallelen Simulationsmodelle. Abb. 1 zeigt vier möglichen DNOVP-Parallelitätsebenen (PE). Nach der Auswahl der numerischen Verfahren werden die den Parallelisierungsansätzen und den Parallelitätsebenen zugeordneten diskreten virtuellen parallelen Simulationsmodelle (DVPSM) und deren Blockdiagramme erstellt.

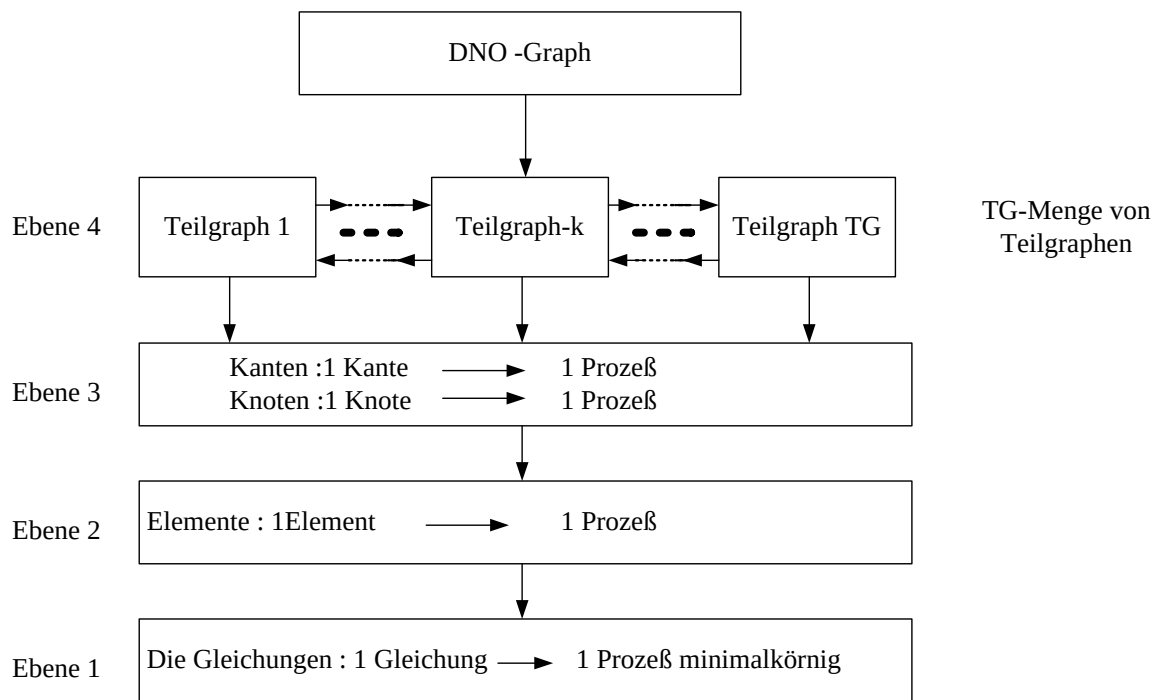


Abb.1 – Die VPSM-Parallelitätsebenen des dynamischen Netzobjektes mit verteilten Parametern

4. DVPSM-Devirtualisierung

Das Zielrechnersystem (ZRS) ist die dem Modellentwickler zur Verfügung stehende lose oder/und eng gekoppelte, nach MIMD-Prinzip funktionierende, beschränkte heterogene Menge der installierten vollfunktionellen Prozessoren mit lokalem oder/und gemeinsamem Speicher und verfügbarem vordefiniertem programmgesteuertem Verbindungsnetzwerk.

Paralleler DNOVP-Simulator wird von uns als vorgegebenes Ziel der parallelen DNOVP-Modellierung und als eine Hardware/Software-Systemorganisation definiert, die den Modellierungsvorgang und diskretes virtuelles paralleles DNOVP-Simulationsmodell auf dem Zielrechnersystem softwaretechnisch effizient realisiert und den Modellentwicklern sowie den Modellanwendern benutzerfreundliche Kommunikation mit den Modellierungs- und Simulationsressourcen erlaubt.

Die Devirtualisierung des diskreten virtuellen parallelen DNOVP-Simulationsmodells ist eine DPVSM-Umwandlung, die zur Simulatorrealisierung auf gegebenen ZRS eindeutig führt und aus folgenden Grundfunktionen besteht:

1. A-priori-Analyse des DVPSM aller Parallelitätsebenen. In Tabelle 2 sind die Ergebnisse der Analyse für Ebenen 1...4 mit den Merkmalen „Gleichmäßigkeit der Prozeßbelastung“, „Verhältnis zwischen den Rechen- und Datenaustauschoperationen“, „Virtuelle Beschleunigung“, „Verbindungsschema für Prozesse“ und „Menge der Prozesse“ dargestellt.
2. Vergleich der DVPSM-PE, Auswahl der optimalen virtuellen Parallelitätsebene (OVPE).
3. Darstellung der vorhandenen ZRS-Ressourcen.
4. Zuordnung „virtuelle Prozesse - reale Prozessoren“, „virtuelles VNW – reales ZRS-VNW“ für ausgewählten OVPE.

Tabelle 2 – Zusammenfassende ApriAna-Ergebnisse

ParEbene	Ebene 1 (VPM-1)	Ebene 2 (VPM-2)	Ebene 3 (VPM-3)		Ebene 4 (VPM-4)
			Variante 1	Variante 2	
Merkmale					
Gleichmäßigkeit der Belastung	Wesentliche Lastungleichmäßigkeit Q-, P-Prozesse, die durch die verschiedene Menge der RS-Operationen hervorgerufen wird. Ungleichmäßigkeitsrate: $\Delta T_{\text{max}} = T_{\text{Qmax}} - T_{\text{Pmin}} = (S_{\text{Qmax}} - S_{\text{Pmin}}) / (t_{\text{Qmax}} - t_{\text{Pmin}}) = 4t_{\text{Qmax}} - S_{\text{Qmax}} + t_{\text{Pmin}}$	Belastung der QP-Prozesse ist größer und gleichmäßiger geworden. UGM-Rate: $\Delta T_{\text{QP}} = 4t_{\text{Qmax}} + S_{\text{Qmax}}$ charakterisiert die RS-Unterschiede und ist kleiner als bei VPM-1: $\Delta T_{\text{max}} - \Delta T_{\text{QP}} = t_{\text{Qmax}} - t_{\text{Pmin}} - t_{\text{Pmax}}$	Belastung der (QP)-Prozesse $L_j = 2M_j = I_j / \Delta t_j^{\text{QP}}$ ist von der gelösten Gleichungsmenge und der I_j -Abweichungen abhängig. Ungleichmäßigkeitsrate: $\Delta L = L_{\text{max}} - L_{\text{min}} = 2(M_{\text{max}}/M_{\text{min}})$ ist wegen I -Diaspan 100–2000x groß.	Jeder $N_j + J$ Prozess löst nicht weniger als $M_{\text{min}} = I_{\text{min}}/\Delta t_j^{\text{QP}}$ Gb-Paare. Es wurde die Formalisierung der Planung und gleichmäßiger Verteilung der Prozessbelastung entsprechend dem vorgeschlagenen Algorithmus, der die zusammenfassende Daten bezüglich jeder Kante in Tabelle TABLAST formiert.	Die Lastgleichmäßigkeit von Prozessen ist von der Graph-Zerlegung in Fragmenten (Teilgraphen) abhängig. Es wurde (kombiniertes Verfahren der Zerlegung nach Fragmente F_i vorgeschlagen, welches technologischer und formaler Ansätze vereint und die Gleichmäßigkeit als Variante 2 der VPM-3 gibt.
Verhältnis $N_{\text{QP}}/N_{\text{DA}}$	$1 \leq N_{\text{QP}}/N_{\text{DA}} \leq 2$	$2 \leq N_{\text{QP}}/N_{\text{DA}} \leq 4$	$2M_j \leq N_{\text{QP}}/N_{\text{DA}} \leq 4M_j$	$2M_{\text{min}} \leq N_{\text{QP}}/N_{\text{DA}} \leq 4M_{\text{min}}$	$\frac{3M_{\text{P}}}{N_{\text{DA}}^{\text{P}}} \leq N_{\text{QP}}^{\text{P}}/N_{\text{DA}}^{\text{P}} \leq \frac{16M_{\text{P}}}{N_{\text{DA}}^{\text{P}}}$
Virtuelle Beschleunigung mit der Berücksichtigung der DA-Operationen	$S_{\text{Q}} = \frac{T(1)}{T(2M)} = \frac{1 + T_{\text{P}}/T_{\text{Q}}}{1 + T_{\text{DA}}/T_{\text{Q}}}$ $S_{\text{Pmin}} = 2M \frac{1}{1 + T_{\text{DA}}/T_{\text{Q}}}$	$S_{\text{Q}} = \frac{T(1)}{T(M)} = \frac{M(T_{\text{Q}} + T_{\text{P}})}{T_{\text{Qmax}} + T_{\text{DA}}}$ $S_{\text{Pmin}} = M \frac{1}{1 + 0,25T_{\text{DA}}/T_{\text{Q}}}$	$S_{\text{Q}}^{(1)} = \frac{T_{\text{DA}}(1)}{T_{\text{DA}}(m)} = \frac{1 + (\sum_{j=1}^{n-1} M_j T_{\text{QP}}) / M_{\text{min}} T_{\text{QPmin}}}{1 + \frac{T_{\text{DA}}(1)}{M_{\text{min}} T_{\text{QPmin}}}}$	$S_{\text{Q}}^{(2)} = T_{\text{DA}}(1) / T_{\text{DA}}(N_{\text{P}})$ $S_{\text{Pmin}}^{(2)} = \frac{\sum_{j=1}^n \frac{M_j T_{\text{QP}}}{M_{\text{min}} T_{\text{QPmin}}}}{1 + \frac{T_{\text{DA}}(1)}{M_{\text{min}} T_{\text{QPmin}}}}$	$S_{\text{P}} = \frac{\sum M_j T_{\text{QP}}}{\sum M_j T_{\text{QP}}} \frac{1}{1 + \frac{T_{\text{DA}}(1)}{\sum M_j T_{\text{QP}}}}$
Verbindungsschema für Prozesse	Es wurden die virtuelle QP- und PQ-Kommutatoren vorgeschlagen, die DA-Operationen minimieren.				
Menge der Q-, P-Prozesse	$N_{\text{TPM1}} = \frac{2}{\Delta t_j^{\text{QP}}} \sum_{j=1}^n I_j$	$N_{\text{TPM2}} = \sum_{j=1}^n M_j = \frac{1}{\Delta t_j^{\text{QP}}} \sum_{j=1}^n I_j$	$N_{\text{TPM3}}^{(1)} = m$	$N_{\text{TPM3}}^{(2)} = \sum_{j=1}^n N_{\text{P}} + (m - m_{\text{min}})$	Prozessenmenge entspricht der Menge von MIMD-Prozessoren

5. Formierung der Spezifikation von OVPE-MIMD-Prozessen, die den Prozessoren zugeordnet wurden.
6. Implementierung und Debugging des parallelen DNOVP-Simulators.

5. Implementierungserfahrungen

Die obige virtuelle Simulationsmodelle und ihre Devirtualisierung werden anhand der DNOVP-Beispiele aus des Gegenstandsgebiets „Grubenbewetterungssysteme“ betrachtet. Nach der Dekomposition und Ortsdiskretisierung des Bewetterungsnetzes mit $m = 117$ Kanten und $n = 61$ Knoten geben diese Arten von Prozessen insgesamt mehr als 100000 Gleichungen des virtuellen parallelen minimalkörnigen Simulationsmodells auf erster Parallelitätsebene (Abb.1). Diese Gleichungen werden nach den Angaben des Topologieanalysators mit dem Gleichungsgenerator automatisch erstellt. Im Rahmen der DNOVP-parallelen Simulationsumgebung mit den HLRS-ZRS-Ressourcen [1, 2] wurde den oben vorgeschlagenen Devirtualisierungsvorgang untersucht und für jede DNOVP-Parallelitätsebene den parallelen Versuchssimulator gebaut. Die Gleichungslösern der parallelen Simulatoren werden aufgrund der parallelisierten expliziten konventionellen sowie impliziten blockartigen numerischen Verfahren entwickelt [1, 3].

6. Zusammenfassung und Ausblick

Die vorgestellten Ergebnisse liefern einen Beitrag für rechnergestützte Erstellung von Modellen, Simulationsmodellen und parallelen Simulatoren der dynamischen Netzobjekte mit verteilten Parametern (DNOVP). Die simulationstechnisch effiziente Nutzung der parallelen Ressourcen und die Erhöhung der Benutzerfreundlichkeit von DNOVP-orientierten parallelen Simulationsumgebung sind die Gegenstände der Zusammenarbeit DonNTU mit der Universität Stuttgart (HLRS, SimTech Cluster). Die aktuelle ParSimTech-Aktivitäten konzentrieren sich auf

der Entwicklung von DNOVP-Simulatoren aufgrund blockartiger paralleler numerischer Verfahren und den universitären sowie industriellen Anwendungen erzielten Ergebnissen.

Literatur

1. Feldmann L.P., Resch M., Svjatnyj V.A., Zeitz M.: Forschungsgebiet: parallele Simulationstechnik. In: DonNTU, FRTI-Werke, Reihe "Probleme der Modellierung und rechnergestützten Projektierung von dynamischen Systemen", Band 9(150). – Donezk, 2008, S. 9-36. Auch in: ASIM'09, Tagungsband Vorträgekurzfassungen, Cottbus, 2009.
2. Svjatnyj V.A., Moldovanova O.V., Feldmann L.P.: Parallele Simulationsumgebung für dynamische Netzobjekte mit verteilten Parametern. In: F.Hülsemann u.a. (Hrsg.), Tagungsband 18. ASIM-Symposium Simulationstechnik, Erlangen 2005, SCS 2005, 416-421.
3. Guseva A.B., Kushnarenko V.G.: Diskretes Simulationsmodell des dynamischen Netzobjektes mit verteilten Parametern aufgrund des blockartigen numerischen Verfahrens (Russisch). In: DonNTU-Werke, Reihe "Probleme der Simulationstechnik und rechnergestützten Projektierung", Ausgabe 9 (179). – Donezk, 2011, S. 274-283.

Master zur Simulatorkopplung via FMI

Christoph Clauß, Fraunhofer IIS EAS, Zeunerstr. 38, 01069 Dresden,

christoph.clauss@eas.iis.fraunhofer.de

Martin Arnold, Tom Schierz, Martin-Luther-Universität Halle-Wittenberg,
06099 Halle (Saale),

{martin.arnold, tom.schierz}@mathematik.uni-halle.de

Jens Bastian, ITI GmbH, Webergasse 1, 01067 Dresden,
bastian@itisim.com

Zusammenfassung

Die Kopplung von Simulationswerkzeugen erfordert den Datenaustausch wie auch die Synchronisation, d.h. die geeignete Definition von Kommunikationszeitpunkten, an denen dieser Datenaustausch erfolgt. Die Kommunikation mit den Simulatoren, die Bereitstellung von deren Eingangsdaten und die Ansteuerung (Start, Stop, Fehlerverwaltung) unterliegen einem übergeordneten Master-Algorithmus. Mit dem Functional-Mock-Up-Interface (FMI)-Standard wurde eine Schnittstelle für die Kopplung von Simulatoren definiert, die Datenaustausch und Simulatoransteuerung vereinheitlicht. Für die Entwicklung und Erprobung von Master-Algorithmen wurde im ITEA2-Forschungsprojekt MODELISAR am Fraunhofer IIS EAS Dresden in enger Zusammenarbeit mit Projektpartnern ein prototypisches Tool für Entwicklung und Test von Master-Algorithmen („EAS-Master“) geschaffen. Weitgehend basierend auf FMI 1.0 gestattet dieser EAS-Master die Kopplung von zwei oder mehreren Simulatoren und bietet je nach Eigenschaften der beteiligten Simulatoren gegenwärtig folgende Verfahren zur Auswahl an: Mit konstanter Kommunikationsschrittweite eine einfache Abarbeitung ohne Iteration, mit einfacher Fixpunktiteration oder mit Newtonverfahren, sowie mit variabler Kommunikationsschrittweite die einfache Abarbeitung, wobei die Schrittweite durch numerische Fehlerabschätzungen gesteuert wird (Richardson-Extrapolation). Im Beitrag werden die Problematik der Simulatorkopplung und FMI erläutert, sowie der aktuelle Stand des EAS-Masters vorgestellt und an drei Beispielen die Funktionsweise demonstriert. Erweiterungs- und Einsatzmöglichkeiten des EAS-Masters werden diskutiert.

1 Einleitung

In Entwicklungsprozessen komplexer Produkte ist die Modellbildung und die Untersuchung dieser Modelle (Simulation) anstelle der Produkte selbst zum Zweck des Studiums der Funktionsweise sowie zur Optimierung oder Gewinnung von Robustheitsaussagen nicht mehr

wegzudenken. Komplexe, heterogene und multidisziplinäre Systeme z.B. im Automobilbau sind strukturiert. Dies impliziert eine Fülle hierarchisch strukturierter Modelle, die bedingt durch verschiedene im System wirkende physikalische Gebiete, durch verschiedene Untersuchungsziele und wegen der Produktionskette auch durch verschiedene Modellautoren uneinheitlich und oft auf verschiedene Zielsimulatoren ausgerichtet sind. Teilsysteme werden für bzw. mit Simulatoren/Tools modelliert, die besonders gut auf die physikalischen Eigenschaften der Teilsysteme zugeschnitten sind. Ist die simulative Untersuchung mehrerer Baugruppen oder die Gesamtsimulation das Ziel, wird es erforderlich, verschiedenartige Modelle zusammenzubringen. Dazu sind verschiedene Wege möglich:

- a) Die Verwendung einer einheitlichen Modellierungssprache, in die alle Modelle zum Zweck der Simulation auf einen einzelnen Zielsimulator konvertiert werden,
- b) der Austausch von simulationsfähigen Modellen zwischen den Simulatoren mit dem Ziel der Ausführung aller Modelle auf einem einzelnen Simulator,
- c) die gekoppelte Simulation mehrerer Simulatoren (Co-Simulation).

Alle diese Wege sind sinnvoll und werden je nach Situation angewendet. Allgemeine Verhaltensbeschreibungssprachen wie VHDL-AMS oder Modelica gestatten theoretisch die Beschreibung einer großen Breite von Modellen, andererseits ist oft aus praktischen Gründen (Zeit, Kapazität, Toolverfügbarkeit) eine nachträgliche Modellkonvertierung, die meist nicht vollständig automatisiert werden kann, ausgeschlossen. Es ist dann entweder auf (standardisierten) Modellaustausch zwischen den Simulatoren, der nicht die Konvertierung auf Sprachebene erfordert, auszuweichen, oder auf Simulatorkopplung.

Dieser Beitrag stellt als ein Ergebnis des ITEA2-Forschungsprojektes MODELISAR (2008-2011) ein Werkzeug (Master) vor, das die Kopplung von dafür vorbereiteten Simulatoren (Slaves) ermöglicht und das der Simulatorkopplung übergeordnete Algorithmen enthält [1] [2]. Als Schnittstelle zwischen den Simulatoren wird das ebenfalls in diesem Projekt entwickelte Functional-Mockup-Interface (FMI) verwendet. Der Master wurde am Fraunhofer IIS EAS Dresden in enger Zusammenarbeit mit der Martin-Luther-Universität Halle und weiteren Projektpartnern entwickelt und befindet sich in einem prototypischen Zustand, der die Implementierung und Erforschung verschiedener Kopplungsalgorithmen gestattet und mehrere Algorithmen zur Anwendung enthält. Nachfolgend wird die seit langem untersuchte Problemstellung der Simulatorkopplung erläutert, auf die Schnittstelle FMI eingegangen sowie der aktuelle Stand des Masters dargestellt und in drei Beispielen angewendet.

2 Simulatorkopplung

Mit „Co-Simulation“ wird eine Vielfalt von Lösungsansätzen zur Simulation von heterogenen Systemen bezeichnet, aber auch zur Kopplung von Tools oder von numerischen Lösern, und zur Einbeziehung real existierender Systemkomponenten in die Simulation. Im Fokus dieses Beitrages steht die Kopplung zweier oder mehrerer Simulationswerkzeuge zur Simulation von gekoppelten, modularen Systemen.

Unter einem Simulator wird hier nicht nur ein leistungsfähiges kommerzielles oder freies Simulationswerkzeug verstanden, es kann sich auch z.B. um ein einfaches Programm zur Lösung von Differentialgleichungen handeln. Der Simulator sollte im Zeitbereich arbeiten, d.h. die vom Simulator berechneten Variablen sollen zeitabhängige Größen sein.

Um einen Simulator koppeln zu können, muss er bestimmte Eigenschaften erfüllen:

- Er muss zeitinkrementell arbeiten.
- Die Simulation muss unterbrechbar und fortsetzbar sein.
- Während der Unterbrechung muss er berechnete Werte ausgeben und für die Fortsetzung der Simulation Werte annehmen können.
- Er sollte vorgegebene Intervalle mehrfach mit unterschiedlichen vorgegebenen Werten abarbeiten können.

Aus Sicht der Kopplung ist ein Simulator S eine Vorschrift (Funktion), die auf vorgegebene Werte $u(t)$ angewendet wird und Ergebnisse $y(t)$ berechnet (Abbildung 1).

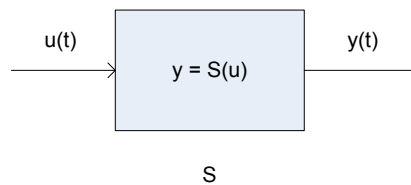


Abbildung 1: Blockdarstellung eines koppelbaren Simulators

Bei der Simulatorkopplung zur Lösung eines partitionierten Problems werden die für die Partitionen gewählten N Simulatoren „verschaltet“, wobei auch möglich ist, gleiche Simulatoren mehrmals zu verwenden. Es entsteht ein gerichteter Graph mit den Simulatoren als Knoten und den Verteilungsvorschriften der berechneten Werten $y(t)$ eines jeden Simulators als Eingänge $u(t)$ anderer Simulatoren als Kanten. Fasst man die Ausgänge $y_i(t)$ aller Simulatoren zum Vektor $Y(t)$ zusammen und alle Simulatoren S_i zu $\bar{S}$, und bezeichne G eine dem Graph zugeordnete Matrix, dann kann die Verkopplung wie folgt kompakt geschrieben werden: $Y(t) = \bar{S}(GY(t))$.

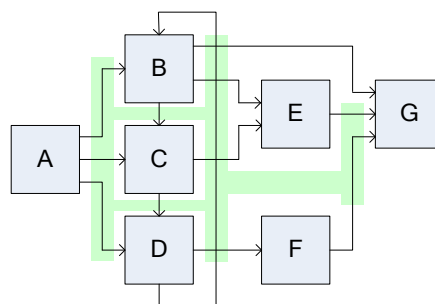


Abbildung 2: Graph mit sieben gekoppelten Simulatoren

Abbildung 2 zeigt als Beispiel eine Verkopplung von mehreren Simulatoren. Statt die Simulatoren direkt zu verkoppeln, wird ein Block zwischen allen Simulatoren angeordnet, mit dem die einzelnen Simulatoren kommunizieren (in Abbildung 2 angedeutet als Fläche

zwischen den Simulatoren). Dieser Block soll **Master** heißen, und die beteiligten Simulatoren **Slaves**.

Der Master hat folgende Aufgaben:

- Der Master arbeitet das vorgegebene, allen Simulatoren gemeinsame Zeitintervall $[t_{start}, t_{stop}]$ ab, indem er Kommunikationszeitpunkte tc_i mit $t_{start} \leq tc_i \leq tc_{i+1} \leq t_{stop}$ definiert und die Slaves in Intervallen $[tc_i, tc_{i+1}]$ zur Simulation auffordert.
- Der Master nimmt an den Kommunikationszeitpunkten tc_i die von den Slaves berechneten Ausgangswerte $y(tc_i)$ auf, verarbeitet diese und stellt entsprechend des Verbindungsgraphen G Eingangsgrößen $u(tc_i)$ an die Slaves bereit.
- Der Master verfolgt für die vorgenannten Aufgaben einen ihm vorgegebenen oder von ihm selbst gewählten Algorithmus, den Master-Algorithmus.

Dieser punktweise Austausch von Werten stellt eine Einschränkung der Allgemeinheit dar, die z.B. den Austausch von kompletten Zeitfunktionen einer bestimmten Klasse wie Polynomen nicht abdeckt. Die Einschränkung wird wenigstens teilweise wieder aufgehoben, indem neben Werten $y(tc_i)$ und $u(tc_i)$ auch Ableitungen nach der Zeit oder Jacobimatrizen übertragen werden können.

Eine Simulatorkopplung nach dem Master-Slave-Prinzip lässt sich grundsätzlich in folgende drei Phasen einteilen:

1. **Initialisierung:** Die beteiligten Simulatoren werden vorbereitet, automatisch oder manuell gestartet, die Datenübertragung wird eingerichtet, und notwendige Initialisierungen erfolgen. Der Master erhält den Verbindungsgraphen G sowie Informationen über die Eigenschaften der Slaves und über den anzuwendenden Master-Algorithmus.
2. **Simulation:** Der Master lässt die Slaves in Intervallen $[tc_i, tc_{i+1}]$ das gesamte Zeitintervall simulieren. Je Kommunikationspunkt übernimmt er die errechneten Werte jedes Slaves, berechnet für diesen neue Eingabewerte und übergibt diese zusammen mit dem nächsten Kommunikationszeitpunkt. Danach wird die Simulation für jeden Slave fortgesetzt. Weiterhin ist je Kommunikationszeitpunkt des Status des Slaves (Fehler, Erfolg, ...) zu kommunizieren.
3. **Abschluß:** Der Master beendet die komplette Simulation einschließlich der Beendigung der Slaves.

Die skizzierten Aufgaben des Masters erfordern eine generelle Lösung der Kommunikation zwischen Master und Slave (Daten und Ablaufkommandos) und die Bereitstellung von Masteralgorithmen. Für die Kommunikation wurde das Functional-Mockup-Interface (FMI) geschaffen, Masteralgorithmen werden z.B. im hier beschriebenen Master behandelt.

Bei vielen Kopplungen vor allem zweier Simulatoren wird kein separater Master bemüht, sondern Daten werden nach einem bestimmten Zeitschema ausgetauscht, oft mit fester Kommunikationsschrittweite und ohne Überprüfung der an der Schnittstelle entstehenden Fehler. Derartige Kopplungen lassen sich in den meisten Fällen adäquat auch mit einem einfachen Master-Algorithmus beschreiben, so dass die hier beschriebene Vorgehensweise keine Einschränkung darstellt. „Implizit“ existiert stets ein Master-Algorithmus.

3 Functional-Mockup-Interface

Das Functional-Mockup-Interface [3] [4] ist ein in Entwicklung begriffener Standard für Kopplung und Austausch von Modellen verschiedener Domänen und für Simulatorkopplung. Eine sogenannte Functional-Mockup-Unit (FMU) enthält C-Funktionen entweder im Quelltext, meist aber vorverarbeitet in Form dynamisch linkbarer Bibliotheken („Binary“ wie dll, oder sharable object), und ein Beschreibungsfile im XML-Format. Die C-Funktionen können gerufen werden, um Werte an die FMU zu schicken (z.B. `fmiSetReal`) und Werte abzuholen (z.B. `fmiGetReal`). Wenn die FMU für Simulatorkopplung gedacht ist, enthält sie einen kompletten Simulator (Slave) einschließlich des Modelles, das dieser simuliert, oder sie stellt die Verbindung zu einem separaten installierten Simulationstool her. Dieser Slave kann gesteuert werden, z.B. fordert `fmiDoStep` auf, ein Kommunikationsintervall abzuarbeiten. Um den im vorigen Abschnitt angegebenen grundsätzlichen Ablauf der Simulatorkopplung einzuhalten, können die C-Funktionen nur in einer bestimmten Reihenfolge gerufen werden, die in der FMI-Definition als Ablaufdiagramm festgelegt ist. Im XML-Beschreibungsfile stehen alle „statischen“ Informationen, die zum Aufruf der FMU erforderlich sind, wie Namen, Anzahlen und Typen von Variablen sowie Referenzen für den Zugriff zur Anwendung in den C-Funktionen. Zur Charakterisierung der Möglichkeiten des Slaves in der Simulatorkopplung enthält das XML-File Attribute. Zum Beispiel bedeutet das auf `true` gesetzte Attribut `canHandleVariableCommunicationStepSize`, dass der Slave variable Kommunikationsschrittweite akzeptiert.

In der Definition des FMI sind keine Algorithmen zur Simulatorkopplung festgelegt. Die Schnittstelle ist aber sehr allgemein gehalten, um sehr viele Algorithmen zu ermöglichen.

4 Master

Zur Entwicklung und zum Test von Master-Algorithmen wurde ein Master prototypisch implementiert, der grundlegende Funktionen des FMI 1.0 unterstützt und bereits verschiedene Master-Algorithmen zur Auswahl anbietet. Das am EAS verfügbare Package enthält den ANSI-C-Code des Masters, einen einfachen Slave (C-Funktion) und eine Beispielsammlung.

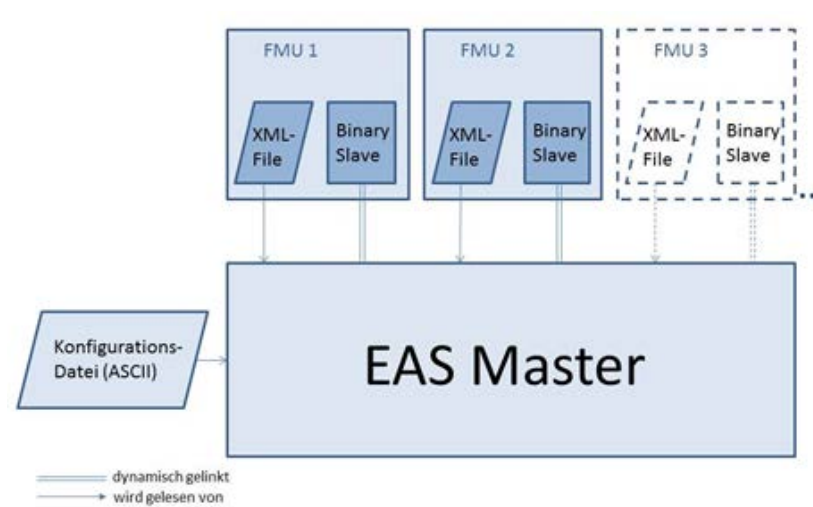


Abbildung 3: Anwendungsprinzip des Masters

Um den EAS-Master mit gegebenen FMUs anzuwenden (siehe Abbildung 3), sind folgende Schritte erforderlich:

- Die Binaries der Slaves sind mit dem übersetzten C-Code des Masters zu verlinken. Dies ist je nach verwendeter oder durch die FMUs vorgegebener Plattform individuell durchzuführen. Gegebenenfalls besitzt die FMU anstelle des Binary auch C-Code, der in die Übersetzung einbezogen werden kann. Ergebnis dieses Schrittes ist ein ablauffähiges Programm, das Master und Slaves enthält. Es ist auch möglich, dass das Binary der FMU eine weitere Schnittstelle (nicht FMI-Schnittstelle) zu einem in einem separaten Prozess z.B. per Hand gestarteten Simulator enthält, der ggf. über Internet angekoppelt ist.
- Die `modelDescription.xml` XML-Files der FMUs werden dem Master durch Angabe des entsprechenden Pfades in der Konfigurationsdatei zur Auswertung bereitgestellt.
- Der Master wird gestartet und erhält dabei als Parameter eine Konfigurationsdatei.

4.1 Aufbau der Konfigurationsdatei

Die Konfigurationsdatei ist eine einfache ASCII-Textdatei mit durch Zeilenwechsel getrennten Angaben. Sie enthält in fest vorgegebener Reihenfolge die Anzahl der auszutauschenden (Einzel-)Werte (`nval`) und die Anzahl der beteiligten Simulatoren (`nsim`). Es folgen Start- und Endzeitpunkt der gekoppelten Simulation (`tstart`, `tend`) und Angaben zur Schrittweite (`tstepmax`, `tstepstart`), deren Bedeutung durch den gewählten Algorithmus festgelegt wird. Weiterhin wird der Algorithmus des Masters festgelegt (`MasterMode`), der Debug-Level (`MasterDebug`) zur Überprüfung der Arbeitsweise des Masters und die Ausgabe einer Textdatei zur Anzeige der errechneten Koppelvariablen mit Gnuplot [6] (`OutputGnuplot`). Es folgen Angaben zur Steuerung der Numerik (`it_max_steps`, `it_tol_abs`, `it_tol_rel`, `sz_tol_abs`, `sz_tol_rel`), die für die einzelnen Master-Algorithmen notwendig sind. Anschließend wird den beteiligten Slaves beginnend ab Null eine Nummer zugewiesen und der Pfad zur FMU angegeben, die dem Slave entspricht. Z.B. bedeutet die Zeile

```
simulator 1 QC_WheelSystem_fmu
```

dass für den Slave 1 im Unterverzeichnis `QC_WheelSystem_fmu` die Datei `modelDescription.xml` für die mit `QC_WheelSystem` bezeichnete FMU liegt.

Anschließend wird der Verbindungsgraph angegeben, wobei die zu übertragenden Variablen ebenfalls mit Null beginnend nummeriert werden. Z.B. bedeutet folgende Zeile

```
2 0 -1 r 335544320
```

dass die Variable 2 (erste Zahl) vom Slave 0 (zweite Zahl in der Zeile) ausgegeben wird (-1, eine Eingabe würde durch 1 gekennzeichnet), dass es sich um eine reelle Zahl handelt (r), und dass sie unter der `fmiValueReference`-Nummer 335544320 mit der FMI-Funktion `fmiGetReal` vom Master beim Slave 2 abgefragt werden kann. Auf diese Weise werden sämtliche Eingabe- und Ausgabegrößen aller Slaves dem Master mitgeteilt. Jede Variable muss dabei genau einmal als Ausgabegröße eines Slaves vorkommen.

Die folgenden Angaben sind vorläufig noch erforderlich und sollen in einer weiteren Ausbaustufe automatisch durchgeführt werden. Es werden zunächst die im Graph vorhandenen Zyklen festgestellt und je Zyklus zu einem „Superslave“ zusammengefasst. Der Graph ist dann frei von Zyklen und es kann eine Berechnungsreihenfolge angegeben werden. Die höchste Priorität (null) erhalten alle Slaves und Superslaves, die keine Eingänge besitzen. Werden diese und deren Ausgangsvariablen aus dem Graph entfernt, entstehen neue Slaves und Superslaves, die keine Eingänge besitzen. Diese erhalten die nächste Priorität (eins) usw. Im Graph der Abbildung 2 gibt es z.B. einen Zyklus (B, C, D). Slave A hat die Priorität 0, während B, C und D die Priorität 1 erhalten, E und F die Priorität 2 und G die Priorität 3, siehe Abbildung 4.

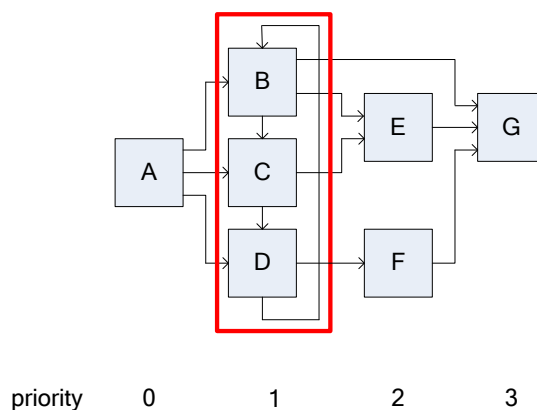


Abbildung 4: Prioritäten im Graph aus Abbildung 2

In der Konfigurationsdatei werden in einer Tabelle die Prioritäten aller Slaves angegeben, und in einer weiteren Tabelle die Anzahl der Zyklen in jeder Priorität. Beispiele von Konfigurationsdateien sind im Punkt 5 angegeben.

4.2 Master-Algorithmen

Die angebotenen Master-Algorithmen rufen die Slaves entsprechend ihrer Priorität auf, beginnend mit der Priorität Null. Sofern sie nicht in Zyklen eingebunden sind, können die Slaves gleicher Priorität parallel abgearbeitet werden. Dies ist vorbereitet, indem bei der Implementierung OpenMP [7] verwendet wurde. Im einzelnen werden folgende Algorithmen angeboten:

- **Einfache Abarbeitung ohne Iteration mit fester Kommunikationsschrittweite:**
Die Slaves werden entsprechend ihrer Priorität abgearbeitet. Falls Zyklen vorhanden sind, werden diese genau einmal durchlaufen. Bei diesem Verfahren lässt sich die Genauigkeit durch Reduktion der Schrittweite erhöhen, falls keine Zyklen vorhanden sind. Es treten dann nur Fehler durch die auf die Kommunikationszeitpunkte eingeschränkte Datenübertragung auf. Ableitungen werden bei der Übertragung nicht berücksichtigt. Falls Zyklen vorhanden sind, sind Annahmen für zumindest einzelne Eingangsgrößen erforderlich, die nach Abarbeitung des Zyklus nicht in jedem Fall bestätigt werden und zu zusätzlichen Fehlern führen. Die einfache Abarbeitung

entspricht der einmaligen Auswertung der im Abschnitt 2 angegebenen kompakten Formel $Y(t) = \bar{S}(GY(t))$ in geeigneter Reihenfolge je Kommunikationszeitpunkt.

- **Fixpunktiteration aller Zyklen bei fester Kommunikationsschrittweite:** Die Abarbeitung erfolgt wie bei einfacher Abarbeitung ohne Iteration, jedoch werden alle Zyklen mehrfach durchlaufen, bis sich alle Ein- und Ausgangsgrößen der einzelnen, am Zyklus beteiligten Slaves im Rahmen einer vorgegebenen Toleranz nicht mehr ändern, das heißt, bis die Iteration $Y^{j+1}(t) = \bar{S}(GY^j(t))$ konvergiert. Dabei werden die Ausgangsgrößen eines gerade abgearbeiteten Slaves für die folgenden bereits wirksam (Gauss-Seidel-Iteration). Die Iterationszahl wird begrenzt, die Grenze wird im Konfigurationsfile festgelegt. Auch hierbei kann die Genauigkeit durch Verkleinerung der Kommunikationsschrittweite verbessert werden, weil damit der Approximationsfehler bei der Datenübertragung verringert werden kann. Voraussetzung ist die Konvergenz der Iteration bei sinnvollen Fehlerschranken. Nichtkonvergenz mit unbrauchbaren Ergebnissen ist möglich. Um dieses Verfahren anwenden zu können, ist es erforderlich, dass alle an den Zyklen beteiligten Slaves Kommunikationsintervalle mehrfach abarbeiten können.
- **Newton-Verfahren für alle Zyklen bei fester Kommunikationsschrittweite:** Die Abarbeitung erfolgt wie im vorhergehenden Algorithmus, allerdings werden die Zyklen mit einem Newtonverfahren angewendet auf $0 = \bar{S}(GY(t)) - Y(t)$ anstelle der einfachen Fixpunktiteration gelöst. Die Genauigkeit kann über die Kommunikationsschrittweite und die Fehlertoleranzen gesteuert werden. Konvergenz tritt ein bei hinreichend genauer Startlösung für das Newtonverfahren, die durch Übernahme der errechneten Werte des letzten Zeitpunktes und kleiner Kommunikationsschrittweite gewonnen werden kann. Schwierigkeiten können beim Start auftreten, weil eine gute Startlösung fehlt. Auch hier müssen die Slaves Kommunikationsintervalle mehrfach abarbeiten können, auch um die für das Newtonverfahren benötigte Jacobimatrix berechnen zu können.
- **Variable Kommunikationsschrittweite, gesteuert über Richardson-Extrapolation [5]:** Ein Kommunikationsintervall wird zunächst in einem Schritt abgearbeitet. Anschließend werden die Slaves zurückgesetzt, und das gleiche Intervall wird in zwei Zeitschritten mit halber Kommunikationsschrittweite abgearbeitet. Weichen die Ergebnisse zu sehr voneinander ab, wird die Kommunikationsschrittweite verringert, andernfalls wird diese beibehalten oder erhöht. Diese Schrittweitensteuerung kann mit allen drei vorgenannten Algorithmen kombiniert werden. Im aktuellen Master wird die Richardson-Extrapolation mit einfacher Abarbeitung aller Slaves angeboten.

Alle Verfahren berücksichtigen nur kontinuierliche, reelle Ein- und Ausgangsgrößen der Slaves.

5 Anwendungsbeispiele

Zur Erprobung der Master-Algorithmen wurden mehrere Testbeispiele einfacher Art bereitgestellt, bei denen die Slaves in C geschriebene Formelauswertungen sind. Weiterhin wurden Kopplungen mit den Simulatoren SimulationX [8] und Dymola [9] durchgeführt. In diesem Abschnitt wird für jede dieser Kopplungen ein Beispiel angegeben.

5.1 Lineares Gleichungssystem mit zeitabhängiger Matrix

Zeitabhängige lineare Gleichungen werden in folgender Weise je einem Slave zugeordnet:

Slave 0: kein Eingang,	Gleichung: $r_1 = 1, r_2 = 0, r_3 = 1$	Ausgang: r_1, r_2, r_3
Slave 1: Eingang r_1, x_2, x_3	Gleichung: $3x_1 + (0.1+t)x_2 + 0.2x_3 = r_1$	Ausgang: x_1
Slave 2: Eingang r_2, x_1, x_3	Gleichung: $0.1x_1 + 3x_2 + (0.1+t)x_3 = r_2$	Ausgang: x_2
Slave 3: Eingang r_3, x_1, x_2	Gleichung: $(0.1+t)x_1 + 0.2x_2 + 4x_3 = r_3$	Ausgang: x_3
Slave 4: Eingang x_1, x_2, x_3	Gleichung: $x_1 + x_2 + x_3 = y$	Ausgang: y

Slave 0 hat Priorität 0, die Slaves 1, 2 und 3 bilden einen Zyklus und haben die Priorität 1, Slave 4 hat Priorität 2 und wird deshalb als Letzter aufgerufen.

Nachfolgend sind die Bestandteile des Konfigurationsfiles für den Master angegeben (allgemeine Angaben, Graph, Prioritätentabelle und Zyklen):

```
nval      7
nsim      5
tstart    0
tend      4
tstepmax  1
tstepstart 1
MasterMode 9
MasterDebug 1
OutputGnuplot 1
it_max_steps 1000
it_tol_abs 1e-6
it_tol_rel 1e-4
simulator 0 bspEmodel0_fmu
simulator 1 bspEmodel1_fmu
simulator 2 bspEmodel2_fmu
simulator 3 bspEmodel3_fmu
simulator 4 bspEmodel4_fmu
```

```
priority #sim priority
0 0
1 2
2 1
3 1
4 1
end
```

```
graph #val #sim -1(out)/1(in)
r|i|b|s valueref
3 0 -1 r 0
4 0 -1 r 2
5 0 -1 r 1
0 1 1 r 2
1 1 1 r 0
2 1 1 r 1
6 1 -1 r 3
0 2 1 r 1
1 2 1 r 0
2 2 -1 r 3
5 2 1 r 2
0 3 -1 r 3
1 3 1 r 0
2 3 1 r 1
4 3 1 r 2
0 4 1 r 1
1 4 -1 r 3
2 4 1 r 0
3 4 1 r 2
end
```

```
cycles #prior 0(no)/1(yes)
0 0
1 1
2 0
end
```

Bei diesem Beispiel liefert unter den Verfahren mit fester Schrittweite nur das Newtonverfahren richtige Ergebnisse (siehe Abbildung 5).

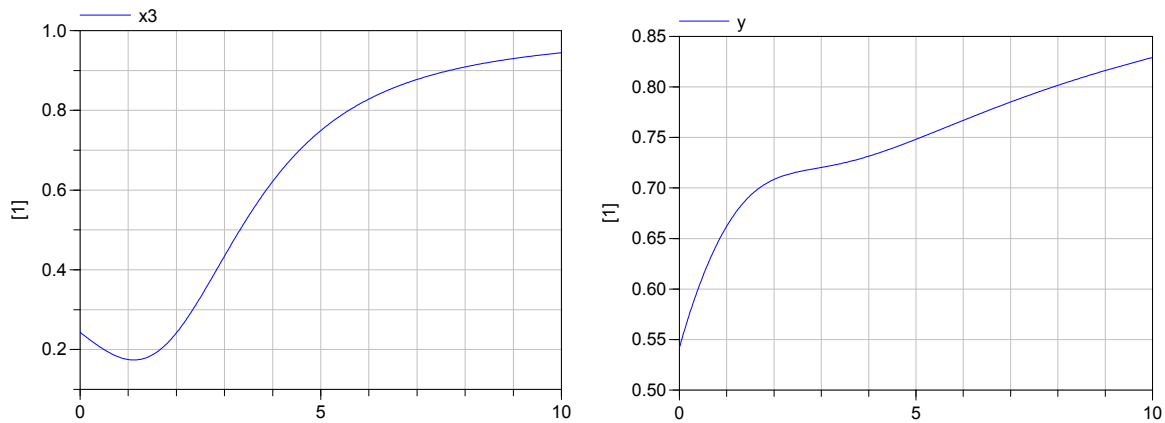


Abbildung 5: Lösungen y und x3 von Beispiel 5.1 (Darstellung über der Zeit in s)

5.2 Geregeltes rotatorisches System

Es werden zwei SimulationX-Slaves (rotatorisches System und Steuerung) und eine einfache C-Funktion mit dem EAS-Master gekoppelt. Das nicht zerlegte Gesamtsystem, dargestellt in Abbildung 6 als SimulationX Modell, ist ein einfaches geregeltes rotatorisches System, dessen Regler durch folgende „Geschwindigkeitsfunktion“ angesteuert wird:

$$f(t) = \begin{cases} 100rpm & 0.2s < t < 1s \\ 0 & \text{sonst} \end{cases}$$

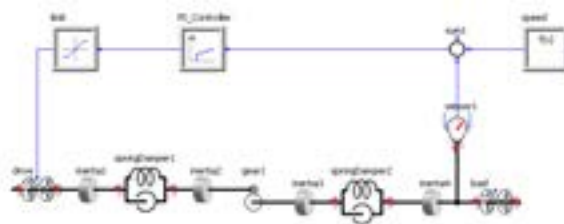


Abbildung 6: Komplettes SimulationX Modell

Dieses Modell wird in drei FMUs unterteilt (Abbildung 7), zwei SimulationX-FMUs für Regler und System und eine C-Funktion-FMU für die Eingabefunktion f(t) (Speed).

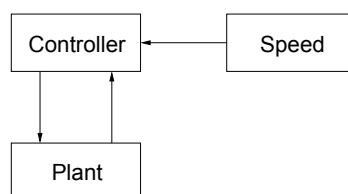


Abbildung 7: Kopplung von drei FMUs

Die FMUs werden mit dem EAS Master gekoppelt unter Verwendung des einfachen Algorithmus‘ ohne Iteration und mit fester Kommunikationsschrittweite von 10^{-3} s. Das Ergebnis der gekoppelten Simulation das und der Referenzsimulation des nicht zerlegten Modells sind in Abbildung 8 dargestellt. Es ist zu erkennen, dass beim originalen Modell eine

kleine, sehr schnell abklingende Oszillation auftritt, nachdem bei 1s die vorgegebene Geschwindigkeit auf Null gesetzt wird. Im Gegensatz dazu entsteht beim gekoppelten Modell eine größere Oszillation, die nicht abklingt. Wurde eine größere Kommunikationsschrittweite genutzt, erhöhte sich die Amplitude dieser Oszillation.

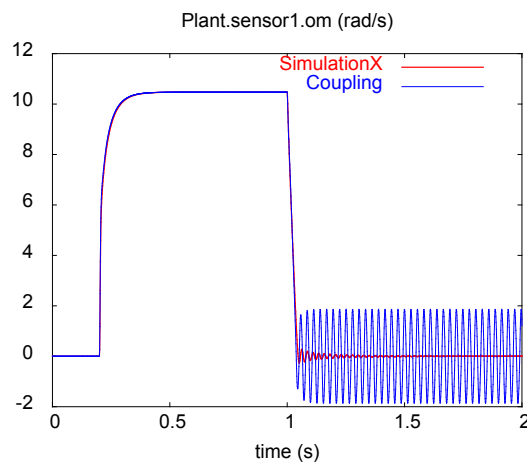


Abbildung 8: Drehzahl bei Simulatorkopplung (stark oszillierend) und ohne Kopplung

Die Konfigurationsdatei des Masters lautet:

```
nval          4
nsim          3
tstart        0
tend          2
tstepmax      0.001
tstepstart    0.001
MasterMode    1
MasterDebug   1
OutputGnuplot 1
it_max_steps  1000
it_tol_abs    1e-6
it_tol_rel    1e-4
sz_tol_abs    1e-4
sz_tol_rel    1e-4
simulator     0  Speed_fmu
simulator     1  Controller_fmu
simulator     2  Plant_fmu
```

```
graph #val #sim -1(out)/1(in)
r|i|b|s valueref
0  0  -1  r   0
0  1   1  r  536870913
1  1  -1  r  805306368
1  2   1  r  536870912
2  1   1  r  536870912
2  2  -1  r  805306369
3  2  -1  r  805306368
end
```

```
priority #sim priority
0  0
1  1
2  1
end
```

```
cycles #prior 0(no)/1(yes)
0  0
1  1
end
```

5.3 Feder-Masse-System

Ein gedämpftes Feder-Masse-System (siehe Abbildung 9) mit zwei Massen soll einen Teil eines Fahrzeuges (Rad und Chassis) beschreiben. Das System wird in Rad- und Chassis-Teil zerlegt. Beide Teile werden getrennt jeweils mit Dymola simuliert, und zwischen den Systemen werden Weg und Geschwindigkeit der Massen ausgetauscht.

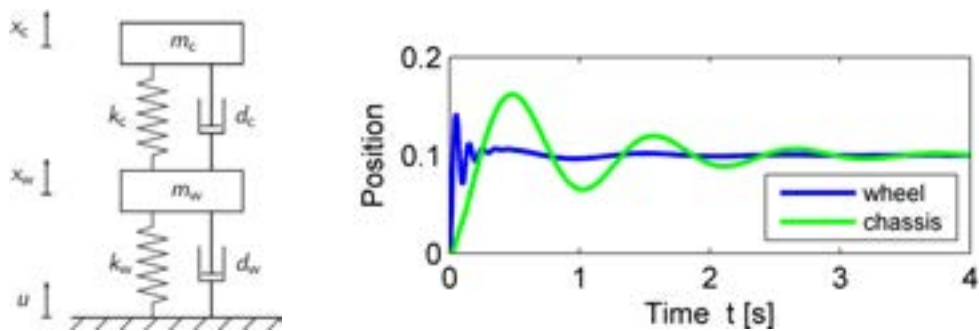


Abbildung 9: Rad-Chassis-System und Simulationsergebnis für die Auslenkungen von Rad und Chassis

An diesem Beispiel wurden bisher vor allem die Master-Algorithmen zur Kommunikationsschrittweitensteuerung basierend auf der Richardson-Extrapolation erprobt. Abbildung 10 zeigt in der unteren Kurve die Änderung der Kommunikationsschrittweite in Abhängigkeit von der Zeit, die mit abklingender Amplitude der Lösung (vergleiche Abbildung 9) tendenziell ansteigt. Übersteigt der geschätzte Fehler des aktuellen Kommunikationsschrittes die Genauigkeitsforderungen (d.h. $EST/TOL > 1$ in der oberen Kurve in Abbildung 10), so führt dies zum Verwerfen und Wiederholung des Kommunikationsschrittes mit reduzierter Kommunikationsschrittweite. Dies hat zur Folge, dass die Genauigkeitsforderung erfüllt werden kann.

Die Bestandteile des Konfigurationsfiles für den Master sind nachfolgend angegeben. Der Master-Mode 4 kennzeichnet das Verfahren der Richardson-Extrapolation.

```
nval          4
nsim          2
tstart        0
tend          4
tstepmax      0.001
tstepstart    0.001
MasterMode    4
MasterDebug   1
OutputGnuplot 1
it_max_steps  0
it_tol_abs    1e-4
it_tol_rel    1e-4
sz_tol_abs    1e-4
sz_tol_rel    1e-4
simulator     0  QC_ChassisSystem_fmu
simulator     1  QC_WheelSystem_fmu
```

```
graph #val #sim -1(out)/1(in)
r|i|b|s valueref
0  0  1      r 352321536
1  0  1      r 352321537
2  0 -1      r 335544320
3  0 -1      r 335544321
0  1 -1      r 335544320
1  1 -1      r 335544321
2  1  1      r 352321536
3  1  1      r 352321537
end
```

```
priority #sim priority
0  0
1  0
end
```

```
cycles #prior 0(no)/1(yes)
0  1
end
```

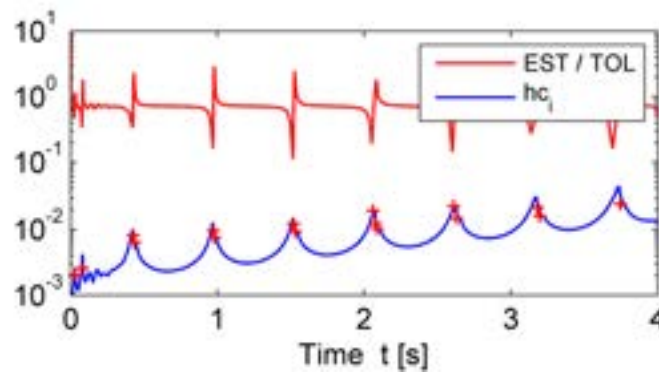


Abbildung 10: Rad-Chassis-System: Kommunikationsschrittweite, Fehlerschätzer und verworfene Schritte (rotes „x“) in Abhängigkeit von der Simulationszeit.

6 Zusammenfassung und Ausblick

Für die Simulatorkopplung zur Simulation von partitionierten Systemen wurde der Prototyp eines Masters vorgestellt, der mittels FMI mit Slaves kommuniziert. Die Slaves sind dabei Bestandteil von Functional-Mockup-Units (FMUs), die das zu simulierende Teilsystem, den zugehörigen Simulator und im FMI 1.0 standardisierte Zugriffs- und Steuerfunktionen enthalten.

Der Master ist zu Forschungszwecken eingerichtet, bietet mehrere Koppelalgorithmen an und kann zur Erprobung weiterer Algorithmen genutzt werden. Neben der Kopplung von Simulatoren kann er auch zum Test von FMUs eingesetzt werden.

Für eine erweiterte, ggf. kommerzielle Nutzung des Masters sind mehrere Verbesserungen notwendig:

- Unterstützung des vollständigen FMI 1.0 und bei Verfügbarkeit FMI 2.0
- Automatische Auswertung des Verbindungsgraphen
- Ausarbeitung weiterer Kopplungsalgorithmen in Abhängigkeit von den Eigenschaften der einbezogenen Simulatoren
- Unterstützung der einfachen Implementierung von nutzerspezifischen Algorithmen
- Erhöhung der Bedienbarkeit, Dokumentation des erreichten Standes
- Intensive Erprobung

Mit dem Master wurde ein Beitrag zur verbreiteten Anwendung des FMI geschaffen und Wege zu erweiterten Erprobung und Anwendung der Simulatorkopplung geleistet.

Dank

Diese Entwicklung wurde vom BMBF innerhalb des ITEA2-Projektes MODELISAR gefördert. Für die Mitwirkung an Teilaufgaben gebührt der Dank T. Blochwitz (ITI GmbH) sowie D. Henriksson und P. Nilsson (Dassault Systèmes).

Literatur

- [1] Bastian, J.; Clauss, C.; Wolf, S.; Schneider, P.: Master for Co.Simulation Using FMI. 8th Modelica Conference, Dresden, 2011
- [2] Wolf, S.; Blochwitz, T.: Master Slave Simulator Coupling. ITI-Symposium, Dresden, 24./25.11.2010
- [3] Arnold, M.; Blochwitz, T.; Clauß, C.; Neidhold, T.; Schierz, T.; Wolf, S.: FMI-for-CoSimulation. 1st International Conference on Multiphysics Simulation, Bonn, 2010
- [4] FMI. The Functional Mockup Interface. <http://www.functional-mockup-interface.org/>.
- [5] Kübler, R.: Modulare Modellierung und Simulation mechatronischer Systeme. Fortschritt-Berichte VDI Reihe 20, Nr. 327. VDI-Verlag GmbH, Düsseldorf, 2000
- [6] Gnuplot: <http://www.gnuplot.info/>
- [7] OpenMP: <http://www.openmp.org>
- [8] SimulationX: <http://www.simulationx.com>
- [9] Dymola: <http://www.dynasim.se>

Eine virtuelle Simulationsumgebung als Prüfplatz für Hochvoltbatterie-Steuergeräte

Matthias Nägeli, Volkswagen AG

matthias.naegeli@volkswagen.de

Dr. Dirk Lichtenthäler, Volkswagen AG

dirk.lichtenthaeler@volkswagen.de

Förderung durch BMBF

Förderkennzeichen: 01IS08002

Zusammenfassung

Ziel des Projektes war die Schaffung eines virtuellen HiL-Prüfplatzes für ein Hochvolt-Batteriesteuergerät, der vergleichbare Möglichkeiten zur Durchführung funktionaler Tests bietet wie ein entsprechender Einzel-HiL-Prüfplatz für dieses Steuergerät. Dazu musste eine integrierte Werkzeugumgebung geschaffen werden, von der aus verschiedene Simulationsumgebungen verkettet, konfiguriert und betrieben werden können, um eine verteilte Durchführung von Integrations-/Systemtests von AUTOSAR Steuerungseinheiten, entsprechend festgelegter Testprogramme, zu ermöglichen.

Im Hinblick auf den Test von AUTOSAR Steuerungseinheiten im Rahmen einer lokalen, virtuellen (simulierten) Testumgebung mit emulierten Interaktionen und Schnittstellen, zielte die angestrebte Entwicklung darauf ab, eine Adapterplattform zu schaffen, die es ermöglicht verschiedene Simulationswerkzeuge in ein Gesamtsystem zu integrieren, das nicht nur einen Modell- sondern einen Werkzeugadapter repräsentiert, der eine Fernbedienung der Simulationswerkzeuge umfasst (z.B. Modelica, Simulink, Targetlink, Dymola, SymtaS etc.).

1 Steuergeräteprüfung und Funktionsintegration

1.1 Integrationsprozess bei Volkswagen

Noch vor ca. 20 Jahren waren die in Kraftfahrzeugen eingesetzten elektronischen Steuergeräte Einzelsteuergeräte, die keine oder nur einzelne Informationen mit anderen Steuergeräten ausgetauscht haben. Entsprechend konnten diese Steuergeräte an Einzelprüfplätzen - wie etwa HiL-Prüfplätze für einzelne Steuergeräte - vollständig geprüft werden. Eine Testabsicherung der Interaktion der verschiedenen Steuergeräte war bis dato nicht notwendig.

In den letzten Jahren hat nicht nur die Anzahl der Steuergeräte im Fahrzeug zugenommen, sondern insbesondere auch der Vernetzungsgrad der Steuergeräte und damit auch die Anzahl der über mehrere Steuergeräte verteilten Funktionen, die auch als „vernetzte Funktionen“ bezeichnet werden. Für den Test von elektronischen Steuergeräten bedeutet dies, dass ein Teil der auf Steuergeräten implementierten Funktionen weiterhin an Einzelprüfplätzen getestet werden kann, und eine stark wachsende Anzahl von vernetzten Funktionen an Prüfplätzen untersucht wird, bei denen die Software verschiedener Steuergeräte im Verbund getestet werden kann.

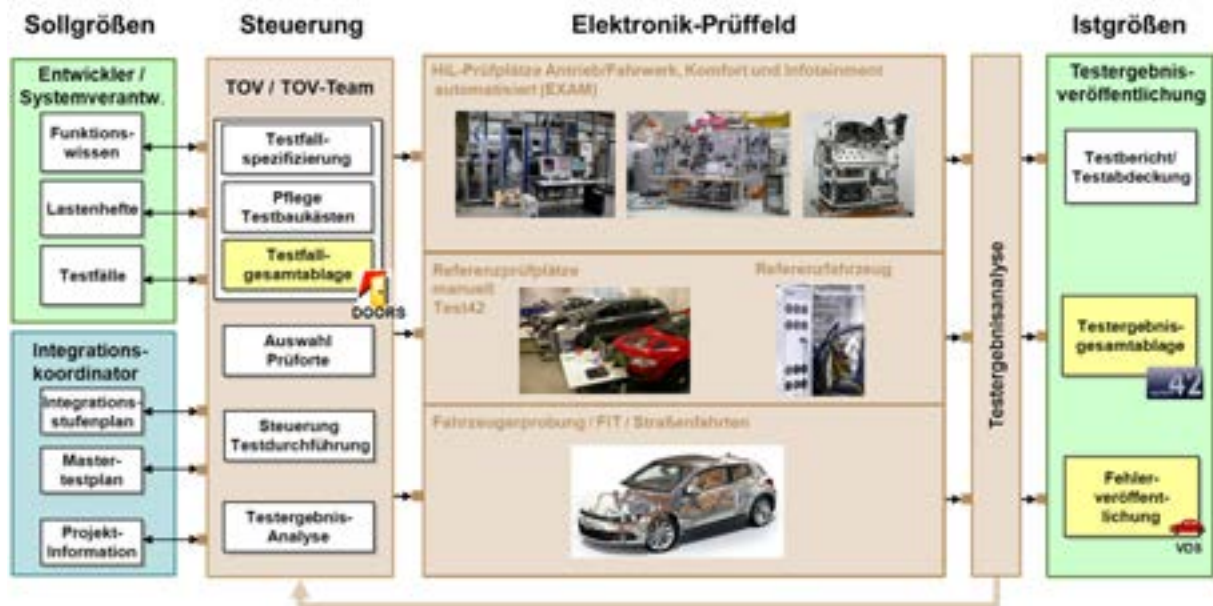


Abbildung 1: Testprozesse für funktionale Steuergeräteintegrationsprüfungen [1]

Dazu werden - wie in Abbildung 1 dargestellt – je nach Prüfaufgabe verschiedene Prüfplätze verwendet, deren gebündelte und konsolidierte Testergebnisse ein detailliertes Monitoring des Fahrzeuggesamtstatus ermöglichen. Die Auswahl der jeweiligen Prüfplätze für die erforderlichen Tests erfolgt nach deren Eignung und Verfügbarkeit aus dem zur Verfügung stehenden Elektronik-Prüffeld. [1]

Bezüglich der für die Steuergeräteprüfung eingesetzten HiL-Prüfplätze ist zwischen Einzelkomponenten HiL-Prüfplätzen und Verbund-HiL-Prüfplätzen zu unterscheiden. Für den Test von Einzelsteuergeräten werden Komponenten-HiL-Prüfplätze eingesetzt, bei denen die Kommunikation mit den übrigen Steuergeräten über eine Restbussimulation nachgebildet wird.

Im Gegensatz zu den relativ kompakten Einzelsteuergeräte-HiL-Prüfplätzen werden für die Steuergeräteintegration vorwiegend Verbund-HiL-Prüfplätze verwendet, bei denen im Prinzip die einzelnen Komponenten-HiL-Prüfplätze zu einem großen Prüfplatz integriert sind. Dabei wird die Buskommunikation, die über Gateways mit anderen Domänen erfolgt, über einen

entsprechenden Restbus nachgebildet. Als ein Beispiel für einen solchen Domänenprüfplatz ist in Abbildung 2 der Antriebs-/Fahrwerks-HiL-Prüfplatz für den Passat CC dargestellt. Die Markierungsrahmen sollen dabei andeuten, wie sich der Prüfplatz aus Einzelprüfplätzen für Motor, Getriebe, Lenkung, etc. zusammensetzt.

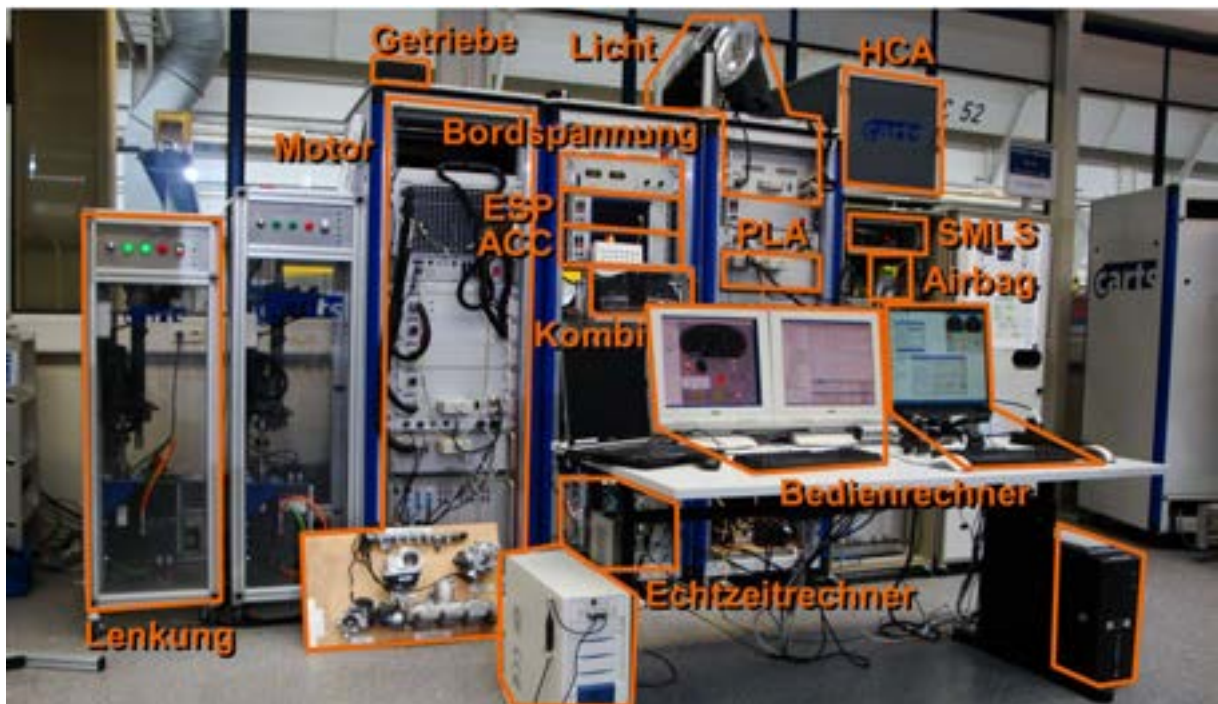


Abbildung 2: Antriebs-/Fahrwerks-HiL-Integrationsprüfplatz [1]

Neben den HiL-Prüfplätzen stehen für die Gesamtelektronikintegration noch verschiedene andere Prüfplätze mit stark unterschiedlicher Komplexität und unterschiedlichen Erprobungsschwerpunkten zur Verfügung. Funktionale Tests werden u.a. auch an sog. Referenzprüfplätzen durchgeführt. Dabei handelt es sich um Fahrzeugkarosserien, in denen der Original-Kabelbaum sowie alle im Fahrzeug verbauten Steuergeräte montiert sind. Lediglich Bauteile wie Motor, Getriebe und Bremsen werden an den Referenzprüfplätzen durch HiL-Simulation nachgebildet.

An den sog. Referenzfahrzeugen werden Themen wie Ruhestrom oder Über- und Unterspannungen untersucht. Bei den Referenzfahrzeugen handelt es sich um komplette Fahrzeuge, bei denen lediglich die Batterie durch eine spezielle Simulation ersetzt wird.

Schließlich werden beim Fahrzeugintestivtest (kurz: FIT) die Funktionen anhand von kompletten Fahrzeugen auf einem Testgelände erprobt.

2 Planung des Prüfplatz-Demonstrators und Projektablauf

Die Planung des Prüfplatz-Demonstrators konzentrierte sich zunächst auf die Bewertung der Machbarkeit und den Aufwand für die Realisierung der geplanten virtuellen HiL-Umgebung. Die überprüften Alternativen waren:

- Processor in the Loop & Debugger
- Hardware in the Loop & Debugger
- Virtuelle Maschine (just-in-sequence)
- Virtuelle Maschine mit Synchronisation an den Schnittstellen

Entschieden wurde schließlich folgende Vorgehensweise:

- Realisierung einer SiL-Simulationsarchitektur mit einer funktionalen Simulation der ECU und unter Verwendung der existierenden Echtzeitmodelle
- Die Test-Prozeduren der Volkswagen Testautomatisierung EXAM werden integriert und evaluiert.

Virtueller HiL: Simulationsarchitektur

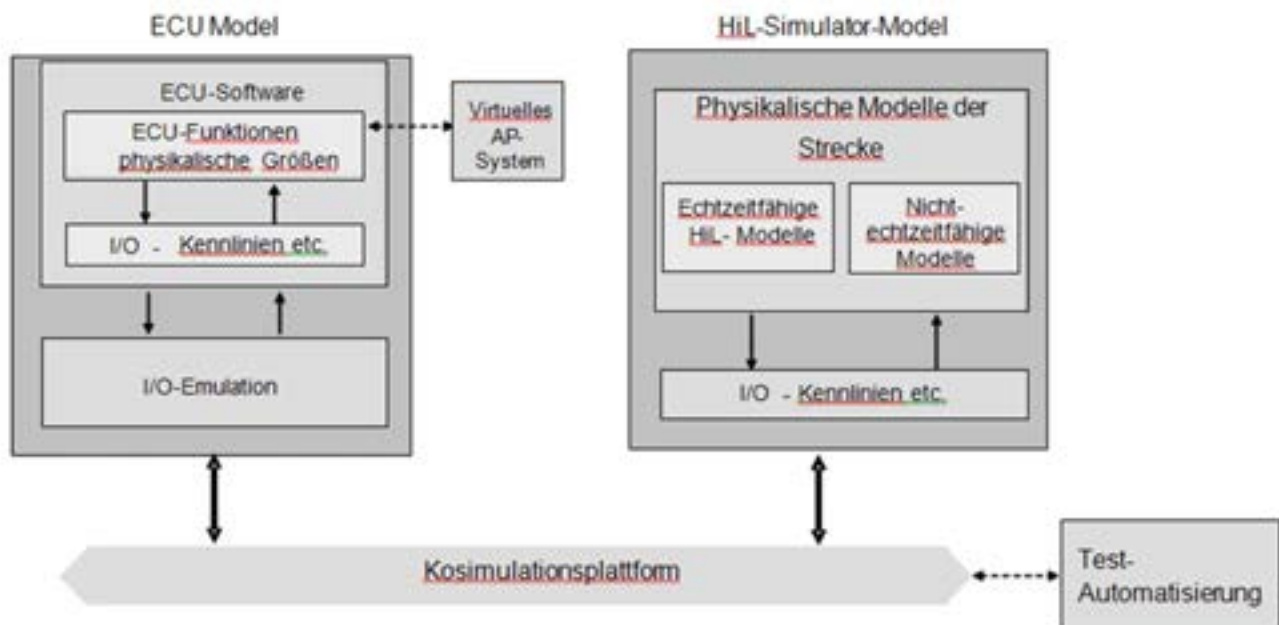


Abbildung 3: Simulationsarchitektur des virtuellen HiL-Prüfplatzes

Für die Umsetzung des Kosimulationsdemonstrators mussten zunächst die Anforderungen für eine Schnittstelle zwischen ATEGO ACE und den speziellen Testautomatisierungswerkzeugen bei Volkswagen ermittelt werden. Danach konnte in Zusammenarbeit mit dem Softwarehersteller der Testautomatisierung eine Schnittstelle zwischen ATEGO ACE und dem Testautomatisierungswerkzeug EXAM erstellt werden.

Für die Ausführung der Steuergerätesoftware auf der Kosimulationsplattform wurde ein Generator zur Erzeugung von auf der Kosimulationsplattform ausführbarem Code aus den Steuergeräte Source-Files erstellt.

Ein Adapter für die Anbindung von ATEGO ACE an die Werkzeuge zur Steuergeräteapplikation bei Volkswagen wurde ebenfalls entwickelt.

Danach konnten Testmöglichkeiten auf der Kosimulationsplattform intensiv evaluiert und der Vergleich mit anderen Testeinrichtungen wie HiL-Plattformen durchgeführt werden. Zu den Vergleichsplattformen gehörte sowohl ein Einzel-HiL-Prüfplatz als auch Integration-HiL-Prüfplatz, an dem fast alle Steuergeräte des Fahrzeugs als Echtteile angeschlossen sind.

Die Realisierung einer Prozessor-Emulation auf der Kosimulationsplattform wurde geprüft. Aufgrund des zum erwarteten Nutzen sehr hohen Zusatzaufwandes wurde aber auf eine Umsetzung verzichtet.

3 Softwaretools und Standards

3.1 Kosimulationsplattform ATEGO ACE

Zentrales Element dieser “Virtuellen HiL-Plattform” ist die Kosimulationsplattform ATEGO ACE der Firma ATEGO.

ATEGO ACE ist eine skalierbare Integrationsplattform für virtuelle Komponenten in der Automobil-, Luftfahrt- und Verteidigungsindustrie. Mit Atego ACE kann eine virtuelle Integration von Software-Funktionen und Software-Komponenten als auch von heterogenen Simulationen unterschiedlicher Physik-Bereiche realisiert werden. Dabei kann eine virtuelle Integration von verteilten Software-Systemen auf Standard-PCs realisiert werden, während der Zielcode für die Echtzeitplattform beibehalten wird. [2]

3.2 Testautomatisierung EXAM

Für die Automatisierung der Tests wurde auf das Testentwicklungswerkzeug EXAM zurückgegriffen.

EXAM definiert eine umfassende Methodik auf Basis der UML, mit der Testfälle dargestellt, durchgeführt und ausgewertet werden können. Testabläufe können dabei grafisch in Sequenzdiagrammen modelliert werden. [3]

EXAM wurde 2006 von der Audi AG, der Volkswagen AG und der MicroNova AG gemeinsam entwickelt und hat sich seitdem als konzerneinheitliche Testautomatisierung für HiL-Simulatoren im Volkswagen Konzern etabliert. EXAM wurde seitdem kontinuierlich weiterentwickelt, so dass ein komplexes Tooling mit zahlreichen Bibliotheken zur Verfügung steht. [3]

EXAM war neben der bereits etablierten Anwendung zur Automation von Tests an HiL-Simulatoren auch für Einsätze in der Embedded-Entwicklung und der Software-in-the-Loop-Simulation(SiL) konzipiert. [3]

Eine Anbindung an die verwendete Kosimulationsplattform war aber zu Projektstart nicht vorhanden.

3.3 Offene FMI aus MODELISAR

Die Zielsetzung des im Rahmen des Projektes MODELISAR entwickelten standardisierten Functional-Mockup-Interface (FMI) ist die Kopplung von zwei oder mehr Simulationstools in einer Kosimulations-Umgebung. Der Datenaustausch zwischen den Subsystemen ist auf diskrete Datenaustauschzeitpunkte beschränkt. In der Zeit zwischen den Austauschpunkten, werden die Subsysteme unabhängig voneinander mit individuellen Solvern berechnet. Ein Master- Algorithmus steuert den Datenaustausch zwischen den einzelnen Subsystemen und die Synchronisation aller Slave-Simulations-Solver. [4]

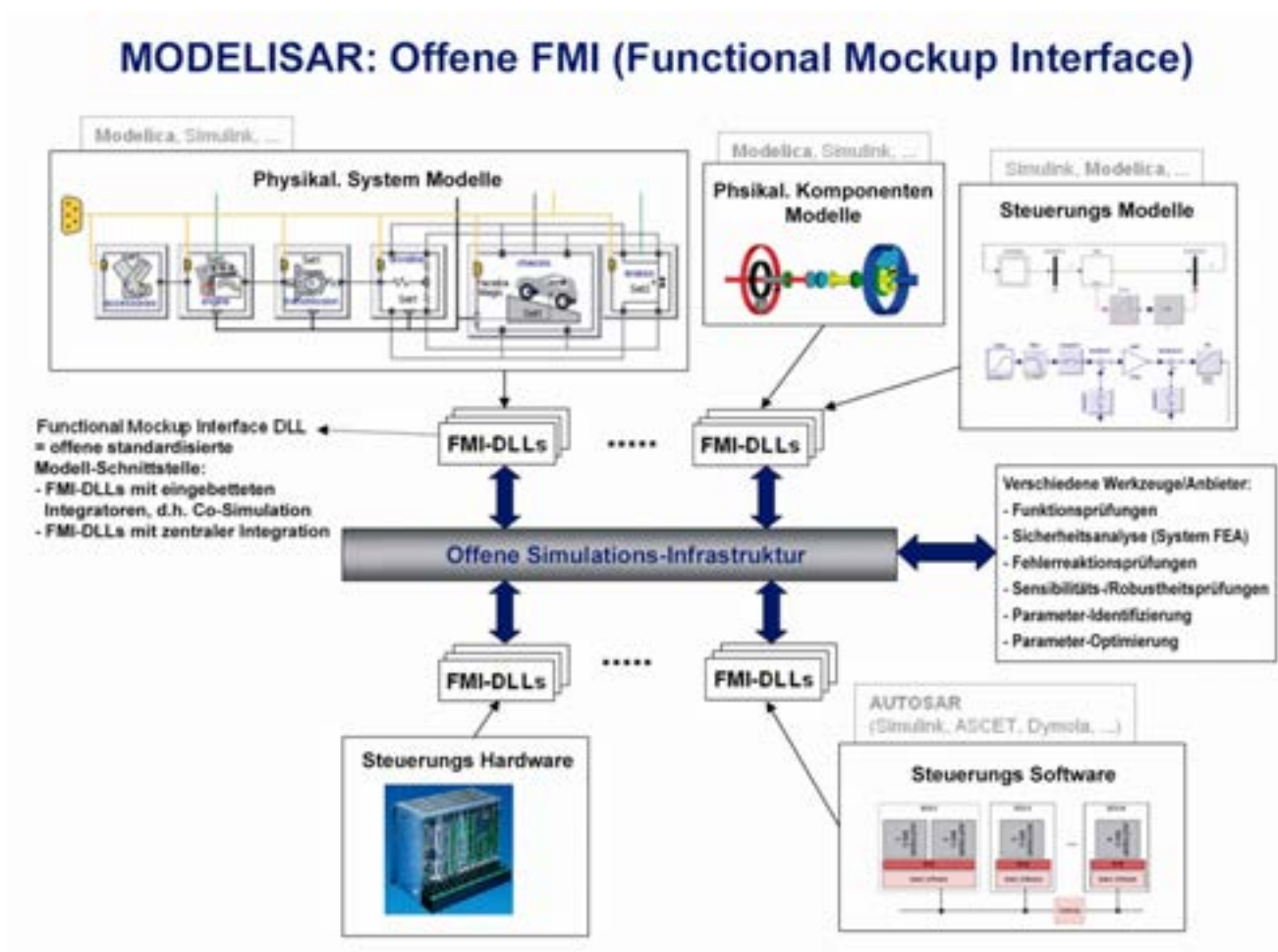


Abbildung 4: Offene FMI aus MODELISAR [5]

Alle Informationen über die Slaves, die für die Kommunikation in der Kosimulationsumgebung relevant sind, werden durch ein spezifisches XML-file bereitgestellt. Hauptsächlich beinhaltet dies einen Satz von Capability-Flags, um die Leistungsfähigkeit des Slaves zu charakterisieren und komplexe Master-Algorithmen zu unterstützen (z.B. durch die Verwendung von variablen Kommunikationsschrittweiten, Signalextrapolationen höherer Ordnung etc.). [4]

Das im Rahmen von Modelisar entwickelte Functional-Mockup-Interface ist frei verfügbar und steht unter URL: <http://www.modelisar.com/fmi.html> als Download zur Verfügung.

4 Realisierung des Softwaredemonstrators

4.1 Anwendungsbeispiel

Der realisierte Softwaredemonstrator ermöglicht den Test der Software eines elektronischen Steuergerätes, unter Bedingungen, die mit denen an einem HiL-Prüfplatz vergleichbar sind. Als Anwendungsbeispiel wurde eine Kosimulationstestplattform für ein Steuergerät für die Hochvoltbatterie ausgewählt. Solche Steuergeräte werden sowohl für Fahrzeuge benötigt, die ausschließlich über einen elektrischen Antriebe verfügen, als auch für Hybridfahrzeuge, die sowohl über einen Verbrennungsmotor als auch über einen elektrischen Antrieb mit einer entsprechenden Hochvoltbatterie verfügen.

4.2 Einbindung der Steuergeräte-Software

Die Steuergeräte-Software war in dem ausgewählten Anwendungsfall anhand von C-Code-Source-Files vorhanden. Die Software, welche auf der Steuergeräte-Hardware läuft, muss hierbei unterteilt werden in die Applikations-Software und in die Basis-Software.

Die Applikationssoftware übernimmt dabei die Kontrollaufgaben auf Zell- und Batterieebene, sowie die Überwachung des Hochvoltbereichs.

Die Basis-Software beinhaltet unter anderem die Bustreiber und die Schnittstelle zur Steuergeräte-Hardware.

Da an dem Testplatz eine Überprüfung des Steuergerätes aus funktioneller Sicht erfolgen sollte, war die Einbindung der Applikationsebene der Steuergeräte-Software ausreichend und auf eine Einbindung der Basis-Software konnte verzichtet werden.

Für die Wandung des Source-Files in eine auf der Co-Simulationsplattform ATEGO-ACE Software-Komponente, wurde von ATEGO ein entsprechendes Skript implementiert und in die Co-Simulationsplattform ATEGO-ACE integriert. Dabei wird eine spezielle Schnittstelle des Build-Prozesses verwendet.

4.3 Einbindung der Umgebungsmodelle

Als Umgebungsmodell wurde für das Hochvolt-Batterie-Steuergerät ein Modell der Hochvolt-Batterie sowie eine Nachbildung der Hochvolt-Schütze eingebunden.

Das Modell, welches in Matlab/Simulink programmiert wurde, konnte über die hier durch ATEGO spezifizierte Matlab/Simulink-Schnittstelle an die Kosimulationsplattform angebunden werden.

Somit wurden Steuergeräte-Code und Umgebungsmodell über die Kosimulationsplattform gekoppelt und wurden “closed-loop” ausgeführt.

4.4 Anbindung der Testautomatisierung

Eine wichtige Zielsetzung dieses Teilprojekts war die automatisierte Ausführung von Testskripten, die für den Einsatz auf HiL-Simulatoren bereits erstellt waren. Dazu war es notwendig die bereits an zahlreichen HiL-Prüfplätzen eingesetzte Testautomatisierungs-Software EXAM an die Kosimulationsplattform anzubinden.

Daher wurde im Rahmen des Projektes in Zusammenarbeit mit den Firmen ATEGO und Micronova eine Schnittstelle umgesetzt, welches es ermöglicht mit dem Testautomatisierungstool EXAM auf das Testsystem bestehend aus Batterie-Steuergeräte-Software und Umgebungsmodell zuzugreifen. Dadurch wurde es möglich die gleichen funktionalen Tests an dem Kosimulationssystem durchzuführen, wie an einem entsprechenden HiL-System.

In Abbildung 5 ist das System mit den realisierten Toolschnittstellen schematisch skizziert.

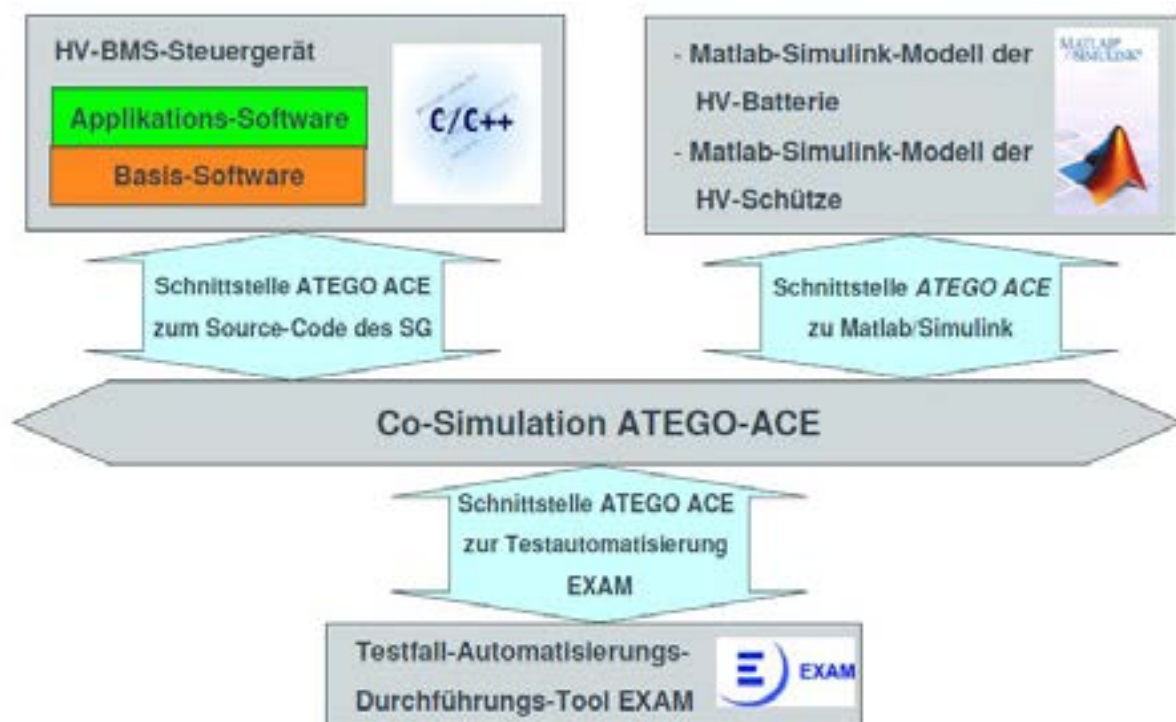


Abbildung 5: Schematische Darstellung der realisierten Toolschnittstellen am Softwaredemonstrator

4.5 Anwendung des Prüfplatzes

Mit Hilfe des Softwaredemonstrators wurden bisher folgende beiden Anwendungsfälle untersucht:

1. Funktionstests

Beim Funktionstest wird eine automatisierte Überprüfung der in der Applikations-Software umgesetzten Steuergeräte-Funktionen durch die angebundene Testautomatisierung EXAM durchgeführt.

Dadurch wird die Möglichkeit geschaffen Testpakete vom realen HiL auf den “virtuellen HiL” zu verlagern. Durch die zusätzliche Testkapazität kann zum einen die Testtiefe erhöht werden, zum anderen können auch Tests im Entwicklungsprozess nach vorne verlagert werden. Da für die Tests nur die Applikationssoftware notwendig ist, während sowohl die Steuergeräte-Hardware als auch die Basis-Software für funktionale Tests auf der Kosimulationsplattform nicht notwendig sind, kann eine funktionale Steuergeräteabsicherung erfolgen, bevor dies an den HiL-Testeinrichtungen möglich ist.

2. CAN-Nachbildung

Bei der CAN-Nachbildung können CAN-Aufzeichnungen aus Versuchsfahrten mit realen Fahrzeugen eingespeist werden. So können Auffälligkeiten nochmal am “Offline-System” nachanalysiert werden. Interne Steuergerätegrößen können genau betrachtet werden, was eine Identifizierung der Problematiken erleichtert.

Der eingebundenen Batterie-Steuergeräte-Software wird dabei quasi eine Restbussimulation aus realen Fahrzeugdaten zur Verfügung gestellt.

5 Ergebnisse

Im Hinblick auf die stetig wachsende Anzahl von zu prüfenden verteilten Funktionen im Kraftfahrzeug sind zukünftig über die bereits bestehenden Testeinrichtungen neue Testeinrichtungen erforderlich, um eine hinreichende Testtiefe zu gewährleisten. Dabei ist auch zu berücksichtigen, dass HiL-Einrichtungen die Tests ausschließlich in Echtzeit ausführen. Unter Berücksichtigung von Rüstzeiten und anderen Ausfallzeiten ist die Netto-Testzeit an HiL-Simulatoren auf weniger als 24 Stunden pro Tag begrenzt.

Da mit Hilfe der Software-in-the-Loop Simulation Tests auch schneller als in Echtzeit ausgeführt werden können, kann die Netto-Testzeit auch deutlich über 24 Stunden pro Tag gesteigert werden.

Außerdem sind die Kosten für die Duplizierung von SiL-Simulatoren deutlich geringer als für HiL-Simulatoren, da die Hardware lediglich aus Standard-PCs besteht.

Bereits während der Inbetriebnahmephase wurden Synergien zwischen dem Kosimulations-Demonstrator, den Einzelsteuergeräte HiL-Systemen für das entsprechende Steuergerät sowie den Verbund-HiL-Systemen deutlich. Insbesondere die durchgängige Verwendung der Modelle bei den genannten drei Arten von Prüfplätzen führte zu einer schnelleren und effizienteren Inbetriebnahme an den übrigen Prüfplätzen, nachdem die Modelle und das jeweilige Steuergerät an einem Prüfplatz in Betrieb genommen wurden.

Der Kosimulations-Demonstrator konnte bereits produktiv in der Testfalldurchführung eingesetzt werden. Dabei stellt eine solche Kosimulationsplattform eine gegenüber einer HiL-Einrichtung relativ kostengünstige Alternative dar. Für sehr viele Tests kann statt einer HiL-Einrichtung auch die Kosimulationsplattform verwendet werden, so dass sowohl weniger HiL-Einrichtungen notwendig werden, als auch Tests zeitlich früher im Testprozess realisiert werden können. Durch die frühere Testrealisierung können Fehler in der Software früher beseitigt werden, was gemäß der über den Entwicklungsprozess exponentiell ansteigenden Kosten zur Fehlerbeseitigung, eine entsprechende Kostenersparnis ermöglicht.

Durch die Arbeiten auf dem Gebiet der Kosimulation konnte die Kosimulationstechnik bei Volkswagen weiter etabliert werden. Derzeit werden bereits einige neue Projekte angedacht bei denen Kosimulation zur Unterstützung des Entwicklungs- und Testprozesses eingesetzt werden soll. Insbesondere bei innovativen Fahrzeugantrieben bietet Kosimulation große Möglichkeiten einer Optimierung des Gesamtsystems über die Grenzen der einzelnen Teilsysteme hinweg.

Der FMI-Standard ist nicht nur für die Realisierung von reinen Software-in-the-Loop-Systemen geeignet. Diverse Zulieferer für modellbasierte Testsysteme wie Hersteller für HiL-Systeme planen oder realisieren die Umsetzung eines Functional-Mockup-Interface in ihren Simulationssystemen. Dadurch wird auch die Möglichkeit geschaffen, SiL-Systeme und HiL-Systeme zu koppeln. D.h. für den Verbundsystemtest, dass die Steuergeräte-Software von einigen Steuergeräte über SiL-Simulation eingebunden werden kann, während andere Steuergeräte als Echtteil über die HiL-Simulation vorhanden sind.

Literatur

[1] Müller, Sven-Oliver; Engel, Thomas; Dr. Brand, Marcus: *Entwicklungsbegleitende*

Integrationstests für Fahrerassistenzsysteme am Antriebs-/Fahrwerks-HiL, VDI-

Tagung: Elektronik im Fahrzeug, Baden-Baden, VDI-Berichte 2075, 2009

- [2] ATEGO Systems GmbH: URL: <http://www.atego.com/products/atego-ace/>
[Stand 2011-01-10]
- [3] MicroNova AG: URL: <http://www.exam-ta.de/> [Stand 2011-01-10]
- [4] Modelisar Konsortium: URL: <http://www.modelisar.com/fmi.html>
[Stand 2011-01-10]
- [5] Modelisar Konsortium: *Vorhabensbeschreibung MODELISAR:*
Von der System-Modellierung zur Software im Fahrzeugeinsatz, BMBF-Förderantrag,
Version 4 vom 9.5.2008



System Architecture

Rapid Control Prototyping

ECU Autocoding

HIL Testing

Fahrerassistenzsysteme denken mit – Getestet mit dSPACE Simulatoren



Um innovative Fahrerassistenz- und aktive Sicherheitssysteme auf die Straße zu bringen, ist die Absicherung der Serienreife ein entscheidender Schritt. dSPACE bietet das volle Programm: von virtuellen Testfahrten auf Basis realer Straßenkarten über Lösungen zum Test von kamerabasierten Systemen bis hin zu offenen Simulationsmodellen für Fahrzeug, Sensorik, Fahrbahn und Umgebung. Mit dSPACE Hardware-in-the-Loop (HIL)-Simulatoren sichern Sie sich Ihren entscheidenden Vorsprung.

Zukünftige Fahrzeuge denken mit ... und Sie?

Embedded Success

dSPACE

Decoupling Test Cases from Real and Virtual Test Systems with ASAM HIL API



Dr. Rainer Rasche, dSPACE GmbH

Dr. Dietmar Neumerkel, Daimler AG

Workshop der ASIM/GI-Fachgruppen Simulation technischer Systeme (STS) und Grundlagen und Methoden in Modellbildung und Simulation (GMMS), Wolfenbüttel 23./24.02.2012

Agenda

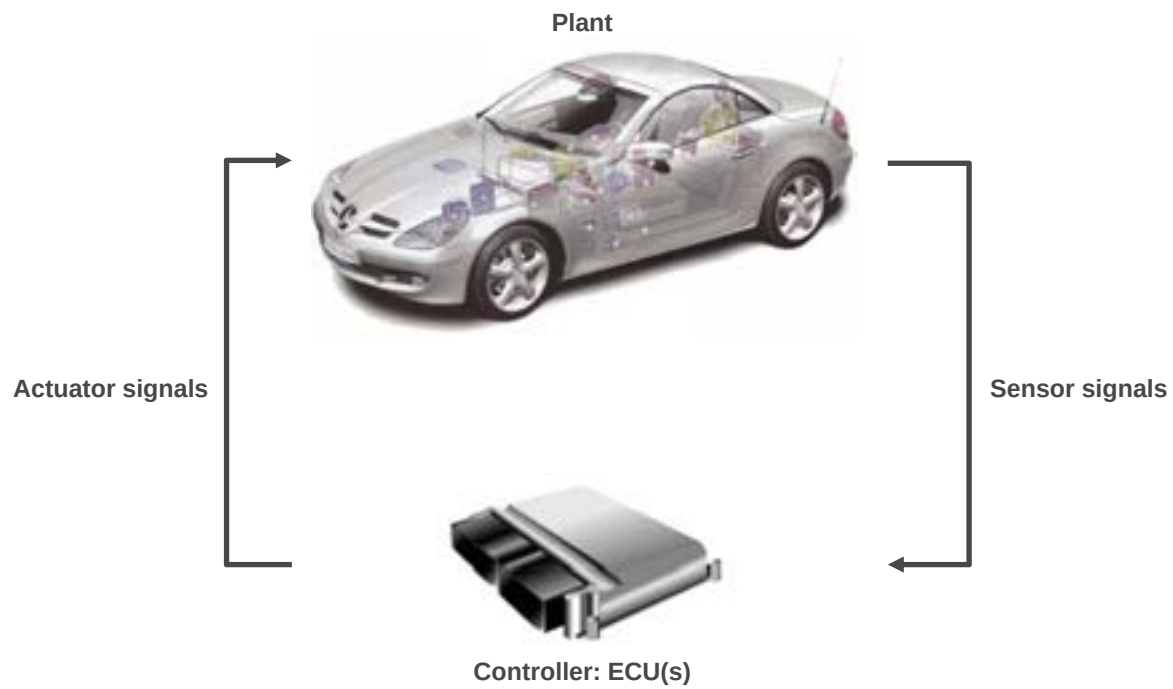
Introduction: HIL Simulation

Testautomation ASAM HIL API 1.0

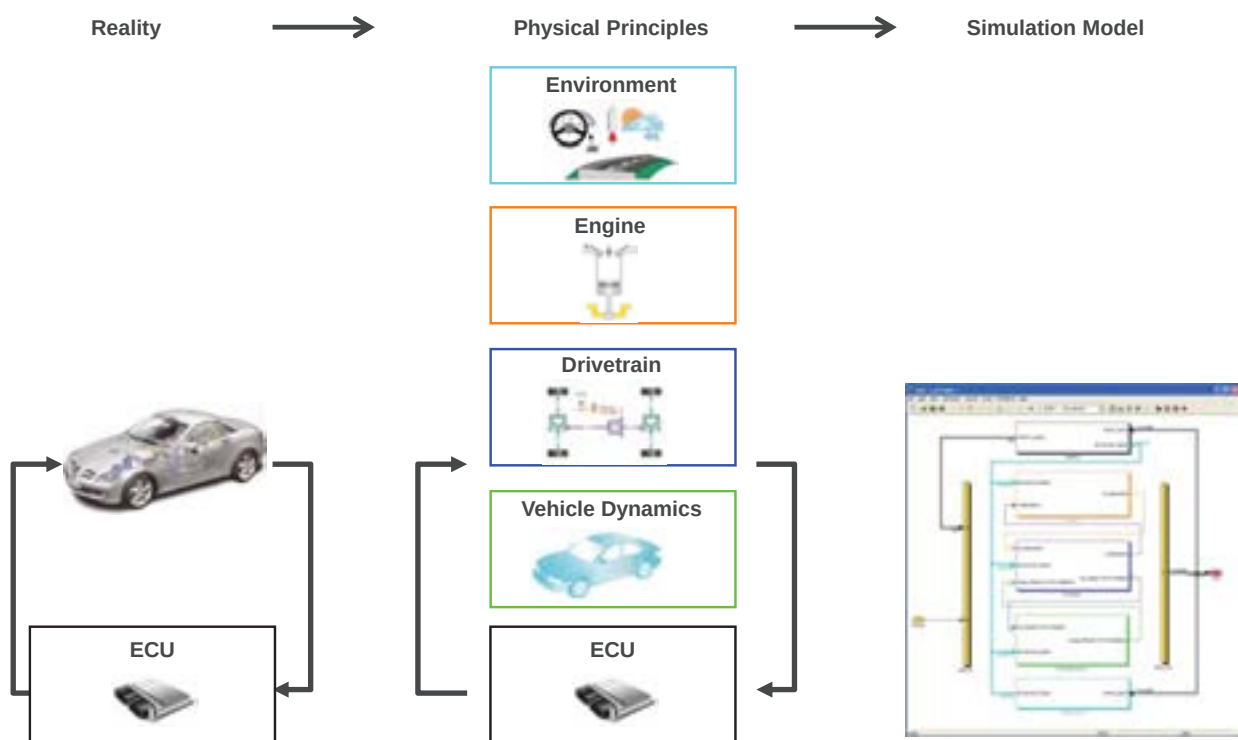
Improvements in ASAM HIL API 2.0

Example

Summary



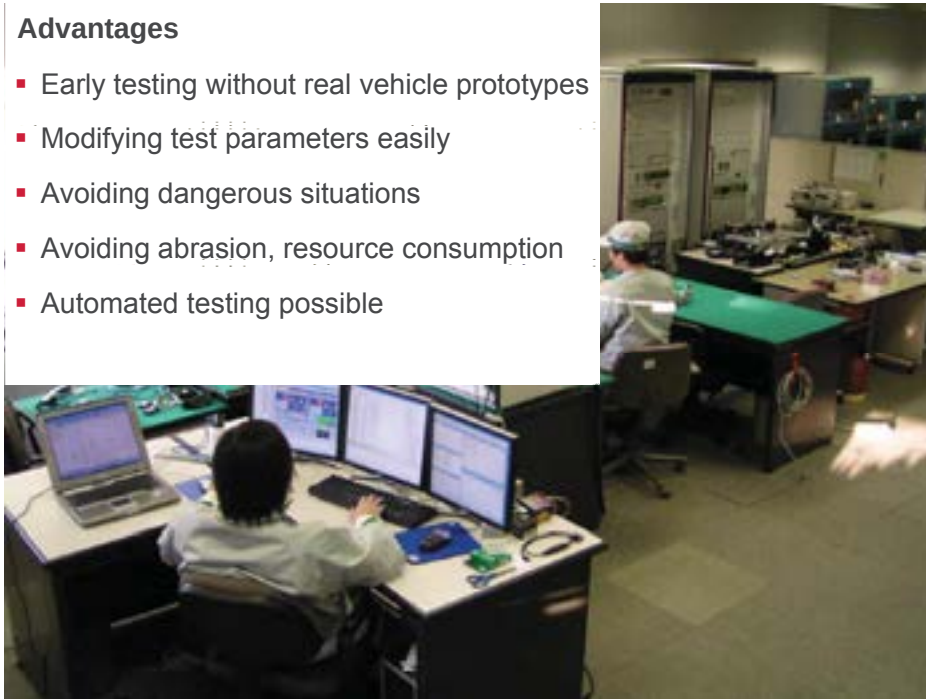
3



4

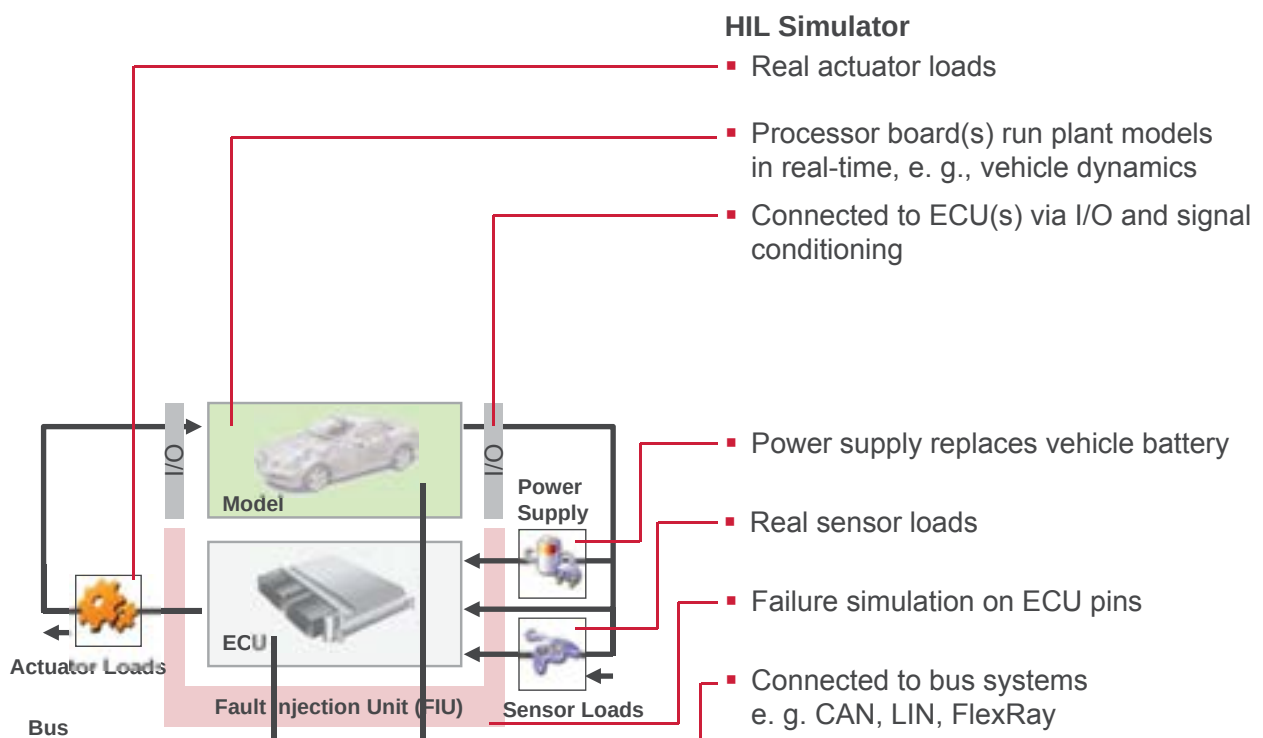
Advantages

- Early testing without real vehicle prototypes
- Modifying test parameters easily
- Avoiding dangerous situations
- Avoiding abrasion, resource consumption
- Automated testing possible

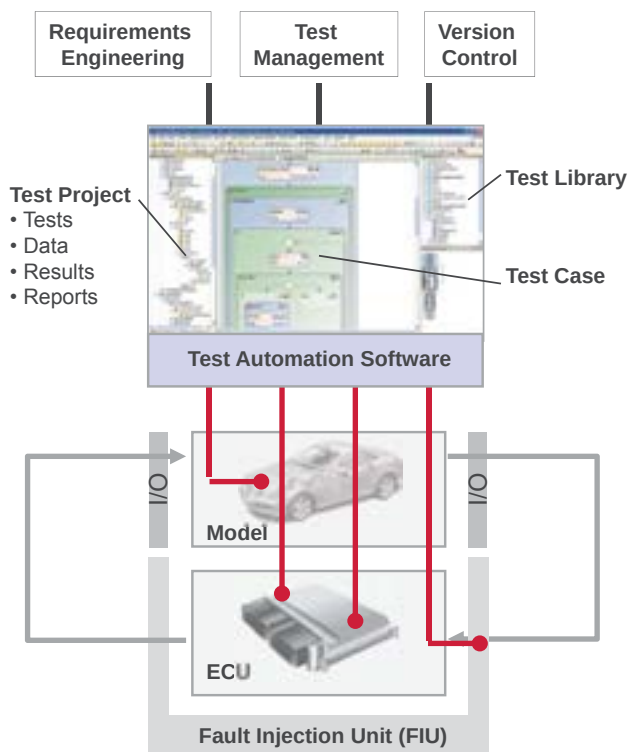


5

Overview HIL Simulation



6



Automated ECU Testing

- Repeating tests precisely and automatically as often as required
- Access to all relevant test interfaces
- State-of-the-art: Convenient PC-based test development and execution

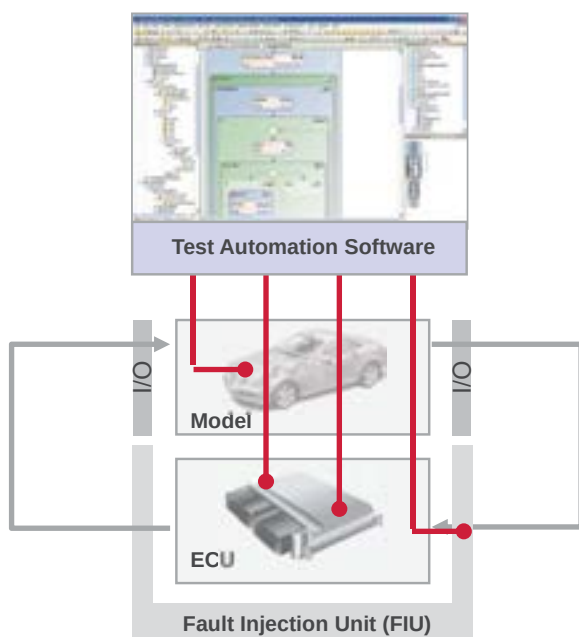
Interfaces to HIL test bench

Real-time simulation

Reality

7

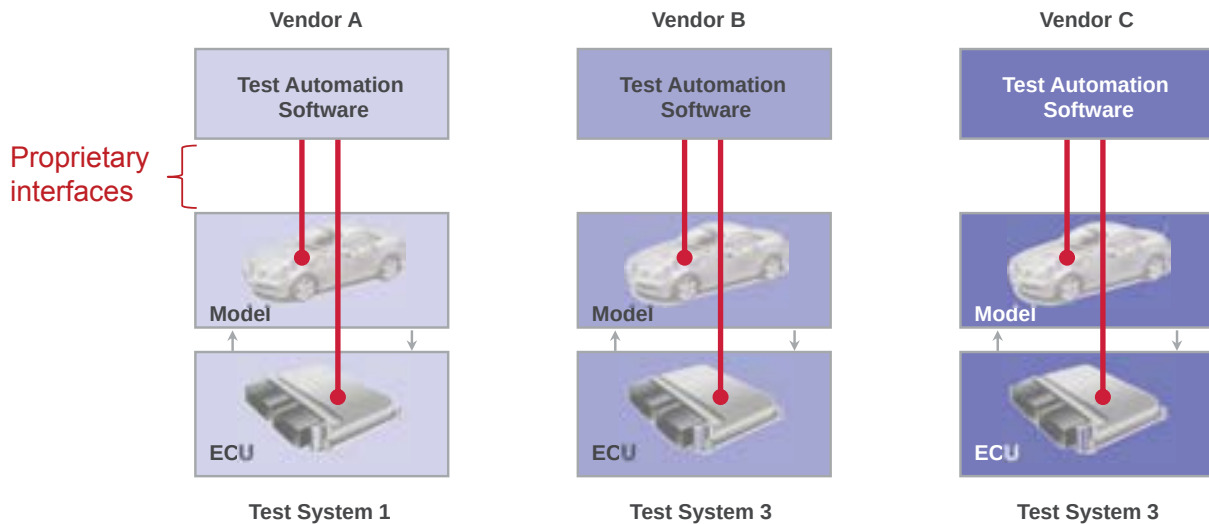
Interfaces to HIL Test Bench: Today's Situation



Today, interfaces to HIL Test benches are often proprietary.

8

Exchange of **test automation software** from different vendors and different **test systems** is **not** possible.



9

Introduction: HIL Simulation

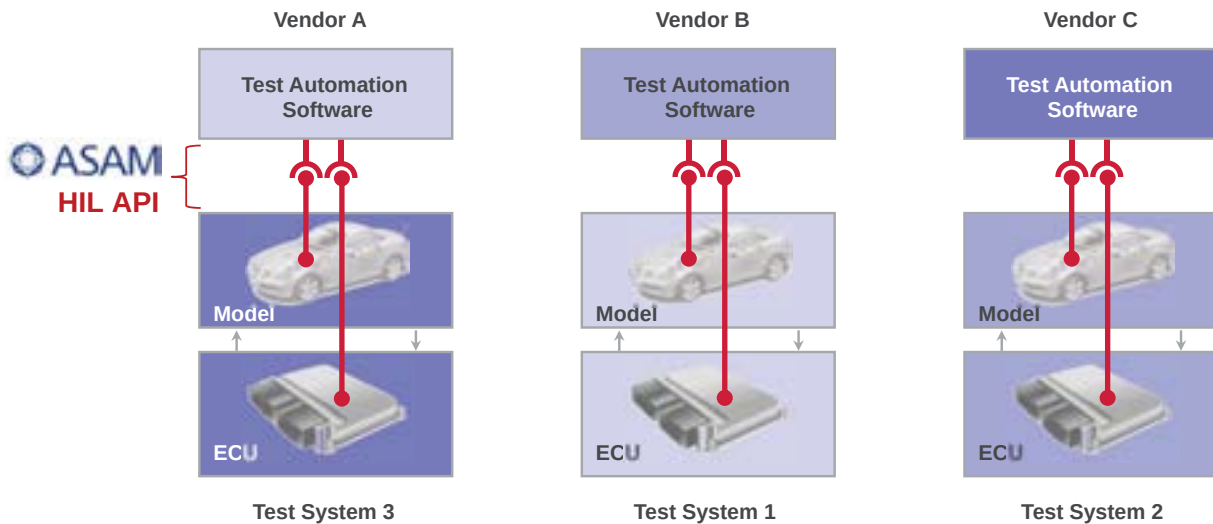
Test Automation ASAM HIL API 1.0

Improvements in ASAM HIL API 2.0

Example

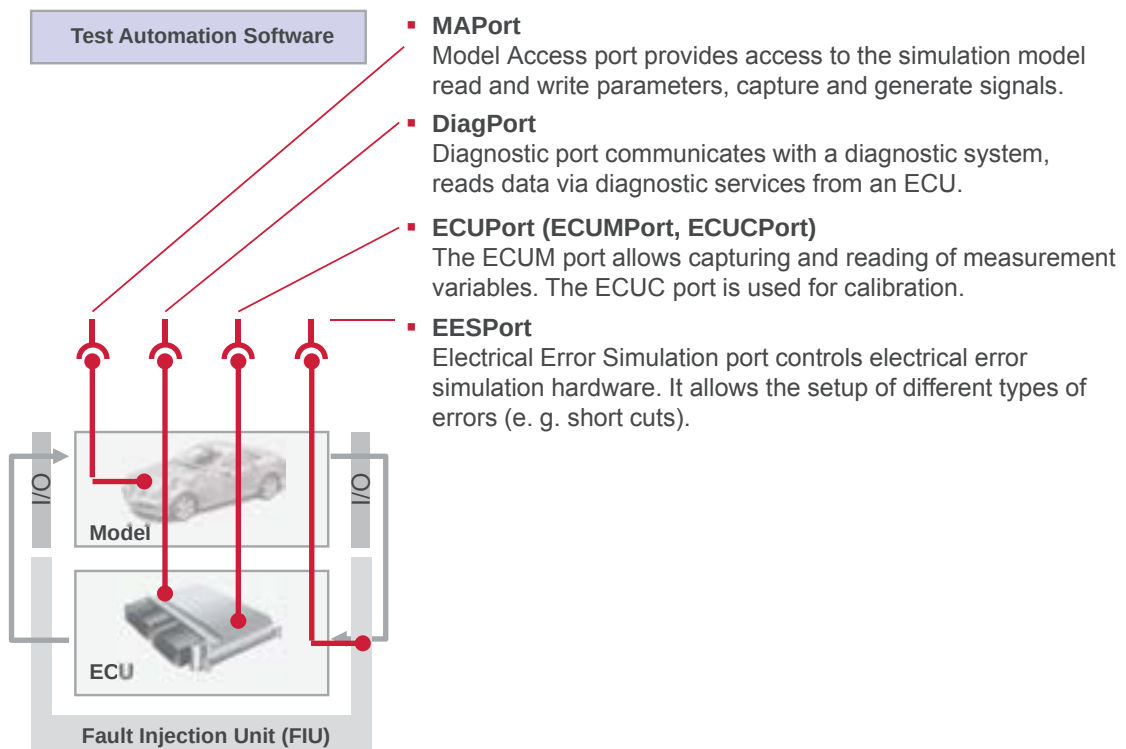
Summary

Exchange of **test automation software** from different vendors and different **test systems** is possible.

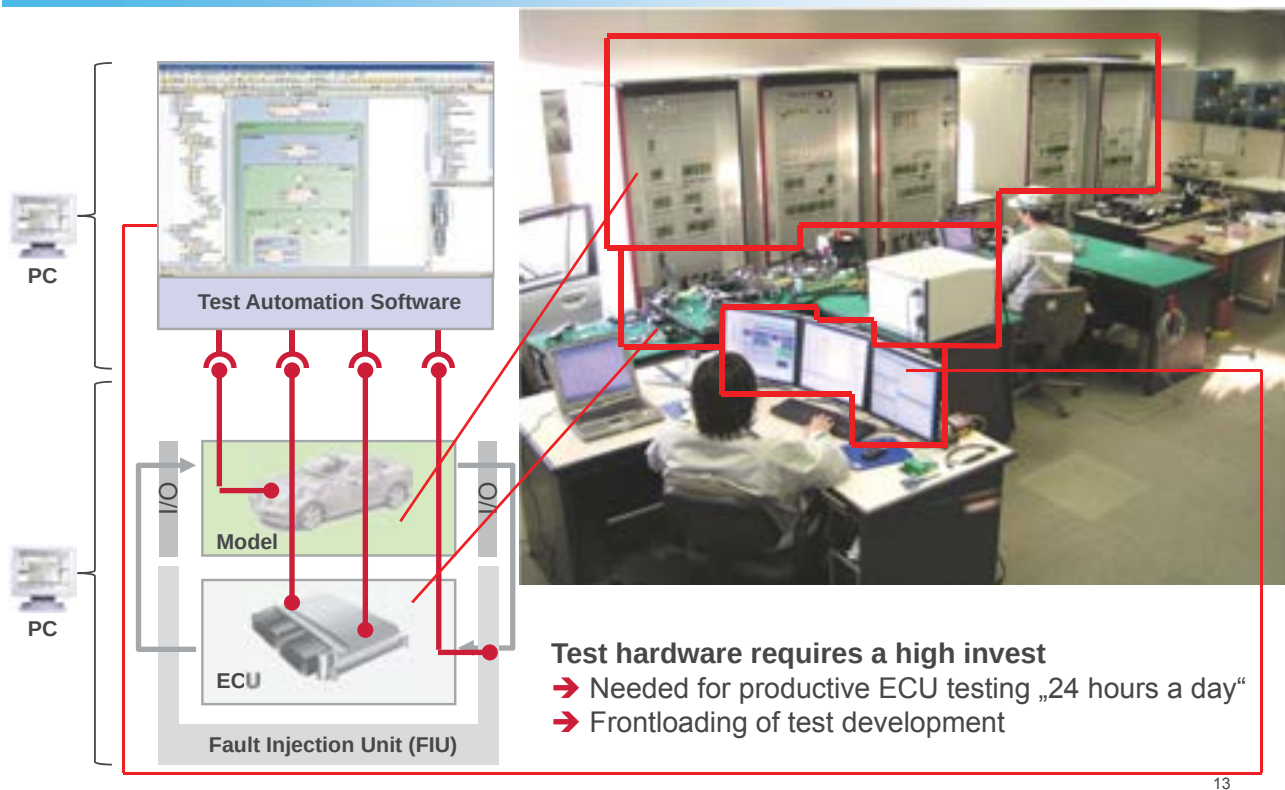


11

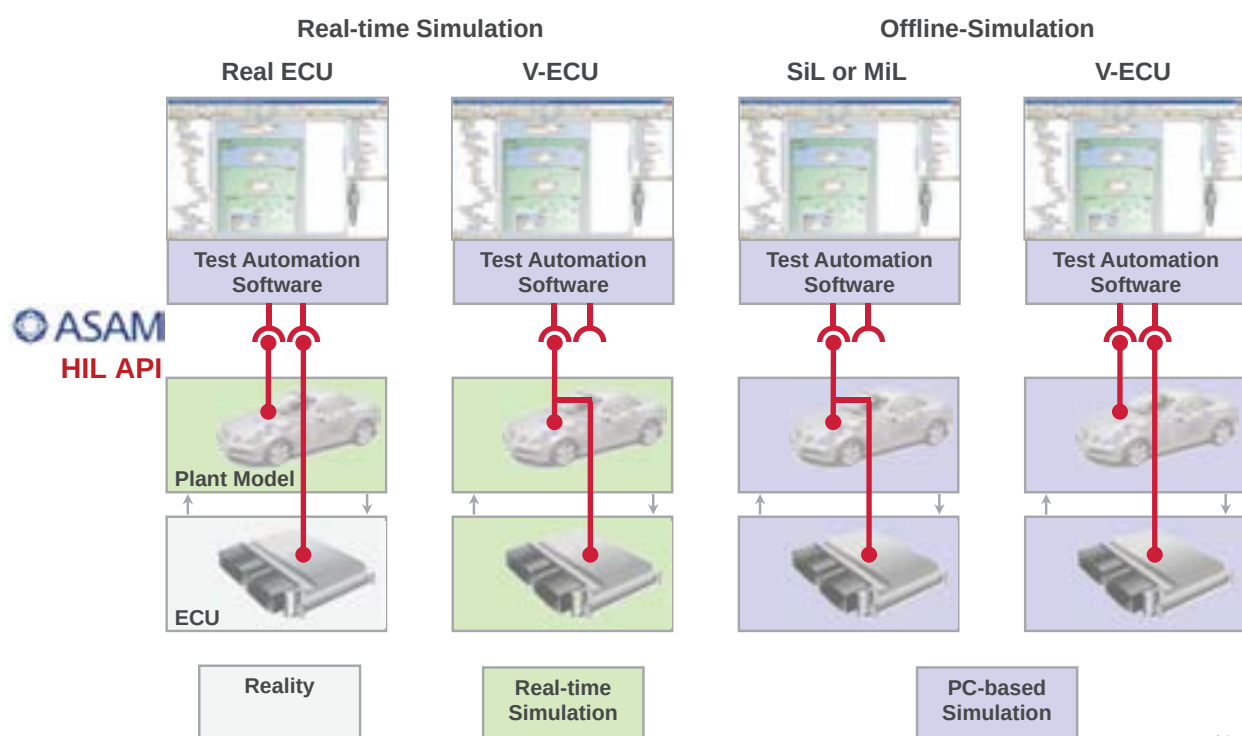
Existing HIL API 1.0 Ports



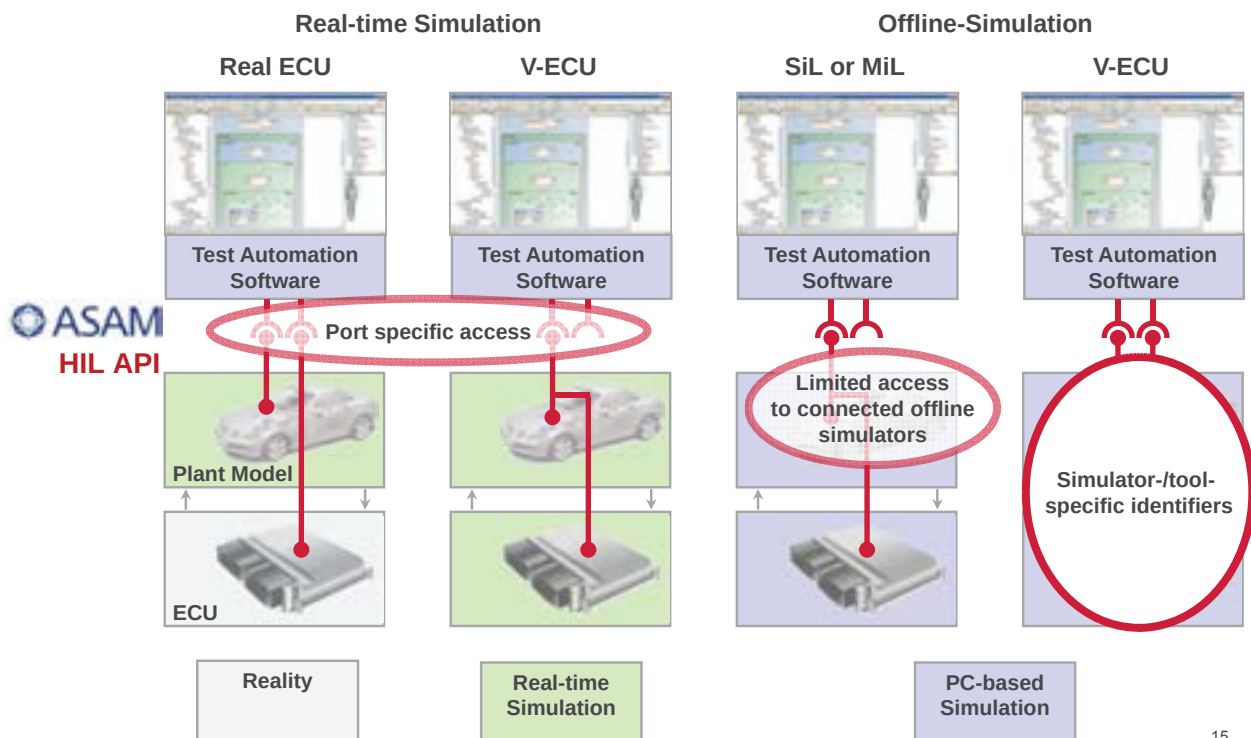
12



13



14



15

Agenda

Introduction: HIL Simulation

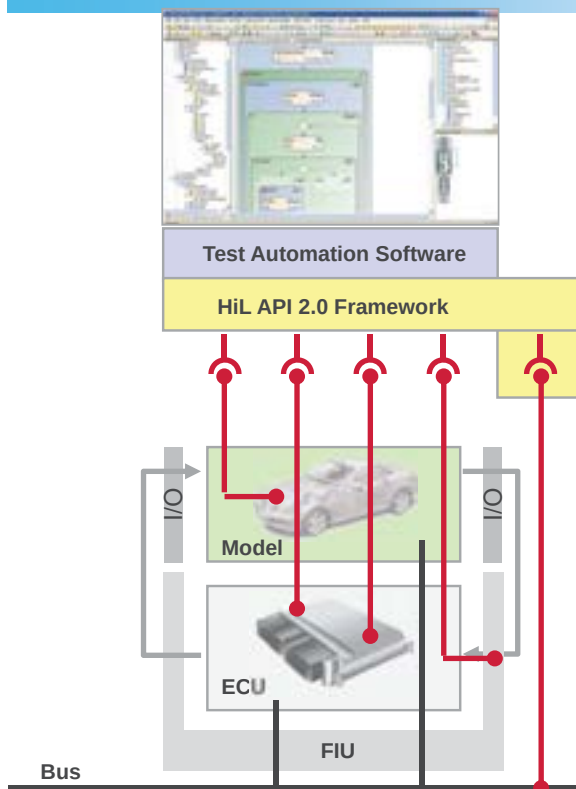
Testautomation ASAM HIL API 1.0

Improvements in ASAM HIL API 2.0

Example

Summary

16



Framework HiL API 2.0

- Mapping decouples test cases and HiL API 1.0 ports
- Simulator control allows standardized initialization of ports, start, stop simulation etc.

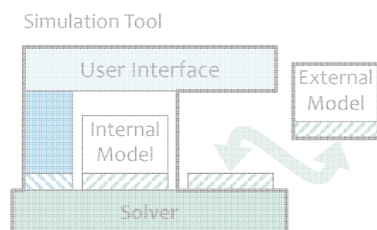
Network Port:

Connects to the bus systems CAN, LIN, and FlexRay

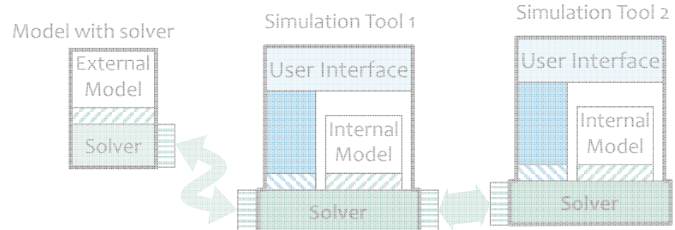
17

Functional Mock-up Interfaces Access to Connected Offline Simulators

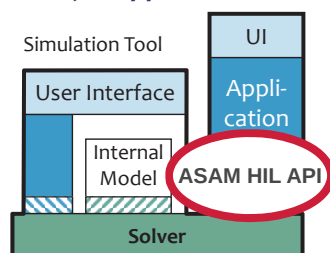
1. FMI for Model Exchange



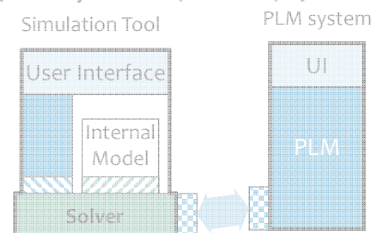
2. FMI for Co-Simulation



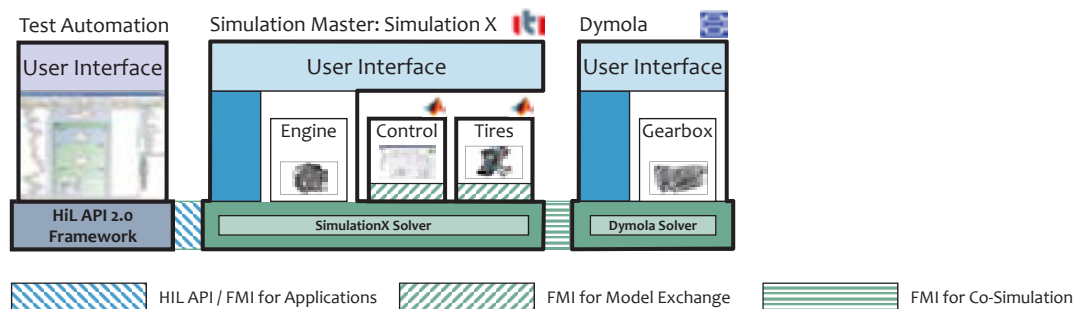
3. FMI for Applications



4. FMI for PLM (Product Lifecycle Management)



- Physical system is simulated by domain-specific tools
- Test automation is connected to master simulator via ASAM HiL API / FMI for Applications
 - Master gives access to all subsystem parameters and signals
 - Test automation starts / stops complete simulation system



19

Agenda

Introduction: HiL Simulation

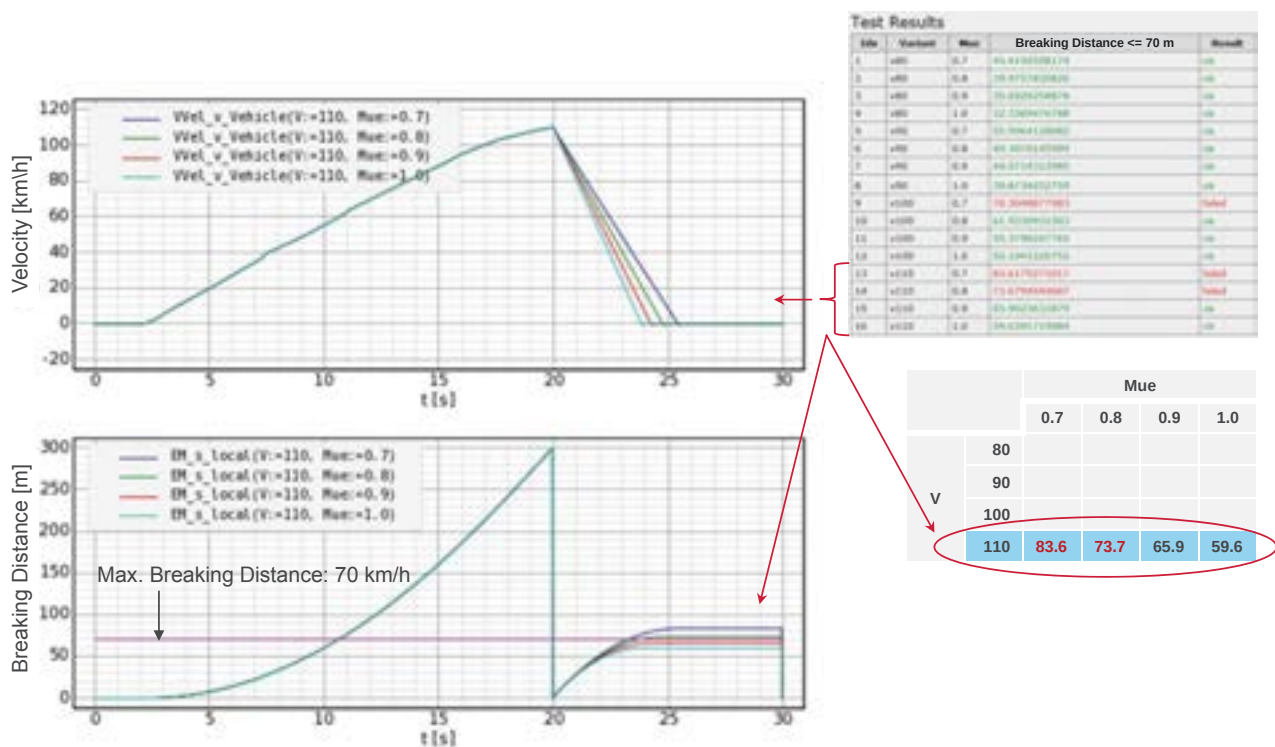
Testautomation ASAM HiL API 1.0

Improvements in ASAM HiL API 2.0

Example

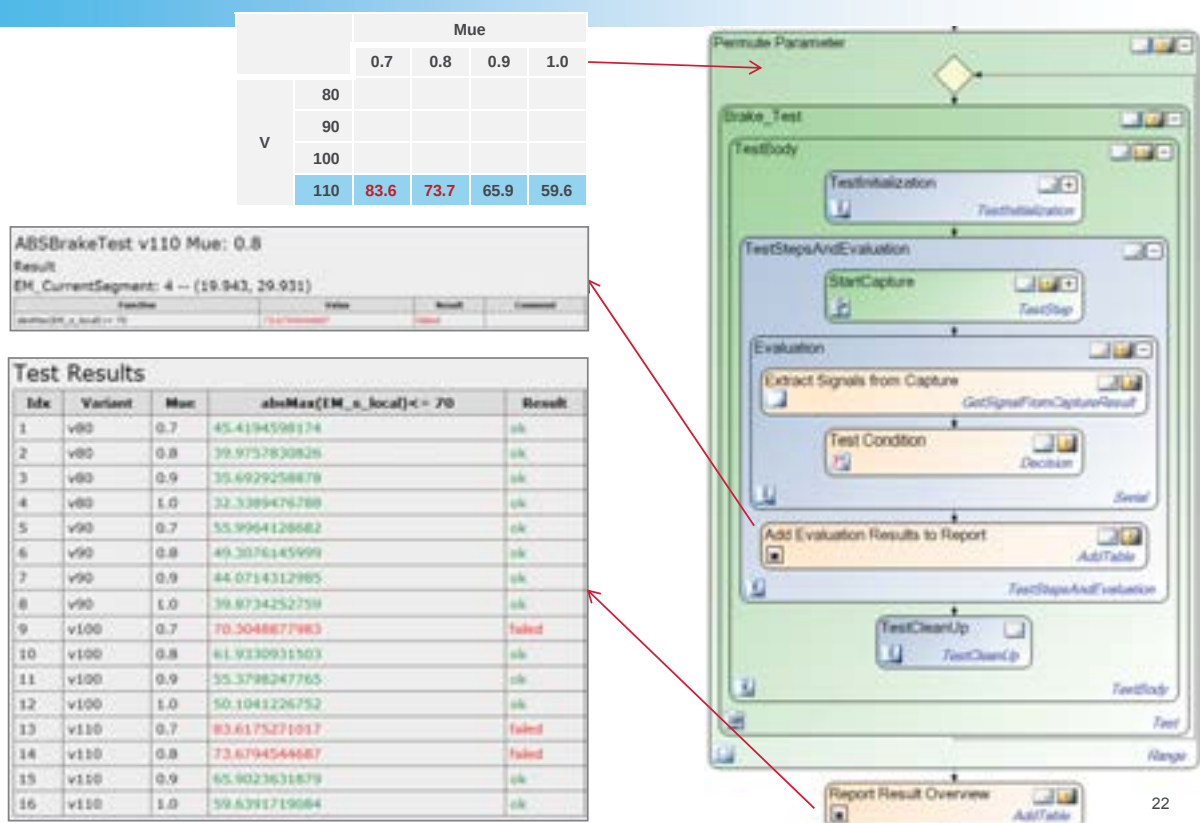
Summary

Breaking Distance at Different Velocities and Mue Values



21

Breaking Distance at Different Velocities and Mue Values



22

Introduction: HIL Simulation

Testautomation ASAM HIL API 1.0

Improvements in ASAM HIL API 2.0

Example

Summary

23

- **Today, test cases often directly depend on the used test hardware**
- **ASAM HIL API 1.0**
Decouples test cases from real and virtual test systems using ports
- **ASAM HIL API 2.0**
Mapping decouples test cases and HIL API 1.0 ports
Standardized simulator control to initialize ports
Network port supports CAN, LIN, FlexRay
- Easy test case exchange between coupled offline simulators in early stages and productive HIL test benches
- Better know-how transfer from one test bench to the other
- Reduced training costs for employees
- **From end users perspective:**
This allows the 'best' test software combined with the 'best' test hardware.

24

Thank you very much for your attention.

dSPACE



© Copyright 2012, dSPACE GmbH. All rights reserved.
Brand names or product names are trademarks or registered trademarks
of their respective companies or organizations.
Contact: rrasche@dspace.de, dietmar.neumerkel@daimler.com
Info: <http://www.asam.net>, <http://functional-mockup-interface.org>

Der Einfluss semi-aktiver Dämpfer auf die Betriebslasten eines PKW

Sebastian Schneider^{1, 2}, Daniel Brechter¹, Andreas Janßen¹,
Heiko Mauch¹, Christian Bohn²

¹ Volkswagen AG

² Institut für Elektrische Informationstechnik, TU Clausthal
sebastian.schneider@tu-clausthal.de

Zusammenfassung

Ansteuerbare mechatronische Komponenten, die mittlerweile in modernen PKW-Fahrwerken vielfach eingesetzt werden, beeinflussen die Kräfte und Momente, welche auf das Fahrzeug und seine Komponenten einwirken. Um die Lebensdauer ermitteln zu können, ist eine genaue Kenntnis der auftretenden Belastungen notwendig. Daher wurde untersucht, inwieweit der Einfluss solcher geregelter mechatronischer Systeme auf die Betriebslasten bei der Ermittlung der Lebensdauer zu berücksichtigen ist. Für diese Untersuchung wurden Fahrbetriebsmessungen an einem Personenkraftwagen mit semi-aktiven Dämpfern durchgeführt. Weiterhin sind die Belastungen, die auf das Fahrzeug einwirken, in Mehrkörpersimulationen analysiert worden. Die Ergebnisse zeigen, dass sich die Lasten am Federbein und im Radmittelpunkt in Abhängigkeit von der Dämpfereinstellung ändern. Die Untersuchungen haben bestätigt, dass der Einfluss semi-aktiver Dämpfer bei der Lebensdauerberechnung berücksichtigt werden muss.

1 Einleitung

Gegenwärtig steigt die Anzahl mechatronischer Komponenten in den Fahrwerken moderner Personenkraftwagen. Viele dieser Systeme stellen Sicherheitsfunktionen bereit und unterstützen den Fahrer in kritischen Fahrsituationen. Des Weiteren werden konventionelle Fahrzeugkomponenten mit Hilfe von elektronischen Bauteilen um neue Funktionen ergänzt. Beispielsweise können konventionelle Fahrzeugdämpfer auf diese Weise zu semi-aktiven Systemen erweitert werden. Die Dämpferkräfte geregelter Systeme sind im Gegensatz zu herkömmlichen Systemen an die jeweilige Fahrsituation anpassbar. Dadurch können variabel einstellbare Dämpfer dazu beitragen, den Zielkonflikt zwischen gutem Fahrverhalten und Fahrkomfort zu lösen. So kann für jede auftretende Fahrsituation eine optimale Fahrwerksabstimmung gefunden werden.

Die Lebensdauer von Bauteilen wird üblicherweise in Prüfstandsversuchen nachgewiesen, bevor Fahrzeugdauerläufe durchgeführt werden. Auf diese Weise können einzelne Bauteile

kostengünstig und schnell optimiert werden, bevor das gesamte System getestet wird. Herkömmliche Versuchsmethoden berücksichtigen den Einfluss mechatronischer Fahrwerkskomponenten auf die Belastungen von Personenkraftwagen nicht. Mechatronische Bauteile können sich jedoch auf das Schwingungsverhalten des Gesamtsystems und die Lasten auswirken [1], [2]. Veränderungen der Fahrzeugbelastungen aufgrund von regelbaren Fahrwerksystemen werfen die folgenden neuen Fragen auf:

- Muss der Einfluss mechatronischer Komponenten bei der Ermittlung der Lebensdauer berücksichtigt werden?
- Wird das Kundenverhalten auf Prüfstrecken in geeigneter Weise repräsentiert?
- Müssen Extrapolationsfaktoren, mit deren Hilfe von der Schädigung in einer repräsentativen Messung auf die Schädigung während der gesamten Nutzungsdauer geschlossen wird, angepasst werden?
- Können Prüfstandsversuche den Einfluss mechatronischer Fahrwerkskomponenten abbilden?

Dieser Artikel behandelt den Einfluss semi-aktiver Dämpfer auf die Belastungen von Personenkraftwagen. Fahrbetriebsmessungen und numerische Simulationen werden durchgeführt, um die Auswirkungen unterschiedlicher Dämpfereinstellungen zu untersuchen. Auf diese Weise kann eine Aussage darüber getroffen werden, ob der Einfluss semi-aktiver Dämpfer auf die Lebensdauer von Fahrzeugen bei der Prüfstandserprobung berücksichtigt werden muss.

2 Regelbare Dämpfer

Die Begriffe, die in den folgenden Ausführungen verwendet werden, um verschiedene Vertikaldynamiksysteme (passive, adaptive, semi-aktive und aktive) zu unterscheiden, entsprechen der Einteilung gemäß [3].

Ein gut abgestimmtes Fahrwerk besitzt sowohl gute Fahreigenschaften als auch solide Komforteigenschaften. Ein gutes Fahrverhalten wird durch eine Minimierung der dynamischen Reifenkräfte erreicht. Um solide Fahrkomforteigenschaften zu erreichen, müssen Schwingungen des Fahrzeugaufbaus reduziert werden. Auch aus medizinischer Sicht ist eine Begrenzung der Aufbauschwingungen erforderlich, um negative Auswirkungen auf die Gesundheit und das Wohlbefinden der Fahrzeuginsassen zu verhindern [4]. Die angesprochenen Anforderungen resultieren in einem Zielkonflikt [5].

In Fahrwerken mit herkömmlichen *passiven* Federn und Dämpfern werden das Kraft-Weg-Diagramm und das Kraft-Geschwindigkeits-Diagramm durch konstante Kennlinien beschrieben [3]. Folglich ist eine Anpassung des Einfederungsverhaltens auf Modifikationen dieser beiden Kennlinien beschränkt. Bei PKW auf befestigten Straßen führen hohe Dämpferraten im Allgemeinen zu guten Fahrdynamikeigenschaften, wohingegen sich

der Fahrkomfort verschlechtert. Aus diesem Grund ist die endgültige Abstimmung des Fahrwerks immer ein Kompromiss.

Mechatronische Komponenten wie regelbare Dämpfer lösen diesen Zielkonflikt auf oder verringern ihn [6]. Daher sind *adaptive* Systeme eingeführt worden, um die Möglichkeiten der Fahrwerksabstimmung zu erweitern. Diese Systeme ermöglichen es, während der Fahrt zwischen verschiedenen Kennlinien zu wechseln. Im Vergleich hierzu werden die Eigenschaften *semi-aktiver* Systeme durch Kennfelder beschrieben. Jeder Arbeitspunkt kann innerhalb von Millisekunden eingestellt werden. Semi-aktive Komponenten führen dem System jedoch keine Energie zu – sie dissipieren nur Energie. *Active* Systeme haben hingegen eine externe Energiequelle. Folglich kann die Wirkrichtung der Dämpferkraft unabhängig von der gegenwärtigen Bewegungsrichtung des Dämpfers geändert werden.

In diesem Artikel wird ein semi-aktiver Dämpfer untersucht, dessen Konstruktion auf einem konventionellen hydraulischen Fahrzeugdämpfer basiert. Die Größe des Durchflussquerschnitts kann während des Betriebs durch ein regelbares Ventil angepasst werden. Dadurch wird die Energiedissipation eingestellt. Das Kraft-Geschwindigkeits-Diagramm semi-aktiver Dämpfer wird durch Kennfelder und nicht durch konstante Kennlinien beschrieben, wie Abbildung 1 zeigt.

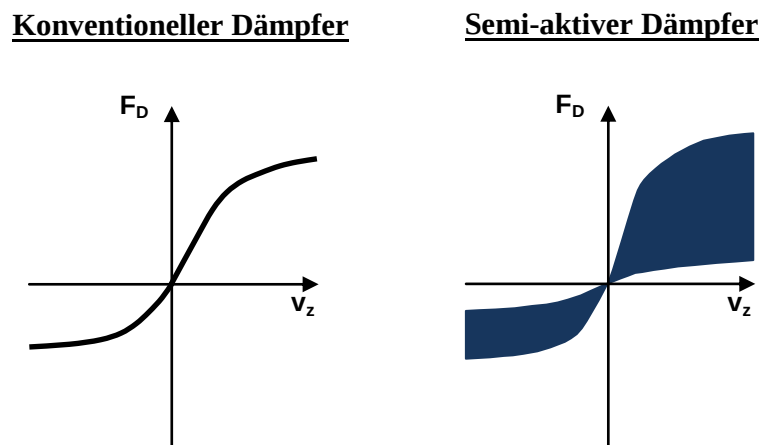


Abbildung 1: Kraft-Geschwindigkeits-Diagramm eines konventionellen (links) und eines semi-aktiven Dämpfers (rechts)

Ein Steuergerät bestimmt die Einstellungen des Dämpfers unter Berücksichtigung der Programmauswahl des Fahrers, der Fahrbahnoberfläche sowie der Fahrsituation (siehe Abbildung 2). Der Fahrer legt die Basiskennlinie des Kraft-Geschwindigkeit-Kennfelds fest, indem er aus drei verfügbaren Programmen (komfortabel, normal, sportlich) eines auswählt. Wird die sportliche Abstimmung gewählt, ist das Kennfeld stärker begrenzt als im komfortablen Modus. Dementsprechend ist der Minimalwert der absoluten Dämpferkraft größer als in der komfortablen Einstellung.

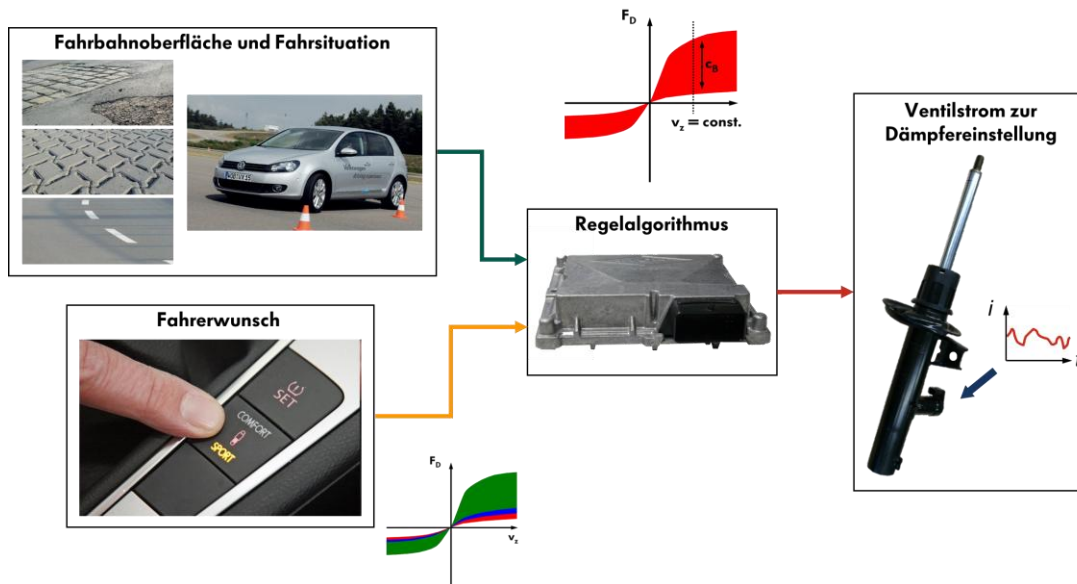


Abbildung 2: Ansteuerungsstrategie des semi-aktiven Dämpfers

Zudem analysiert das Steuergerät die Fahrsituation und den Straßenzustand, um Wank- und Nickbewegungen zu reduzieren. Die Dämpferrate wird innerhalb von Millisekunden angepasst, indem die Bestromung des regelbaren Ventils verändert wird. Wenn beispielsweise kritische Situationen auftreten, in denen ein gutes Fahrverhalten besonders wichtig ist, wird eine hohe Dämpferrate eingestellt. Führt das Fahrzeug hingegen auf einer geraden und ebenen Straße, ist die Dämpferrate geringer, um einen guten Fahrkomfort zu erreichen.

3 Numerische Simulation in der Betriebsfestigkeit

Mehrkörpersimulationen werden eingesetzt, um die Belastungen zu bestimmen, die auf ein Fahrzeug und seine Komponenten einwirken [7]. Simulationen von Fahrbetriebsmessungen auf Prüfstrecken sind Stand der Technik. Durch numerische Berechnungen können die Lasten in einem frühen Stadium des Entwicklungsprozesses ermittelt werden, bevor reale Hardware-Prototypen aufgebaut werden. Weiterhin ist es möglich, Kräfte und Momente, die auf Gelenke und Gummilager einwirken, zu berechnen. Diese können während Fahrbetriebsmessungen nicht mit wirtschaftlich vertretbarem Aufwand ermittelt werden. Darüber hinaus werden mit Hilfe von Mehrkörpersimulationen Prüfstandsversuche entwickelt und optimiert [8], [9]. Auf diese Weise kann die technische Umsetzbarkeit von Prüfstandtests im Voraus überprüft werden.

Numerische Berechnungen sind bereits erfolgreich eingesetzt worden, um die Belastungen zu bestimmen, die während einer Bremsung mit einem Antiblockiersystem auftreten [10]. Zur Kopplung oder Integration von Mehrkörpersystemen und Reglersystemen in Simulationen existieren verschiedene Methoden [11]. Bei dem sogenannten Tight Coupling werden die Differentialgleichungen, die das Mehrkörpersystem beschreiben, sowie die Gleichungen des Regelalgorithmus von dem gleichen Programm gelöst. Folglich muss der Regelalgorithmus in das Mehrkörpersystem integriert werden oder umgekehrt. Im Gegensatz

dazu werden beim Weak Coupling verschiedene Solver genutzt, um die Gleichungen des Mehrkörpersystems und des Reglers zu lösen. Dementsprechend müssen Prozesse zum Datenaustausch und zur Steuerung der Simulation definiert und bereitgestellt werden.

4 Methode

Der Einfluss verschiedener Dämpfereinstellungen auf die Belastungen wird an einem Serienfahrzeug untersucht, das mit semi-aktiven Dämpfern ausgerüstet ist. In Fahrbetriebsmessungen sowie numerischen Berechnungen werden Veränderungen der Vertikalkräfte bestimmt. Weiterhin wird das gleiche Fahrzeug mit herkömmlichen Dämpfern ausgerüstet. Folglich können die Belastungen eines konventionellen Fahrwerks mit denen eines Fahrwerks mit semi-aktiven Dämpfern verglichen werden. Auf diese Weise kann überprüft werden, ob bestehende Verfahren zur Lebensdauervorhersage angepasst werden müssen, um den Einfluss mechatronischer Fahrwerkskomponenten zu berücksichtigen.

Zur Abschätzung der Lasten, die während Fahrbetriebsmessungen auftreten, werden Simulationen mit einem Viertelfahrzeugmodell durchgeführt. Dann wird ein Personenkraftwagen mit Sensoren und Messgeräten ausgestattet, um Messungen durchzuführen. Danach wird ein Modell für Mehrkörpersimulationen aufgebaut und mit Hilfe der Messergebnisse validiert (siehe Abbildung 3).

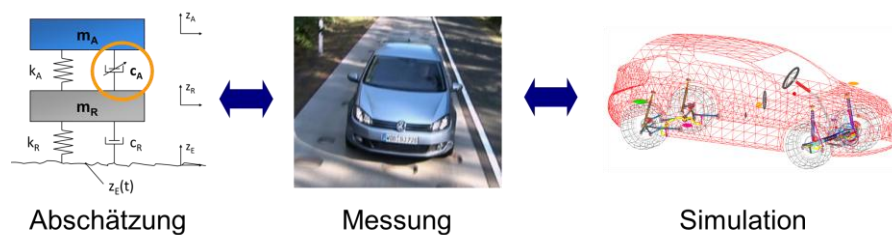


Abbildung 3: Vorgehen zur Bestimmung des Einflusses semi-aktiver Dämpfer auf die Vertikalbelastungen

4.1 Abschätzung

Um den möglichen Einfluss des semi-aktiven Fahrwerkssystems auf die Belastungen des Fahrzeugs und seiner Komponenten abzuschätzen, wird ein Viertelfahrzeugmodell untersucht (siehe Abbildung 4). Zur Modellierung des Schwingungsverhaltens in Vertikalrichtung werden die Radmasse m_R und die Aufbaumasse m_A verwendet. Erstere bildet die ungefederten Massen ab, wohingegen letztere die gefederten Massen des Fahrzeugs repräsentiert. Die Reifensteifigkeit k_R sowie die Reifendämpfung c_R werden mittels Konstanten beschrieben. Weiterhin werden die Federung k_A und Dämpfung c_A des Aufbaus abgebildet. Die Aufbausteifigkeit wird als konstant angenommen. Die Dämpfung des Aufbaus ist bei einem Fahrzeug mit semi-aktiven Dämpfern veränderlich. In ersten Berechnungen werden konstante Werte für die minimale und maximale Dämpfungsrate eingesetzt.

Das Viertelfahrzeugmodell wird durch die Bewegungsgleichungen

$$m_A \cdot \ddot{z}_A = -c_A \cdot (\dot{z}_A - \dot{z}_R) - k_A \cdot (z_A - z_R) \quad (1)$$

und

$$m_R \cdot \ddot{z}_R = -c_A \cdot (\dot{z}_R - \dot{z}_A) - k_A \cdot (z_R - z_A) - c_R \cdot (\dot{z}_R - \dot{z}_E) - k_R \cdot (z_R - z_E) \quad (2)$$

beschrieben [12].

Für die kinematische Straßenanregung z_E werden Messsignale eines vergleichbaren Fahrzeugs verwendet. Weiterhin werden sinusförmige Anregungen unterschiedlicher Frequenz und Amplitude verwendet. Das vereinfachte Modell ist ausreichend genau, um erste Abschätzungen von Beschleunigungs- und Kraftamplituden, die während einer Fahrbetriebsmessung auftreten, durchzuführen.

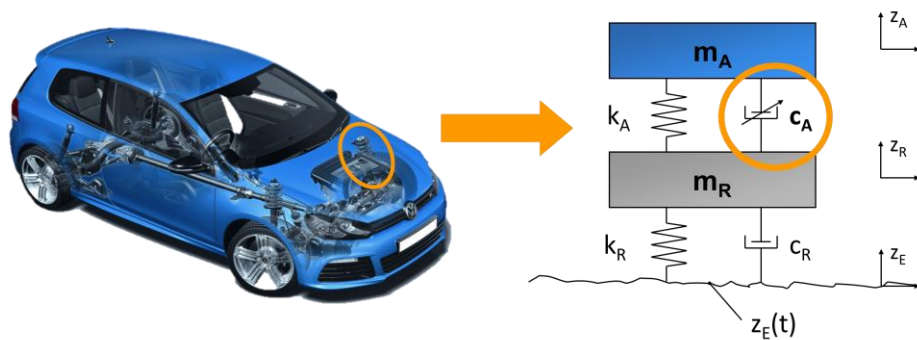


Abbildung 4: Viertelfahrzeugmodell

4.2 Messung

Um den Einfluss semi-aktiver Dämpfer in Fahrbetriebsmessungen zu untersuchen, wird das Fahrzeug mit Sensoren ausgerüstet, welche Beschleunigungen, Kräfte, Dehnungen und die Dämpferwege erfassen. Die Messdaten enthalten alle Informationen, welche auch für Untersuchungen an Gesamtfahrzeugprüfständen benötigt werden. Die Messaufnehmer befinden sich auf der Vorder- und Hinterachse sowie dem Aufbau des Fahrzeugs, wie in Abbildung 5 dargestellt. Des Weiteren werden die Kräfte und Momente, welche im Radmittelpunkt wirken, mit Hilfe von Messrädern erfasst. Die Anordnung der Messsensoren ermöglicht eine Analyse des Kraftflusses im Fahrzeug.

Zusätzlich zu den Messdaten, die das mechanische System beschreiben, werden Informationen des Kommunikationssystems des Fahrzeugs aufgezeichnet. CAN-Bus-Nachrichten wie beispielsweise der Lenkradwinkel, die Bremsanforderung und das Fahrerwunschmoment werden zur Plausibilisierung der Messsignale herangezogen. Mit Hilfe eines Entwicklungssteuergeräts werden zudem Eingangs- und Ausgangsvariablen sowie interne Variablen des Dämpfersteuergeräts erfasst.

Die Messungen werden auf Prüfstrecken und öffentlichen Straßen durchgeführt. Auf diese Weise können der Einfluss verschiedener Fahrbahnoberflächen sowie die Auswirkungen unterschiedlicher Fahrzustände untersucht werden.



Abbildung 5: Sensoranordnung für Fahrbetriebsmessungen

4.3 Simulation

Der Einfluss semi-aktiver Dämpfer auf die Belastungen eines Fahrzeugs und seiner Komponenten wird zudem in Mehrkörpersimulationen untersucht. Die Methode des Tight Coupling wird verwendet, um das Dämpfersteuergerät abzubilden. Dementsprechend werden die Gleichungen, die den Regelalgorithmus beschreiben, in das Mehrkörpermodell integriert. Der Softwarecode des Modells entspricht dem des Steuergeräts im Messfahrzeug. Zudem werden die Steuergerätesignale mit Messsignalen verglichen. Auch das Mehrkörpermodell wird auf diese Weise mit dem Messfahrzeug abgestimmt. Die Dämpferkräfte werden durch einen dreidimensionalen Spline abgebildet, der aus der Vermessung von Dämpferkennlinien stammt. Folglich können die Dämpferkräfte in Abhängigkeit vom Ventilstrom und der Relativbewegung zwischen Dämpferrohr und Dämpferstange berechnet werden. Weiterhin liegt ein digitales Bodenmodell der Prüfstrecke vor. In numerischen Simulationen, können die Auswirkungen verschiedener Dämpfereinstellungen auf die Betriebslasten unabhängig vom Einfluss des Fahrers untersucht werden.

4.4 Auswertung

Da die Dämpfereigenschaften hauptsächlich die Vertikalbewegungen des Fahrzeugs beeinflussen, wird diese Richtung in den Auswertungen betrachtet. Ein weiterer Vorteil dabei ist, dass diese Richtung nahezu unabhängig vom Fahrerverhalten ist, welches insbesondere die Längs- und Querkraften in starkem Maße beeinflusst. Um Veränderungen der Fahrbahnanregung und der Fahrzeuggeschwindigkeit zu vermeiden, werden die Messungen auf Prüfstrecken mit festgelegten Geschwindigkeitsprofilen durchgeführt. Sowohl die Messsignale als auch die Simulationsergebnisse werden gemäß dem Verfahren, das in Abbildung 6 beschrieben ist, ausgewertet.

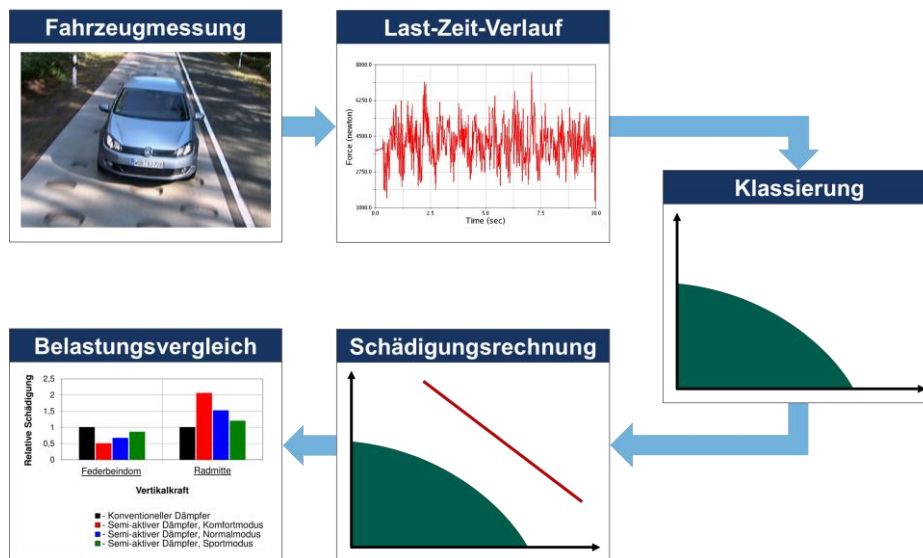


Abbildung 6: Auswertungsverfahren

Die Last-Zeit-Verläufe aus Fahrbetriebsmessungen und Simulationen werden mittels Bereichspaarzählung klassiert. Danach wird eine virtuelle Wöhlerkurve gemäß einem standardisierten Verfahren definiert. Diese bildet nicht die reale Beanspruchbarkeit des betrachteten Bauteils ab, sondern sie wird nur für einen Vergleich der Schädigungen unterschiedlicher Last-Zeit-Verläufe verwendet. Daher ist die Neigung der Wöhlerlinie ein Näherungswert, der für verschiedene Fahrzeugkomponenten angewendet werden kann.

Danach wird eine lineare Schadensakkumulation entsprechend der elementaren Miner-Regel [13] durchgeführt. Die resultierenden Schädigungen werden mit den Schädigungswerten verglichen, welche aus Fahrbetriebsmessungen mit konventionellen Dämpfern stammen.

5 Ergebnisse

Die Vertikalkräfte des Fahrzeugs, welche in Fahrbetriebsmessungen sowie in Simulationen ermittelt werden, sind von der Einstellung des semi-aktiven Dämpfers abhängig. Die Ergebnisse sind in Abbildung 7 dargestellt. Im Folgenden werden die Resultate der Belastungen in den Radmittelpunkten sowie an den Federbeindomen diskutiert. Erstere beschreiben die Kräfte auf die ungefederten Massen, wohingegen letztere die Kräfte auf die gefederten Massen des Fahrzeugs darstellen.

Der Vergleich verschiedener Einstellungen des semi-aktiven Dämpfers zeigt, dass die größten Kräfte an den Federbeindomen im Sportmodus beobachtet werden. Geringere Dämpferraten führen zu niedrigeren Vertikalbelastungen des Fahrzeugaufbaus. Jedoch resultiert eine verringerte Aufbaudämpfung in höheren vertikalen Radlasten. Folglich treten die größten Vertikallasten in den Radmittelpunkten im Komfortmodus auf.

Diese Ergebnisse stimmen mit den Untersuchungen am Viertelfahrzeugmodell (Abschnitt 4.1) und den Ergebnissen der Mehrkörpersimulationen (Abschnitt 4.3) überein.

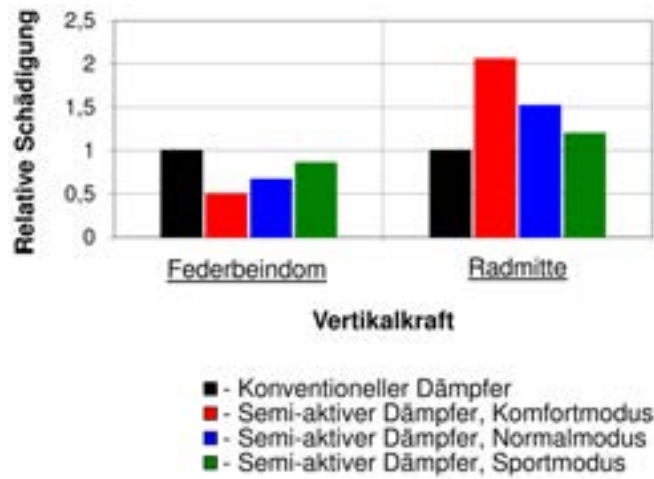


Abbildung 7: Relative Schädigung, welche auf der Prüfstrecke durch die Vertikalkräfte am Federbeindom und im Radmittelpunkt hervorgerufen wird

6 Schlussfolgerungen

Die Analyse der Messergebnisse zeigt, dass der Einfluss semi-aktiver Dämpfer bei der Ermittlung der Lebensdauer eines Fahrzeugs berücksichtigt werden muss. Mehrkörpersimulationen eines Fahrzeugs mit integriertem Steuergerät (Tight Coupling) sind erfolgreich eingesetzt worden, um den Einfluss unterschiedlicher Dämpfereinstellungen auf die Lebensdauer zu untersuchen. Folglich ist es möglich, die Belastungen eines Personenkraftwagens mit Hilfe numerischer Berechnungen in einem frühen Stadium des Entwicklungsprozesses vorherzusagen. Zudem kann die Konstruktion von Bauteilen sowie die Genauigkeit von Lebensdauervorhersagen verbessert werden.

Das Hauptaugenmerk des in diesem Artikel vorgestellten Projektes liegt auf Fahrbetriebsmessungen sowie Simulationen. Jedoch ist es erforderlich, auch Prüfstandsversuche in die Untersuchungen einzubeziehen (siehe Abbildung 8). Daher werden zukünftig mittels Mehrkörpersimulationen Testverfahren analysiert, die den Einfluss semi-aktiver Dämpfer in Laborversuchen berücksichtigen.

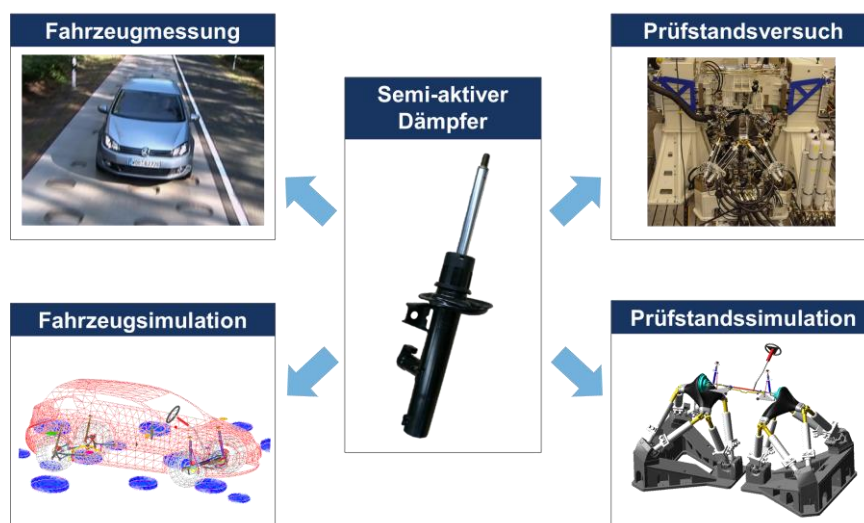


Abbildung 8: Semi-aktive Dämpfer in der Lebensdauererprobung

Literatur

- [1] Schneider, S., C. Bohn, D. Brechter, A. Janßen und H. Mauch. 2011. The influence of semi-active dampers on the loads of passenger cars. *Proceedings of the International Conference on Fatigue Design 2011*. Senlis, Frankreich. CD-ROM(S1.2).
- [2] Schneider, S., D. Brechter, A. Janßen und H. Mauch. 2011. The influence of semi-active dampers on the vibration behaviour of passenger cars. *Proceedings of the 10th International Conference on Vibration Problems*. Prag. 189-94.
- [3] Heißing, B. und M. Ersoy (Hgg.). 2011. *Chassis Handbook*. Wiesbaden: Vieweg Teubner.
- [4] Guglielmino, E., T. Sireteanu, C. W. Stammers, G. Ghita und M. Giuclea. 2008. *Semi-active Suspension Control*. Berlin: Springer.
- [5] Karnopp, D. und D. Margolis. 1984. Adaptive suspension concepts for road vehicles. *Vehicle System Dynamics* 13:145-60.
- [6] Elbeheiry, E. M., D. C. Karnopp, M. E. Elaraby und A. M. Abdelraaouf. 1995. Advanced ground vehicle suspension systems – a classified bibliography. *Vehicle System Dynamics* 24:231-58.
- [7] Eichler, M. und A. Lion. 2000. Gesamtfahrzeugsimulationen auf Prüfstrecken zur Bestimmung von Lastkollektiven. *Tagungsband der VDI-Tagung Berechnung und Simulation im Fahrzeugbau*. Würzburg. 369-98.
- [8] Mauch, H., A. Ahmadi, G. Zhang und T. Kersten. 2007. Numerische Simulation von Prüfsystemen. *Tagungsband der 34. DVM-Tagung Lastannahmen und Betriebsfestigkeit*. Wolfsburg. 291-302.
- [9] Ahmadi, A., H. Mauch und G. Bremer. 2007. Entwicklung von Prüfzenarien mittels der numerischen Betriebsfestigkeitssimulation. *Tagungsband der 33. VDI-Jahrestagung Schadensanalyse*. Würzburg. 67-81.
- [10] Mack, G. 2009. *Eine neue Methodik zur modellbasierten Bestimmung dynamischer Betriebslasten im mechatronischen Fahrwerkentwicklungsprozess*. Schriftenreihe des

- [11] Vaculin, O., W. R. Krüger und M. Valasek. 2004. Overview of coupling of multibody and control engineering tools. *Vehicle System Dynamics* 41:415-29.
- [12] Wallentowitz, H. 2006. *Vertikal- und Querdynamik von Kraftfahrzeugen*. Schriftenreihe Automobiltechnik. Aachen: Forschungsgesellschaft Kraftfahrwesen Aachen.
- [13] Miner, M. A. 1945. Cumulative damage in fatigue. *Journal of Applied Mechanics* 12:A159-64.

Effizientes Thermomanagement – Einsatz der Gesamtfahrzeugsimulation zur Quantifizierung von Kraftstoffverbrauchseinsparungen im PKW



Rashad Mustafa, Ferit Küçükay
Institut für Fahrzeugtechnik

Gliederung

Einleitung

Systemmodellierung

Validierung

Kühlerjalousie – Regelungsstrategie

Ergebnisse

Zusammenfassung

Einleitung

Systemmodellierung

Validierung

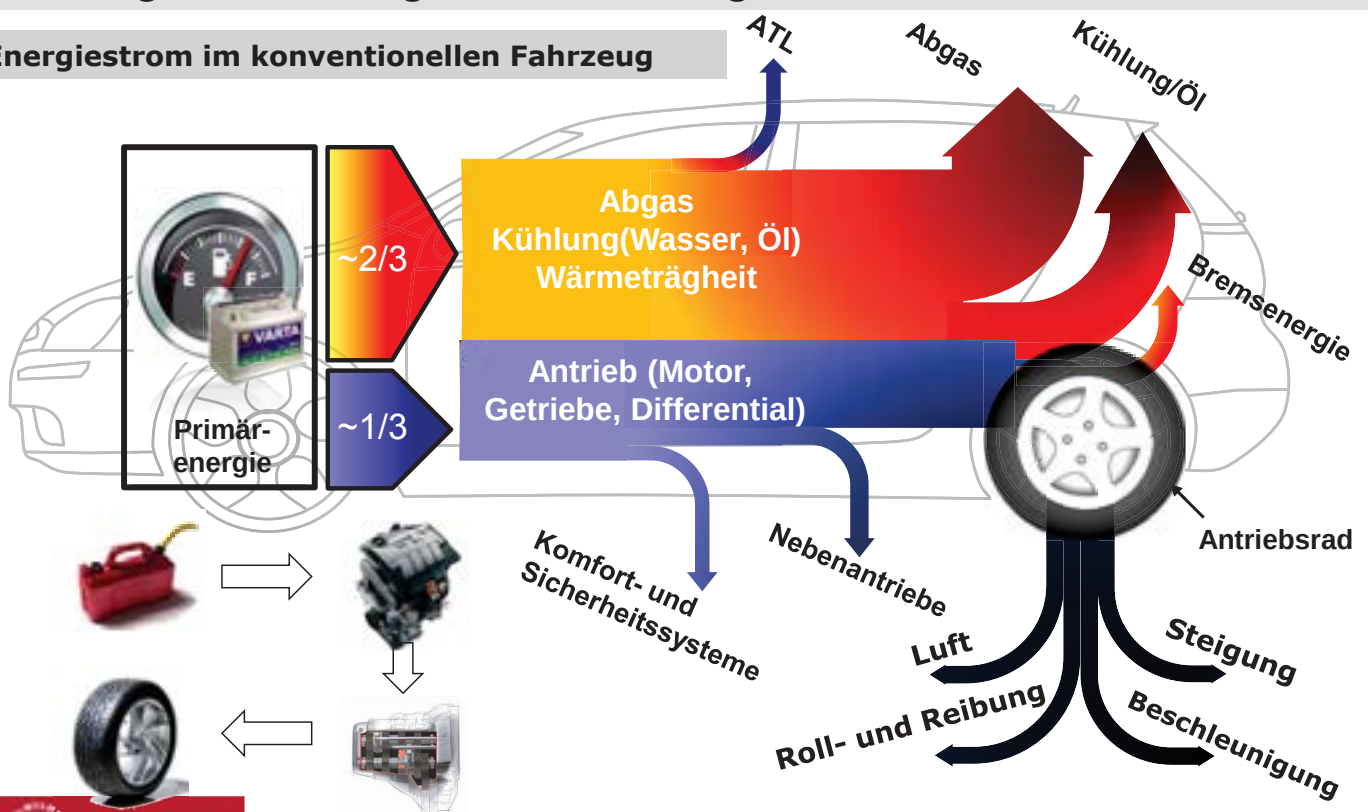
Kühlerjalousie – Regelungsstrategie

Ergebnisse

Zusammenfassung

Energieumsetzung - Wärmemanagement

Energiestrom im konventionellen Fahrzeug



Einleitung

Systemmodellierung

Validierung

Kühlerjalousie – Regelungsstrategie

Ergebnisse

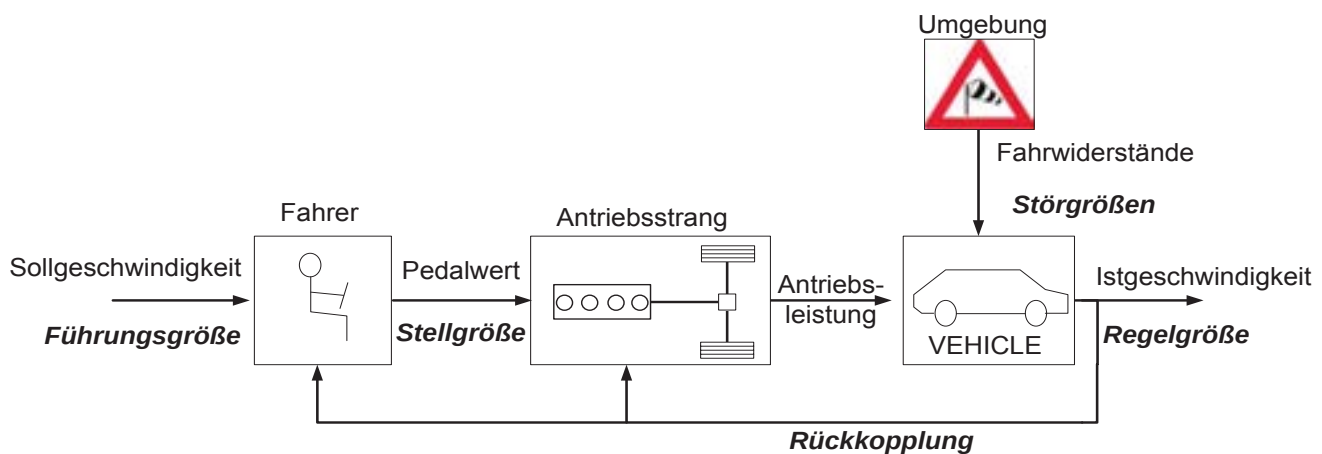
Zusammenfassung



Technische
Universität
Braunschweig



Fahrzeugmodell



Eingangsgrößen:

- Führungsgröße
- Störgrößen

Ausgangsgrößen

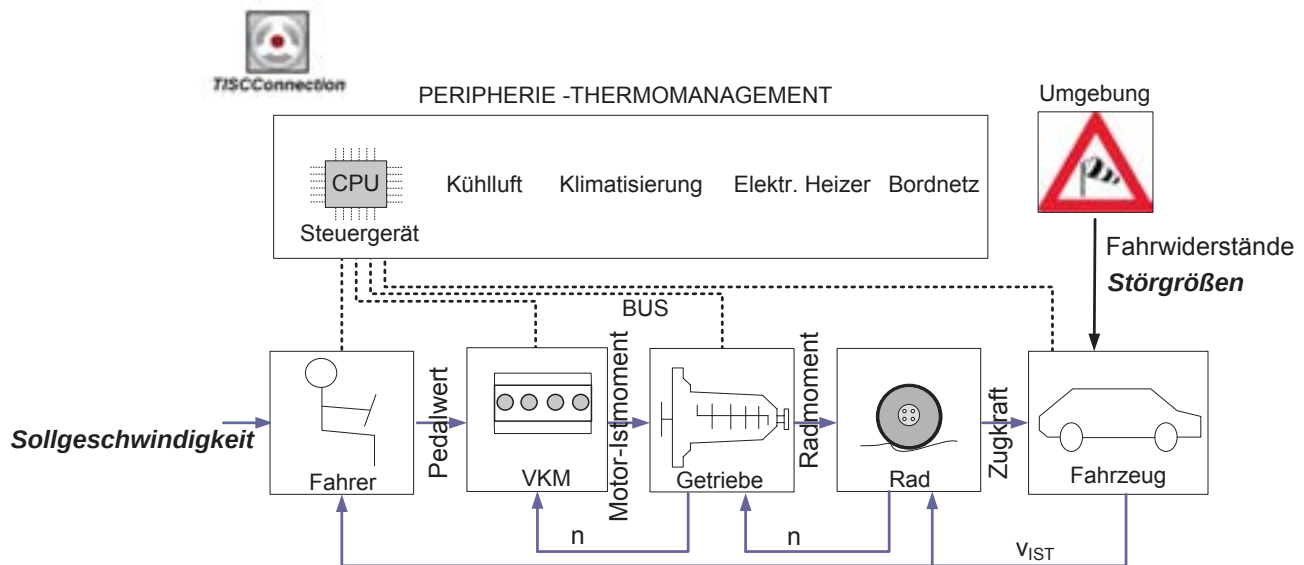
- Istgeschwindigkeit
- ...



Technische
Universität
Braunschweig



Fahrzeugsimulationsumgebung



Zur Verbindung der einzelnen Teilmodelle wurde eine **Co-Simulations-Software** verwendet.

Motorraumkapselung & Kühlerjalousie

Ziel:

- Verkürzung der Warmlaufphase
- Verlängerung der Abkühlzeit
- Vorteile im Kraftstoffverbrauch durch verbesserten c_w - Wert

Die Oberflächen der gekapselten Bauteile:

- Zylinderkopf
- Zylinderkurbelgehäuse
- Motorölwanne
- Getriebegehäuse

Materialien:

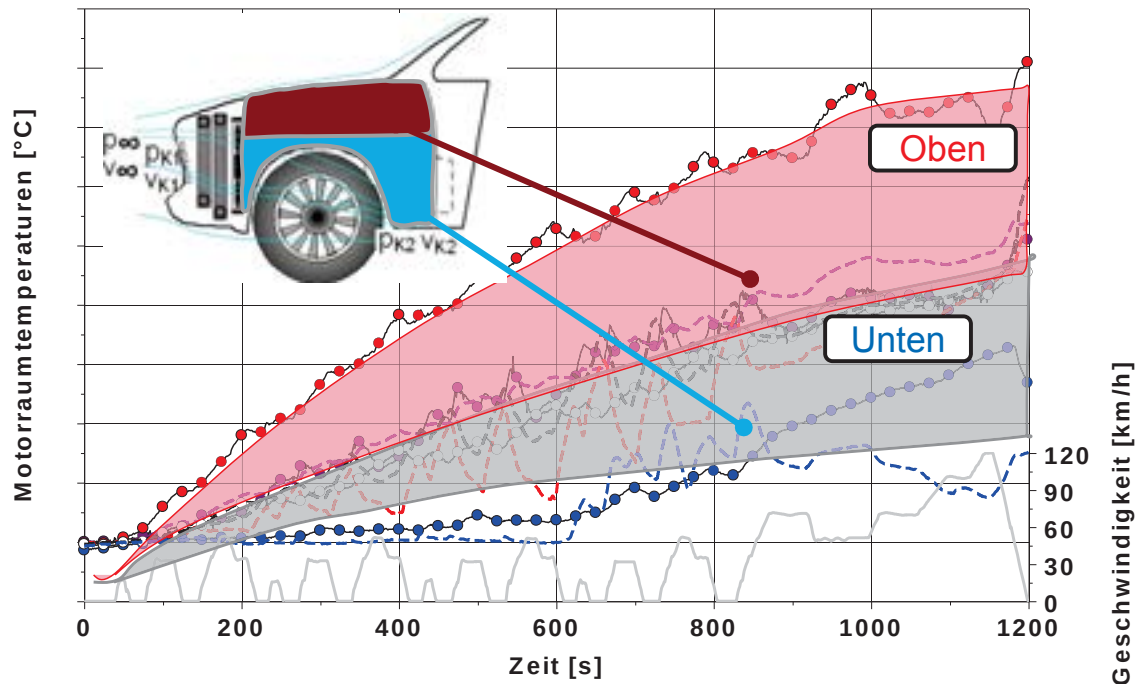
- Dämmmatte mit Alu-Beschichtung
 - 7 mm Matte,
 - 0.1 mm Alu-Schicht;
 - Wärmeleitfähigkeit 0.04 W/mK (bei $T_{Umg} = 50\text{ }^{\circ}\text{C}$)
- Zusatzgewicht (5.5 kg)



Motorraumkapselung

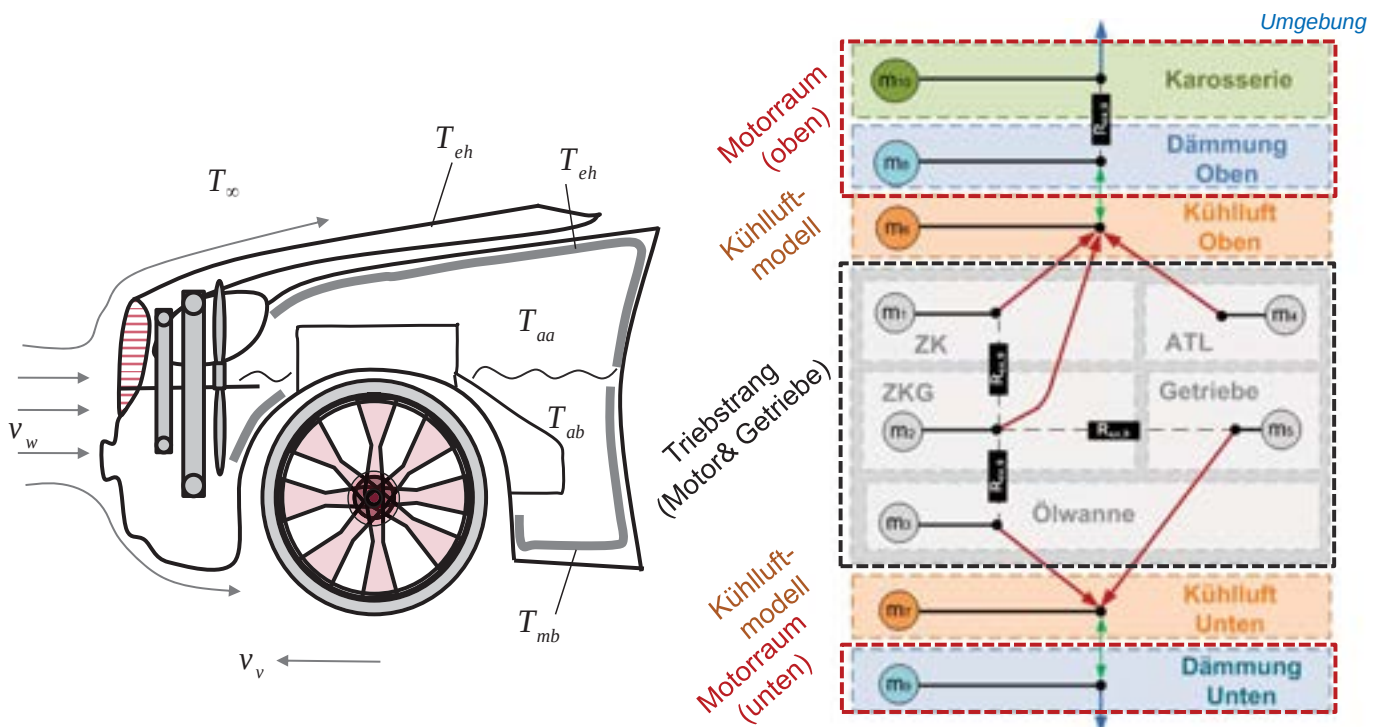
Messdaten: Klimawindkanal

Gemessene Temperaturverläufe von einem gekapselten Fahrzeug im Klimawindkanal



Motorraummodell

Thermisches Netzwerk



Motorraumkapselung Modellierung

1. Karosserie

$$C_{eh} \dot{T}_{eh} = \frac{T_{ma} - T_{eh}}{R_{Cond}} - \alpha_{\infty a} A_A (T_{eh} - T_{\infty}) - \epsilon_{eh} A_A \sigma (T_{eh}^4 - T_{\infty}^4)$$

2. Dämmung Oben

$$C_{ma} \dot{T}_{ma} = \alpha_{aa} A_A (T_{aa} - T_{ma}) - \frac{T_{ma} - T_{eh}}{R_{Cond}}$$

3. Kühlluft Oben

$$C_{aa} \dot{T}_{aa} = \dot{m}_{caa} c_{pc} (T_{RM} - T_{aa}) - Q_{PTA} - \alpha_{aa} A_A (T_{aa} - T_{ma})$$

4. Kühlluft Unten

$$C_{ab} \dot{T}_{ab} = \dot{m}_{cab} c_{pc} (T_{RM} - T_{ab}) - Q_{PTB} - \alpha_{ab} A_B (T_{mb} - T_{ab})$$

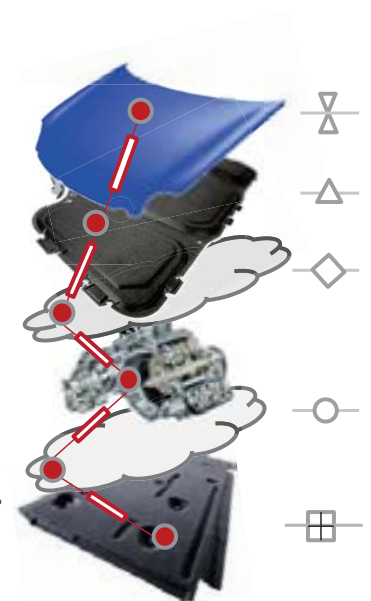
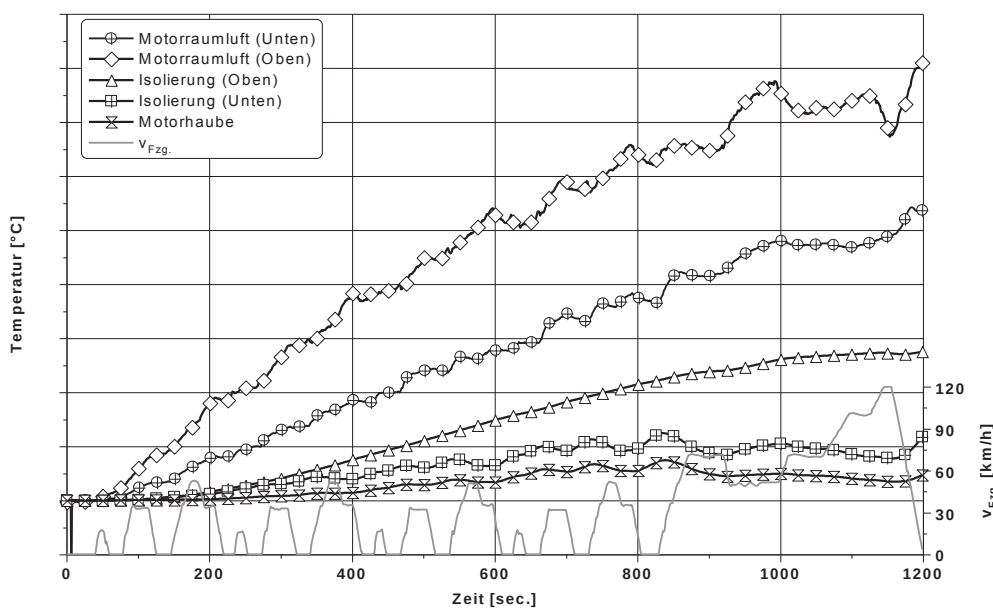
5. Dämmung Unten

$$C_{mb} \dot{T}_{mb} = \alpha_{ab} A_B (T_{mb} - T_{ab}) - \alpha_{\infty b} A_B (T_{mb} - T_{\infty}) - \epsilon_{mb} A_B \sigma (T_{mb}^4 - T_{\infty}^4)$$



Motorluftmodell Systemverhalten

Simulierte NEFZ Temperaturprofile bei einer Umgebungstemperatur von 20 °C.



Einleitung

Systemmodellierung

Validierung

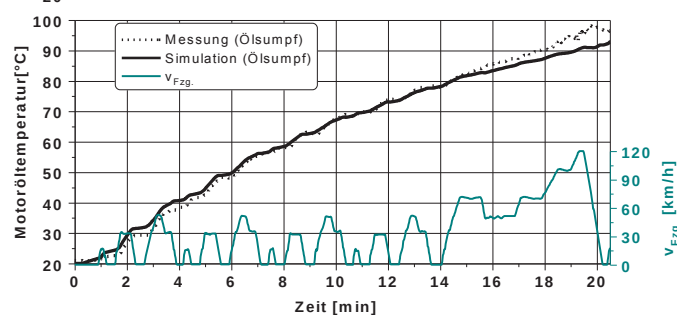
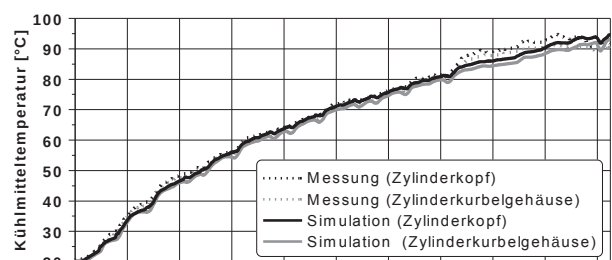
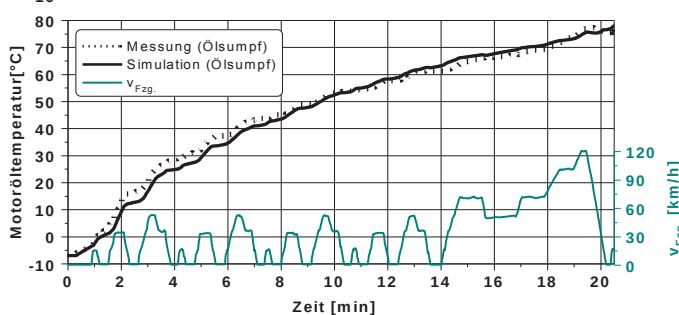
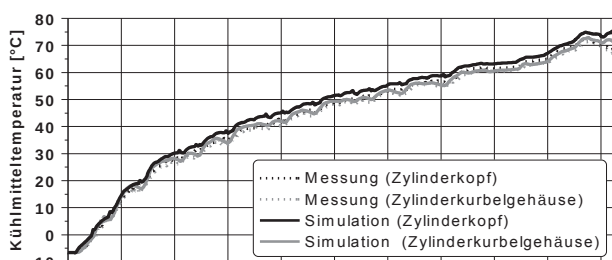
Kühlerjalousie – Regelungsstrategie

Ergebnisse

Zusammenfassung

Modellvalidierung: NEFZ, 20°C, -7°C

- Die Kühlmittel- und Öltemperaturen eignen sich sehr gut für die Validierung, weil sich diese Temperaturen im Motor nur geringfügig ändern.



- Die Temperaturen von Kühlmittel und Öl bei einer Umgebungstemperatur von 20 C sowie -7 C stimmen sehr gut überein.

Einleitung

Systemmodellierung

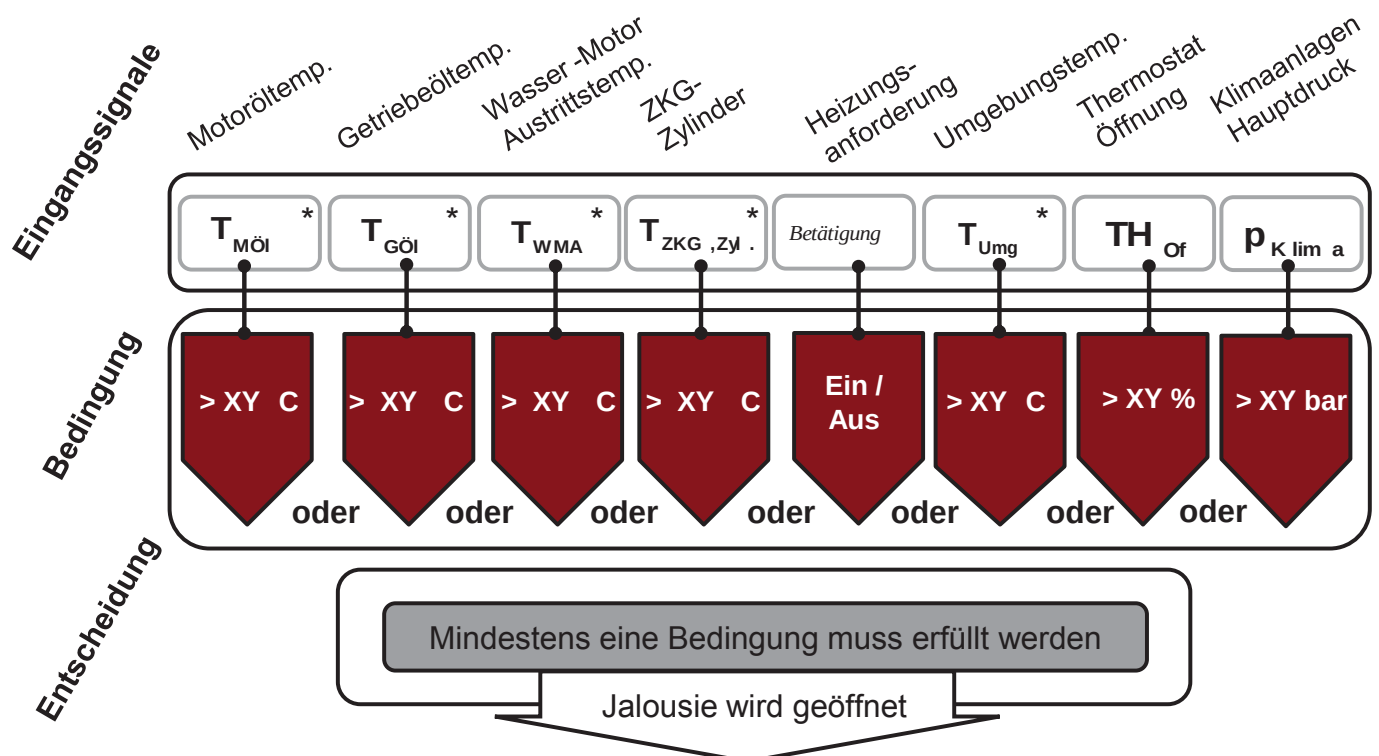
Validierung

Kühlerjalousie – Regelungsstrategie

Ergebnisse

Zusammenfassung

Jalousiesteuerung Betriebsstrategie



Einleitung

Systemmodellierung

Validierung

Kühlerjalousie – Regelungsstrategie

Ergebnisse

Zusammenfassung

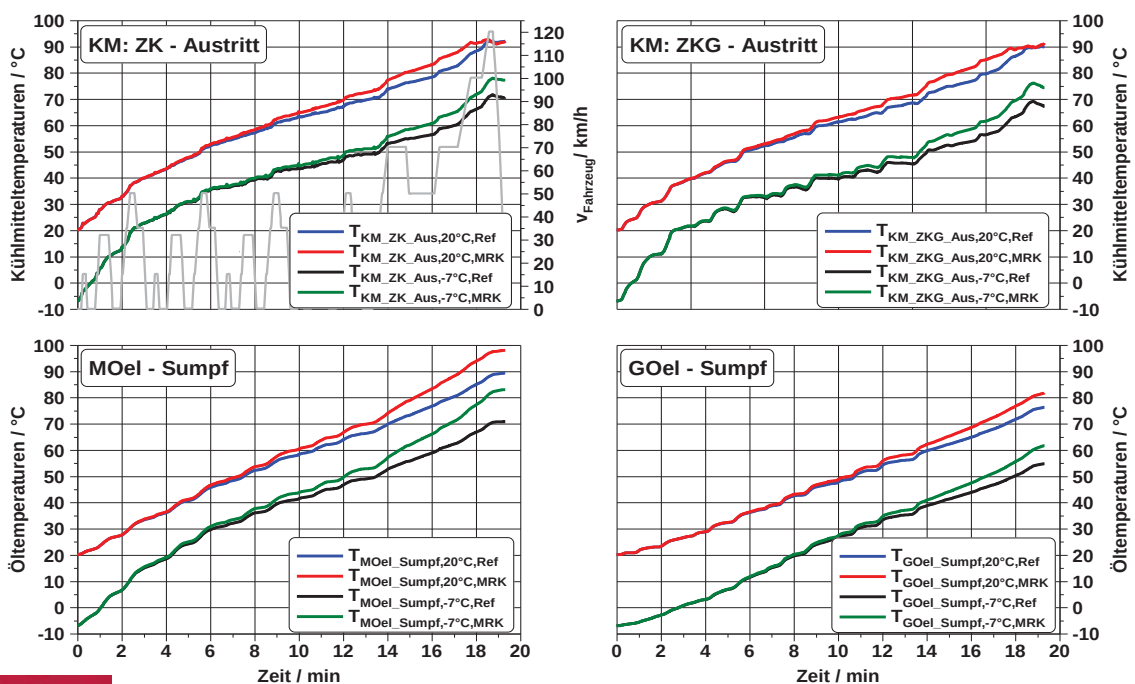


Technische
Universität
Braunschweig



Motorraumkapselung Temperaturverläufe: NEFZ-Zyklus

NEFZ: Temperaturen im Auslasskanal des Zylinderkopfes / Kurbelgehäuse sowie in der Motor- und Getriebeölwanne bei -7 °C und 20 °C

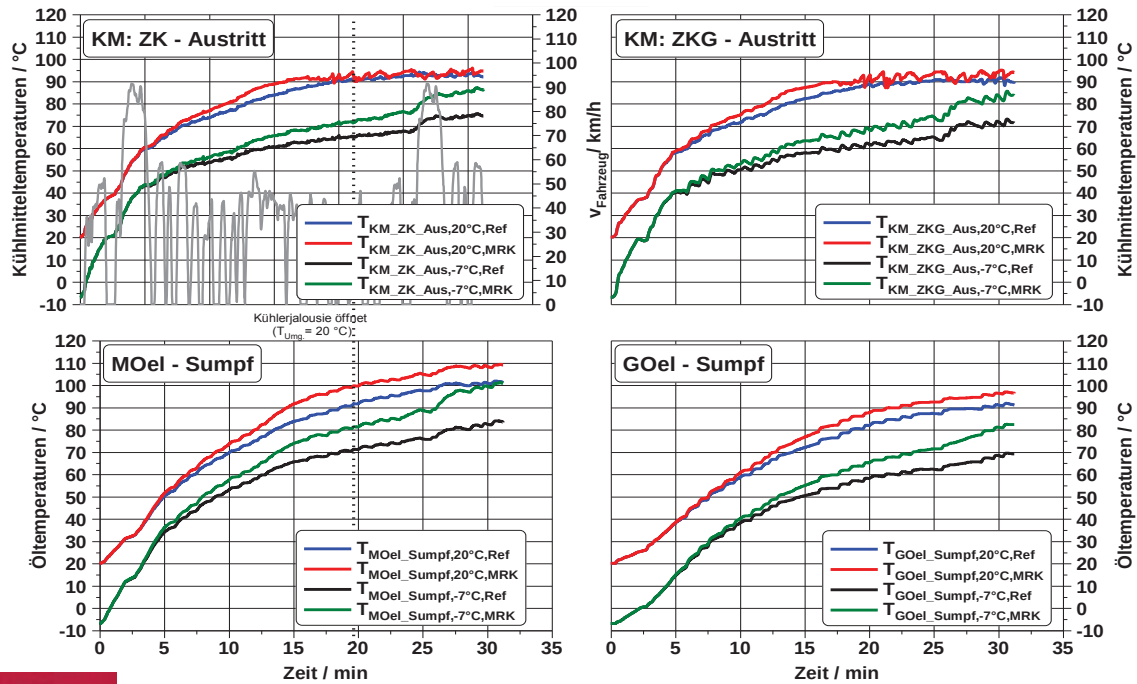


Technische
Universität
Braunschweig



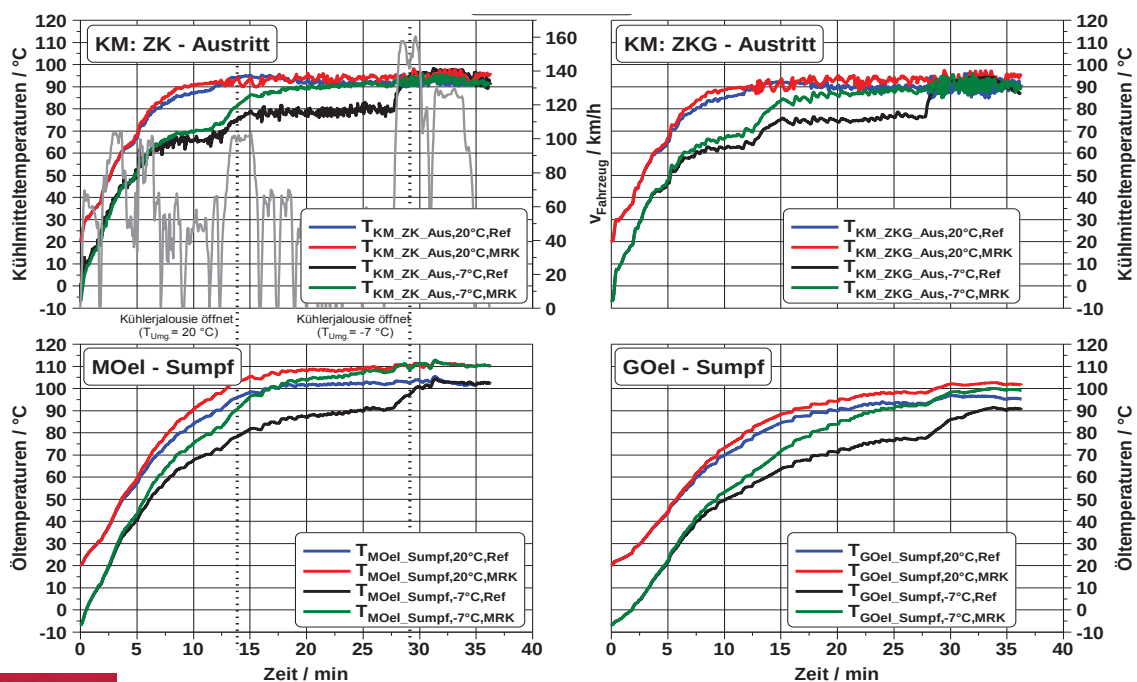
Motorraumkapselung Temperaturverläufe: FTP 75 - Zyklus

FTP 75 Fahrzyklus: Temperaturen im Auslasskanal des Zylinderkopfes / Kurbelgehäuse sowie in der Motor- und Getriebeölwanne bei -7 °C und 20 °C



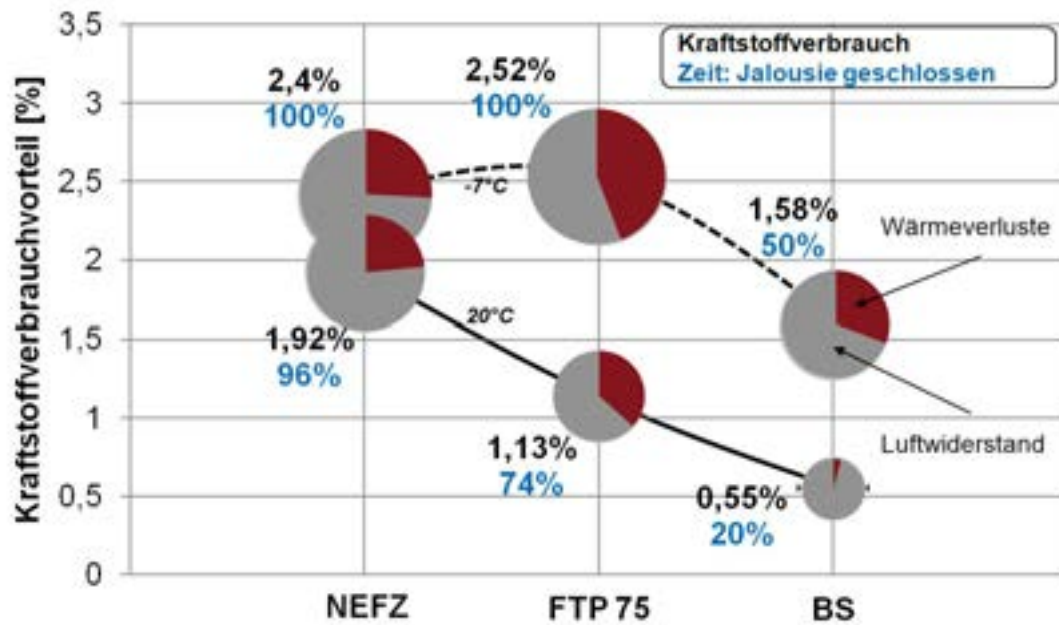
Motorraumkapselung Temperaturverläufe: BS - Zyklus

Braunschweig (BS) Fahrzyklus: Temperaturen im Auslasskanal des Zylinderkopfes / Kurbelgehäuse sowie in der Motor- und Getriebeölwanne bei -7 °C und 20 °C



Kühlerjalousie/Motorraumkapselung Simulationsergebnisse

Zyklen: NEFZ, FTP75, BS-Zyklus / Umgebungstemperatur: -7 C, 20 C



■ : ΔF_{LW} (Luftwiderstand)

■ : ΔQ_{umg} (Wärme an die Umgebung)

Gliederung

Einleitung

Systemmodellierung

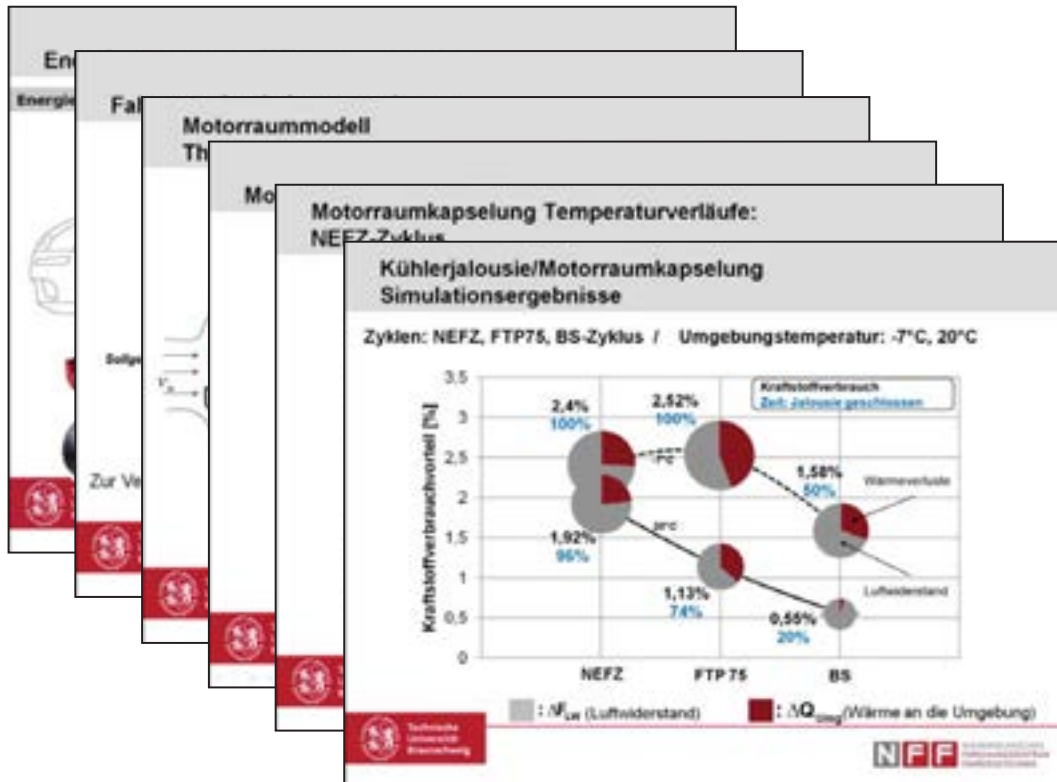
Validierung

Kühlerjalousie – Regelungsstrategie

Ergebnisse

Zusammenfassung

Zusammenfassung



Vielen Dank für Ihre Aufmerksamkeit

Rashad Mustafa, Ferit Küçükay
Institut für Fahrzeugtechnik

Simulation eines Fahrzeugs mit geregelter Hinterradlenkung

Antje Holm,

an.holm@ostfalia.de

Yangfang Yu,

ya.yu@ostfalia.de

Volker Dorsch,

v.dorsch@ostfalia.de

Ostfalia - Hochschule für angewandte Wissenschaften, Fakultät
Maschinenbau, Institut für Konstruktion und angewandten Maschinenbau

Zusammenfassung

In diesem Beitrag wird eine aktive PKW-Hinterradlenkung für ein Simulationsmodell entwickelt. Zunächst erfolgt eine Einführung in den Aufbau des Mehrkörperfahrzeugmodells. Des Weiteren werden in Matlab/Simulink® eine Steuerung, eine Regelung sowie eine Kombination dieser beiden Strategien für eine lenkbare Hinterachse aufgebaut. Diese verbessert das Fahrverhalten des Fahrzeugs und wird über eine Co-Simulation mit dem Fahrzeugmodell eingebunden.

1 Einleitung

Bereits in den 80-iger Jahren gab es einige Fahrzeuge mit einer rein mechanischen Hinterradlenkung. Diese diente nicht nur dem Erreichen eines kleineren Wendekreises, sondern sorgte durch Einstellen eines zusätzlichen Schräglaufwinkels für die Stabilisierung des Fahrzeugs in schnell gefahrenen Kurven. Durch zusätzliche Seitenkräfte konnte so ein Übersteuern vermieden werden. Neben dem ESP und Torque-Vectoring-Systemen, gibt es inzwischen aktive Vorder- und Hinterradlenkungen als weitere Systeme zur Fahrdynamikregelung. Dabei wird über einen Aktor ein zusätzlicher Lenkwinkel aufgegeben, der das Fahrzeug stabilisiert und somit Unter- bzw. Übersteuern vermindert.

2 Aufbau des Fahrzeugmodells

2.1 Fahrzeugkoordinatensystem

Abbildung 1 zeigt das Fahrzeugkoordinatensystem inklusive der Variablen, die zur Beschreibung des Fahrverhaltens nötig sind.

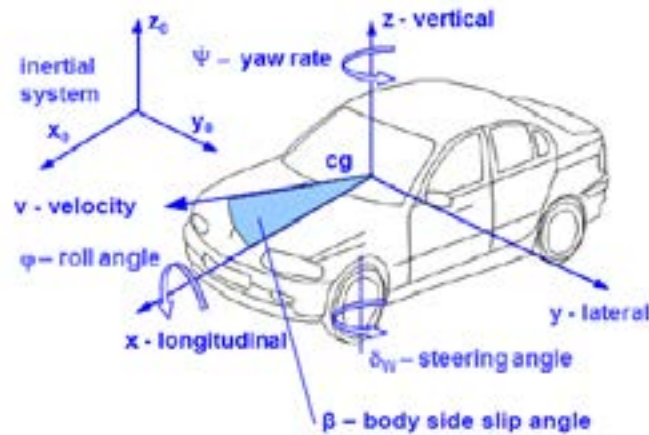


Abbildung 1: Koordinatensystem

Das Gieren beschreibt die Bewegung des Fahrzeugs um dessen Hochachse. Durch die Vorgabe eines Vorderradlenkwinkels wird ein Fahrzeuggierwinkel erzeugt. Die Ableitung des Gierwinkels nach der Zeit wird Gierrate genannt. Sie spielt in dieser Arbeit eine wichtige Rolle.

2.2 Mehrkörperfahrzeugmodell

Der Aufbau des Fahrzeugmodells erfolgt in dem Mehrkörpersimulationstool Simpack® (siehe Abbildung 2). Es besteht aus den Subsystemen Reifenmodell, Vorder- und Hinterachse, Chassis, Lenkung, Antrieb und Bremsen und dient als Basis für die Entwicklung einer geregelten Hinterradlenkung.

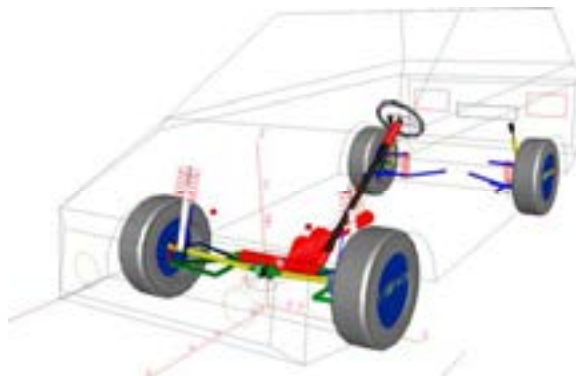


Abbildung 2: Fahrzeugmodell in Simpack®

Um ein realitätsnahes Fahrverhalten und damit korrekte Simulationsergebnisse zu erhalten, wurde das Fahrzeugmodell mit Hilfe von realen Messungen validiert und getestet [1].

3 Entwicklung der Hinterradlenkung

Das zunächst nur vorderradgelenkte Fahrzeug wird hier um eine Hinterradlenkung erweitert. Bei der Entwicklung einer Hinterradlenkung wird zwischen gleichsinnigem und gegensinnigem Lenkeinschlag differenziert (siehe Abbildung 3). Langsame Fahrgeschwindigkeiten sollten von einem gegensinnigen Lenkeinschlag unterstützt werden, weil dadurch der Wendekreis verringert wird. Bei hohen Fahrgeschwindigkeiten sollten die Hinterräder gleichsinnig gerichtet werden. Dadurch kann die Fahrstabilität erhöht werden [3].



Abbildung 3: Radeinschlag bei der Hinterradlenkung bei pos. Vorderradlenkwinkel [2]

Beide Hinterräder des Simulationsmodells werden deshalb jeweils um einen Freiheitsgrad erweitert. Zunächst werden die Auswirkungen auf das Fahrverhalten durch eine „manuelle“ Voreinstellung des Hinterradlenkwinkels im reinen MKS-Modell getestet. Dies dient zur Kontrolle der Funktionsfähigkeit der Hinterradlenkung des Simulationsmodells. Anschließend erfolgt die Modellierung der aktiven Steuerung bzw. der Regelung in Matlab/Simulink®. Über die sogenannte „Simat“-Schnittstelle tauschen das MKS-Programm Simpack® und die Entwicklungsumgebung Matlab/Simulink® ihre Ein- bzw. Ausgabevariablen in jedem Zeitschritt aus. Diesen Prozess nennt man Co-Simulation.

4 Steuerung der Hinterradlenkung

Nach der Erweiterung der Hinterachse um einen zusätzlichen Freiheitsgrad der Räder erfolgt die Modellierung der aktiv gesteuerten Hinterradlenkung. Diese gesteuerte Lenkung ist kennfeldbasiert (siehe Abbildung 4) [4]. Sie nutzt die aktuelle Fahrgeschwindigkeit v und den aktuellen Vorderradlenkwinkel δ_v des Fahrzeugmodells als Eingangsgrößen, um daraus den aktuell benötigten Hinterradlenkwinkel δ_h zu berechnen und stellt diesen an den Hinterrädern ein.

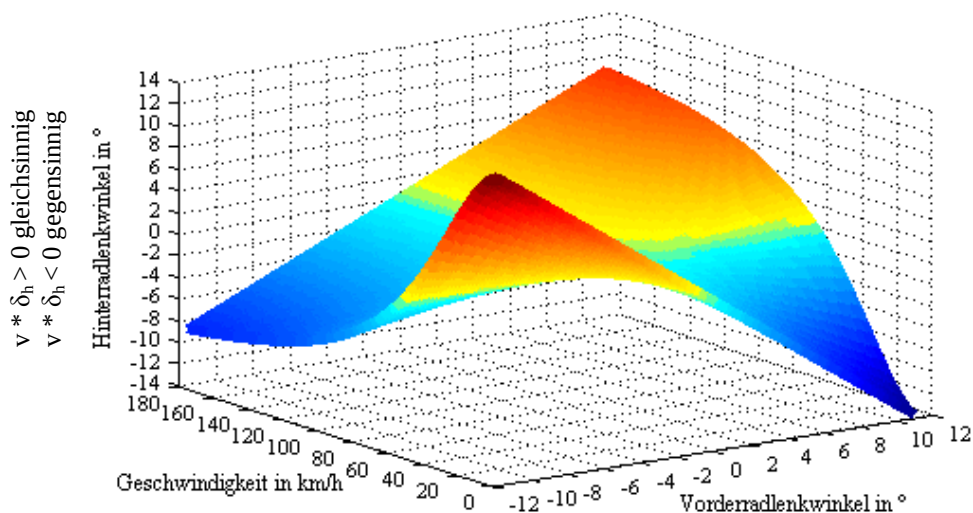


Abbildung 4: Kennfeld zur Steuerung des Hinterradlenkwinkels

Im Gegensatz zur Regelung erfolgt hierbei kein Vergleich zwischen der Soll- und Ist-Gierrate des Fahrzeugs. Einerseits ergibt sich durch die Steuerung aus dem Kennfeld (Abbildung 4) bei kleinen Geschwindigkeiten ein gegensinniger Hinterradlenkwinkel, der den Wendekreis verkleinert. Andererseits stellen sich bei hohen Geschwindigkeiten immer gleichsinnige Hinterradwinkel ein. In kritischen Fahrsituationen kann dies ein unerwünschtes Fahrverhalten sogar verstärken, weil das Fahrzeug nur abhängig vom Vorderradlenkwinkel reagiert. Deshalb wird in einem weiteren Arbeitsschritt eine Hinterradregelung aufgebaut, die das Unter- bzw. Übersteuern berücksichtigt und dieses mit einem verbesserten Lenkwinkel vermindert.

5 Regelung der Hinterradlenkung

5.1 Einspurmodell

Als Führungsgröße des Regelkreises dient die Gierrate $\dot{\psi}_{\text{soll}}$, die mittels eines linearisierten Einspurmodells berechnet wird (siehe Abbildung 5).

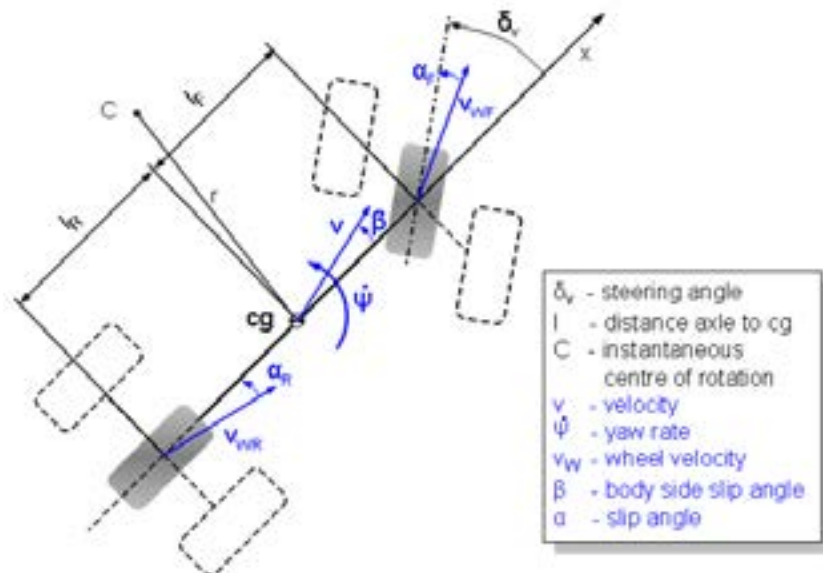


Abbildung 5: Einspurmodell

$$\text{Für das linearisierte Einspurmodell gilt: } \dot{\psi}_{\text{soll}} = \frac{v_x \delta_v}{(l_F + l_R) \left(1 + \frac{v_x^2}{v_{ch}^2} \right)}$$

Die Eingabeparameter sind der Vorderradlenkwinkel δ_v , die Längsgeschwindigkeit v_x , die charakteristische Geschwindigkeit v_{ch} sowie die Abstände von Vorder- und Hinterachse zum Fahrzeugschwerpunkt l_F und l_R .

5.2 Regelstruktur

Im nächsten Schritt erfolgt die Entwicklung einer geregelten Hinterradlenkung (siehe Abbildung 6).

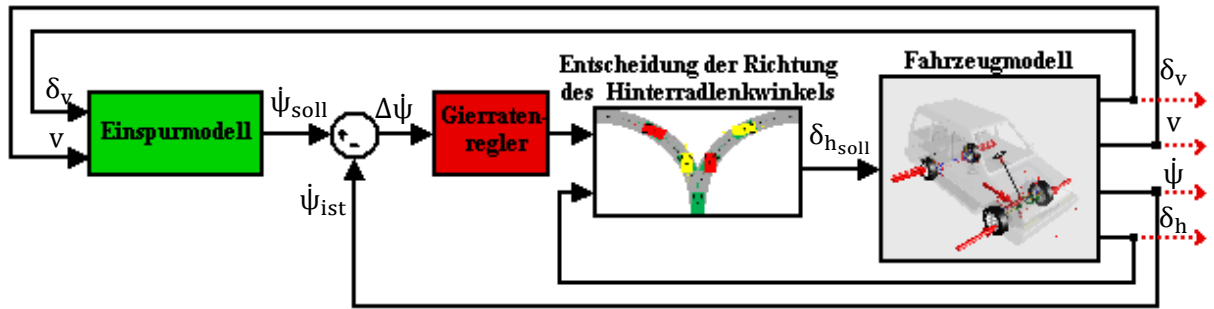


Abbildung 6: Struktur der Co-Simulation mit geregelter Hinterradlenkung

Zusätzlich zur aktuellen Gierrate $\dot{\psi}_{ist}$ werden die aktuellen Messgrößen Fahrgeschwindigkeit v , der gemittelte Vorderradlenkwinkel δ_v und der aktuelle Hinterradlenkwinkel δ_h benötigt. Sie bilden die Eingangsgrößen in den Regelkreis, während der Hinterradlenkwinkel als Ausgangsgröße an das Fahrzeugsimulationsmodell weitergeleitet wird.

Nachdem die Differenz zwischen Soll- und Ist-Gierrate gebildet wurde, übernimmt ein P-Regler die Aufgabe der Verstärkung. Mittels verschiedener Testfälle stellte sich heraus, dass die Regelung durch einen rein proportionalen Anteil die Regelabweichung für die vorgesehenen Zwecke hinreichend ausgleichen kann.

Allerdings wird die Differenz zwischen Soll- und Ist-Gierrate nicht nur als Eingangsgröße in den Gierratenregler benötigt, sondern auch zur Fahrzustandsbeurteilung. Tabelle 1 fasst die Einteilung der Fahrzustände in über-, unter- und neutralsteuerndes Verhalten je nach Vorzeichen der Gierrate zusammen.

Fahrzustand	Bedingung
Übersteuern	$\Delta\dot{\psi} = (\dot{\psi}_{soll} - \dot{\psi}_{ist}) < 0$
Neutral	$\Delta\dot{\psi} = (\dot{\psi}_{soll} - \dot{\psi}_{ist}) = 0$
Untersteuern	$\Delta\dot{\psi} = (\dot{\psi}_{soll} - \dot{\psi}_{ist}) > 0$

Tabelle 1: Fahrzustandsbeurteilung

Die Gierratendifferenz ist als Folge der numerischen Simulation so gut wie nie exakt Null. Jedoch sollen geringe Abweichungen als neutralsteuernd betrachtet werden. Daher wird ein Toleranzbereich eingefügt, indem die Regelung inaktiv ist.

Nachdem der Fahrzustand nach Tabelle 1 beurteilt wird, lässt sich eine weitere Entscheidung über die Richtung des Hinterradlenkwinkels treffen. Die dazu notwendige Vorgehensweise zeigt das Flussdiagramm in Abbildung 7.

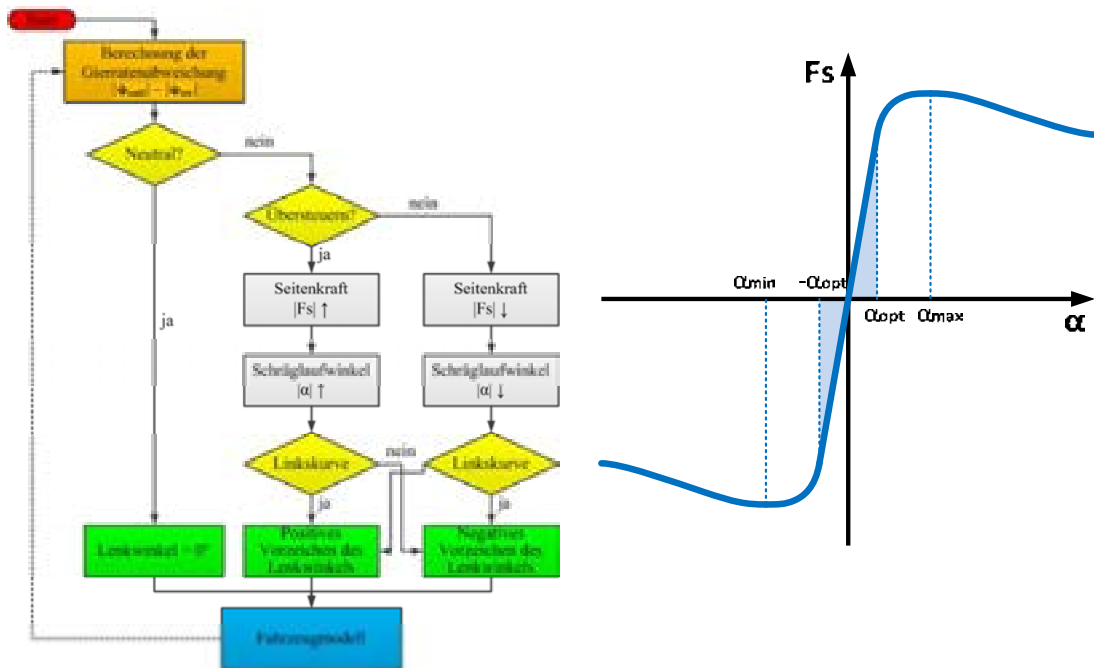


Abbildung 7: Flussdiagramm zur Auslegung des Hinterradlenkwinkels

Beispielsweise fährt ein Fahrzeug in eine Linkskurve ein. Da es zu schnell ist, bricht das Heck zum Kurvenäußeren aus. Dieser Vorgang wird „übersteuern“ genannt. Das Flussdiagramm zeigt, dass in dieser kritischen Fahrsituation der Betrag der Seitenkraft $|F_S|$ an den Hinterrädern erhöht werden sollte. Um dieses zu erreichen, muss auch der Betrag der Schräglaufwinkel $|\alpha|$ vergrößert werden. Da sich das Fahrzeug in diesem Beispiel in einer Linkskurve befindet, erhält der geregelte Lenkwinkel ein positives Vorzeichen. Das bedeutet, dass sich die Hinterräder nach links drehen und somit zunächst gleichsinnig zum Vorderradlenkwinkel ausgerichtet werden. Danach wird der so bestimmte Lenkwinkel an das Fahrzeugmodell in Simpack[®] weitergeleitet und dort eingestellt.

5.3 Kombination von Steuerung und Regelung

Im Anschluss an die Entwicklung der Steuerung und der Regelung werden beide Strategien miteinander kombiniert und in verschiedenen Fahrmanövern getestet. Der durch die kennfeldbasierte Steuerung ermittelte Hinterradlenkwinkel fließt hierbei in die Regelung als Voreinstellung ein. So liegt in jeder Situation schon eine Tendenz für den Lenkwinkel vor, wodurch die Regelung „nur“ noch eine Optimierung des Lenkwinkels vornehmen muss.

6 Simulationsergebnisse

Der Lenkwinkelsprung als beispielhaftes Fahrmanöver liefert die in Abbildung 8 dargestellten Ergebnisse.

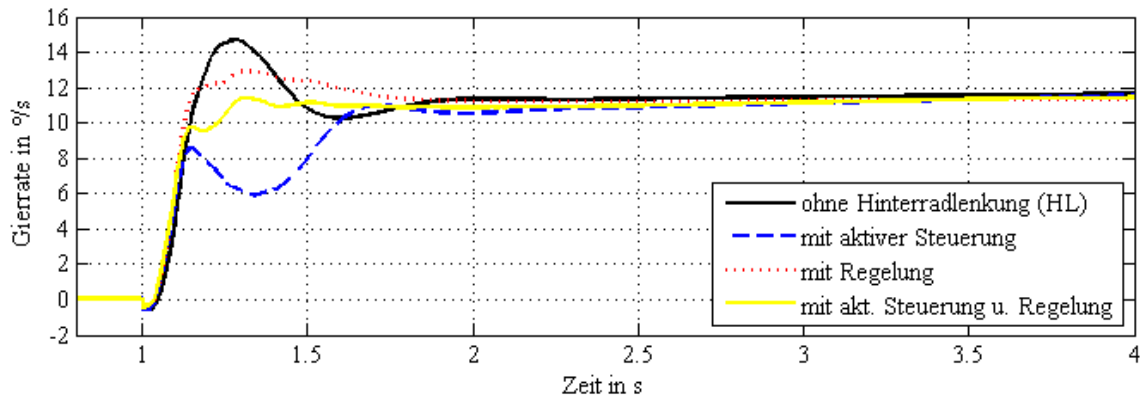


Abbildung 8: Gierratenverlauf während eines Lenkwinkelsprungs

Der gewählte Lenkwinkel ändert sich sprunghaft nach links, so dass sich für die Querbeschleunigung ein Wert von 4 m/s^2 ergibt. Dadurch ist die durch das Einspurmodell berechnete Soll-Gierrate größer als die aktuelle Gierrate des Fahrzeugmodells. Es liegt somit zunächst ein Untersteuerfall vor. Die aktive Steuerung berechnet in diesem Fall einen gleichsinnigen Hinterradlenkwinkel, der dazu führt, dass sich die Gierrate drastisch ändert. Im Gegensatz dazu wird durch die Regelung ein gegensinniger Lenkwinkel eingestellt, wodurch das unerwünschte Überschwingen der Gierrate vermindert werden kann. Der Gierratenverlauf zeigt in diesem Fall bei der Kombination von Steuerung und Regelung die kürzesten und kleinsten Überschwingwerte und somit das beste Ergebnis [5].

In einem weiteren Manöver wird der doppelte Fahrspurwechsel durchgeführt. Die Fahrbahn wird dafür nach ISO 3888-1 aufgebaut. Das Fahrzeug fährt in diesem Fall mit einer Anfangsgeschwindigkeit von 80 km/h in das Manöver ein.

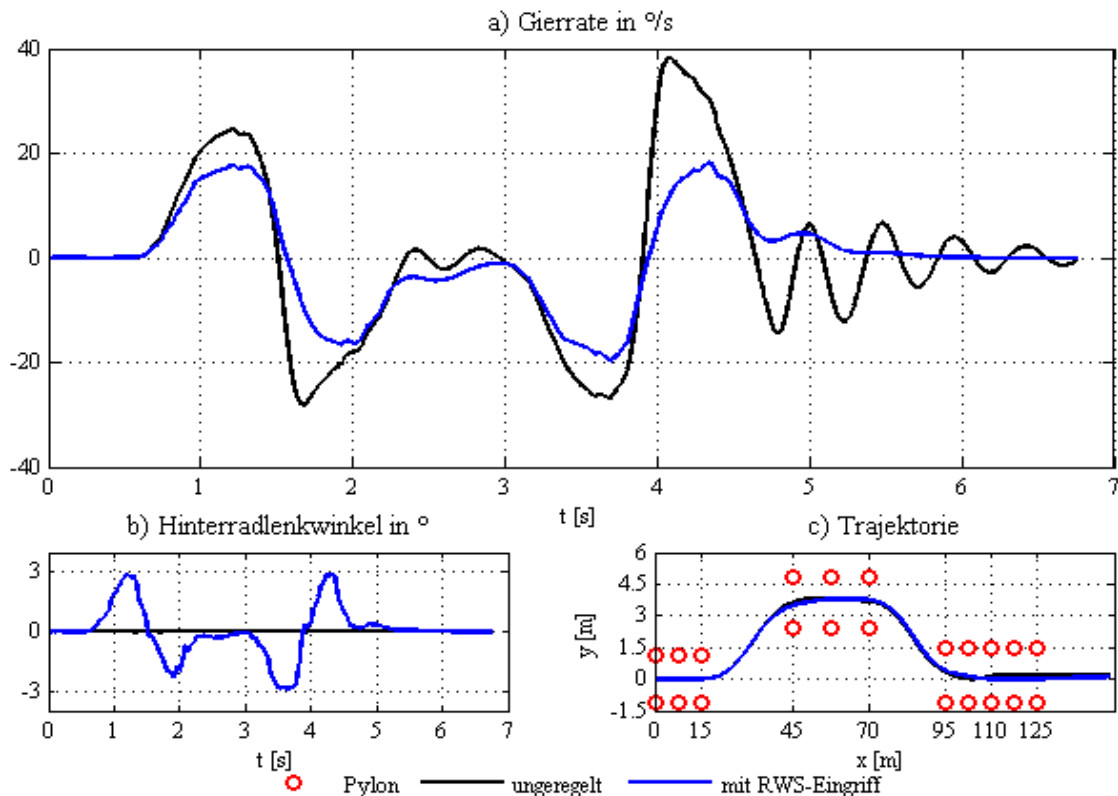


Abbildung 9: Doppelter Fahrspurwechsel

Ohne eine weitere Stabilisierungsmaßnahme bricht das Fahrzeug zwar nicht aus, aber es sind erhöhte Überschwingwerte der Gierrate zu erkennen (siehe Abbildung 9, a)). Das Standard-Fahrermodell des Simpack®-Fahrzeugs richtet die Vorderräder immer zur Sollspur der Fahrbahn aus. Dadurch ergeben sich nach dem zweiten Spurwechsel starke Oszillationen der Gierrate. Die geregelte Hinterradlenkung in Kombination mit der Steuerung führt zu einer deutlichen Abschwächung dieser Schwingungen und einer Verbesserung des Schwingverhaltens während der Kurvendurchfahrt. Die Fahrstabilität kann somit durch die entwickelten Strategien verbessert werden.

7 Fazit und Ausblick

In dieser Arbeit wurde eine Hinterradlenkung zur Stabilisierung des Fahrverhaltens eines Mehrkörpersimulationsfahrzeugs entwickelt. Die Berechnung des benötigten Hinterradlenkwinkels erfolgte in Matlab/Simulink® und wurde durch die sogenannte Co-Simulation mit dem MKS-Programm Simpack® gekoppelt. Nachdem zunächst eine kennfeldbasierte Steuerung programmiert wurde, folgten die Umsetzung einer Regelung und später die Kombination der beiden Strategien. Die Funktionsfähigkeit dieser Systeme in kritischen Situationen wurde durch verschiedene Fahrmanöver, wie dem Lenkwinkelsprung und dem doppelten Fahrspurwechsel, dargestellt.

Da in dieser Arbeit der Radträger einfach um einen Drehfreiheitsgrad ergänzt wurde, sollen sich zukünftige Arbeiten mit der realitätsnahen konstruktiven Veränderung der Radaufhängung beschäftigen. Abbildung 10 zeigt die Hinterradaufhängung des Simpack®-Fahrzeugmodells.

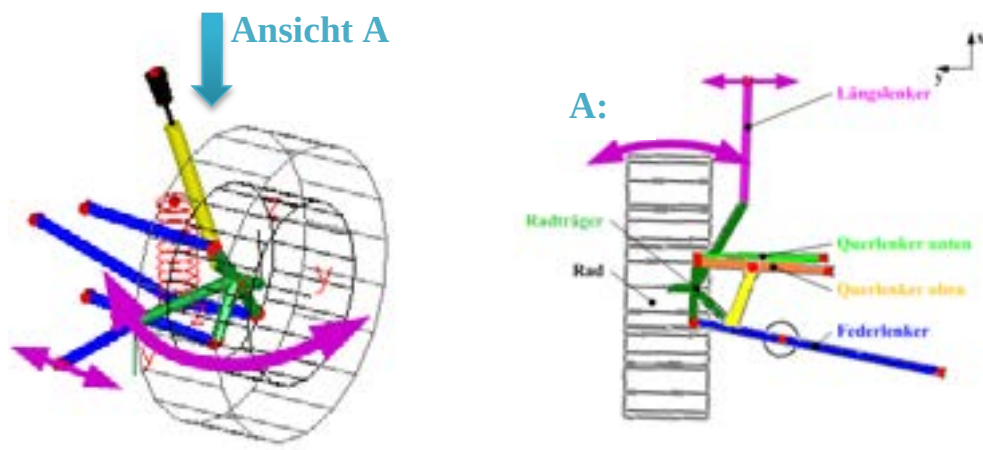


Abbildung 10: Konzept einer lenkfähigen Hinterachse in Simpack®

Zum Beispiel kann die Lage der Längslenkeranbindung am Fahrzeugaufbau verändert werden, wodurch sich das Rad dann schwenken lässt.

Literatur

- [1] Dorsch, V., Holm, A.: *Model-in-the-Loop Simulation including automotive Control Systems*. Simpack User Meeting 2011, Salzburg, Österreich, 18.+19. Mai 2011.
- [2] Heißing, B., Ersoy, M.: *Fahrwerkhandbuch*, Friedr. Vieweg & Sohn Verlag | GWV Fachverlage GmbH, Wiesbaden, 2007.
- [3] Pfeffer, P., Harrer, M.: *Lenkungshandbuch*. Vieweg+Teubner Verlag | Springer Fachmedien Wiesbaden GmbH, Wiesbaden, 2011.
- [4] Woernle, Ch.: *Fahrmechanik*. Skript zur gleichnamigen Vorlesung, Universität Rostock, 2002.
- [5] Yu, Y.: *Erweiterung eines Fahrzeugmodells um eine Hinterradlenkung und eine aktive Lenkung*, Masterarbeit, unveröffentlicht, Ostfalia – HaW, Wolfenbüttel, 2012.

Untersuchungen von Regelstrategien für die Omnibusklimatisierung mit Hilfe einer Gesamtfahrzeugsimulation

Christian Kaiser, TLK-Thermo GmbH

c.kaiser@tlk-thermo.de

Sven Försterling, TLK-Thermo GmbH

s.foersterling@tlk-thermo.de

Wilhelm Tegethoff, TLK-Thermo GmbH

w.tegethoff@tlk-thermo.de

Jürgen Köhler, TU Braunschweig, Institut für Thermodynamik

Juergen.koehler@tu-braunschweig.de

Kurzfassung

Derzeitig konzentrieren sich die Anstrengungen zur Reduzierung des Primärenergieverbrauches von Omnibussen auf die Optimierung der Hardwarekomponenten und der Verschaltung der Komponenten. Für das Klimatisierungssystem des Businnenraumes mittels eines Kaltdampf-Kältekreislaufes bedeutet dies zum Beispiel den Einsatz effizienter Kältemittelverdichter, optimal gestaltete Wärmeübertrager sowie optimierter Verschaltungskonzepte.

Weitere Einsparpotenziale bieten verbesserte Regelungs- und Steuerungskonzepte, insbesondere in Verbindung mit optimierten Klimatisierungssystemen. Ziel ist hierbei das Einsparpotenzial durch bedarfsgerechte Maßnahmen optimal auszunutzen, wodurch bedarfsorientierte Regel- und Steuerungen in modernen Energiemanagementmaßnahmen eine wesentliche Stellung einnehmen.

Für die Untersuchung und Entwicklung energieeinsparender Konzepte für die Klimatisierung eines Omnibusses wird in diesem Beitrag eine Gesamtfahrzeugsimulation vorgestellt, mit der die Auswirkung alternativer Regelungsstrategien auf den Primärenergieverbrauch verglichen und mit einer konventionellen Klimatisierungsregelung bewertet werden kann.

1 Einleitung

Bei der simulativen Entwicklung neuer Konzepte im Bereich der Omnibusklimatisierung wird vermehrt neben Kosten- und Komfortanforderungen zunehmend auch eine Reduktion des Primärenergieverbrauchs angestrebt. In diesem Zusammenhang wurden in den letzten Jahren Studien vorangetrieben deren Augenmerk auf der Optimierung von Hardwarekomponenten und der Verschaltung von Komponenten liegt. Für das Klima- und Heizsystem des Businnenraumes mittels eines Kaltdampf-Kältekreislaufes wird der Einsatz effizienter Kältemittelverdichter, optimierter Verschaltungskonzepte mit Wärmepumpenfunktion sowie Ejektor untersucht (siehe z.B. [1], [2], [3], [6]).

Weitere Einsparpotenziale bietet die Anwendung verbesserter Regelungs- und Steuerungskonzepte, insbesondere in Verbindung mit optimierten Klimatisierungssystemen. Während die Regelung der Businnenraumtemperatur momentan mittels einer Gegenheizung über den Heizkreislauf erfolgt, werden als zukünftige Konzepte verschiedene Regelungen z.B. mit Zylinderbankabschaltung, Drehzahl, Hochdruck und Frischluftzufuhr untersucht ([4], [5]).

Ein wichtiger Beitrag für die differenzierte Untersuchung neuer Regel- und Anlagenkonzepte für Omnibusse ist die Einbeziehung detaillierter Randbedingungen wie z.B. das detaillierte Fahrprofil und die während dieser Fahrt auftretenden Klimarandbedingungen ([5], [7]). Unter Verwendung dieser Randbedingungen können konkrete Verbrauchsaussagen über eine reale Busfahrt bzw. über den Jahresverbrauch einer bestimmten Buslinie getroffen werden. Insbesondere für eine vergleichende simulative Bewertung verschiedener Regel- und Steuerkonzepte können damit verlässliche Aussagen getroffen werden.

Für die simulative Untersuchung des Gesamtfahrzeugsystems werden in bisherigen Studien verschiedene Herangehensweisen gewählt. Einerseits ist es möglich die verschiedenen Komponentenmodelle der Omnibus-Simulation zu einem Modellcode zusammenzufassen und mit einem sehr komplexen Modell zu rechnen. Alternativ kann auch eine hybride Struktur des Simulationsmodells unter Verwendung einer Kosimulationsumgebung verwendet werden [5], [6].

Für die Untersuchung wird in diesem Beitrag eine Gesamtfahrzeugsimulation vorgestellt, mit der die Auswirkung alternativer Regelungsstrategien auf den Primärenergieverbrauch, verglichen und mit einer konventionellen Klimatisierungsregelung, bewertet werden kann. Die Modellstruktur ist so gewählt, dass mit einem Modellcode simuliert wird, wobei aber auch eine Gesamtsystemsimulation unter Verwendung einer Kosimulationsumgebung möglich ist.

Im ersten Abschnitt erfolgt eine Beschreibung des Gesamtmodellverbundes bestehend aus Modellen für den Kältekreislauf, Innenraum, Motorkühl- und Heizkreislauf, elektrisches Bordnetz und die Längsdynamik. Anschließend werden die verwendeten Randbedingungen festgelegt. Die untersuchten Regelungsstrategien werden im Zusammenhang mit der Beschreibung des Klimasteuergerätes diskutiert. Schließlich werden die Regelungsstrategien vergleichend untersucht und hinsichtlich Innenraumtemperatur und Verbrauch bewertet.

2 Modelle des Gesamtfahrzeuges

Nachstehend werden die Modelle für die Gesamtfahrzeugsimulation eines Omnibusses vorgestellt. Hierzu wird zunächst der Aufbau des Gesamtmodellverbundes dargestellt und anschließend die Teilsysteme (Kältekreislauf, Innenraum, Motorkühl- und Heizkreislauf, elektrisches Bordnetz und Längsdynamik) näher beschrieben. Die Randbedingungen für die Simulation und die Regelungsstrategien des Klimasteuergeräts werden in sich anschließenden Kapiteln separat beschrieben.

2.1 Gesamtmodellverbund

Für die Bewertung alternativer Klimatisierungsregelstrategien unter der Prämisse der Komforterhaltung und hinsichtlich des Gesamtenergieverbrauches müssen alle für den Komfort und Verbrauch maßgeblichen Teilsysteme identifiziert und in einem Gesamtmodellverbund zusammengeführt werden. Die in dem Verbund dargestellten Modelle sind in der objektorientierten Modellierungssprache Modelica [8] implementiert. Daher ist es möglich den gesamten Verbund in einer Simulationsumgebung auszuführen. Abbildung 1 zeigt den in Modelica umgesetzten Gesamtmodellverbund.

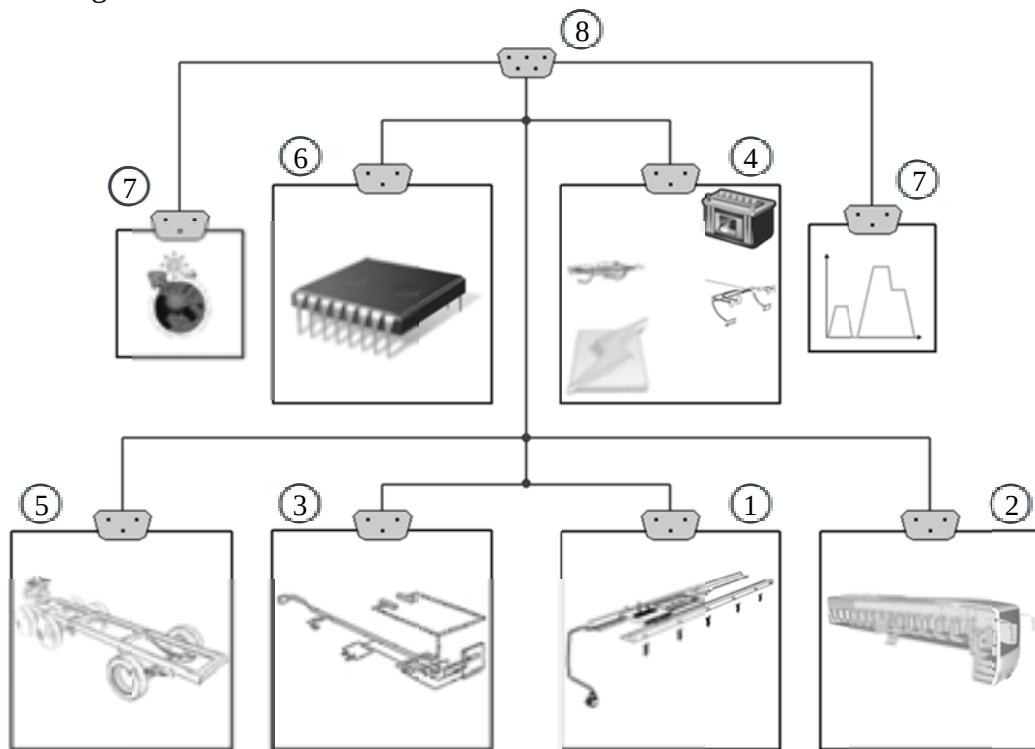


Abbildung 1: Gesamtmodellverbund bestehend aus 1) Kältekreislauf, 2) Innenraum, 3) Motorkühl- und Heizkreislauf, 4) elektrisches Bordnetz, 5) Längsdynamik, 6) Klimasteuergerät, 7) Randbedingungen und 8) Kommunikationsbus (Abbildungen z.T. aus [9])

2.2 Kältekreislaufmodell

Das Teilmodell des Kältekreislaufes ist für die nachfolgenden Untersuchungen in der konventionell eingesetzten Ausführung der Klimaanlage umgesetzt, siehe Abbildung 2. Für

die Modellierung des Kreislaufes werden Modelle aus der Modelica-Bibliothek TIL [10] eingesetzt.

Verflüssiger (2) und Dachverdampfer (5) sind zusammen mit dem Hochdrucksammler (3) in der Kompaktanlage auf dem Dach des Omnibusses untergebracht. Sowohl der Verflüssiger (3) als auch der Dachverdampfer (5) bestehen im realen System aus zwei symmetrisch angeordneten Wärmeübertragern. Im Modell werden diese Wärmeübertrager aufgrund ihrer Symmetrieeigenschaften zu einem Modell für Verflüssiger (2) und Dachverdampfer (5) zusammengefasst.

Der Kältemittelverdichter (1) ist im Motorraum des Omnibusses platziert und wird direkt mit dem Riemenantrieb von der Motorkurbelwelle angetrieben. Der Fahrer-Verdampfer (4) befindet sich im Bug des Fahrzeuges und klimatisiert im Wesentlichen den Fahrerarbeitsplatz. Die elektrischen Expansionsventile (6), kurz EXV, regeln die Überhitzung an den Verdampfern. Die Umluftklappe (7) am Dachverdampfer regelt den Frischluftanteil des Luftvolumenstroms über den Dachverdampfer. Die Umluftklappe am Fahrer-Verdampfer wird in der Praxis nicht im Mischluftbetrieb betrieben, sie kann entweder nur Frischluft oder nur Umluft führen. Das verwendete Kältemittel ist R-134a.

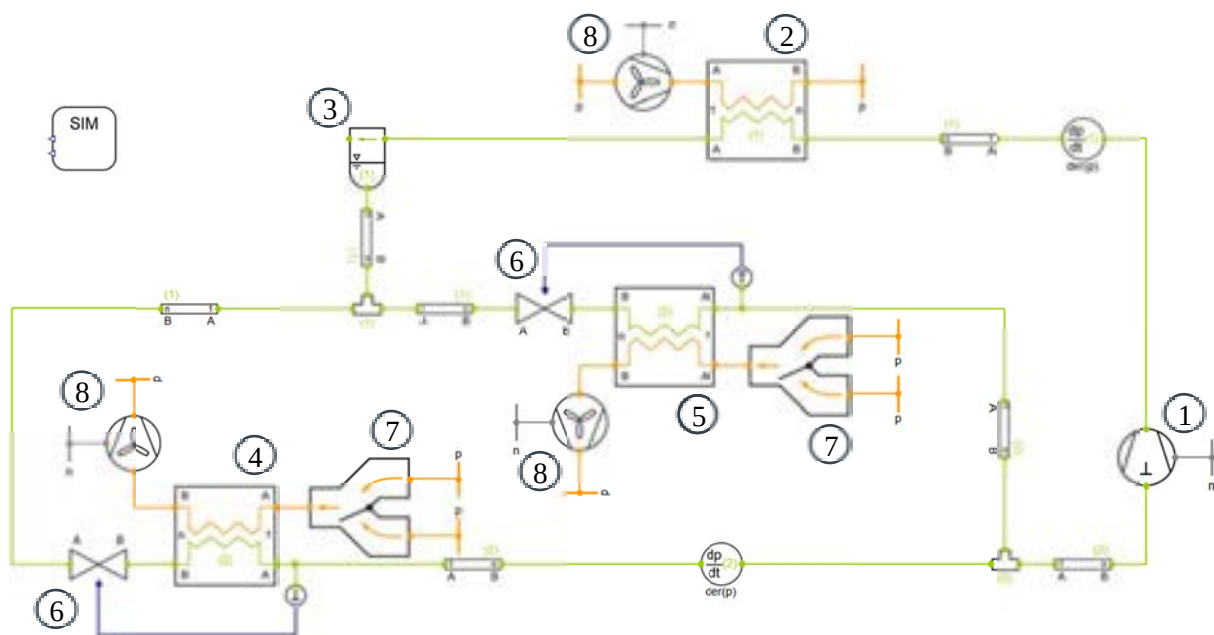


Abbildung 2: Konventioneller Klimakreislauf mit den Komponenten 1) Verdichter, 2) Verflüssiger, 3) Hochdrucksammler, 4) Fahrer-Verdampfer, 5) Dachverdampfer 6) EXV, 7) Umluftklappe und 8) Lüfter.

2.3 Innenraummodell

Der Innenraum des Busses wird mit einer Erweiterungsbibliothek von TIL für Fahrgastinnenräume modelliert. Diese Bibliothek wurde speziell für die Simulation von Fahrzeuginnenräumen entwickelt und beinhaltet Modelle für die Simulation der Fahrgastzelle eines Pkws, von Personenzügen und Omnibussen.

Das eingesetzte Innenraummodell ist in Längsrichtung in drei Sektionen unterteilt. Die vordere Sektion beinhaltet den Fahrerarbeitsplatz und die erste Sitzreihe. Die zweite Sektion

wurde so gewählt, dass diese sich unter der Kompaktanlage befindet. Die dritte Sektion berücksichtigt die letzten zwei Sitzreihen, die Heckscheibe und die Abwärme des Motorraumes.

Abbildung 3 zeigt das Schema der im Modell verwendeten Bilanzräume. Die drei Luftbilanzräume sind hierbei als ideal durchmischte Volumina abgebildet. In der ersten Sektion wird Luft vom Fahrer-Verdampfer eingeblasen. In der zweiten Sektion wird Luft vom Dachverdampfer eingeblasen, zusätzlich wird hier Innenraumlufte für den Umluftbetrieb abgesaugt. Weiterhin befindet sich in der zweiten Sektion die Innenraumentlüftung.

In den Wandmodellen wird konvektiver Wärmeübergang in den Innenraum und zur Umgebung berücksichtigt, ebenso wie langwellige und kurzwellige Strahlung sowie Absorption. Die Scheibenmodelle berechnen zusätzlich transmittierte Strahlung, die in einem vereinfachten Ansatz direkt in den Bilanzraum der Einbauten, beispielsweise die Sitze, eingeht. Als zusätzliche Wärme- und Feuchtequellen werden die Insassen im Modell berücksichtigt.

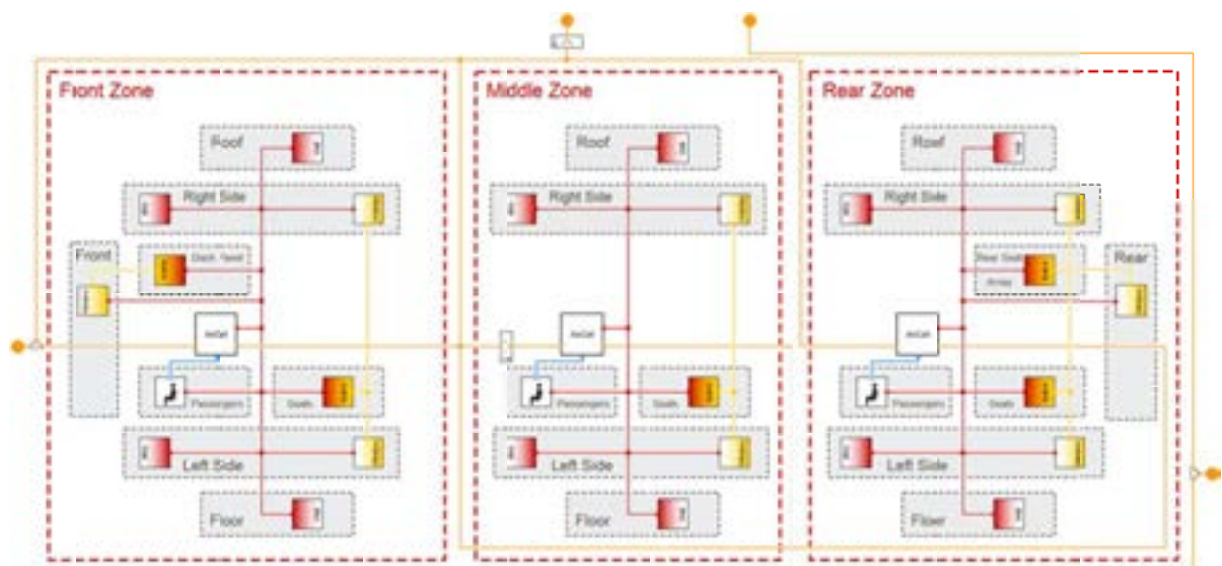


Abbildung 3: Modellschema des Innenraummodells.

2.4 Motorkühl- und Heizkreislaufmodell

Das Motorkühl- und Heizkreislaufmodell (siehe Abbildung 4) ist für die Klimatisierung des Omnibusses von Interesse, da es die Heizwärme für den Innenraum bereitstellt. Diese wird zum einen durch die Motorabwärme, welche im thermischen Motormodell (2) vgl. [11] an den Motorkühlkreislauf übertragen wird, und zum anderen mit einem Zusatzheizgerät (7), zur Verfügung gestellt. Zum Heizen des Innenraumes ist im Luftpfad nach jedem Verdampfer ein Heizungswärmeübertrager (8/9) nachgeschaltet. Zusätzlich werden Konvektorheizungen (10) zum Heizen des Innenraumes eingesetzt. Für die Modellierung des Kreislaufes wird die Modelica-Bibliothek TIL eingesetzt.

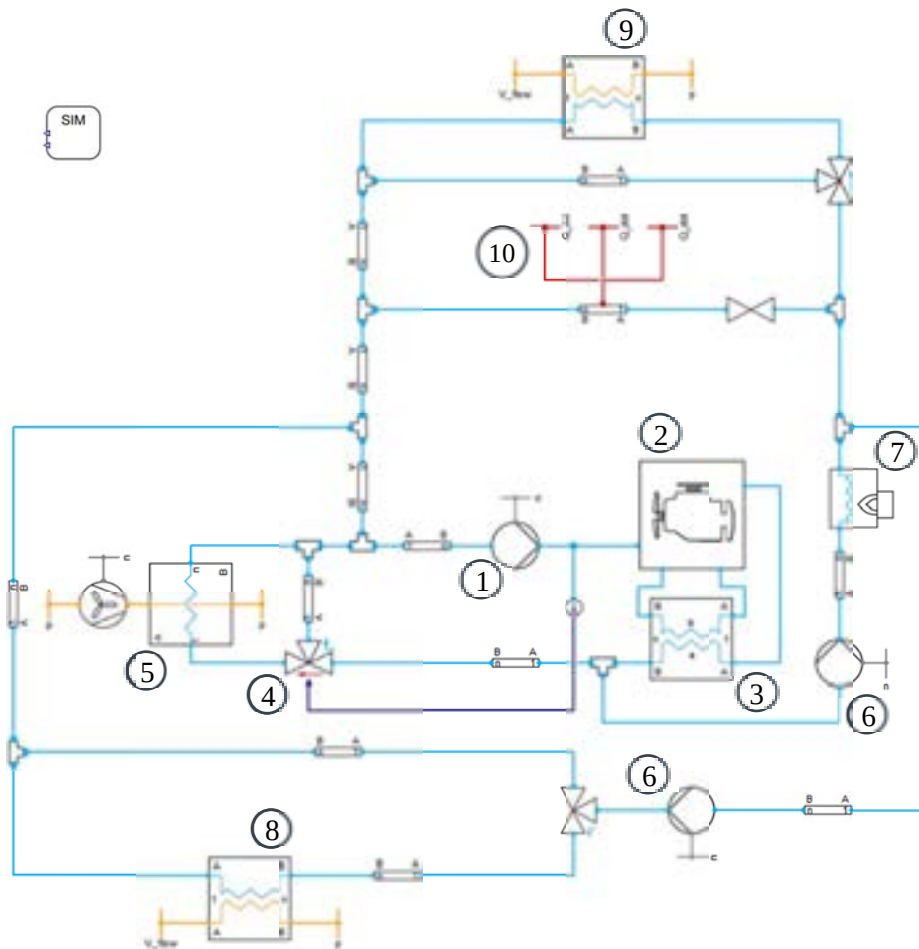


Abbildung 4: Motorkühl- und Heizkreislaufmodell bestehend aus 1) Hauptwasserpumpe, 2) thermisches Motormodell, 3) Motorölkühler, 4) Thermostat, 5) Hauptwasserkühler, 6) Zusatzwasserpumpe, 7) Zusatzheizgerät, 8) Dachheizungswärmeübertrager, 9) Fahrer-Heizungswärmeübertrager und 10) Konvektorheizung.

2.5 Elektrisches Bordnetzmodell

Die Lüfter am Kondensator und den Verdampfern im Kältekreislauf und die Zusatzwasserpumpen im Heizkreislauf werden mit Elektromotoren angetrieben. Damit diese als Verbraucher in der Gesamtenergiebilanz berücksichtigt werden können, wird hierzu ein Modell für ein elektrisches Bordnetz erstellt. Die Modellierung beschränkt sich in diesem Modell auf eine Leistungsbilanzierung der Verbraucher und des Generators. Die Verlustleistung der elektrischen Maschinen und des Generators berechnet sich hierzu aus dem aktuellen Betriebszustand, also der aktuellen Drehzahl und des anliegenden Drehmomentes.

Aus dem Übersetzungsverhältnis zwischen Verbrennungsmotor und Generator ist die Drehzahl des Generators bekannt. Somit kann das nötige Drehmoment des Generators aus der Leistungsbilanzierung berechnet werden und dem Verbrennungsmotor als Zusatzlastmoment aufgeprägt werden.

2.6 Längsdynamikmodell

Zur Berechnung des aktuellen Primärenergieverbrauches wird das in Abbildung 5 dargestellte Einspurlängsdynamikmodell verwendet. Das darin enthaltene Fahrermodell (1)

regelt hierzu mit den Stellgrößen Gaspedalstellung, Bremspedalstellung und Getriebestufe die Ist-Geschwindigkeit an die Soll-Geschwindigkeit eines Fahrprofils. Die dafür benötigte Leistung wird im Fahrzeugkörpermodell (2), Triebstrangmodell (3) und Rad-Straßen-Kontaktmodell (4) berechnet. Zusammen mit der zusätzlichen eingebrachten Last der Nebenaggregate wird im Modell des Verbrennungsmotors (5) der Kraftstoffverbrauch berechnet.

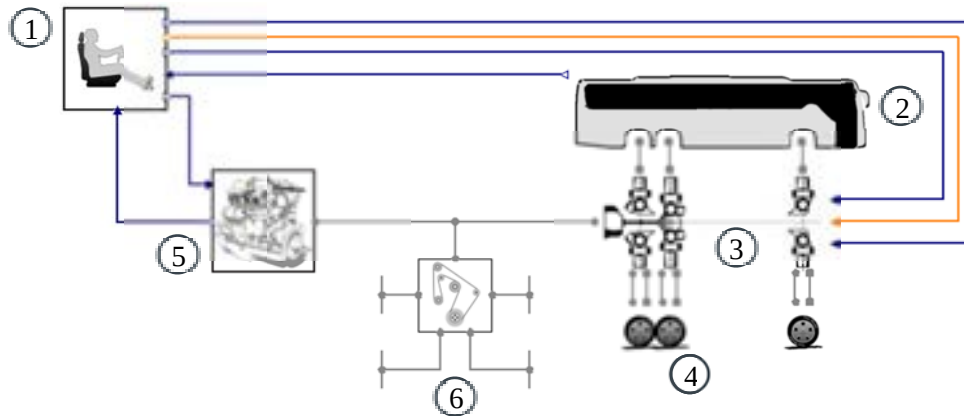


Abbildung 5: Einspurlängsdynamikmodell mit 1) Fahrer, 2) Fahrzeugkörper, 3) Triebstrang, 4) Rad-Straßen-Kontakt, 5) Verbrennungsmotor und 6) Riemenantrieb für Nebenaggregate, wie Klimaverdichter, Kühlerlüfter, Hauptwasserpumpe und Generator.

3 Randbedingungen

Als Randbedingungen werden Fahrzustand und Klimarandbedingungen in der Simulation berücksichtigt. Diese werden im Simulationsmodell in getrennten Modellen abgelegt, um das Simulationsmodell anwenderfreundlich zu gestalten. Die Zusammenstellung der Fahrzustände und der Klimarandbedingungen wird nachfolgend für die in der Simulation eingesetzten Randbedingungen beschrieben.

Für die Ermittlung des Primärenergieverbrauches von Reisebussen gibt es im Gegensatz zur Automobilindustrie (Zyklen zur Verbrauchs- und CO₂-Ausstoßermittlung: NEFZ, FTP75, 10-15 Mode) keine standardisierten Fahrzyklen die das gesamte Fahrzeug berücksichtigen. Einzig der Verbrennungsmotor wird bei schweren Nutzfahrzeugen mit einem standardisierten Lastprofil [12] bezüglich des CO₂-Ausstoßes bewertet.

Aus diesem Grund wird für die nachfolgenden Simulationen eine virtuelle Testfahrt, als spezielle Methode zur transienten Erprobungssimulation, erstellt. Der Fahrzustand ergibt sich hierbei aus einer Reiseroute, die mit Hilfe eines Routenplaners erstellt wird. Die für die Reiseroute geltenden Höchstgeschwindigkeiten definieren dabei das Geschwindigkeitsprofil. Weitere Adaptierungen, wie Abbiegegeschwindigkeiten, Stand- und Haltephasen können zusätzlich integriert werden (vgl. Abbildung 6).

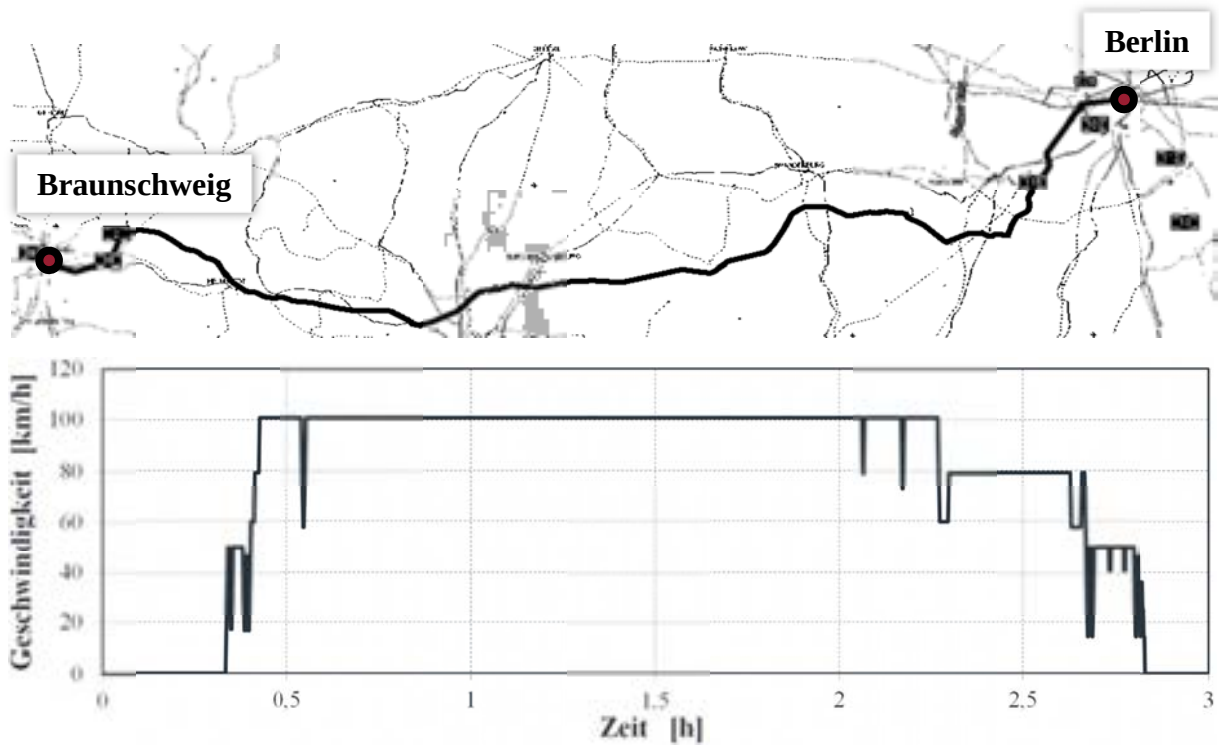


Abbildung 6: Reiseroute (oben) und Geschwindigkeitsprofil (unten) für eine virtuelle Testfahrt.

Für die ermittelte Route können anschließend Klimarandbedingungen mit Hilfe eines Berechnungstools [13] für einen durchschnittlichen, frei wählbaren Tag im Jahr berechnet werden. Die darin enthaltenen Klimagrößen beschreiben den transienten Verlauf der Umgebungstemperatur, der relativen Luftfeuchtigkeit, der solaren Strahlung und des Sonnenstandes entsprechend der zeitlichen Position entlang der Reiseroute, siehe Abbildung 7.

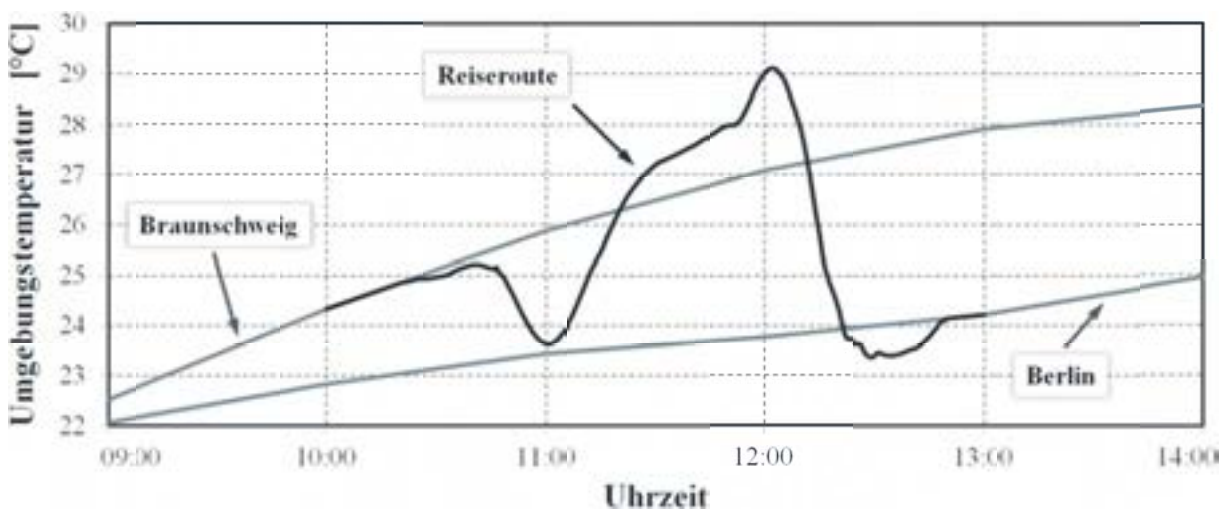


Abbildung 7: Umgebungstemperatur von Start- und Zielort sowie der Reiseroute.

4 Klimasteuergerät und Regelstrategien

Im Teilsystem Klimasteuergerät sind modular die konventionellen Regelstrategien betreffend der Omnibusklimatisierung gemäß [14][15] hinterlegt. Im Wesentlichen gehört hierzu die Regelung der Umluftklappenstellung, die Regelung der Drehzahl von Verdampfer- und Verflüssigerlüfter, die Regelung der Drehzahl der Zusatzwasserpumpen, das Zuschalten des Zusatzheizgerätes sowie die Regelung der Ventilstellung der Bypassventile an den Heizungs-wärmeübertrager.

Die Führungsgröße der aufgeführten Regelungen ist die Soll-Innenraumtemperatur, ausgenommen die Drehzahlregelung des Verflüssigerlüfters, welche durch den maximalen Hochdruck des Kältekreislaufes geführt wird. Ebenso unterscheidet sich die Führungsgröße der Regelung des Zusatzheizgerätes, welche durch die Soll-Wassertemperatur des Heizkreislaufes geführt wird.

Der primär zu identifizierende Verbraucher im Kontext der Klimatisierung ist der Kältemittelverdichter. Dieser wird direkt über die Kurbelwelle mittels Keilriemen und Riemenscheibe angetrieben, wodurch im Großteil des Betriebes eine höhere Kälteleistung erzeugt als benötigt wird. Daher sollen folgende Regelstrategien zur Verdichterregelung getestet werden um mögliche Einsparpotenziale zu ermitteln:

- Kontinuierliche Drehzahlregelung des Kältemittelverdichters mit Hilfe eines CVT-Getriebes (Continuously Variable Transmission) zwischen Verbrennungsmotor und Verdichter, maximales Übersetzungsverhältnis entspricht der konventionellen Übersetzung, minimales Übersetzungsverhältnis entspricht 25% der maximalen Übersetzung, keine Temperaturregelung mittels Gegenheizen
- Zylinderbankabschaltung am Kältemittelverdichter, bei einem Zweibank-Vierzylinder-Verdichter zwischen den Temperaturschwellwerten $\pm 2\text{K}$ von der Soll-Innenraumtemperatur, Gegenheizen erst bei Unterschreiten des unteren Schwellwertes
- Ab- und Einschalten des Kältemittelverdichters zwischen den Temperaturschwellwerten $\pm 2\text{K}$ von der Soll-Innenraumtemperatur, keine Temperaturregelung mittels Gegenheizen

Zusätzlich wird eine Umluftklappenregelung am Dachverdampfer mit der Führungsgröße CO_2 -Konzentration im Innenraum untersucht, wodurch eine Reduzierung der benötigten Kälteleistung am Dachverdampfer erwartet wird, die schließlich zur Reduzierung der Verdichterarbeit führt. Die maximale CO_2 -Konzentration im Innenraum soll den Wert von 0,5 Vol.-%, vgl [16], nicht überschreiten. Daher wird bei Erreichen dieses Wertes die Dachumluftklappe umgeschaltet und der Innenraum mit einer Zeitvorgabe von 10min mit Frischluft gespült.

5 Vergleich der Regelungsstrategien

Nachfolgend werden die im vorangegangenen Kapitel beschriebenen Regelungsstrategien in das Gesamtfahrzeugmodell eingesetzt und die Simulationsergebnisse mit denen des konventionellen Systems verglichen.

Abbildung 8 zeigt die mittlere Innenraumtemperatur der konventionellen Regelung verglichen mit den Innenraumtemperaturen der alternativen Regelungen. Die Toleranz der konventionellen Regelung beträgt hierbei $\pm 1\text{K}$. Bei den alternativen Regelungen sind davon die Regelung der Zylinderbankabschaltung und die der Verdichterabschaltung ausgenommen, da deren Regelbereich größer ist als die Toleranz der Innenraumtemperaturregelung. Darum wird in diesen beiden Fällen die Toleranz auf die Temperaturregelgrenzen erweitert. Die Soll-Innenraumtemperatur ist in allen Untersuchungen auf 23°C eingestellt.

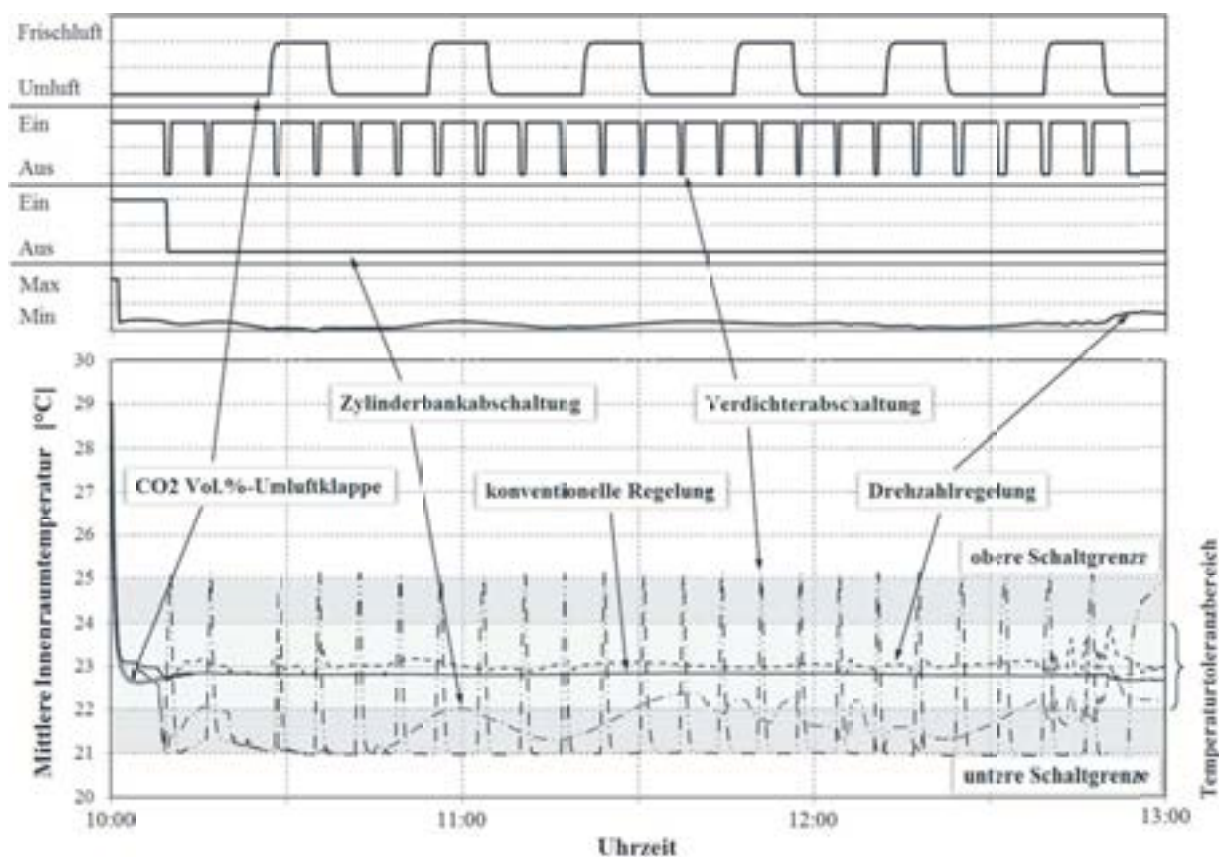


Abbildung 8: Vergleich der mittleren Innenraumtemperaturen der konventionellen und der alternativen Regelungen.

Abbildung 9 zeigt die relativen Gesamtenergieverbräuche der alternativen Regelungen die mit dem Simulationsmodell und den beschriebenen Randbedingungen berechnet wurden. Der Referenzwert der konventionellen Regelung wird durch die Abszisse repräsentiert. Verbrauchsverläufe oberhalb der Abszisse stellen demzufolge einen Mehrverbrauch und Verläufe unterhalb der Abszisse eine Verbrauchsreduzierung dar. Alle untersuchten Alternativen zeigen einen Verbrauchsvorteil gegenüber der konventionellen Regelung. Die

größtmögliche Einsparung kann mit der Drehzahlregelung des Verdichters erreicht werden. Realisiert wird dies durch die bedarfsgerechte Erzeugung von Kälteleistung. Außerdem wird hiermit die gleichmäßigste Innenraumtemperatur, verglichen mit den weiteren Alternativen, erzielt.

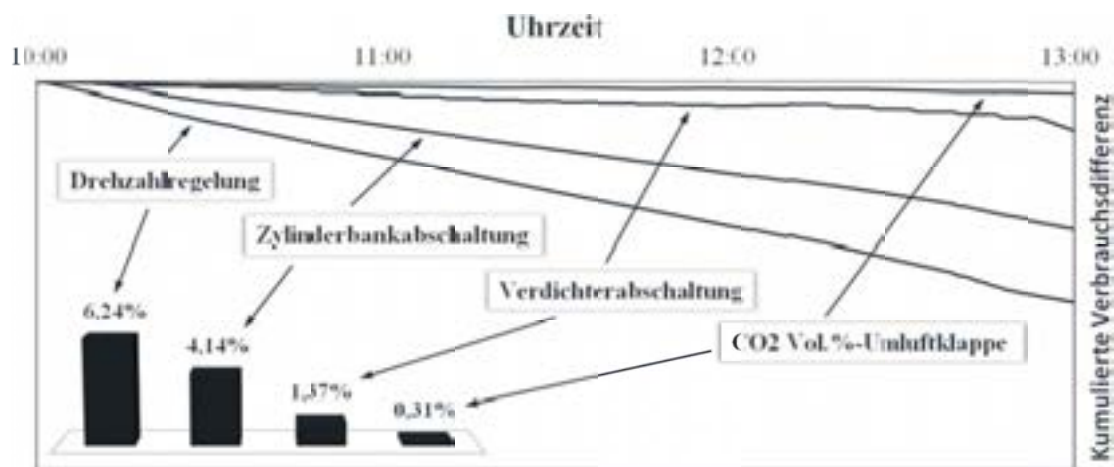


Abbildung 9: Vergleich der relativen Differenz des Primärenergieverbrauches der konventionellen und der alternativen Regelungen.

6 Zusammenfassung und Ausblick

Für die Berechnung der Einsparpotenziale wird eine Gesamtfahrzeugsimulation eines Omnibusses durchgeführt, in der alle mit Komfort und Verbrauch in Verbindung stehenden Teilsysteme berücksichtigt werden. Ebenso wird dargestellt, wie realitätsnah transiente Einsatzbedingungen ermittelt und zu einer virtuellen Testfahrt zusammengefasst werden. Das Gesamtfahrzeugmodell wird innerhalb der virtuellen Testfahrt simuliert und die beschriebenen Regelstrategien zur Klimatisierung des Omnibusses mit der konventionellen Regelung verglichen. Die vorgestellten Untersuchungen von Regelstrategien zeigen, dass allein durch den regelungstechnischen Einfluss folgende Verbrauchsreduzierungen realisiert werden können:

Regelungstyp:	Einsparung:	Komfort:
Konventionell	0%	++
Verdichterdrehzahl	-6.2%	++
Verdichterabschaltung	-1.37%	--
Zylinderbankabschaltung	-4.1%	+-
CO₂ - Umluftregelung	-0.31%	++

Zukünftige Arbeiten konzentrieren sich auf die Optimierung der vorgestellten und zusätzlichen Regelstrategien. Ein Optimierungsschwerpunkt ist dabei die Komforterhaltung durch Minimierung der derzeitigen Temperaturschwankungen. Außerdem werden zusätzlich alternative Kältekreislaufverschaltungen und Kältemittel hinsichtlich des Verbrauchs und Komfort untersucht.

Literatur

- [1] Sonnekalb, Michael; Försterling, Sven; Tegethoff, Wilhelm: *CO₂ basierte Aircondition und Heizung mit Wärmepumpe für Stadtbusse (COACH)*. DKV-Tagung Berlin, 2009
- [2] Raabe, Gabi; Sonnekalb, Michael; Köhler, Jürgen: *Untersuchung eines CO₂-Ejektorskreislaufts für Omnibusklimaanlagen*. Abschlussbericht, Deutsche Bundesstiftung Umwelt, Az: 27385, 2011
- [3] Hafner, Armin; Försterling, Sven; Bø Andreassen, H.E.; Walnum, H.T.: *Air reversing HVAC unit for public trains*. ICR-Tagung, Prag, August 2011
- [4] *Webasto Schulungs-Handbuch*, Kälte-Klima, 12/2000
- [5] Kossel, Roland: *Hybride Simulation thermischer Systeme am Beispiel eines Reisebusses*. Dissertation TU Braunschweig, Oktober 2011
- [6] Kossel, Roland; Försterling, Sven; Tegethoff, Wilhelm: *Einsatz hybrider Simulationstechnik für die Bewertung mobiler Heiz- und Kühlkonzepte*. In: Brill, U. (Hrsg.); Haus der Technik (Veranst.): *Wärmemanagement des Kraftfahrzeugs VI*, Expert-Verlag, Juni 2008, ISBN 978-3-8169-2820-1, S. 150-162
- [7] Strupp, Nils C. ; Lemke, Nicholas: *Klimatische Daten und Pkw-Nutzung (Klimadaten und Nutzungsverhalten zu Auslegung, Versuch und Simulation an Kraftfahrzeug-Kälte-/Heizanlagen in Europa, USA, China und Indien)*. Forschungsvereinigung Automobiltechnik e.V. (FAT), 2010 (FAT-Schriftenreihe 224)
- [8] Modelica Association. *Modelica and the Modelica Association – Modelica Portal*. <http://www.modelica.org> (03.Februar 2012)
- [9] Icon Finder. *Portal For Webdesigners And Developers*, <http://www.iconfinder.net> (03.Februar 2012)
- [10] Tegethoff, Wilhelm et al.: *TEMO –Thermische Echtzeitfähige Modelle*. Abschlussbericht zu einem vom BMBF geförderten Verbundvorhaben mit den Förderkennzeichen 01IS08013A, 01IS08013B und 01IS08013C, Braunschweig 2011
- [11] Samhaber, Christof: *Simulation des thermischen Verhaltens von Verbrennungsmotoren*. Disseration TU Graz, 2002
- [12] Robert Bosch GmbH: *Kraftfahrtechnisches Taschenbuch*. Vieweg Verlag, September 2003, ISBN 3-528-23876-3
- [13] Kossel, Roland et al.: *Distributed Energy System Simulation Of A Vehicle*. Conference Proceedings, Vehicle Thermal Management Systems (VTMS), SAE International, Warwickshire UK, Mai 2011
- [14] MAN, *Technische Information Heizung-Lüftung-Klima*. Service Handbuch, Juli 2003
- [15] Spheros, *Wasser-Heizgeräte*, Werkstatt-Handbuch, Juli 2009
- [16] Temming, J.: *Fahrzeugklimatisierung und Verkehrssicherheit: Auswirkungen sommerlichen Klimas in Kfz auf die Leistungsfähigkeit der Fahrer*. Forschungsvereinigung Automobiltechnik e.V. (FAT), Schriftreihe Nr. 177, 2003

Regelung der Quer- und Gierbewegung eines Elektrofahrzeugs mittels radnahen Direktantrieben

Robert Buchta,
Xiaobo Liu-Henke,
Hochschule Ostfalia
(Hochschule Braunschweig/Wolfenbüttel)
Fakultät Maschinenbau, Institut für Mechatronik, 38302 Wolfenbüttel
Ro.Buchta@ostfalia.de

Zusammenfassung

Fokus dieses Beitrags ist ein Regelungskonzept zur aktiven Beeinflussung der Quer- und Gierdynamik. Dieses Fahrdynamikregelsystem verwendet die Reifenlängskräfte zur Erzeugung des Giermoments als Stellgröße. Kern der Regelung ist eine Entkopplung der Quer- und Gierbewegung. Neben einer Erhöhung der Fahrdynamik hin zu einem neutralen Fahrverhalten kompensiert die Fahrdynamikregelung auch störende Einflüsse wie Seitenwind und trägt damit zur aktiven Fahrsicherheit bei.

1 Querdynamik von Kraftfahrzeugen

Bei konventionellen Fahrzeugen mit Vorderradlenkung ist das Lenk- und Querverhalten abhängig von der Fahrgeschwindigkeit. Eine Analyse der Eigenwerte des charakteristischen Verhaltens der Eigenwerte der Fahrzeugquerdynamik (1) zeigt, dass mit zunehmender Fahrgeschwindigkeit die Dämpfung der Gierbewegung sinkt (Abbildung 1). Vom Fahrer erfordert dies Übung und Erfahrung ein Fahrzeug bei schnellen Lenkmanövern, z.B. in kritischen Situationen, unter Kontrolle zu halten. Mit aktiven Systemen kann ein unerwünschtes Verhalten verbessert und ein wesentlicher Beitrag zur Erhöhung der Fahrsicherheit geleistet werden. Weiterhin kann auch der Fahrkomfort positiv beeinflusst werden.

Aktive Systeme können in ihren Stellgrößen unterschieden werden. Neben Systemen, die aktiv zusätzliche Reifenquerkräfte erzeugen wie eine Überlagerungslenkung oder eine Hinterradlenkung, existieren Systeme, welche ein Giermoment durch die Veränderung der Reifenlängskräfte bewirken. Für ein Elektrofahrzeug mit radnahen Direktantrieben bietet es sich an, diese zum Stellen des erforderlichen Giermoments zu verwenden. Damit wird einer Erhöhung der Fahrzeugmasse durch den Wegfall zusätzlicher Systeme entgegengewirkt und

die Möglichkeit des Rekuperierens durch fahrdynamische Regeleingriffe wird eröffnet [2]. In dem in diesen Beitrag vorgestellten Regelkonzept werden radnahe Antriebe als Aktoren der Fahrdynamikregelung verwendet.

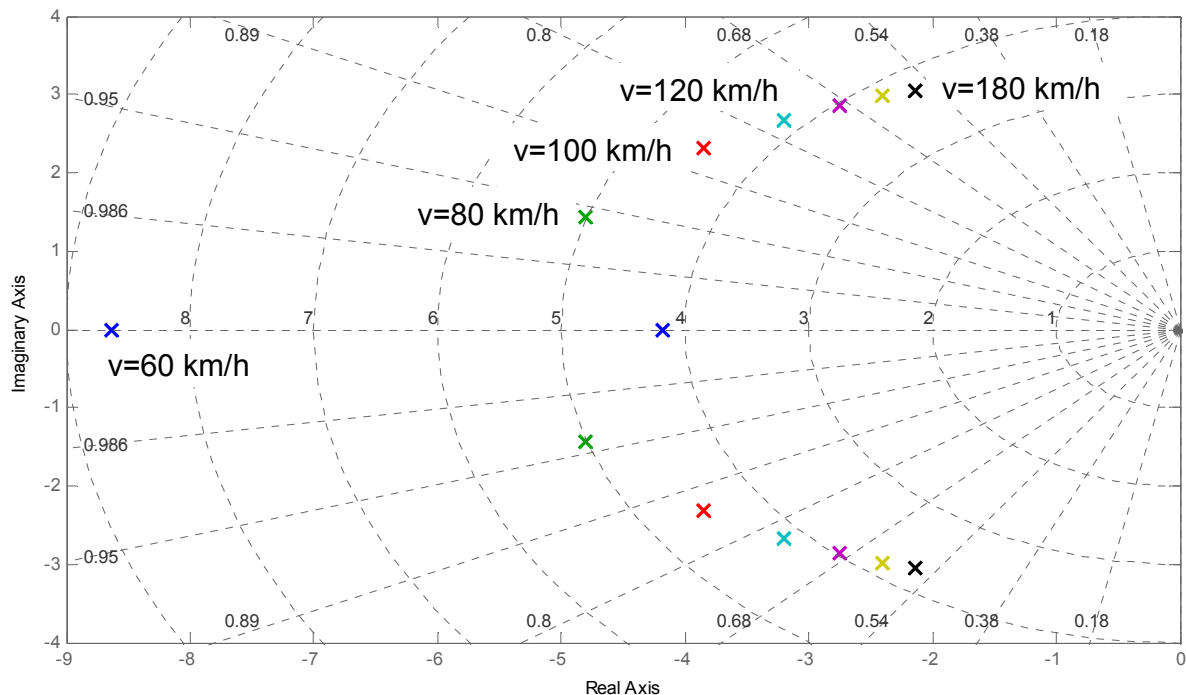


Abbildung 1: Einfluss der Fahrgeschwindigkeit auf die Eigenwerte der Querdynamik

Ein konventionelles Fahrzeug mit Vorderradlenkung und einem Giermoment zur gewünschten Beeinflussung des Fahrverhaltens stellt ein vermaschtes Mehrgrößensystem dar, bei dem die Quer- und Gierbewegung miteinander verkoppelt sind. Diese Schwierigkeit stellt für die Reglerauslegung eine besondere Herausforderung dar. Nachfolgend wird die Synthese einer Fahrdynamikregelung, die auf einer Entkoppelung basiert, beschrieben.

2 Modellbasierte Reglerauslegung

Die Regelung der Fahrdynamik für das Elektrofahrzeug M-Mobile ist hierarchisch strukturiert (Abbildung 2). An oberster Hierarchiestufe steht die Querführung des Fahrzeugs bestehend aus einer antizipatorischen Steuerung der Querbewegung und einer kompensatorischen Regelung der Abweichung von der Solltrajektorie. Diese Aufgabe kann von einem menschlichen Fahrer oder bei einer autonomen Querführung von einer Informationsverarbeitung mit entsprechender Sensorik und Aktorik erfüllt werden. Die Querführung liefert die Referenzgrößen für die unterlagerte Fahrdynamikregelung. Auf die Querführung wird in diesem Beitrag nicht näher eingegangen. Mit der Fahrdynamikregelung wird das gewünschte kontrollierte Quer- und Gierverhalten des Fahrzeugs eingestellt. Als Stellgröße verwendet es ein Giermoment M_z , welches durch die hierarchisch unterlagerte Regelung eingestellt wird.

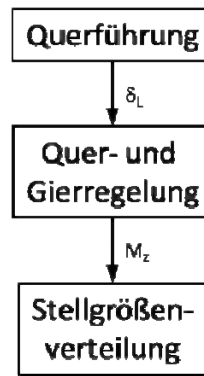


Abbildung 2: Hierarchische Struktur der Fahrdynamikregelung

Zur modellbasierten Reglerauslegung wird das Einspurmodell nach Rieckert und Schunck verwendet, welches um ein Giermoment M_z erweitert wurde (Abbildung 3). Bei diesem Einspurmodell werden die Räder achsweise zusammengefasst, die Aufbaubewegung wird vernachlässigt und das Reifenverhalten wird als linear betrachtet. Unter Vernachlässigung des degressiven Reifenverhaltens genügt das Modell für Untersuchungen der Querdynamik bis zu einer Querbeschleunigung von $a_y = 4 \text{ m/s}^2$ [4]. Die Darstellung im Zustandsraum mit dem Zustands- und Eingangsvektor lautet:

$$\begin{aligned} \dot{\underline{x}}_{ESM} &= \underline{A}_{ESM} \cdot \underline{x}_{ESM} + \underline{B}_{ESM} \cdot \underline{u}_{ESM} \\ \underline{y}_{ESM} &= \underline{C}_{ESM} \cdot \underline{x}_{ESM} + \underline{D}_{ESM} \cdot \underline{u}_{ESM} \end{aligned} \quad \text{mit } \underline{x}_{ESM} = \begin{bmatrix} \beta \\ \dot{\psi} \end{bmatrix} \text{ und } \underline{u}_{ESM} = \begin{bmatrix} \delta_v \\ M_z \end{bmatrix} \quad (1)$$

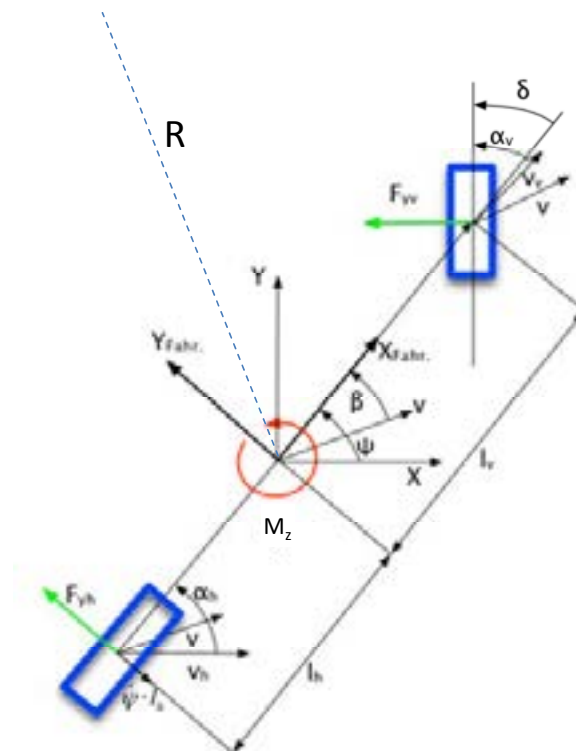


Abbildung 3: Einspurmodell erweitert um Giermoment M_z

Zum Reglerentwurf eines vermaschten Mehrgrößensystems ist es hilfreich dieses in mehrere Eingrößensysteme zu zerlegen und dann die Regelungen separat für jedes Teilsystem zu entwerfen. In [1] wird eine robuste Regelung mit einer Hinterradlenkung zur Beeinflussung des Quer- und Gierverhaltens aufgezeigt, dessen Kern eine triangularisierende Entkopplung der Quer- und Gierbewegung ist. Dies erfolgt durch sinnvolle Auswahl einer neuen Basis des Zustandsraums für das Einspurmodell und einer Koordinatentransformation. Für das Konzept der hier vorgestellten Fahrdynamikregelung wurde die Idee aufgegriffen und auf ein Fahrzeug mit radnahen Direktantrieben modifiziert und übertragen. Mit der Transformationsmatrix T ergibt sich der neue Zustandsvektor und die transformierten Zustandsgleichungen zu:

$$\underline{\underline{T}} = \begin{bmatrix} c_1 & c_2 & d_1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2)$$

$$\begin{aligned} \dot{\underline{x}} &= \underline{\underline{A}} \cdot \underline{x} + \underline{\underline{B}} \cdot \underline{u} \\ \underline{y} &= \underline{\underline{C}} \cdot \underline{x} + \underline{\underline{D}} \cdot \underline{u} \end{aligned} \quad \text{mit } \underline{x} = \begin{bmatrix} a_v \\ \dot{\psi} \\ \delta_v \end{bmatrix} \text{ und } \underline{u} = \begin{bmatrix} e_v \\ M_z \end{bmatrix} \quad (3)$$

Im transformierten Zustandsvektor ist der Schwimmwinkel eliminiert und an Stelle dessen ist die Querschleunigung der Vorderräder eine neue Zustandsgröße. Zum Gelingen der Entkopplung ist der Radlenkwinkel der Vorderräder als integrierendes Stellglied ohne Rückführung angenommen. Damit wird der Radlenkwinkel eine weitere Zustandsgröße.

Zu den Vereinfachungen des Einspurmodells wird eine weitere Annahme getroffen. Die Fahrzeugmasse wird auf die Vorder- und Hinterachse verteilt, so dass gilt:

$$m_v \cdot l_v = m_h \cdot l_h \quad (4)$$

Bei der Berechnung des Trägheitsmoments um die Hochachse führt diese Annahme zu einer Darstellung des Gierträgheitsmoments aus der Fahrzeugmasse und den Schwerpunktabständen:

$$\begin{aligned} \Theta_{zz} &= m_v \cdot l_v^2 + m_h \cdot l_h^2 \\ &= m_h \cdot l_h \cdot l_v + m_v \cdot l_v \cdot l_h \\ &= m \cdot l_h \cdot l_v \end{aligned} \quad (5)$$

Diese Annahmen tragen zur Entkopplung des Systems bei. Weiterhin ist für die Entkopplung der Quer- und Gierbewegung die Rückführung der Gierrate auf den ersten Systemeingang notwendig:

$$e_v = -\dot{\psi} + u_v \quad (6)$$

Die damit gewonnene Zustandsdarstellung lautet:

$$\begin{aligned} \begin{bmatrix} \dot{a}_v \\ \ddot{\psi} \\ \dot{\delta}_v \end{bmatrix} &= \begin{bmatrix} d_{11} & 0 & 0 \\ d_{21} & d_{22} & d_{23} \\ 0 & -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} a_v \\ \dot{\psi} \\ \delta_v \end{bmatrix} + \begin{bmatrix} d_1 & e_1 \\ 0 & b_{23} \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} e_v \\ M_z \end{bmatrix} \\ \begin{bmatrix} a_v \\ \dot{\psi} \\ \delta_v \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_v \\ \dot{\psi} \\ \delta_v \end{bmatrix} + \begin{bmatrix} 0 & d_3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} e_v \\ M_z \end{bmatrix} \end{aligned} \quad (7)$$

Für eine Entkopplung der Quer- und Gierbewegung stört der Einfluss des Giermoments M_z auf die Änderung der Querbeschleunigung in der Eingangsmatrix. Dieser Einfluss kompensiert sich jedoch mit dem Durchgriff von M_z auf die Querbeschleunigung stationär. Eine vollständige Kompensation auch im Instationären kann durch ein ideales PD-Glied erfolgen. Diese Maßnahme wurde mit einem realen PD-Glied durchgeführt. In einem Blockschaltbild werden die Eingriffe aufgezeigt (Abbildung 4).

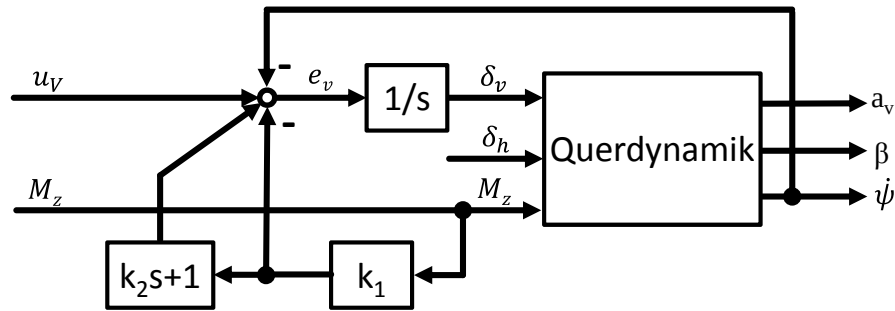


Abbildung 4: Blockschaltbild mit Maßnahmen zur Entkopplung

Dieses System als Zustandsraum dargestellt lautet:

$$\begin{aligned} \begin{bmatrix} \dot{a}_v \\ \ddot{\psi} \\ \dot{\delta}_v \end{bmatrix} &= \begin{bmatrix} d_{11} & 0 & 0 \\ d_{21} & d_{22} & d_{23} \\ 0 & -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} a_v \\ \dot{\psi} \\ \delta_v \end{bmatrix} + \begin{bmatrix} d_1 & 0 \\ 0 & b_{23} \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} u_v \\ M_z \end{bmatrix} \\ \begin{bmatrix} a_v \\ \dot{\psi} \\ \delta_v \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_v \\ \dot{\psi} \\ \delta_v \end{bmatrix} \end{aligned} \quad (8)$$

Aus (8) wird folgendes erkennbar:

- Die Querbeschleunigung der vorderen Räder a_v ist nicht vom Systemeingang M_z steuerbar
- Die Gierrate $\dot{\psi}$ und der Radlenkwinkel der vorderen Räder δ_v sind nicht von der Querbeschleunigung der vorderen Räder a_v beobachtbar.

Die Lenkdynamik ist nun in zwei Teilsysteme getrennt werden, die zur Reglerauslegung separat betrachtet werden können.

Wird mit der folgenden Gleichung die Querbeschleunigung der Vorderräder auf den Systemeingang u_v zurückgeführt, so ergibt sich ein PT1-Übertragungsverhalten für die Querbeschleunigung der Vorderräder (9), das unabhängig von der Fahrgeschwindigkeit ist und dessen Verzögerungszeit über den Faktor k_s gewählt werden kann.

$$u_v = k_s \cdot (a_{vref} - a_v) + \frac{1}{T_{av}} \cdot a_v \quad (9)$$

$$\frac{a_v(s)}{a_{v,ref}(s)} = \frac{1}{1 + T_{av} \cdot s} \quad \text{mit } T_{av} \sim \frac{1}{k_s} \quad (10)$$

Im zweiten Teilsystem, welches die Gierbewegung abbildet, besteht das charakteristische Polynom aus einem PT2-Element mit der Eigenkreisfrequenz und dem Dämpfungsmaß:

$$\omega_0 = \sqrt{\frac{c_h}{m \cdot l_v}} \quad (11)$$

$$D = \frac{l}{2 \cdot v} \sqrt{\frac{c_h}{m \cdot l_v}} \quad (12)$$

Das Dämpfungsmaß ist antiproportional zur Fahrgeschwindigkeit. Mit zunehmender Fahrgeschwindigkeit nimmt das Dämpfungsmaß ab, das Fahrzeug wird schwerer beherrschbar. Mit der folgenden Rückführung der Gierrate auf den Systemeingang M_z (13) lässt sich ein geschwindigkeitsunabhängiges und über den Parameter k_3 gewünschtes Dämpfungsmaß einstellen.

$$M_z = u_{M_z} - (k_4 - k_3) \cdot \dot{\psi} \quad (13)$$

Das Dämpfungsmaß und die Eigenkreisfrequenz für das Verzögerungsglied 2. Ordnung des geregelten Systems entsprechen:

$$\omega_0 = \sqrt{\frac{c_h}{m \cdot l_v}} \quad (14)$$

$$D = \frac{k_3}{2 \cdot l_h} \sqrt{\frac{1}{m \cdot l_v \cdot c_h}} \quad (14)$$

Mit der durchgeführten Entkopplung werden die beiden regelungstechnischen Probleme der Querbewegung und der Dämpfung der Gierbewegung separat gelöst. In Abbildung 2 ist das Blockschaltbild der Fahrdynamikregelung dargestellt.

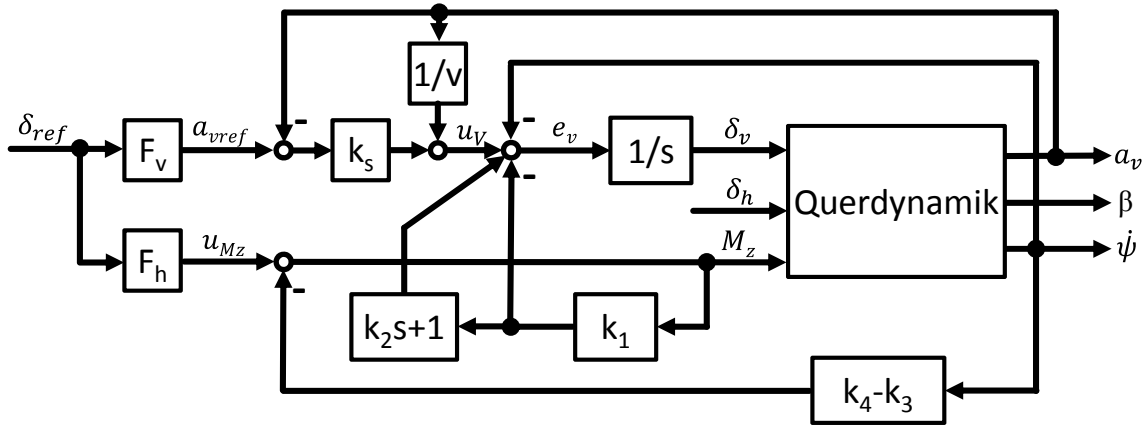


Abbildung 5: Blockschaltbild der Fahrdynamikregelung

Das Vorfilter F_v wurde hinsichtlich eines gewünschten neutralen Fahrverhaltens gewählt. Für eine optimale Fahrdynamik können verschiedene fahrdynamische Ziele definiert werden, jedoch wird ein neutrales Fahrverhalten von den meisten Fahrern als fahrdynamisch sehr gut empfunden. Mit dem Vorfilter F_h wird der Endwert des Radlenkwinkels der Vorderräder proportional zum Referenzwinkel aus der überlagerten Fahrzeugquerführung eingestellt.

Als unterlagerte Regelung zur Umsetzung der Referenzgröße aus der Fahrdynamikregelung erfolgt derzeit lediglich eine einfache Aufteilung des Giermoments auf alle vier Räder proportional zur Radlast, da diese die Größe des Kamm'schen Kreises und damit das Potential zur Kraftübertragung bestimmt. Unter Berücksichtigung der dynamischen Radlastverteilung gilt für die Radlasten:

$$\begin{aligned} F_{VL} &= \frac{m}{2} \cdot \left(g \cdot \frac{l_h}{l} - a_x \cdot \frac{h}{l} - a_y \cdot \frac{h}{s_v} \right) \\ F_{VR} &= \frac{m}{2} \cdot \left(g \cdot \frac{l_h}{l} - a_x \cdot \frac{h}{l} + a_y \cdot \frac{h}{s_v} \right) \\ F_{HL} &= \frac{m}{2} \cdot \left(g \cdot \frac{l_v}{l} + a_x \cdot \frac{h}{l} - a_y \cdot \frac{h}{s_h} \right) \\ F_{HR} &= \frac{m}{2} \cdot \left(g \cdot \frac{l_v}{l} + a_x \cdot \frac{h}{l} + a_y \cdot \frac{h}{s_h} \right) \end{aligned} \quad (15)-(18)$$

Nachfolgend erfolgt eine Verifikation der Fahrdynamikregelung in der Simulation mit einem Gesamtfahrzeugmodell.

3 Ergebnisse

Das entworfene Reglerkonzept wurde in MATLAB/Simulink¹ implementiert und deren Funktionsweise mittels eines MKS-Gesamtfahrzeugmodells verifiziert. Das Gesamtfahrzeugmodell bildet die Dynamik des Aufbaus und der vier Räder ab. Es besitzt 16 Freiheitsgrade. Als Reifenmodell wird ein vereinfachtes MF-Tyre [5] verwendet. Damit werden die Wechselwirkungen im Kamm'schen Kreis und der degressive Radlasteinfluss berücksichtigt. Weiterhin ist das Einlaufverhalten der Reifen als PT1-Element nachgebildet.

In der Simulation verfügt das passive Fahrzeug über Frontantrieb. Beim aktiven Fahrzeug wird die Geschwindigkeit über die Vorderräder eingeregelt. Die zusätzlichen Antriebskräfte aus der Fahrdynamikregelung werden entsprechend der Radlastverteilung auf alle vier Räder aufgebracht. Die Aktordynamik der Radantriebe und der Lenkung ist nachgebildet, jedoch erfolgt keine Stellgrößenbegrenzung.

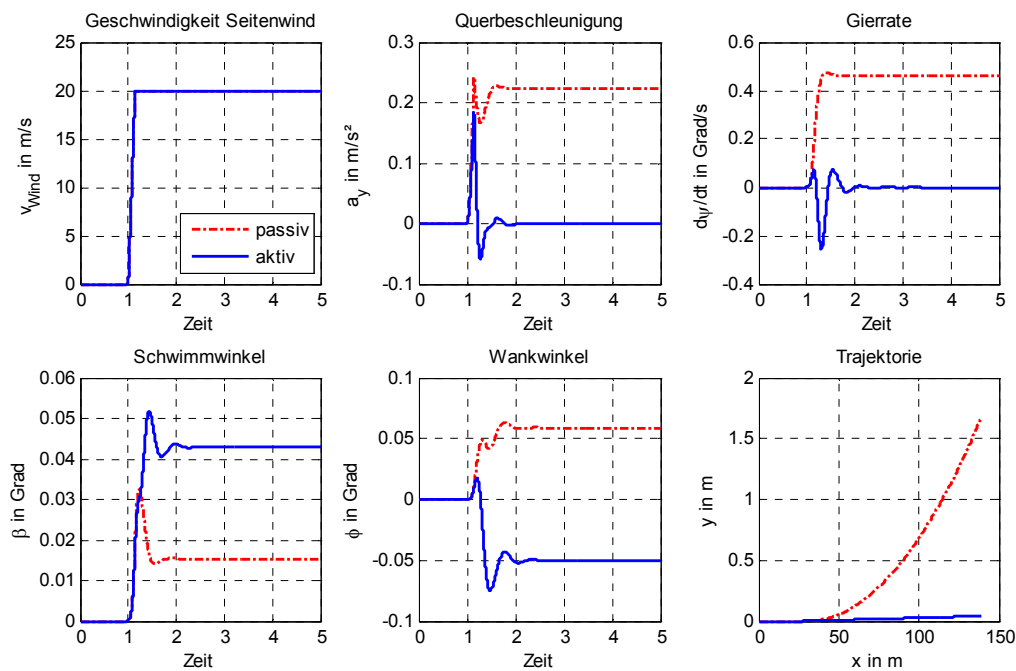


Abbildung 6: Verhalten bei Seitenwind

Eine Gegenüberstellung von passiven und aktiven Verhalten unter dem störenden Einfluss von Seitenwind zeigt Abbildung 6. Hierbei wurde das Befahren einer Strecke mit Seitenwind bei einer Fahrgeschwindigkeit von $v_x = 100$ km/h und einer Seitenwindgeschwindigkeit von $v_{Wind} = 20$ m/s nachgebildet. Nach einer Zeit von 1 sec befährt das Fahrzeug den Bereich mit

¹ MATLAB/Simulink ist ein Software-Produkt von The MathWorks

Seitenwind, die resultierenden Windkräfte entstehen über die Fahrzeuglänge hinweg bis das gesamte Fahrzeug stationär angeströmt wird. Als Wirkung stellen sich eine störende Querbeschleunigung und Gierrate ein, die das Fahrzeug vom Sollkurs abweichen lassen und beim passiven Verhalten einen Eingriff des Fahrers erfordern. Im aktiven Betrieb werden die unerwünschte Querbeschleunigung und Gierrate kompensiert. Die Querabweichung fällt sehr gering aus, so dass diese vom Fahrer nahezu nicht bemerkt wird. Damit wird der Fahrkomfort spürbar gesteigert.

Als weiteres Simulationsergebnis wird ein Lenkwinkelsprung bei einer Fahrgeschwindigkeit von $v_x = 100$ km/h im linearen Bereich der Querdynamik durchgeführt (Abbildung 7). Am Trajektorienverlauf ist ganz deutlich das gewünschte neutrale Fahrverhalten zu erkennen. Der Krümmungskreis für neutrales Fahrverhalten ist in roter Farbe gestrichelt eingezeichnet. Mit dem aktiven System wird das neutrale Fahrverhalten besser erreicht als mit dem passiven Fahrzeug. Im Übergangsverhalten weist das aktive System einen steileren Anstieg auf, jedoch führt dies auch zu einem stärkeren Überschwingen. Der negative Endwert des Giermoments bewirkt bei Linkskurven ein Eindrehen des Fahrzeugs.

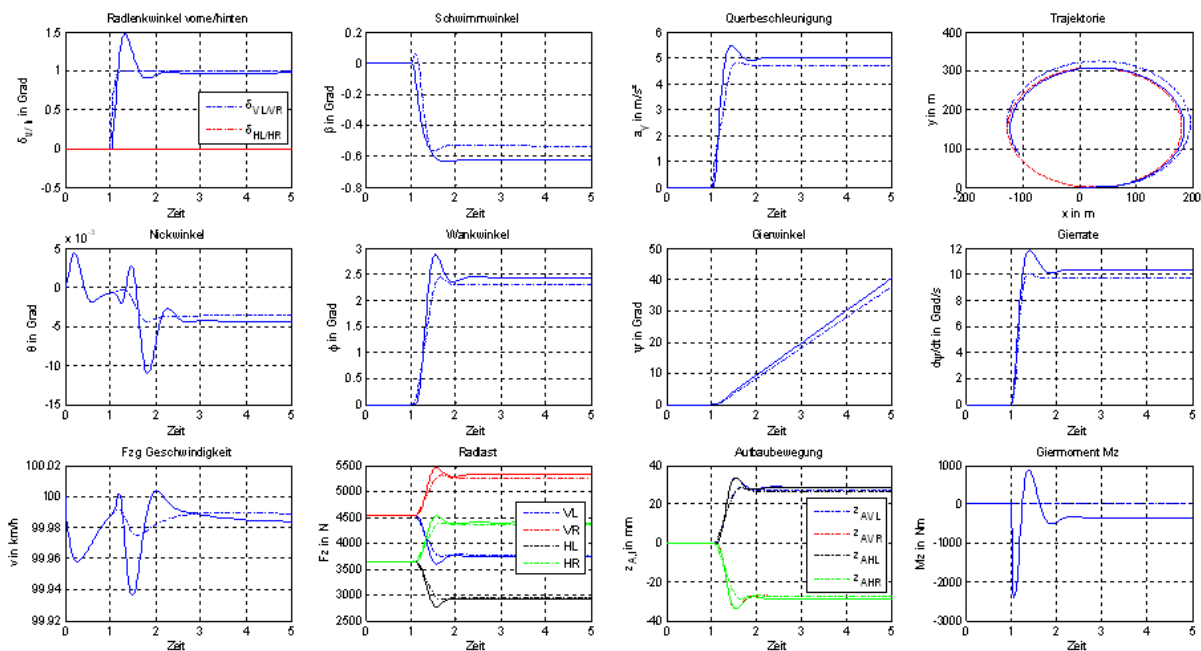


Abbildung 7: Sprungantwort im linearen Bereich der Querdynamik

Neben einer Sprunganregung im linearen Bereich der Querdynamik erfolgt auch eine Simulation im nichtlinearen Bereich (Abbildung 8). Dabei wird bei einer Fahrgeschwindigkeit von $v_x = 100$ km/h im passiven Betrieb ein Radlenkwinkel von $2,5^\circ$ sprungartig aufgeprägt. Am Trajektorienverlauf ist ganz deutlich das untersteuernde Fahrverhalten des passiven Fahrzeugs erkennbar. Das aktive Fahrzeug befindet sich in einem grenzstabilen Zustand, bricht aber nicht aus. Anhand des schwingenden Verlaufs des

Schwimmwinkels und der Fahrgeschwindigkeit treten die Wechselwirkungen der Längs- und Querkräfte der Reifen in Erscheinung. Mit der derzeitigen Fahrdynamikregelung wird kein stabiler Arbeitspunkt an der physikalischen Grenze eingestellt. Problematisch ist, dass im fahrdynamischen Grenzbereich der Teilregelung der Querbewegung eine zu hohe Referenzgröße übergeben wird. Diese kann durch die Sättigung der Reifen nicht eingestellt werden. Hinzu kommt der Einfluss des Geschwindigkeitsreglers, der konsequent durch Erhöhung des Längsschlupfs konsequent versucht die Geschwindigkeit zu halten. Mit zukünftig vorgesehenen Sicherheitsmechanismen sollen die Effekte der Wechselwirkungen aus Längs- und Querdynamik in der unterlagerten Giermomentaufteilung berücksichtigt werden.

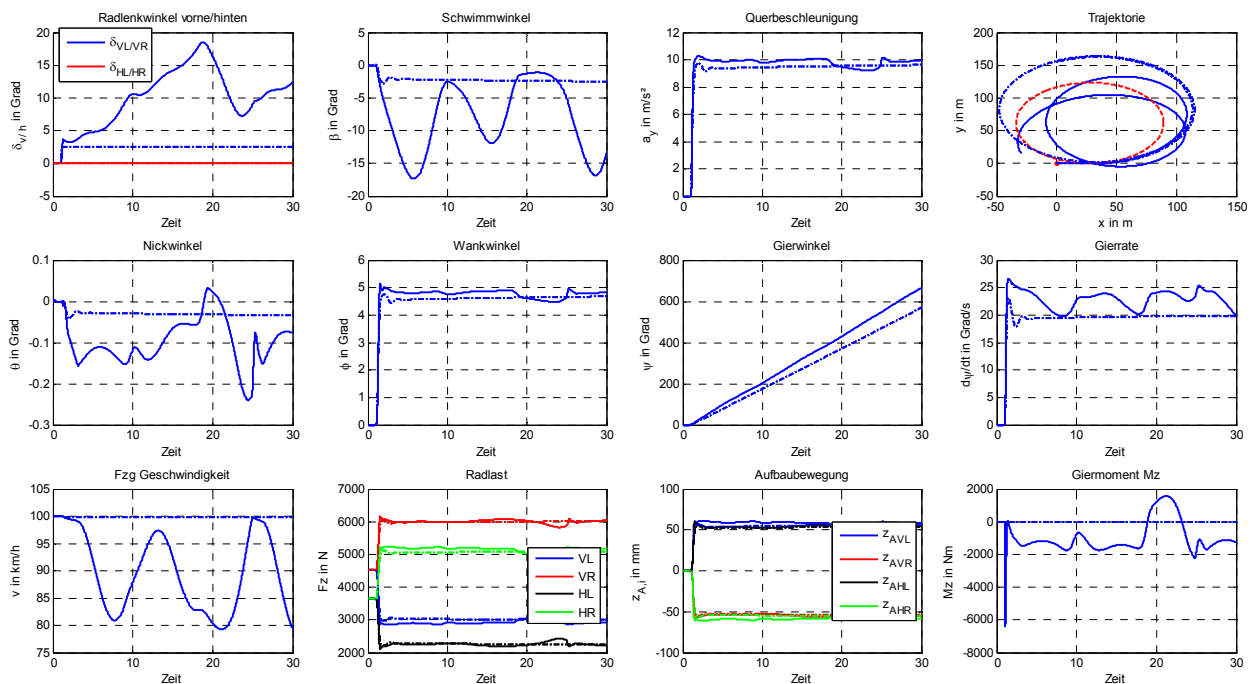


Abbildung 8: Sprungantwort im nichtlinearen Bereich der Querdynamik

4 Zusammenfassung

In diesem Beitrag wurde für eine hierarchisch strukturierte Fahrdynamikregelung ein Regelungskonzept, dessen Kern eine Entkopplung der Quer- und Gierbewegung ist, aufgezeigt. Zur Einstellung eines gewünschten neutralen Fahrverhaltens wird ein Giermoment über zusätzliche Reifenlängskräfte, die proportional zur Radlast auf alle vier Räder verteilt werden, eingestellt. An einem MKS-Gesamtfahrzeugmodell erfolgt eine Verifikation des Konzepts in der Simulation. Dabei wird sowohl ein störender Einfluss wie Seitenwind als auch das lineare und nichtlineare Verhalten der Querdynamik betrachtet.

Literatur

- [1] Ackermann, J.: Robust Control, VDI-Springer, 2002
- [2] Graf, M.; Wiesbeck, F.; Lienkamp, M.: Fahrdynamikauslegung des Elektrofahrzeugs
Mute. ATZ 06/2011
- [3] Liu-Henke, X; Buchta, R; Quantmeyer, F: Simulation eines mechatronischen
Lenkungsmoduls für ein Elektrofahrzeug mit dezentralen Direktantrieben. Workshop
ASIM/GI-Fachgruppen STS und GMMS, Krefeld, 24./25.02.11
- [4] Mitschke, M.; Wallentowitz, H.: Dynamik der Kraftfahrzeuge, VDI-Springer, 2003
- [5] Orend, R.: Integrierte Fahrdynamikregelung mit Einzelradaktorik. Dissertation,
Erlangen 2006



HEUTE SCHON AN MORGEN GEDACHT?

An unserem Standort in Tappenbeck (bei Wolfsburg) bieten wir Ihnen (m/w) folgende Einstiegsmöglichkeiten:

Praktikum, Abschlussarbeit oder Berufseinstieg

Sie wollen alles werden? Ingenieur in der Berechnung? Führungskraft für vielfältige Entwicklungsaufgaben? Lead Engineer bei führenden Lösungen? Sie können! Setzen Sie bei uns in einem Praktikum um, was Sie bisher in Ihrem Studium gelernt haben, fertigen Sie Ihre Abschlussarbeit an oder ergreifen Sie die Chance zu einem erfolgreichen Berufsstart. Bei uns nehmen Sie konkrete Projekte selbständig in Angriff, profitieren von der Erfahrung gestandener Profis und nutzen am eigenen Arbeitsplatz modernstes Equipment.

Bei Bertrandt hat Ihre Zukunft viele Chancen. Hier arbeiten Sie für alle, die die mobile Welt bewegen. Sie erwartet ein internationales Engineering-Unternehmen, das Partner der Zukunft ist. Und ein weiteres Mal in Folge ausgezeichnet wurde: als Top-Arbeitgeber 2011.

Was wollen Sie bewegen?

» Eike Fromhage +49 5366 9611-174, career-tappenbeck@de.bertrandt.com
Bertrandt Ing.-Büro GmbH, Krümke 1, 38479 Tappenbeck

Dynamische Simulationen in der Entwurfsphase mit Parameteranpassung durch implementierte Teillastrechnungen

Dipl.-Ing. Peter Tusche, Institut für Prozeßtechnik, Prozeßautomatisierung
und Meßtechnik (IPM), Hochschule Zittau/Görlitz

ptusche@hs-zigr.de

Prof. Dr.-Ing. habil. Tobias Zschunke, Hochschule Zittau/Görlitz

tzschunke@hs-zigr.de

Zusammenfassung

In der Kraftwerkstechnik stehen Forschungen bezüglich der Effizienzsteigerung und des Umweltschutzes durch den Einsatz neuer Verbrennungstechnologien und durch die Optimierung der Teillast- und Lastwechselfahrweisen sowie bezüglich der Verminderung der Klimaschädlichkeit von Kohlekraftwerken durch CO₂-Abtrennung auf der Tagesordnung.

Der Oxyfuel-Prozess, eine klimaschonende Verbrennungstechnologie von Braun- und Steinkohle, wird in naher Zukunft serienreif zum Einsatz in Großkraftwerken kommen. Wie sich der Dampferzeuger bei bestimmten dynamischen Fahrweisen und in untypischen Situationen verhält kann bis dahin allein mit Hilfe von dynamischen Simulationsrechnungen analysiert werden.

Die Modellierung des komplexen dynamischen Systems kann grundsätzlich auf zwei Wegen erfolgen. Der empirische Weg geht von der Analyse von Messwerten aus und kommt im vorliegenden Fall wegen fehlender Daten nicht in Frage. Der analytische Weg hingegen umfasst eine mathematische Beschreibung des Transport- und Speicherverhaltens auf der Basis von Konstruktions- und Materialdaten.

Besondere Herausforderung ist das Herstellen der Konsistenz der dynamischen Simulationen mit den normalerweise noch parallel laufenden Entwurfsberechnungen. In der Entwurfsphase werden für stationäre Voll- und Teillastvarianten die Energie- und Stoffstrombilanzen in entsprechend dafür entwickelten Softwaretools wie z.B. Epsilon [5] gerechnet. Für instationäre Berechnungen liegen zugeschnittene Tools wie z.B. DynStar [6] vor.

Die Kopplung dieser beiden Schritte unter Nutzung der jeweiligen Vorzüge ist der Grundgedanke der Arbeiten, die dieser Veröffentlichung zu Grunde liegen. Wesentliche Vorteile sind das Einsparpotential im Modellierungsaufwand und die Unabhängigkeit der Modelle von bestimmten Lastfällen und Arbeitspunkten.

Im Folgenden wird darauf eingegangen, wie die Nachberechnung in der Entwurfsphase und die dynamische Simulationsrechnung miteinander verknüpft werden.

Es wird nur noch einen Modellierungsschritt geben und die Parameter der dynamischen Übertragungsfunktionen werden an den aktuellen Arbeitspunkt angepasst. Somit stellen auch große Wertänderungen während der Simulation von dynamischen Prozessen kein Problem mehr dar.

Das Forschungsvorhaben wird mit Mitteln des Europäischen Sozialfonds und dem Freistaat Sachsen gefördert.



1 Einleitung

1.1 Dynamische Simulation

Die Bedeutung von dynamischen Simulationen in Wissenschaft und Technik ist in den letzten Jahren enorm gestiegen. Die Analyse von instationären Vorgängen bietet meist die Grundlage für optimierende Eingriffe durch sollwertgeführte Fahrweisen oder durch Prozessregelungsverfahren. In vielen technischen Systemen sind mehrere dynamische Teilprozesse verknüpft oder so überlagert, dass es unumgänglich wird, dynamische Simulationstools für die Analyse des Gesamtverhaltens der Anlage einzusetzen.

Außerdem können auftretende Störfälle oder Lastwechselfahrweisen am Computermodell einfach und kostengünstig nachgefahren werden und man verhindert dadurch die mögliche Zerstörung bestimmter Anlagenteile durch unzulässige Betriebsparameter. Die enormen Kosten, die durch einen Totalausfall von Großanlagen wie z.B. Kraftwerken entstehen können, werden dadurch minimiert oder sogar verhindert.

An Simulationsmodellen werden auch Tests durchgeführt, die mit der Optimierung des Prozesses im Zusammenhang stehen, oder mit denen bestimmte Fahrweisen nachgebildet werden, die sollwertgeführt ablaufen sollen. Auch bei der Teillastfahrweise oder bei der Lastwechselfahrweise werden unter Anwendung von Simulationsmodellen nach effizienzsteigernden Lösungen gesucht und mithilfe regelungstechnischer Eingriffe werden diese Lastfälle optimiert.

Auch im Gebiet der Auslegung von Kraftwerken steigt der Einsatz der Simulationstechnik stark an. Oft wird eine Kostenminimierung bei der Inbetriebnahme durch vorherige Analysen des dynamischen Verhaltens der Prozesse erzielt. Bei großen Investitionskosten wie z.B. in der Kraftwerkstechnik wird der Einsatz von dynamischen Simulationen im Vorfeld unabdingbar für die Unternehmen.

1.2 Oxyfuel-Prozess

Als innovative Verbrennungstechnologie ist der Oxyfuel-Prozess in die Verbrennungsprozesse von Braun- und Steinkohlen einzuordnen. Das klimaschädliche Kohlendioxid CO₂, welches bei der Verbrennung mit reinem Sauerstoff und destillativer Abtrennung von Wasser aus dem Abgas in hoher Konzentration anfällt, wird aufbereitet und verdichtet und soll schließlich unter Tage gespeichert werden. Diese Technologie wird in Forschungsanlagen und unter Leitung von führenden deutschen Energieerzeugerunternehmen an der CCS-Pilotanlage (Carbon Capture and Storage) an einem Kraftwerksstandort in der

Nähe des Ortes „Schwarze Pumpe“ untersucht, weiterentwickelt und optimiert. Der Oxyfuel-Prozess verläuft dem in Abbildung 1 dargestellten und im Folgenden kurz beschriebenen Schema:

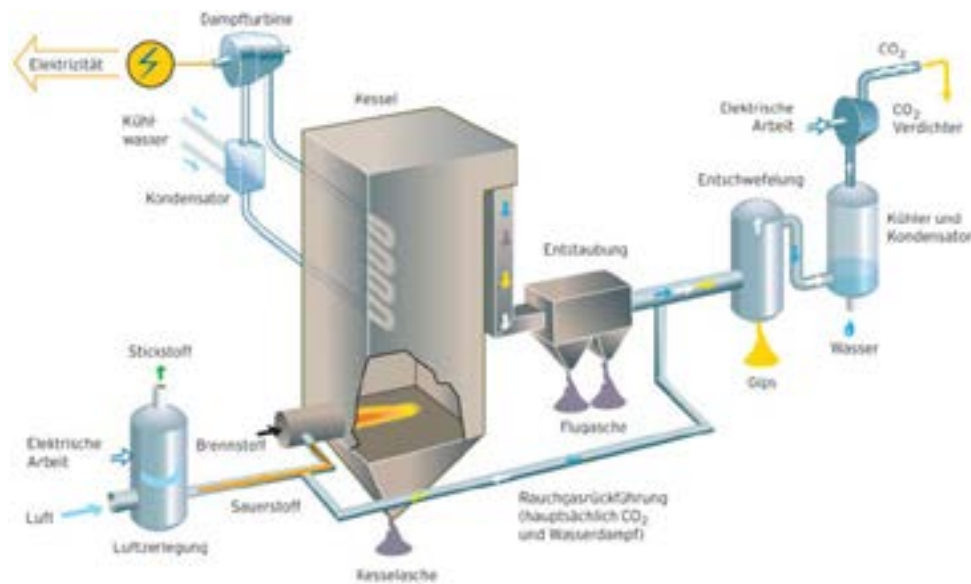


Abbildung 1: Schema Oxyfuel-Prozess [4]

In der Luftzerlegungsanlage wird reiner Sauerstoff aus der angesaugten Frischluft abgeschieden und dem Brennstoff als Oxidationsmittel zugeführt. Der dazwischengeschaltete Speicher für den Sauerstoff dient der Minderung von dynamischen Effekten bei der Bereitstellung des Oxidanten. Wird der vorgetrockneten Kohle nur Sauerstoff zugeführt, übersteigt die Verbrennungstemperatur die zulässigen Werte für die Wärmeübertragerheizflächen. Durch die Rezirkulation von staub- und aschefreien Rauchgas sinkt die Verbrennungstemperatur auf dampferzeugerzulässige Parameter. Der fehlende inerte Stickstoff im Oxidationsmittel wird durch die Rezirkulation von Rauchgas ersetzt, so dass sich der Massenstrom des Verbrennungsgases im Vergleich zur konventionellen Verbrennung mit Luft nur unwesentlich ändert. Entscheidend sind hier die Zusammensetzung und die damit verbunden Stoffwerte und Eigenschaften des Rauchgases für die Übertragungseigenschaften im Dampferzeuger. Nach der Staubabscheidung wird das Rauchgas teilweise rezirkuliert. Das noch vorhandene Abgas, bestehend aus einem Großteil an Kohlendioxid wird zum Zwecke der Wasserabscheidung gekühlt und schließlich verdichtet und unter der Erde gespeichert.

2 Motivation

2.1 Dynamik des Oxyfuel-Prozess

Insbesondere wird dabei die Lastwechselfahrweise betrachtet und das Umschaltprocedere von dem Anfahrvorgang mit Luft auf die Verbrennung mit reinem Sauerstoff.

Beim Umschaltvorgang wird der inerte Stickstoff, der sich bei der anfänglichen Luftverbrennung mit einer hohen Konzentration noch im Rauchgas und somit auch im Rezirkulationsgas befindet, nach und nach durch Kohlendioxid ersetzt. Dieser Vorgang hat durch die Rezirkulation ein dynamisches Verhalten und bewirkt zusätzlich durch die Veränderung in der Gaszusammensetzung auch Änderung in den Stoffparametern. Die

veränderte Gaszusammensetzung bedingt dadurch auch ein verändertes Verhalten in der Wärmeübertragung durch andere thermophysikalische Transporteigenschaften. Die Untersuchungen an Simulationsmodellen ist die kostengünstigste Variante, mögliche Probleme beim Verbrennungsprozess zu erkennen und vor dem Bau des Großkraftwerkes zu beheben.

2.2 Dynamik des Dampferzeugers

Die instationären Vorgänge im Dampferzeuger bestimmen auch entscheidend den Verbrennungsprozess. Die Feuerungskinetik, strömungstechnische Vorgänge, Speichervorgänge, kompressible Medien und die Stoffmischung sind einige Prozesse, die die Dynamik des Dampferzeugers beschreiben. Meist treten sie nicht einzeln auf, sondern überlagern sich bzw. sind miteinander gekoppelt. Durch die Rezirkulation, dem Einsatz einer Aufbereitung vor dem Dampferzeuger, sowie der Reinigung, Verflüssigung und Kompression am Ende des Abgasweges sind zusätzliche Komponenten in dem Prozess integriert worden, die dem dynamischen Verhalten weitere Faktoren geben. Die Untersuchung durch eine Analyse des veränderten Systemverhaltens zur konventionellen Verbrennung mit Luft werden die Simulationsmodelle liefern.

2.3 Auslegung und Simulation

Bedingt durch den Einsatz einer neuen Technologie und den Bau des Prototyps gibt es keine Messdaten für die Parametrierung von dynamischen Modellen für das Oxyfuel-Kraftwerk. Durch Berechnungen mit konstruktiven Daten in der Auslegungsphase durch Softwaretools wie z.B. Epsilon erhält man den stationären Arbeitspunkt. Für mehrere Lastfallsituationen erhält man ein Kennfeld von Übertragungsparametern für die einzelnen Komponenten. Mit diesen Kennfeldern werden die Simulationsmodelle parametrieren und erst dann ist die Analyse des dynamischen Systemverhaltens realisierbar. Diese komplizierte Herangehensweise kann durch die Implementierung des Zeitverhaltens der Komponenten in nur einem Schritt erfolgen. Im nachfolgenden Beispiel aus der Energietechnik wird erläutert, wie das Modell aufgebaut wird.

3 Methodik

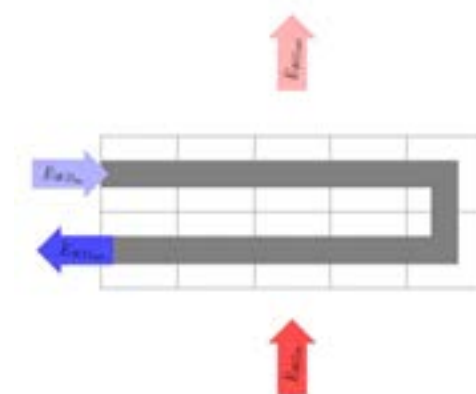
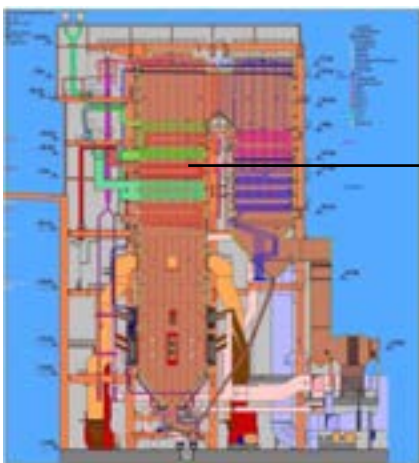


Abbildung 2: Schema „Rohrbündelwärmeübertrager“

Abbildung 3: Modell „Dampferzeuger“ [Quelle: Vattenfall Europe Generation AG]

Am Beispiel eines Rohrbündelwärmeübertragers eingesetzt als Überhitzer, Zwischenüberhitzer oder Economiser wird die Methode der Kopplung der dynamischen Modellierung mit der Implementierung des Nachberechnungsschrittes durch das Bekanntsein von konstruktiven Daten und Inputparametern dargestellt.

3.1 Modularer Aufbau

Ein modularer Aufbau der Berechnungsschritte ermöglicht einen leichten Austausch zwischen verschiedenen aber ähnlich aufgebauten Komponenten bzw. denen, die ähnliches Übertragungsverhalten besitzen. Durch die Unterteilung in verschiedene Übertragungsmechanismen wird dies realisiert.

- Reibungsbehaftete Strömung der Fluide in einem Strömungskanal mit Wärmeübergang an eine feste Wand (Modell: „Fluid“)
 - a) Bilanzgleichungen für Masse(2), Energie (1,4) und Stoffstrom (3)
 - b) Wärmeübergang (5) durch Konvektion und Strahlung
- Instationäre Wärmeleitung durch die Wand (Modell: „Wand“)
 - a) Bilanzgleichungen für den Energiestrom (6,7)

Diese zwei Teilmodelle werden zu einem Wärmedurchgangsmodell verknüpft, welches zwei Modelle „Fluid“ mit der Wärmequelle und der Wärmesenke beinhaltet, die durch das Modell „Wand“ über die beiden Wandtemperaturen gekoppelt sind (Abbildung 6).

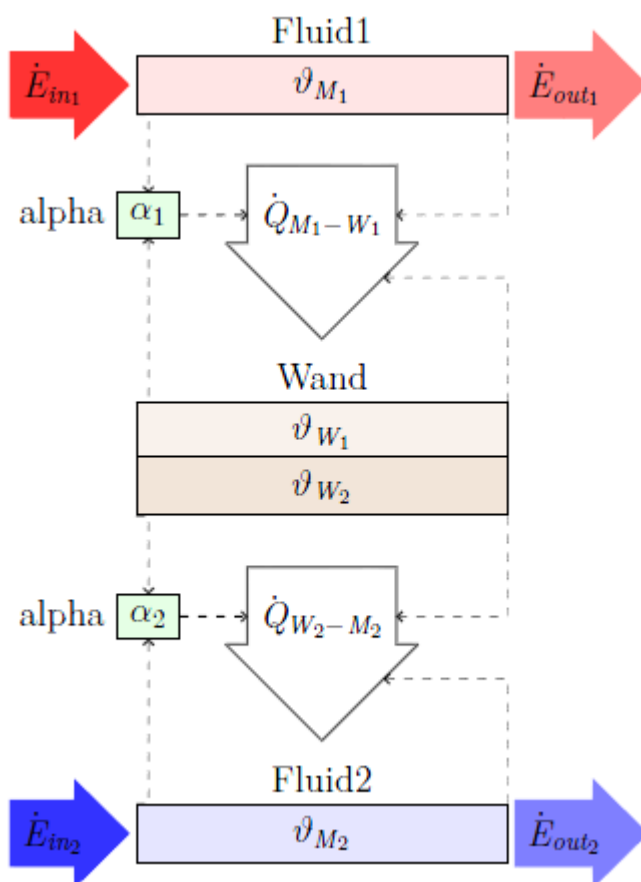


Abbildung 6: Modell „Wärmedurchgang“

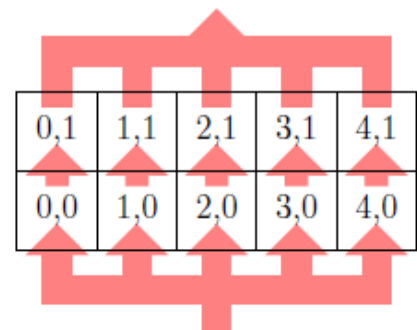


Abbildung 4: Rauchgasströmung HDÜ

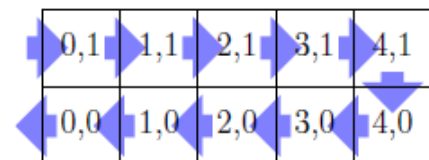


Abbildung 5: Dampfströmung HDÜ

Eine dem Aufbau und den Strömungsrichtungen entsprechende Verschaltung mehrerer dieser Wärmedurchgangsmodelle bildet den Überhitzer nach. Die Verschaltung erfolgt nach dem in Abbildung 4 und 5 gezeigten Matrixaufbau, in dem die Teilabschnitte mit Nummern zugeordnet sind.

3.2 Berechnungsgleichungen

Die Wahl der impliziten Berechnungsmethode des instationären Prozesses erlaubt eine iterative Anpassung der Übertragungsfaktoren an die aktuellen Bedingungen. Als Beispiel für die Übertragungsparameter sind hier die Druckverlustbeiwerte (8) für die reibungsbehaftete Strömung sowie die Wärmeübergangskoeffizienten (9) für die Wärmeübertragung auf die Wand zu nennen, die temperatur- und stoffabhängig sind, sowie durch Strömungsparameter beeinflusst werden. Die Berechnungsvorschriften aus dem VDI-Wärmeatlas [1] finden hier Verwendung. Die Energie-, Massen-, Stoffbilanzen sowie die Druckverlustberechnung für strömende Fluide sind implementiert. Die Wärmeübertragung, beschrieben durch den Wärmeübergang, die Wärmeleitung und die Wärmespeicherung sind modelliert.

$$\frac{V}{v} \cdot \frac{dh}{dt} + V \cdot \frac{dp}{dt} + h \cdot \frac{dm}{dt} = P + \dot{Q} + \dot{m}_{in} \cdot \left(h_{in} + \frac{1}{2} \cdot \left[\frac{\dot{m}_{in} \cdot v_{in}}{A_{in}} \right]^2 + g \cdot z_{in} \right) - \dot{m}_{out} \cdot \left(h_{out} + \frac{1}{2} \cdot \left[\frac{\dot{m}_{out} \cdot v_{out}}{A_{out}} \right]^2 + g \cdot z_{out} \right) \quad (1)$$

$$-\frac{V}{v^2} \cdot \frac{dv}{dt} = \frac{dm}{dt} = \dot{m}_{in} - \dot{m}_{out} \quad (2)$$

$$\frac{V}{v} \cdot \frac{d\xi}{dt} + \xi \cdot \frac{dm}{dt} = \dot{m}_{in} \cdot \xi_{in} - \dot{m}_{out} \cdot \xi_{out} \quad (3)$$

$$p_{in} - p_{out} = g \cdot \left(\frac{z_{in}}{v_{in}} - \frac{z_{out}}{v_{out}} \right) + \frac{1}{2} \cdot \left(\left[\frac{\dot{m}_{in}}{A_{in}} \right]^2 \cdot v_{in} - \left[\frac{\dot{m}_{out}}{A_{out}} \right]^2 \cdot v_{out} \right) + \zeta \cdot \frac{L}{d} \cdot \frac{\dot{m}^2 \cdot v}{2 \cdot A^2} \quad (4)$$

$$\dot{Q} = \alpha \cdot A \cdot (\vartheta_F - \vartheta_W) \quad (5)$$

$$\dot{Q} = \frac{\lambda}{s} \cdot A \cdot (\vartheta_{W1} - \vartheta_{W2}) \quad (6)$$

$$\frac{dQ}{dt} = m \cdot c \cdot \frac{d\vartheta_W}{dt} \quad (7)$$

$$\zeta = f(\dot{m}, \vartheta_W, p, h, d, L) \quad (8)$$

$$\alpha = f(\dot{m}, \vartheta_W, p, h, d, L) \quad (9)$$

Unter Nutzung der Stoffwertbibliotheken für ideale Gase [2] und für Wasser bzw. Wasserdampf [3] werden die Stoffparameter Temperatur (11) und spezifisches Volumen (10) für die Übertragungsfunktionen berechnet.

$$v = f(p, h, \xi) \quad (10)$$

$$\vartheta_M = f(p, h, \xi) \quad (11)$$

Mit der Vielzahl der Verknüpfungen, der Abhängigkeiten, der Nichtlinearitäten entsteht ein nichtlineares Differentialgleichungssystem welches durch Iterationsverfahren implizit gelöst wird. Dabei werden die nichtlinearen Parameter in jedem Berechnungsschritt an die aktuellen Bedingungen angepasst.

3.3 Örtliche Diskretisierung

Die örtliche Unterteilung des Rohres für die instationäre Wärmeleitung wird nach dem Schema einer energetisch gewichteten radialen Diskretisierung vorgenommen.

$$\frac{d\vartheta}{dt} = (1 - x) \cdot \frac{d\vartheta_{in}}{dt} + x \cdot \frac{d\vartheta_{out}}{dt} \quad (12)$$

$$x = \frac{\ln\left(1 + \frac{s}{2 \cdot r_{in}}\right)}{\ln\left(1 + \frac{s}{r_{in}}\right)} \quad (13)$$

Die örtliche Unterteilung des Strömungskanales erfolgt nach dem Schema einer Diskretisierung für ein durchströmtes offenes thermodynamisches System für den Druck (14), die Enthalpie (15) und die massebezogene Stoffzusammensetzung (16).

$$p = \frac{p_{in} \cdot A_{in} + p_{out} \cdot A_{out}}{A_{in} + A_{out}} \quad (14)$$

$$h = \frac{h_{in} \cdot A_{in} \cdot v_{out} + h_{out} \cdot A_{out} \cdot v_{in}}{A_{in} \cdot v_{out} + A_{out} \cdot v_{in}} \quad (15)$$

$$\xi = \frac{\xi_{in} \cdot A_{in} \cdot v_{out} + h_{out} \cdot A_{out} \cdot \xi_{in}}{A_{in} \cdot v_{out} + A_{out} \cdot v_{in}} \quad (16)$$

4 Ergebnisse

Der oben modellhaft dargestellte doppelbahnige Gegenstrom-Hochdrucküberhitzer wird axial in 5 Unterteilungen je Bahn und radial in 3 Unterteilungen für den Rohrdurchmesser diskretisiert. Die Eingangsdaten für die beiden Fluidströme sind nachfolgend tabelliert. Das Rauchgas hat die Zusammensetzung (Massenanteile) $O_2=0,0385$ $N_2=0,6401$ $CO_2=0,1846$ $H_2O=0,1368$;

Größe	Einheit	Rauchgas	Wasserdampf
p	[bar]	1,01325	174,5
h	[kJ/kg]	1381	3250
ϑ_M	[°C]	842,5	491,6
$\dot{m}$	[kg/s]	415	226

Tabelle 1: Eingangsdaten für Simulation Überhitzer-Beispiel

In der Simulation wird das Öffnen des Einspritzkühlers vor dem Hochdrucküberhitzer simuliert. Das Abkühlen von Dampf mit flüssigem Wasser ($\dot{m}=5\text{kg/s}$; $\vartheta_M=250^\circ\text{C}$) setzt bei $t=100\text{s}$ schlagartig ein.

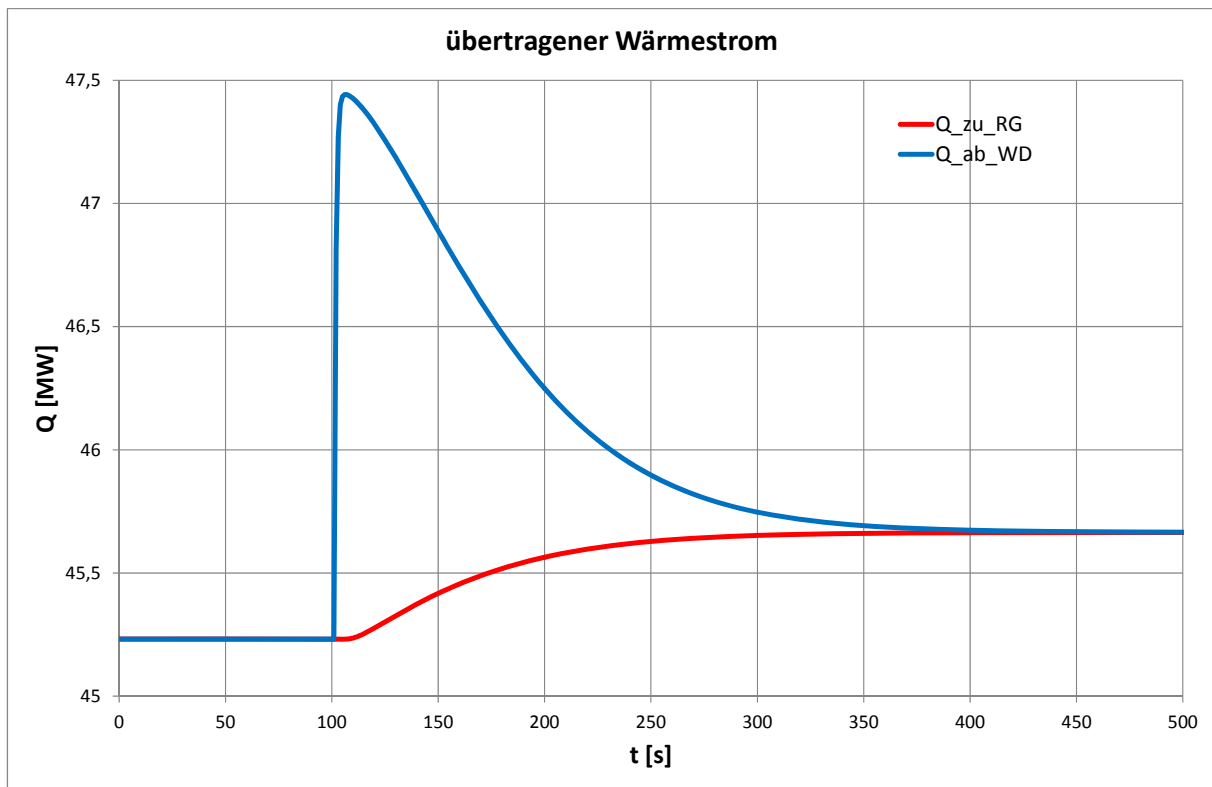


Abbildung 7: zeitlicher Verlauf der Wärmeströme des Hochdrucküberhitzers

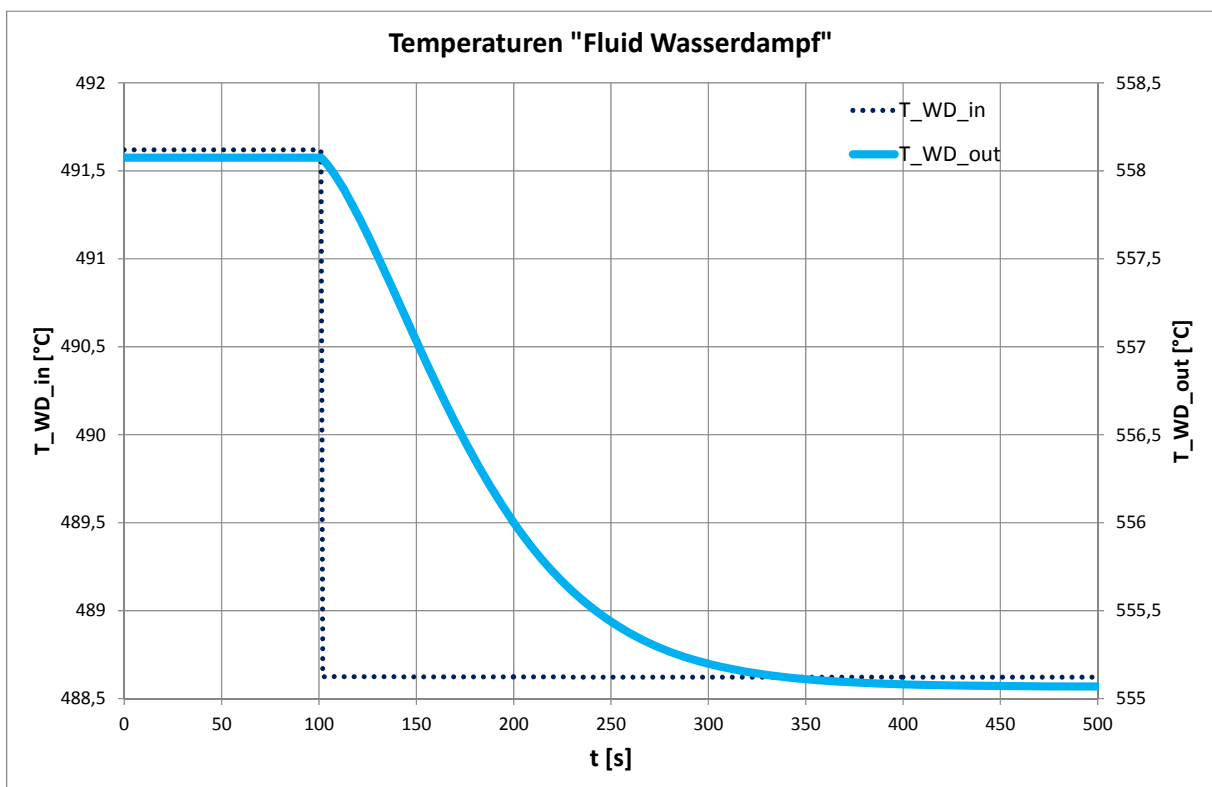


Abbildung 8: zeitlicher Verlauf der Fluidtemperaturen der Wärmesenke „Wasserdampf“

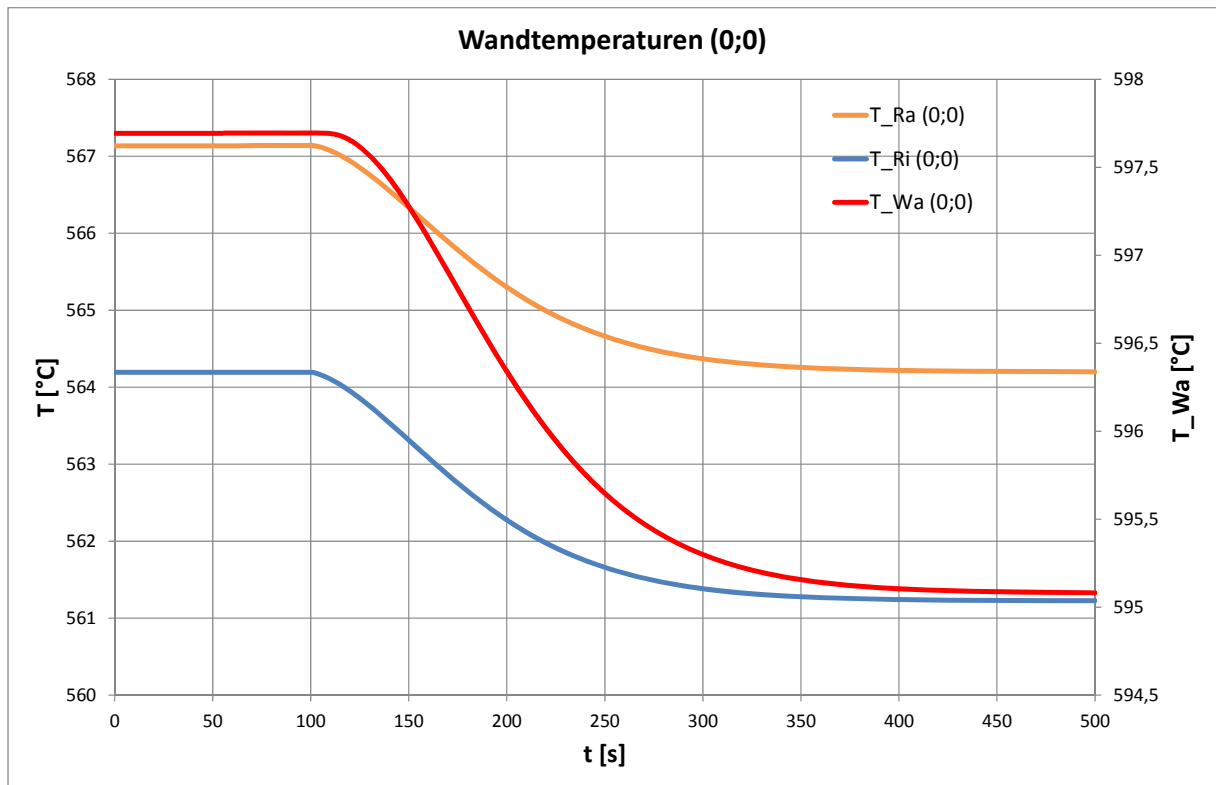


Abbildung 9: zeitlicher Verlauf der Wandtemperaturen des Elementes 0,0 im Matrixaufbau (siehe Abbildung 4,5)

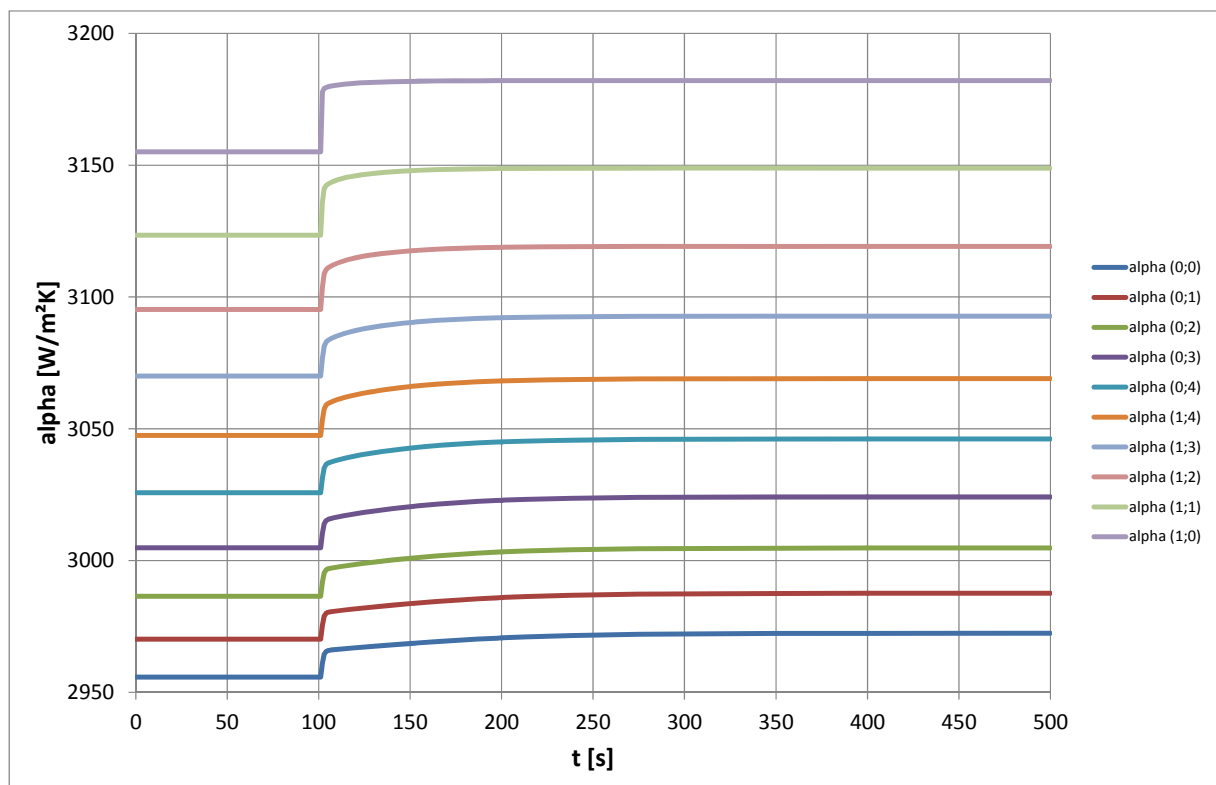


Abbildung 10: zeitlicher Verlauf der Wärmeübergangskoeffizienten auf der Wasserdampfseite

Das Systemverhalten auf diese sprunghafte Änderung am Eingang der Wärmesenke des Hochdrucküberhitzers in Bezug auf den übertragenen Wärmestrom (Abbildung 7) und auf das

dynamische Verhalten der Fluid- (Abbildung 8) und Wandtemperaturen (Abbildung 9) entlang des Überhitzers werden im weiteren Verlauf grafisch dargestellt und ausgewertet.

Der Übergangsprozess von dem Ausgangspunkt in den neuen stationären Arbeitspunkt ist visuell nach ca. 5 min abgeschlossen. In dieser Zeit kühlt die Rohrtemperatur innen T_{Ri} und außen T_{Ra} um fast 3 K ab (Abbildung 9). Die Energie wird vom Fluid der Wärmesenke „Wasserdampf“ abgeführt und ein verzögertes Abkühlen der Austrittstemperatur $T_{\text{WD_out}}$ ist die Folge. Die Abkühlung der Temperatur der äußeren Ablagerungsschicht (heiße Seite) T_{Wa} auf dem Rohr verzögert sich durch die instationäre Wärmeleitung. Der übertragene Wärmestrom auf den Wasserdampf während des Abkühlvorganges ist deutlich grösser als der Wärmestrom vom Rauchgas auf die Rohrwand (Abbildung 7).

5 Diskussion

Eine schnelle und stabile Simulation ist das Ergebnis dieser Modellierungsart. Dieser Diskretisierungsgrad ist für die Optimierung durch regelungstechnische Eingriffe ausreichend genau um das Übergangsverhalten klar darzustellen. Zusätzlich können die Ergebnisse der Wandtemperaturen und kritische Wärmestromdichten für bestimmte Ortspunkte in die Regelung oder die Sollwertgeführten Fahrweisen als Parameter mit einbezogen werden.

Der weitaus größere Vorteil dieser Modellierung besteht in der Aktualität der Übertragungsparameter. Die am Beispiel des Wärmeübergangskoeffizienten (Abbildung 10) aufgezeigten Änderungen durch Abweichungen vom Arbeitspunkt werden für jeden Berechnungsschritt an die aktuellen Bedingungen angepasst. Dies ermöglicht die Abdeckung der Simulation eines sehr großen Lastfallbereiches der Kraftwerkskomponente ohne den Fehler durch eine Interpolation der Punkte in einem Lastfall-Kennfeld.

6 Ausblick / Schlussfolgerung

Unter diesen Voraussetzungen kann man in einem weiteren Arbeitsschritt den Fall eines Kaltstarts eines Dampferzeugers simulieren. Des Weiteren können auch große Laständerungsanforderungen ohne Probleme simuliert und dargestellt werden. Wichtige Schritte für die kommenden Jahre, in denen Einsatz fluktuierender erneuerbarer Energien immer größer wird und die derzeit noch in Grundlast fahrenden Großkraftwerke mit Braun- und Steinkohle zur Spitzenlastfahrweise wechseln. Die Lastwechselfahrweise wird in den kommenden Jahren verstärkt eintreten und das Hauptaugenmerk für den Betrieb dieser Kraftwerke liegt auf der zunehmend dynamischen Verfügbarkeit und der Dynamik der Lastfallumschaltungen. Anhand von stabilen und schnellen Simulationen können diese Szenarien analysiert und optimiert werden, um insbesondere auch Materialüberlastungen vermeiden oder minimieren zu können.

Literatur

- [1] Verein Deutscher Ingenieure, VDI-Gesellschaft Verfahrenstechnik und Chemieingenieurwesen (GVC). VDI-Wärmeatlas 10. Auflage. Berlin Heidelberg : Springer-Verlag,

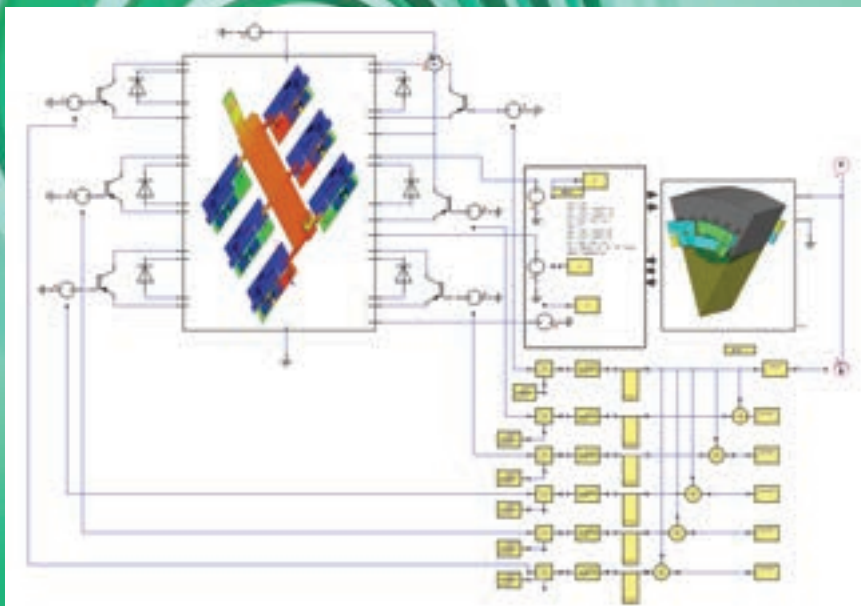
2006. ISBN-10 3-540-25504-4.

- [2] Fachgebiet Technische Thermodynamik. Stoffwert-Programmbibliothek LibIDGAS für Verbrennungsgasgemische, berechnet als ideales Gemisch der idealen Gase CO₂, CO, SO₂, H₂O, N₂, O₂, Ar, Ne, trockenen Luft und Luftstickstoff nach der VDI-Richtlinie 4670. Hochschule Zittau/Goerlitz, 2008.
- [3] Fachgebiet Technische Thermodynamik. Stoffwert-Programmbibliothek LibIF97 für Wasser und Wasserdampf, berechnet nach der internationalen Industrie-Formulation IAPWS-IF97 (Revised Release 2007) und den ergänzenden Standards IAPWS-IF97-S01, -S03rev, -S04 und -S05. Hochschule Zittau/Goerlitz, 2007.
- [4] http://www.ev-akademie-meissen.de/uploads/media/Vortrag_Wolfgang_Dierschauer.pdf (27. Januar 2012)
- [5] http://www.steag-systemtechnologies.com/ebsilon_professional.html (27. Januar 2012)
- [6] <http://www.hs-zigr.de/~livecms/cmsdocs/IPM/de/data/artikel/Software1/dynstar.html>
(27. Januar 2012)

Realize Your Product Promise

ANSYS Simplorer

Simplify multidomain simulation from detailed components to the system level, all within a single design environment.



ANSYS Simplorer® enables engineers to accurately and quickly design complex power electronic and electrically controlled systems. It is an intuitive, multidomain, multitechnology simulation program used to simulate, analyze, and optimize complex systems, including electromechanical, electromagnetic, power and other mechatronic designs. In industries such as automotive, aerospace and industrial automation, organizations use Simplorer to identify problems in the early design stages that other simulation or build-and-test methods cannot detect.

Softwaretool zur Standardisierung des Modellierungsprozesses permanenterregter Synchronmaschinen

Gerhard Edmound Stebner, Institut für Mechatronik

ge.stebner@ostfalia.de

Christoph Hartwig, Institut für Mechatronik

ch.hartwig@ostfalia.de

Zusammenfassung

Im Rahmen dieser Veröffentlichung wird die Software EaSync des Instituts für Mechatronik (IMEC) der Ostfalia Hochschule für angewandte Wissenschaften vorgestellt. Es handelt sich dabei um ein wachsendes Tool für Synchronmaschinen auf Basis der Finiten Elemente Methode (FEM) und analytischen Modellen. Das Ziel der Softwareentwicklung stellt eine Arbeitsumgebung dar, welche von der Maschinenauswahl, über die Geometrie-, Material- und Wicklungsauslegung alle wesentlichen Prozesstools zur Auslegung von permanenterregten Synchronmaschinen umfasst. EaSync basiert vollständig auf studentischen Arbeiten.

1 Motivation

Die Einsparung fossilen Brennstoffs und die Reduzierung der CO₂-Emissionen im Automobil fordert nicht nur die Entwicklung immer effektiverer Antriebskonzepte – längst werden Lenkunterstützungen und Wasserpumpen durch elektrische Antriebe realisiert, wobei größtenteils permanenterregte Synchronmaschinen aufgrund ihres hohen Wirkungsgrades zum Einsatz kommen. Die Anforderungen an Synchronmaschinen beschränken sich jedoch nicht nur auf hohe Wirkungsgrade und Leistungsdichten, sondern umfassen auch Grenzwerte für Drehmomentwelligkeiten. Diese meist ungewollten Komponenten des Drehmoments werden in Cogging und Ripple unterteilt. Erstere ist dabei die Komponente, die auch ohne Anlegen einer Spannung an der Antriebswelle vorhanden ist und aus der Wechselwirkung zwischen Permanentmagneten und Nutung resultiert. Bei niedrigen Drehzahlen fällt das Cogging auch bei angelegter Spannung besonders stark ins Gewicht und überlagert sich mit dem womöglich geringen Sollmoment. Der Drehmomenten Ripple bezeichnet dagegen die Welligkeit, die sich auf das Nutmoment überlagert und nur dann auftritt, wenn die Wicklungen der Maschine Strom führen. Die Amplitude und Frequenz der Ripple Komponenten ergeben sich aus den gewählten Eigenschaften der Magnete, der Wicklung, der Geometrie und der Stromstärke. Die Modellierung und Reduzierung der

Drehmomentwelligkeit steht im Fokus der Software EaSync des IMEC. Dabei steht die Bedienfreundlichkeit und Verständlichkeit der Software ebenfalls im Vordergrund, sodass neben der Eignung als Entwicklungstool auch Lehrinhalte mit ihr vertieft werden können.

2 Struktur und Funktionsweise

EaSync basiert auf der Kombination einfacher analytischer Modelle und der Finiten Elemente Methode (FEM). Die analytischen Modelle und das grafische Benutzer Interface (GUI) werden in MATLAB Simulink realisiert. Weiterhin wird zur Optimierung von Maschineneigenschaften die MATLAB Optimization Toolbox verwendet. Als FEM-Software wird COMSOL Multiphysics verwendet. Der Großteil des Preprocessing, der Simulationssteuerung und des Postprocessing basiert auf parametrisierten Modellen und Auswertungsalgorithmen, die aus dem MATLAB GUI heraus konfiguriert und gestartet werden. Dabei werden dem Benutzer automatische Modellreduzierungen wie beispielsweise durch Symmetrienausnutzung angeboten.

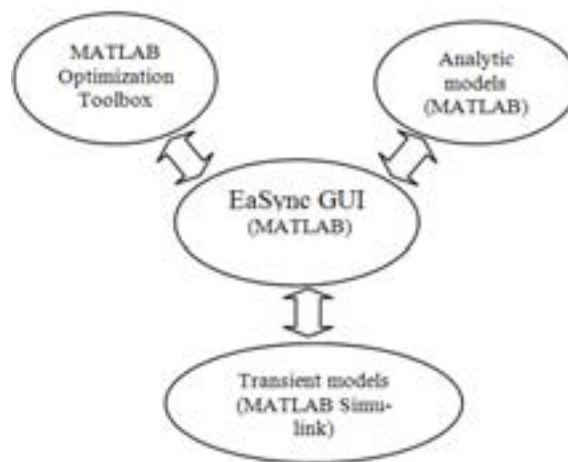


Abbildung 1: Struktur der Software EaSync

Die Symbiose aus analytischen Maschinenmodellen und FEM-Modellen führt zu einer hohen Flexibilität bei der Modellierungstiefe und Simulationsdauer. Hierauf wird in Kapitel 3.2 eingegangen, in welchem die ableitbaren Maschinenmodelle beleuchtet werden. Des Weiteren werden Modelle unabhängig von der Modellierungstiefe aus einer Quelle und einem Parametersatz erzeugt, was die Konsistenz der Modelle im Rahmen der Modellierungstiefe garantiert.

Die Verbindung zwischen MATLAB Simulink und COMSOL Multiphysics wird über die LiveLink Toolbox hergestellt. Diese Toolbox ermöglicht die Ausführung der Software in einer Konsole als Server, auf welchem MATLAB Simulink als Client zugreift. Der Funktionsumfang als Client ist Vergleich zu der COMSOL Oberfläche eingeschränkt. Diese Einschränkungen erstrecken sich primär auf den Bereich des Pre- und Postprocessing, da hierzu die Auswahl von Kanten, Flächen und Volumina nötig ist. Die Problematik liegt in der Identifizierung von Geometrieobjekten, die nicht anhand von Namen, sondern durch Nummerierung erfolgt. Diese Nummerierung erfolgt nicht systematisch und erschwert somit

die Manipulation von Geometrieobjekten, die Zuordnung von Materialdaten und physikalischen Randbedingung sowie die Auswertung lokaler Simulationsergebnisse. Ein erheblicher Teil der geleisteten Arbeit liegt somit in der Erweiterung der LiveLink Toolbox, um eine durchgängige Identifizierung und Manipulation von Objekten zu ermöglichen.

Bei der Verwendung des EaSync GUI definiert der Benutzer alle relevanten Maschinendaten wie Geometrie, Materialien, Wicklungsart sowie das Ziel der Simulation in Form der Modellierungstiefe und der relevanten Größen und Verläufe. Anschließend werden diese Informationen in Konsolenbefehle für die LiveLink Toolbox umgewandelt, welche auf den COMSOL Server weitergeleitet werden. Dabei werden die in Abbildung 2 dargestellten Schritte durchlaufen, welche typisch für eine FEM-Analyse sind. Hierbei ist der Eingriff des Nutzers in den meisten Fällen nicht nötig [3].

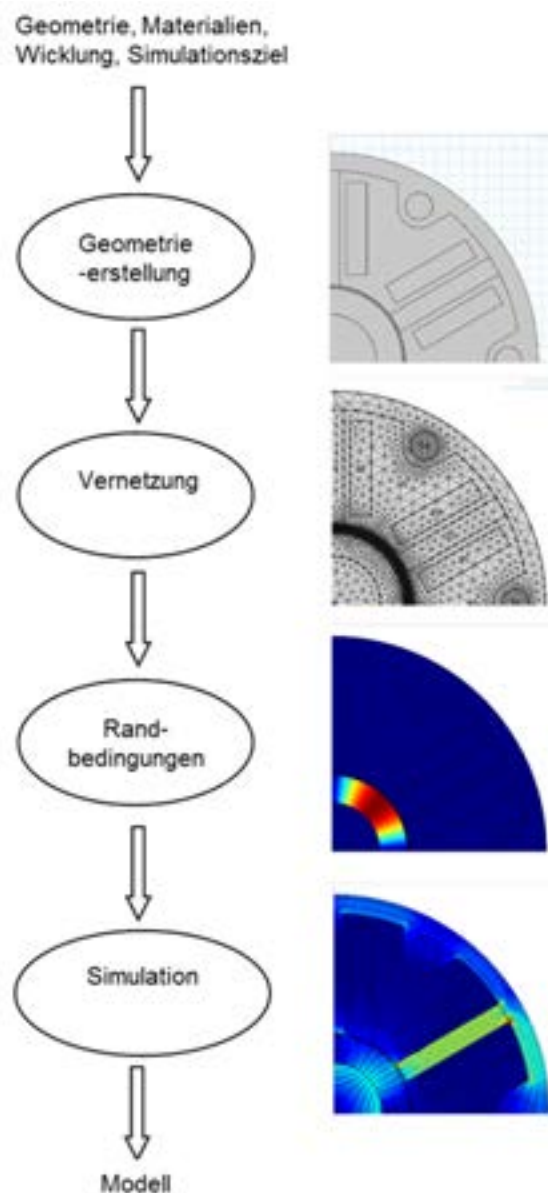


Abbildung 2: Modellgenerierung über
COMSOL LiveLink™

3 Funktionsumfang

3.1 Abbildbare Maschinentypen

Zum jetzigen Zeitpunkt unterstützt EaSync die Berechnung von permanenterregten Innenläufern mit aufgeklebten Magneten oder Zylindermaterial (siehe Kapitel 0). Dabei sind Stator und Rotor durch über 30 Parameter geometrisch anpassbar, wobei bisher nur Vollpolmaschinen abbildbar sind. Des Weiteren lässt sich die Form und die Magnetisierung annähernd beliebig gestalten. Dies erfolgt durch ein sehr feines Netz im Bereich des Luftspalts und einer funktionsbasierten und ortsabhängigen Eingabe der Magnet-eigenschaften. Die bisher vordefinierten Magnetformen erstrecken sich von einfachen rechteckigen oder trapezförmigen Magneten bis hin zu frei definierbaren Formen wie sie durch die Verwendung von Fourier-Reihen entstehen. Speziell durch letztere Möglichkeit ist es im Bereich der Lehre einfach, die Entstehung des Nutzmoments und der Pendelmomente zu erläutern, da das Läuferfeld entsprechend einfach manipuliert werden kann.

Im EaSync GUI besteht die Möglichkeit Ganzloch-, Bruchloch- und Zahnspulenwicklungen in ein- und zweischichtiger Ausführen zu definieren. Dies erspart die manuelle und fehleranfälligere Eingabe.

3.2 Ableitbare Maschinenmodelle

Die Modellierungstiefe in EaSync reicht von einfachen analytischen Maschinenmodellen, wie der Abbildung des Stromkreises der Synchronmaschine, bis hin zu CO-Simulationen in denen der Antrieb in ein Simulink-Modell eingebettet wird. Dazwischen liegen komplexere Ld/Lq-Modelle mit gemittelten Parametern oder aber Modelle mit detaillierten Kennfeldern, welche winkel- und stromabhängige Daten zur Längs- und Querinduktivität sowie zu Drehmomentwelligkeiten enthalten. Hierbei leiten sich alle Modelle aus einer zentralen Benutzeroberfläche und Parameterliste ab. Dies beschleunigt den Modellierungsprozess und maximiert die Konsistenz der Modelle untereinander. In Abbildung 3 sind die gebräuchlichsten Maschinenmodelle gegenübergestellt, diese Modelle lassen sich durch Feldorientierte Regelungen und Lasten ins MATLAB Simulink erweitern.

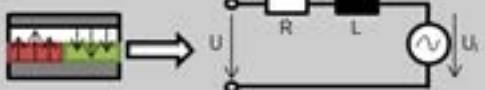
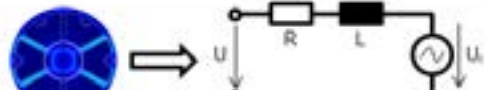
Modell	Dauer zur Erzeugung des Modells	Echtzeitfähigkeit
 Analytisches Modell aus analytischem Luftspaltmodell zur Modellierung eines Strangs	Wenige Minuten	1
 Analytisches Modell aus FEM-Daten zur Modellierung eines Strangs	$\Psi_u(i), L_d(const), L_q(const)$ → wenige Stunden $\Psi_u(i, \varphi, \beta), L_d(i, \varphi, \beta), L_q(i, \varphi, \beta)$ → mehrere Stunden / Tage	1
 CO-Simulation	Wenige Stunden (seit COMSOL Version 4.0 nicht mehr verfügbar)	0

Abbildung 3: Übersicht der ableitbaren Modelle aus EaSync

Außerdem kann für die Berechnung der Induktivität zwischen zwei Verfahren gewählt werden [1]. Beim ersten Verfahren wird aus Drehmoment-, Spannungs- und Stromverläufen auf die Längs- und Querinduktivität geschlossen. Hierzu müssen die folgenden Gleichungen gelöst werden:

$$u_d = \omega(\psi_M + L_d i_d) \quad (1)$$

$$u_q = -\omega L_q i_q \quad (2)$$

$$M = \frac{m}{2} p [(\psi_M + L_d i_d) i_q - L_q i_q i_d] \quad (3)$$

Dazu wird im Rahmen einer Leerlaufsimulation die Lage der Polradspannung festgelegt. Erst durch die Lage der Polradspannung können Ströme und Spannungen der D- und Q-Achse zugeordnet werden. Es folgt eine Simulation zur Bestimmung der Flussverkettung ψ_M sowie eine weitere abschließende Simulation zur Bestimmung des D- und Q-Anteils der Induktivität. Das Vorgehen ist in [1] detailliert beschrieben.

Die dafür nötige Aneinanderreihung von Simulationen und Auswertungen von Spannungs- und Stromverläufen geschieht ohne Zutun des Benutzers und aus dem EaSync GUI heraus. Im zweiten und vereinfachten Verfahren wird ein Spannungssprung auf zwei Klemmen der Maschine gegeben und somit das PT1-Verhalten des Strangs analysiert. Durch die Wicklungseigenschaften ist der Widerstand R bekannt und damit auch die Längsinduktivität L der Maschine (1),(2). Auch diese Schritte werden durch EaSync automatisiert ausgeführt.

$$i(t) = i_{\text{stat}} \left(1 - e^{-\frac{t-t_0}{\tau}} \right) \quad (4)$$

$$\tau = R \cdot L \quad (5)$$

Es zeigt sich somit ein weites Spektrum bezüglich der Modellierungstiefe und der Dauer zur Erzeugung des Modells. Es ist jedoch seit der COMSOL Version 4.0 nicht mehr möglich, eine CO-Simulation mit Simulink zu erzeugen, da der entsprechende Simulink-Block durch LiveLink nicht zur Verfügung gestellt wird. Es ist zu erwarten, dass diese Option durch künftige COMSOL Versionen wiederhergestellt wird.

Neben der Qualität des Drehmoments sind vor allem elektrische, magnetische und thermische Verluste wichtig für die Beurteilung einer Synchronmaschine. Diese Eigenschaften können bisher nicht oder nicht mit ausreichender Genauigkeit simuliert werden und sind Inhalt zukünftiger studentischer Arbeiten.

4 Anwendungsbeispiel anhand einer Parameterstudie

Zur Überprüfung der Leistungsfähigkeit der Software EaSync wurde ein existierendes Motorkonzept mit folgenden Eigenschaften herangezogen (Tabelle 1, Abbildung 4):

Tabelle 1: Daten des Probanden

Nennleistung	100.5W
Nennspannung	3~400V
Nenndrehzahl	1500U/min (50Hz)
Nennmoment	0.64Nm
Nennstrom	228mA
Wicklung	Zahnspulenwicklung
Polpaarzahl	2
Nutzahl	6



Abbildung 4: Proband zur Validierung der Modellierungsansätze

Bei dieser Maschine werden keine aufgeklebten Magnete verwendet. Das Magnetmaterial ist zylinderförmig und wird auf die Welle der Maschine geschoben (siehe Abbildung 4). Die Magnetisierungsbreite der Pole wurde durch den Hersteller experimentell durch Fertigung zahlreicher Rotoren optimiert. Dabei lag das Augenmerk auf der Reduzierung der Nutrastmomente. Mit Hilfe von EaSync und der Anbindung an die MATLAB Optimization Toolbox wurde dieser Prozess wiederholt. Dabei wurde ebenfalls die Leistungsfähigkeit zweier Methoden zur Modellreduzierung verglichen. Diese basieren zum einen auf Symmetrienausnutzung innerhalb der Geometrie und der Wicklung (Urwicklung) als auch auf dem Abwickeln der Maschine. Dabei wird die Maschine als Linearaktuator simuliert, wobei hoher Wert auf die Ähnlichkeit zum ursprünglichen Modell im Bereich des Luftspalts gelegt wird (siehe Abbildung 5).

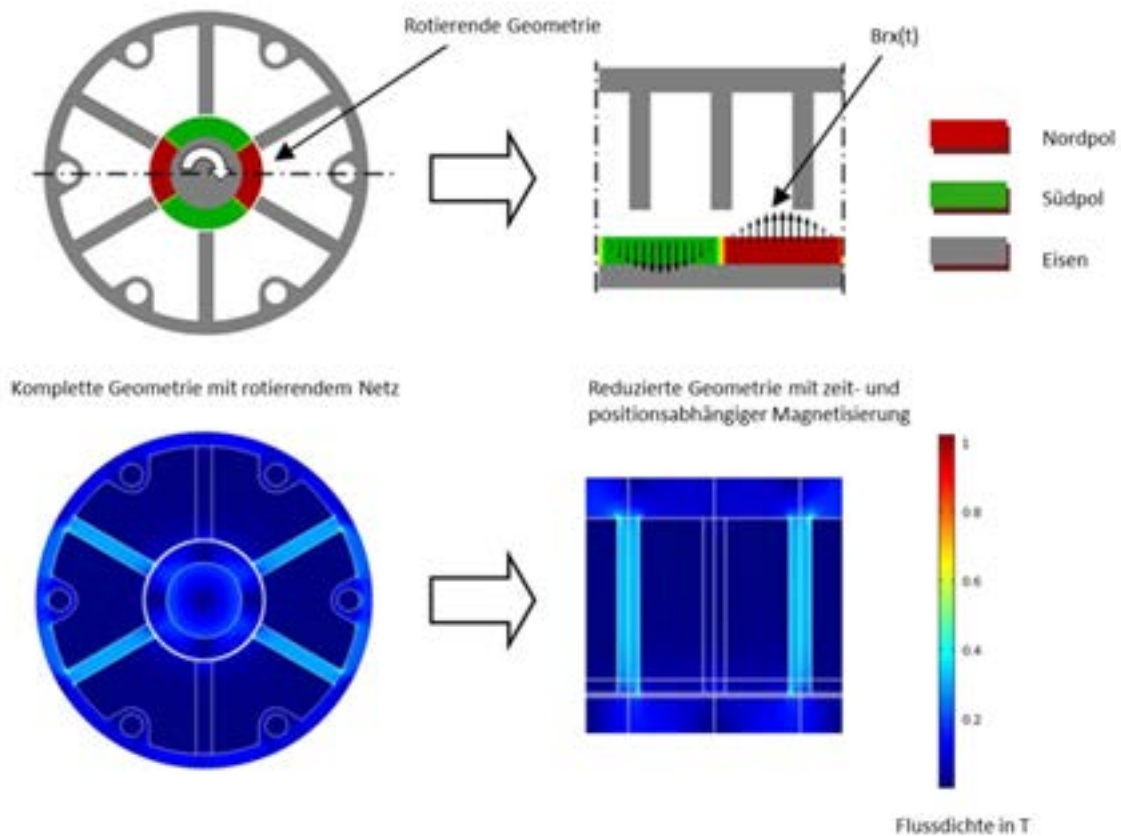


Abbildung 5: Modellreduzierung

Des Weiteren kann auf ein rotierendes Netz verzichtet werden, wenn stattdessen die Magneteigenschaften durch mathematische ortsabhängige Funktionen abgebildet werden und nicht an Geometrieobjekte in COMSOL gebunden sind. Abbildung 6 zeigt das Ergebnis der Optimierung.

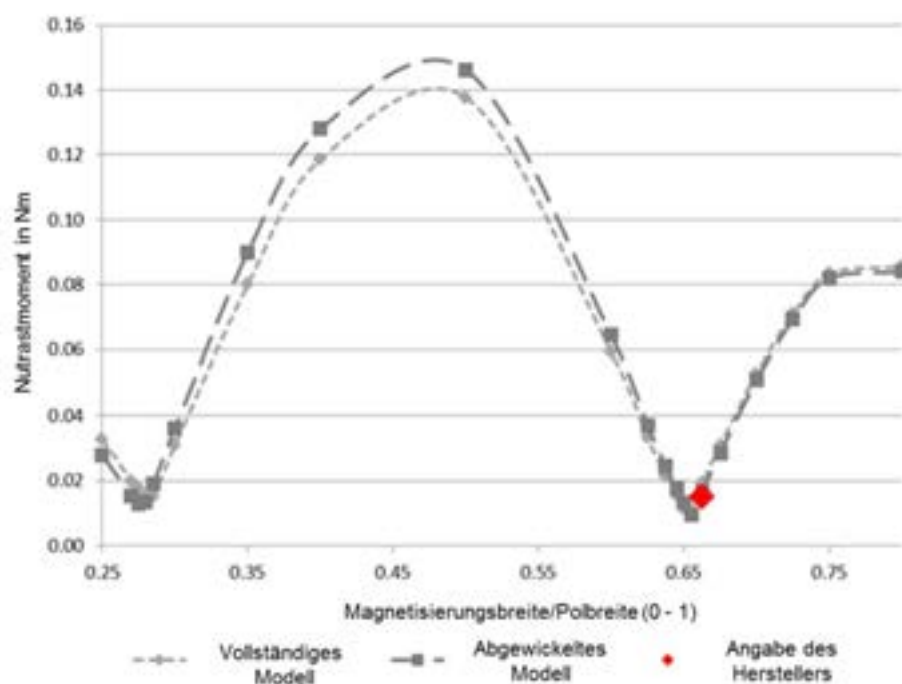


Abbildung 6: Parameterstudie

Es wird deutlich, dass die experimentell ermittelte Magnetisierungsbreite zur Reduzierung des Rastmoments gut mit der durch Simulation ermittelten Breite übereinstimmt. Außerdem bestätigen die ähnlichen Verläufe für das ursprüngliche und abgewickelte Maschinenmodell, dass eine Eingrenzung für Parameterstudien mit Hilfe des schnelleren abgewickelten Modells möglich ist. Um die verwendeten Modellierungsmethoden in EaSync zu validieren, wurde ein Prüfstand konzipiert der in Kapitel 0 näher beleuchtet wird. Das somit ermittelte Nutrastmoment ist in Abbildung 7 dem durch FEM ermittelten Drehmoment gegenübergestellt.

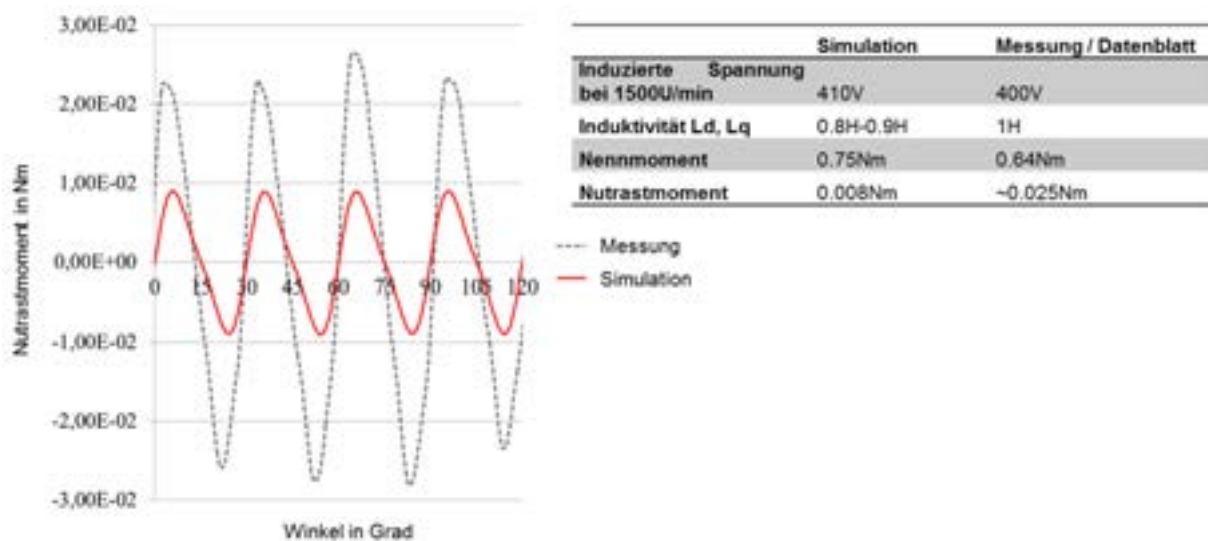


Abbildung 7: Vergleich der Mess- und Simulationsergebnisse

Abgesehen von der Filterung der Messwerte wurde das Drehmoment abgezogen, welches durch die Hysteresebremse erzeugt wurde (siehe Kapitel 0). Es ist zu erkennen, dass nach Abzug des Drehmoments der Hysteresebremse trotzdem ein Gleichanteil im Drehmoment enthalten ist. Dieser Gleichanteil ist auf Reibung zurückzuführen, da Nutrastmomente per se keinen Gleichanteil besitzen. Es zeigt sich eine gute Übereinstimmung in der Periodizität jedoch weniger in den Amplituden. Dabei sind die Schwankungen der Amplitude auf dem Messverlauf einer Exzentrizität des Rotors oder aber anderen Asymmetrien zuzuordnen. Die Abweichung der gemittelten Amplituden des Messverlaufs zu dem der Simulation hat mehrere Gründe. Zum einen führen Fertigungstoleranzen in aller Regel zu höheren Nutrastmomenten als in der Simulation. Laut Hersteller ist eine maximale Abweichung der Magnetisierungsbreite im Bereich von $\pm 2.5\%$ aufgrund drastischer Kostensteigerung bei höherer Genauigkeit unumgänglich. Aus Abbildung 6 geht hervor, dass eine Verdopplung des Nutrastmoments damit möglich wäre. Hinzu kommen Ungenauigkeiten bei der Simulation der Magnete aufgrund fehlender Daten. Um die Präzision bei der Modellierung von Magnetmaterialien zu steigern, wird im Rahmen einer studentischen Arbeit ein Messplatz

entwickelt, der mittels eines Fluxmeters die genaue Bestimmung des Magnetisierungsverlaufs ermöglicht.

5 Entwicklung des Prüfstands

Zur Validierung der Software EaSync wurde der Prüfstand in Abbildung 8 entwickelt. Dieser ermöglicht die Messung von Nutrastrmomenten. Dabei diente der Prüfstand der RWTH-Aachen als Vorbild [2]. Dieses Konzept basiert auf einem antreibenden Motor mit Untersetzung, einer Drehmomentmesswelle (Genauigkeit 0.5mNm), einer Vorrichtung zum Anheben des Drehmoments und dem Prüfling (DUT). Die Hysteresebremse überlagert auf das Signal des Nutrastrmoments einen Gleichanteil. Das Aufbringen eines zusätzlichen Drehmoments verschiebt den Messbereich aus dem Bereich der Nulllage der Drehmomentmesswelle, welcher durch Nichtlinearitäten gekennzeichnet ist.

Um eine Auswahl der Komponenten zu vereinfachen, wurde auf ein MATLAB Simulink Modell zurückgegriffen, welches die Dynamik des Systems abbildet. Aus diesem konnte die benötigte Untersetzung des Getriebes sowie die Elastizität der Kupplungen ermittelt werden, um eine Beeinflussung des Messsignals durch Störgrößen zu verhindern.

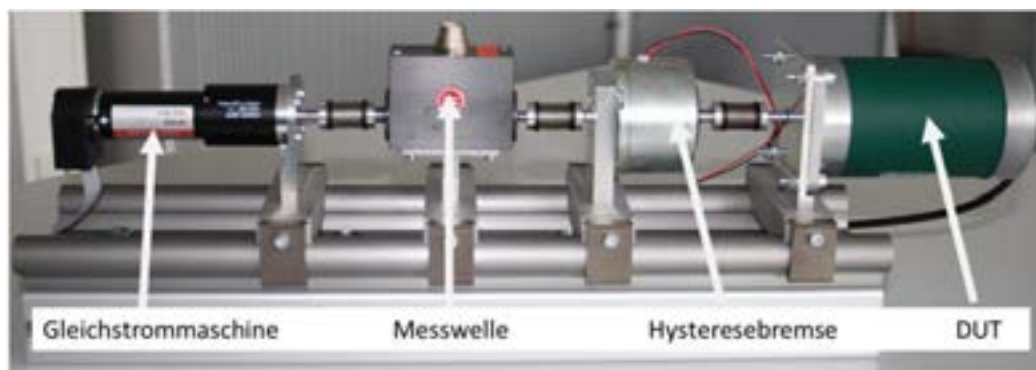


Abbildung 8: Prüfstand

Literatur

- [1] Müller, Germar; Vogt, Karl; Ponick Bernd: *Berechnung elektrischer Maschinen*, WILEY-VCH Verlag GmbH & Co. KGaA, 2011.
- [2] Schlensok, Christoph; van Riesen, Dirk; Schmülling, Benedikt; Schöning, Marc Christian; Hameyer, Kay: *Cogging-Torque Analysis on Permanent-Magnet Machines by Simulation and Measurement*, Technisches Messen 74, Oldenbourg Verlag, 2007.
- [3] Stebner, Gerhard E.; Hartwig, Christoph: *Cogging Torque Simulation Focused On Automated Preprocessing*, 12th International Workshop on Research and Education in Mechatronics, Kocaeli University, 2011.

Ein schnelles Verfahren zur digitalen Simulation von Regelungssysteme hoher Ordnung

Prof. Dr.-Ing. A. Makarov, Hochschule Magdeburg-Stendal
anatoli.makarov@hs-magdeburg.de

Zusammenfassung

Es wird ein Simulationsverfahren vorgestellt, welches gestattet, den Rechenaufwand bei der Simulation von Regelsystemen erheblich zu senken, ohne dass die Genauigkeit der Ergebnisse gesenkt wird. Für die Verwendung des Verfahrens wird vorausgesetzt, dass im zu simulierenden nichtlinearen Regelsystem einige lineare Teilsysteme hoher Ordnung formell ausgegliedert werden können, welche durch lineare gewöhnliche Differenzialgleichungen hoher Ordnung beschrieben werden können.

1 Problemstellung

Sowohl beim rechnergestützten Entwurf von Regel- und Steuereinrichtungen, als auch bei Echtzeitsteuerung und Regelung mit Prozessleitsystemen wird unter anderem eine Simulation des dynamischen Verhaltens des Objektes oder des Gesamtsystem durchgeführt. In Ost und West existieren Dutzende von solchen Simulationssystemen für verschiedene Anwendungsgebiete, unter anderem für die digitale Simulation und Optimierung von kontinuierlichen und diskontinuierlichen Regelkreisen. Man spricht heute von Simulationssystemen der 5. Generation. Sie unterscheiden sich hauptsächlich in der Gestaltung der Benutzerschnittstellen (block- oder gleichungsorientiert, Daten Ein- und Ausgabeformen). Wichtige Merkmale der Systeme 5. Generation sind dessen Objektorientiertheit und das Vorhandensein offener Schnittstellen, Möglichkeiten zur Einbettung von ActiveX- und COM-Komponenten. Zur eigentlichen Simulation des dynamischen Verhaltens werden aber in den meisten Simulationssystemen klassische Verfahren der numerischen Integration verwendet. Dies führt dazu, dass bei der Lösung vieler Praxisaufgaben der Rechenzeitaufwand die vertretbaren Grenzen überschreitet. Deshalb ist weiterhin die Forderung aktuell, numerische Simulationsverfahren zu entwickeln, die die höchst mögliche Genauigkeit bei möglichst geringem Rechenaufwand erreichen.

2 Mathematisches Modell eines Regelkreises

Allgemein gesehen kann man in einem Regelkreis ein oder mehrere Teilsysteme formell ausgliedern, z.B. solche, die durch nichtlineare Differentialgleichungssysteme

$$\dot{\bar{u}}(t) = \underline{F}(\bar{u}(t), y(t), t), \bar{u}(0) = \bar{u}_0, \quad (1)$$

und solche, die durch lineare gewöhnliche Differentialgleichungen

$$\dot{\bar{x}}(t) = A\bar{x}(t) + \bar{B}u(t), \bar{x}(0) = \bar{x}_0, \quad (2)$$

$$y(t) = \bar{C}^T \bar{x}(t) + du(t), \quad (3)$$

beschrieben werden können.

Wenn ein nichtlineares Teilsystem (1) durch q nichtlineare Gleichungen und ein lineares Teilsystem (2) durch n Gleichungen ersten Ordnung dargestellt wird, dann ist bei der digitalen Simulation des dynamischen Verhaltens bei jedem Simulationsschritt ein insgesamt nichtlineares Gleichungssystem bestehend aus (q+n) Differentialgleichungen zu lösen. Wenn noch dazu kommt, dass das zu simulierende System zu den steifen gehört und man implizite Integrationsverfahren verwenden muss, dann steigt der Rechenaufwand enorm.

3 Differenzengleichung zur Simulation des linearen Teilsystems

Das mathematische Modell eines jeden linearen Teilsystems (2-3) lässt sich zu einer gewöhnlichen linearen Differentialgleichung hoher Ordnung umwandeln mit entsprechender Übertragungsfunktion

$$F(p) = \frac{y(p)}{u(p)} = \frac{b_m p^m + b_{m-1} p^{m-1} + \dots + b_0}{p^n + a_{n-1} p^{n-1} + a_{n-2} p^{n-2} + \dots + a_1 p + a_0}. \quad (4)$$

Die Übertragungsfunktion (4) kann im Zustandsraum durch bekannte Steuerungsnormalform

$$\begin{bmatrix} \dot{x}_0 \\ \dot{x}_1 \\ \dots \\ \dot{x}_{n-2} \\ \dot{x}_{n-1} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 0 & 1 \\ -a_0 & -a_1 & -a_2 & \dots & -a_{n-2} & -a_{n-1} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_{n-2} \\ x_{n-1} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dots \\ 0 \\ 1 \end{bmatrix} u, \quad (5)$$

$$y = [b_0 \quad b_1 \quad \dots \quad b_m \quad 0 \quad \dots \quad 0] \times \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_{n-2} \\ x_{n-1} \end{bmatrix}, \quad (6)$$

dargestellt werden. In der Steuerungsnormalform sind alle Differentialgleichungen von der gleichen Art bis auf die letzte Gleichung. Das lässt sich ändern, indem eine zusätzliche

Zustandsvariable x_n eingeführt wird, sodass das Gleichungssystem (5)-(6) folgende modifizierte Form annimmt:

$$\begin{bmatrix} \dot{x}_0 \\ \dot{x}_1 \\ \dots \\ \dot{x}_{n-2} \\ \dot{x}_{n-1} \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 0 & 1 \\ -a_0 & -a_1 & -a_2 & \dots & -a_{n-1} & -1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dots \\ 0 \\ 0 \\ 1 \end{bmatrix} u, \quad (7)$$

$$y = [b_0 \ b_1 \ b_2 \dots b_{n-2} \ 0 \ 0] \times \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix}. \quad (8)$$

Der Zweck dieser Modifikation besteht darin, das algebraisierte Gleichungssystem zu vereinfachen. Durch Anwenden der impliziten numerischen Integrationsverfahren

$$x_{i+1} = \sum_{j=0}^s \gamma_j x_{i-j} + h \sum_{j=1}^s \beta_j f(x_{i-j}, t_{i-j}) \quad (9)$$

auf das Gleichungssystem (7), geht diese in ein algebraisches Gleichungssystem

$$\begin{bmatrix} 1 & -\alpha & 0 & \dots & 0 & 0 \\ 0 & 1 & -\alpha & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 0 & -\alpha \\ a_0 & a_1 & a_2 & \dots & a_{n-1} & 1 \end{bmatrix} \begin{bmatrix} x_{0,i+1} \\ x_{1,i+1} \\ \dots \\ x_{n-2,i+1} \\ x_{n-1,i+1} \\ x_{n,i+1} \end{bmatrix} = \begin{bmatrix} x_{0,i} \\ x_{1,i} \\ \dots \\ x_{n-2,i} \\ x_{n-1,i} \\ u_{i+1} \end{bmatrix} \quad (10)$$

über, wobei $\alpha = h\beta_{-1}$ ist.

Das Gleichungssystem $A^* \bar{x}_{i+1} = \bar{b}^*$ (10) wurde analytisch mit Hilfe der Cramerschen Formeln

$$x_{k,i+1} = \det A^*_k / \det A^* \quad (11)$$

analytisch gelöst. Entsprechende Determinanten haben folgende Ausdrücke

$$\det A^* = 1 + a_{n-1}\alpha + a_{n-2}\alpha^2 + a_{n-3}\alpha^3 + \dots + a_1\alpha^{n-1} + a_0\alpha^n, \quad (12)$$

$$\det A^*_k = \alpha^{n-k} u_{i+1} + \sum_{l=k+1}^n \left[a_l \alpha^{n-l} \sum_{j=k+1}^l p_j \alpha^{j-k-1} \right] - \sum_{l=1}^k \left[p_l \alpha^{n-l} \sum_{j=0}^{l-1} a_j \alpha^{l-j-1} \right], \quad (13)$$

wobei

$$p_k = \sum_{j=0}^s (\gamma_j x_{k,i-j} + h\beta_j x_{k,i-j}). \quad (14)$$

Wenn nun analytische Ausdrücke (12)-(13) in (8) eingesetzt werden, so erhält man schließlich folgende explizite Differenzengleichung zur Simulation des linearen Teilsystems

$$y_{i+1} = \frac{b_m \alpha^{n-m} + b_{m-1} \alpha^{n-m-1} + \dots + b_0 \alpha^n}{1 + a_{n-1} \alpha + a_{n-2} \alpha^2 + \dots + a_1 \alpha^{n-1} + a_0 \alpha^n} u_{i+1} + v_i. \quad (15)$$

Für die Simulation von einem beliebigen linearen Teilsystem kann diese eine Differenzengleichung (15) verwendet werden. Dabei ist die Ordnung des Integrationsverfahrens (9), passende Schrittweite h zu wählen und die jeweilige Anfangswerte x_{k-j} nach (14) zu berücksichtigen.

4 Genauigkeit und Rechenaufwand

Da die resultierende Differenzengleichung (15) analytisch hergeleitet wurde, sind keine zusätzlichen Fehler reingebracht worden. Man kann also auf diese Gleichung alle Eigenschaften der impliziten Integrationsverfahren übertragen und von den gleichen lokalen Fehlern sprechen.

Besonders vorteilhaft ist der Einsatz dieses Verfahrens bei der Simulation von nichtlinearen Systemen, wo für jedes lineare Teilsystem anstatt von n Gleichungen (2) nur diese eine Differenzengleichung (15) verwendet werden kann. Damit sind für das nichtlineare System anstatt $(q+n)$ nur $(q+1)$ Gleichungen auf jedem Simulationsschritt zu lösen. Der Rechenaufwand wird um ein $(q+n)/(q+1)$ -faches geringer, ohne dass die Genauigkeit der Simulation verschlechtert wird. Diesen Sachverhalt bestätigen zahlreiche und erfolgreiche Lösungen der Praxisaufgaben.

Literatur

- [1] Makarov, Anatoli: *Regelungstechnik und Simulation*, Vieweg-Verlag, 1996

Simulation und Regelung eines auf der Resonanzschwingungstechnologie basierenden Mischungsreaktors

Yongjian Ding
Institut für Elektrotechnik, Hochschule Magdeburg-Stendal
Breitscheidstraße 2, D-39114 Magdeburg
yongjian.ding@hs-magdeburg.de

Zusammenfassung: In dieser Arbeit wird zuerst die theoretische Untersuchung des Resonanzschwingungsverhaltens eines Mischungsreaktors aus der Verfahrenstechnik beschrieben. Mit Hilfe des Tools Matlab/Simulink wird anschließend das Verhalten der Regelstrecke simuliert und dabei gute Übereinstimmung mit den Ergebnissen der experimentellen Untersuchung erzielt. Darauf bauend wird ein Amplitudenregelalgorithmus entwickelt und auf einem Microcomputer realisiert. Weitere Lösungsansätze sowie Untersuchung komplexerer Reaktorstrukturen sind in Planung.

Stichworte: Resonanzschwingungstechnologie, RST, Simulation, Amplitudenregelung

1 Einleitung

In der Verfahrenstechnik wird seit über 20 Jahren an der sogenannten Resonanzschwingungstechnologie (RST) geforscht [1]. Sie eignet sich vorwiegend zum Mischen oder Dispergieren von Flüssigkeiten und darin enthaltenen festen oder gasförmigen Bestandteilen. Darüber hinaus wird durch einen verbesserten Wärmeaustausch die Reaktionszeit chemischer Prozesse beschleunigt. Nach Untersuchungen der Hochschule Merseburg – Initiator eines FHProfUnd-Forschungsprojekts - kann bei großtechnischer Anwendung mit einer Energieersparnis von 60% bis 90% gegenüber klassischen Rührwerken gerechnet werden [2].

Die Hochschule Magdeburg–Stendal entwickelt als Kooperationspartner im Projektverbund eine Regelung für diese neue Technologie. Um diese Regelung auf eine wissenschaftliche Grundlage zu stellen, wurde im Rahmen einer Bachelorarbeit zunächst ein Modell entworfen, welches das Streckenverhalten der Anlage aus Sicht einer Regeleinheit simulieren kann [3]. In einer weiteren studentischen Seminararbeit konnte die theoretische Analyse durch praktische Messungen bestätigt und ein Amplitudenregelalgorithmus auf einem Microcomputer realisiert werden [4].

2 Funktionsprinzip der Resonanzschwingungstechnologie (RST)

RST funktioniert nach dem Prinzip periodischer Kompression. Ein Gas, z.B. Luft, wird in den Tank periodisch hinein und wieder herausgepresst. Das Mischgut (Flüssigkeit) wird dadurch in Bewegung versetzt und so zu einer Schwingung angeregt.

Sind Erregerfrequenz und Eigenfrequenz der Schwingmasse (nahezu) identisch, herrscht Resonanz und eine vergleichsweise geringe Erregung führt zu einer großen Bewegung im Tank. Die Aufgabe einer energetisch optimalen Regelung besteht darin, den resonanten Schwingungszustand mit möglichst wenig Energieeinsatz dauerhaft aufrechtzuerhalten. Bild 1 zeigt das Grundprinzip einer geregelten RST-Apparatur.

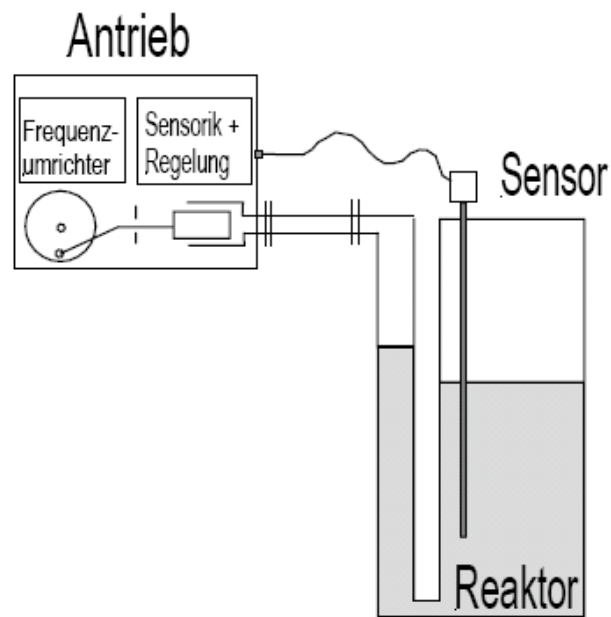


Bild 1: Grundprinzip einer geregelten RST-Apparatur /2/

3 Theoretische und praktische Untersuchungen des Streckenverhaltens

Die Anlage lässt sich ersatzweise auch als ein auf beiden Seiten geschlossenes U-Rohr mit je Seite konstantem aber ungleicher Querschnittsfläche ansehen (Bild 2).

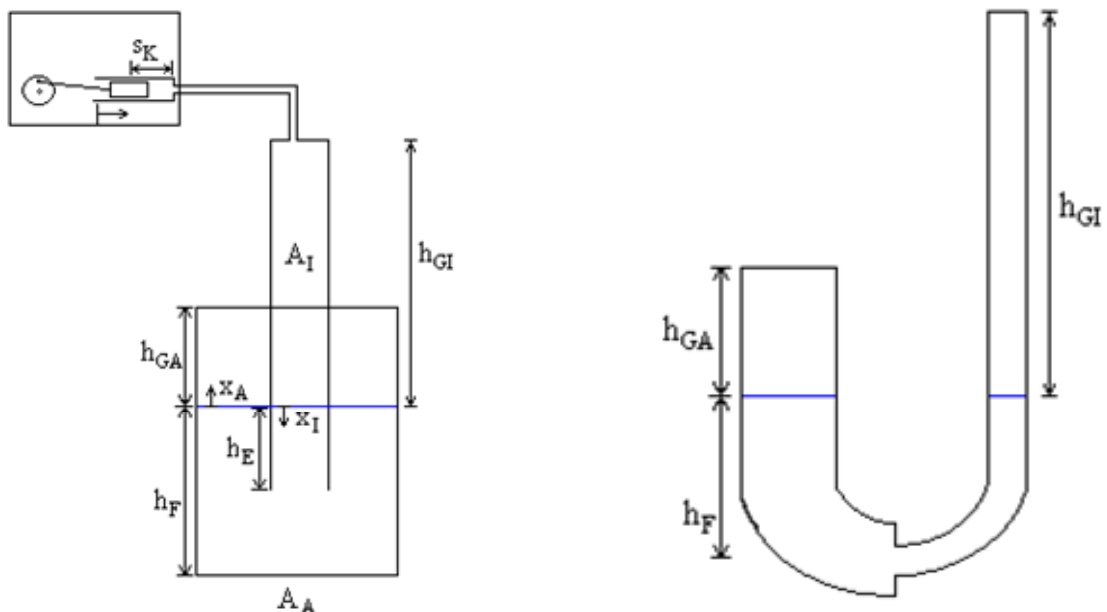


Bild 2: Ersatzschaltbild der Mischanlage

Um das Modell aufzustellen, kann die mittlere Bewegung des Mischgutes herangezogen werden. Dessen Verhalten lässt sich mit Hilfe der Bewegungsgleichung beschreiben. Dazu werden die wirkenden Kräfte wie in Gleichung 1 zusammengefasst.

Trägheitskraft + Reibungskraft + Schweredruck + Gaskompression Außenrohr +
+ Gaskompression Innenrohr = äußere Kraft

bzw.

$$F_M + F_R + F_G + F_{GA} + F_{GI} = F_A \quad (1)$$

Zur Beschreibung der einzelnen Kräfte wird der einfacheren Darstellung wegen, die 1. Ableitung des Weges x des Mischgutes nach der Zeit durch $\dot{x}$ dargestellt. Außerdem muss bei den Betrachtungen beachtet werden, dass das Außenrohr (**Index A**) und das Innenrohr (**Index I**) unterschiedliche Querschnittsflächen haben. Gemäß der Kontinuitätsgleichung gilt: $A_A \cdot \dot{x}_A = A_I \cdot \dot{x}_I$. Da der Querschnitt auf beiden Seiten jeweils konstant ist, ist die Fläche Zeit unabhängig. Damit gilt auch: $A_A \cdot x_A = A_I \cdot x_I$.

Durch Betrachtung der einzelnen Kräfte kann die Bewegungsgleichung durch folgenden Ausdruck dargestellt werden.

$$2\rho A_I h_F \ddot{x}_I + \rho A_I \left(\frac{A_I}{A_A} - 1 \right) x_I \ddot{x}_I + 8\pi\eta h_E \left(\frac{A_I}{A_A} + 1 \right) \dot{x}_I + \rho g (A_I + A_A) x_I + \frac{A_I \cdot p_0}{h_{GA} - \frac{A_I}{A_A} x_I} x_I +$$

$$+ p_0 \left(\frac{A_I \cdot x_I}{h_{GI} + x_I} - \frac{(r[1 - \cos \omega t]) \cdot A_K}{h_{GI} + x_I} \right) = \frac{p_0 \cdot A_K}{h_{GI} + x_I} r[1 - \cos \omega t]$$

Dieser lässt sich zu Gl. (2) umformen:

$$\left(2\rho A_I h_F + \rho A_I \left(\frac{A_I}{A_A} - 1 \right) x_I \right) \ddot{x}_I + \left(8\pi\eta h_E \left(\frac{A_I}{A_A} + 1 \right) \right) \dot{x}_I +$$

$$+ \left(\rho g (A_I + A_A) + \frac{A_I \cdot p_0}{h_{GA} - \frac{A_I}{A_A} x_I} + \frac{A_I \cdot p_0}{h_{GI} + x_I} \right) x_I = \frac{2p_0 \cdot A_K \cdot r}{h_{GI} + x_I} [1 - \cos \omega t] \quad (2)$$

Auch wenn die Gleichung (2) auf den ersten Blick die Form eines PT2-Übertragungsglieds hat, trifft die Vermutung nur unter den Bedingungen $x_I \ll h_{GI}$ und $A_A \gg A_I$ zu, wie dies in der Geometrie der verwendeten Laboranordnung annähernd der Fall ist. Außerdem gilt das Kräftemodell eigentlich nur für den Fall der laminaren Strömung, die besonders im Bereich der Resonanz nicht mehr zutrifft.

Wird die Gleichung (2) nach $\ddot{x}_I$ umgestellt, so ergibt sich das Strukturbild in Bild 3:

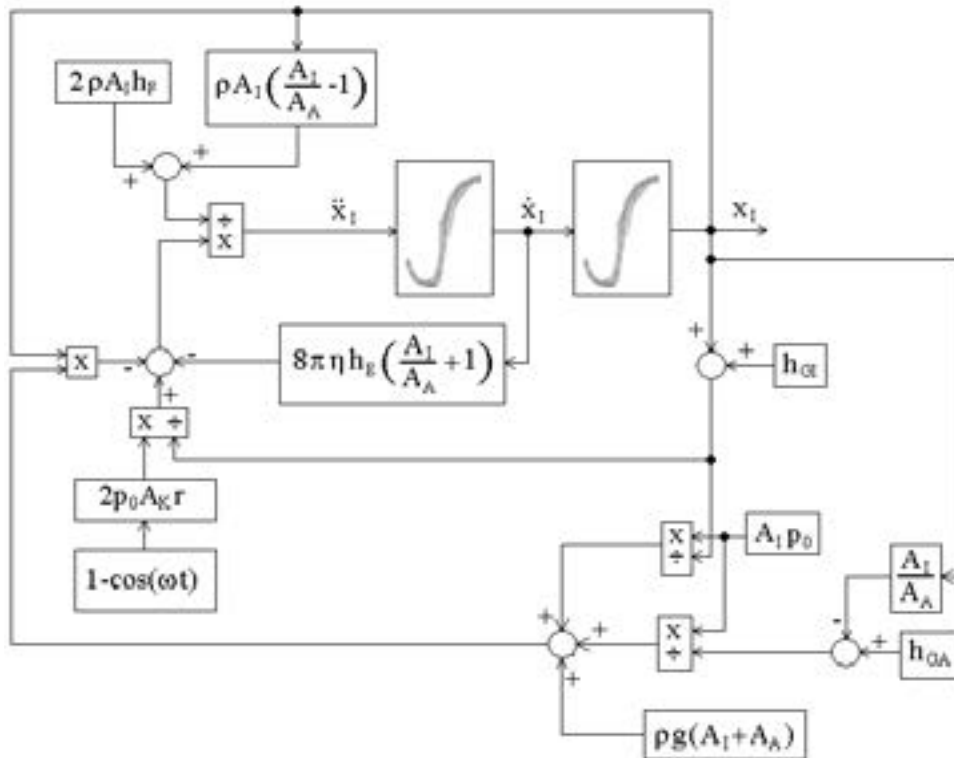


Bild 3: Strukturbild des vollständigen Anlagenmodells /3/

Die Simulation dieses Modells mit dem Programm Simulink mit verschiedenen Parametereinstellungen am Computer lieferte erstaunlicherweise dennoch ein PT2-ähnliches Schwingungsverhalten, wie der Verlauf des Amplitudengangs für verschiedene Anregefrequenzen zeigt (Bild 4).

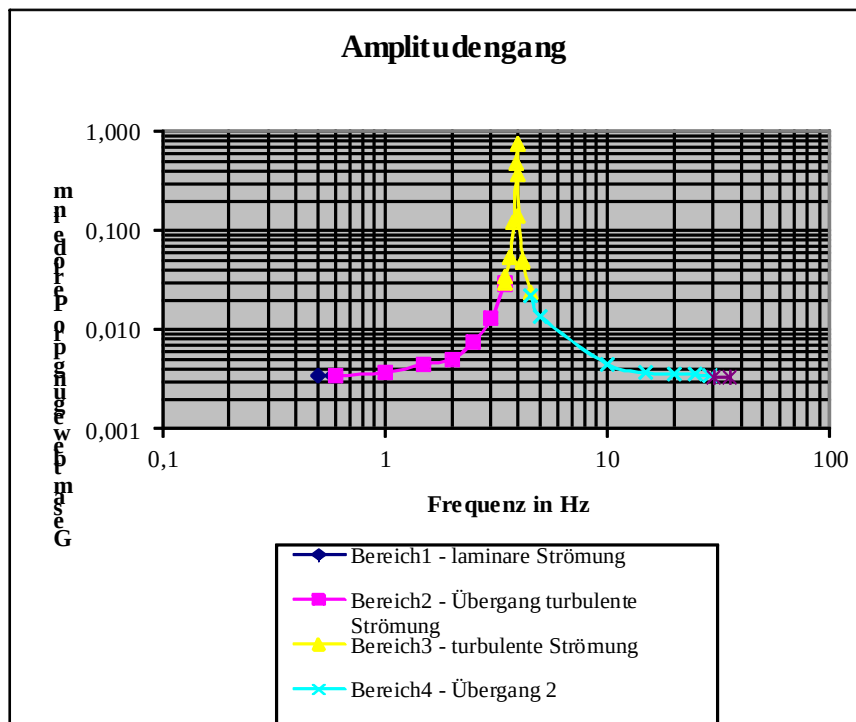


Bild 4: Simulierter Amplitudengang der Anlage mit Wasser als Mischgut

4 Experimentelle Untersuchungen und die Amplitudenregelung

Die theoretischen Ergebnisse werden nun durch experimentelle Untersuchungen verifiziert. Mit Blick auf die Randbedingungen der praktischen Anwendung in der späteren industriellen Umgebung ein Messkonzept erarbeitet, welches im Wesentlichen aus einem Lichtschrank zur Detektion des oberen Totpunkts des Pumpenkolbens, sowie einem Drucksensor in der Luftsäule besteht. Dadurch können sowohl die Pumpendrehzahl wie auch der Phasenversatz zwischen der Anregung und der Flüssigkeitsschwingung – indirekt gemessen anhand des Drucksensors - ausgewertet werden.

Bild 5 zeigt die Verläufe des Anregesignals und des Druck-Istwertes. Dabei wird die Frequenz der Kolbenbewegung (blaue Linie) über die Erhöhung der Spannung am Frequenzumrichter kontinuierlich beschleunigt, die Schwingung – dargestellt durch Druckverlauf (rote Kurve) – steigt zuerst mit bis zur Resonanz. Danach bricht die Schwingung zusammen und die Amplitude des Drucksignals sinkt rapide ab. Durch Bestimmung der Nulldurchgänge beider Signale lässt sich auch die Phasenverschiebung leicht ermitteln.

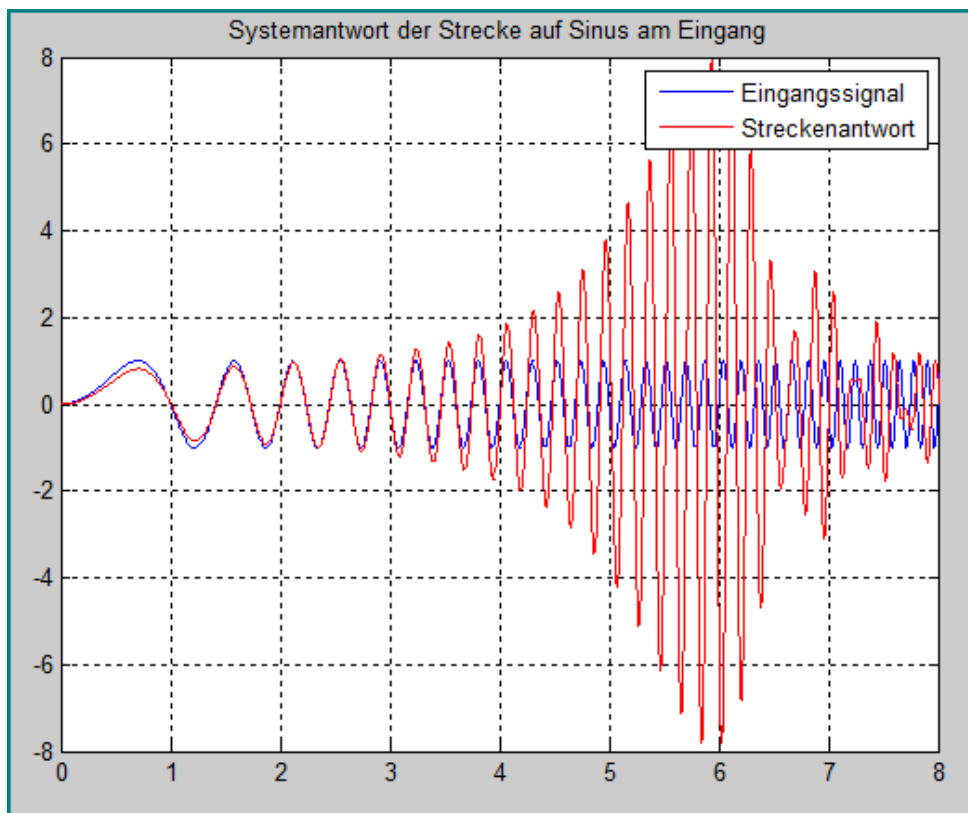


Bild 5: Messungen der durch die Schwingung erzeugten Druckschwankungen als Streckenanalyse

Als Regelgröße kämen daher prinzipiell sowohl die Phase als auch die Amplitude in Frage, wobei die Amplitudenregelung etwas leichter (eindeutiger) zu realisieren war. Anhand einer Frequenzrampe kombiniert mit der Suche nach dem Amplitudenmaximum (genauer gesagt nach dem starken Amplitudenabfall kurz vor der Resonanz) lässt sich die Schwingung im resonanten

Bereich halten. Mögliche Streckenänderungen im Laufe der Zeit können durch (ggf. regelmäßige) Triggerung der neuen Suchvorgänge kompensiert werden.

5 Realisierung der Amplitudenregelung

Bild 6 zeigt das Blockschaltbild der gesamten RST-Regelung. In Abstimmung mit den Projektpartnern wurde ein Combi-Modul der Fa. Phytect verwendet. Für die SW-Entwicklung wurden die Tasking EDE Tools for C166 verwendet [5].

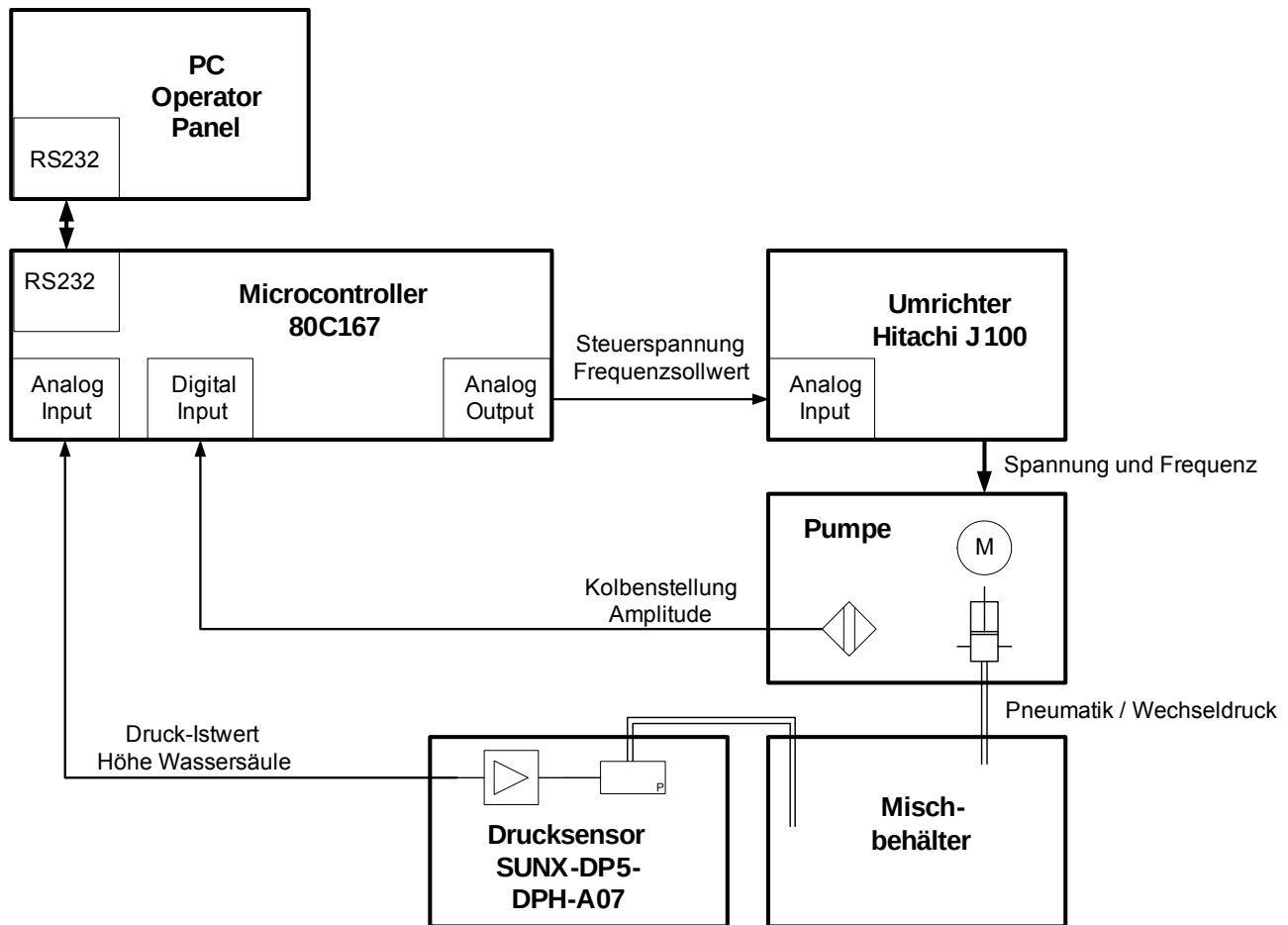


Bild 6: Blockschaltbild der realisierten Amplitudenregelung

In der Zwischenzeit wurde zusätzlich ein Prototyp des Reglers basierend auf einer Industrie-SPS für den praktischen Einsatz realisiert und vom Projektpartner als Vorführgerät verwendet.

6 Zusammenfassung und Ausblick

Für die Weiterentwicklung der Resonanzschwingungstechnologie wurde ein μ P-basierter Amplitudenregler an der Hochschule Magdeburg-Stendal entwickelt. Dazu wurde zuerst ein analytisches Mischbehältermodell entwickelt und das Streckenverhalten simuliert. Die theoretischen Ergebnisse wurden dann mit praktischen Messungen an einem Versuchsreaktor im Labor verifiziert. Der implementierter Amplitudenregler funktioniert als Prototyp einwandfrei

und es gibt Überlegungen, den Algorithmus auf größere Industrieanlagen mit evt. mehreren Anregepumpen zu erweitern bzw. andere Algorithmen wie eine phasenbasierte Regelung zu untersuchen.

7 Literatur

- [1] Ostrovsky, G.M., P.A. Malyschew, E.G. Aksjonowa: Über Pulsationsapparate im Fluid-Resonanzregime, Teortitscheskije Osnowy chimitscheskoj Technologij, 24 (1990), Akademia Nauk SSSR, Moskau
- [2] R. Säuberlich et.al.: Einführung der Resonanzschwingungstechnologie (RST) in den Apparatebau, Projektantrag im Rahmen des Programms Forschung an Fachhochschulen mit Unternehmen (FHprofUnd), 2008.
- [3] Henry Adam: Modellbildung und Untersuchung einer auf der Resonanzschwingungstechnologie basierenden Mischungsanlage als Vorbereitung für Regelungsaufgaben. Bachelorarbeit am Institut für Elektrotechnik, Hochschule Magdeburg-Stendal, 2008.
- [4] S. Arndt, S. Arnold, W. Steinbach und A. Storbeck: Projektdokumentation steuerungs- und regelungstechnisches Seminar SoS 2010 am Institut für Elektrotechnik, Hochschule Magdeburg-Stendal, 2010.
- [5] Yongjian Ding und Theodor Schmied: Regelung einer Mischungsanlage basierend auf der Resonanzschwingungstechnologie. 8. Fachwissenschaftliches Kolloquium „Angewandte Automatisierungstechnik in Lehre und Entwicklung an Fachhochschulen“, Göttingen 2011.



Domänenspezifische Konfiguration von Gesamtfahrzeugsimulationen

M.Sc. Florian Pramme | Prof. Dr. Gert Bikker | Dipl.-Ing. Martin Loeffler



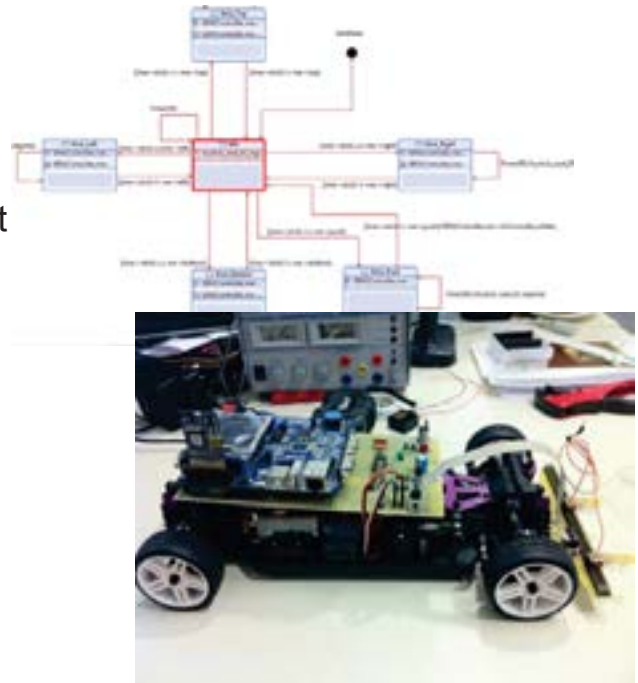
Wollenbüttel

Agenda

- Motivation
- Virtuelle Absicherung
- Projekt SIMBA
 - Grafische Notation
 - Beschreibung einer Simulation

Fakultät Informatik – Institut für verteilte Systeme

- Forschungsgebiete:
 - Simulation und Test
 - Anforderungs- u. Testmanagement
 - Embedded Software Engineering
 - Modellbasierte Entwicklung / Test
- Autonomes Fahren
- Robotik



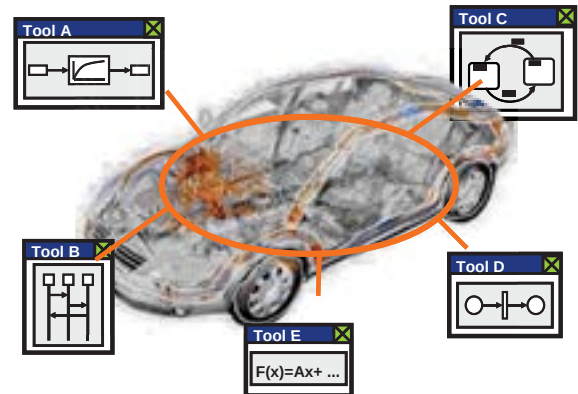
M.Sc. F. Pramme | Domänenspezifische Konfiguration von Gesamtfahrzeugsimulationen | 23.02.2012

3

MOTIVATION

Motivation

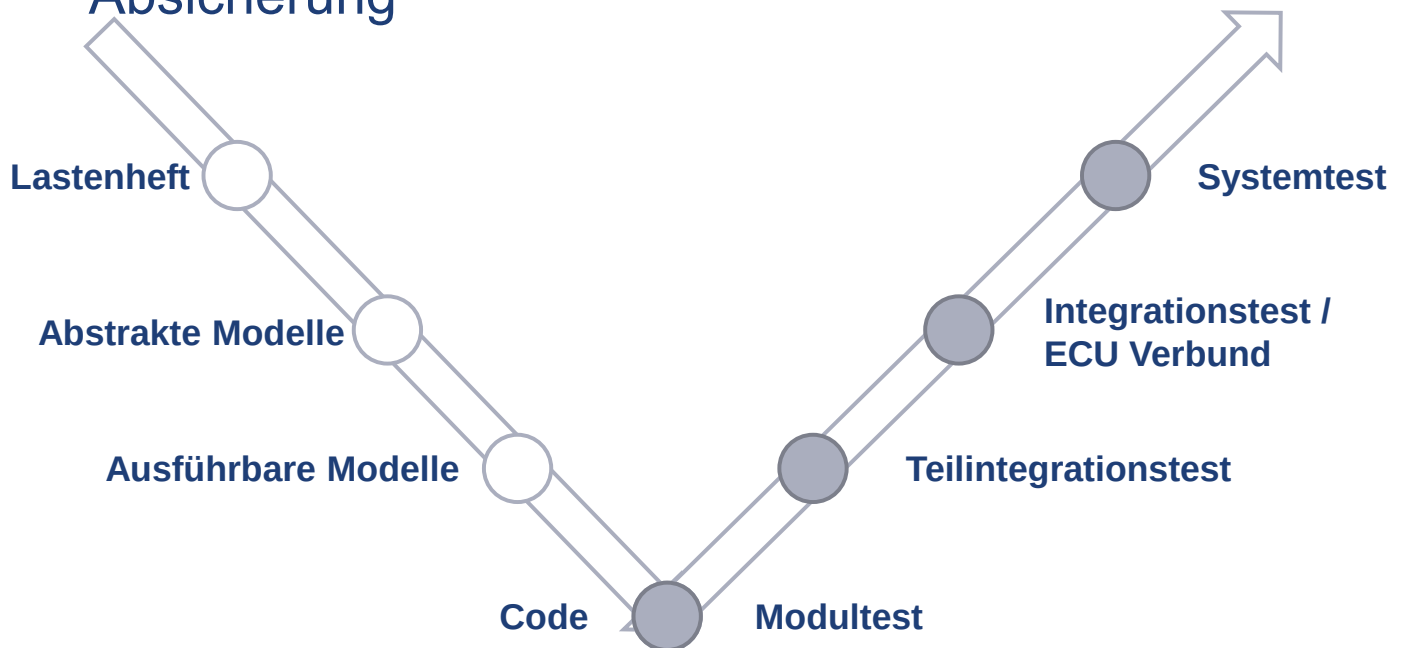
- Digital Mockups
- Vorliegende Modelle
- Redundanz von Modellen
- Warten auf reale Hardware



- Ziel:
 - Multiuser Appliance für Gesamtfahrzeugsimulation

VIRTUELLE ABSICHERUNG

Idee der virtuellen Integration und Absicherung



M.Sc. F. Pramme | Domänenspezifische Konfiguration von Gesamtfahrzeugsimulationen | 23.02.2012

7

Virtuelle Absicherung und Kooperation

Bisher: **getrennte** Entwicklung



Auf dem Weg: **gemeinsame** Entwicklung



- Teamübergreifender Austausch von Modellen / Modulgruppen:
 - Reduzierung des Modellierungsaufwands
 - Bessere Simulationsergebnisse durch weniger Abstraktion
 - Wiederverwendung des Experten-Know-hows
 - Absicherung ist Architekturentwicklung
- Voraussetzung:
 - Toolunterstützung
 - **Modulbasierter** Entwicklungsprozess



Simulation Backplane Automotive

SIMBA

Gefördert durch das



Bundesministerium
für Wirtschaft
und Technologie



Projektarchitektur Simba

Benutzeroberfläche

Co-Simulation

Virtualisierung

Ressourcenmanagement

Hardwaremanagement

Hardware

Partner



TLK-Thermo GmbH



CHRISTMANN
INFORMATIONSTECHNIK + MEDIEN



IQS
Industrie-Elektronik



UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

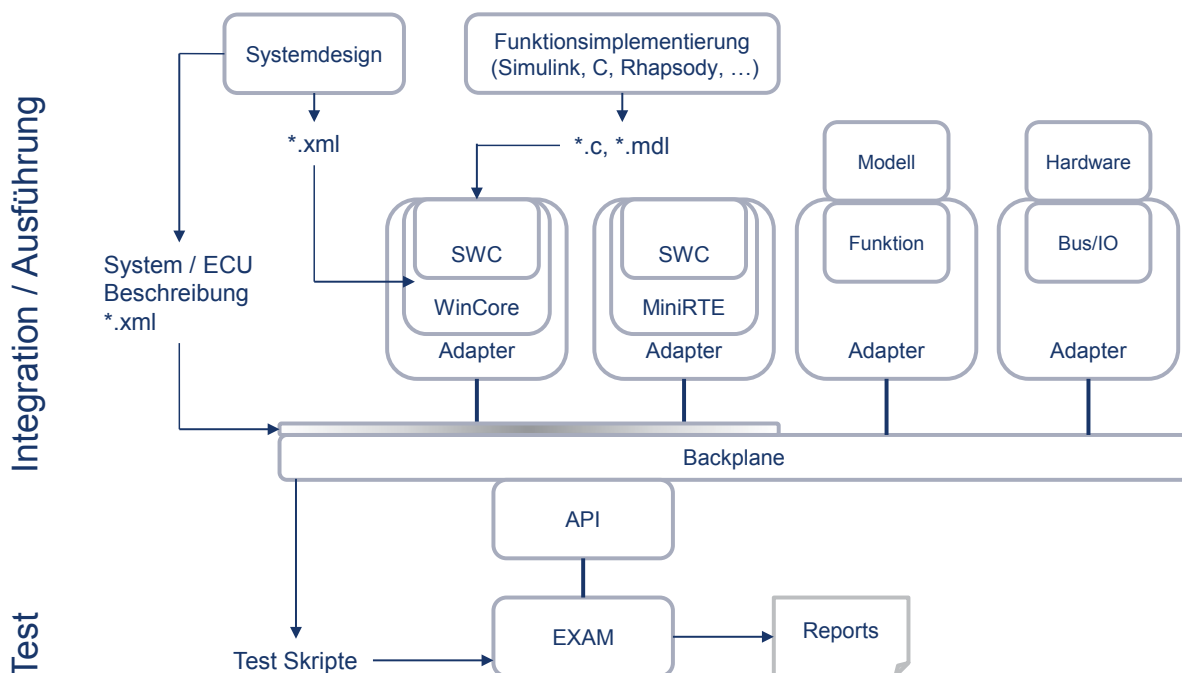


Projektarchitektur Simba

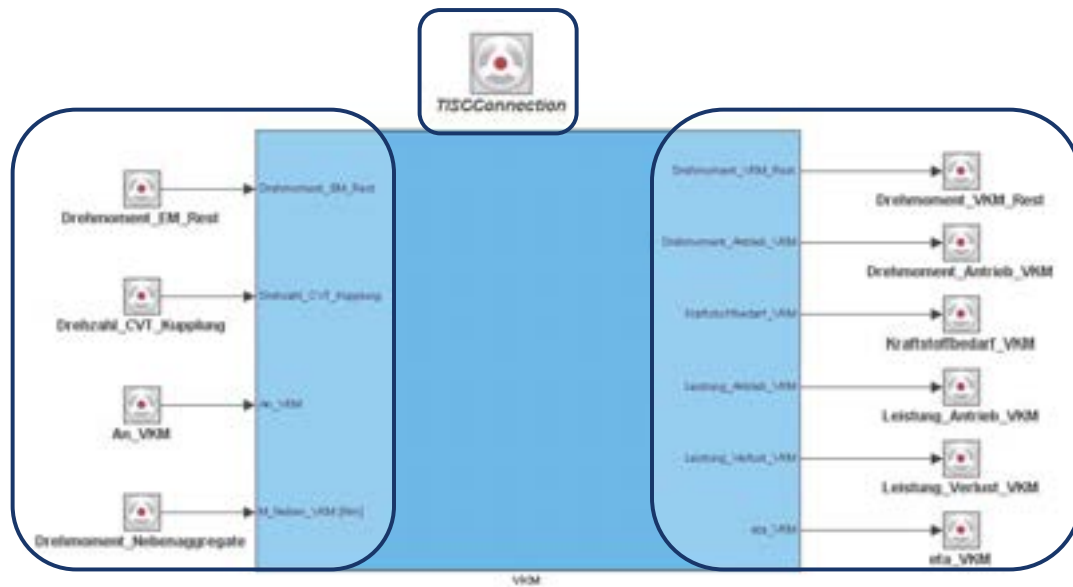


- Entwicklung einer grafischen Notation
- Weiterentwicklung vorhandener Strukturen
- Schaffung von Schnittstellen zum Datenaustausch
- Beschreibung von Teil und Gesamtsystemen
- Domänenspezifische Konfiguration von Gesamtfahrzeugsimulationen

Zielarchitektur



TISC | Deklaration der Variablen



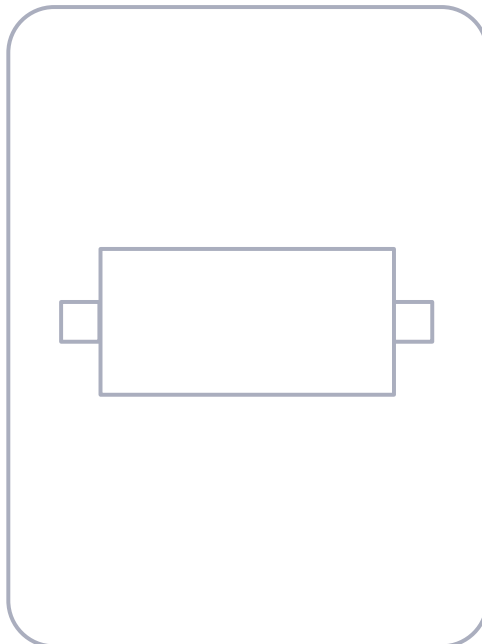
M.Sc. F. Pramme | Domänenspezifische Konfiguration von Gesamtfahrzeugsimulationen | 23.02.2012

13

Beschreibung einer Simulation

GRAFISCHE NOTATION

Grafische Notation



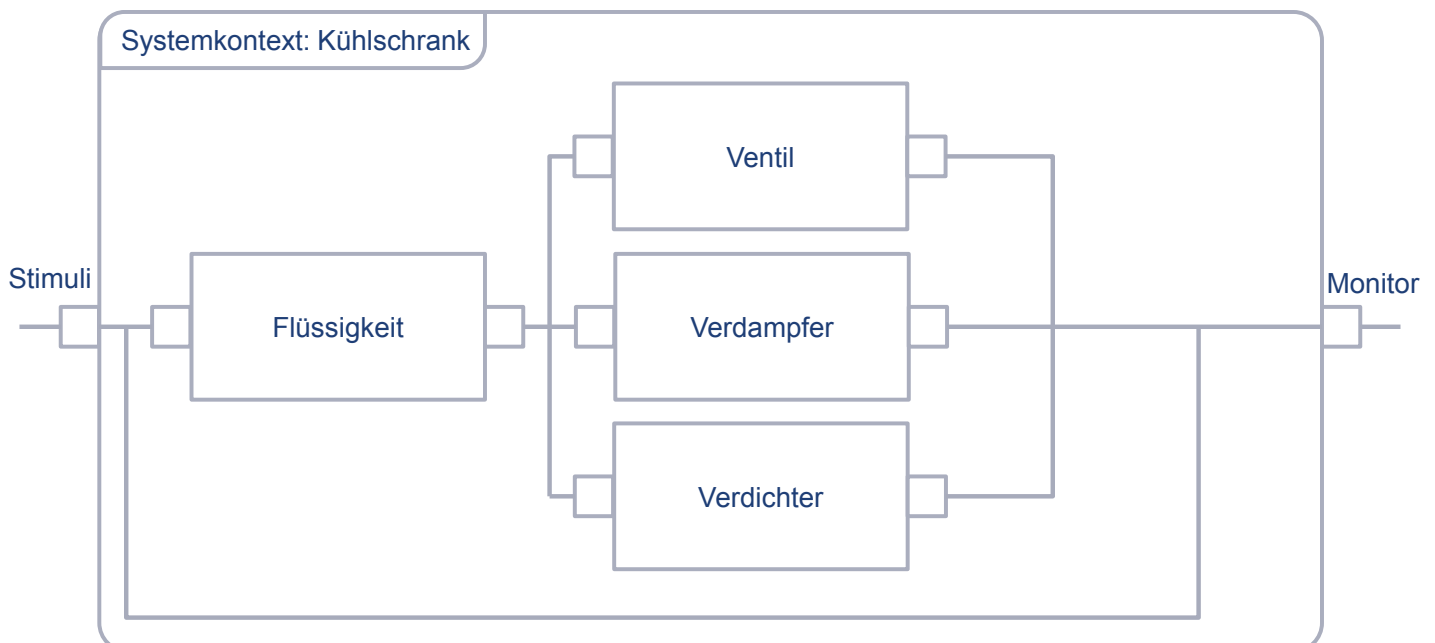
Teilsystem / System

- Systemkomponente
- Softwarekomponente

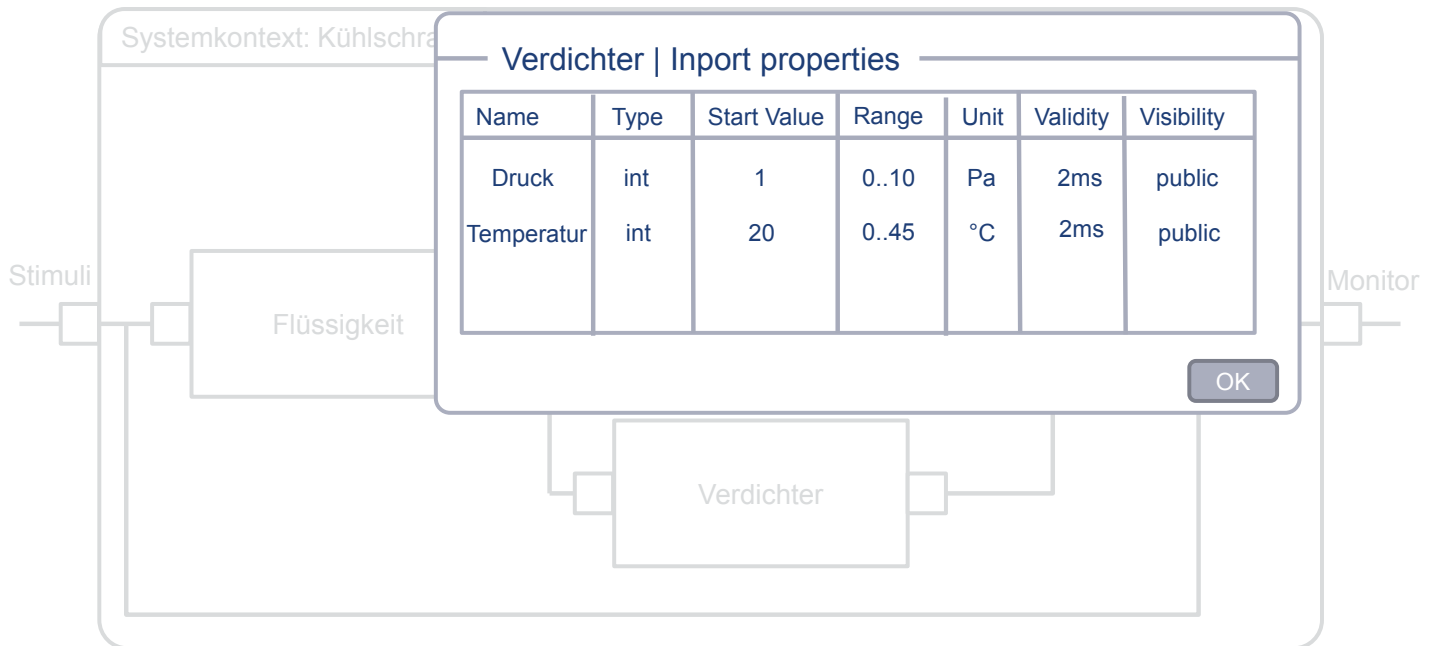
Ports

- Unterteilt in IN und OUT Ports
- Zusammenfassung unterschiedlicher eingangs-, und ausgangsvARIABLEN

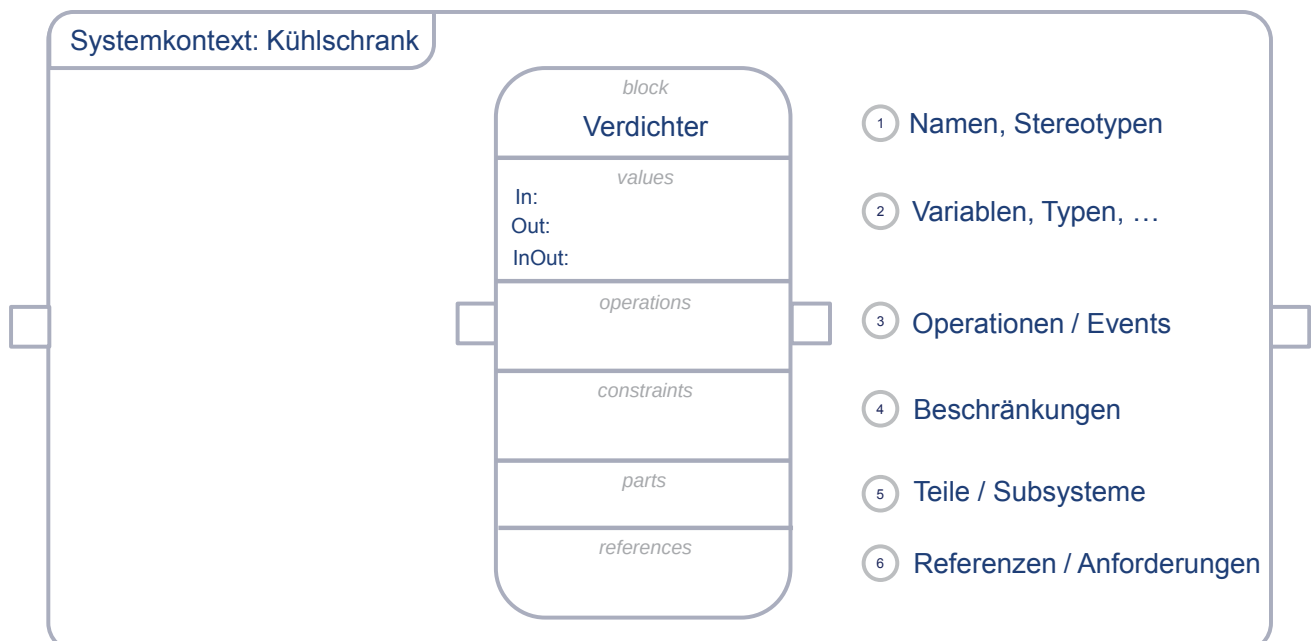
Grafische Notation



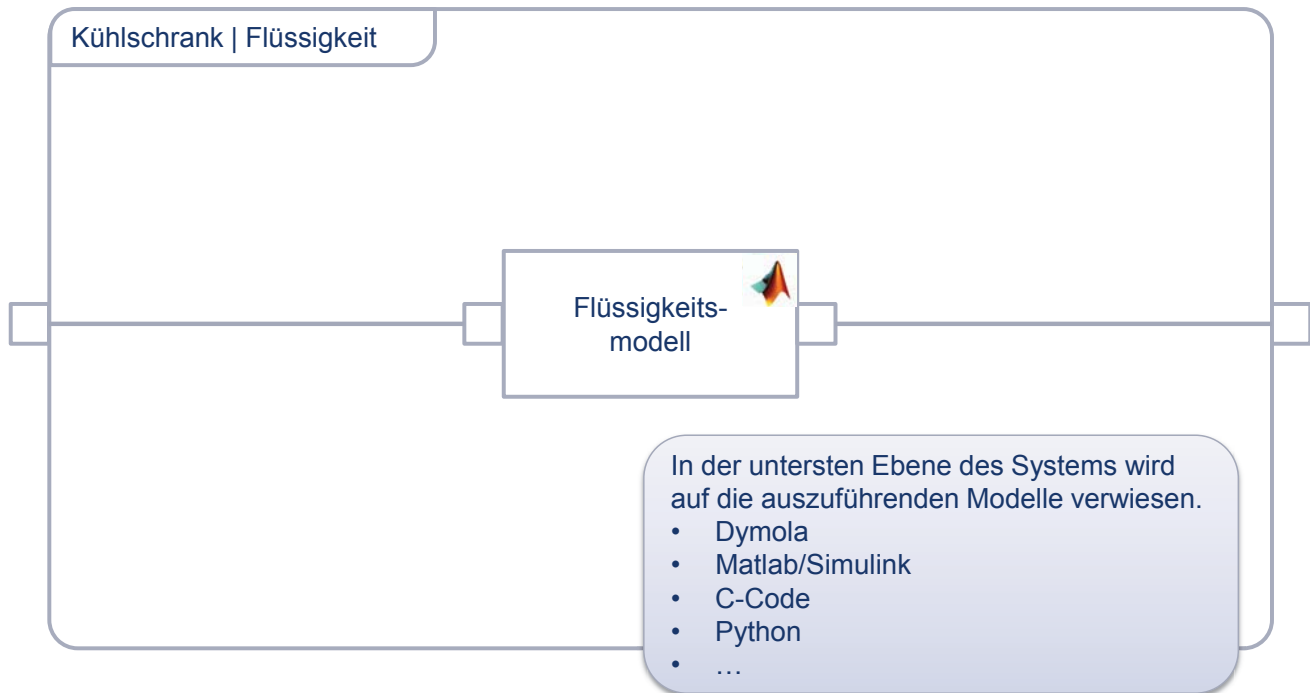
Grafische Notation | Parametrisierung



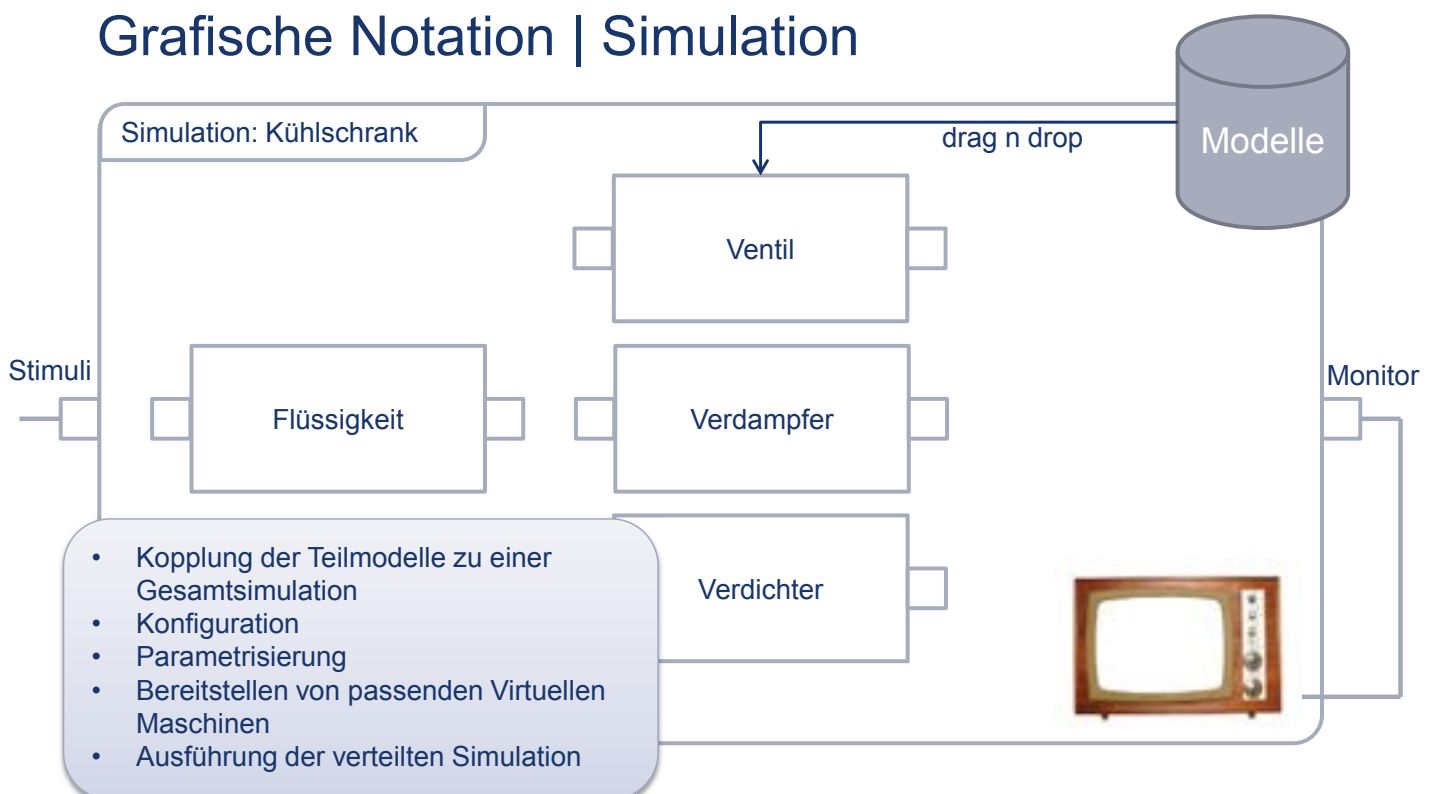
Grafische Notation | Expertensicht



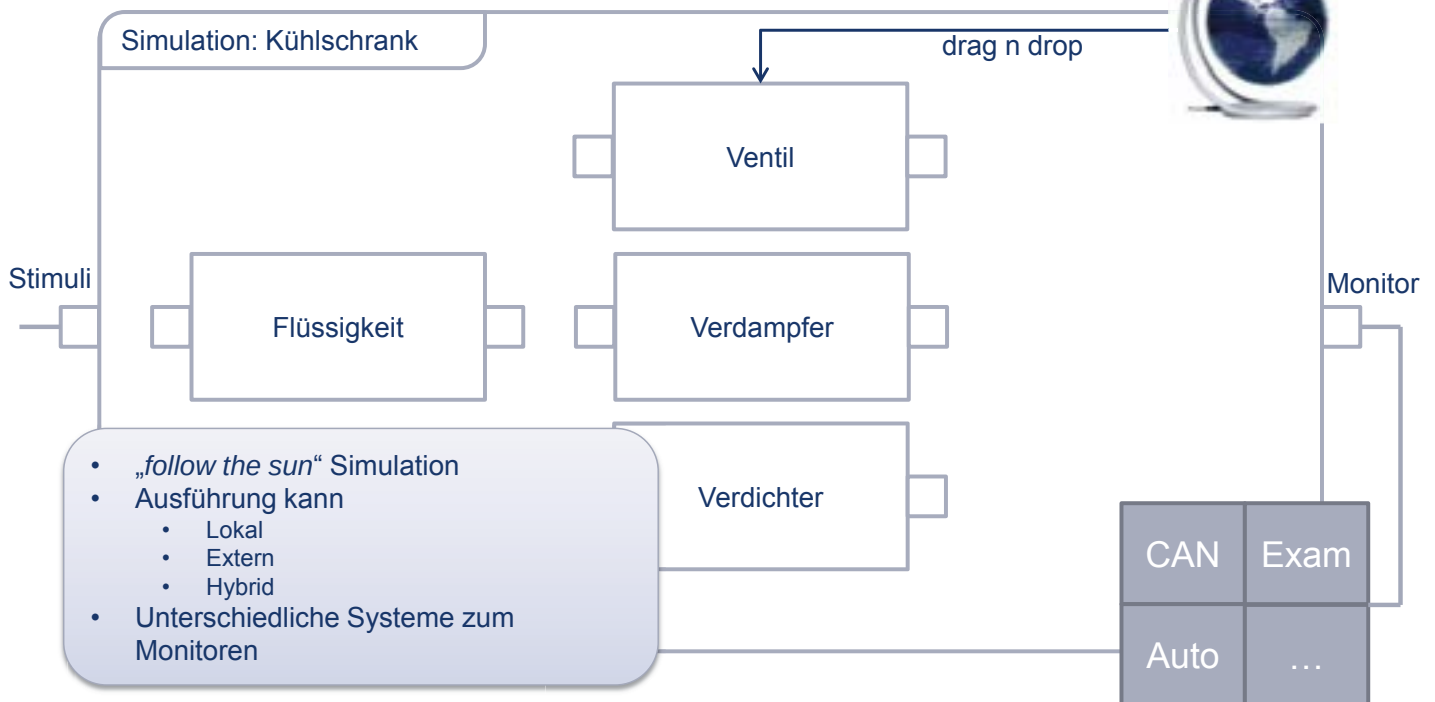
Grafische Notation | Auszuführende Modelle



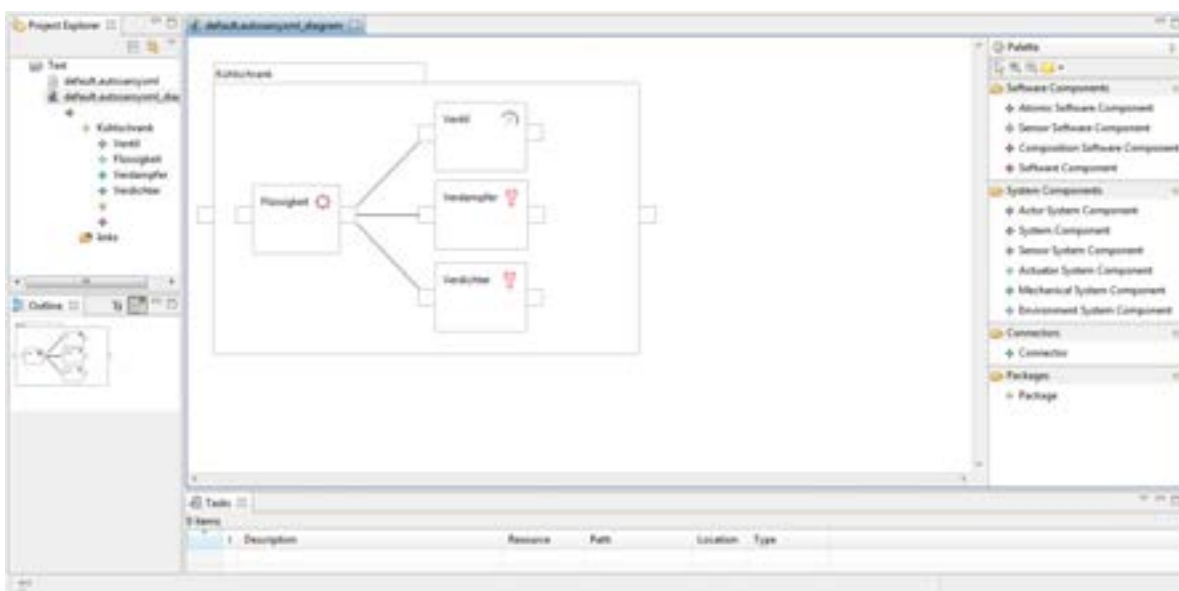
Grafische Notation | Simulation



Grafische Notation | Simulation



Grafische Notation | Tool



Vorteile

- Modelle liegen bereits früh im Entwicklungsprozess vor
- Gemeinsames Nutzen führt zu schnelleren Ergebnissen
- Dokumentation durch Modelle
- Beschreibung von Gesamtsystemen
- Verteilte Simulation (online / offline)
- „*follow the sun*“ Simulation
- Frühzeitiges validieren von Systemfunktionalitäten

VIELEN DANK FÜR IHRE AUFMERKSAMKEIT

Autor

Kontakt:

M.Sc. Florian Pramme

Ostfalia Hochschule für angewandte
Wissenschaften

Fakultät Informatik
Institut für Verteilte Systeme
Salzdahlumer Str. 46/48
38302 Wolfenbüttel
Deutschland

Telefon:

(+49) 5331 939 32 420

Telefax:

(+49) 5331 939 32 422

Email:

florian.pramme@ostfalia.de

SystemC-AMS basierte Modellierung, Simulation und HiL Echtzeitsimulation für Anwendungen der Automobilelektronik

Karsten Einwich, Thomas Arndt; Fraunhofer IIS/EAS

karsten.einwich@eas.iis.fraunhofer.de, thomas.arndt@eas.iis.fraunhofer.de

Zusammenfassung

Elektronische Komponenten in der Automobiltechnik sind immer mehr mit den Komponenten anderer Domänen vernetzt. Damit werden deren Spezifikation und Entwurf zu einer wachsenden Herausforderung. Dieser Beitrag stellt eine auf SystemC/SystemC-AMS basierende Modellierungs- und Simulationsmethodik vor, mit deren Hilfe diesen neuen Anforderungen begegnet werden kann.

1 Einführung

Elektronik spielt in der Automobiltechnik eine immer größer werdende Rolle. Sie ermöglicht neben der Einführung neuer Sicherheits- und Komfortfunktionen die Reduzierung des Schadstoffausstoßes und des Energieverbrauchs sowie signifikante Kosteneinsparungen. Um diese Ziele zu erreichen, muss die elektronische Hard- und Software immer enger mit den nichtelektronischen (z.B. mechanischen) Komponenten verwoben werden. Da Elektronik und im speziellen Software im Produktionsprozess extrem kostengünstig sind, wird mehr und mehr versucht, teure mechanische bzw. hydraulische Komponenten zu vereinfachen und die dadurch entstehenden Nachteile durch Elektronik zu kompensieren. Speziell durch die Software wird zusätzlich eine hohe Flexibilität und Konfigurierbarkeit erreicht. Dies hat zur Konsequenz, dass im Entwicklungsprozess die einzelnen Komponenten immer weniger getrennt voneinander betrachtet werden können. Die Systemfunktionalität ergibt sich zunehmend aus einer immer engeren Interaktion von nicht elektrischen (z.B. mechanischen) und elektronischen Hard- und Softwarekomponenten. Diese enge Verknüpfung wird für den Entwurfsprozess zunehmend zu einer Herausforderung. Da speziell in der Automobiltechnik die Entwicklung des Gesamtsystems (Auto) über viele Firmen verteilt erfolgt, ist diese enge Interaktion schwer zu handhaben. Klassisch besteht die Zulieferkette vom Halbleiterhersteller (TIER2) über den Komponentenhersteller (TIER1) zum Autoproduzenten (OEM). Außerdem müssen verschiedene Fachgebiete zusammenarbeiten, wobei die Schnittstelle zwischen diesen immer mehr verwischt, d.h. immer schwerer klar zu spezifizieren ist. Damit gewinnen Simulations- und Modellierungstechniken, welche eine Gesamtsystembetrachtung und einen Austausch zwischen verschiedenen Fachgebieten und Firmen erlauben eine wachsende Bedeutung. Solche Techniken ermöglichen zum einen die gemeinsame Erarbeitung der Komponentenspezifikationen als auch die Entwicklung der Hard- und Software unter Berücksichtigung der Gesamtsystemfunktionalität. In diesem Beitrag möchten wir solch eine vielversprechende Methode vorstellen und die Möglichkeiten, die diese bietet diskutieren. Diese beruht auf der Hardwarebeschreibungssprache SystemC und deren Erweiterung zur Modellierung analogen Verhaltens SystemC-AMS.

2 SystemC / SystemC-AMS

SystemC [1] ist eine Hardwarebeschreibungssprache welche auf der Programmiersprache C++ aufbaut. Somit ist SystemC eine Definition von C++ Klassen und Funktionen welche die Beschreibung der Struktur und des Verhaltens von Hardwarekomponenten erlauben. SystemC wird von der Accellera Systems Initiative – einer Industrievereinigung – entwickelt und standardisiert. Somit ist SystemC auch ein IEEE Standard (IEEE 1666). Der Fokus von SystemC liegt in der Beschreibung von komplexen digitalen Hard-/Softwaresystemen auf höheren Abstraktionsebenen. SystemC-AMS [2,3,4] – auch standardisiert von Accellera – erweitert SystemC mit der Möglichkeit analoges und gemischt analog/digitales Verhalten auf hohen Abstraktionsebenen zu beschreiben. Ziel ist es die Beschreibung und Simulation des Gesamtsystems bestehend aus analoger und digitaler Hard- und Software in seiner Umgebung zu ermöglichen. Dabei sollen komplette Anwendungsszenarien mit sinnvollem Aufwand simulierbar sein. Damit ist der Hauptanwendungsbereich von SystemC/SystemC-AMS die Erstellung einer ausführbaren Spezifikation, der Architekturentwurf und die Bereitstellung eines Referenzmodells zur Stimulierung und Überprüfung von implementierten Komponenten. Um vor allem die erforderliche Simulationsgeschwindigkeit zu erreichen, setzt SystemC/SystemC-AMS auf das Konzept von verschiedenen sogenannten Berechnungsmodellen (Models of Computation - MoC). Dabei ist die Idee, dass die unterschiedlichen Systemkomponenten mit dem für sie besten Beschreibungsmittel hinsichtlich Aufwand, erforderlicher Genauigkeit und Simulationsgeschwindigkeit beschrieben werden und damit dann mit optimierten Algorithmen berechnet werden können. Damit lässt sich für Systemebenenbetrachtungen eine Beschleunigung der Simulation um mehrere Größenordnungen im Vergleich zu konventionellen Verhaltensbeschreibungssprachen erreichen. Dadurch wird eine funktionale Gesamtsystems simulation von komplexen Anwendungsszenarien möglich. Für SystemC und SystemC-AMS wird eine Open Source „proof-of-concept“ Implementierung zur Verfügung gestellt, wobei die Lizenzbedingungen so gestaltet sind, dass eine kommerzielle Nutzung möglich ist. Zusätzlich stehen zahlreiche kommerzielle Implementierungen zur Verfügung, welche sich an der Open Source Implementierung orientieren bzw. diese für die entsprechenden Werkzeuge anpassen um z.B. komfortable Debugmöglichkeiten oder die Integration in andere Simulatoren (z.B. VHDL oder Verilog) zu ermöglichen. Dadurch, dass SystemC und SystemC-AMS auf der weit verbreiteten Programmiersprache C++ beruhen, ergeben sich zahlreiche Möglichkeiten, die unter anderem für die Anwendung in der Automobilelektronik interessant sind. Im folgenden Abschnitt werden solche Anwendungsszenarien diskutiert.

3 Anwendungsszenarien

3.1 Ausführbare Spezifikation

Wie im Abschnitt 1 beschrieben, interagiert die elektronische Hard- und Software immer enger mit z.B. den mechanischen Komponenten. Eine unabhängige Spezifikation von Hard- und Software, bzw. Elektronik und nicht Elektronik ist immer schwerer umzusetzen. Die traditionelle Papier-basierte Spezifikation wird immer schwerer erstellbar, sie ist mehr und mehr unvollständig bzw. enthält Fehler welche oft erst während der Systemintegration feststellbar sind. Eine ausführbare Spezifikation ersetzt nicht die Papier basierte Spezifikation, wir aber mehr und mehr ein unentbehrliches Hilfsmittel zur Erstellung dieser. Sie erlaubt es das Zusammenspiel der Elektronik mit den anderen Komponenten schon im Spezifikationsprozess zu verifizieren. Damit lassen sich viele teure Spezifikationsfehler vermeiden und Missverständnisse wie sie z.B. zwischen unterschiedlichen Fachgebieten auf Grund unterschiedlicher Terminologie oder Definitionen immer wieder auftreten rechtzeitig ausräumen.

Damit eine ausführbare Spezifikation erfolgreich verwendet werden kann, muss es mit ihr möglich sein Anwendungsszenarien mit der Umgebung bzw. mit vereinfachten

Umgebungsmodellen in akzeptabler Zeit zu simulieren. Dabei muss die Funktionalität und das Zeitverhalten soweit es die Funktion beeinflusst korrekt abgebildet werden, dagegen ist die Berücksichtigung von parasitären Effekten im Allgemeinen nicht erforderlich. Da die Spezifikation als Kommunikationsplattform zwischen unterschiedlichen Gruppen genutzt werden soll, muss sie von diesen auch ohne spezialisierte Werkzeuge ausführbar bzw. in deren Werkzeuge integrierbar sein.

3.2 Architekturentwurf

Ähnlich den Anforderungen zur ausführbaren Spezifikation, müssen auch beim Architekturentwurf möglichst komplette Anwendungsszenarien simuliert werden um unterschiedliche Architekturvarianten bewerten und somit optimieren zu können. Zusätzlich muss es möglich sein, das Modell punktuell zu verfeinern um z.B. die Überlastung von Bussystemen oder die Verletzung von Echtzeitbedingungen zu detektieren.

3.3 Entwicklung eingebetteter Software

Fast alle aktuellen integrierten Komponenten enthalten mehr oder weniger komplexe Prozessoren und Controller. Diese übernehmen Aufgaben wie z.B. die Ablaufsteuerung, Kalibrierung, Regelung bis hin zur intelligenten Verarbeitung von Daten bzw. der Ansteuerung von Aktoren. Da die erforderliche eingebettete Software sehr eng mit der Hardware interagiert ist eine Entwicklung unabhängig von dieser nicht möglich. Da zudem die Softwareentwicklung einen immer erheblicher werdenden Zeitaufwand beansprucht, ist ein Start dieser nach Verfügbarkeit der Hardware inakzeptabel. Zusätzlich ist die Beobachtbarkeit interner Knoten in der Regel nicht ohne weiteres möglich. Aus diesen Gründen bringt der Softwareentwurf mit Hilfe einer virtuellen Plattform eine Reihe von Vorteilen. Voraussetzung dafür ist, dass das Modell schnell genug simuliert und sich in die Softwareentwurfsumgebung einbinden und debuggen lässt. Wobei für den Softwareentwurf bis auf den Prozessor häufig sehr vereinfachte Modelle verwendet werden können. Allerdings sind dafür die Anforderungen an die Simulationsgeschwindigkeit sehr hoch, da Softwareentwickler eine interaktive Arbeitsweise erwarten.

3.4 Referenzmodell

Bei der Implementierung von integrierten elektronischen Komponenten ist die Verifikation eine große Herausforderung. Zum einen ist es schwer relevante Stimuli für die Subkomponenten zu erzeugen und zum anderen ist es nicht einfach die Antwort der Komponente zu bewerten. Damit gestaltet sich speziell der Aufbau von Regressionstests, wie sie zur Sicherstellung der Entwurfsqualität dringend erforderlich sind sehr schwierig. Ein Ausweg, ist ein Referenzmodell, wie es z.B. eine ausführbare Spezifikation oder das Architekturmodell darstellt. Aktuelle Verifikationsmethoden wie z.B. UVM (Universal Verification Methodology) setzen das Vorhandensein einer Referenz voraus. Da die Modelle für die ausführbare Spezifikation schon während deren Erstellung überprüft wurden, ist es wünschenswert diese als auch die Verifikationsszenarien zur Stimulierung als auch als Referenz für die Komponentenimplementierung weiter zu verwenden. Dazu muss es möglich sein, diese Modelle in die entsprechenden Entwurfswerkzeuge und Simulatoren einzubinden.

3.5 Kundenmodell

Typisch für Systeme der Automobiltechnik ist, dass die Einzelkomponenten von verschiedenen Zulieferern entwickelt werden. So geht die typische Wertschöpfungskette vom Halbleiterhersteller (TIER2) zum Komponentenhersteller (TIER1) bis zum Automobilhersteller (OEM). Für die gemeinsame Erarbeitung der Spezifikation wird es immer wichtiger, Modelle auszutauschen und diese im jeweiligen Gesamtsystemkontext Anwendungsszenarien zu verifizieren. Damit ist es möglich virtuell eine Systemintegration vor dem festschreiben der Spezifikation und somit weit vor der Verfügbarkeit der Hardware vorzunehmen. Da ausreichend detaillierte Modelle in der Regel immer Implementierungsdetails und

Algorithmen enthalten, welche zur „Intellectual Property – IP“ zählen, ist aus kommerzieller Sicht, ein Modellaustausch zwischen unterschiedlichen Unternehmen nur akzeptabel, wenn die innere Struktur und die Algorithmen entsprechend geschützt sind, d.h. das Modell nicht mehr über den inneren Aufbau preisgibt, als es die später verfügbare Hardware tut. D.h., das Modell muss für den Kunden eine „Blackbox“ sein welches er nur wie die reale Hardware über die äußeren Klemmen ansteuern kann (bzw. bei dem er nur vom Zulieferer gewollte interne Knoten beobachten kann). Zusätzlich muss das Modell in die Simulationsumgebung des Kunden einbindbar sein und sollte keine zusätzlichen Lizenzen von anderen Simulationswerkzeugen erfordern.

4 Technische Lösungsansätze

4.1 Modellierungstechniken

Ein großer Vorteil von SystemC/SystemC-AMS ist die parallele Anwendbarkeit verschiedener Modellierungs- und Simulationstechniken und die Möglichkeit SystemC/SystemC-AMS um weitere zu erweitern. SystemC/SystemC-AMS ist in Schichten aufgebaut. SystemC hat ein sogenanntes generisches Model of Computation, d.h. aufbauend auf Basiselemente wie z.B. Prozesse und Ereignisse sind unterschiedliche Berechnungsmodelle und darauf aufbauend verschiedene Modellierungstechniken definiert. Eine wichtige Technik ist die sogenannte TLM (Transaction Level Modeling) [5] Modellierung. Diese wird zur Modellierung von digitaler Kommunikation – z.B. der Kommunikation eines Prozessors mit seinem Speicher oder Peripherie eingesetzt. Diese Kommunikation erfolgt in der Regel über ein Bussystem bestehend aus Daten-, Adressbus und Steuerleitungen. Klassisch wird dieses digitale Verhalten mittels ereignisgesteuerter Modellierung/Simulation (was eigentlich auch schon eine starke Abstraktion des eigentlich analogen elektrischen Verhaltens darstellt) abgebildet. Wenn dabei z.B. ein Zugriff des Prozessors auf seinen Speicher erfolgt, werden taktgesteuert die entsprechenden Steuer- Adress- und Datensignale angelegt. Während der ereignisgesteuerten Simulation wird für jeden Signalwechsel ein Ereignis erzeugt und bei jeder Taktflanke erfolgt ein sogenannter Kontextswitch, d.h. ein Wechsel von einem zu einem anderen. Bei der TLM Modellierung wird ein solcher Buszugriff auf einen Funktionsaufruf abgebildet – welcher einen Wert vom Speicher liest bzw. schreibt, d.h. den Wert einer Variable im Speichermodell liest bzw. setzt. Da solch einem Funktionsaufruf eine Zeit annotiert werden kann, lässt sich das Verhalten auch zeitlich sehr genau an das ereignisgesteuerte Modell anpassen. Da rechentechnisch die Realisierung eines Funktionsaufrufs um Größenordnungen weniger aufwendig ist, sind durch TLM Modellierung je nach Detaillierungsgrad Simulationsgeschwindigkeitsgewinne um den Faktor 10.000 möglich. Dabei ist, wenn das zeitliche Verhalten nur ungefähr abgebildet werden muss, der Modellierungsaufwand wesentlich geringer.

Ein ähnliches Konzept verfolgt SystemC-AMS zur Modellierung von analogem Verhalten. SystemC-AMS erweitert SystemC um Berechnungsmodelle, welche speziell zur abstrakten Modellierung von analog und gemischt analog-digitalen (mixed-signal) Verhalten geeignet sind. Dabei spielt die Datenflussmodellierung eine besondere Rolle. SystemC-AMS enthält dazu das sogenannte Timed Dataflow (TDF) Berechnungsmodell. Ähnlich wie bei TLM erlaubt dieses Berechnungsmodell die Ausnutzung der abstrakten Modellierung zur Steigerung der Simulationsgeschwindigkeit um Größenordnungen. Bei der Datenflusssimulation werden Sample vom Eingang gelesen prozessiert und die Ergebnissample auf den Ausgang geschrieben. Damit lässt sich sehr einfach z.B. das Verhalten analoger Signalverarbeitung bestehend aus Verstärker, Filter und A/D-Wandler abbilden. Ein Verstärker ist z.B. im einfachsten Fall die Multiplikation des Eingangswert mit einem Faktor und evtl. einer Begrenzung des Ausgangswertes. Der Ausgangswert wird dann z.B. im Filter weiterverarbeitet, wobei die Rückwirkung für die funktionale Modellierung vernachlässigbar ist. Solch abstraktes Verhalten lässt sich auch mit klassischen

Verhaltensbeschreibungssprachen wie Verilog-AMS/VHDL-AMS beschreiben – da diese dann aber auf einen allgemeinen nichtlinearen Gleichungslöser abgebildet werden müssen, ist die Simulation um Größenordnungen langsamer als eine Datenflusssimulation. Somit werden die Modellierungsmöglichkeiten soweit wie möglich eingeschränkt, um das Verhalten auf einen möglichst effizienten Berechnungsalgorithmus abbilden zu können. Durch diese Vorgehensweise ist eine sehr hohe Simulationsgeschwindigkeit erreichbar, welche um Größenordnungen über der von klassischen Werkzeugen liegen kann.

4.2 Softwaremodellierung

Ein großer Vorteil von SystemC ist die homogene Integration und die Möglichkeit Software auf verschiedenen Abstraktionsebenen zu modellieren. So kann Software über ein Prozessormodell – z.B. einem sogenannten Instruktionsetsimulator als Binary eingebunden werden. Speziell zum Entwurf der Algorithmen kann die Software auch direkt in das Modell kompiliert werden. Da SystemC auf C++ beruht, ist das ganz einfach möglich. Dabei können dann Zugriffe auf externe Komponenten auf TLM Transaktionen abgebildet werden, wodurch die Software dann mit dem Hardwaremodell kommuniziert. Um das zeitliche Verhalten annähernd abzubilden, kann einzelnen Codeabschnitten eine Zeit annotiert werden. Die Vorteile dieses Ansatzes sind die einfache Debugbarkeit und die extrem schnelle Simulation. Da das Softwaremodell über eine TLM Schnittstelle mit der Hardware kommuniziert, ist es möglich dieses durch den oben beschriebenen Instruktionsetsimulator zu ersetzen, wodurch dann das Zeitverhalten taktgenau abgebildet wird.

4.3 Werkzeugintegration

Viele Simulationswerkzeuge erlauben es Modelle über C-Schnittstellen einzubinden. Da SystemC/SystemC-AMS auf C++ (was ein Superset von C ist) beruht ist eine Einbindung einfach möglich. Dies ist speziell sehr einfach möglich, da eine Open Source und somit lizenzfreie Implementierung für SystemC und SystemC-AMS zur Verfügung steht. Die Einbindung erfolgt mit Hilfe einer Koppelbibliothek, welche die Synchronisation der Zeitachsen und die Konvertierung der Datentypen realisiert. Die SystemC/SystemC-AMS Modelle werden dann zusammen mit der Koppelbibliothek je nach Betriebssystem zu einem shared object oder einer dll gelinkt, welche dann dynamisch von dem entsprechenden Simulationswerkzeug geladen werden kann. Damit kann das Modell vorkompiliert weitergeben werden, womit der oben beschriebene IP-Schutz realisiert werden kann.

4.4 Hardware in the Loop Simulation

Wie bereits beschrieben, lässt sich mit SystemC/SystemC-AMS die Simulationsgeschwindigkeit sehr gut optimieren. Somit ist es für bestimmte Modelle sogar möglich, diese in Echtzeit zu simulieren. Somit werden sogenannte Hardware in the Loop Simulationen möglich [6]. Damit wird die sich im Entwurf befindliche Systemkomponente mit bereits existierender Hardware – z.B. der Systemumgebung verbunden. Somit kann das Systemverhalten schon in Echtzeit vor Verfügbarkeit der Hardware und auch schon in der Spezifikationsphase getestet werden. Es existieren verschiedene Hardwareplattformen zur Durchführung solcher Simulationen. So bietet z.B. die Firma dSpace Plattformen auf der Basis von Intel und PowerPC Prozessoren an. Die Programme für diese Plattformen werden mit nahezu vollständig C++ implementierten Compilern übersetzt. Da die Open Source Bibliotheken von SystemC/SystemC-AMS als Quelltext vorliegen, lassen diese sich so anpassen, dass sie für die Hardware in the Loop Plattformen compilierbar sind. Zusammen mit einer Koppelbibliothek, welche vor allem für die Synchronisation der simulierten Zeit mit der realen Zeit zuständig ist, lassen sich Modelle erzeugen, welche auf die Plattformen herunterladbar sind.

5 Werkzeugunterstützung

Der SystemC/SystemC-AMS Standard definiert nur die Klassen, Makros und Funktionen zur Beschreibung des Systemverhaltens. Die Open Sourcebibliotheken stellen eine Referenzimplementierung frei zur Verfügung. Damit ist es schon möglich Systeme zu beschreiben und zu simulieren. Allerdings passiert dies in diesem Fall nur textbasiert und ist somit sehr aufwendig und fehleranfällig. Um den Modellierungsprozess effizient zu gestalten, gibt es für SystemC und inzwischen auch für SystemC-AMS Werkzeuge z.B. [7,8], die z.B. eine grafische Eingabe (Schaltplaneditor), die Modellgenerierung und die Generierung von Testumgebungen erlauben. Außerdem enthalten solche Werkzeuge auch komfortable Debugmöglichkeiten für die verschiedenen Modellierungsmöglichkeiten (z.B. Debuggen von TLM Transaktionen, Waveform viewer mit leistungsfähigem Postprocessing). Solche Werkzeuge liefern auch Bibliotheken mit vorgefertigten Modellen, welche dann mit Hilfe des Schaltplaneditors verbunden werden können. Auch die Bibliotheken welche für die oben erwähnten Möglichkeiten wie die Werkzeugintegration und Hardware in the Loop Simulation notwendig sind werden zur Verfügung gestellt. Trotzdem generieren diese Werkzeuge letztlich C++ Quelltexte bzw. stellen C++ Bibliotheken zur Verfügung, so dass auch bei deren Verwendung die oben beschriebenen Möglichkeiten bis auf wenige Ausnahmen nicht eingeschränkt werden. Somit machen diese Werkzeuge die komplexen Modellierungs- und Simulationsmöglichkeiten auch für nicht Spezialisten nutzbar.

6 Zusammenfassung

Es wurde eine Modellierungs- und Simulationsmethodik vorgestellt, welche Lösungsansätze für Probleme welche bei der Spezifikation und dem Entwurf elektronischer Hard-Softwarekomponenten der Automobiltechnik bestehen, bietet. Da immer mehr die Notwendigkeit besteht, dass Spezifikation und Entwurf disziplin- und firmenübergreifend erfolgen muss bietet die auf SystemC/SystemC-AMS basierende Methodik interessante Möglichkeiten. Dies sind z.B. angepasste Modellierungstechniken welche es erlauben Modelle so zu optimieren, dass eine Gesamtsystemsimulation kompletter Anwendungsszenarien möglich wird, die Einbindung von Modellen in verschiedene Simulationswerkzeuge, ein IP geschützter Modellaustausch, Hardwaremodelle für die Softwareentwicklung und die Weiternutzung der Modelle für die Echtzeit Hardware in the Loop Simulation. Um die recht komplexe Methodik auch für nicht Spezialisten anwendbar zu machen stehen inzwischen komfortable Werkzeuge zur Verfügung.

Literatur

- [1] "IEEE Standard for Standard SystemC Language Reference Manual." IEEE Std 1666 2011 (Revision of IEEE Std 1666 2005) (2012): 1–638.
- [2] "Standard SystemC AMS extensions Language Reference Manual", Open SystemC Initiative, March 8 2010
- [3] "Standard SystemC AMS extensions User's Guide", Open SystemC Initiative, March 8 2010
- [4] Einwich, K. "SystemC AMS for the design of complex analog mixed signal SoC's: Presentation held at edaWorkshop 2009, Dresden, Germany, May 26-28, 2009". 2009.
- [5] Ghenassia, Frank. "Transaction Level Modeling with SystemC: TLM Concepts and Applications for Embedded Systems." Boston, MA, 2005.
- [6] Voit, Florian, "Echtzeitsimulation von Electronic-System-Level-Modellen", ASIM 2011
- [7] www.systemc-ams.eas.iis.fraunhofer.de
- [8] <http://www.synopsys.com/Systems/ArchitectureDesign/pages/PlatformArchitect.aspx>

Parametrierung von Batteriesimulationsmodellen mithilfe eines geeigneten Prüfstands

Florian Quantmeyer, Xiaobo Liu-Henke, Waldemar Diehl, Sascha Bode

Ostfalia – Hochschule für angewandte Wissenschaften

Institut für Mechatronik

{Flo.Quantmeyer, X.Liu-Henke, W.Diehl, Sa.Bode}@Ostfalia.de

GEFÖRDERT VOM



Bundesministerium
für Bildung
und Forschung

Förderkennzeichen: 17N0911

Zusammenfassung

Für die modellbasierte Entwicklung von Softwarealgorithmen für das Batteriemanagementsystem von Hybrid- und Elektrofahrzeugen werden verschiedene Batteriemodelle vorgestellt. Es folgt die Darstellung eines Prüfstands der die Ermittlung der erforderlichen Parameter eines ausgewählten Modells auf Zellebene ermöglicht. Mittels Durchführung eines geeigneten Testzyklus wird das Modellverhalten mit dem Verhalten des realen Systems verglichen.

1 Einleitung

Im Rahmen des BMBF-geförderten Forschungsvorhabens „Electronic Vehicle Management“ wird zurzeit ein globales Fahrzeugmanagement für Elektrofahrzeuge entwickelt. Die wesentlichen Teilsysteme dieses Fahrzeugmanagements sind das Antriebs-, Batterie- und Fahrwerkmanagement. Es zeichnet sich durch einen hohen Vernetzungsgrad der Softwarefunktionen der Teilsysteme aus, wodurch eine ungewollte negative Beeinflussung der Funktionen vermieden und Synergieeffekte genutzt werden sollen.

Die resultierende hohe Systemkomplexität stellt hohe Anforderungen an den Entwicklungsprozess. Zur Beherrschung der Komplexität hat sich die mechatronische Entwicklungsmethodologie mit ihrem durchgängig-modellbasierten und verifikationsorientierten Charakter bewährt [1].

Für die modellbasierte Auslegung von Funktionen für das Batteriesystem sowie die modellbasierte Absicherung verschiedener Funktionen mittels Gesamtfahrzeugsimulation

werden geeignete Batteriemodelle benötigt, die das Batterieverhalten mit einer ausreichenden Genauigkeit beschreiben. Aus dem stark nichtlinearen Verhalten der Batterie resultiert die Notwendigkeit eines umfangreichen Parametersatzes zur Bedatung akkurater Modelle. Dieser Beitrag fokussiert die Vorgehensweise zur Gewinnung dieses Parametersatzes mithilfe eines Prüfstands, der die automatische Durchführung der zeitaufwändigen Versuche gewährleistet.

2 Batteriemodelle

Es wurde bereits eine große Anzahl von Batteriemodellen entwickelt, die sich anhand ihres Aufbaus, der Detaillierung und der Parametrierung voneinander unterscheiden. Sie lassen sich in die vier Kategorien: Empirische-, Elektrochemische-, Stromkreis- und Abstrakte Modelle unterteilen [2].

In empirischen Modellen wird das Realverhalten der Batterien, das durch Messungen aufgezeigt wird, mit geeigneten mathematischen Gleichungen und empirischen Parametern angenähert. Handelt es sich dabei um konstante Parameter, bilden sie das dynamische Verhalten der Lithium-Ionen-Batterie, das stark nichtlinear ist nur unzureichend genau ab.

Abstrakte Modelle basieren auf reinen mathematischen (z.B. stochastischen) Methoden, die das Batterieverhalten beschreiben. Derartige Modelle sind praktisch nahezu bedeutungslos, da in ihnen nur einzelne Eigenschaften der Batterie wiedergegeben werden [2].

Elektrochemische Modelle weisen die höchste Modellierungstiefe auf. Die Abläufe im Inneren des Akkumulators werden in diesen Modellen abgebildet. Derartige Modelle sind für die Entwicklung von Akkumulatoren geeignet. Trotz des höchsten Detaillierungsgrades weist es einige wesentliche Nachteile für die Batteriesimulation innerhalb von Elektrofahrzeugen auf. Für die Entwicklung solcher Modelle ist ein hohes Know-How im Bereich Elektrochemie erforderlich. Für die Parametrierung ist eine hohe Anzahl von Parametern und damit verbunden eine Vielzahl von Messungen erforderlich. Des Weiteren sind die Berechnungen sehr aufwändig (gekoppelte zeitvariante partielle Differentialgleichungen), wodurch sich sehr hohe Rechenzeiten einstellen (nicht echtzeitfähig).

Neue Ansätze zur Vereinfachung dieser Modelle und damit zur Erreichung der Echtzeitfähigkeit für den Einsatz in Regelungsanwendungen werden in [3] und [4] beschrieben.

Im Rahmen dieses Vorhabens werden die Batteriemodelle dazu genutzt das Verhalten von Batterien innerhalb komplexer Systeme wie Elektro- und Hybridfahrzeugen zu simulieren. Besonders im Hinblick auf den Einsatz im Rahmen von Hardware-in-the-Loop-Simulation stellt die Echtzeitfähigkeit des Gesamtsimulationsmodells eine wesentliche Anforderung dar. Batteriemodelle, bei denen das Verhalten der Batterie durch elektrische Schaltkreise beschrieben wird eignen sich für diesen Einsatz besonders, da sie trotz geringem Rechenaufwand eine ausreichend hohe Genauigkeit bieten. Detailtiefe und Rechenaufwand dieser Modelle können durch Modifikation des elektrischen Schaltkreises (0...n RC-Glieder) variiert werden, wodurch sie auf ihren Einsatzzweck zugeschnitten werden können.

Abbildung 1 zeigt ein Ersatzschaltbildmodell der Batteriezelle. Die Leerlaufspannung (auch OCV: Open Circuit Voltage) der Zelle wird von einer Spannungsquelle bereitgestellt. Der ohmsche Widerstand R_S beschreibt den Innenwiderstand der Batteriezelle und das RC-Glied bestehend aus R_D und C_D charakterisiert das dynamische Verhalten der Batterieelektroden.

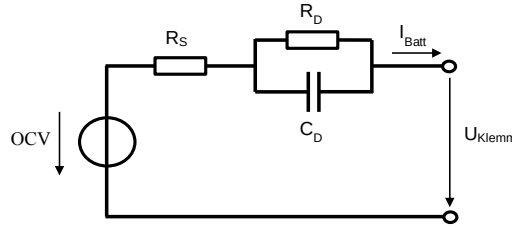


Abbildung 1: Ersatzschaltbild der Batteriezelle nach [5], [6]

Neben diesem beispielhaften Ersatzschaltbildmodell werden in der Literatur einige weitere Modelle beschrieben. [7] und [8] erhöhen beispielsweise die Zuverlässigkeit des Modells durch die Steigerung der Anzahl von RC-Gliedern. Für nicht sehr dynamische Anwendungen sind auch Ersatzschaltmodelle geeignet, die gar kein RC-Glied besitzen.

Das in Abbildung 1 dargestellte Modell unter den Ersatzschaltbildmodellen einen guten Kompromiss dar. Es ist in der Lage die Dynamik des Akkus abzubilden und dennoch ist sowohl der Rechenaufwand als auch der Aufwand zur Parameterermittlung relativ gering.

Zur Implementierung des Modells in die CAE-Umgebung wird die mathematische Beschreibung des Systems mithilfe der Kirch'hooffschen Gesetze gewonnen.

$$U_{Klemm} = OCV + R_S \cdot I_{Batt} + U_{RC} \quad (1)$$

$$I_{Batt} = \frac{U_{RC}}{R_D} + C_D \cdot \frac{dU_{RC}}{dt} \quad (2)$$

Aus (2) erhält man nach Laplace-Transformation folgende Übertragungsfunktion:

$$G(s) = \frac{U_{RC}(s)}{I_{Batt}(s)} = \frac{K}{Ts + 1} \quad (3)$$

$$\text{mit : } K = R_D \text{ und } T = R_D C_D$$

Nach Laplace-Transformation von (1) und einsetzen von (3) erhält man:

$$U_{Klemm}(s) = OCV(s) + \left(R_S + \frac{R_D}{R_D C_D s + 1} \right) \cdot I_{Batt}(s) \quad (4)$$

Die Leerlaufspannung als auch die Parameter R_S , R_D und C_D sind sowohl vom Ladezustand, der Stromstärke als auch von der Temperatur und dem Alterungszustand abhängig.

3 Beschreibung des Prüfstands zur Parametrierung

Für Durchführung der erforderlichen Versuche zur Parametrierung von Batteriemodellen wurde ein Prüfstand konzipiert (Abbildung 2), mit dem die Batteriezellen innerhalb ihres zulässigen Betriebsbereichs definiert geladen bzw. entladen werden können.

Parallel zur Batteriezelle sind ein programmierbares Netzgerät sowie eine elektronische Last geschaltet. Das programmierbare Netzgerät dient zum Laden der Batteriezelle. Es kann sowohl strom- ($I_{\max} = 100 \text{ A}$) und spannungs- ($U_{\max} = 80 \text{ V}$) als auch leistungsgeregelt ($P_{\max} = 3 \text{ kW}$) betrieben werden. Mittels elektronischer Last wird die Batteriezelle entladen. Die Last ermöglicht dieselben Regelbetriebe ($I_{\max} = 200 \text{ A}$, $U_{\max} = 80 \text{ V}$, $P_{\max} = 2,4 \text{ kW}$) wie das programmierbare Netzgerät und erlaubt außerdem einen Widerstandsregelbetrieb.

Mittels eines übergeordneten Rapid Control Prototyping Systems werden beide Geräte im Zwei-Quadranten-Betrieb angesteuert. Darüber hinaus dient das RCP-System zur Erfassung und Verarbeitung von Messwerten für Strom, Spannung und Temperatur.

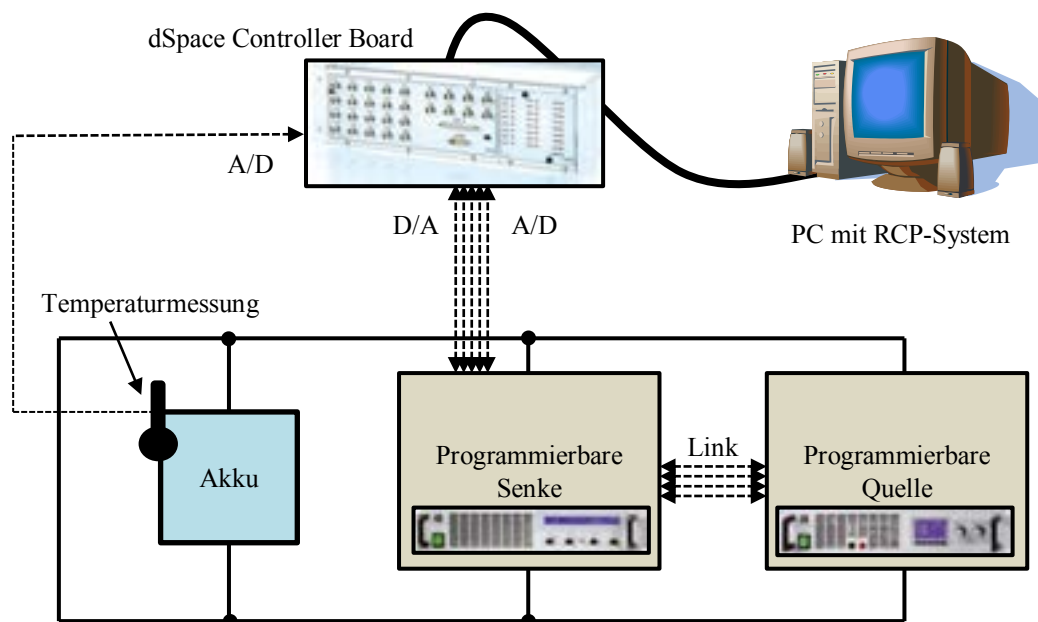


Abbildung 2: Aufbau Batterieprüfstand

Das RCP-System kommuniziert mit einem PC, der in Kombination mit der geeigneten Software die Nutzerschnittstelle darstellt (s. Abbildung 3). Über die Benutzerschnittstelle werden die zuvor definierten Testzyklen gewählt. Diese werden daraufhin automatisiert ausgeführt. Die Zustände der Batteriezelle werden während der Messung in Echtzeit dargestellt. Kritische Zustände der Batterie werden durch Sicherheitsmechanismen und Abschaltstrategien vermieden.

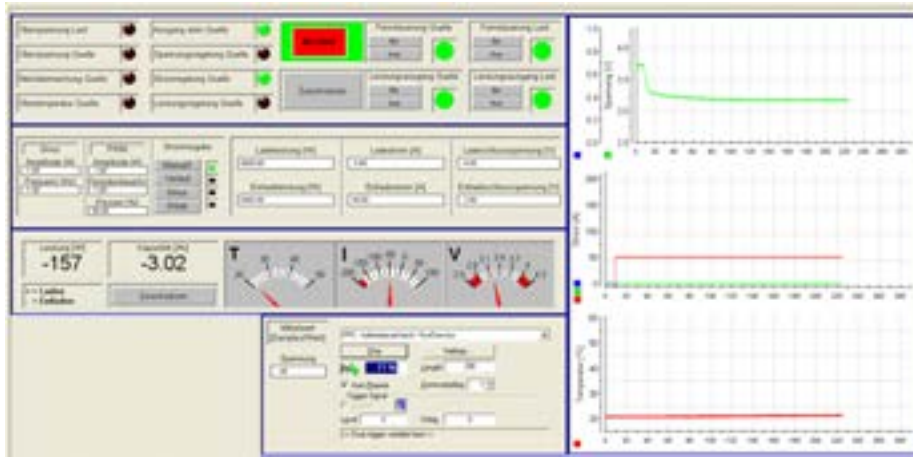


Abbildung 3: Bedienoberfläche zur Steuerung und Überwachung des Batterieprüfstands

4 Parametrierung des Batteriemodells

Bei der realen Batteriezelle handelt es sich um eine Lithium-Ionen-Hochenergiezelle mit dem Kathodenmaterial Lithiumeisenphosphat (LiFePO_4). Sie wird derzeit im Batteriesystem des maßstäblichen Forschungsfahrzeugs *M-Mobile* [9] eingesetzt, wo sie mit Strömen von bis zu 100 A entladen wird.

Spannungsbereich	2,8V ... 4,0V
Nennkapazität	60 Ah
Maximaler Entladestrom	180A
Maximaler Ladestrom	180A

Tabelle 1: Spezifikation der Batteriezelle

Für diese Batteriezelle sollen beispielhaft die Parameter ermittelt werden, mit denen das Verhalten des Modells der Batteriezelle (Kap. 2) das reale Verhalten möglichst genau widerspiegelt. Aus dieser Forderung ergibt sich eine Vielzahl notwendiger Versuche:

- Ermittlung des Innenwiderstands bei unterschiedlichen Ladeständen (SoC: State of Charge) der Zelle
- Aufnahme des Verlaufs der Leerlaufspannung über dem Ladestand
- Ermittlung der Dynamik bei unterschiedlichen Ladezuständen der Zelle

Mit der Betrachtung des Einflusses von Temperatur und Alterungserscheinungen der Batteriezelle vervielfacht sich die Anzahl der notwendigen Versuche. Da der Temperatureinfluss nicht im Fokus der derzeitigen Untersuchungen steht, wurden die Messungen bei konstanter Raumtemperatur von 20°C durchgeführt, wodurch deren Anzahl drastisch reduziert werden konnte. Dennoch ist die Anzahl der erforderlichen Messungen hoch, weshalb die Automatisierung sinnvoll ist.

Als erster Parameter für das Simulationsmodell wird der ohmsche Innenwiderstand ermittelt. Da der Innenwiderstand beim Laden und Entladen unterschiedlich ist, wird dieser für beide Betriebsarten des Akkus ermittelt. Dafür wird die Batterie bei je 10% SoC sprunghaft mit 20 A geladen bzw. entladen und der senkrechte Spannungsanstieg ermittelt (s. Abbildung 4). Anhand des Spannungsanstiegs und der Stromstärke lässt sich der Innenwiderstand nach Gleichung 5 berechnen.

$$R_i = \frac{\Delta U_{\text{Sprung}}}{\Delta I} \quad (5)$$

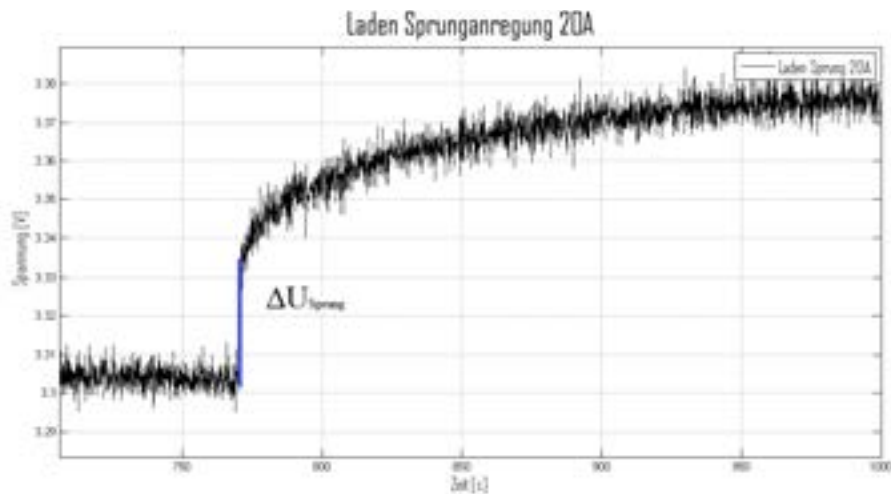


Abbildung 4: Spannungsänderung infolge eines Stromsprungs von 20 A

Für verschiedene Ladungsstände und Ströme wurde der Innenwiderstand der Zelle nach Abbildung 5 bestimmt.

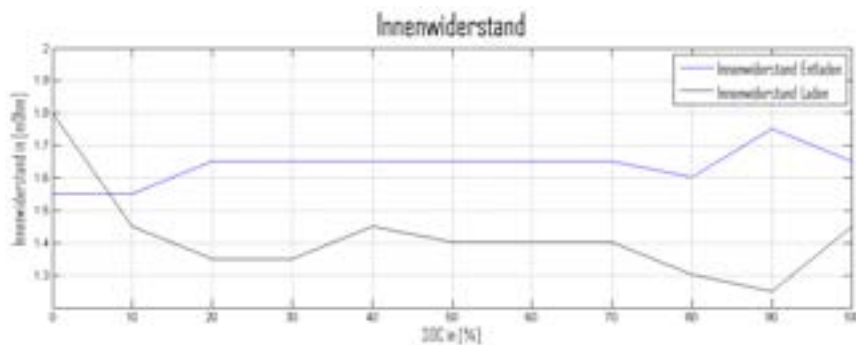


Abbildung 5: Innenwiderstand in Abhängigkeit des SoC Laden und Entladen

Nachdem der Innenwiderstand bestimmt wurde, wird die Leerlaufspannung (OCV) in Abhängigkeit des Ladestands (SoC) ermittelt. Damit die OCV-Kurve möglichst wenig von anderen Effekten beeinflusst wird, wird die Batterie mit einer kleinen Stromstärke von $1/20C^1$ entladen und anschließend geladen. Der dabei entstehende Spannungsverlauf beinhaltet den Spannungsabfall am Innenwiderstand und muss deshalb korrigiert werden. Abbildung 6 zeigt

¹ Entspricht einem Zwanzigstel der Nennkapazität, d.h. Lade- / Entladestrom von 3 A für die vorhandene Zelle

die korrigierten Spannungsverläufe Entladen und Laden. Sie weichen voneinander ab. Die Batterie weist ein Hystereseverhalten auf. In [6] wurde bei der Modellierung der Mittelwert aus Lade- und Entladekurve gebildet. Diese Vorgehensweise wurde hier zunächst auch verfolgt. Teilweise hohe Abweichungen führten jedoch zu dem Entschluss, die Hysterese mittels Kennlinien in das Simulationsmodell zu integrieren.

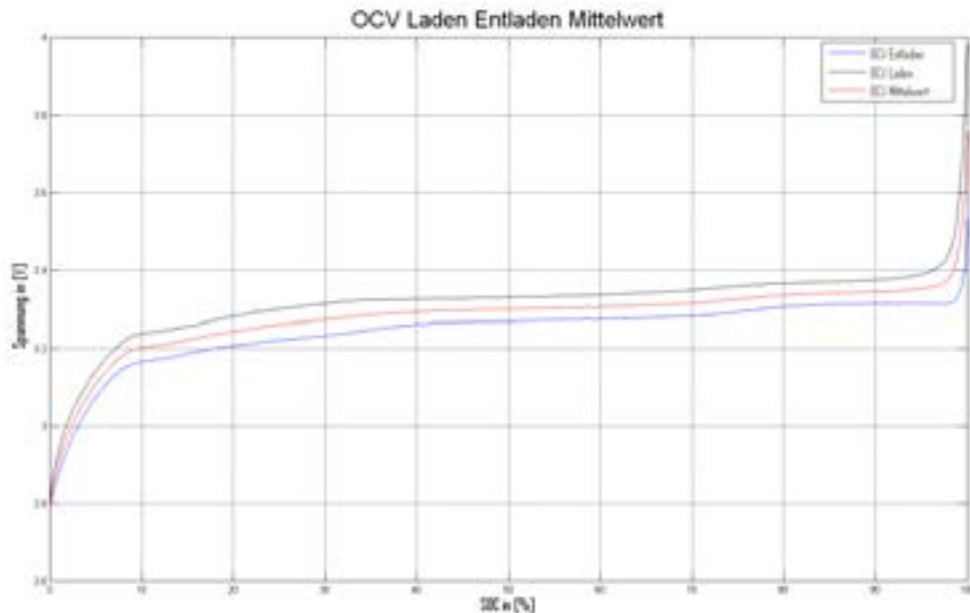


Abbildung 6: Verlauf der Spannung beim Entladen, Laden und Mittelwert über SoC

Im nächsten Schritt werden die Parameter des RC-Glieds ermittelt, indem die Zelle mit einem konstanten Strom von 20 A geladen wird. In Schritten von 10% Ladezustand wird eine Stromsprunganregung von 20 A auf 0 A durchgeführt, wobei die Klemmenspannung der Zelle aufgezeichnet wird. Das Übergangsverhalten der Spannung lässt sich in die Bereiche ΔU_{Sprung} und ΔU_{RC} unterteilen. ΔU_{Sprung} kann als die Spannung gedeutet werden, die zuvor über dem Innenwiderstand des Modells abgefallen ist. Diese Spannungsänderung findet unmittelbar statt. Anschließend sinkt die Klemmenspannung weiter und nähert sich exponentiell einer Spannung die nochmals um ΔU_{RC} geringer ist. Die Dynamik dieses zusätzlichen Spannungsabfalls wird durch das RC-Glied modelliert. Der Widerstand für das RC-Glied kann über das ohmsche Gesetz mit ΔU_{RC} und ΔI ermittelt werden. Die Kapazität des Kondensators wird nach Ablesen von τ gemäß Gleichung 6 bestimmt.

$$\tau = R_D \cdot C_D \quad (6)$$

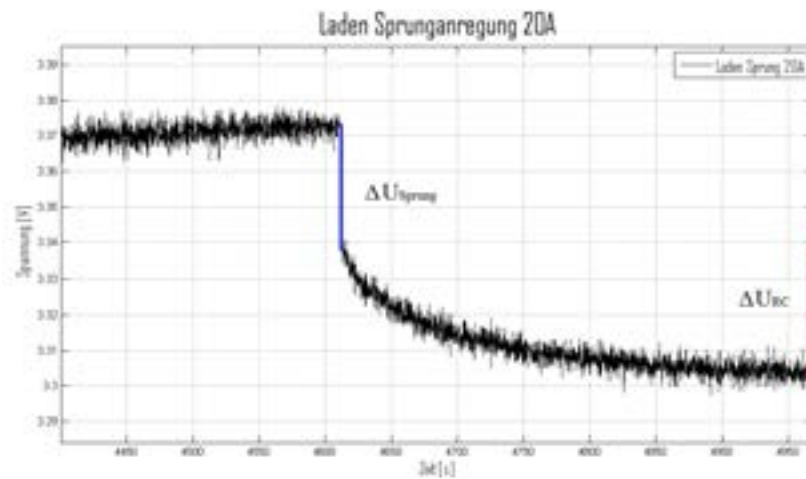


Abbildung 7: Spannungsantwort auf Stromsprung zur Ermittlung der Parameter R_D und C_D

Die Parameter des RC-Glieds werden in Schritten von 10% SoC ermittelt und in das Simulationsmodell eingebunden, siehe Abbildung 8.

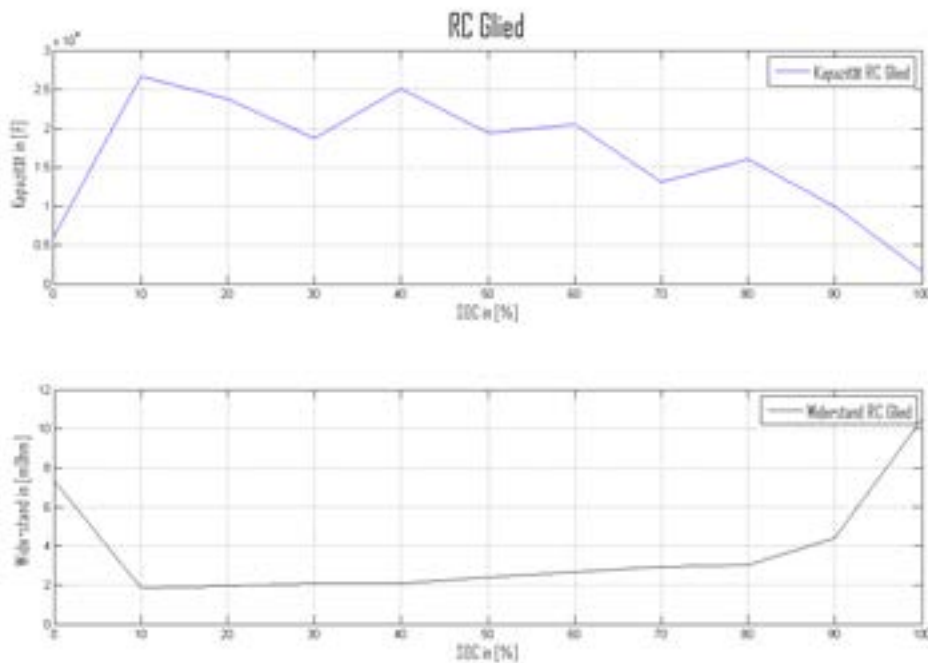


Abbildung 8: Parameter R_D und C_D in Abhängigkeit des Ladezustands (SoC)

5 Modellvalidierung anhand Zyklus

Das Simulationsmodell soll anhand eines definierten Testzyklus validiert werden. Dafür wird die reale Lithium-Ionen-Zelle aufgeladen und wiederholt mit dem Testzyklus belastet bis sie entladen ist. Das Simulationsmodell wird auf dieselbe Weise angeregt.

Als Testzyklus wurde der Dynamic Stress Test Power Cycle gewählt, siehe Abbildung. Der DST Power Cycle ist eine vereinfachte Version des USABC FUDS² Test Cycle und wurde speziell für den Test von Batterien unter Laborbedingungen für den Automobileinsatz

² USABC: United States Advanced Battery Consortium, FUDS: Federal Urban Driving Schedule

entwickelt. Der Batterieprüfstand wird dabei leistungsgeregelt betrieben. Ist die Leistung positiv, wird die Batterie geladen, ist sie negativ wird die Batterie entladen.

Der Testzyklus wird so lange wiederholt, bis die Entladeschlussspannung der Zelle erreicht ist. Die Spitzenleistung wird so gewählt, dass der relevante Betriebsbereich durch die Messungen abgedeckt wird.

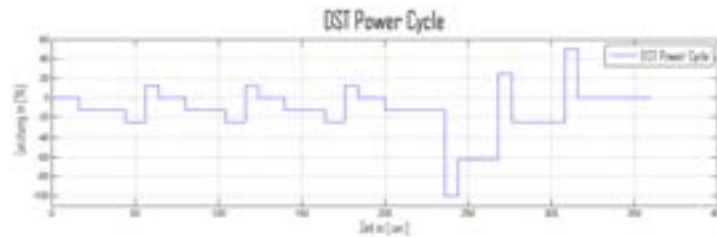


Abbildung 9: Dynamic Stress Test Power Cycle

Während des Versuchs werden Strom und Spannung aufgezeichnet. Der aufgezeichnete Stromverlauf wird anschließend dem Simulationsmodell vorgegeben und die simulierte Spannung mit der gemessenen realen Spannung verglichen, siehe Abbildung 10.

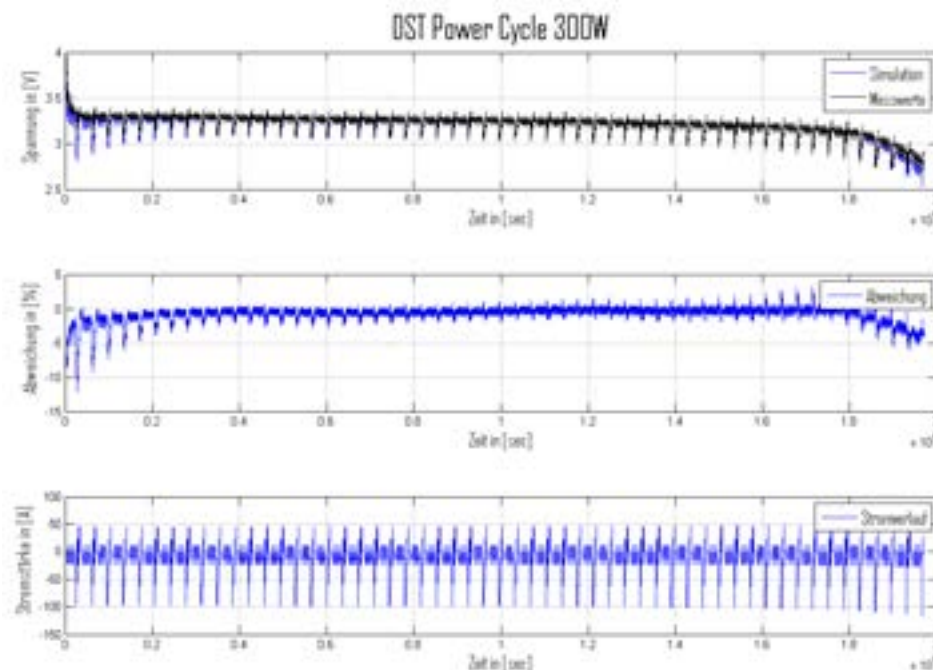


Abbildung 10: Vergleich Simulation und Messung anhand des DST Power Cycle

Die relative Abweichung zwischen Messung und Simulation ist bei geringem und hohem Ladungsstand am größten und beträgt in diesen Bereichen bis zu 12%. Dieser Effekt ist durch die hohen Parameteränderungen in diesen Bereichen erklärbar. Im mittleren Bereich des SoC sind die Abweichungen von Simulation und realem System hingegen gering.

In Abbildung 11 ist ein Zyklus aus dem Vergleich aus Abbildung 10 aus dem mittleren Bereich des SoC im Detail dargestellt. Insgesamt ist auch hier eine hohe Übereinstimmung zwischen Messung und Simulation erkennbar. Lediglich nach dem Wechsel vom Laden auf Entladen und umgekehrt treten nennenswerte Abweichungen auf.

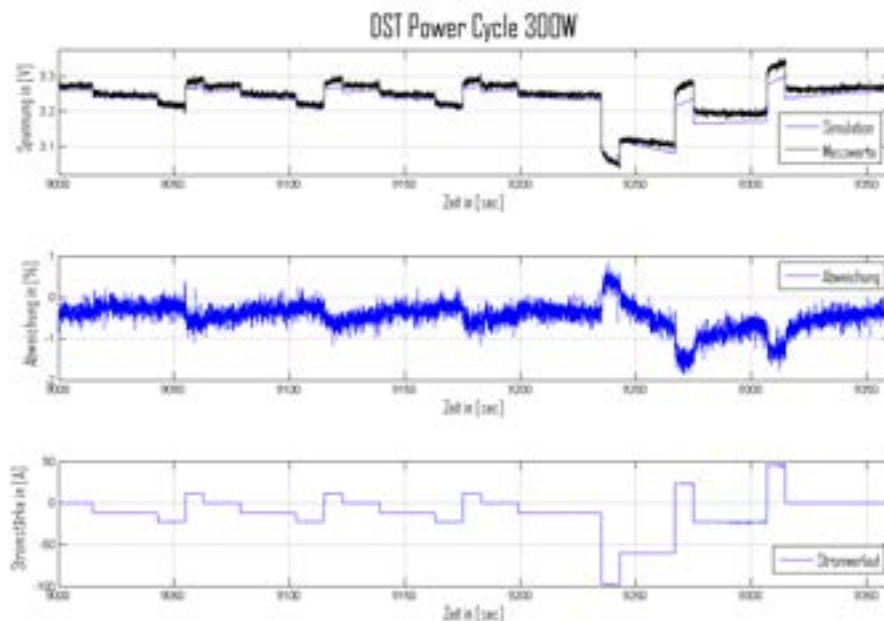


Abbildung 11: Detailansicht DST Power Cycle

Der Vergleich zwischen Simulation und Messung zeigt, dass das parametrierte Simulationsmodell das Verhalten der realen Batteriezelle im Arbeitsbereich (10% ... 90% des Ladezustand) sehr detailliert abbildet. Der Ausschnitt aus Abbildung 11 zeigt jedoch, dass besonders das transiente Verhalten noch Potential zur Steigerung der Genauigkeit durch eine Steigerung der Detailtiefe bietet.

6 Zusammenfassung und Ausblick

In diesem Beitrag wurde die Modellierung der Lithium-Ionen-Batterie auf Zellebene beschrieben. Nach der Vorstellung verschiedener Verfahren zur Modellierung des technischen Systems Batteriezelle, fiel die Auswahl auf die Modellierung mittels eines elektrischen Ersatzschaltbildes. Weiterhin wurde ein Prüfstand konzipiert, der es ermöglicht die Parameter für dieses Modell durch automatisierte Messungen zu ermitteln. Mithilfe dieses Prüfstands wurden die Parameter, die von verschiedenen Größen abhängig sind, am Beispiel eines Ersatzschaltbildmodells ermittelt. Zur Validierung dieses Modells wurde die reale Batteriezelle mittels eines wiederholten Testzyklus entladen. Durch einen Vergleich von Messung und Simulation konnte eine ausreichende Genauigkeit des Modells im Betriebsbereich der Batterie festgestellt werden.

Das Batteriemodell soll zukünftig für die modellbasierte Auslegung und den Test von Algorithmen für das Batteriemanagementsystem eingesetzt werden. Für spezielle Untersuchungen sind noch Folgearbeiten geplant, wie beispielsweise:

- Berücksichtigung des Temperaturverhaltens und Einfluss der Temperatur auf das Verhalten der Batterie

- Erweiterung des Batteriemodells um zusätzliche RC-Glieder zur Verbesserung der Genauigkeit des transienten Verhaltens
- Durchführung der Parameterermittlung an verschiedenen Batteriezellen desselben Typs zur Berücksichtigung der Parameterstreuung einzelner Zellen bei Simulation der Gesamtbatterie

Literatur

- [1] Liu-Henke, X.; Duym, S.: *Modellgestützte Funktionsabsicherung des vernetzten mechatronischen Kraftfahrzeugs*, VDI-Tagung Mechatronik 01./02.06.2005, Wiesloch, 2005
- [2] Tsang, K.M.; Chan, W.L.; Wong, Y.K.: *Lithium-ion Battery Models for Computer Simulation*, Proceedings of the 2010 IEEE International Conference on Automation and Logistics, August 16-20 2010, Hong Kong and Macau
- [3] Smith, K. A.; Rahn, C. D.; Wang, C. Y.: *Control oriented 1D electrochemical model of lithium ion battery*, Energy Conversion and Management 48, S. 2565-2578, Elsevier Verlag, 2007
- [4] Smith, K. A.: *Electrochemical Control of Lithium-Ion Batteries*, IEEE Control Systems Magazine, April 2010
- [5] Gao, L.; Shengyi, L.; Dougal, R. A.: *Dynamic Lithium-Ion Battery Model for System Simulation*, IEEE Transactions on Components and Packaging Technologies, Vol. 25, No. 3, September 2002
- [6] Einhorn, M.; Conte, V. F.; Kral, C.; Fleig, J.; Permann, R.: *Parameterization of an Electrical Battery Model for Dynamic System Simulation in Electric Vehicles*, Vehicle Power and Propulsion Conference (VPPC), September 01-03 2010, Lille
- [7] Chenglin, L.; Huiju, L.; Lifan, W.: *A Dynamic Equivalent Circuit Model of LiFePO₄ Cathode Material for Lithium Ion Batteries on Hybrid Electric Vehicles*, Vehicle Power and Propulsion Conference (VPPC), September 07-10 2009, Dearborn, Michigan, USA

- [8] Kroeze, R. C.; Krein, P. T.: *Electrical Battery Model for Use in Dynamic Electric Vehicle Simulation*, Power Electronics Specialists Conference (PESC), June 15-19 2008
- [9] Liu-Henke, X.; Buchta, R.; Quantmeyer, F.: *Simulation eines mechatronischen Lenkungsmoduls für ein Elektrofahrzeug mit dezentralen Direktantrieben*, ASIM-Konferenz STS/GMMS, Krefeld, 2011

Virtual-Platform in the Loop Simulation for Accurate Timing Analysis of Embedded Software on Multicore Platforms

Maher Fakh, OFFIS – Institute for Information Technology

Maher.Fakh@offis.de

Kim Grüttner, OFFIS – Institute for Information Technology

Kim.Gruettner@offis.de

Abstract

The design of embedded systems with real time requirements is a challenging task. On one side, timing predictability is fundamental for guaranteeing safe system operation. On the other side, complex functional behavior needs to be validated at all refinement levels during the design. For multicore platforms this task becomes even more challenging due to the increased complexity in platform parallelism including access arbitration to shared resources such as memories or peripherals, which impact software execution times.

This paper describes a co-simulation based validation method for embedded software implemented on multicore hardware platforms. The co-simulation is realized between Simulink and the SystemC-based SoCLib virtual-platform framework. Simulink is used to implement the system environment and functional model of an embedded control system. SoCLib is used to model a multicore execution platform with shared resources. Our design flow enables code generation and deployment from a Simulink model, and execution of this code on a multicore platform. In addition our virtual-platform model allows the observation of software execution and its timing measurement at a cycle accurate level.

We demonstrate the applicability of our method through validation of a real-time critical ignition controller system by running our virtual multicore platform in the loop with the Simulink environmental model.

1 Introduction

Multicores are strongly emerging in the area of embedded systems. Due to their significantly increased performance and decreased energy consumption, they offer an appealing alternative to traditional architectures. For this reasons we should expect platforms comprising hundreds of cores in the near future. This fact together with the growing computational demand of real-time applications (in automotive/avionics/multimedia) stresses the need for methods to evaluate the performance and the functionality of software applications running on such architectures.

Typically a system engineer starts modeling at a high level of abstraction to capture the functionality using a certain modeling framework (such as Matlab/Simulink). For instance, in case of a control system, the model consists of the controller and the environment (or the process to be controlled) without considering any architectural properties of the target hardware platform. This has the advantage that unnecessary platform details can be omitted for exploration of different control algorithms and a fast functional verification and validation (V&V). At this entry level timing properties can be defined, e.g. through sampling times. Yet this approach is suitable at the design entry but can lead to severe problems and expensive re-designs in the late development process if no guided evaluation of non-functional properties, such as timing and power consumption, of the available target platforms is supported. Today virtual-platforms are state of the art in embedded software design as they enable the V&V of the design's functional and non-functional requirements in early design phases. SystemC [2] based virtual-platform frameworks such as SoCLib [3] have been proven to be efficient in terms of simulation performance [14]. Nevertheless, the direct modeling and verification of complex algorithms in such frameworks is not an intuitive task and requires much effort and experience.

In this paper we present an approach to combine the benefits of the above two concepts. The main contribution of this work is the extension of [7] for multi-core platforms managed by a POSIX compatible operating system and the evaluation of its applicability for such platforms. We propose to start with a functional model specified in Matlab/Simulink with a guided refinement to an executable model on a virtual multicore platform modeled in SoCLib. A co-simulation of Matlab/Simulink (controller environment) and SoCLib (embedded control software running on multicore hardware platform) is set up to allow a virtual-platform in the loop verification which enables embedded software timing evaluation for a multicore platform.

The paper is structured as follows. In section 2 we discuss the related work to our approach. Next we describe the extension of our former design methodology, including the performance evaluation and the functional verification procedures, for multicores platforms. Section 4 describes the use-case of an ignition controller mapped on a dual-core platform and discusses the results. The last section concludes the paper and gives an outlook on future work.

2 Related Work

Mühleis et al. [10] presented a co-simulation between a SystemC and a Simulink model for a control algorithm performance evaluation, which is very similar to our approach. As proposed in our approach the model is automatically compiled from Simulink into an executable for the virtual-platform. Yet in their approach the delay caused by the communication and

synchronization between different (dependent) control applications mapped on different cores was not considered.

Boland et al. [8] and Tomasena et al. [9] have also achieved a co-simulation between SystemC and Simulink using Matlab's engine interface functions. In [8] the intention was to reuse test-cases and golden models in Simulink to verify refined hardware components in SystemC. The approach in [9] employs a description of the architecture of the system as a SystemC transaction level model and a description of the algorithm in Matlab. During co-simulation the SystemC architectural elements use Matlab's engine to execute and synchronize with the Matlab model. In both, [8] and [9], the SystemC model was developed independent from the Simulink model and no code-generation was done.

In [11] Bartolini et al. presented a co-simulation between SystemC and Simulink with the focus on exploring power, thermal and reliability management control strategies in high-performance multicores. The difference to our approach is that the controller in Simulink has been coupled with a model of the plant (environment) represented in SIMICs (a virtual-platform framework) while we did the opposite by coupling the controller code executed on the virtual-platform with the Simulink plant model.

Cha et al. [12] proposed an automatic synthesis of real-time multicore systems based on Simulink applications. In difference to our work, no virtual-platform in the loop, but a traditional HIL (Hardware In the Loop) approach directly evaluating the implementation on a hardware platforms has been performed. Moreover, communication and synchronization overheads/times between control blocks mapped on different cores were not considered.

Huang et al. [13] presented a Simulink-based heterogeneous Multiprocessor SoC (System on Chip) design flow for mixed hardware/software refinement and simulation. In their design flow the Simulink model is refined manually to achieve a Simulink CAAM (Combined Algorithm and Architecture Model). This offers advantages in terms of modular code generation and fast simulation of the refined system. They did not explicitly describe how the partitioning in the CAAM model was done without considering explicit knowledge of the timing properties after mapping it on the multi-core platform. So our virtual-platform in the loop with the Simulink plant model concept would be complementary for this work towards achieving this.

With respect to the related work our proposed approach benefits from the co-simulation of a flexible and well accessible virtual-platform with a timing accuracy up to a cycle accurate level. The VP in the loop approach enables reuse of test-cases and bi-simulation with golden models for the functional verification and validation of the stepping and timing requirements of the control algorithm implemented on a dedicated multicore platform.

3 Virtual-Platform in the Loop Simulation

3.1 Design Flow

In [7] we described how our virtual platform in the loop embedded software timing verification has been realized for a single core platform. In this paper we will extend this approach to realize a virtual platform in the loop embedded software timing verification for multicore platforms.

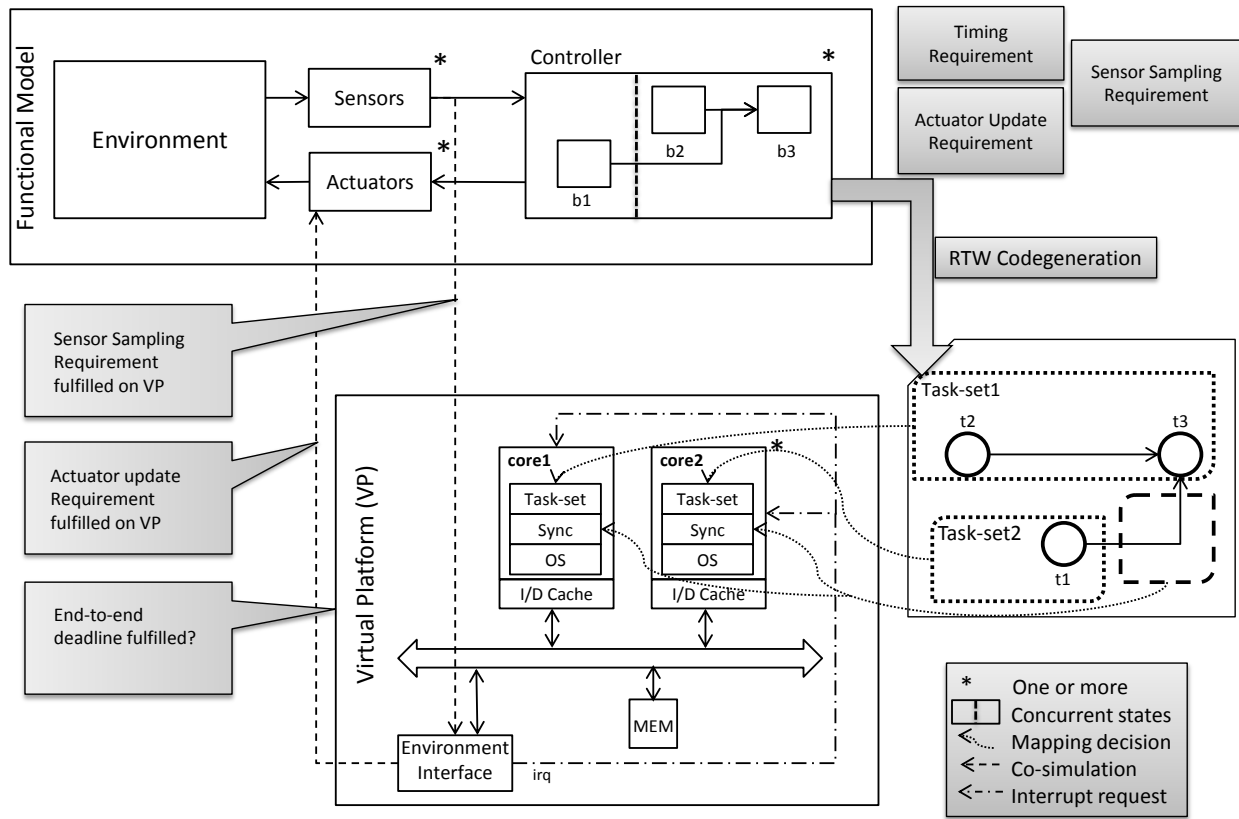


Figure 1 Design Flow

Figure 1 shows an overview of our design flow. As it can be seen, the functional model can consist of several sensors and actuators and possibly multiple controllers that control the same environment model (or the process to be controlled). After defining functional and non-functional requirements of the control system to be developed, the next step is to model it in Simulink.

The controller(s) can be implemented using pure Stateflow [1] with the combination of other Simulink blocks that must be supported by the code-generator. At this level, the control algorithm can be partitioned into concurrently executing blocks in Simulink or concurrently executing hierarchical states in Stateflow (see b1, b2, b3 in Figure 1).

After verifying and validating the functionality of the controller model within Matlab/Simulink, RTW code-generation is used to generate executable code for single cores of the target platform.

Tasks are then created from this generated code. After that task-sets are constructed from these tasks and each of these task-sets is then allocated to one core. Every core as it can be seen in Figure 1 has its own operating system, a synchronization and communication layer, and the mapped task-set to be executed. The inter-core communication and synchronization is then realized with the help of locked-based shared FIFO queues (more in section 3.2.2). In a next step these task-sets together with the corresponding synchronization and communication primitives are cross-compiled, linked to an operating system (murekh [5]) and downloaded into a memory component in the SoCLib virtual-platform.

The environment interface component in the virtual-platform is intended to communicate with the functional Simulink model allowing the co-simulation of SoCLib and Simulink and also the execution time measurement of the generated code. Whenever the target processor gets an interrupt signal from the environment interface, the corresponding task-set is executed and runs to completion. Finally the measured execution time values are analyzed and evaluated w.r.t to the functional requirements and performance requirements. In the case of functional mismatch, the implementation on the virtual-platform must be examined and corrected until the expected behavior is reached. In the case of timing violation detection the following actions can be performed: modifications in the functional model (e.g. from floating point to fixed-point calculation or other controller algorithm), changing the mapping or changing the platform architecture (e.g. faster cores, faster interconnect).

More implementation details of the co-simulation and the timing measurement method can be found in [7]. The basic idea behind the timing measurement in our co-simulation environment will be outlined in the following. In a Simulink simulation with a fixed-step solver type, models are stepped periodically according to a fixed sampling period called fixed-step size (this variable can be specified in the simulation configuration). The simulated time proceeds till the fixed-step size value is reached and updates the model at this moment (takes an input, makes the internal computation and produce an output) according to a specified solver¹.

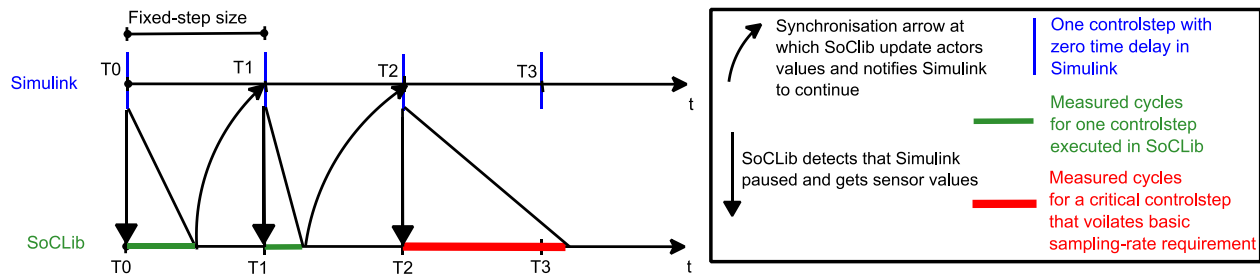


Figure 2 Co-simulation of Simulink and SoCLib

We define a control-step as one update of the controller model (one transition in Stateflow) in a given period which corresponds to one update of its generated code. In this update one execution

¹ A solver implements a specific numerical integration technique to interpolate values between two consequent time instants of simulation.

of the generated step functions² of all task-sets of the partitioned controller including the communication and synchronization between them is done.

SoCLib is the master of the co-simulation. It starts by initializing all virtual-platform components and then launches an instance of the Simulink model. As seen in Figure 2 the Simulink model steps once and then pauses. The environment interface component in SoCLib detects this pause and it gets the current sensor values corresponding to this step from the Simulink environment model and issues interrupts to notify the cores. The activated tasks on the corresponding cores can then get the sensor values from a dedicated core which communicates with the environment interface, execute their step function once and send the updated actuator values back to it. During software execution the environment interface component records the end-to-end execution time which is the time from the moment where the sensors data were received until the moment where the last actuator was updated including the inter-core communication and synchronization at a cycle accurate level. After software execution for the control-step has been completed, the environment interface sends the updated actuator values and the timing measurement values back to Simulink. Then the co-simulation interface wakes Simulink to resume the execution of another control-step. This procedure can be iterated until the desired number of control-steps has been executed.

If the measured cycles of any control-step exceeds the fixed-step size, as we see in Figure 2 (see fat horizontal line), then we have detected a timing violation. It is important to note here that even if any control-step lasts longer than the upper bound time requirement, the target processor still executes it till its end. This means that timing violations would not manipulate the time instants at which the actuator values are updated and thus does not have any influence on the measured functional results of the controller.

3.2 Extending approach for Multicore

3.2.1 Assumptions

Following assumptions were made to enable the extension of our approach, towards multicores. Till now our general approach was limited to single rate Simulink models. An adaption of our approach to multi-rate models should be possible, if we constrain the input model as a statically schedulable synchronous data flow (SDF) model and generate for every inter-core communication its own FIFO queue with a fixed size. If the model is statically schedulable we can calculate a fixed FIFO queue size and guarantee a deadlock free implementation [15]. But this is not in the focus of this work and should be evaluated in future research.

For the multi-core platform architecture, we assume shared memory architecture with homogenous cores. When using caches, coherence is implemented in hardware. In addition the inter-core communication is realized through a lock-based shared memory and no explicit

² The codegenerator generates for every block a step function which should be executed according to the given period. When executed it takes the input values makes one transition (in Stateflow) and updates the output values.

message passing between the cores was considered. Furthermore we assume that only one dedicated core among the cores is allowed to make I/O access to the environment interface, getting the sensors and updating the actuators. This is important to guarantee that no race-conditions occur between the cores when updating same actuators in order to preserve our non-invasive timing measurement.

We consider a static mapping of tasks to cores, i.e. no dynamic task migration is considered. We also assume a distributed POSIX-compatible operating system supporting non-preemptive rate monotonic (static) scheduling on each core (asymmetric multi-processing). Moreover, we consider task level parallelism while partitioning the Simulink model. In contrast to data-level parallelism where the data is distributed between different cores with the same task executed on them, in task-level parallelism each core executes a different task on the same or different data.

3.2.2 Procedure

When mapping a Simulink application on a multicore platform we are faced with two main challenges. Firstly it is necessary to preserve the execution semantics in terms of causality of the SDF model of computation to be able to validate functionality. This means that in every control-step, it must be ensured that the tasks being executed at the virtual-platform level are the corresponding tasks to the blocks that were executed at the reference model level. Secondly if dependent tasks are mapped onto different cores, then an inter-core communication and synchronization mechanism will be needed to guarantee the correct behavior at the virtual-platform level.

We dealt with the second challenge by introducing a locked-based shared FIFO queue implementation in shared-memory for inter-core communication and synchronization. If the queue is empty/full and a consumer/producer task tries to access the queue, a blocking mechanism sets the running task to a waiting/blocking state and notifies it when values are available. For this and for future extensions an operating system is needed. The mutekh [5] operating system implemented in SoCLib is POSIX compliant and offers a suitable solution for our needs. Tasks are implemented here as POSIX threads (pthreads). In addition we ported an open source C implementation of a bounded buffer queue using POSIX threads [4]. Using this queue, which takes advantage of pthread's mutexes and conditions, the inter-core communication and synchronization is realized.

Concerning the first challenge we used the concept of deferred interrupt processing in which interrupts can be used to unblock waiting tasks using binary semaphores. With the help of this concept, every time the environment interface component receives values from the Simulink environment model it notifies every core which in turn locally activates the task-subset which has to be run according to a static schedule in this control-step.

Our timing measurement method was enhanced to measure the end-to-end execution time which is the time from the moment where the sensors data were received until the moment where the last actuator was updated in a non-invasive cycle accurate manner. In other words, and according to our approach, the end-to-end execution time is the time from the moment where the first interrupt is issued from the environment interface component till the moment where last actuator

is updated by the dedicated core responsible of communicating with the environment interface component.

4 Evaluation

For the evaluation of our concept we have chosen a simple and academic ignition controller system [6] (which was already analyzed for single core platforms in [7]). In the future we intend to map more computational intensive applications, since these allow better exploitation of the computational parallelism of multi-cores.

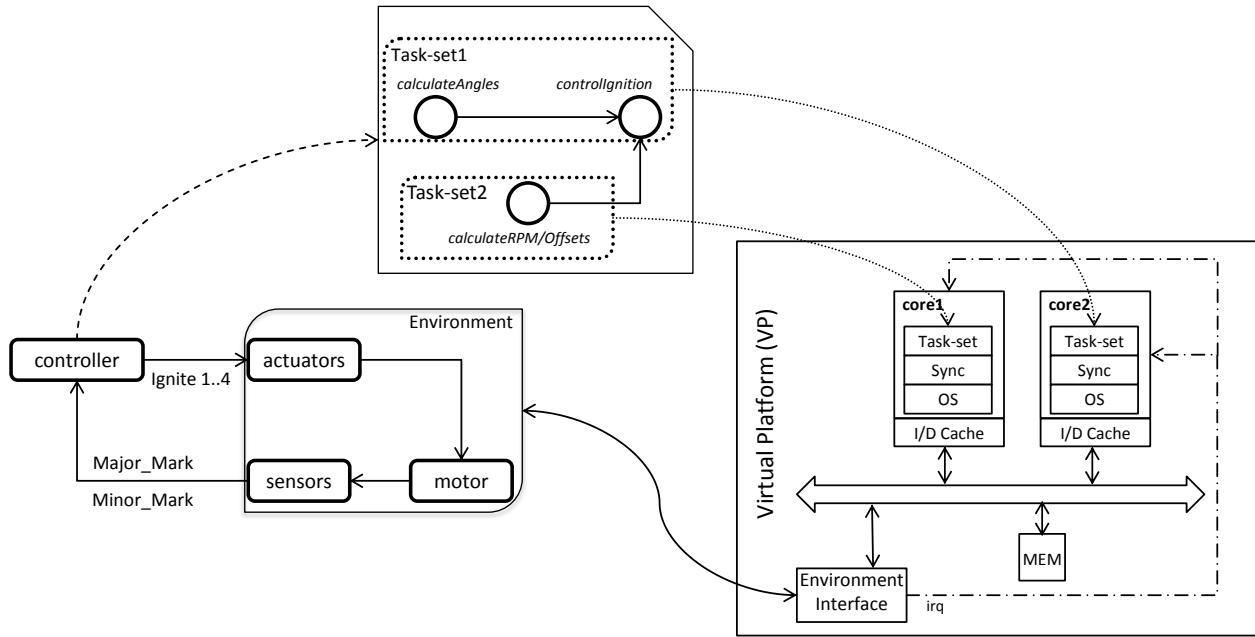


Figure 3 *Left:* the controller model with the environment modeled in Simulink, *Right:* the platform architecture implemented in SoCLib, *top:* the task-set extracted from the controller model and mapped to the cores on the target platform.

The system was first modeled in Simulink (Figure 3 left side). It consists of an engine model mimicking an Otto-motor and an ignition controller block modeled in Stateflow where the control algorithm is implemented. In order to optimize ignition times the controller depends on two feedback sensors labeled “Major Mark” and “Minor Mark” in the Figure, and on four actuators labeled ignite1..4, which generate a spark on the corresponding spark plug of the motor cylinder. The ignition controller needs to ensure adequate firing times of the spark plugs (typically one spark plug per cylinder). Early or late firing could lead to mechanical stress, undesired behavior, and damage of the engine. After partitioning and functional verification in Simulink, C code was generated from the controller model for each partitioned block. The tasks were implemented as pthreads executing the corresponding functions of the generated C code. The ignition controller was partitioned into three tasks (Figure 3 top), two of them which can run concurrently, and the third depends on the output of the first two. In Figure 3 (right) we see the developed virtual multicore platform. It consists of two ARM6 processor (each @ 400 MHz) with their associated

data and instruction cache, one on-chip memory (RAM) and the environment interface. All these components were connected via VCI ports to a simple bus model implemented at cycle accurate bit accurate (CABA) level. “Task-set2” which consists of *calculateRPM/Offsets* task is mapped to “core1” while “Task-set1” comprising the two other tasks *calculateAngles* and *controlIgnition* is mapped to “core2”. The inter-core communication and synchronization was implemented as mentioned before (see section 3.2). Afterwards the operating system with the tasks was cross-compiled and loaded to the RAM. Whenever an interrupt is initiated from the environment interface, in every core the assigned task-set would be activated to execute with the needed synchronization. The environment interface in turn initiates interrupts whenever Simulink values are available, namely at every period as already described in section 3.1. The co-simulation was made for 8000 control-steps, with a sampling period of 5 μ s. This imposes the requirement that the execution of every control-step, including inter-core communication and synchronization, must not be larger than this sampling period.

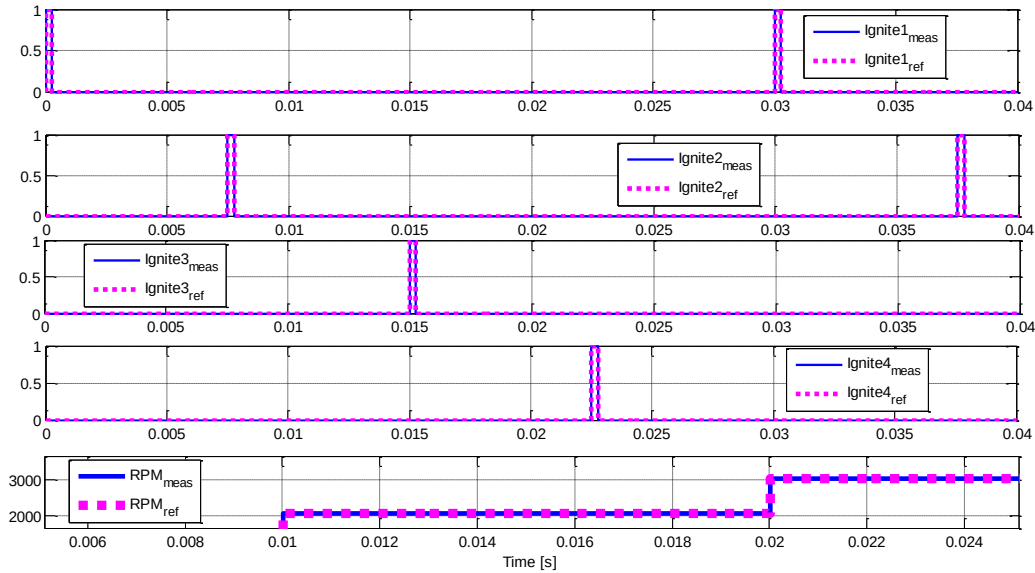


Figure 4 Co-simulation results of the controller measured and the reference values

The measured values plotted together with the reference results can be seen in Figure 4. The reference model values (dotted line) and the measured values (thick solid line) show exactly the same results. For the given RPM input values the functionality and timings of the generated code mapped to our multi-core platform has been successfully verified against the golden reference model in Simulink. Since the time instants at which the actuators are updated, are not influenced with timing violations according to our co-simulation concept (see section 3.1) we must still examine if the timing requirement was satisfied.

Since the task *controlIgnition* on “core2” must wait till task *calculateRPM/Offsets* finishes, the end-to-end execution time depends only on the finish time of the task-set of “core2” (see Figure 4 top). In Figure 5 we have plotted the measured duration of every control-step (end-to-end deadline) of the task-set running on “core2”. According to the measurements we can observe that all the control-steps were executed within a time interval of [14.8, 16] μ s which larger than our timing requirement (every control-step should be executed in no more than 5 μ s). In order to

handle this timing violation, the designer has the choice to replace the core with a faster one, optimizing the ignition control algorithm (implementation) or optimizing the generated code.

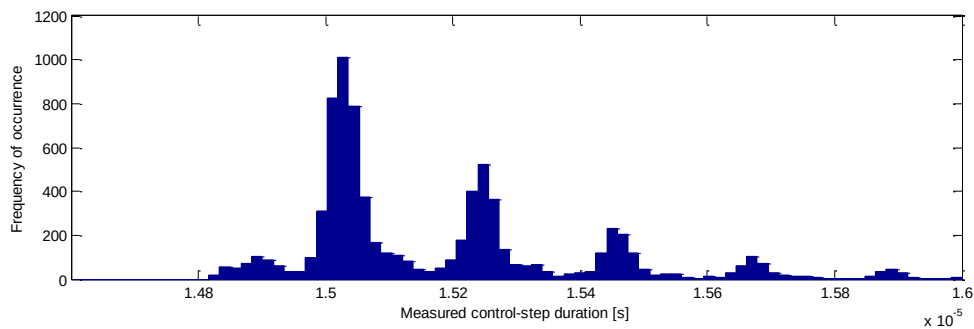


Figure 5 Measured control-steps duration

5 Conclusion

In this paper, we demonstrated the applicability of our method to validate a real-time critical ignition controller system mapped on a dual-core platform in terms of timing and functionality. This was achieved through running a cycle accurate virtual-platform, with the control application running on the concrete multi-core architecture, in the loop with the Simulink environmental model. With the help of deferred interrupt-based multitasking and lock-based shared memory inter-core communication and synchronization, the approach was capable to preserve the causality of the golden model and to enable simulation-based verification of the timing and functionality of a control algorithms mapped on a multicore platform. The proposed approach contributes to the validation of timing requirements of critical control algorithms through measuring their execution times on a cycle accurate level. Using these measurements, design decisions can be made with minimized costs and shorter development times in early cycles. Moreover, it supports a seamless functional verification via the semi-automated virtual-platform-in-the-loop V&V technique. Future research should try to evaluate the concept for scalability involving more computational intensive applications mapped on many-cores. Furthermore, by constraining the input model to a statically schedulable synchronous data flow (SDF) model an evaluation should be made for the applicability of our approach for validating streaming applications with multi-rate tasks.

References

- [1] The Mathworks, Inc: Simulink 7.4, Stateflow 7.4, Real-Time Workshop 7.4, <http://www.mathworks.com/products/>.
- [2] SystemC Library, <http://www.systemc.org>
- [3] SoCLib open platform for virtual prototyping of multi-processors system on chip, <http://www.soclib.fr/>

- [4] C-pthread-queue: GNU GPL-v3 C implementation of a bounded buffer queue using POSIX threads, <http://code.google.com/p/c-pthread-queue/>
- [5] MutekH: Lightweight POSIX compatible operating system for embedded platforms, <http://www.mutekh.org/>
- [6] Fränzle, M. Introduction to Stateflow: Ignition Control, 2004
- [7] Fakih, M.; Poppen, F; Grüttner, K. & Rettberg, A.
“Simulink and Virtual Hardware Platform Co-Simulation for Accurate Timing Analysis of Embedded Control Software” in *Proc. of ASIM/GI Workshop*, February 2011, pp. 17-26
- [8] Boland, J.; Thibeault, C. & Zilic, Z.
Using Matlab and Simulink in a SystemC Verification Environment,
In Proceedings of Design and Verification Conference, DV-Con, 2005
- [9] Tomasena, K.; Sevillano, J.; Arrue, N.; Cortes, A & Velez, I.
Embedding Matlab in SystemC Transaction Level Modeling for Verification,
CEIT and TECNUN (University of Navarra), 2009
- [10] Mühleis, N.; Glaß, M.; Zhang, L. & Teich, J.
A co-simulation approach for control performance analysis during design space exploration of cyber-physical systems *ACM SIGBED Review*, 2011, 8, 23-26
- [11] Bartolini, A.; Cacciari, M.; Tilli, A.; Benini, L. & Gries, M.
A virtual platform environment for exploring power, thermal and reliability management control strategies in high-performance multicores *Proceedings of the 20th symposium on Great lakes symposium on VLSI*, 2010, 311-316
- [12] Cha, M. & Kim, K.
Automatic Building of Real-Time Multicore Systems Based on Simulink Applications
Ubiquitous Computing and Multimedia Applications, 2011, 209-220
- [13] Huang, K.; Yan, X.; Han, S.; Chae, S.; Jerraya, A.; Popovici, K.; Guerin, X.; Brisolara, L. & Carro, L. Gradual refinement for application-specific MPSoC design from Simulink model to RTL implementation *Journal of Zhejiang University-Science A*, 2009, 10, 151-164
- [14] Viaud, E., Pecheux, F. & Greiner, A.
An efficient TLM/T modeling and simulation environment based on conservative parallel discrete event principles 2006, pp. 94-99
- [15] Lee, E.A. & Messerschmitt, D.G.
Synchronous data flow Proceedings of the IEEE, **1987**, Vol. 75(9), pp. 1235-1245



DYMOLA - INTERDISZIPLINÄRE SIMULATION VON PHYSIKALISCHEN SYSTEMEN

Wir haben die Lösung für Sie!

Um die wachsende Komplexität moderner Produkte und Projekte beherrschbar zu machen, hat sich die Simulation als disziplinübergreifender, integrierter Entwicklungsansatz etabliert. Darunter versteht man die geschlossene Modellierung der physikalischen und logischen Eigenschaften zu entwickelnder Produkte. Die anschließende Simulation soll das Produktmodell bezüglich der geforderten Eigenschaften überprüfen und optimieren. So entstehen letztlich bessere Produkte bei deutlich kürzeren Entwicklungszeiten und mit weniger realen Prototypen und Versuchsaufwand.

Nutzen & Vorteile

- Besseres Abbild der Wirklichkeit durch Berücksichtigung physikalischer Phänomene in einem Simulationsmodell
- Einfacher und schneller Modellaufbau durch Nutzung, Wiederverwendung und Abwandlung Modellbibliotheken
- Freier Zugang zu einer stetig wachsenden Sammlung von Modellen der Modelica Association
- Effizientere Ressourcennutzung und schnellere Simulationsmodelle infolge Vereinfachung der Modellgleichungen
- Bessere Produkte dank disziplinübergreifender Simulation bereits in der Spezifikationsphase

Kostenlose CENIT Web-Seminare

Zu Dymola bietet die CENIT Akademie einige kostenlose Web-Seminare als Onlinekurse an. Darin werden Anwendungsbereiche und Funktionen der Lösung übersichtlich von Experten dargestellt. Zusätzlich besteht selbstverständlich die Möglichkeit, Fragen zu stellen. CENIT Web-Seminare vermitteln Ihnen weitergehendes Detailwissen - ohne, dass Sie Ihr Büro verlassen müssen.

Weitere Informationen sowie aktuelle Termine finden Sie unter www.cenit.de/webseminare.

Haben Sie noch Fragen?

Kontaktieren Sie doch einfach unser CENIT-Dymola Kompetenzteam per E-Mail unter info@cenit.de



Ansätze zur Systemsimulation im Automobilbau

Ewald Hessel, Hella KGaA Hueck & Co, Sprecher AK30

Joachim Haase, Fraunhofer IIS/EAS Dresden

Ewald.Hessel@hella.com, joachim.haase@eas.iis.fraunhofer.de

Zusammenfassung

Probleme in Zusammenhang mit der Erstellung und Nutzung von Simulationsmodellen elektrischer Komponenten in einem Fahrzeug und ihre Nutzung zur Untersuchung der Wechselwirkung mit der Fahrzeugumgebung und nichtelektrischen Baugruppen sind Gegenstand des vorliegenden Beitrags. Einen Schwerpunkt bildet dabei die Auseinandersetzung mit Fragen, die sich durch die unterschiedlichen Anforderungen der im Entwicklungsprozess Beteiligten – Fahrzeughersteller, Zulieferer und Halbleiterhersteller – und deren Simulationslösungen ergeben. Der Beitrag will zu weiterer Diskussion darüber anregen wie in diesem Zusammenhang insbesondere der Aufwand für die Modellerstellung reduziert werden kann.

1 Einleitung

Neben der Nutzung von Simulationsverfahren bei der Entwicklung einzelner Fahrzeugkomponenten wird die Simulation zunehmend für die Untersuchung, Validierung und Optimierung der aus den Komponenten zusammengefügt Systeme und Teilsysteme eingesetzt. Eines dieser Teilsysteme ist das Bordnetz, das aus allen elektrischen Komponenten eines Fahrzeugs gebildet wird.

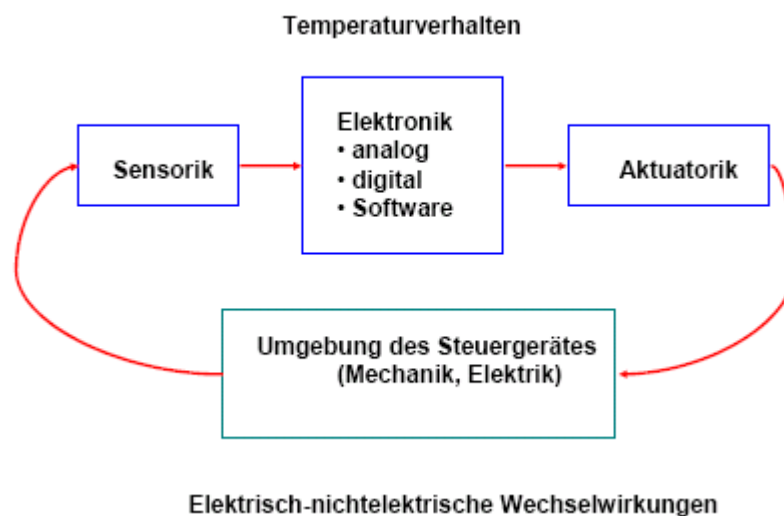


Abbildung 1: Zu betrachtende Wechselwirkungen bei der Systemsimulation

Bei der Simulation des Bordnetzes spielen elektrisch-nichtelektrische Wechselwirkungen innerhalb von Komponenten ebenso wie zur Umgebung sowie die gemeinsame Betrachtung von Hard- und Softwarebestandteilen eine Rolle. In Abb. 1 ist diese Situation kurz skizziert. Auch ist speziell bei der Fahrzeugentwicklung die Einbeziehung realer Objekte oder materieller Modelle davon in die gemeinsame Simulation von Bedeutung.

Der Erfolg einer Simulation hängt neben der Qualität der verwendeten Simulationswerkzeuge wesentlich von den eingesetzten Simulationsmodellen ab. Modelle entstehen im Entwicklungsprozess an den unterschiedlichsten Stellen, für die unterschiedlichsten Zwecke, auf unterschiedlichsten Abstraktionsebenen und für die unterschiedlichsten Simulationswerkzeuge, um nur einige Gesichtspunkte zu nennen. Einige dieser Fragestellungen sind in Abb. 2 aufgeführt.

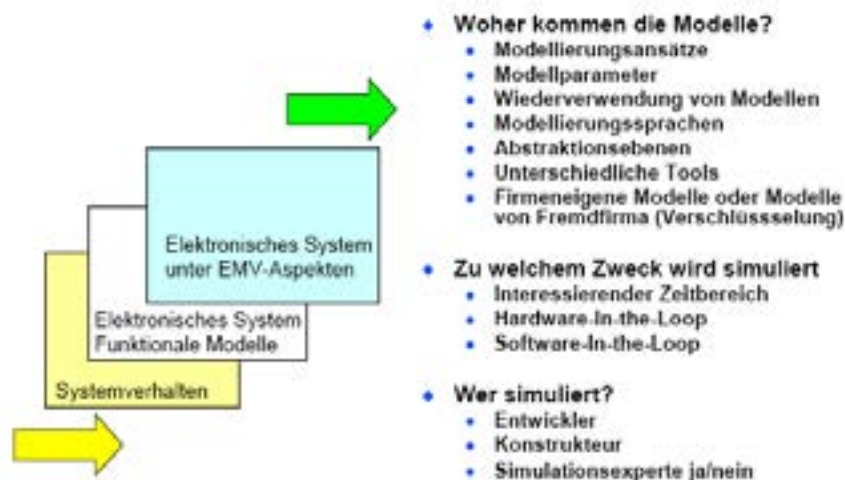


Abbildung 2: Fragestellungen bei der Systemsimulation

Für den Einsatz der Systemsimulation im Entwicklungsprozess spielen die Verfügbarkeit von Modellen und die Beurteilung ihrer Eignung für eine spezielle Aufgabe eine wichtige Rolle. Typische Merkmale in diesem Zusammenhang sind dabei zum einen die Frage, welches Objekt modelliert wird (Abbildungsmerkmal), zum anderen aber auch, welche Eigenschaften des Objektes durch das Modell beschrieben werden und welche nicht (Verkürzungsmerkmal) und für welchen Zweck ein Modell erstellt worden ist (Pragmatisches Merkmal) [1]. Beispiele für unterschiedliche Abstraktionsebenen – Beschreibung des Systemverhaltens, Beschreibung des funktionalen Verhaltens, Beschreibung des Verhaltens unter EMV-Aspekten - von Modellen eines Objektes sind in Abb. 2 skizziert.

Einerseits liegen auf Grund von Arbeiten in Zusammenhang mit der Entwicklung von Komponenten eine Reihe von Modellen für diese vor, andererseits ist es für den Nutzer oft schwer, ein geeignetes Modell zu finden und zu entscheiden, ob es sich für seine Aufgabe eignet. Durch die sehr unterschiedlichen Simulationsumgebungen ist ein weiteres Problem die Verwendung von Modellen, die für eine Umgebung erstellt worden sind, und in einer anderen verwendet werden sollen.. Auch die Bereitstellung von Ausgangsdaten für die Modell-erstellung ist häufig problematisch. Die unterschiedlichen Aspekte in Zusammenhang mit

Modellerstellung und Modellaustausch aus Sicht der Systemsimulation sollen im Folgenden kurz skizziert werden.

2 Modellanforderungen

Üblich ist bei der Modellierung für die genannten Aufgaben ein objektorientierter Ansatz. Dabei wird zwischen der Beschreibung des Modellrandes (Verbindungspunkte und Modellparameter) und des eigentlichen Modellverhaltens unterschieden. An den Klemmen werden entweder Fluss- und Differenzgrößen (bei Netzwerkmodellen) oder Ein- und Ausgangsverläufe (bei Signalflussmodellen) betrachtet. Eine Wechselwirkung zwischen den Modellen wird durch die Zusammenschaltung und in der Regel nur über die Signalverläufe an den Klemmen hergestellt.

Das rein *analoge Verhalten* wird dabei über Differentialgleichungen oder auch durch lineare oder nichtlineare Abbildungen vorgegeben. Durch das Abstraktionsniveau ergibt sich ein Kompromiss zwischen Modellgenauigkeit und Simulationsgeschwindigkeit (siehe auch Abb. 3). Modelle für die Systemsimulation werden in der Regel nur langsamere Änderungen berücksichtigen können als Modelle, die für den Entwurf von Komponenten benötigt werden. Werden Modelle im Rahmen einer Echtzeitsimulation benötigt, ist die Auswertung der Modellgleichungen innerhalb einer definierten Zeit (WCET – worst case execution time) zu sichern. Letztlich kann dies im Allgemeinen nur durch Modellbeschreibungen, die in Form gewöhnlicher Differentialgleichungen (ODE – ordinary differential equations) vorliegen, erfolgen. Die Überführung eines in DAE-Beschreibung (DAE – differential algebraic equations) vorliegenden Modells in eine (konsistente) ODE-Darstellung ist eine üblicherweise anspruchsvolle Aufgabe [2]. Diese Aussage trifft auch zu, wenn als Ausgangsbeschreibung ein auf partiellen Differentialgleichungen beruhender Ansatz vorliegt.

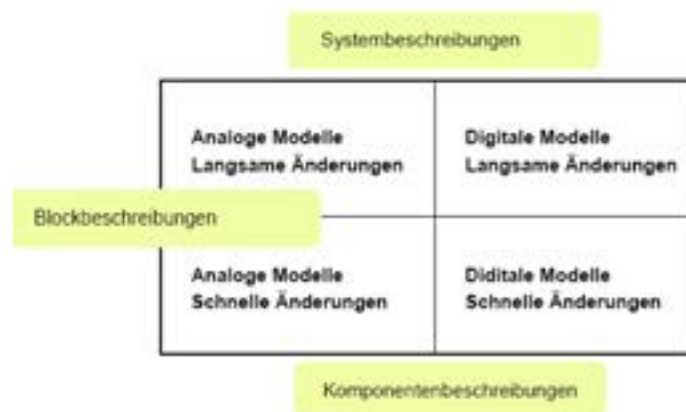


Abbildung 3: Anforderungen unter Genauigkeits- und Rechenzeitaspekten [3]

Unter Konsistenz von Modellen ist dabei üblicherweise qualitativ annähernd ähnliches Verhalten zu verstehen, wenn an den Klemmen eine Anregung mit gleichen Signalverläufen auftritt. Diese Ähnlichkeit formal zu beschreiben und zu prüfen ist, zumindest für das analoge Verhalten, eine aus unserer Sicht nicht abschließend gelöste Aufgabe (siehe dazu auch [4]).

Die Konsistenz zu sichern ist insbesondere dann von Interesse, wenn Modelle unterschiedlicher Abstraktionsebenen (siehe auch Abb. 4) miteinander verbunden werden sollen.



Abbildung 4: Abstraktionsebenen und Beschreibungsmittel

Die Beschreibung des *diskreten Verhaltens* läuft in vielen Fällen auf die Beschreibung mit zeitdiskreten Zustandsgleichungen hinaus [5]. Ereignisgesteuerte Simulationsalgorithmen oder auf Datenflussbeschreibungen basierende Algorithmen werden zur Simulation eingesetzt. Dabei sind (in den meisten Fällen) die Signalwerte Elemente endlicher Mengen. Digitale Hardware wird üblicherweise so beschrieben.

Modelle, die sowohl analoges als auch diskretes Verhalten beschreiben, werden als *hybride Modelle* oder mixed-signal Modelle bezeichnet. Die analoge Verhaltensbeschreibung greift auf die Beschreibung des diskreten Verhaltens zu und umgekehrt. Dieses Wechselspiel unterscheidet sich im Detail vielfach in Abhängigkeit von den verwendeten Modellbeschreibungssprachen und Simulationswerkzeugen. Eines der vielen Probleme, das bei der Modellerstellung vermieden werden muss, ist das Auftreten sehr vieler (theoretisch unendlich vieler) Ereignisse in einem endlichen Zeitintervall, auch als Zeno-Verhalten bezeichnet [6]. Zur Erklärung des Phänomens wird oft der springende Ball herangezogen. Das Problem kann aber auch bei der Beschreibung des Übergangs von Gleit- zu Haftreibung oder bei schwellwerthängigem Verhalten auftreten. Zur Vermeidung sind in der Regel von der Modellbeschreibungssprache und dem eingesetzten Simulationswerkzeug abhängige Vorkehrungen zu treffen.

Die dritte für die Systemsimulation interessante Komponente ist die *Software*, die beispielsweise in den Steuergeräten zum Einsatz kommt. Stand bei den in Hardware eingebetteten Systemen (embedded systems) historisch gesehen die effiziente Nutzung der beschränkten Ressourcen wie geringer verfügbarer Speicher, kleine Wortbreite und relativ geringe Taktfrequenz im Vordergrund, so verschiebt sich die Fragestellung, auch unter dem Gesichtspunkt parallel arbeitender Softwaresysteme, stärker in Richtung Vorhersagbarkeit und Zuverlässigkeit [7], Fehlertoleranz, Sicherheit, Reduzierung des Energieverbrauch usw. Der neuen Qualität hat man versucht mit der Einführung des Begriffs Cyber-Physical Systems (CPS) Rechnung zu tragen. Damit ergeben sich auch neue Anforderungen an Entwurf und Verifikation der Software.

Benötigt werden in diesem Zusammenhang Modelle, die analoges und diskretes Verhalten von Komponenten und der Umgebung eines Systems beschreiben. Ein erfolgreiches Vorgehen verlangt, den Aufwand an dieser Stelle zu reduzieren, neue Modelle einfach erstellen zu können und existierende Modelle schnell nutzbar zu machen. Forderungen in diesem Zusammenhang sind, auf Standardlösungen für die Modellierung („commercial-off-the-shelf“ modeling environments) zugreifen zu können, eine automatisierte Transformation von Modellen von einer Abstraktionsebene in eine andere zu unterstützen und eine zeitaufwändige und fehleranfällige Validierung der Modelle zu vermeiden [8].

3 Modellnutzung in heterogenen Simulationsumgebungen

3.1 Modellerstellung

Bei der Erstellung von Modellen sind einerseits die Anforderungen an die Modelle zu beschreiben, andererseits die für die Modellierung verfügbaren Daten wie Messergebnisse oder Datenblattangaben bereit zu stellen. Beim Übergang von einer Abstraktionsebene zur anderen baut die Modellerstellung auch vielfach auf Simulationsergebnissen auf (z. B. unter Anwendung von Methoden der Systemidentifikation, siehe auch [9]). Versuche, diesen Prozess zu systematisieren haben sich allgemein verbindlich - wie z. B. in [10] angeregt - bisher offenbar nicht durchsetzen können.

Dabei tragen klare Festlegungen durchaus zur Reduzierung des Modellierungsaufwandes und zur Vermeidung von Missverständnissen bei. Alleine die Verfügbarkeit von Diagrammen aus Datenblättern in maschinenlesbarer Form vereinfacht diesen Prozess. Die Beschreibung der Anforderungen an das Modell (siehe auch Abb. 3) beeinflusst die Auswahl des mathematischen Modellansatzes (linear, nichtlinear, statisch, dynamisch, ...). Auch ist festzulegen, für welche Simulationsarten (Nominalwertanalyse, statistische Simulationen, Zeitbereichs- und/oder Frequenzbereichssimulationen) das zu erstellende Modell eingesetzt werden soll. Ist beispielsweise nur eine Frequenzbereichsanalyse mit dem Modell vorgesehen, braucht ggf. kein Zeitbereichsmodell erstellt werden.

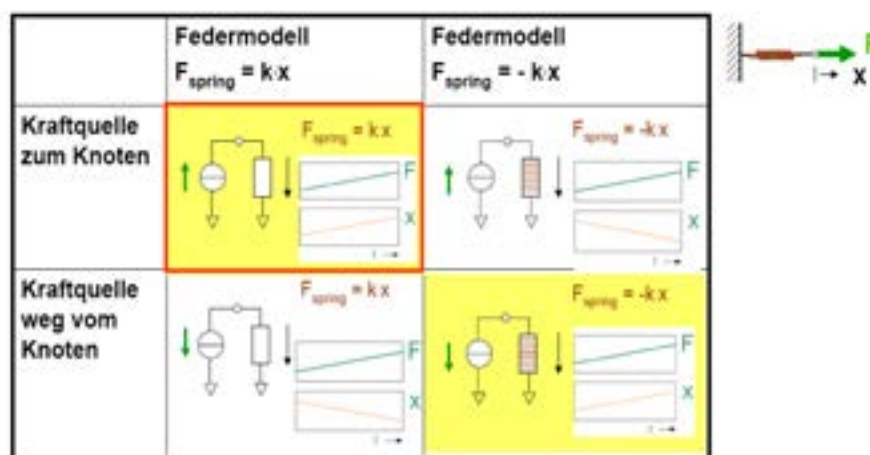


Abbildung 5: Varianten für Zählrichtungen bei mechanischen Netzwerkmodellen

Der Dokumentation der Modelle ist insofern besonderes Gewicht beizumessen, weil bestimmte Eigenschaften der Modelle (verwendete Maßeinheiten, Zählrichtungen – siehe Abb. 5, ...) sich teilweise nicht aus rein formalen Kriterien bei Verwendung verbreiteter Beschreibungsmitteln wie VHDL-AMS, Modelica, Matlab/Simulink und Simscape, Verilog-AMS ableiten lassen (siehe auch [11]).

Bestandteil der Dokumentation muss letztlich auch eine Beschreibung des Vorgehens bei der Parametrisierung und zur Validierung eines Modells sein. Einzuschätzen oder zu Klassifizieren ist dabei auch die Qualität des Modells – der Grad des Vertrauens, den man dem Modell entgegenbringen kann. Eine Weiterverwendung von Modellen verlangt zudem eine nachvollziehbare Versionierung. Trotz gewünschter Automatisierung dieser Schritte sind diese in vielen Fällen „von Hand“ durchzuführen. Als ein Fazit ergibt sich an dieser Stelle: Eine Voraussetzung für eine bessere Unterstützung ist eine stärkere Formalisierung der Beschreibung der der Modellierung zugrunde liegenden Ausgangsdaten und der Anforderungen an das zu erstellende Modell. Eine derartige formale Beschreibung kann auch zum Wiederauffinden von Modellen u. U. sinnvoll eingesetzt werden.

3.2 Modellbibliotheken

Modellierungsaufwand kann reduziert werden, wenn auf Modelle aus Modellbibliotheken (siehe z. B. [12]) zurückgegriffen werden kann. Es bleibt dann „nur“ die Aufgabe für einen Anwendungsfall, die passenden Modelle zu finden und zu parametrisieren. Zu unterscheiden ist u. a. zwischen Tool-spezifischen, unternehmensübergreifenden und unternehmensweiten sowie nutzereigenen Bibliotheken. Die unter 3.1 angesprochenen erforderliche Konsistenz kann bei der Erstellung der Modell für eine Bibliothek sicher gestellt werden. Durch bekannte Struktur der Modelle einer Bibliothek wird die Bereitstellung von Parametrisierungsvorschriften und –werkzeugen erleichtert. Diese verfügbar zu machen trifft in erster Linie auf Modelle zu, die in breiterem Umfang eingesetzt werden. Ein weiterer Vorteil der Verwendung von Modellbibliotheken besteht darin, dass Modelle „en bloc“ von einer Modellierungssprache in eine andere oder für eine andere Simulationsumgebung umgesetzt werden können. Wenn sich diese Schritte nicht automatisieren lassen, ist sicher ein einmaliger manueller Aufwand akzeptabel. Allerdings bleibt dann das Problem, dass bei Änderung von Modellen in einer Bibliothek die Modelle in anderen Bibliotheken nachgezogen werden müssen. Als Fazit ergibt sich: Modellbibliotheken sind ein möglicher Weg, den Modellierungsaufwand beim Nutzer zu reduzieren und Modelle von einer Simulationsumgebung oder einer Modellbeschreibungssprache in eine andere zu überführen. Voraussetzung für die Nutzung der Vorteile ist die Auswahl von Grundmodellen, deren wiederholte Verwendung gesichert ist, die erprobt sind und über einen längeren Zeitraum stabil genutzt werden können. Unter dem Aspekt Systemsimulation ist die Festlegung des Umfangs der für diese Zwecke zu verwendenden Modelle und die Festlegung auf zugrunde liegende Modellierungsansätze eine nicht abgeschlossene Aufgabe.

3.3 Modelltransformation und Modellaustausch

Im vorangegangenen Abschnitt ist auf die Frage der Modelltransformation kurz eingegangen worden. Bei der Umsetzung ganzer Modellbibliotheken, die wiederholt verwendet werden, ist sicher ein gewisser Aufwand akzeptabel. Etwas anders sieht die Situation aus, wenn einmalig einzelne Modelle umgesetzt werden müssen. In Sonderfällen mag die Umsetzung von Modellen aus einer Modellbeschreibungssprache in eine andere automatisierbar sein. In der Regel ist dies aber eine Aufgabe, die manuelle Eingriffe verlangt. Ein Grund mag darin zu suchen sein, dass Eigenheiten der jeweils genutzten Simulationsumgebung berücksichtigt werden müssen. Im Zusammenhang mit hybriden Modellen ist auf einen diesbezüglichen Aspekt im Abschnitt 2 kurz eingegangen worden.

Günstiger erscheint es, als Ausgangspunkt für eine automatisierte Umsetzung eine formale Beschreibung zu wählen, die im Wesentlichen die dem Modell zugrunde liegende mathematische Beschreibung repräsentiert. Aus dieser Beschreibung können dann Modelle in unterschiedlichen Beschreibungssprachen generiert werden. Erste Standardisierungsbemühungen hierzu hat es unter der Bezeichnung „Common Model eXchange“-Format [13] gegeben. Sehr viel erfolgversprechender scheinen die Aktivitäten zum Functional MockUp Interface [14] zu sein, die im Umfeld der Modelica-Standardisierung angesiedelt sind. Möglichkeiten und Grenzen des FMI in Zusammenhang mit der Nutzung anderer Beschreibungssprachen, z. B. VHDL-AMS, zu untersuchen, ist sicher eine interessante Problemstellung. Als kurzes Fazit in Zusammenhang mit der Modelltransformation ist festzustellen: Die Transformation von Modellen von einer Beschreibungssprache in eine andere ist nach vorliegenden Erfahrungen ein Problem, dass sich in der Regel ohne manuelle Eingriffe nicht lösen lässt. Günstiger erscheint es, von einer formalen Beschreibung zugrunde liegender mathematischer Zusammenhänge auszugehen und darauf aufbauend Modelle in unterschiedlichen Beschreibungssprachen und für unterschiedliche Simulationsumgebungen zu generieren.

3.4 Co-Simulationslösungen

Umgangen werden kann die Modelltransformation, wenn Simulationswerkzeuge die gleichzeitige Verwendung von Modellen in unterschiedlichen Beschreibungssprachen unterstützen. Das ist bei kommerziellen Werkzeugen häufig der Fall. Speziell für die Verwendung von Modelica-Beschreibungen in im Elektronikentwurf typischerweise eingesetzten Werkzeugen sind aber beispielsweise keine Lösungen bekannt.

Daneben ist der Einsatz von Co-Simulationslösungen eine Möglichkeit, Modelle, die für unterschiedliche Simulationswerkzeuge erstellt worden sind, gemeinsam im Rahmen einer Systembeschreibung zu nutzen. Solche Co-Simulationslösungen sind u.a.

- TISC Suite [15]
- Functional Digital Mock-Up FDMU [16]
- System Architect Designer SyAD [17]

Die zugrunde liegenden Ansätze unterscheiden sich u.a. hinsichtlich der Kopplung der Simulatoren und der Beschreibung des zu untersuchenden Gesamtsystems.

4 Weiterführende Schritte

Die zu untersuchenden Systeme werden immer komplexer. Analoge und digitale Hardware müssen gemeinsam mit Softwarekomponenten und Beschreibungen der Umgebung betrachtet werden. Ein entscheidender Punkt für die Systemsimulation ist die Verfügbarkeit geeigneter Modelle. Verschiedene in diesem Zusammenhang zu beachtende Aspekte sind in diesem Beitrag erwähnt worden. Einerseits existiert eine Reihe von Modellen, die in unterschiedlichen Schritten des Entwicklungsprozesses entstanden sind. Andererseits erweist sich deren Nutzung für Zwecke der Systemsimulation – auch auf Grund der Vielfalt der eingesetzten Lösungen – und unter Umständen schon ihr Auffinden teilweise als schwierig. So besteht der – zumindest subjektive – Wunsch nach etwas mehr Systematik als Grundlage für formalisierte Beschreibungen, die dann auch dazu beitragen können, durch automatisierte Lösungen, den teilweise hohen Aufwand zu reduzieren. Folgende Möglichkeiten erscheinen u.a. als zweckmäßig:

Eine formale Beschreibung von Ausgangsdaten für die Modellierung, Klassifizierung von Modellanforderungen und Modellierungsansätzen kann helfen, den Aufwand bei der Erstellung von Modellen zu reduzieren und gleichzeitig dazu beitragen, die Suche nach existierenden Modellen zu erleichtern. Ein Ansatz könnte in diesem Zusammenhang sein zu prüfen, ob sich für diese Fragestellung vorhandene XML Schemata erweitern lassen oder die Definition eines neuen sinnvoll ist.

Allgemein akzeptierte Modellbibliotheken können sich als wichtiges Hilfsmittel beim Modellaustausch erweisen. Die Definition von Bestandteilen derartiger Bibliotheken, speziell auch für die Systemsimulation im Automobilbau, kann dazu beitragen, Werkzeuge für den automatisierten Übergang von einer für den Komponentenentwurf genutzten Abstraktionsebene auf eine für die Systemsimulation zweckmäßige verfügbar zu machen.

Für eine Transformation von Modellen erscheint als Ausgangspunkt die Beschreibung der mathematischen Zusammenhänge zwischen den Klemmengrößen ein vielversprechender Ansatz. Die Konsequenzen von Standardisierungsbestrebungen wie FMI für die Nutzung im Umfeld anderer Modellbeschreibungssprachen wie VHDL-AMS bedürfen aber noch einer näheren Untersuchung.

Die dargestellte Sicht ist sicher sehr subjektiv geprägt, kann aber vielleicht trotzdem Ausgangspunkt für weitere Überlegungen und Diskussionen, auch im Rahmen der ASIM, sein.

Literatur

- [1] Stachowiak, H.: Allgemeine Modelltheorie. Springer-Verlag, 1973
- [2] Faure, C.; Gaid, M. B. et al: Methods for real-time simulation of Cyber-Physical Systems : application to automotive domain. 7th Int. Wireless and Mobile Computing Conference (IWCMC), 2011, pp. 1105-1110. DOI 10.1109/IWCMC.2011.5982695
- [3] Schwarz, P.: Requirements for the Simulation of Complex Heterogeneous Systems. 8th Modelica Conference Dresden, 2011. Online: https://www.modelica.org/events/modelica2011/Proceedings/pages/keynote/keynote_schwarz.pdf

- [4] AMS assertion subcommittee des Accellera Verilog-AMS Technical Subcommittee.
Online: <http://www.eda.org/twiki/bin/view.cgi/VerilogAMS/AmsAssertions>
- [5] Abdallah, A.; Feron, E. M. et al: Hardware/Software Codesign for Aerospace and Automotive Systems. Proc. of the IEEE, vol. 98 (2010), issue 4, pp. 584 – 602.
DOI 10.1109/JPROC.2009.2036747
- [6] Derler, P.; Lee, E. A.; Sangiovanni-Vincentelli, A.: Modeling Cyber-Physical Systems. Proc. of the IEEE, vol. 100 (2012), issue 1, pp. 13-28.
DOI 10.1109/JPROC.2011.2160929
- [7] Lee, E.A.: Cyber Physical Systems: Design Challenges. 11th IEEE International Symposium on Object Oriented Real-Time Distributed Computing (ISORC), 2008, pp. 363 – 369. DOI 10.1109/ISORC.2008.25 (auch verfügbar als Technical Report UCB/EECS-2008-8, University of California at Berkeley, 2008)
- [8] Cook, J.: Introduction: Cyber-physical systems and the twenty-first century automobile. NSF Workshop On Cyber-Physical Systems, October 2006.
Online: <http://varma.ece.cmu.edu/CPS/Position-Papers/Ford.pdf>
- [9] Senger, P.; Dölling, R.; Rosenstiel, W.: Overview of an environment for automated electrical behavioural modelling using data-based methods. Proc. 2010 Conference on Research in Microelectronics and Electronics (PRIME), 2010, pp. 1-5.
- [10] SAE J2546: Model Specification process Standard. SAE Electronic Design Automation Standards Committee, 2002. Online: http://standards.sae.org/j2546_200202/
- [11] Richtlinien für die Entwicklung von VHDL-AMS-Modellbibliotheken. In: Modellbasierte Systementwicklung. FAT-Schriftenreihe Nr. 226, 2009. Online: http://www.vda.de/de/publikationen/publikationen_downloads/detail.php?id=781
- [12] Erstellung einer VHDL-AMS-Modellbibliothek für die Simulation von Kfz-Systemen. FAT-Schriftenreihe Nr. 207, 2006. Online: http://www.vda.de/de/publikationen/publikationen_downloads/detail.php?id=897
- [13] Common Model eXchange Format (CMX) Version Beta 1.1.0, 2006.
Online: <http://cmx.sourceforge.net/>
- [14] Functional Mockup Interface (FMI) definition.
Online: <http://www.modelisar.com/>
- [15] TISC Suite – Software zur Kopplung mehrerer Simulationswerkzeuge. TLK Thermo GmbH Braunschweig.
Online über: <http://www.tlk-thermo.com/>
- [16] Functional Digital Mock-Up FDMU. Projekt der Fraunhofer-Gesellschaft e.V.
Online: <http://www-past.igd.fraunhofer.de/igd-a2/fdmu/>
- [17] System Architect Designer SyAD: A tool for system simulation and design. CISC Semiconductor GmbH Klagenfurt.
Online: <http://www.cisc.at/system-architect-designer-2.html>

Modellbeispiele mechatronischer Systeme

Dr. Heinz-Theo Mammen, Hella KGaA Hueck & Co.

Heinz-Theo.Mammen@hella.com

Dr. Udo Buschmann, Hella KGaA Hueck & Co.

Udo.Buschmann@hella.com

Zusammenfassung

Im Automotive-Bereich gewinnt die Simulation mechatronischer Systeme immer mehr an Bedeutung, da damit über alle Phasen des Produktentstehungsprozesses hinweg, von der Problemstellung bis zur Produktion, eine zusätzliche Absicherung erfolgt. Es können Kosten eingespart, Zeitaufwände reduziert und Aussagen zu Systemeigenschaften getroffen werden.

In diesem Beitrag werden Beispiele aus dem Automotive-Bereich vorgestellt. Hierbei handelt es sich um das Systemmodell eines Fensterhebers und das eines Gleichstrom-Stellantriebes. Die Modelle wurden auf Basis der Modellbeschreibungssprache Modelica entwickelt, die Simulation erfolgte mit der mechatronischen Entwicklungsumgebung Dymola.

1 Einleitung

Modellierungs- und Simulationstechniken sind im Automotive-Bereich in den letzten Jahren zu einem festen Bestandteil im Entwicklungsprozess geworden. Die Simulation bietet ideale Hilfen zum Verständnis und zur Optimierung von Systemen, die aus mehreren Teilsystemen bestehen und deren Gesamtverhalten von der Wechselwirkung der Teilsysteme bestimmt wird. Bei der Systemsimulation wird das Gesamtsystem zunächst in seine Teilsysteme zerlegt und das physikalische Verhalten jedes Teilsystems in einem eigenen Modell beschrieben. Deswegen ist der erste Schritt einer Simulation stets die Modellbildung.

Die mit dem Modell erzielten Simulationsergebnisse können für Rückschlüsse auf das korrespondierende reale System und den damit verbundenen Problemen genutzt werden. Einmal formulierte Modelle können mit anderen Modellen kombiniert und somit wiederverwendet werden. Voraussetzung dafür ist, dass bei allen Modellen eine einheitliche Modellbeschreibungssprache (z.B. Modelica [1] oder VHDL-AMS [4/5]) und kompatible Schnittstellen verwendet werden. Durch Kombination verschiedener Modelle können zunehmend komplexere Systemmodelle aufgebaut werden, deren Verhalten insgesamt durch die Systemsimulation analysiert werden kann.

2 Beispiele aus dem Automotive-Bereich

Im Folgenden werden zwei Modellbeispiele vorgestellt. Hierbei handelt es sich um ein Systemmodell eines Fensterhebers zur Entwicklung und Optimierung von Einklemmalgorithmen, sowie eines Gleichstrom-Stellantriebes für Scheinwerfer zur Untersuchung des Temperaturverhaltens sowohl im Betriebsfall als auch im Blockierfall.

2.1 Systemmodell Fensterheber

Beim mechatronischen System eines Kfz-Fensterhebers handelt es sich um ein interdisziplinäres System, das die Disziplinen Mechanik, Elektronik und Steuerung miteinander verbindet und den Menschen als Teil des Systems mit berücksichtigt. Der Aufbau des Systems besteht aus Aktorik (Handkurbel oder Motorantrieb), Mechanik (Seil- oder Scherensystem) und Fensterscheibe.

Bei den Fensterhebersystemen wird unterschieden zwischen mechanisch (vorrangig bei den hinteren Fahrzeugtüren) und elektrisch angetriebenen Fensterhebern. Im Folgenden wird bzgl. der Modellierung auf die elektrisch angetriebenen Fensterheber eingegangen. Bei diesen Systemen wird mit Hilfe eines über ein Steuergerät angesteuerten Elektromotors die Hebemechanik in Bewegung gesetzt. Die Komponente Fensterscheibe wird mit Hilfe der Hebemechanik in eine Auf- oder Abwärtsbewegung versetzt, die Scheibe wird dabei im Türrahmen geführt.

Da es sich beim Fensterheber um ein mechatronisches System handelt, wurde zur Modellierung und Simulation ein Werkzeug eingesetzt, das sowohl den mechanischen als auch den elektrischen Teil abdeckt. Hierbei handelt es sich um die mechatronische Entwicklungsumgebung Dymola[®], deren Modellierungsphilosophie auf dem Prinzip der objektorientierten Modellierung basiert [2].

Die für das Beispiel relevanten Merkmale lassen sich kurz wie folgt charakterisieren:

- *Physiknahe, domänenübergreifende Modellierung:* Der Aufbau des realen (heterogenen) Systems findet sich in der gleichen Weise im Modell wieder. Die Verknüpfung der Teilmodelle erfolgt durch explizite Modellierung der (physikalischen) Kopplungen.
- *Mehrkörpermodellierung und Animation.* In der gleichen Weise lassen sich auch Mehrkörpersysteme (MKS) aufbauen und direkt in ihren Bewegungsabläufen visualisieren.
- *Implementierung eigener Modelle und Bibliotheken.* Für viele Fachdisziplinen sind unter Dymola[®] bereits umfangreiche Modell-Bibliotheken verfügbar. Darüber hinaus können aber auch eigene Modelle und Bibliotheken aufgebaut werden. Als Modellierungssprache wird Modelica verwendet [1].

Der Fensterheber ließ sich mit Hilfe von Dymola[®] relativ einfach modellieren. Zur Beschreibung des Fensterrahmens mit Scheibe, der Haft- und Gleitreibungsvorgänge im Getriebe sowie des Einklemmszenarios konnten Standardelemente aus der Modelica-Bibliothek verwendet werden.

Das entwickelte Systemmodell eines Fensterhebers (siehe Bild 1) dient unter anderem dazu, Systemverständnis aufzubauen, sowie auf einfache Weise Parameterstudien und reproduzierbare Tests durchführen zu können.

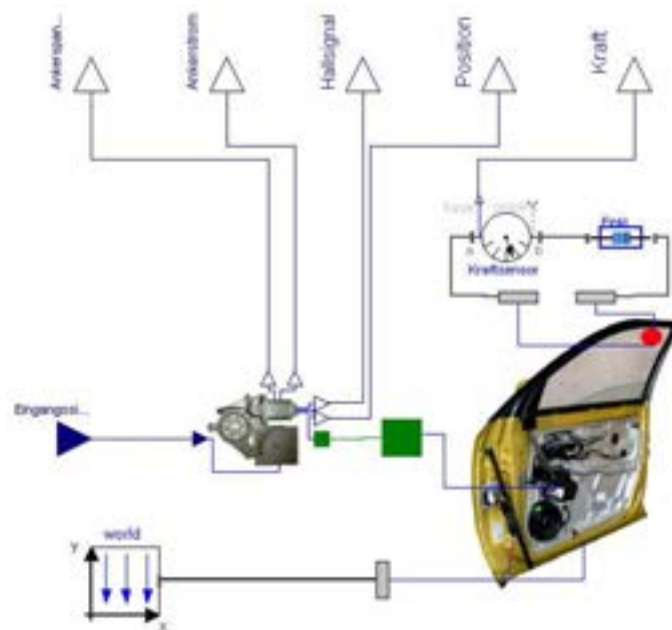


Bild 1: Dymola-Modellaufbau des Fensterhebers

Dieses Systemmodell dient weiterhin als Umgebungsmodell im Rahmen der Steuergeräteentwicklung. Dazu ist es notwendig, dass das Umgebungsmodell mit dem Steuergerätemodell, das z.B. mit Hilfe des Werkzeuges Simulink[®] entwickelt werden kann, kombinierbar ist (z.B. durch Co-Simulation oder Modellimport). Im vorliegenden Fall wurde das entwickelte Dymola-Systemmodell automatisch in eine C-Funktion kompiliert und in Matlab/Simulink [8] als Teilmodell integriert. Dort wurde es dann um das Modell der Steuerung ergänzt, für das (nach dem Test der Gesamtanordnung) automatisch C-Code für die Integration in das reale Steuergerät generiert werden kann.

Bei der Entwicklung der Steuergerätefunktionalität für den Fensterheber ist es notwendig, den Einklemmschutz als wichtige Funktion mit zu berücksichtigen, also auch den Faktor Mensch (z.B. Einklemmen einer Hand beim Schließen der Scheibe). Im Dymola-Systemmodell wurde dem durch ein Einklemmmodell Rechnung getragen. Diese Komponente wurde auf Basis der gesetzlichen Vorgaben für einen Einklemmfall parametrisiert.

Einen wichtigen Schritt in der modellbasierten Systementwicklung stellt die Validierung des Modells dar. Dazu wird, wenn möglich, das Modell so parametrisiert, dass die Ergebnisse aus dem Modell mit gemessenen Werten aus der Realität möglichst gut übereinstimmen. Bild 2 zeigt das Modellverhalten im Vergleich zur Messung beim Verfahren der Scheibe.

Wie Bild 2 zu entnehmen ist, stimmen Messung und Simulation gut überein, so dass das entwickelte Modell unter anderem für Parameterstudien und zur Untersuchung der Mensch/System-Schnittstelle eingesetzt werden kann.

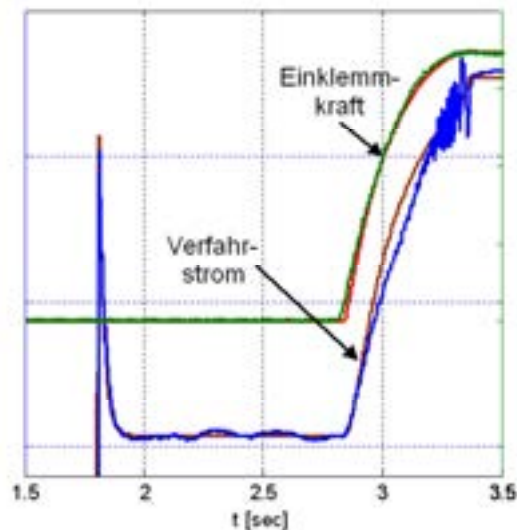


Bild 2: Vergleich Simulation/Messung: Hochfahren der Fahrzeugtürscheibe zur Ermittlung der maximalen Einklemmkräfte (— — Messung; — — Simulation)

2.2 Modell eines Gleichstrom-Stellantriebes

Der Trend zu immer mehr mechatronischen Komponenten im Fahrzeug ist beispielhaft am Scheinwerfer erkennbar. Wo früher Leuchtmittel mit Reflektor in einem einfachen Gehäuse für eine Ausleuchtung der Straße gesorgt haben, werden heute komplexe mechatronische Scheinwerfersysteme eingesetzt. Diese Systeme können ein dynamisches Kurvenlicht beinhalten, über eine automatische Leuchtweitenregelung verfügen, für eine angepasste Ausleuchtung je nach Fahrsituation (innerorts, Landstraße, Autobahn) sorgen und ähnliche weitere Funktionen beinhalten.

Da diese mechatronischen Systeme immer komplexer werden, ist eine ganzheitliche Entwicklung sinnvoll und notwendig. Unterstützt werden kann diese Entwicklung durch eine entwurfsbegleitende Modellierung von der Problemstellung bis zur Serienreife.

Der Gleichstrom-Stellantrieb ist eine Teilkomponente des mechatronischen Systems Scheinwerfer. Diese Teilkomponente stellt für sich auch bereits ein mechatronisches System dar, bestehend aus einer Ansteuerung, einer elektrischen Verstellkomponente und einem Getriebe inklusive eines Verstellarms. Bild 3 zeigt den Stellantrieb eines mechatronischen Scheinwerfermoduls.



Bild 3: Gleichstrom-Stellantrieb eines Scheinwerfermoduls

Das zu diesem mechatronischen System entwickelte Modell wurde ebenfalls auf Basis der Modellbeschreibungssprache Modelica [1] und dem Simulator Dymola [2] entwickelt. Die Ansteuerung dieses Gleichstrom-Stellantriebes besteht im Wesentlichen aus zwei Halbbrücken und einer Logik zur Steuerung dieser Halbbrücken. Das Motormodell beschreibt das elektrische und mechanische Verhalten eines Gleichstrom-Motors. Weiterhin wird mit dem Modell das thermische Verhalten eines Motors nachgebildet, d.h. es wird einerseits die Umgebungstemperatur des Motors berücksichtigt und andererseits die Eigenerwärmung im Betrieb. Darüber hinaus wird das Kommutierungsverhalten des Motors beschrieben. Mit dem Getriebemodell werden das Übersetzungsverhältnis, die Anschläge für Hin- und Rücklauf sowie die Reibung innerhalb dieses Teilsystems formuliert.

Neben der Modellbildung spielt die Modellvalidierung eine wichtige Rolle, damit am Ende des Prozesses ein Modell zur Verfügung steht, das für Untersuchungen während des gesamten Entwicklungspfad es „uneingeschränkt“ genutzt werden kann. Die Parametrisierung des Modells erfolgte hierbei auf Basis von Datenblattangaben und messtechnischen Untersuchungen.

In Bild 4 ist beispielhaft ein Vergleich zwischen Messung und Simulation dargestellt. Es wurde der Ankerstrom des Motors betrachtet und sowohl die simulierte als auch die gemessene Größe aufgetragen.

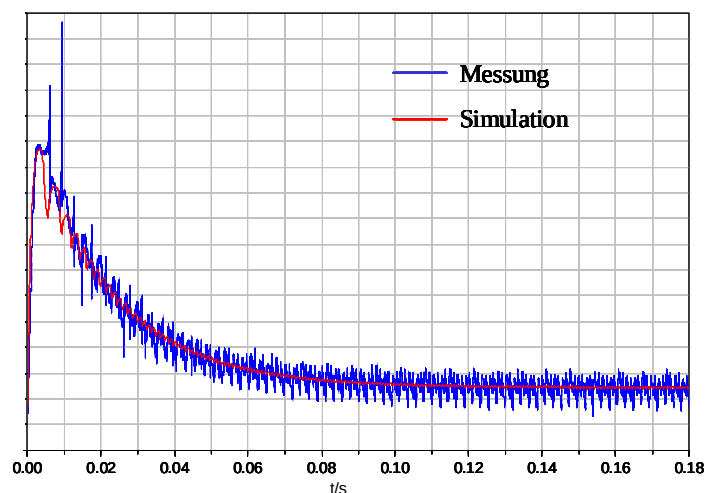


Bild 4: Ankerstrom des Motormodells – Vergleich Messung / Simulation

Der Vergleich zwischen Messung und Simulation zeigt eine gute Übereinstimmung der Ergebnisse. Weitere Simulationen unter verschiedenen Belastungszuständen haben dieses Ergebnis bestätigt. Das Modell ist validiert und somit geeignet, Untersuchungen am virtuellen System durchzuführen und die Ergebnisse für die Entwicklung des realen Systems zu verwenden.

3 Zusammenfassung

Die Systemsimulation birgt viele Vorteile, da über alle Phasen des Produktentstehungsprozesses hinweg von der Problemstellung bis zur Produktion eine zusätzliche Absicherung erfolgt. Es können Kosten eingespart, Zeitaufwände reduziert und

Aussagen zu Systemeigenschaften getroffen werden. Allein durch Hardware-Aufbauten ist dies nicht oder nur mit erheblichem Aufwand möglich (z.B. Kraftmessungen in einem gekapselten System).

Die Modellbeispiele Fensterheber und Stellantrieb haben gezeigt, dass mit der Simulation Kundenanforderungen schnell überprüft, kritische Betriebsfälle simulativ abgebildet sowie Parameter- und Aufbaustudien leicht durchgeführt werden können. Die dabei gewonnenen Erkenntnisse tragen zu einem umfassenderen Systemverständnis bei. Entwicklungsfehler, die gerade in frühen Entwicklungsphasen fatale Folgen haben können, lassen sich auf diese Weise reduzieren.

Ein weiterer Vorteil der Systemsimulation ist in der Reproduzierbarkeit der Versuchsbedingungen zu sehen. Im Gegensatz zu Untersuchungen an realen Prototypen ist eine exakte Wiederholung beliebiger Prüfzyklen durch Simulation jederzeit möglich. Insbesondere für die spätere Testphase ist dies ein wichtiger Punkt.

Bezogen auf den Entwurfsablauf lassen sich Systemmodelle auch als ausführbare Lastenhefte auffassen. Auf diese Weise dokumentiert sich der Entwicklungsprozess mehr oder weniger von selbst. Alle Experimente lassen sich auch im Nachhinein reproduzierbar nachvollziehen. Dies erleichtert den Nachweis von Fehlerursachen und vereinfacht überdies die nachträgliche Durchführung von Änderungen.

Literatur

- [1] Modelica Association (2007): *Modelica – A Unified Object-Oriented Language for Physical Systems Modeling*, Language Specification, Version 3.0.,
<http://www.modelica.org/documents/ModelicaSpec30.pdf>.
- [2] Dymola: *Multi-Engineering Modeling and Simulation*, www.dynasim.se,
www.dymola.com.
- [3] VDI 2206: *Entwicklungsmethodik für mechatronische Systeme*, Beuth-Verlag, Berlin 2003.
- [4] Gerke, Th.; Hessel, E.: *Modelle werden kompatibel*, Elektronik automotive 1/2, 2009, S. 58 - 60.
- [5] Commerell, W.; Mammen, H.-T.; Panreck, K.; Haase, J.: *Simulation technischer Systeme – Anforderungen und Perspektiven*, 13. ASIM-Fachtagung, Simulation in Produktion und Logistik, 1./2. Oktober 2008.
- [6] Bode, H.:
Matlab-Simulink – Analyse und Simulation dynamischer Systeme, Verlag: Teubner B. G. GmbH, 2. Auflage, 09/2006

Erstellung einer firmenweiten Modellbibliothek - Potentiale, Anforderungen und Herausforderungen

Armin Schön, Continental

Armin.Schoen@continental-corporation.com

Zusammenfassung

Eine firmenübergreifende Bibliothek für Simulationsmodelle mit Fokus auf Elektronik- und Systemsimulation bietet viele Vorteile und Möglichkeiten. Aber auch der darin enthaltene Aufwand darf nicht unterschätzt werden. Im Folgenden sollen verschiedene Aspekte beleuchtet werden. Die vorgestellten Aussagen sind, soweit möglich, allgemein gehalten. Diese Ausführungen sind kein fester, eingeführter Prozess, sondern vielmehr generelle Betrachtungsweisen.

1 Anforderungen an eine Modellbibliothek

1.1 Anforderungen aus Sicht der Anwender und Projekte

Im Bereich der Elektroniksimulation gibt es eine große Zahl verschiedenster Analysen, die von den benötigten Modellen unterstützt werden müssen.

Große Designs mit vielen hundert Komponenten müssen auf ihre Funktionalität hin überprüft und optimiert werden. Ebenso müssen aber statistische Betrachtungen des Designs möglich sein, um die notwendigen Qualitätsstandards zu erreichen. In kritischen Umgebungen oder bei Leistungskomponenten ist auch eine entsprechende thermische Untersuchung notwendig.

Bei heute üblichen Arbeitsfrequenzen und Schaltgeschwindigkeiten muss zum Teil noch genauer auf die Details geachtet werden. In die Simulation mit einbezogen sind hier die Themen Signalintegrität und Elektromagnetische Verträglichkeit.

Als Systemarchitekt benötigt man desweiteren die Möglichkeit, die Untersuchungen auf einer abstrakteren Ebene durchzuführen. Hier ist eine Simulation auf Komponentenebene nicht mehr durchführbar und auch nicht sinnvoll. Es müssen also verhaltensbasierende Modelle für die Untersysteme bzw. Module verfügbar sein.

Mit der Modellierung im Entwicklungsbereich ist aber noch nicht das Ende erreicht. Kunden wollen das Subsystem auch im Zusammenspiel mit Komponenten von anderen Lieferanten in Simulationen verifizieren. Hierzu sind Modelle notwendig, die das Verhalten an den

Schnittstellen detailliert beschreiben, aber keine Rückschlüsse auf die technischen Details erlauben.

Viele dieser Anforderungen sind in verschiedensten Entwicklungsbereichen zu finden. Aber dennoch sind für die Auslegung eines Antriebsstranges andere Modelle notwendig als für Entwicklungen auf Leiterplatten- oder gar ASIC-Ebene. Bereits hieraus wird ersichtlich, dass eine flexible Struktur einer Bibliothek benötigt wird.

Neben den Anforderungen aus unterschiedlichen Entwicklungsstadien und –bereichen gibt es aber auch noch die Produktgruppen, die aufgrund von Sicherheitsanforderungen erhöhte Qualitätsansprüche an Modelle und Simulationsumgebungen haben. Während eine nicht entdeckte Designschwäche in einem Multimediasystem „nur“ Unannehmlichkeiten bereitet, kann eine selbige in einem sicherheitsrelevanten System, wie beispielsweise Airbags oder Bremsen, Menschenleben kosten.

1.2 Anforderungen aus firmenstrategischer Sicht

Abhängig von der Organisation innerhalb eines Konzerns ergeben sich Anforderungen an die Anwendungslandschaft. Es ist nicht wirtschaftlich, für jede Abteilung oder gar für jedes Projekt individuelle Software zu beschaffen. Durch eine strukturierte Landschaft und ein überlegtes Softwareportfolio ergibt sich eine Reihe von Vorteilen:

- Höheres Einkaufsvolumen erlaubt wirtschaftliche Preisgestaltung.
- Durch den Aufbau von globalen oder regionalen Lizenzpools ergibt sich eine bessere prozentuale Auslastung der einzelnen Lizenzen.
- Eine einheitliche IT-Landschaft reduziert den administrativen Aufwand.

Alle diese Punkte ermöglichen erst den produktiven Einsatz einer übergreifenden Bibliothek. Es gibt aber auch andere Randparameter, die die Gestaltung beeinflussen. Dazu gehören z.B. auch Kundenanforderungen oder firmeninterne Prozesse und Einrichtungen wie Datenmanagement oder Netzwerkinfrastrukturen. Auch diese müssen bei der Realisierung berücksichtigt werden.

Eine Bibliothek zu definieren und auch zu implementieren wird sicherlich nicht alle Anforderungen zu 100% erfüllen können. Dennoch ist es wichtig, die verschiedenen Aspekte zu beleuchten und in Priorität und Aufwand zu bewerten.

2 Anforderungen an eine Modellbibliothek

2.1 Auswahl einer Modellierungssprache

Auf dem Markt gibt es unterschiedliche Modellierungssprachen, die alle ihre Daseinsberechtigung haben, von denen aber einige nur bedingt für eine Bibliothek geeignet sind.

- Die Sprache sollte sowohl detaillierte physikalische Gleichungen abbilden können als auch Strukturen zur Verhaltensmodellierung beherrschen.

- Um im Bereich der Elektronik effektiv simulieren zu können sind konservative Netzwerke unabdingbar. Um auf höheren Abstraktionslevels zu arbeiten sind allerdings auch Signalflussbeschreibungen und funktionale Blöcke notwendig.
- Um Modellaustausch zu ermöglichen sollte die Sprache von möglichst vielen Simulatoren verwendet werden können. Ein offener Standard wäre hier also vorteilhaft.
- Es müssen unterschiedliche physikalische Domänen beschrieben werden können.
- Um die Anforderungen in der Entwicklung zu erfüllen müssen statistische Verhalten integriert werden können.
- Vorteilhaft wären unterschiedliche Abstraktionsstufen der Beschreibung innerhalb eines Modells.
- Die Modellierungssprache sollte leicht zu erlernen und gut lesbar sein, kurz, sie muss anwenderfreundlich sein.

Hinzu kommen sicherlich noch weitere Anforderungen, um Kundenanforderungen abdecken zu können, das technische Knowhow zu schützen oder individuelle Konzernpräferenzen zu ermöglichen.

Unter diesen Aspekten betrachtet, ist für den Bereich Elektronik- und Systemsimulation die Modellierung in „VHDL-AMS“ wohl am geeignetsten. Aber auch hier gibt es Schwächen. Es bedeutet z.B. einen relativ hohen Aufwand, Toleranzen für statistische Betrachtungen einzuführen und zu definieren. Auch die Notwendigkeit, Simulationen im Zeit- und Frequenzbereich durchzuführen, benötigt zusätzlichen Aufwand. Die Möglichkeiten der Sprache und deren dadurch notwendigen Algorithmen in den Simulatoren haben auch negative Auswirkungen auf die Simulationszeiten.

Aber durch die ansonsten weitreichende Abdeckung der oben genannten Punkte erscheint diese Wahl als die Richtige. Variierende Anforderungen und andere strategische Ausrichtungen können aber durchaus zu einem anderen Ergebnis führen.

2.2 Definition des Modellinhalts

Die Sprache „VHDL-AMS“ bietet die Möglichkeit, verschiedene Beschreibungen, eine sogenannte „architecture“, mit einer Umgebungsbeschreibung („entity“) zu kombinieren. Dadurch wird es möglich, eine Beschreibungsebene auszuwählen, die möglichst gut zum derzeitigen Ziel der Simulation passt. Verschiedene Komplexitätslevels

können also in einem Modell untergebracht werden. Der Anwender hat die Möglichkeit anhand von Parametern für sein Design (oder auch einzelne Komponenten) bestimmte Abstraktionslevel zu wählen um die notwendigen Simulationen durchzuführen.

Sollten die in der Bibliothek enthaltenen Level nicht für spezielle Anforderungen geeignet sein, kann der Benutzer auch eigene Beschreibungen dem Modell hinzufügen.

In Tabelle 1 sind Level definiert, die einen Großteil der Anforderungen abdecken können.

Level	Abgedeckte Funktionalität
1: Nominal	• Funktionale Simulation von komplexen Designs • Optimieren von Designparametern • Schnelle Konzeptverifikation
2: Extended	• Untersuchung des statistischen Verhaltens (Monte Carlo, Worst Case) • Detaillierte Untersuchung unter Berücksichtigung parasitärer Effekte • Weiterführende Optimierung
3: Thermal	• Detaillierte thermische Effekte • Selbsterwärmung von Leistungskomponenten • Kühlauslegung unter verschiedenen Umgebungsbedingungen
4: High Frequency	• Verifikation der Signalintegrität • Untersuchung der elektromagnetischen Verträglichkeit

Tabelle 1: Abstraktionslevel der Simulationsmodelle

2.3 Verwaltung der Bibliothek

Für eine konzernweite Bibliothek ist Datenmanagement unabdingbar. Es muss eine Versionierung der Modelle erfolgen, um Änderungen bzw. Simulationsergebnisse aus früheren Läufen nachvollziehen zu können.

Es ist auch sicher zu stellen, dass die Anwender von Modellen über Änderungen informiert werden, um eventuelle Einflüsse auf Ergebnisse oder gar Designparameter verifizieren zu können.

Über Datenbanken und integrierte Synchronisationsmechanismen muss auch sichergestellt werden, dass globale Entwicklungsteams mit identischen Modellen und Versionen arbeiten. Anderenfalls besteht die Gefahr von Inkonsistenzen und daraus resultierenden Fehlentscheidungen.

Weiterhin ist bei der Replikation auch die Netzwerkperformance zu berücksichtigen. Aus Gründen der Arbeitsgeschwindigkeit ist eine lokale Verfügbarkeit der Bibliothek auf dem ausführenden System meistens vorteilhaft.

Um die Qualität der Modelle, die über die Bibliothek verteilt werden, sicherstellen zu können, sind weiterhin verschiedene Prozesse notwendig. Es soll zwar für den Anwender möglich sein, Modelle für seine Zwecke anzupassen, zu ändern oder zu ergänzen, diese Änderungen müssen aber lokal erfolgen und dürfen nicht direkt in die offizielle Datenstruktur zurück gelangen. Sollte dies erwünscht sein, dann ist der Weg über das firmeninterne Team zur Bibliotheksverwaltung einzuhalten. In Abbildung 1 ist ein möglicher Replikationsweg für Simulationsmodelle aufgezeigt.

Je nach Konfiguration des Firmennetzwerkes kann es vorteilhaft sein, auf verschiedenen regionalen Servern die Datenbank / Modellbibliothek für die Synchronisation der Anwender PCs vorzuhalten. Erfahrungsgemäß ist eine Rücksynchronisation problematisch. Wenn beispielsweise zwei Mitarbeiter gleichzeitig das gleiche Modell bearbeiten möchten, ist eine eindeutige Replikation nicht mehr sicherzustellen. Desweiteren wird das Thema Versionierung und Nachvollziehbarkeit der Änderungen (Change Management) nicht

sichergestellt. Daher sind für das Release vorgesehene Modelle in einer Master-Datenbank zu pflegen. Weitere Replikationsserver synchronisieren ausschließlich mit diesem Master.

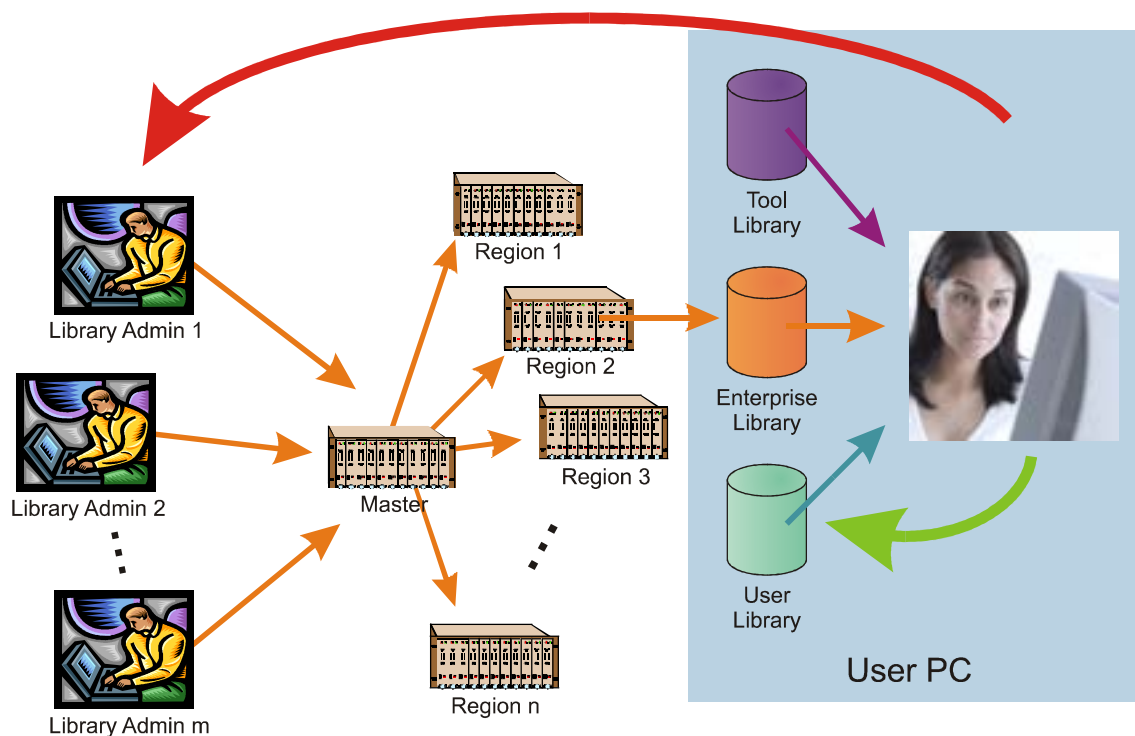


Abbildung 1: Replikation einer Firmenbibliothek

3 Wartung einer Firmenbibliothek

3.1 Interne Mitarbeiter

Um eine eingeführte Bibliothek verwendbar zu halten, ist ein gewisser Aufwand notwendig. Ein relativ großer Anteil dieses Aufwands entfällt auf interne Mitarbeiter. Anhand der Anforderungen von Projekten und Firmenbereichen müssen existierende Modelle angepasst oder Neue erstellt werden. Außerdem ist entsprechendes Wissen notwendig, um Anwender und Modellierer in den Entwicklungsbereichen bei ihren Tätigkeiten zu unterstützen.

Externe Modelle von Zulieferern und in Auftrag gegebene Modelle müssen verifiziert und eventuell leicht angepasst werden, um ins Konzept zu passen.

Ebenso muss es eine Stelle geben, die Ansprechpartner für Modelllieferanten sowie den internen Anwendern ist.

Hinzu kommen die ganzen administrativen Tätigkeiten. Als wichtigstes ist hier das Datenmanagement zu erwähnen. Dadurch wird Versionierung, Nachverfolgbarkeit und Archivierung sichergestellt. Bei Ausfällen im Feld ist unter Umständen noch viele Jahre nach Abschluss des Entwicklungsprojektes eine Nachvollziehbarkeit der Entwicklungsprozesse notwendig. Hierzu müssen die Modelle in der ursprünglich verwendeten Version vorhanden sein.

Des Weiteren folgen die zeitnahe Verteilung sowie die Einhaltung der geforderten Prozesse zur Qualitätssicherung.

3.2 Externe Modellquellen

Aus Aufwandsgründen ist es nicht möglich, alle Modelle firmenintern zu erstellen. Hier sind entsprechende Quellen von außerhalb notwendig.

Bereits im Einkaufsprozess für Zulieferer sollte festgelegt werden, dass bei der Lieferung von Subsystemen oder Komponenten entsprechende Modelle beinhaltet sein müssen. Hierzu muss es Spezifikationen geben, die die Anforderungen klar beschreiben. Ziel ist es auch, firmenübergreifend ein gemeinsames Verständnis zu finden, welches es den Zulieferfirmen ermöglicht, die Modelle so einheitlich wie möglich zu generieren. Um solche Anforderungen zu definieren eignen sich gemeinsame Verbände wie beispielsweise der VDA (Verband der deutschen Automobilindustrie) in Deutschland oder auch die AIAG (Automotive Industry Action Group) in den Vereinigten Staaten.

Modelle können auch bei Institutionen, Universitäten und anderen Firmen in Auftrag gegeben werden.

Nicht zuletzt entstehen auch in den Entwicklungsbereichen Modelle für spezifische Komponenten oder Anforderungen. Auch solche Modelle können in einer Firmenbibliothek der Allgemeinheit zur Verfügung gestellt werden.

4 Zusammenfassung

Eine firmeninterne Bibliothek hat nicht von der Hand zu weisende Vorteile. Die Entwicklungsingenieure können sich auf die wesentliche Arbeit konzentrieren und müssen nicht zunächst nach passenden Modellen suchen und diese auch noch verifizieren. Ebenso stellt eine solche Bibliothek eine zuverlässige Möglichkeit dar, die rechtlichen Anforderungen an Archivierung und Änderungsverfolgung zu garantieren. Ein einheitliches Verständnis von Modellstruktur und –inhalt erleichtert es Lieferanten von Modellen außerdem, für verschiedene Kunden passende Modelle mit verringertem Aufwand zu schicken.

Es kann aber auch nicht ignoriert werden, dass ein hoher Aufwand für Erstellung und Pflege notwendig ist. Hier ist auch schwer zu messen, welche Kosten effektiv in den verschiedenen Abteilungen eingespart werden können. Dadurch wird es für die umsetzende Abteilung auch nicht einfach, den Nutzen für ein derartiges Projekt anschaulich darzustellen.

Auch wird es kaum möglich sein, alle Anforderungen und Wünsche der Anwender und deren Projekte zu erfüllen. Es muss versucht werden, einen gemeinsamen Nenner zu finden, der die meisten Bedürfnisse erfüllt.

Insgesamt ist zu sagen, dass die Vorteile den Aufwand überwiegen und ein derartiges Projekt viele Vorteile für die Entwicklungsabteilungen eines Konzerns bringt.

BroSAnT – an example of the mechatronical way of thinking

Petkun, Sergey

Brose Fahrzeugteile GmbH & Co. Kommanditgesellschaft, Hallstadt

Sergey.Petkun@brose.com

Zhao, Xin

Brose Fahrzeugteile GmbH & Co. Kommanditgesellschaft, Hallstadt

Xin.Zhao.temp2@brose.com

Abstract

The desire of the modelling of mechatronics system is daily growing. But already at the formulation of aims such simulation there are many different meaning what should deliver system simulation as resulting output. It is connected together with system responsibility on whole system. Mechatronics systems are naturally domain distributed systems and the main interest on system simulation is at the acquisition phase (concept assurance) and at the final phase (functional application the whole system). At the beginning the question is "Which components should be taken"? , at last phase will be asked "Why it doesn't work how we have imagined it?". The question, usually coming from management "How can I spare money with system simulation?", should be transferred in another one "Can I reach the required quality without system simulation?".

Very often by system modelling there is not enough information about components at project beginning, at the end of it - vice versa: too much information. The next system simulation feature is in it's naturally multi domain character. The boundary condition between domains is known problem almost in every companies. As consequence for system simulation is its membership and place in simulation infrastructure remains open.

The system analysis has character of multi criteria because the model behaviour will be estimated from different point of views: mechanical, electrical and etc. from such point of view, **BroSAnT** (Brose System Analyse Tool) tries to realise the new conceptual solution. The origin of false thinking is the meaning that the system modelling can and should be made like simple joining the modelling of specific domains. In our opinion, only from system view, the necessary modelling level of corresponding domain counterpart can be right determined.

1 Introduction

Everybody, who tries to simulate the whole system, has own imagination what the simulation should do. In reality, the system simulation is concentrated with Know-How about the whole system. Under whole system, is meant some mechatronics system: system, consisting of electrical power source, energy transformer (electrical into mechanical), mechanical executive mechanism and intellectual control system (algorithms realized in software and controlling hardware). There are a number of approaches to simulate the components: solid multi bodies in mechanics like Adams or Simpack, Matlab-Simulink in Algorithms/Software, Spice/VHDL-AMS modelling in hardware. If there are some models of different components in system, the first idea that comes is to combine these available models in whole system. It leads to co-simulation concept. There are a lot of techniques for co-simulation but it is not an optimal way for system analysis. The system analysis should begin from question “is the system appropriate to its purposes?”

During developing mechatronic systems, it is highly recommended to optimize appropriation to constitutive components. Common functionality of whole system is connected with functionality of every component: if the system does not right work – from system point of view, it is not so important which component is defect – the result is clear: the device is defect. It doesn't matter whether electronics fails or mechanics breaks down.

There is a false conviction that with help of advanced software the constructional errors can be corrected. The poor constructed mechanical structure can be brought to functional state, but compelled solution is not the best solution. The optimization on system level has a number of different and very often contradictive criteria. This contradictory consists in reducing price of component on one hand and on the other hand in providing the required functionality. It is not seldom case when the device is defect only because of some trifle fails.

2 Functionality under "all conditions"

Requirements, described in specification, predestine the choice of parameters for mechatronics simulation: all working condition: climatic, constructive, dynamic and energetic (e.g. voltage fluctuation on the power source). It is clear that the whole range of all parameter variations can not be investigated for different reasons but it is known (and partly it is company know-how) which effects should be taken into account during modelling components.

All these questions to simulation can be yet further extended or reduced, depending on situation. For example, for some industrial project with huge number of mass production, the constructive tolerances take great importance which is not so interesting in the case of some single sample.

The most of mechatronic systems have some "adjustable" part in form of algorithms parameter set (firmware parameter part), which allows optimizing the system for certain "working range".

To summarize all these requirements on mechatronics simulation, all input variation can be divided in groups:

- Constructive setting
- Climatic environment
- Dynamic environment
- Energy supply
- Algorithmic setting

The purpose of system is usually described in specification on system. The working environment is usually prescribed in the specification. For mechatronics system, it is, as a rule, ambience, dynamic and constructive tolerances. Under all these conditions, the system should be complied with the purpose determined in specification.

From this list, it can be seen that there are already enough parameters only at system level without taking component level into account. That is why, some flexible cable, working far from large deformation range, can be treated as only force transfer element with certain elasticity properties. So the complex constructive element will be reduced to some functional property as force transfer with definite stiffness.

Such approach is well known since a long time through modelling the electronic schemas where complex electrical behaviour of transistor will be replaced by behaviour model at working point. From system point of view, mechanics is nothing else as force-velocity transformer in dynamical system. From view behavioural modelling, the whole mechanical effects can be described as combination of elastic, inertial and frictional properties, describing these effects.

Elasticity is responsible for recoverable energy transfer, inertia - for time delay by the movement, frictional for energy loss. Naturally, all these properties strongly depend on conditions, mentioned above, but requirement in specification is also based on certain "working range". Replacing the complex mechanical geometry through behavioural description is not simple task but brings enormous advantages with it.

To provide industrial projects with CAE support, the simulation should be "real time" able in sense of, not only fast calculation but also system building, overview- and analysis-able. Nobody needs simulation results after successful start of production. At present, the development of mechatronics system is connected with combination of prefabricated hardware components (electrical, mechanical) available on commercial market and that is why the importance of simulation at system level rises continually. The lack of domain compatible models leads to the development of new modelling languages as Modelica [1] or effort to describe the wanted components of system in term of old one as VHDL-AMS [2]. There is already first effort in standardisation of models on component level [3]. The electronics development sets a good example for future technology of development - the

product is impossible to sell without accompanying model with it. The risk of implementation of inappropriate component will be reduced through virtual assembling test.

Successful and reasonable simulation of whole system is not possible without participation of domain experts in it, because the simulation engineers can not right estimate if the behaviour of the whole system reflects all necessary effects under investigation. The aim of simulation developers is to give appropriate component models with minimal number of parameters but with all necessary effects in it. It leads to the well-known solution: server-client application.

There are not so many problems connected with embedded control algorithms, because usually using platform specific C-code at end application phase allows to bind this code directly as "black box" in simulation model. Almost all simulators can use the "black box" with little adaptation by embedding it in simulation model. This feature is very important for system simulation because it give us some so necessary criteria for estimation how close the rest of model is to reality.

Very important point of system simulation is the ability of model verification. The focus of system analysis is dynamical behaviour, and that is why, the comparison with real measured behaviour must be possible. What can not be measured is not well appropriate for the estimating the quality of model.

The traceability of simulation runs is obligatory by system simulation. The number of parameters, even in the case of simple simulation, is not small and the number of possible investigation variants is very big.

There are some different meanings about GUI (Graphical User Interface) for end user. In our opinion, the same simulation interface should be adapted to end user for specific domains. By analyzing the same system, the experts from different domains have different points of interests to the same object and various analyzing methods: for mechanics the forces are in the focus of analyses, current or voltage can be in the focus of analysing for electronics.

System simulation is in certain sense also documentation for making some conceptual decision. That means the simulation results should be reproducible in a later time point.

The requirements to system simulation depend also on simulation users. System developers have other demands to simulation as system application team.

3 System components

To divide system into components is not complex - it is determined by construction of product. The constituent part of product can be used from self manufacture or bought on the commercial market. The most important feature is that this component can be non-destructively replaced in device. The second one is that the part should possess some self-completed functionality.

The system analysing tool should be based on some data base with component models. It determines the choice of modelling language. It can be some standard language as VHDL-AMS or another popular one as Modelica or at all tool specific language like MAST for simulator Saber (Synopsys). The modelling of components is very important to have, if it is

possible, the similarity to reality by choosing connection points. For example, for drive there are three connection points: two electrical and one mechanical. It gives possibility to use different drive models and exchange them in a light way. It does not matter if the used drive with another parameter set or at all some other models, maybe with another modelling depth. From system point of view, a drive has the main functionality to transform electrical energy into mechanical movement.

This sameness allows the product developer to use different drives by simple replacement of component. Already at modelling level the interface between system components should be clear determined.

There is big risk during determining the depth of component description to go down too deep in the component description. Component level should not be mixed up with system level. Domain experts are trying continually to deep in each own domain. The practical rule is following: main parameters, which are needed for component description on system level, can be found in the specification sheets of this component. It stands to reason that, if the number of parameters is not enough for self-consistent description of component, it means that this specification gives component supplier more freedom and can lead later to problems on the side system responsibility. It is well-known case: the component is complied with its specification but system with it does not work right.

It is obvious that every additional requirement in specification is connected with price or opposition from supplier side but the depth of specification on the product shows "gold" practical middle between customer and supplier and also reflects current industrial state in this manufacturing branch.

Electronics development is leading in development based on CAE techniques and it can be useful to transfer huge practical experience that this domain already has, in other domains. As evidence, it may be mentioned the rate of increasing amount of transistors in modern CPU.

4 BroSAnt objectives

The development of **BroSAnt** is based on multi purpose basic:

- Personal education
- Keeping company know-how
- Part of project documentation
- Advantage in competence over competitors
- Project insurance at acquisition/development phase
- Proofing of new concepts on development phase
- Error analysing in mass production
- Saving manufacture costs by avoiding errors

The personal education in mechatronics area is not a short-period task and many as human so also hardware resources should be involved in it. With the help of virtual system, the learning personal can be quicker and cheaper to reach a higher professional state, which is a necessary condition for successful business today.

With permanent personal education, the company know-how is transferred from experts to younger generation and this know-how should be kept, also taking into account fast changing market situation and development state in the world independent from human resources form. Every effect modelled in the system should be remained in data base of company knowledge independent from human presence. It does not reduce the role of human factor but makes insurance factor higher.

Also corresponding documentation process belongs to consequent project loop. In documentation process, it is important to have some possibility to recover simulation state and results for possible analysis in case of manufacture problems

With rising globalisation in the world, the number of manufactures grows rapidly, almost every month, and the problem to choose the right component supplier becomes of great meaning. One of factors to distinguish among huge proposals on global commercial market is competence degree of the component manufacture. And from this point, a good simulation model is yet "another brick in the wall".

It is not to wonder that wishes are rising quicker than performance. The customer should be consulted not only emotionally ("we can do it") but rather technically (let us analyse the simulation results). Sometimes, the customer requirements are contradictory but without technical analysis it is invisible. The system analysis tool is a good helper in this case to avoid excessive discussion.

Innovation by development is a locomotive of the progress but only flawless development and production lead to commercial success and sure business. To avoid errors by new development is one of the main tasks of every company. The virtual development is one of many steps in this trend. That is why, there are so many simulation concepts proved at present. BroSAnT gives the possibility to prove the technical innovations at early development phase without big expensive or with relative low costs.

Every error in development brings additional costs. However, it is impossible at all to develop without some errors which occur sometimes at different phases of manufacture: it can be connected with some changing of technological process or supplier. Some improvements in the manufacturing tool can have side effects which are not taken into account and so on. Any simulation tool can not find the reason itself but rather it can be served as assistant instrument to indentify cause for this failure. It should be clear, that any tool is only facility but rather "wonder solution" for all problems. The quality of using tool is the same important as the professionalism of the personal working on it and tight cooperation between simulation and domain experts.

It is very important to use simulation analyses at right development phases. The simulation costs also money and time, which means in the end also money.

It is well known that the costs of error correction being close to start of production rise almost exponentially. That is why the failure analysis (failure prediction) at early development or application phases is almost obligatory for every commercial project.

Last but not least are the expenses for simulation tool. To be profitable, the simulation tool should be used as extensive as possible, which is reachable either with many experts, constantly using this tool, or with organization of some calculation task pool for it. The software manufactures are trying to sell as many software as possible, promising to increase the quality of development, but meanwhile they are silent about expanded costs for personal educational, maintenance and so on. The complexity of tools is enhanced and the average efficiency of using the tools, without correct usage, is diminished.

5 BroSAnT concept

From reason above mentioned, the server-client concept is ideally appropriate for realization of system simulation. To give domain experts some possibility to use the simulation tool, it is necessary to hide complexities of simulation and modelling techniques from them. The client part should be independent, as possible as it can, from simulation environment. The focus of analysis should be laid on domain problem but rather simulation nuances.

According to this concept, the grey boxes do not have any environment of simulator and almost do not connect with peculiarities of some simulation knowledge.

During the task preparing for simulation, technical knowledge of simulation object is desirable. The default setting for component is suitable for reasonable functionality of basic model. The parameter changing requires the exact understanding about what will be done.

The most investigations are done in the time range because the dynamical behaviour is the focus of analysing typical mechatronics system.

The client part (Figure 1) is written in Tcl/Tk [4] and principally consists of two parts: results analyser and simulation configurator. The first part presents ordinary two-dimension curve browser with zoom and measure facilities. The second part is responsible for simulation content: from choosing model of whole mechatronics system up to changing component parameters and necessary settings for output. At the beginning of building simulation, the macro model (device) for simulation should be chosen the first of all. After this step, the user has possibility to build the selected system from different components available in data base.

The main focus of application is modelling electrical power window regulator. It is classical example for modelling mechatronics system, because the window regulator possesses all components that are needed for mechatronics system: electronics, drive, mechanics and control part.

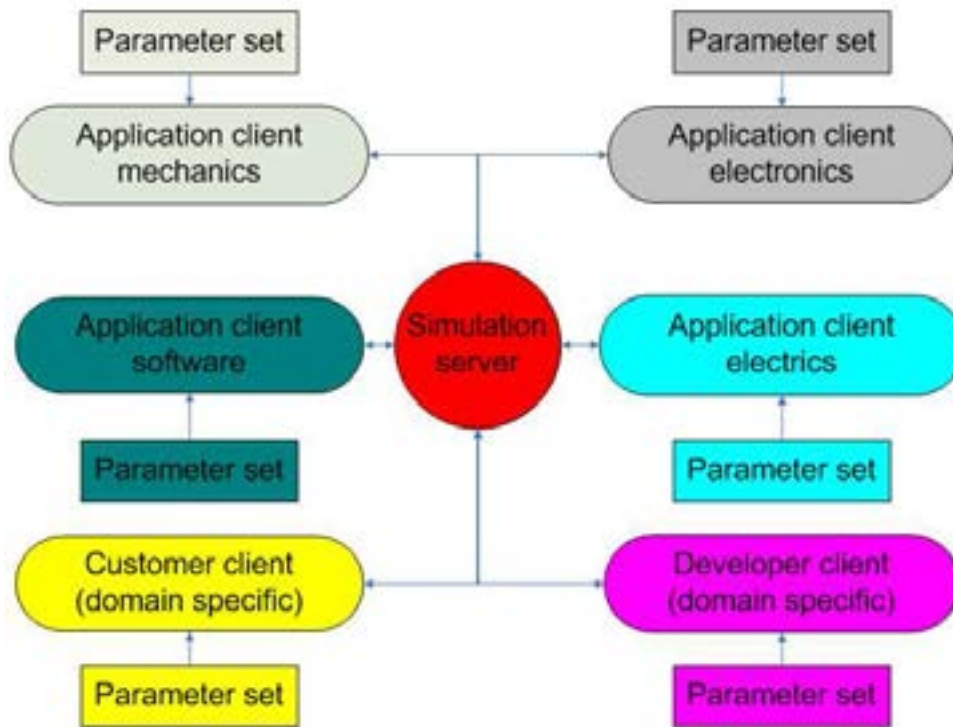


Figure 1: Simulation concept of **BroSAnT**

According to Albert Einstein "... we should make things as simple as possible, but not simpler", there are controlling module and also power source for it. In controlling part, the switching sequence as time or target function should be given.

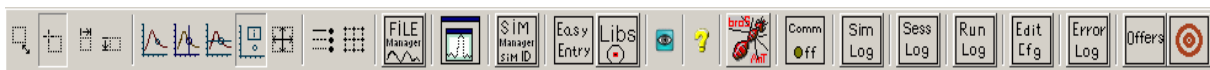


Figure 2: Toolbar of BroSAnT

There are also a number of simulation protocols for tracking the simulation runs. The big amount of simulation possibilities should be cleared through analysing history of the parameter changing. The possibility to save user-tuned component model with help of parameter is realized by means of user library concept. All date base and personal library be kept at server space and can be use from any place if the access to calculation server is granted. On the toolbar (Figure 2) beside zoom/measure and different log elements, there are two important managers: simulation manager and file manager, which allow users to work with simulation results and also with measured data.

The server part is implemented in Linux, and it is responsible for simulation runs/queues and result return. To server task belongs also the service for component data base, model administration and the protocol of whole simulation processes. Thanks to the server-client solution, calculation duration is independent from client side. All calculations are performed at server and the single restriction on calculation time is the amount of required results. Practically, simulation duration is comparable with "real time". Of course, it depends on simulated model and required precision.

At present the whole modelling is implemented in MAST [5] modelling language and is planned to transfer on VHDL-AMS language. With it, higher degree of freedom in selecting simulator manufacture can be reached. The simulator developer does not really support such simulation concepts, because today almost all simulators are domain oriented. Hence, there are two schema of using the simulation methods: the domain specialist makes the simulation itself based on own demand (as a rule in electronics and in algorithm development) or simulation results are produced by simulation experts according to demands from domain specialists (usually other domains). At present, Saber simulator from company Synopsys is used as simulation engine.

6 An example of simulation with BroSAnT a electrical power window regulator model

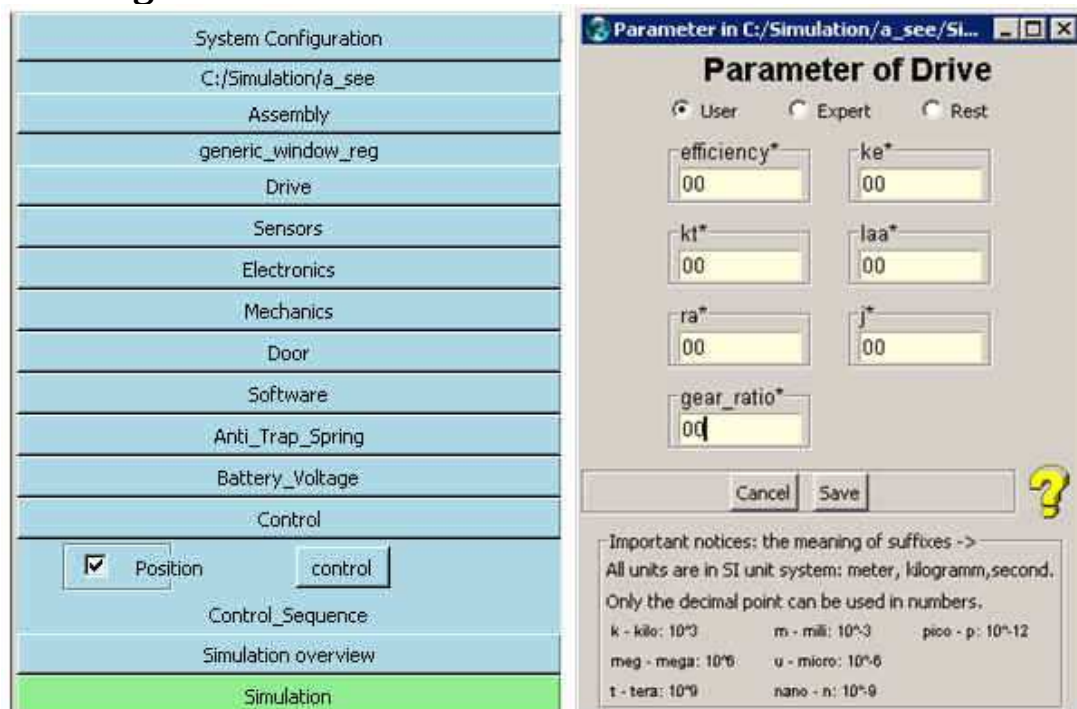


Figure 3: Component's part of the BroSAnT configurator

By choosing a power electrical window regulator as mechatronics system, the simulation configurator gives all necessary components to combine the whole system:

different drives with sensors, mechanics on the cable basis or cross arm construction, electronics part with switches, corresponding sets of possible software algorithms, carrying door model with anti trap measure gauge, source of power like car battery and necessary controlling switch sequences. At the Figure 3 this part of configurator can be seen and one component has its open parameter set. It is drive consisting of DCPM (Direct Current Permanent Magnet) motor and worm gear.

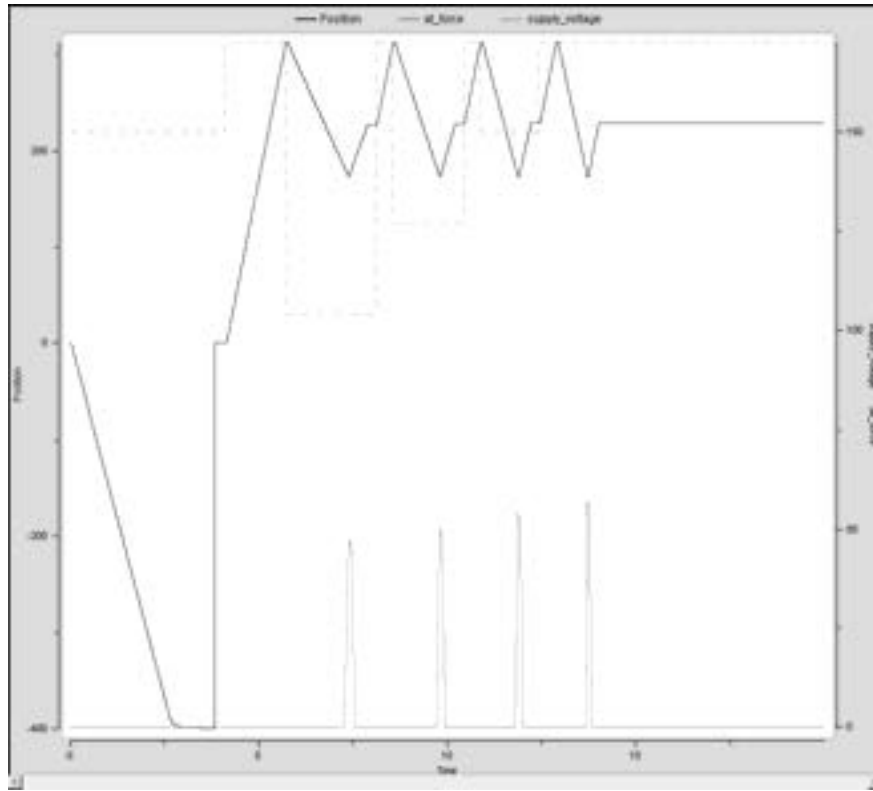


Figure 4: Result of the window regulator working against obstacle (10N/mm)

After simulation run the results is delivered from server to client and the Figure 4 shows the position (in electronics unit) at left axis and maximal reached anti trap force for this system under such condition. There are four anti trapping cases and it can be seen the four different value of anti trap force before the window glass reverses (right axis). It is connected with different supplying voltage by anti trapping (car board voltage - 10 12 14 16 Volt). The supplied voltage to the motor is shown as dashed curve in conventional units on right axis. From this curve it can be concluded that voltage compensation module in algorithms is switched off. The simulation results reflect the behaviour of the system in reality. Such tests can be made with the real hardware too but sometimes it is cheaper and quicker to prove some ideas virtually, especially, if hardware does not exist yet.

If the system is combined, there are many possibilities to test it from various domains. An expert from a domain can study what consequences can be happened under one or another changing in his subsystem/component or how stable his component remains under different working condition.

With practical experiences using **BroSAnT**, it can be stated that different domains have each own imaginations about what is the most important in system and how it should be studied [6]. A number of practical rules can be concluded:

- Simulation results without exact information, about under which condition simulation was done, are not only useless but rather harmful
- the same with validity of test data, it is only reasonable to analyse measured data if the reliability more or less was confirmed

- the most useful information is what is known from daily business and on the contrary, what is not known is no more than interesting but not especially useful

Also after cooperative working, the system simulation developer and product management, a couple "important" remarks come in sight:

- attention to possible problems rises exponentially with rising number of failures in the field or by mass production
- avoided errors and failures will not be taken into account by profit calculation
- every end user wishes each own interface

7 Acknowledgements

We would like to thank Joachim Haase, Roland Kalb, and Detlef Ruß for their valuable feedback and advices. My great acknowledge to George A. Howlett and ActiveState team for their excellent package for 2D graphics - BLT /RBC

Literature

- [1] A Unified Object-Oriented Language for Physical System Modelling: Language Specification Version 3.2. Modelica Association, March 24, 2010. URL:
<https://www.modelica.org/documents/ModelicaSpec32.pdf>
- [2] Behavioural languages - Part 6: VHDL Analog and Mixed-Signal Extensions. IEC Std 61691-6/IEEE Std 1076.1, Dec. 14, 2009. URL:
ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5465882&isnumber=5465881
- [3] SAE J2546_200202- Model Specification Process Standard. SAE Electronic Design Automation Standards Committee, February 15, 2002. URL:
http://standards.sae.org/j2546_200202
- [4] ActiveTclHelp8.5.chm
- [5] MAST - The Analog, Mixed-Technology and Mixed-Signal HDL for Saber URL:
<http://www.synopsys.com/systems/saber/pages/mast.aspx>
- [6] Petkun, Sergey: *Mechatronics simulation on system level: BroSAnT approach*, Proceeding ASIM 2011, Winterthur, Switzerland, 7-9.09.2011

Simulative Erprobung schwarmbasierter Routingalgorithmen im Straßenverkehr

Horst F. Wedde, TU Dortmund Informatik III

horst.wedde@tu-dortmund.de

Sebastian Senge, TU Dortmund Informatik III

sebastian.senge@tu-dortmund.de

Zusammenfassung

Die sich immer mehr häufenden Verkehrsstaus in Ballungsgebieten weltweit verursachen in ökonomischer und ökologischer Hinsicht erhebliche Schäden. Infrastrukturelle Maßnahmen zur Stauvermeidung, wie der Ausbau von Hauptverkehrsadern, sind teuer und in Ballungsgebieten aufgrund mangelnden Platzes nicht ohne weiteres möglich. Eine Alternative stellen Verfahren zur Koordinierung der einzelnen Verkehrsteilnehmer dar, um das vorhandene Straßennetz effizienter auszunutzen. Systeme mit zentraler Steuerung erreichen dabei schnell ihre Grenzen: die zentrale Sammlung, Bereitstellung und Verteilung von zeitgenauer und individueller Verkehrsinformation an die Verkehrsteilnehmer wird ab einem zu großen Abdeckungsgebiet zu lange dauern. Verspätete Informationen können jedoch sogar gegenteilige Effekte haben. Verteilte Systeme, wie schwarmbasierte Verfahren, stellen hier eine Lösung dar.

Indem wir einen schwarmbasierten Routingalgorithmus für große Straßenverkehrsnetze daher dezentral modellierten, also dabei die zentrale Datenhaltung durch koordiniertes dezentrale Datenmanagement unter verteilter Kontrolle ersetzten, konnten wir die obigen Probleme der Realzeitfähigkeit vermeiden, ohne uns auf Daten weniger Hauptverkehrsadern zu beschränken (im Gegensatz zu den bis jetzt gebräuchlichen kommerziellen Systemen). Der auf Prinzipien des Bienenverhaltens aufsetzende Algorithmus heißt BeeJamA (*Bee-Inspired Traffic Jam Avoidance*). Technische Einzelheiten dazu sind etwa unter [1] zu finden.

Um BeeJamA gegenüber gebräuchlichen Verfahren unter realitätsnahen Gegebenheiten zu evaluieren, mussten diese Algorithmen unter gleichen Bedingungen in realistischen Verkehrsumgebungen simuliert werden. In unserem Fall sollten für 200.000 oder mehr Fahrzeuge in der östlichen Hälfte des Ruhrgebiets mit verschiedensten Start- und Zielpunkten die durchschnittlichen Fahrtzeiten (im Stundentakt 5 Stunden lang, über eine über eine Gesamtsimulationszeit von 6 Stunden gestartet) wie auch die jeweils längste Reisedauer festgestellt werden - letztere auch ein indirektes Maß für das Auftreten von Staus.

Simulation des Straßenverkehrs im großen Maßstab stellt somit einen integralen Bestandteil der Arbeit dar. Es wird geschildert, wie Simulationen von Straßenverkehr im Allgemeinen und im Speziellen für den benannten Routingalgorithmus aufgebaut werden können und wo die Fallstricke liegen.

Literatur

[1] Horst F. Wedde, Sebastian Senge et al.: **Towards Hybrid Simulation of Self-Organizing, Online Distributed Vehicle Routing in Large Traffic Systems**. In: Proceedings of the 7th Intl. Conference on Natural Computing (invited paper), Shanghai, China , 2011-07-26

Netzweite Bewertung kooperativer Verkehrssysteme mittels mikroskopischer Verkehrsflusssimulation am Beispiel des eCoMove Projektes

Jonas Lüßmann, Technische Universität München, Lehrstuhl für
Verkehrstechnik

jonas.luessmann@vt.bv.tum.de

Claudia Dittrich, Technische Universität München, Lehrstuhl für
Verkehrstechnik

claudia.dittrich@vt.bv.tum.de

Zusammenfassung

Im Rahmen des durch das siebte Rahmenprogramm der EU geförderten Projektes eCoMove wird ein System mit dem Ziel der Reduzierung von CO₂-Emissionen und des Kraftstoffverbrauchs entwickelt. Das System nutzt dabei kooperative Technologien der C2X-Kommunikation (Fahrzeug-Fahrzeug Kommunikation oder Fahrzeug-Infrastruktur Kommunikation). In den einzelnen Teilprojekten werden dazu fahrzeuginterne und infrastrukturseitige Anwendungen entwickelt. Im Rahmen des Projektes ist es nur möglich, eine begrenzte Anzahl an Fahrzeugen auszustatten. Netzweite Effekte können folglich in den realen Testfeldern nicht gezeigt werden. Eine Abschätzung des Systempotentials ist daher nur mittels Verkehrsflusssimulation möglich. In diesem Beitrag soll das Konzept der in eCoMove verwendeten Simulationsumgebung inklusive sämtlicher notwendiger Komponenten vorgestellt werden.

1 Einleitung

Das Projekt eCoMove hat sich zum Ziel gesetzt, neue verkehrstechnische Anwendungen zu entwickeln, mittels derer sich 20 % des Treibstoffverbrauch und CO₂ Emissionen von Fahrzeugen einsparen lassen. Die Anwendungen zielen auf die beiden Hauptquellen für vermeidbaren Treibstoffverbrauch: Privatfahrten und Gütertransport. Mittels Kommunikation zwischen der Infrastruktur und Fahrzeugen oder auch Fahrzeugen untereinander bekommen die Fahrer Informationen, wie sie im Stadtverkehr oder auch auf der Autobahn effizienter fahren können.

Zum Einen wird den Fahrern beispielsweise beim Annähern an eine Lichtsignalanlage mitgeteilt, bei welcher Geschwindigkeit sie den Knotenpunkt ohne Anzuhalten passieren

können. Zum Anderen nutzen infrastrukturseitige Anwendungen erweiterte Fahrzeuginformationen, beispielsweise zur Verbesserung der Lichtsignalsteuerung.

Da insbesondere netzweite Wirkungen, aufgrund der im Projekt begrenzten Anzahl an mit dem eCoMove System ausgestatteten Fahrzeugen, in der Realität nicht untersucht werden können, wird zur Entwicklung und Evaluation des Systems die mikroskopische Verkehrssimulation VISSIM eingesetzt. Die Simulationsumgebung spiegelt die realen Testfelder aus München und Helmond (Niederlanden) wieder. Um das reale infrastrukturseitige eCoMove System mit ausreichend Verkehrsdaten zu versorgen, werden Adapter entwickelt, die es möglich machen, die Realität durch die Simulation zu ersetzen.

Der Beitrag beschreibt den Ansatz des eCoMove Projektes, das grundsätzliche Konzept der in eCoMove verwendeten Simulationsumgebung, die Anbindung der Simulationsumgebung an das infrastrukturseitige eCoMove System, die Modellierung der Testfeldes München, sowie die Modellierung der fahrzeuginternen Systeme und deren Kalibrierung.

2 Das eCoMove Projekt

An dem durch das siebte Rahmenprogramms der europäischen Kommission geförderten Projekt eCoMove sind 32 Partner aus 10 Ländern beteiligt. Die Partner sind im Wesentlichen Automobilhersteller, Automobilzulieferer, Kommunikationsunternehmen, Kartenhersteller, Forschungseinrichtungen und Hersteller infrastrukturseitiger Verkehrssysteme. Die Projektlaufzeit beträgt 3 Jahre (April 2010 – März 2013). Ziel des Projektes ist es kooperative Systeme zu entwickeln, die es ermöglichen den Kraftstoffverbrauch und die CO₂ Emissionen zu reduzieren.

Dazu ist das Projekt in 6 Teilprojekte unterteilt. Neben dem Projektmanagement sind dies die Teilprojekte

- „Core Technology Integration“, zur Entwicklung von Kerntechnologien des eCoMove Systems, wie der Kommunikationsplattform, der Kommunikationsprotokolle, der Datenbanken und digitalen Karten und verschiedener Modelle zur Beschreibung des Verkehrs,
- „ecoSmart Driving“, zur Entwicklung fahrzeugseitiger Systeme zur Fahrerunterstützung bei Navigation und allgemeinem Fahrverhalten,
- „ecoFreight & Logistics“, zur Entwicklung fahrzeugseitiger Lkw Systeme zur Fahrerunterstützung bei Navigation und allgemeinem Fahrverhalten sowie zentralenseitiger Tourplanung,
- „ecoTraffic Management and Control“, zur Entwicklung infrastrukturseitiger Systeme (innerstädtisch und außerorts), wie zentralenseitiges Routing, Lichtsignalsteuerung, sowie
- „Validation and Evaluation“, zur Wirkungsermittlung und Bewertung der entwickelten Systeme sowie des Gesamtsystems.

Der Nachweis der Wirksamkeit der entwickelten Systeme ist aufgrund der geringen Ausstattungsrate im Feld nur für einzelne Fahrzeuge möglich, nicht aber für ein Verkehrsnetz.

Aus diesem Grund wird im Projekt die Wirkungsermittlung im Verkehrsnetz mittels mikroskopischer Verkehrsflusssimulation durchgeführt.

3 Konzept der Simulationsumgebung

Die Simulation soll zum Einen als Datenlieferant zum Testen und zur Parametrierung, der im Teilprojekt „ecoTraffic Management and Control“ entwickelten Anwendungen verwendet werden. Zum Anderen soll im Laufe des Projekts eine netzweite Wirkungsanalyse des eCoMove Gesamtsystems mittels mikroskopischer Verkehrsflusssimulation erfolgen.

Um diese beiden Anforderungen erfüllen zu können, wurde mittels Software in-the-Loop das infrastrukturseitigen Systems an die Simulation angebunden. Das infrastrukturseitige System entspricht softwaretechnisch der Implementierung in den realen Testfeldern. Die Simulation spiegelt die realen Testfelder wider. Mittels entsprechender Adapter wird zudem sicher gestellt, dass nicht nur dieselben Daten zur Verfügung sondern diese auch im gleichen Format wie aus realen Fahrzeugen und realen Detektion zur Verfügung stehen. Die fahrzeugseitigen Anwendungen sowie die Kommunikation werden dagegen nicht simuliert, sondern über Modelle abgebildet.

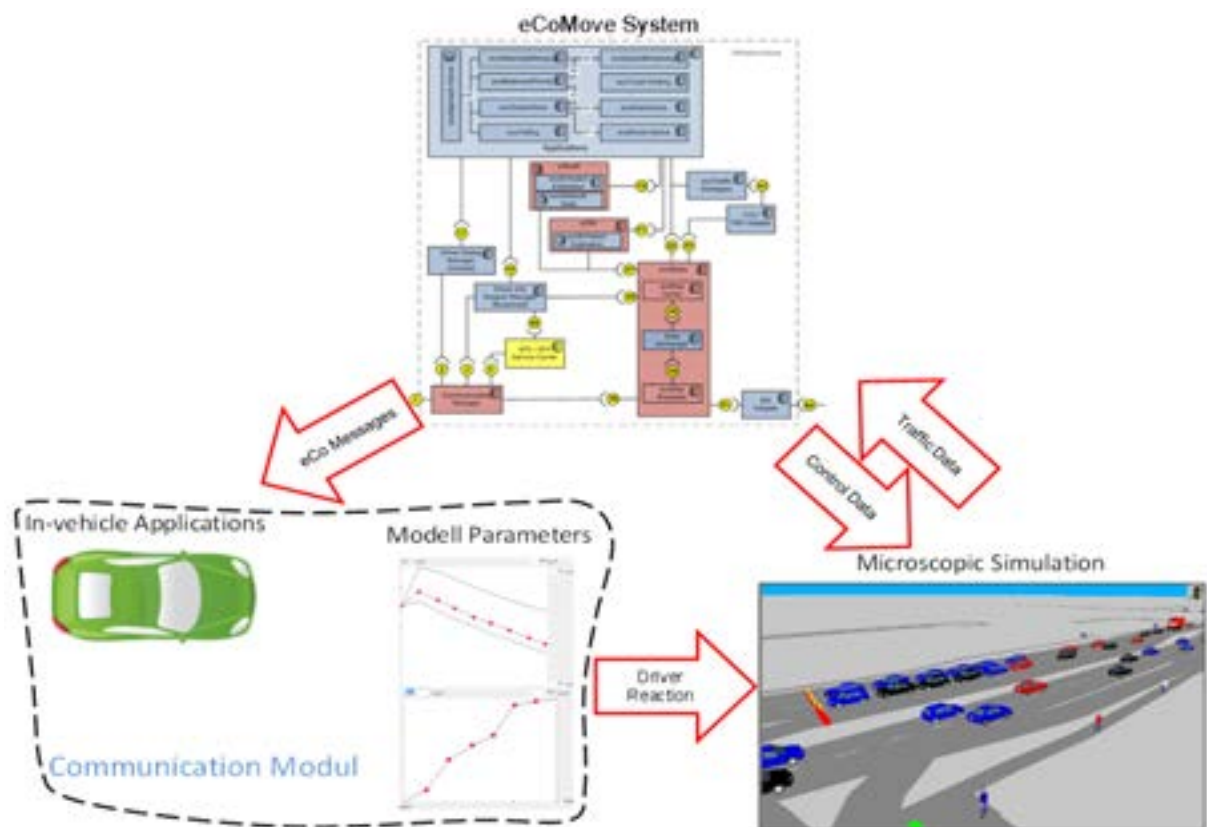


Abbildung 1: Konzept der eCoMove Simulationsumgebung [1]

Als Simulationsumgebung wird die mikroskopische Verkehrsflusssimulation VISSIM [2] des eCoMove Partners PTV AG eingesetzt. VISSIM verwendet als Verkehrsflussmodell ein stochastisches, zeitschrittbasiertes mikroskopisches Modell. Die Fahrer-Fahrzeug-Einheiten werden dabei als elementare Einheiten betrachtet.

4 Technische Anbindung des Infrastruktursystems an die Simulation

Im folgenden wird beschrieben, wie die Simulation an das Infrastruktursystem von eCoMove angebunden wird. Dazu werden zuerst die Gesamtarchitektur und die Vissim-eigenen Kommunikationsmodellierung beschrieben, bevor auf die konkrete Implementierung der Komponenten und Schnittstellen eingegangen wird.

4.1 Architekturübersicht

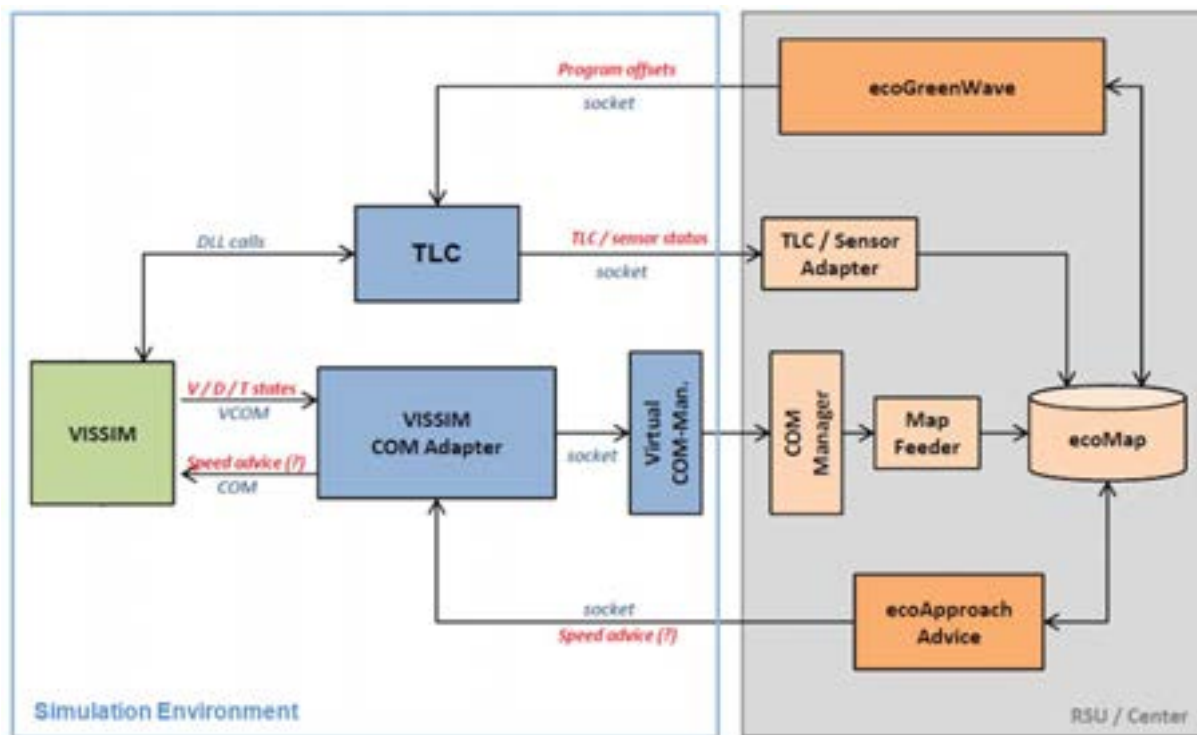


Abbildung 2: Architektur der Kopplung von Simulation mit dem Infrastruktursystem adaptiert von [3]

Die Architektur ist als Übersicht in Abbildung 2 dargestellt. Auf der linken Seite steht die Simulationsumgebung und auf der rechten Seite das Infrastruktursystem, welches sowohl auf einer sogenannten Road Side Unit wie auch auf einem zentralen Server laufen kann. Während die linke Seite im späteren Projektverlauf durch die realen Testfelder ersetzt wird, bleibt die rechte Seite bestehen. Um die Übersichtlichkeit nicht zu beeinträchtigen, beinhaltet die Darstellung nur zwei exemplarisch ausgewählte Anwendungen, die ecoGreenWave als Beispiel für eine Anwendung der Lichtsignalsteuerung und den ecoApproach Advice als Beispiel für eine Anwendung die mit dem fahrzeugseitigen System kommuniziert.

Das System besteht aus mehreren Komponenten mit loser Kopplung. Die Verkehrsflusssimulation, VISSIM, als Stellvertreter für die realen Testfelder ist über mehrere Schnittstellen mit der ecoMap verbunden. Die ecoMap stellt den zentralen Datencontainer dar. Über die ecoMap sind alle statische Kartendaten und dynamischen Daten, die im Laufe der Zeit generiert werden verfügbar. Es handelt sich dabei nicht um eine Datenbank mit der

Möglichkeit Daten persistent abzuspeichern, sondern sie dient vorrangig als Austausch flüchtiger Daten für die Anwendungen.

Die Anwendungen werden in Form von OSGi Diensten implementiert. Sie kommunizieren untereinander mittels eines eigenen Nachrichtendienstes „ecoMessage service“. Wenn sich eine Anwendung für Daten einer anderen Anwendung interessiert, muss sie die entsprechenden Nachrichten abonnieren und bekommt die Daten, sobald sie vorliegen.

4.2 Kommunikationsmodellierung mittels VCOM

Zur Modellierung der Kommunikation von Fahrzeugen bietet VISSIM ein separates Modul, das VCOM Modul, welches als dynamische Bibliothek (dll) implementiert ist. Vorteil der Implementierung als dll ist, dass Informationen von Fahrzeugen mit Kommunikationseinheit effizient durch direkten Methodenaufruf abgefragt werden können. Diese Informationen umfassen unter anderem Position, Geschwindigkeit, Beschleunigung, Sicherheitsabstand, Fahrzeugtyp und Bewegungsvektor der Fahrzeuge.

Um die Kommunikation abzubilden und keine extra Kommunikationssimulation anwenden zu müssen verfügt VCOM über eine Modellierung der Kommunikation. Dazu werden die Informationen in Abhängigkeit der Entfernung zwischen Sender und Empfänger und der Anzahl der kommunizierenden Fahrzeuge pro Kilometer mit einer gewissen Wahrscheinlichkeit übermittelt.

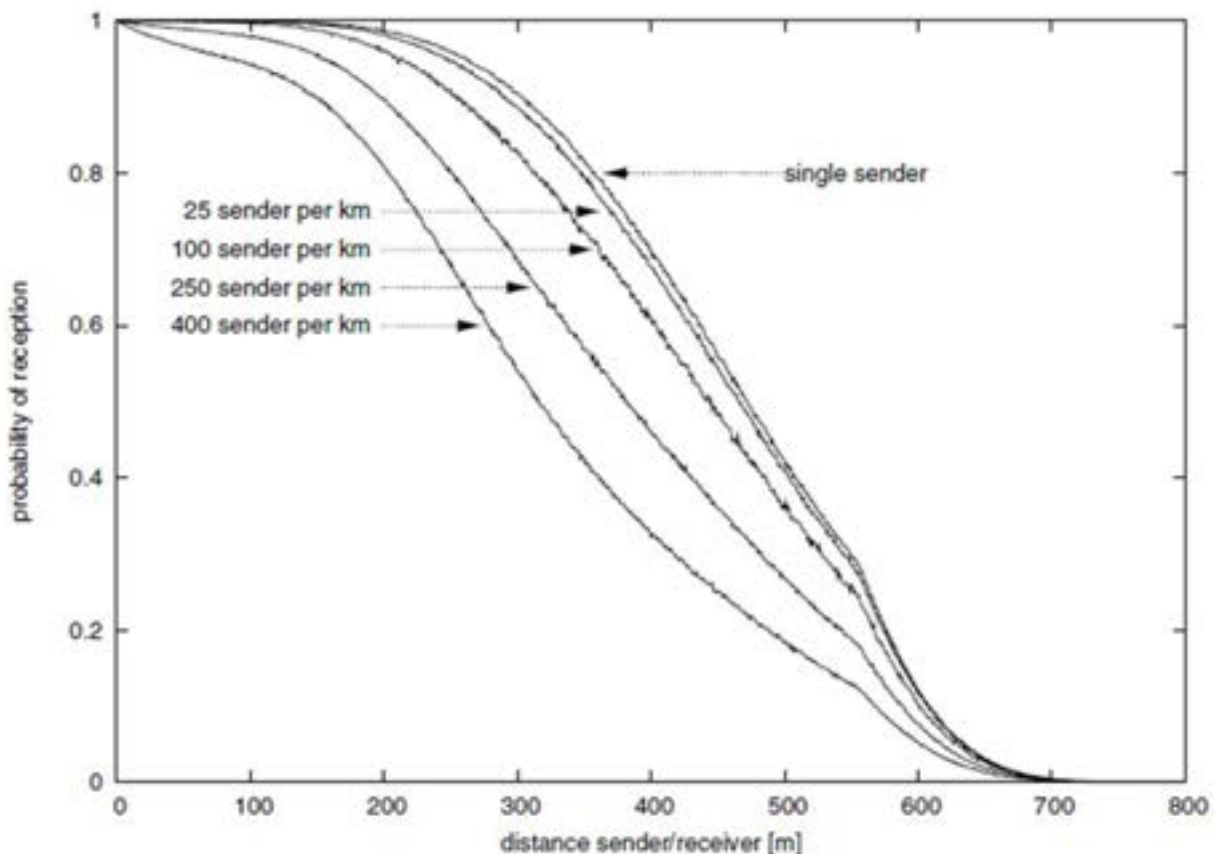


Abbildung 3: Empfangswahrscheinlichkeit in Abhängigkeit von der Entfernung zwischen Sender und Empfänger und der Anzahl der kommunizierenden Fahrzeuge pro Kilometer. [4]

Die Empfangswahrscheinlichkeit wurde mittels des „Network Simulator ns-2“ kalibriert. Mit verschiedenen Empfangswahrscheinlichkeitsverteilungen kann neben 802.11p Kommunikation beispielsweise auch UMTS abgebildet werden.

Um die Fahrzeuginformationen zu erhalten und zu verarbeiten, übergibt VISSIM nach jedem Simulationszeitschritt die Kontrolle an die entsprechende Anwendung ab. Die Anwendung bekommt über eine VCOM Schnittstelle alle Informationen der kommunizierenden Fahrzeuge. Eine interne Logik der Anwendung prüft durch Abfrage des VCOM Moduls, ob die aktuellen Informationen ausgetauscht werden konnten und ob sie eine Aktion auslösen. Ist dies der Fall, wird die Aktion (z.B. Anpassung der Geschwindigkeit) durchgeführt und die Kontrolle wird wieder an VISSIM zurück übergeben.

4.3 Konkrete Implementierung

Um die Simulationsumgebung VISSIM an das infrastrukturseitige eCoMove System anzubinden, sind im Wesentlichen zwei Schnittstellen notwendig: Eine, um die Fahrzeugdaten wie die Fahrzeugpositionen aus der Simulation abzugreifen und eine, um Informationen über die Signalsteuerung und den Detektorzustand zu bekommen. Der Zugriff auf die Fahrzeugdaten erfolgt über das VCOM Modul in dem VISSIM COM Adapter. Der Virtual COM Manager wandelt die Fahrzeugdaten lediglich in das in eCoMove definiert Nachrichtenformat um. Der Zugriff auf die Signalsteuerung geschieht in der Komponente Traffic Light Control mittels Aufruf der entsprechenden dynamischen Bibliotheken (dlls) von VISSIM. Der VISSIM COM Adapter und die Traffic Light Control (TLC) Schnittstelle zusammen bilden die notwendige Schale um die Simulationsumgebung herum zur Anbindung der Simulation an das Infrastruktursystem. Die Anbindung selbst erfolgt über Socket-Verbindungen. So ist die Kommunikation zwischen beiden Systemen leicht möglich, auch wenn sie auf unterschiedlichen Programmiersprachen basieren (die Simulationsumgebung basiert auf C++ und das eCoMove Infrastruktursystem ist in Java programmiert). Theoretisch ist auch eine hardwaretechnische Trennung möglich, sodass die Simulationsumgebung auf einem anderen Rechner mit anderem Betriebssystem laufen kann wie die Infrastrukturumgebung.

Fahrzeugdaten und die Daten aus der Lichtsignalsteuerung gelangen äquivalent zu den realen Testfeldern in die ecoMap und werden von dort von den Anwendungen abgegriffen. Die Anwendungen verarbeiten die aktuell für sie relevanten Daten und stellen die Ergebnisse zur Verfügung. Die Ergebnisdaten fließen in Form von Signalprogrammanpassungen (ecoGreenWave) oder Geschwindigkeitsempfehlungen (ecoApproach Advice) zurück in die Simulationsumgebung. Durch den Rückkanal können die Anwendungen getestet werden. Wobei zu beachten ist, dass die Simulation dabei in Realzeit abläuft, d.h. ein Zeitschritt entspricht einer realen Sekunde, damit die Funktionsweise des Gesamtsystems der Laufzeit in den Testfeldern entspricht.

Außerdem können die Auswirkungen der Anwendungen evaluiert werden. Beispielsweise können dadurch Aussagen getroffen werden, wie sich die Fahrweise der Verkehrsteilnehmer

und die Emissionen im Zuge von Geschwindigkeitsbeeinflussungen an Knotenpunkten oder durch Änderungen in der Lichtsignalsteuerung verändern.

5 Modellierung des Münchener Testfeldes

Das Testfeld München umfasst große Teile des Münchner Nordens inklusive dem Autobahnring der A99. Es wurde so gewählt, dass sämtliche infrastrukturseitigen Anwendungen im Testfeld umgesetzt werden können. Im Wesentlichen sind damit die Routinganwendungen für die zu modellierende Netzgröße ausschlaggebend.

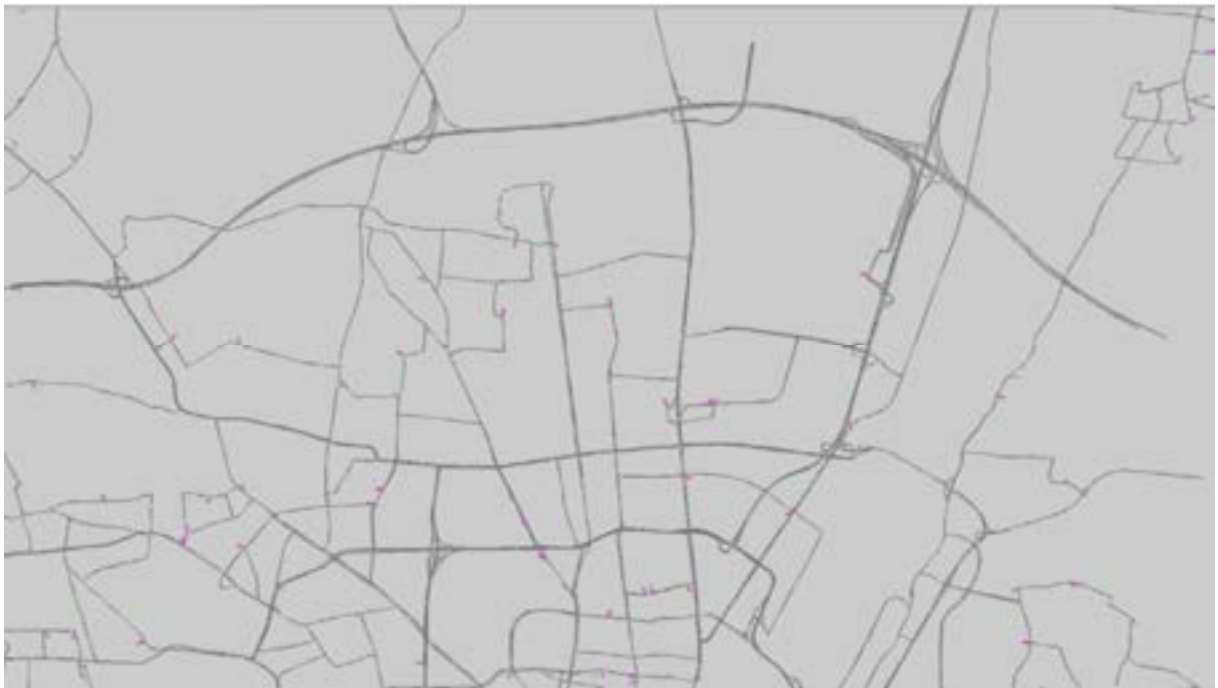


Abbildung 4: Modelliertes Testfeld München

Die Erzeugung des Netzes erfolgt über eine im Rahmen von sim^{TD} entwickelte Geo-Datenbank [5] aus der für die gewünschten Simulationen unterschiedliche Netze exportiert werden können. Wesentliche Datenobjekte der Datenbank sind das Straßennetz, Detektoren, Lichtsignalanlagen sowie Verkehrsnachfragewerte.

Da sowohl die Geo-Datenbank als auch die ecoMap auf NAVTEQ Kartendaten basieren ist ein Mapmatching der Fahrzeuge in der Simulation in der infrastrukturseitigen Datenbank leicht möglich.

6 Fahrermodellierung

Das eCoMove System beeinflusst im Wesentlichen zwei Komponenten des Fahrverhaltens. Dies sind die Längsbewegung und die Routenwahl.

6.1 Längsbewegung

Für die Längsbewegung verwendet VISSIM verwendet als Fahrzeugfolgemodel das psycho-physisches WIEDEMANN Model [4][5]. Die Grundidee des Modells nach

Wiedemann ist die Annahme, dass sich ein Fahrer in einem von vier Fahrzuständen befinden kann:

- **Freies Fahren:** Es ist kein Einfluss eines vorausfahrenden Fahrzeugs zu beobachten. Der Fahrer versucht seine Wunschgeschwindigkeit zu erreichen und dann beizubehalten.
- **Annäherung:** Der Fahrer passt seine Geschwindigkeit an ein vorausfahrendes langsames Fahrzeug an. Er verzögert so, dass im Idealfall die Geschwindigkeitsdifferenz zum Vorderfahrzeug Null ist, wenn er seinen gewünschten Sicherheitsabstand erreicht hat.
- **Folgen:** Der Fahrer folgt einem vorausfahrenden Fahrzeug, ohne zu bremsen oder zu beschleunigen. Wegen der unvollkommenen Beherrschung des Gaspedals und der damit einhergehenden Geschwindigkeitsdifferenz, die in einem kleinen Bereich um Null oszilliert, schwankt der Abstand damit.
- **Bremsen:** Der Fahrer verzögert, falls der Abstand zum Vorderfahrzeug unter den gewünschten Sicherheitsabstand fällt.

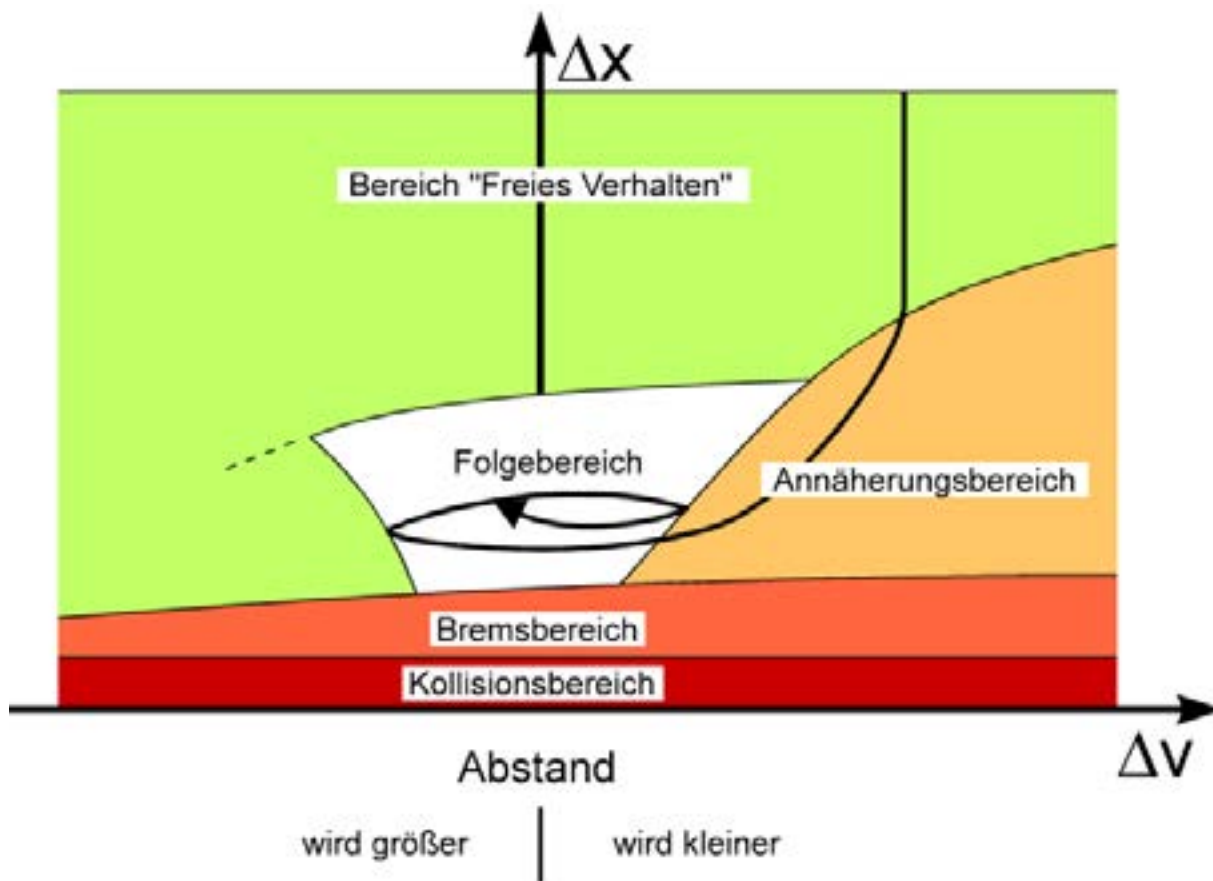


Abbildung 5: Fahrzeugfolgmodell (nach: WIEDEMANN 1974) [2]

Folgende Parameterwerte für unbeeinflusste Fahrer sind aus zahlreichen Messungen bereits bekannt. Auch für vom eCoMove System beeinflusste Fahrer wird für diese Parameter keine Änderung erwartet, da sie sich nur auf den direkt das Fahrzeug umgebenden

Verkehr beziehen, die Informationen des Systems an den Fahrer aber wesentlich weiter vorausschauen, als es dem Fahrer selbst möglich ist:

- **Vorausschauweite** gibt die Entfernung an, die ein Fahrzeug nach vorn schauen kann, um auf Fahrzeuge vor oder neben sich zu reagieren.
- **Anzahl Vorderfahrzeuge** bestimmt, wie gut Fahrzeuge das Fahrverhalten anderer Fahrzeuge voraussehen und darauf reagieren können.
- **Zurückschauweite** gibt die Entfernung an, die ein Fahrzeug zurück schauen kann, um auf Fahrzeuge hinter sich zu reagieren.
- **Vorübergehende Unaufmerksamkeit:** Fahrzeuge reagieren für eine gewisse Zeit nicht auf vorausfahrende Fahrzeuge.
- **Wiedemann Modellparameter:** In Abhängigkeit vom gewählten Fahrzeugfolgemodell

Weitere wesentliche Eingangsparameter für das Fahrzeugfolgemodell sind die Verteilungen der:

- **Wunschgeschwindigkeit,**
- **Wunschbeschleunigung** und
- **Wunschverzögerung.**

Für unbeeinflusste Fahrer liegen hier bereits Messungen vor. Für die durch das System beeinflussten Fahrer werden die Parameter mittels einer Studie im Fahrsimulator für verschiedene Anwendungsfälle erhoben.

6.2 Routenwahl

Das Verhalten der Fahrer auf die Beeinflussung der Routenwahl hängt von einer Vielzahl von Faktoren ab, wie dem Übermittlungsmedium, der Anzahl der Alternativrouten sowie deren Streckenlängen und Reisezeiten. Eine Kalibrierung der Routenentscheidung mittels Fahrsimulatorstudie macht daher keinen Sinn.

Für ortunkundige Fahrer wird ein hundertprozentiger Befolgungsgrad angenommen. Für den Befolgungsgrad ortskundige Fahrer werden Daten aus dem Forschungsprojekt wiki (**W**irkungen von **i**ndividueller und **k**ollektiver ontrip **V**erkehr**i**nformation und -beeinflussung auf den Verkehr in Ballungsräumen) [8] verwendet. Im Projekt wurde das Routenwahlverhalten ortskundiger Fahrer im Münchner Norden untersucht.

Literatur

- [1] Lüßmann J., Schendzielorz T., Vreeswijk J.: *Extension of Simulation Functionalities and Test Site Modelling*. eCoMove deliverable 5.3, 2011.
- [2] PTV AG, *VISSIM 5.40 Benutzerhandbuch*, Karlsruhe, 2011.
- [3] MAT.TRAFFIC, *interner Bericht aus eCoMove*, 2011.

- [4] Killat, M.; Schmidt-Eisenlohr F.; Hartenstein, H.; Rössel, C.; Vortisch, P.; Assenmacher, S.; Busch F., *Enabling efficient and accurate large-scale simulations of VANETS for vehicular traffic management, Proceedings of the fourth ACM International Workshop on Vehicular Ad Hoc Networks (VANET)*, pp. 29-38, Montreal, Quebec, Canada, September 2007.
- [5] MAT.TRAFFIC, *Dokumentation der Geo-Datenbank für Feldversuch und Verkehrssimulation in simTD*, Aachen, 2011.
- [6] Wiedemann, R. , *Simulation des Straßenverkehrsflusses*, Schriftenreihe des Instituts für Verkehrswesen der Universität Karlsruhe, Heft 8, 1974.
- [7] Wiedemann, R., *Modeling of RTI-Elements on multi-lane roads*, Advanced Telematics in Road Transport, DG XIII, Brussels, 1991.
- [8] wiki – *Wirkungen von individueller und kollektiver ontrip Verkehrsbeeinflussung auf den Verkehr in Ballungsräumen*, Gemeinsamer Schlußbericht, 2012.

Simulation and optimization of Cologne's tram schedule

Oliver Ullrich^a, Daniel Lückcrath^{a,b}, Sebastian Franz^a,
Ewald Speckenmeyer^a

^aInstitut für Informatik, Universität zu Köln

^bInstitut für Nachrichtentechnik, Fachhochschule Köln
ullrich|lueckerath|franz|esp@informatik.uni-koeln.de

Abstract

In many tram networks multiple lines share tracks and stations, thus requiring robust schedules which prevent inevitable delays from spreading through the network. Feasible schedules also have to fulfill various planning requirements originating from political and economical reasons.

In this paper we present a tool set designed to generate schedules optimized for robustness, which also satisfy given sets of planning requirements. These tools allow us to compare time tables with respect to their applicability and evaluate them prior to their implementation in the field.

This paper begins with a description of the tool set focusing on optimization and simulation modules. These software utilities are then employed to generate schedules for our hometown Cologne's tram network, and to subsequently compare them for their applicability.

1 Introduction

In many tram networks, several lines share resources like stations and tracks. This results in very dense schedules, with vehicles leaving platforms every minute at peak times. In order to prevent inevitable local delays from spreading through the network, a schedule has to be robust.

Many additional planning requirements to real world tram schedules originate from political, economical and feasibility reasons. Thus it is not sufficient to exclusively consider general criteria like robustness or operational costs when generating time tables. Typical requirements include fixed start times at certain stations, e.g. interfaces to national railway systems, core lines that relieve high passenger load, e.g. for lines which traverse city centers, warranted connections at certain stations, and safety distances to be complied with at bidirectional tracks.

In this paper we present an introduction to our project to generate and evaluate robust time tables which also satisfy given sets of planning requirements. We describe a tool chain which enables us to generate optimized schedules, compare their applicability and evaluate them prior to their implementation in the field.

This paper continues with a description of the current state of the project, focusing on our approaches on optimization and simulation (Section 2). We then present some experimental results obtained by applying the described software to our hometown Cologne's tram network (Section 3). The paper closes with a short summary of lessons learned and some thoughts on further research (Section 4).

2 Simulating and optimizing tram schedules

Our project “*Computer Aided Traffic Scheduling*” (CATS) is built around a database complying with the ÖPNV5 data model released by the *Association of German Transport Companies* (*Verband Deutscher Verkehrsunternehmen*, see [19]). Visualization, optimization, and simulation modules are connected via operations on the database and through XML configuration files (see figure 1). Due to its compliance with the ÖPNV5 data model our framework is capable of working on many European tram networks.

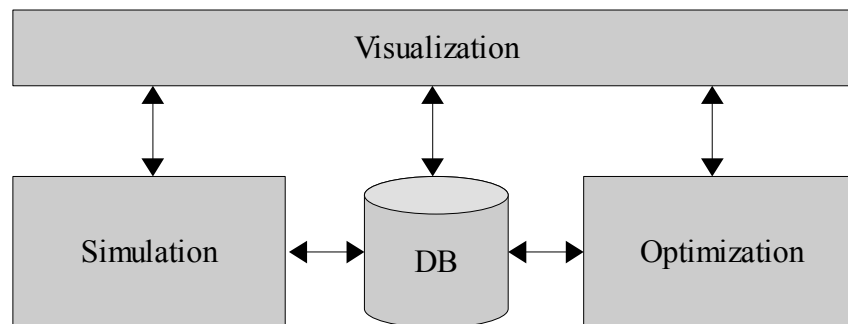


Figure 1: Project modules

2.1 Optimization of tram schedules

Various approaches to optimize tram and railway schedules are known (see e.g. [1, 3, 4, 7, 16, 17, 18]). Most of them aim at one general objective like minimizing vehicle delay (see [16, 18]) or maximizing robustness to restrict the global impact of small, local disturbances (see [4, 7]). Others use a combination of objectives, like operational profit and robustness in [3], or combining social opportunity cost and operational cost in [17].

Because of the complex nature of the problem, many authors use heuristic approaches like Lagrangian heuristics (see [3]) or simulated annealing (see [17]). Others, like Bampas et al. in [1] introduce exact algorithms for restricted subclasses, like chain and spider networks.

In our project, we combine heuristics and exact methods to generate optimal synchronized time tables for general tram networks, targeting maximal robustness and adherence to transport planning requirements at the same time.

To calculate the robustness of a time table we examine at each platform the safety distance $\delta_{f, pred(f)}$ of any trip f and its predecessor $pred(f)$, i.e. the time elapsed between the departures of $pred(f)$ and f at the examined station. To reduce complexity we aggregate subsequent similar platforms operated by the same lines to a maximal platform type h' , weighted by the number of included platforms φ_h (see figure 2). The reduced set of platforms is denoted by H' .

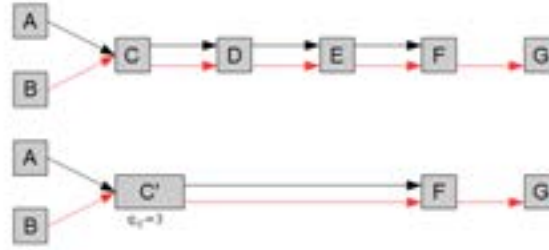


Figure 2: Example of platform reduction

To calculate the robustness $\Phi_a(\lambda)$ of schedule λ , we add the inverse of $\delta_{f, pred(f)}$ for all platforms and all trips, thus applying a penalty for small safety distances. With F_h representing all trips that serve platform h , the resulting function is as follows:

$$\Phi_a(\lambda) = \sum_{h \in H'} \sum_{f \in F_h} \frac{1}{\delta_{f, pred(f)}} * \varphi_h$$

In order to calculate the compliance with transport planning requirements we introduce $\rho_v \in \{1, 2, 3, \infty\}$, the compliance factor of requirement v . A compliance factor of 1 means that the requirement is completely satisfied, 2 and 3 denote tolerable compliance, and ∞ means that the constraint is not met and the time table candidate must be rejected. With V denoting the set of all planning requirements, we add the compliance values and get the following:

$$\Phi_b(\lambda) = \sum_{v \in V} \rho_v$$

Depending on the network under consideration and the number of planning requirements, the two parts of the objective function may not be comparable directly. Thus we define a normalizing factor σ , which reflects the relationship between the theoretically optimal safety distance $\delta_{f, pred(f)}^{\text{opt}}$, obtained by dividing the tact interval by the number of serving lines, and the optimal compliance factor ρ_v^{min} . We define σ as:

$$\sigma = \left(\sum_{h \in H'} \sum_{f \in F_h} \frac{1}{\delta_{f, pred(f)}^{\text{opt}}} * \varphi_h \right) / \sum_{v \in V} \rho_v^{\text{min}}$$

Combining $\Phi_a(\lambda)$ and $\Phi_b(\lambda)$ yields the overall objective function $\Phi(\lambda)$, normalized by σ and weighted by α , the relative weight of the fulfillment of planning requirements.

$$\Phi(\lambda) = (1 - \alpha) * \sum_{h \in H'} \sum_{f \in F_h} \frac{1}{\delta_{f, pred(f)}} * \varphi_h + \alpha * \sum_{v \in V} \rho_v * \sigma \text{ with } \alpha \in [0, 1)$$

We identify seven types of transport planning constraints: Interval constraints, start time constraints, core line constraints, bidirectional track constraints, turning point constraints, warranted connection constraints and follow-up connection constraints.

Upon closer inspection it becomes clear that interval and start time constraints are elemental and all other constraint types can be expressed using these two. E.g. a bidirectional track constraint can be expressed by two interval constraints covering opposite platforms. Subsequently only interval and start time constraints are considered in the remainder of this paper.

To accelerate the computational process the implemented branch-and-bound solver is preceded by a genetic algorithm. We encode a time table into one individual, consisting of the first trip start time of each line, i.e. the offset in minutes from the start of the operational day. All other trips follow determined by their line's tact interval. The application generates a start population using random start time values, testing validity against planning constraints and collisions on network nodes. To reduce computational complexity we apply simple deterministic tournament selection and two-point-crossover (as described in [5]). After evaluation of several mutation methods, including random, minimal, and maximum enhancement mutation we choose a minimal random mutation method that only allows start times to be altered by one minute. We utilize a steady state replacement method, also described in [5]. At the end of each run a hill climbing algorithm is applied to the best individual to further improve its fitness.

As described above we use the best individual encountered by the genetic algorithm to provide the branch-and-bound solver with an initial upper bound, thus avoiding a cold start. Each inner node of the search tree represents a partial solution of the problem (see [8]). The root of the tree corresponds to a solution in which no line's start time is fixed. With each level of the tree admissible start times for an additional line are set.

The objective function is modified in order to cut branches off the tree as soon as possible. The set of lines L is divided into subsets of lines that are already fixed $\hat{L}$ and lines that are not yet fixed $\tilde{L}$. Accordingly we divide the set of transport planning constraints V into $\hat{V}$ and $\tilde{V}$ as well as the set of platforms H into $\hat{H}$ and $\tilde{H}$. The modified objective function $\Phi'(\lambda)$ is shown below.

$$\Phi'(\lambda) = (1 - \alpha) * \left(\sum_{h \in \hat{H}} \sum_{f \in F_h} \frac{1}{\delta_{f, pred(f)}} * \varphi_h \right) + \sum_{h \in \tilde{H}} \sum_{f \in F_h} \frac{1}{\tilde{\delta}_{f, pred(f)}} * \varphi_h + \alpha * \left(\sum_{v \in \hat{V}} \rho_v + \sum_{v \in \tilde{V}} \rho_v^{min} \right) * \sigma$$

$\tilde{\delta}_{f, pred(f)}$ represents the theoretically best safety distance value under consideration of lines already fixed. Again, ρ_v^{min} denotes the optimal compliance factor for constraint v . These values are applied in order to find a lower bound for solution candidates in the current branch of the search tree.

For further implementation details, see [6].

2.2 Simulation of tram schedules

Most rail-bound traffic simulations are designed for long distance train or railway networks, see e.g. [13, 15]. While those systems feature similarities to tram networks, e.g. passenger exchange or maneuvering capabilities, they differ greatly in important aspects. Tram networks are often mixed, i.e. trams travel on underground tracks as well as on street level, and are thus subject to individual traffic and corresponding traffic regulation strategies. Subsequently, tram behavior is a mixture between train and car behavior, e.g. line-of-sight operating/driving. Therefore a simple adaption of railway simulation methodologies is not feasible.

Bearing the similarities with individual traffic in mind Joisten implemented an adapted Nagel/Schreckenberg model (see [14]) for tram simulation, which suffered from the setbacks of the high aggregation inherent to cellular automats (see [10]). Therefore Lückemeyer developed an event based simulation model which avoids some of those setbacks as described in [9, 10]. To further eliminate inaccuracies we apply an updated model, as described in [12].

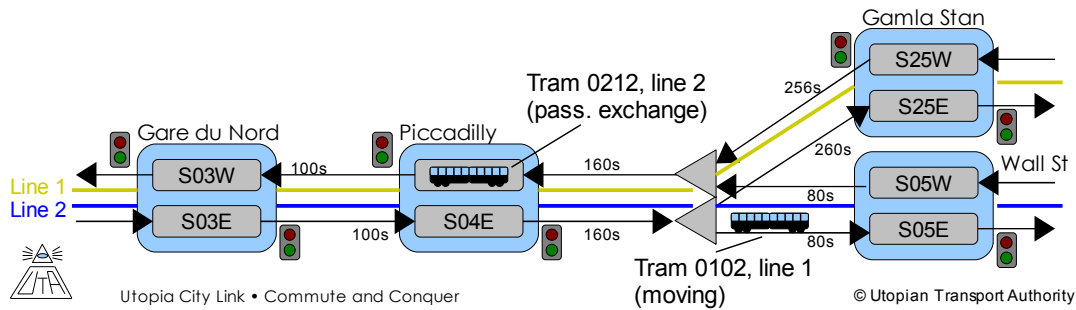


Figure 3: Part of a tram network

Our application is based upon a model-based parallelization framework, which exploits the embedded model's intrinsic parallelism. The mixed tram network is modeled as a directed graph with platforms, tracks and track switches represented by nodes. Connections between nodes are represented as edges. Figure 3 shows part of an example network, which is mapped on the graph depicted in figure 4, where squares represent platforms, rectangles tracks and triangles track switches. The rectangles around platforms indicate that these platforms form a station.

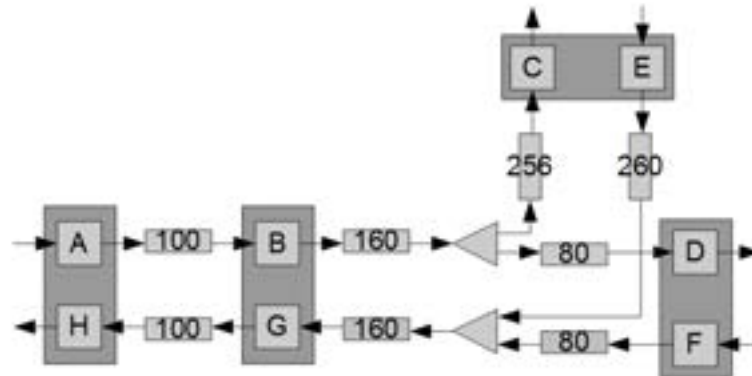


Figure 4: Example graph representing part of a tram network

Passenger boarding and deboarding time distributions are specific to platform and tram type with the combined duration of opening and closing the vehicle doors as minimum value.

Main tram attributes are specified by the type of tram, which holds functions for the maneuvering capabilities, e.g. acceleration and braking.

3 Experiments

We apply the developed software suite to our hometown Cologne's tram network based on the time table data of 2001 (see figure 5). It consists of 528 platforms and 58 track switches connected via 584 tracks. These tracks cover a total length of 407.4 kilometers, resulting in an average track length of 697.6 meters. 15 lines with 182 line routes are served by 178 vehicles which execute 2,814 trips per operational day.

Figure 5: Cologne's tram network in 2001

other maintenance trips at the rim of the network. For the following optimization run, we assume a tact interval of ten minutes, and define a set of example constraints, which can be decomposed to four start time constraints and 34 interval constraints. These include minimum turn-around times at line ends, an additional core line 1A to satisfy high demand for line 1 in Cologne's town center, guaranteed connections between certain lines, and fixed start times at the Bonn national railway hub.

3.2 Comparing two tram schedules

From the genetic algorithm's initial pool of valid solution candidates we randomly take a schedule A with an objective function value of 7,655.14 (see table 1). After a 166 minutes run, the optimizer yields a best solution candidate B with an objective function value of 6,786.60 (see table 2).

Direction	Line														
	1	1A	3	4	5	6	7	8	9	12	13	15	16	18	19
Forward	4	5	9	4	0	3	5	3	7	2	9	4	6	9	8
Backward	5	2	8	6	9	8	1	0	6	0	4	5	5	7	3

Table 1: Schedule A – Initial schedule

We examine both schedules by executing 30 simulation runs and comparing the results. Schedule A yields an average line delay of 23.6 seconds, while schedule B yields one of 17.7 seconds. As seen in figure 6, implementation of schedule B enhances punctuality of every line at least marginally. Lines 6, 7 and 18 in particular are improved significantly, reducing line delay between 20 and 40 percent. Lines 1, 8 and 13 feature an even more improved punctuality (see table 3) and thus deserve a closer examination.

Direction	Line														
	1	1A	3	4	5	6	7	8	9	12	13	15	16	18	19
Forward	8	4	3	9	5	4	9	4	2	5	7	5	6	9	7
Backward	4	2	4	1	3	8	0	1	7	8	7	8	5	7	2

Table 2: Schedule B – Best schedule

Line 1 (combined with line 1A) traverses the highly frequented city center every 5 minutes and shares important resources with lines 7, 8 and 9. Thus it is very susceptible to small disturbances originating in those highly requested areas. In comparison to schedule A, schedule B's better utilization of safety distances improves punctuality by 43 percent.

Outside the town center, line 8 yields a particularly high delay under schedule A due to a marginal safety distance between its vehicles and those of its immediate predecessor line 7. Therefore trams of line 8 are prone to resource conflicts with vehicles of that line. In schedule B the resulting delay is reduced to 28 percent by increasing the safety distances to 4 and 6 minutes respectively.

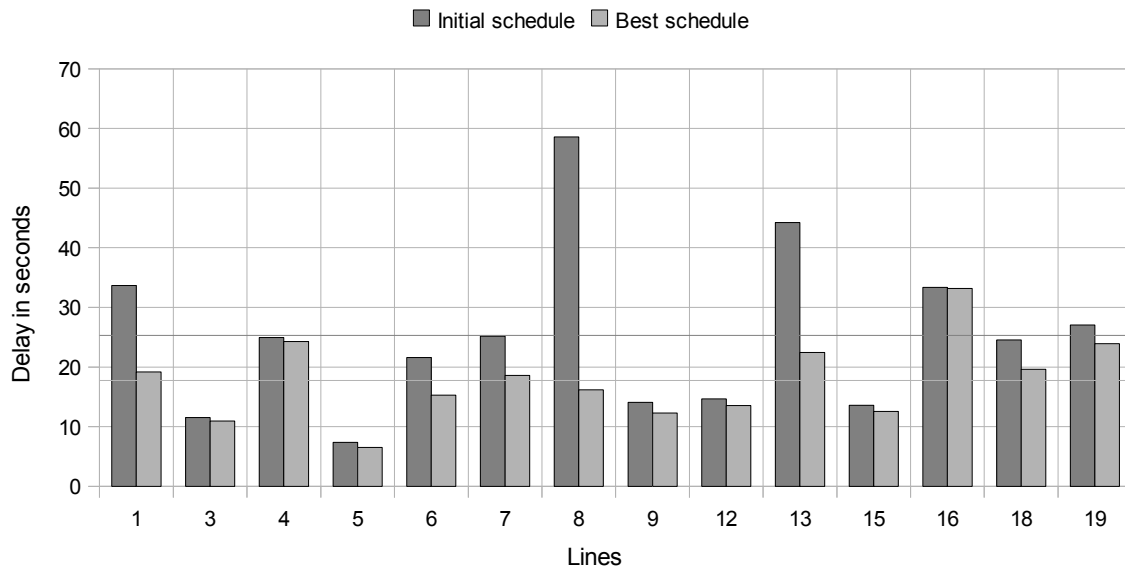


Figure 6: Average delay of lines

Examining the planned departure times of north-east bound line 13 under time table A shows that even small delays resulting from conflicts with lines 5 or 7 cause vehicles of the line to fall directly behind those of line 15. This further prolongs their delays and makes it impossible to catch up on pre-existing delays. Also under schedule A, south-west moving trams of line 13 are placed directly behind vehicles of lines 15 and 16, thus resulting in a high receptiveness for delay. Schedule B resolves those issues, resulting in a decrease in delay of 49 percent.

Line		1	3	4	5	6	7	8	9	12	13	15	16	18	19
Ø Delay	A	33,7	11,5	24,9	7,3	21,6	25,2	58,6	14,1	14,7	44,2	13,6	33,3	24,5	27,1
	B	19,2	10,9	24,3	6,5	15,3	18,6	16,2	12,3	13,5	22,4	12,5	33,2	19,6	23,9
Abs. gain		14,5	0,6	0,6	0,8	6,3	6,6	42,4	1,8	1,2	21,6	1,1	0,1	4,9	3,2
Rel. gain		0,43	0,05	0,02	0,11	0,29	0,26	0,72	0,13	0,08	0,49	0,08	0,00	0,20	0,12

Table 3: Comparing schedules: Lines

Simulation data collected at the important hubs Barbarossaplatz (BAB-1 to BAB-4), Ebertplatz (EBP-1 to EBP-4), and Neumarkt (NEU-1 to NEU-4) is presented in table 4. Under schedule B, delay was reduced significantly at eight of those platforms, staying on about the same level at further two (see figure 7). The increase in punctuality can be explained by the better reliability of the frequenting lines under the optimized schedule.

The rise in delay at platforms EBP-3 (8.5 seconds) and NEU-4 (2.3 seconds) remains to be explained. Both platforms are preceded by highly frequented tracks, used by lines whose punctuality does not improve significantly by applying schedule B. This would partly explain the average delay to be stagnant. Furthermore, these tracks are merged by arrays of underground track switches, which have to be negotiated by every incoming vehicle. The timing changes from schedule A to schedule B could yield adverse configurations of switch tongues, which would explain the observed small increase in delay.

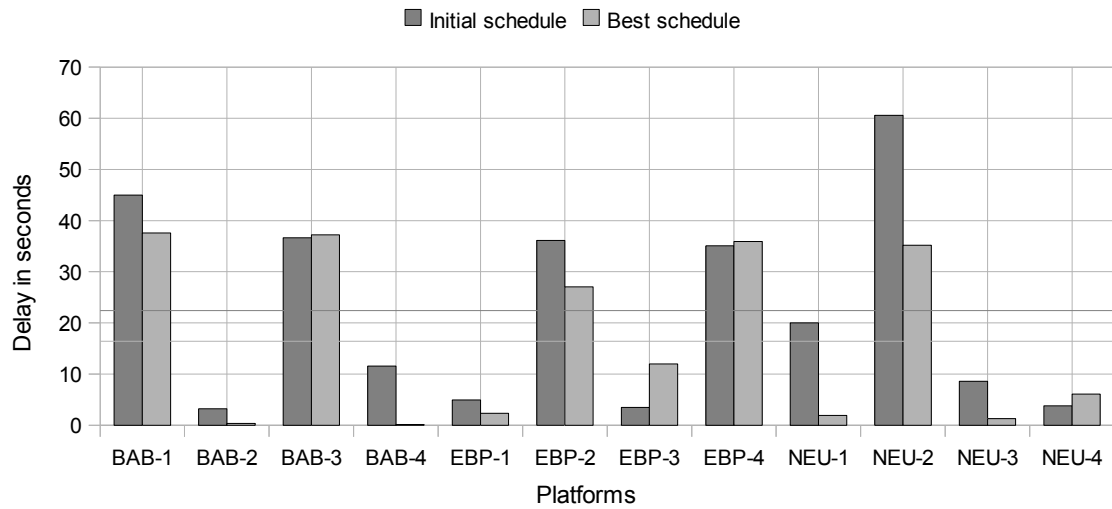


Figure 7: Average delay at platforms

Platform		BAB-1	BAB-2	BAB-3	BAB-4	EBP-1	EBP-2	EBP-3	EBP-4	NEU-1	NEU-2	NEU-3	NEU-4
Ø Delay	A	45,0	3,2	36,7	11,6	5,0	36,1	3,5	35,1	20,0	60,6	8,6	3,8
	B	37,6	0,4	37,2	0,1	2,3	27,1	12,0	35,9	1,9	35,2	1,3	6,1
Abs. gain		7,4	2,9	-0,6	11,5	2,6	9,1	-8,5	-0,8	18,1	25,4	7,3	-2,3
Rel. gain		0,16	0,89	-0,01	0,99	0,53	0,25	-2,41	-0,02	0,9	0,42	0,85	-0,61

Table 4: Comparing schedules: Platforms

4 Conclusion and future work

In this paper we presented a tool chain to generate and evaluate tram schedules. The described optimization module is capable of generating robust time tables which fulfill planning requirements as found in real world projects. We also presented a simulation engine which makes it possible to test real and generated schedules for their applicability and so to further validate them.

We applied the described tool chain to our hometown Cologne's mixed tram network. A random but valid time table A was compared to the resulting best schedule B. As to be expected, the average delay under schedule B is significantly lower than that under schedule A. All lines gain punctuality, though at some core platforms the average delay rises for up to nine seconds.

In further steps more detailed studies of tram networks and schedules will be carried out, including Cologne's new underground tracks currently under construction, which are designed to relieve the central Neumarkt tunnel. We found it desirable to be able to manually apply small incremental changes to a schedule while getting instant visual assessment of expected consequences. A tool with those capabilities is in the planning stage. Furthermore the optimizer module will be parallelized to further reduce its run time. Especially the applied branch-and-bound algorithm's load can be balanced relatively easy, so the application should scale well.

Acknowledgements

This work was partially supported by *Rhein Energie Stiftung Jugend Beruf Wissenschaft* under grant number W-10-2-002. We thank the other members of our work group Patrick Kuckertz and Bert Randerath.

References

- [1] Bampas, E., Kaouri, G., Lampis, M., Pagourtzis, A.: *Periodic Metro Scheduling*. In: Proceedings of ATMOS, 2006
- [2] Banks, J., Carson, J.S., Nelson B.L., Nicol D.M.: *Discrete-Event System Simulation*. Pearson, 2010
- [3] Cacchiana, V., Caprara, A., Fischetti, M.: *A Lagrangian Heuristic for Robustness, with an Application to Train Timetabling*. Transportation Science, to appear
- [4] Caimi, G., Fuchsberger, M., Laumanns, M., Schüpbach, K.: *Periodic Railway Timetabling with Event Flexibility*. In: Networks, 2010, Volume 57, Number 1, pp. 3-18
- [5] Dréo, J., Pétrowski, A., Siarry, P., Taillard, E.: *Metaheuristics for Hard Optimization*. Springer, 2006
- [6] Franz, S.: *Entwurf und Entwicklung eines mehrstufigen Optimierungsverfahrens für Stadtbahnfahrpläne unter Berücksichtigung verkehrsplanerischen Vorgaben*. Diplomarbeit, Univ. Köln, 2011
- [7] Genç, Z.: *Ein neuer Ansatz zur Fahrplanoptimierung im ÖPNV: Maximierung von zeitlichen Sicherheitsabständen*. Dissertation, Mathematisch-Naturwissenschaftliche Fakultät, Universität zu Köln, 2003
- [8] Hu, T.C.: *Combinatorial Algorithms*, Addison Wesley, 1982
- [9] Lückemeyer, G.: *A Traffic Simulation System Increasing the Efficiency of Schedule Design for Public Transport Systems Based on Scarce Data*. Dissertation, Shaker Verlag, 2007
- [10] Lückemeyer, G., Speckenmeyer, E.: *Comparing Applicability of Two Simulation Models in Public Transport Simulation*. In: Becker, M., Szczerbicka, H. (Ed.): Proceedings of ASIM 2006, Hannover, 2006
- [11] Lückerath, D.: *Entwurf und Entwicklung einer Anwendung zur parallelen Simulation von schienengebundenem Öffentlichen Personennahverkehr*. Diplomarbeit, Univ. Köln, 2011
- [12] Lückerath, D., Ullrich, O., Speckenmeyer, E.: *Modeling time table based tram traffic*. In: Bödi, R., Maurer, W. (Ed.): Proceedings of ASIM 2011, Winterthur, 2011
- [13] Middelkoop, D., Bouwman, M.: *SIMONE: Large Scale Train Network Simulations*. In: B.A. Peters, J.S. Smith, D.J. Medeiros, M.W. Rohrer (Ed.): Proceedings of the 2011 Winter Simulation Conference, Arlington, 2001
- [14] Nagel, K., Schreckenberg, M.: *A cellular automaton model for freeway traffic*. In: Journal de Physique I, Volume 2, Issue 12, December 1992, pp. 2221-2229
- [15] Nash, A., Huerlimann, D.: *Railroad Simulation Using OpenTrack*. In: Allan, J., R.J. Hill, C.A. Brebbia, G. Sciutto, S. Sone (Ed.): Computers in Railways IX, WIT Press, Southampton, 2004, pp. 45-54
- [16] Schöbel, A.: *A Model for the Delay Management Problem based on Mixed-Integer-Programming*. In: Proceedings of ATMOS, 2001
- [17] Speckenmeyer, E., Li, N., Lückerath, D., Ullrich, O.: *Socio-Economic Objectives in Tram Scheduling*. Technical Report, Universität zu Köln, to appear

- [18] Suhl, L., Mellouli, T.: *Managing and preventing delays in railway traffic by simulation and optimization*. In: Mathematical Methods on Optimization in Transportation Systems 2001, pp. 3–16
- [19] Verband Deutscher Verkehrsunternehmen e.V.: *VDV-Standardschnittstelle Liniennetz/Fahrplan*, VDV-Schriften 452, 2008

A simulation based approach on robust airline job pairing

Patrick Kuckertz^{a,b}, Oliver Ullrich^a, Hubert Randerath^b

^aInstitut für Informatik, Universität zu Köln

^bInstitut für Nachrichtentechnik, Fachhochschule Köln

kuckertz|ullrich@informatik.uni-koeln.de

hubert.randerath@fh-koeln.de

Abstract

Job pairing, i.e. the composition of duty rosters from single activities, is an important part of the airline operations planning process. With labor costs being a major factor in an airline's cost structure, such personnel schedules have to ensure efficiency to be of practical relevance. At the same time they have to improve customer acceptance by offering best possible robustness, keeping inevitable local delays from spreading through the airline's flight network.

In this paper we present a project currently in development which aims for generating robust personnel schedules for airline operations. The resulting tool set will allow us to effectively allocate flight personnel, using optimization and simulation techniques to generate and compare schedules with respect to their applicability and their demand for standby personnel, and to evaluate them prior to their implementation in the field.

This paper begins with a short introduction of the airline planning process, focusing on the job pairing problem. We then describe our project, presenting our optimization and simulation approaches.

1 Introduction

During their extensive process of operations planning airlines are challenged by a set of interdependent planning problems (see figure 1). This process starts with the design of the flight schedule and the assignment of aircraft types to the flights. It continues with the routing of individual aircrafts and the determination of crew schedules, and is concluded by short-term flight plan management and recovery measures. Within this process the construction of a valid and efficient operations schedule for flight personnel is one of the most complex tasks. A part of this task is the crew pairing procedure which is concerned with the construction and optimal combination of anonymous crew rotations in order to cover all flights of a given flight schedule while complying with a multitude of regulations coming from labor legislation (see [4]), union agreements and operational procedures.

The majority of existing studies analyzes the *crew pairing problem* (CPP) against a cost reducing background due to its high economic significance (see e.g. [2], [8]). The use of costs as exclusive quality objective however may lead to personnel schedules with a low degree of fault tolerance and a high degree of delay propagation. In order to confine occurring disruptions and to support practical applicability a personnel schedule has to be robust.

This paper describes and outlines a project in development which aims for a better understanding of robust personnel schedules. The project follows a more detailed approach than the CPP describes by not dividing tasks on crew level but on the level of individual crew members, leading to a *job pairing problem* (JPP). In the context of robustness this approach is more realistic, since delays and drop outs of individuals can be accounted for. Furthermore individual qualifications can be incorporated which enables the analysis of efficient substitution strategies and standby structures. This approach also allows a more detailed view on the fault propagation in personnel employment strategies. Schedules resulting from our optimization process are to be simulated under realistic conditions. A concurrence of results of a robustness assessment by a static objective function with those of a dynamic simulation would demonstrate a certain suitability of practical use of our approach.

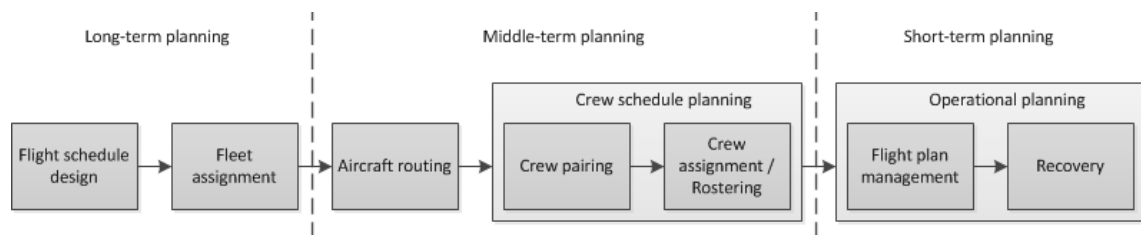


Figure 1: Operations planning process of airlines

The remainder of this paper is organized as follows. In section 2 our project is introduced. Four subsections explain technical backgrounds and give insights into different project modules with their objectives and approaches. Section 3 concludes with a summary and some thoughts on future work.

2 Project approach

Our project *Dynamic Optimization of Group Schedules (DOGS)* is build around a database containing airline schedule and network data. A network generator, simulation, optimization, and evaluation modules are connected via operations on the database and through XML configuration files (see figure 2).

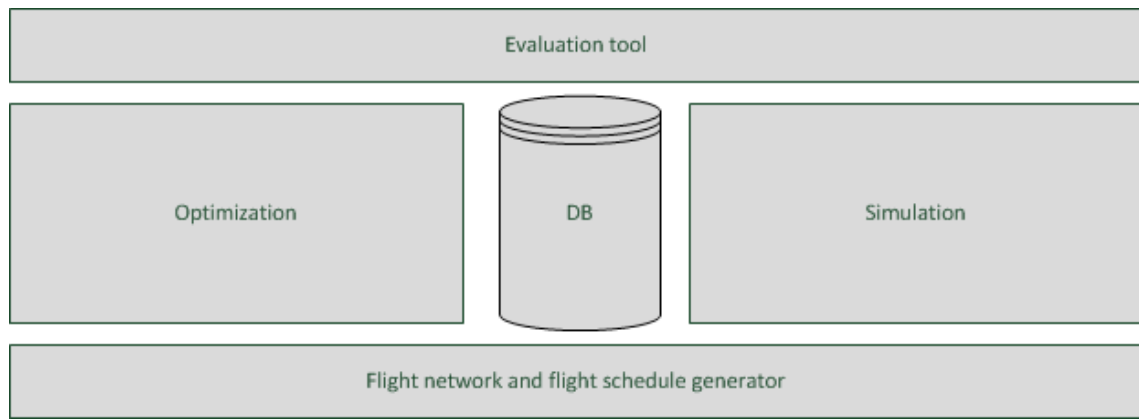


Figure 2: Modular project architecture

2.1 Technical background

The development of aircraft rotations is preceded by flight schedule design and fleet assignment which are both based upon passenger demand forecasts (see figure 1). Results of these planning steps are an airline's flight connections as well as the allocation of aircraft types to these connections. *Flight connections* are defined by their origin and destination airports as well as by their departure and arrival times. Aircraft types differ e.g. in passenger capacities and personnel requirements. All this information merges into the flight schedule which serves as input to the crew scheduling process. During the job pairing, which is part of crew scheduling, tasks combined and packaged. In the following crew assignment or rostering phase these work packages are assigned to members of the flight personnel.

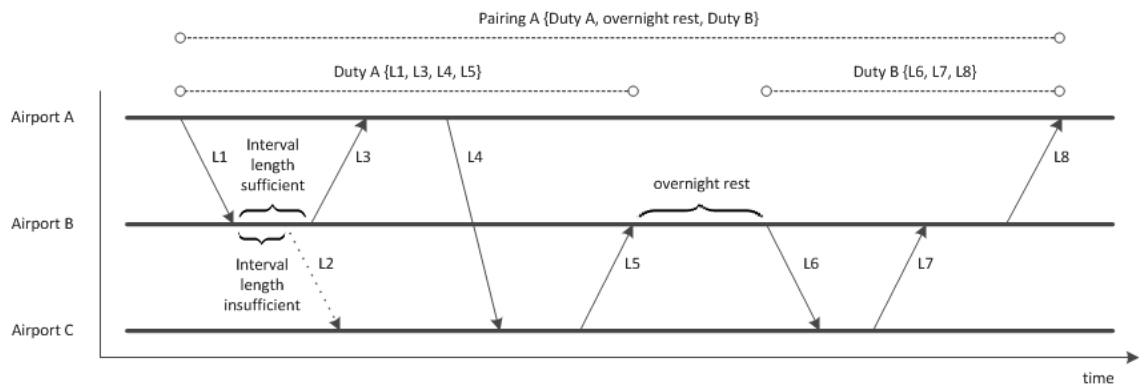


Figure 3: Diagram of a flight schedule fragment

A *flight schedule* provides detailed information about flight connections, informing about the time intervals and weekdays a connection is carried out. Connections in a schedule are identified by flight numbers while individual flights are identified by their connections and days of departure. There are different types of connections to be found in a schedule, depending on their number of *flightlegs*. Figure 3 shows a diagram of a flight schedule fragment in which flightlegs are pictured as arrows. A non-stop connection, also called *non-stop flight*, has no stops between its airports of origin and destination and therefore only one flightleg. A direct connection, also called *direct flight*, has at least one intermediate stop and

thus consists of two or more flightlegs. It does not include any changes of aircraft and its flightlegs operate under a single flight number. Examples can be found in figure 3, connecting the airports A and C. A non-stop flight between these airports consists only of flightleg L4, while a direct flight stopping at airport B consists of flightlegs L1 and L2. Within a schedule the type of a connection is denoted by the number of stops it includes.

For the JPP not all connections found in a flight schedule are considered. To avoid redundant information direct flights are ignored since they are composed of non-stop flights already named in the schedule. An airline schedule often contains connections actually carried out by alliance partners. This way a flight might be offered by different airlines under more than one flight number, allowing customers to book at their preferred airline in their own language and currency. Those *code share flights* have to be disregarded since we only want to solve the JPP for single airlines.

The connections an airline offers form its *flight network* which can be viewed as a graph with airports being nodes and flight connections being directed edges. Flight networks of large airlines often show hub and spoke structures which support efficient operations (see [7]). Coordinated with adequate schedules they provide passengers with a manifold choice of connections and short waiting times. Commonly airlines choose large airports with strategically favorable positions within their networks to serve as *hubs*. Hubs are usually fully interconnected. *Spokes* connect the hubs to all other airports which are accessed by the airline. Within such a network structure the surrounding airports are normally not interconnected with the possible exception of *shuttle connections* extending spokes to other smaller airports which have no connections to a hub themselves.

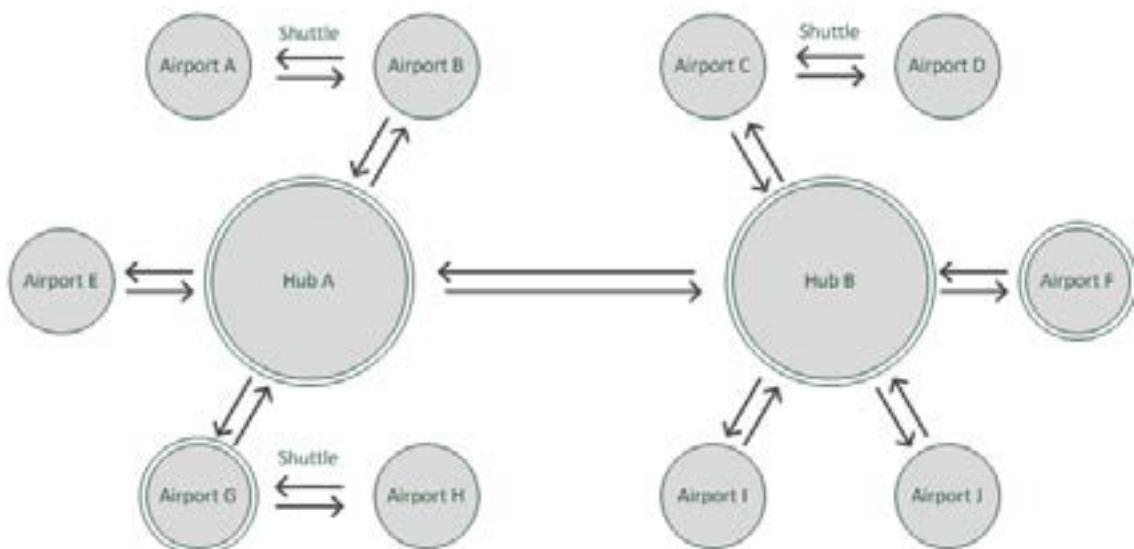


Figure 4: Example of an airline hub and spoke network structure (double lined circles picture crew bases)

Also depending on their relative position within the network airlines choose at least one airport to serve as crew base. An airport is called a *crew base* if it is the place of employment of airline's personnel. Another term used by airline personnel is *home base* which is the employees' view on a crew base. Each employee has exactly one home base while a crew base

must be home base to at least one employee. Figure 4 illustrates the described network structure including crew bases.

2.2 Modeling

Following our job pairing approach, each flightleg brings up a number of *jobs*, i.e. single tasks, all requiring individual combinations of professions and qualification profiles. Depending on aircraft type, number of passengers and flight distance, different sizes of flight deck and cabin crews are mandatory. Different aircraft types and countries of origin and destination require different piloting, language and service skills.

During the job pairing the jobs of all flightlegs have to be assorted into work packages which will be assigned to flight personnel in the following rostering process. Jobs are bundled into *duties* which can be viewed as single workdays. The work packages, called *pairings*, again are bundles of duties with overnight rest periods in between (see figure 3). They are round trips, starting and ending at the same crew base. The allowed numbers of take-offs and landings within duties and pairings, maximum flying and service times, minimum rest periods and other work rules concerning the packaging process are determined by public authorities and are further subject to operational procedures and union agreements. During the pairing process it may become necessary to reallocate flight personnel to other airports. The transportation of off duty personnel is called *deadhead*.

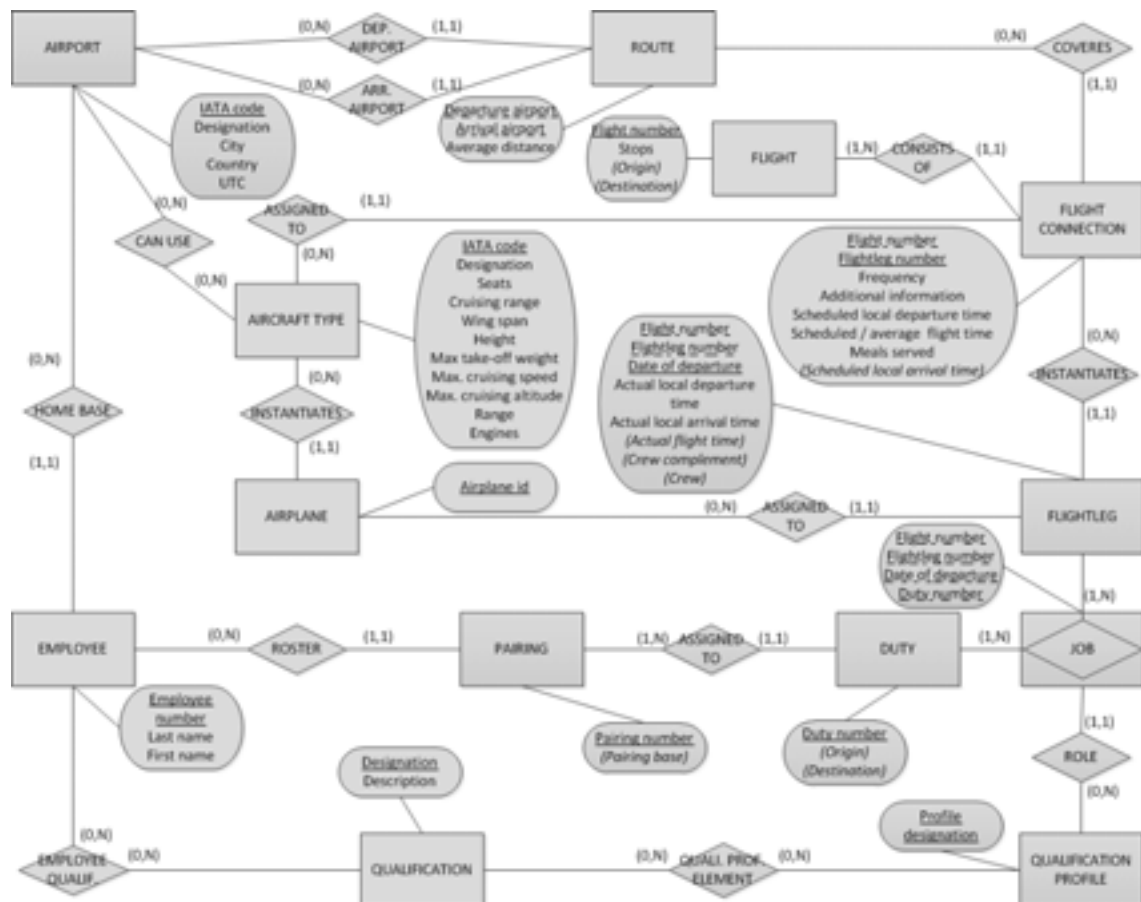


Figure 5: Entity relationship diagram of fundamental data elements within an airline operations planning process

The current state of our project's data model is pictured in figure 5. The entity relationship diagram (see [6]) illustrates the composition and relationships of the entities substantial for an airline operations planning process. The structure of the project's database is derived from this diagram.

The scope of the project includes the development of a flight network and flight schedule generator (see figure 2). With this tool a set of realistic and hypothetical test instances are to be generated to support the robustness analysis. Assessing diverse instances may yield information about the underlying graph structures' influences on the robustness potential. The network graphs of past flight schedules undergo a structural analysis regarding connectivity, reachability and distance measures. Once adequate parameters and realistic specifications have been found the algorithm's method of operation has to be determined. After applying a few modifications the R-MAT generator described in [3] might be a promising candidate.

2.3 Optimization

The JPP is a large scheduling problem whose complexity grows with each additional variable representing jobs, qualification requirements and types of work shifts. Due to its combinatorial structure the number of possible solutions is huge. Problem instances with over 1,000 flights a day and a monthly coordination of over 15,000 crew members are not uncommon. In addition a wide spectrum of government regulations upholding aviation safety have to be respected (see [4]).

Cost reduction is the traditional motivation of research on this topic. Personnel costs account for the second highest part of an airline's overall expenses, right after fuel costs which hardly can be impaired (see [7]). The primary aim of the project presented in this paper however is not to reduce the costs of a flight schedule but to improve its robustness. A robust schedule is to be distinguished by a low rate of delay propagation and a high fault tolerance. Delays and drop outs of personnel members or flightlegs cannot be fully avoided, but measures can be taken to reduce their occurrence probabilities and possible consequences for the flight schedule.

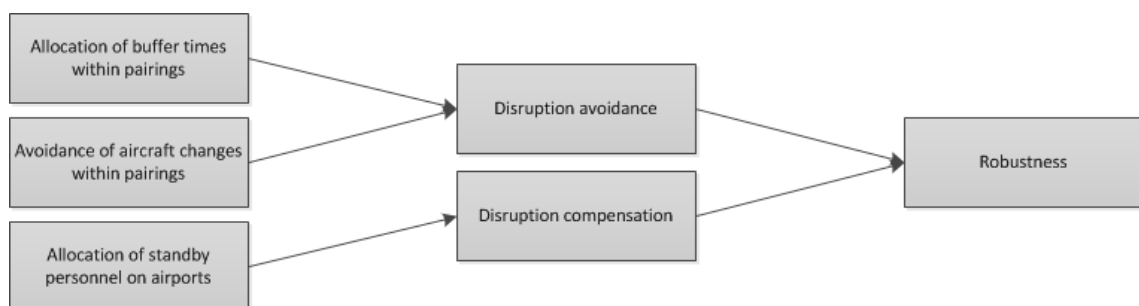


Figure 6: Overview over robustness improving measures during crew pairing

Each step of the airline operations planning process has its own options to account for disruptions. Crew pairing provides measures to avoid disruptions as well as to compensate for them (see figure 6). Our project's optimization approach focuses on disruption avoidance. One policy to create flight schedules with a maximum of stability is to demand a minimum time

interval between two consecutive jobs to buffer delays. Figure 3 illustrates that a job of flightleg L2 cannot follow a job of L1 within one personnel member's duty because of the insufficient length of the intermediate time interval. Another measure is the minimization of the number of personnel's aircraft changes within a pairing, reducing dependencies between aircraft rotations and hence delay propagation.

The CPP is often discussed in literature, and a plurality of mathematical models and solution approaches are presented. For our project we haven't yet decided which approaches to adjust to our JPP. Thus we describe a common crew pairing procedure at this point. Crew pairing divides all flightlegs of a given flight schedule into pairings. The problem of covering each flightleg exactly once by a single pairing is described by the *set partitioning problem* (see [2]). In order to include deadheads into the process of optimization the coverage of a flightleg by more than one crew, and hence more than one pairing, must be allowed. The formulation as a *set covering problem* includes the condition to cover each flightleg at least once (see [7]). Solving the CPP for a major airline includes a large set of pairings which leads to a huge number of possible combinations.

Because of its large scale the CPP is often divided into a master problem and a subproblem. The subproblem, including only a manageable amount of pairings, is solved and then iteratively expanded by column generation. Applying the local search heuristic 2-opt (described in [8]), the size of the subproblem stays constant because promising new pairings replace pairings of the previous solution. A common approach for approximating a global optimum is described by the *restricted shortest path problem* (see [9]). Here a problem's graph structure is used to evaluate the quality of all pairings outside the current subproblem so that only the most promising pairings have to be calculated in the next iteration.

Commonly these procedures are used to optimize a cost function. The costs of a pairing can be determined by measuring its time consumption. Gopalakrishnan et. al. define the costs by the difference between the *time away from base* and the *flying time* (see [7]). The time away from base is the time interval between leaving and returning to a crew base. The flying time is the summation of the differences between the arrival and departure times of all the pairing's legs. This calculation determines non-productive waiting times of pairings. Analogous to [5] we want to treat the aspect of robustness using penalty costs for insufficient intervals between flightlegs and for aircraft rotation changes. The formulation of the robustness objective as a cost reduction problem allows the use of already approved optimization procedures.

2.4 Simulation

We plan to develop a model and implement an application to simulate flight schedules. This will enable us to evaluate given personnel schedules prior to their implementation in the field and to compare schedules generated by optimization methods with respect to their applicability. Schedules considered feasible by a static objective function, can be evaluated for their dynamic applicability, and thus lead to a higher degree of validity.

Another focus of the simulation system lies on disruption compensation, i.e. to evaluate a given personnel schedule for its recoverability characteristics (see figure 6). For this, we simulate a personnel schedule under a predefined flight schedule, as well as given fault tolerance policies, and take note of requested numbers and qualifications of standby or reserve personnel. After an adequate number of simulation runs, we thus can recommend standby policies for each airport and time slot. A further aim is to assess different scenarios' impact on schedules to reveal consequences of temporary resource losses, e.g. damaged runways or raised probabilities of staff shortage in certain personnel clusters.

The simulation system currently under development is based on the event-based simulation approach (as described in [1]). Here, events of certain types yield state changes, which manifest at discrete points in time. The events are administrated in a priority queue, ordered by the time stamp of their occurrence. In a loop, the simulation engine extracts the event with the lowest time stamp, advances the simulation time accordingly, processes the event, updates the affected entities' states, and generates appropriate follow-up events, which are again entered into the priority queue. This is repeated until the priority queue is empty, i.e. all scheduled actions of the operational period are processed, and no more follow-up events are generated. Using this technique, scenarios and occurrences of rare events can be handled by injecting corresponding simulation events into the priority queue prior to the simulation run.

In our simulation system airplanes encapsulate most of the simulation dynamics. Planes change their state at events like landing or opening doors. Main attributes are specified by the plane type, which holds functions for capacity, number and position of doors, avionic capabilities, etc. Combined with requirements of flight types, e.g. the number and qualifications of flight attendants, the demand for personnel is calculated. While processing these state changes, the simulation engine takes note of statistical data about delays and dynamic requests for standby personnel.

3 Conclusion and future work

In this paper we presented our project currently in development on robust airline job pairing. After explaining the context of the general airline operations planning process we gave an insight into the modules of the intended project architecture. The models and approaches presented differ from other models of airline personnel planning by considering single crew members instead of whole crews. We illustrated the common graph structure of flight networks and its potential influence on the JPP.

Since this project is still in its beginnings a lot of work has yet to be done. At the moment the real-world model, the setup of the database and a robust optimization program are refined in parallel. We look forward to our next milestone, the completion of the flight network and flight schedule generator, and to the comparison of the potentials of different graph structures on options of robust job pairing.

Acknowledgments

This work was partially supported by *Rhein Energie Stiftung Jugend Beruf Wissenschaft* under grant number W-10-2-002. We thank the other members of our work group Daniel Lückerrath and Ewald Speckenmeyer.

References

- [1] Banks, Jerry; Carson II, John S.; Nelson, Barry L.; Nicol, David M.: *Discrete-Event System Simulation*. Pearson, 2010.
- [2] Borndörfer, Ralf; Schelten, Uwe; Schlechte, Thomas; Weider, Steffen: *A Column Generation Approach to Airline Crew Scheduling*. Operations Research Proceedings 2005, pp. 343-348. Springer, 2006.
- [3] Chakrabarti, Deepayan; Zhan, Yiping; Faloutsos, Christos: *R-MAT: A Recursive Model for Graph Mining*. Proceedings of the fourth SIAM International Conference on Data Mining, pp. 442-446. Society for Industrial and Applied Mathematics, 2004.
- [4] Commission of the European Union: *Regulation (EC) No 859/2008*. Subsections N, O and Q. Official Journal of the European Union, 2008.
- [5] Dück, Viktor; Ionescu, Lucian; Kliever, Natalia; Suhl, Leena: *Increasing stability of crew and aircraft schedules*. Transportation Research Part C: Emerging Technologies, Vol. 20, Issue 1, pp. 47-61. Elsevier Ltd., 2011.
- [6] Elmasri, Ramez; Navathe, Shamkant B.: *Fundamentals of Database Systems*. Pearson International Edition, pp. 57-129. Addison-Wesley, 2007.
- [7] Gopalakrishnan, Balaji; Johnson, Ellis L.: *Airline Crew Scheduling: State-of-the-Art*. Annals of Operations Research, Vol. 140, pp. 305-337. Springer, 2006.
- [8] Graves, Glenn W.; McBride, Richard D.; Gershkoff, Ira; Anderson, Diane; Mahidhara, Deepa: *Flight Crew Scheduling*. Management Science, Vol. 39, No. 6, pp. 736-745. The Institute of Management Sciences, 1993.
- [9] Lavoie, Sylvie; Minoux, Michel; Odier, Edouard: *A new approach for crew pairing problems by column generation with an application to air transportation*. European Journal of Operational Research, Vol. 35, Issue 1, pp. 45-58. Elsevier Science Publishers B. V., 1988.

Cognitive Aspects of Traffic Simulations in Virtual Environments

Sven Seele¹, Thomas Dettmar¹, Rainer Herpers^{1,2,3},
Christian Bauckhage⁴, Peter Becker¹
Contact: sven.seele@h-brs.de

Introduction

Traffic simulations for virtual environments are concerned with the behavior of individual traffic participants. The complexity of behavior in these simulations is often rather simple to abide by the constraints of processing resources. In sophisticated traffic simulations, the behavior of individual traffic participants is also modeled, but the focus lies on the overall behavior of the entire system, e.g. to identify possible bottle necks of traffic flow [8].

One objective of the FIVIS project [3][4] is to create a realistic bicycle simulator to be used for road safety training of children. For that the traffic agents need to be persistent and react realistically to changing environmental conditions in real-time. However, within the context of virtual environments, traffic simulations need to be realistic only within the viewing frustum of the visualization setup. If simulated content is not directly visible, there is no need to spend major computing resources on its calculation.

Training simulators are widely used to increase safety in almost all areas of modern transportation (car, train, plane, etc.). With their help trainees are taught how to behave in dangerous situations without the risk of causing physical harm. The FIVIS project attempts to apply this advantage to bicycles, as there seem to be no such simulators commonly available today. So far dangerous traffic situations are realized in general by scripted events, defined for each car or any other traffic agent. This process is tedious, inflexible and might require trainees to follow a specific route.

Thus, for the simulation of traffic agents in virtual environments, a cognitive traffic modeling approach is proposed that combines techniques from the field of traffic research and cognitive architecture research to address the stated challenges within a project called AVeSi (“Agentenbasierte Verkehrssimulation mit psychologischen Persönlichkeitsprofilen” – Engl. “Agent-based traffic simulation with psychological personality profiles”).

¹ Bonn-Rhine-Sieg University of Applied Sciences

² York University, Canada

³ University of New Brunswick, Canada

⁴ University of Bonn

Modeling of Traffic Agents

The AVeSi project aims at developing an autonomous traffic simulation as an extension to the FIVIS bicycle simulator that achieves the aforementioned criteria of persistence and realistic behavior. The concept developed to fulfill this objective is based on previous research by Kutz et al. [5], who planned to utilize psychological personality profiles for traffic agents in virtual environments. The new concept extends these ideas and is summarized below.

Persistent Traffic Agents

An obvious way to save computation resources in traffic simulations for virtual environments is to remove traffic agents from the simulation once they leave the user's field of view. This can lead to situations which are irritating to a user, but simulating all agents at all times would be too costly. Thus, the idea is to combine a microscopic simulation within the user's vicinity with an additional macroscopic simulation, simulating traffic agents with less detail. Since traffic can be described by density and flow on a macroscopic level, fluid dynamics can be used to simulate it. Applying smoothed particle hydrodynamics (SPH) to approximate common numerical solutions is proposed to decrease processing time; similar to the approach presented by Rosswog and Wagner [6][7]. Simultaneously, this approach should establish a link between the macro and micro level, since the approach is based on individual particles. Rosswog and Wagner only briefly mention the macro-micro link as possible benefit of their method in [6], but do not provide any details.

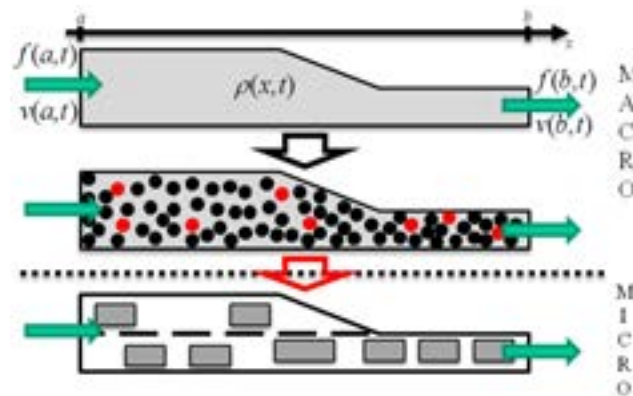


Figure 1: An illustration exemplifying the idea of using a particle-based approach to simulate a macroscopic traffic simulation (top) and to establish a macro-micro link (bottom) using the red particles to represent individual traffic participants. (Illustration based on [1]; $f(a,t)$ denotes traffic flow, $v(x,t)$ velocity, and $\rho(x,t)$ vehicle density)

Cognitive Traffic Agents

To achieve a realistic traffic simulation, agents need to show misbehavior like their human counterparts, but they must do so only in situations where it makes sense to an observer. Therefore, the authors believe that it is not enough to deviate from “normal” behavior by introducing fuzzy logic or random events, which is done frequently (e.g. [2]). Instead an agent must consider its current perceivable situation when determining its actions. For this purpose

the relevant cognitive processes should be modeled to induce realistic human behavior in traffic (e.g. perception, anticipation, decision making, etc.). Particularly modeling the influence of risk propensity on action selection will be a central aspect of the project. Other factors which affect the behavior as well (e.g. mood, age, gender, time of day, weather, etc.) might also be considered. The application of ideas from cognitive architecture research to the domain of microscopic traffic simulation is anticipated to achieve these objectives.

In conclusion, modeling persistent traffic agents whose behavior is inspired by human cognitive processes might improve road safety training applications, since they reflect real world traffic conditions more realistically. Implementing the current ideas might be challenging, but if successful, other driving/traffic simulators and in particular traffic simulations in digital games will benefit from this research as well.

Acknowledgments

The FIVIS project was funded by the BMBF-FH³ Program “Angewandte Forschung an Hochschulen im Verbund mit der Wirtschaft”, funding for the FIVIS 2 project was provided by the DGUV (FP 307), and the AVeSi project is being funded by the FHprofUnt Program of the BMBF (17028X11).

References

- [1] Bungartz, Hans-Joachim et al.: *Modellbildung und Simulation*, Springer, 2009
- [2] Fellendorf, Martin, Vortisch, Peter: Microscopic Traffic Flow Simulator VISSIM, In *Fundamentals of Traffic Simulation*, Springer, 2010
- [3] Herpers, Rainer, et al.: Multimedia Sensory Cue Processing in the FIVIS Simulation Environment, In *Multiple Sensorial Media Advances and Applications: New Developments in MulSeMedia*, IGI Global, 2011
- [4] Herpers, Rainer, et al.: FIVIS – A Bicycle Simulation System, World Congress on Medical Physics and Biomedical Engineering (WC 2009), IFMBE Proceedings Vol. 25/4, Springer, 2009
- [5] Kutz, Michael, Herpers, Rainer: Urban Traffic Simulation for Games, In *Future Play '08 Proceedings of the 2008 Conference on Future Play: Research, Play, Share*, ACM, 2008
- [6] Rosswog, Stephan, Wagner, Peter: “Car-SPH”: A Lagrangian Particle Scheme for the Solution of the Macroscopic Traffic Flow Equations, In *Traffic and Granular Flow '99*, 1999
- [7] Rosswog, Stephan, Wagner, Peter: Towards a Macroscopic Modeling of the Complexity in Traffic Flow, In *Phys. Rev. E Stat. Nonlin. Soft Matter Phys* 65(3 Pt 2A):036106, 2002
- [8] Treiber, Martin, Kesting, Arne: *Verkehrsdynamik und –simulation*, Springer, 2009

Standardisierte effiziente Speicherung von Simulationsergebnissen

Dr. Ingrid Bausch-Gall
Ingrid.Bausch-Gall@bausch-gall.de
BAUSCH-GALL GmbH
Wohlfartstraße 21 b, 80939 München

Kurzfassung

Alle kennen das Problem: Jedes Simulationsprogramm legt seine Daten in einem eigenen internen Format ab. Die meisten Simulationsprogramme stellen nur unzureichende Möglichkeiten zur Auswertung bereit. Hierfür werden zumeist Programme wie Python, MATLAB oder eigene Programme verwendet. Um diese Simulationsergebnisse zu verwenden, müssen die Ergebnisse exportiert werden. Die Ergebnisse werden häufig in Formate wie CSV, ASCII oder MAT-Files exportiert. CSV- und ASCII-Files eignen sich nicht zur Abspeicherung großer Datenmengen, MAT-Files können nur in MATLAB effizient eingelesen werden. Daher beschäftigt sich die Modelica Ass. mit der Entwicklung eines standardisierten, herstellerunabhängigen Dateiformats zur effizienten Speicherung von Simulationsergebnissen.

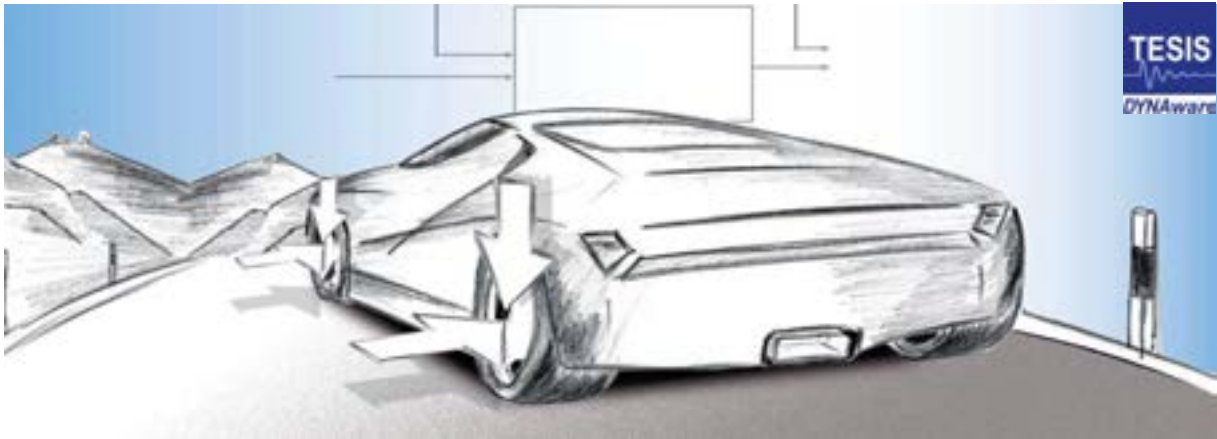
Auf der ASIM-Tagung in Winterthur 2011 haben wir den damaligen Entwicklungsstand vorgestellt. Das Format soll die folgenden Ansprüche erfüllen:

- Große Datenmengen lassen sich effizient abspeichern, d.h. sie lassen sich schnell schreiben und die Dateien sollen so klein, wie möglich sein.
- Einzelne Signale oder Teile der Signale lassen sich auch aus großen Dateien effizient und schnell lesen.
- Alle Informationen, die in der FMI-Definition festgelegt wurden, lassen sich übernehmen. Dies sind z.B. Informationen zu den Zeitreihen, wie deren Einheit oder Informationen zur Berechnung, wie Tool, Autor oder Verfahren.
- Es soll ein offenes und international anerkanntes Dateiformat sein.
- Zusätzliche Informationen, wie Diagramme, Dokumentation sollen sich abspeichern lassen.
- Einfacher Zugriff aus C, C++, MATLAB, Python, Java und FORTRAN soll möglich sein.
- Das Format soll weit verbreitet und so zukunftssicher wie möglich sein.

In diesem Vortrag wird über den Stand der Entwicklung berichtet. Zwei unterschiedliche Konzepte stehen zur Zeit zur Diskussion. In diesem Vortrag werden die beiden Konzepte vorgestellt und deren Vor- und Nachteile abgewogen.

1 Literatur

- [1] Functional Mock-up Interface for Model Exchange, MODELISAR (07006)
Document version 1.0, January 26, 2010
www.functional-mockup-interface.org
- [2] Modelica – A unified Object-Oriented Language for Physical Systems
Modeling Language Specification, Version 3.2, March 24, 2010
- [3] HDF5: www.hdfgroup.org
- [4] ASAM-Standards: www.asam.net
- [5] Bausch-Gall, Ingrid, Pfeiffer, Andreas: Standardisierte effiziente Ablage von
Simulationsergebnissen in HDF5
ASIM Symposium Simulationstechnik, Winterthur 2011

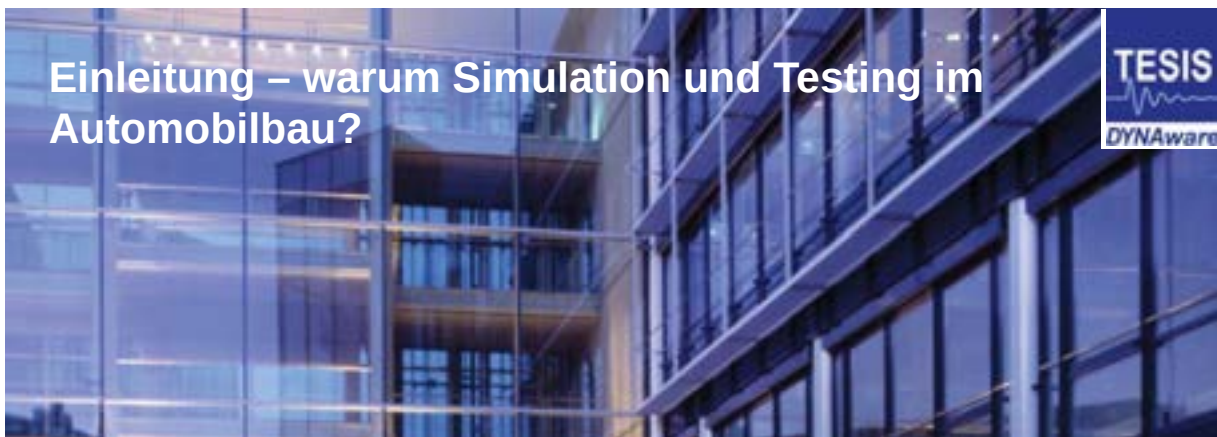


Simulation 2.0: Simulationsbaukasten und Team-Modellierung

Vortrag an der FH Ostfalia, ASIM 2012

Daniel Frechen TESIS DYNAware GmbH

24. Februar 2012

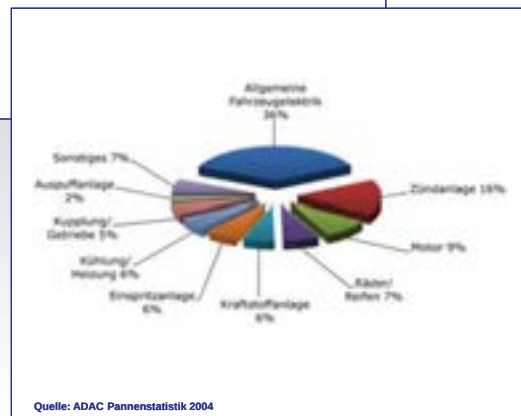


Einleitung – warum Simulation und Testing im Automobilbau?



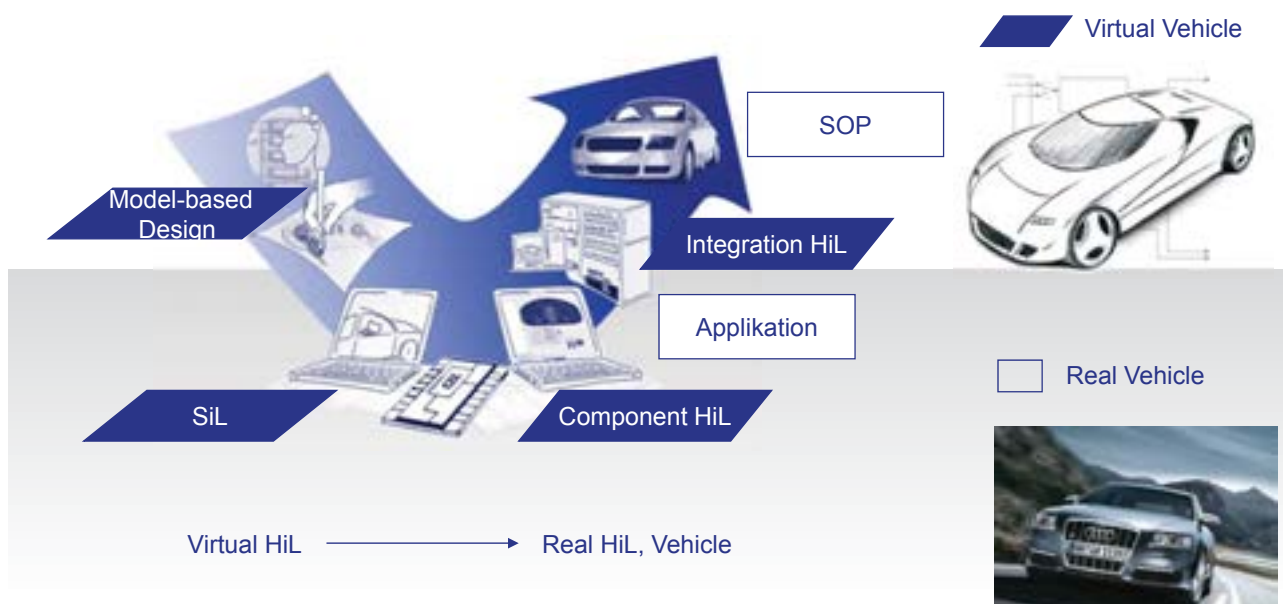
Einleitung – warum Simulation und Testing im Automobilbau?

- mehr als 80 Steuergeräte in Oberklasse-Pkw, stark vernetzt
- Innovationen heute fast nur noch in Elektronik
- Großteil der Pannen fallen auf ECUs



3

Eingliederung der Simulation in Gesamtprozess



Einleitung – was wird simuliert und getestet?



5

Einleitung – was wird simuliert und getestet?



Untersuchung von Fahrwerks-Steuergeräten am HiL/SiL:

Komponententests

- elektrische Fehler (Leitungsunterbrechung, Kurzschlüsse)
- Kommunikations-Fehler (Ausfall von CAN-Botschaften)

Funktionstests

- ABS, ESP, ASR/MSR, Hill-Hold Control, ...

Funktionstests vernetzter Funktionen

- ESP-Lenkung (ESP-Anhängerstabilisierung mit Lenkungseingriff, DSR)
- Lenkung-Parklenkassistent

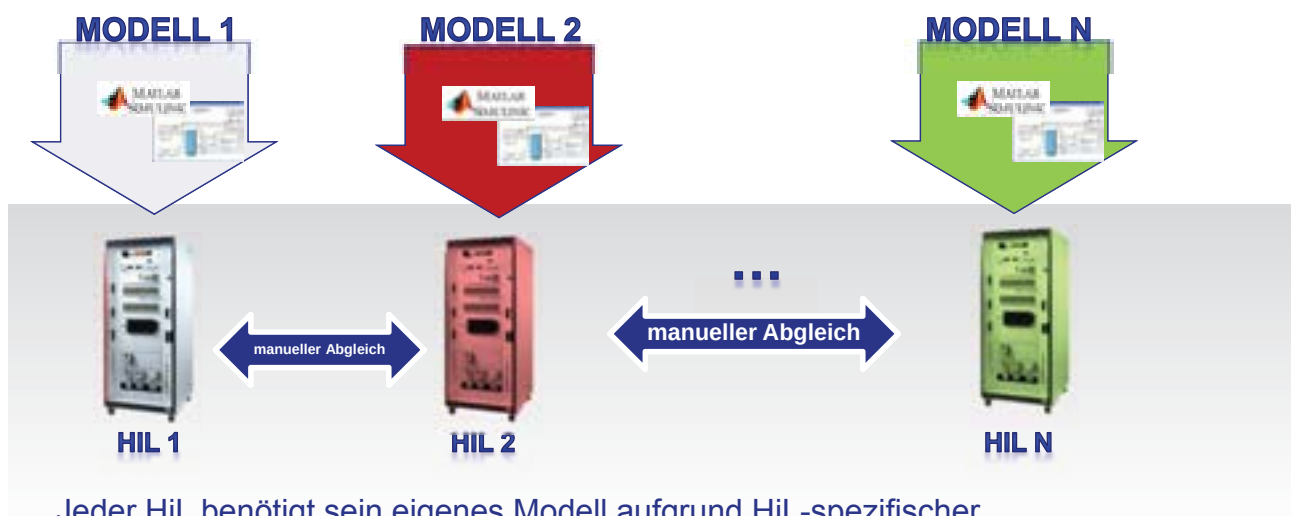
Applikation

- Vorapplikation von Steuergeräten (Wahl geeigneter Parameter für Parklenkassistent)

6

Aufbau eines Simulationsbaukastens

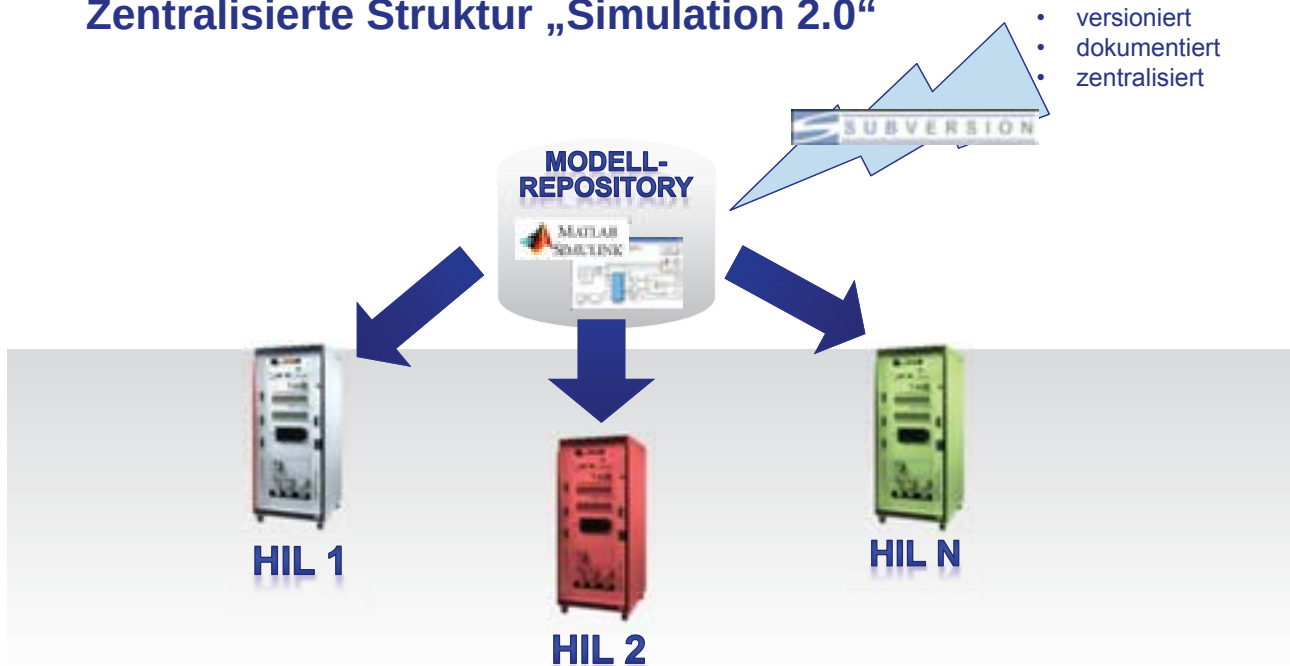
Organisation von HiL-Modellen – Parallele Struktur „Simulation 1.0“



Jeder HiL benötigt sein eigenes Modell aufgrund HiL-spezifischer Konfigurationen

Problem: Synchronität aller HiL-Modelle schwer zu gewährleisten

Organisation von HiL-Modellen – Zentralisierte Struktur „Simulation 2.0“

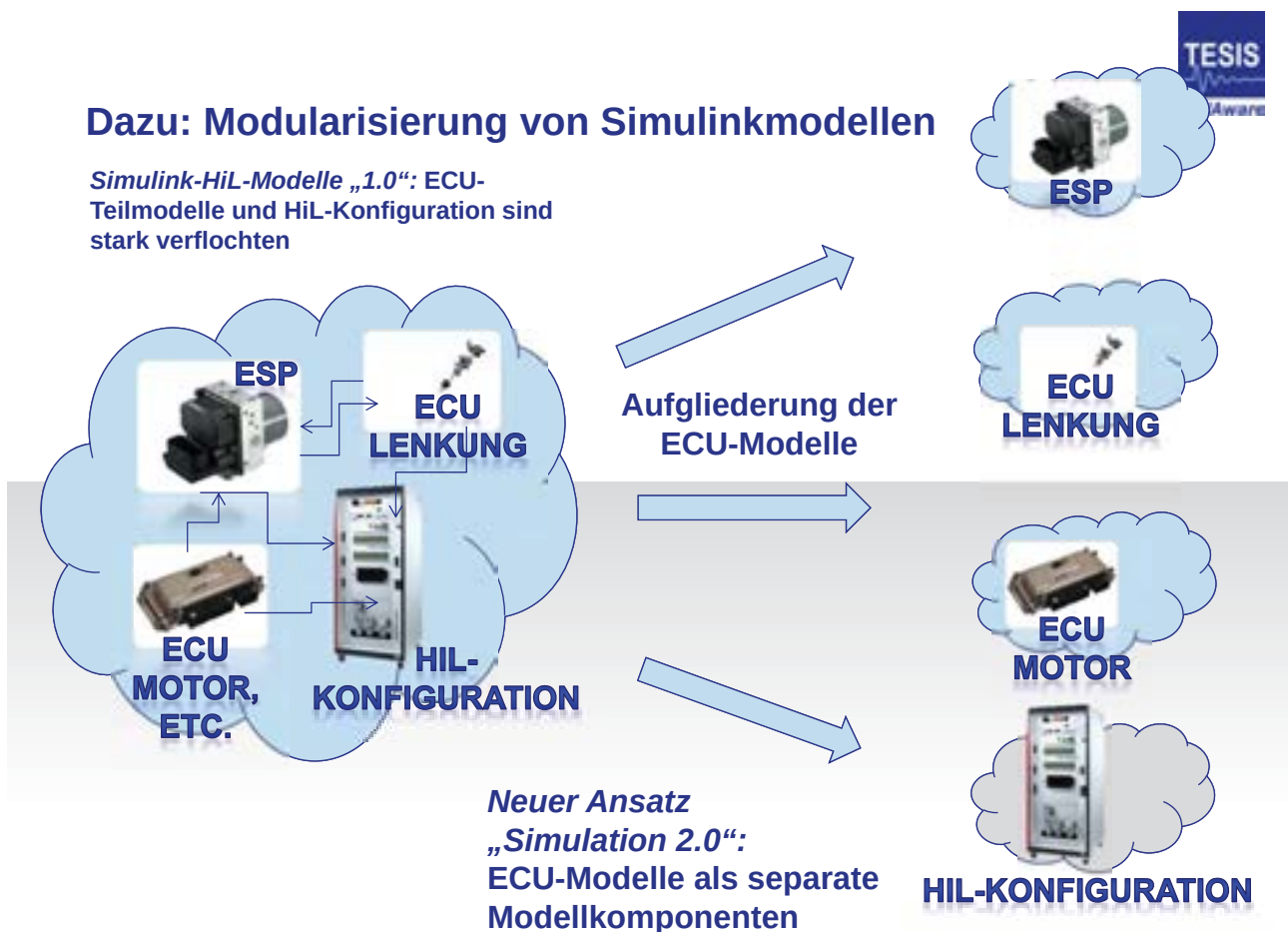


Verbesserung: Zentrale Datenhaltung und Modularisierung

→ Dafür sind Modellteile zerlegt, z.B. nach ECUs

Dazu: Modularisierung von Simulinkmodellen

Simulink-HiL-Modelle „1.0“: ECU-Teilmodelle und HiL-Konfiguration sind stark verflochten



„Baukastensystem“ für SiL/HiL-ECUs

Reale ECUs

SoftECUs

Motor

ESP

Lenkung

ECU-Modelle im „Baukasten“

- Für diverse Bauteile können Modelle von ECUs mit I/O und CAN-Blöcken sowie SoftECU-Modelle abgelegt werden
- jede ECU-Komponente ist **für sich lauffähig** und liegt im „Baukasten“ zur Nutzung bereit
- jede Real-ECU-Komponente ist **unabhängig von der HiL-Hardware-Konfiguration** verwendbar



„Baukastensystem“ für SiL/HiL-ECUs

Reale ECUs

SoftECUs

Motor

ESP

Lenkung

+
IO-Mapping
für HiL

Kombinationen

verschiedener

ECUs

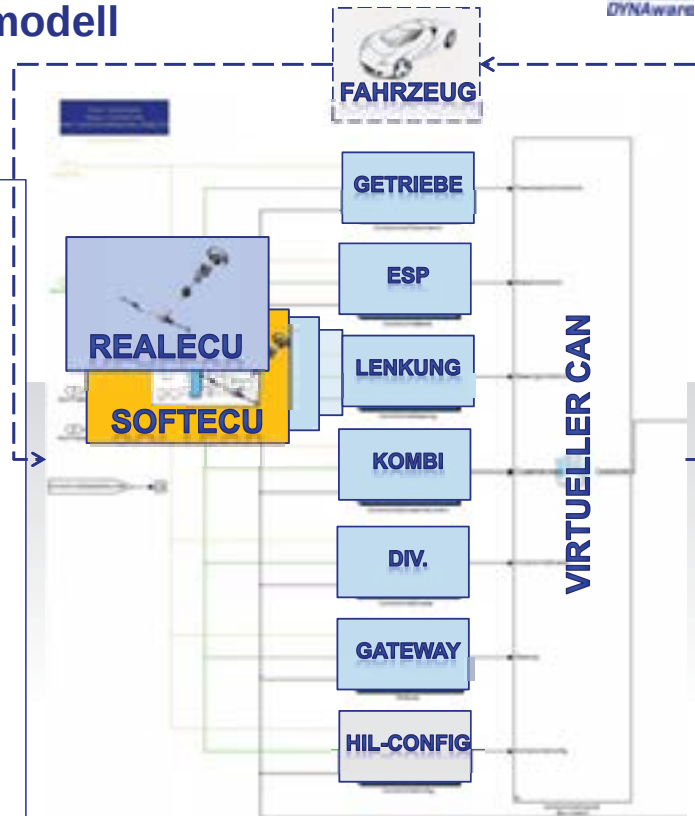
Gesamtmodell
für
HiL/SiL



Baukasten im Simulinkmodell

Unterteilung des Steuergerät-Subsystems nach Funktion:

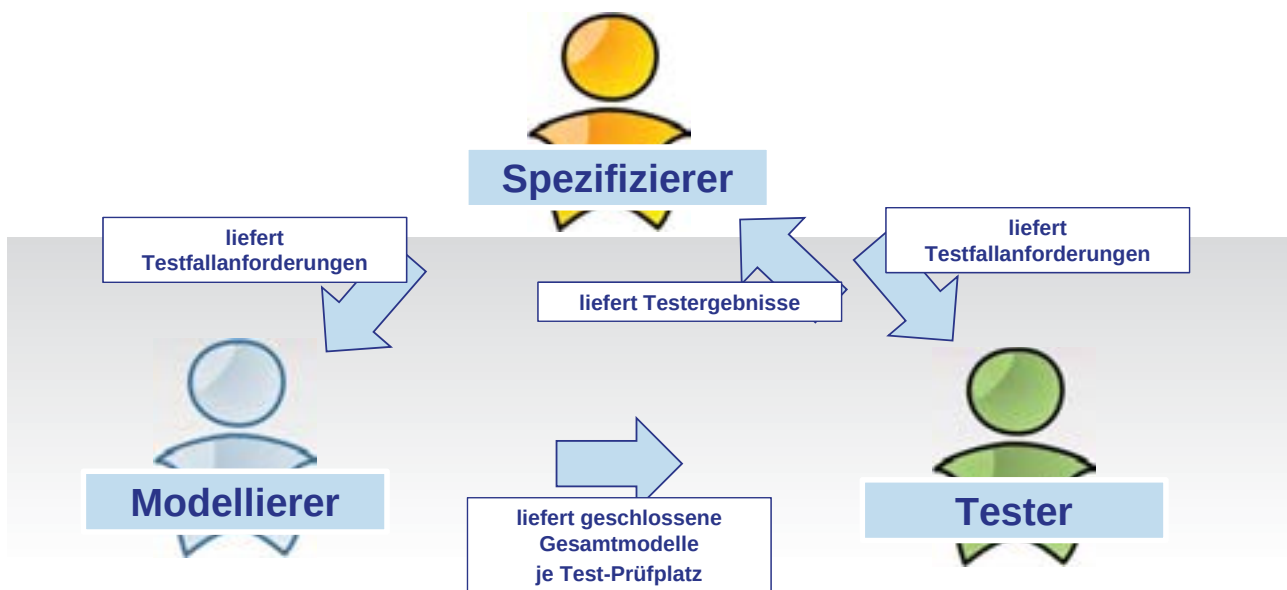
- Struktur und CAN-Kommunikation zwischen ECUs analog zum realen Fahrzeug
- Austausch der ECU-Komponenten z.B. mit realer ECU mit realer CAN-Signalerzeugung oder SoftECU
- ggf. Tausch der „HiL-Config“
- restliches Modell bleibt unberührt



Team-Workflow und Rollenkonzept

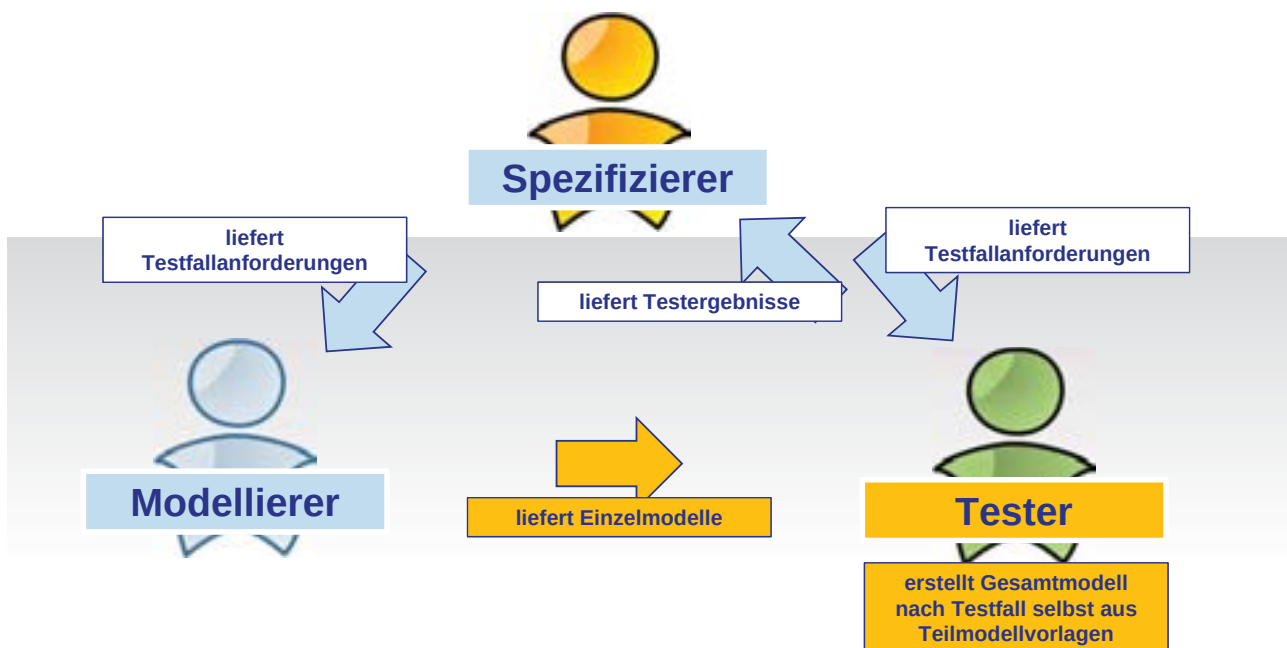


Rollenverteilung im Entwicklungsprozess in „Simulation 1.0“



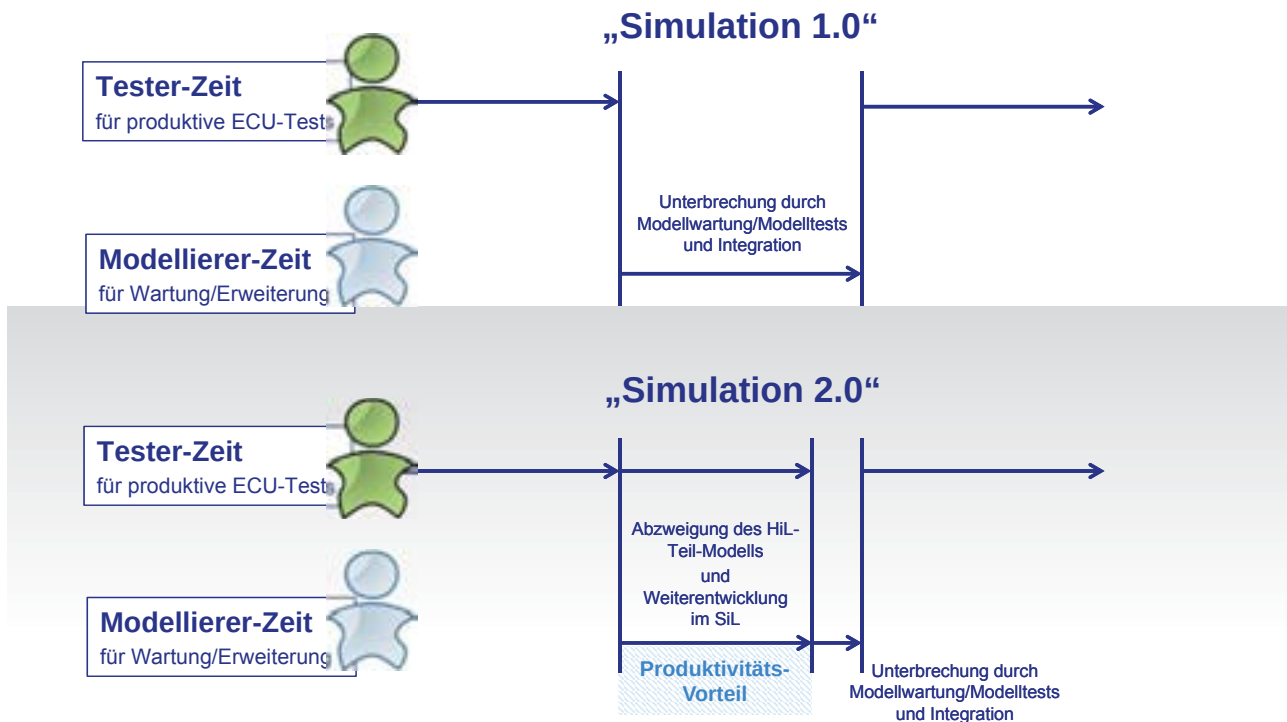
15

Rollenverteilung im Entwicklungsprozess in „Simulation 2.0“



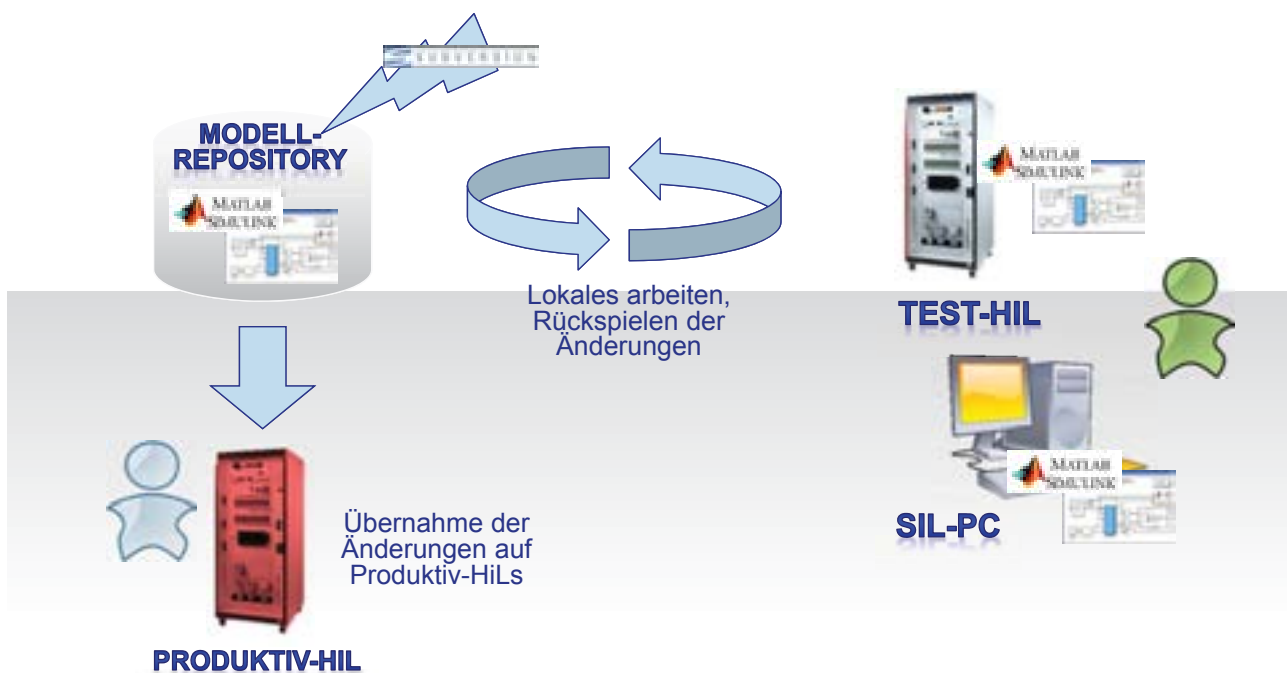
16

HiL-Produktivität während Wartungsarbeiten

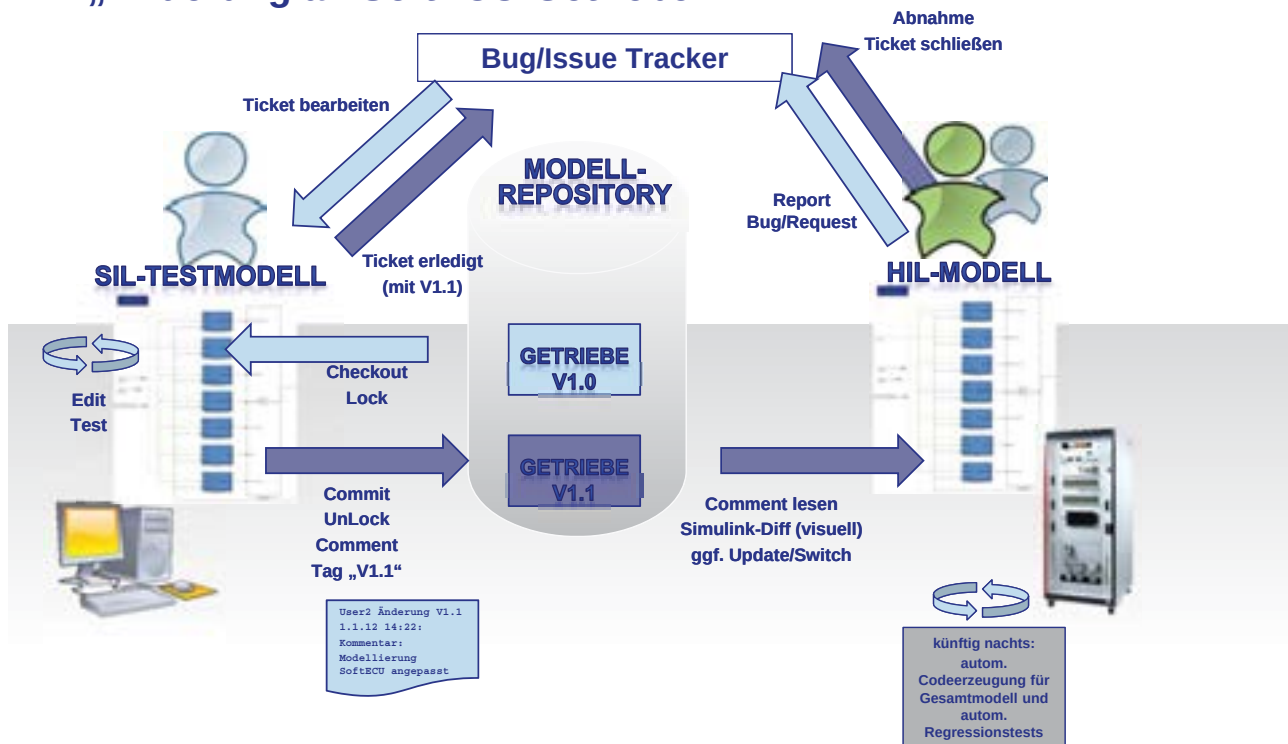


17

Verwaltung von Simulinkmodellen – Teamarbeit durch Modularisierung



Workflow-Beispiel: „Änderung an SoftECU-Getriebe“



Zusammenfassung und Ausblick

Zusammenfassung Features „Simulation 2.0“

- Zentralisierung & Modularisierung der Modelle und Daten
- Rollenkonzept
- „Softwareentwicklungsprozess“ für Simulationsumgebungen
- Abstraktion der Modelle von Hardware

21

Zusammenfassung Nutzen der „Simulation 2.0“

Verringerung der sichtbaren Modellkomplexität

- durch Abbildung der ECU-Topologien realer Fahrzeuge
- durch Kapselung von Funktionen

Verkürzung der Umbau- und Wartungszeit von HiL-Modellen

- durch paralleles Arbeiten
- durch Wiederverwendbarkeit von Komponenten

klare Aufgabenteilung

- Unterstützung des Rollenkonzepts durch eingesetzte Werkzeuge

Verbesserung des Testprozesses

- durch Testbarkeit der Testumgebung
- Dokumentation der Modelländerungen

22

Ausblick

Methoden der Softwareentwicklung für HiL/SiL-Testsysteme etablieren

- Issue/Bugtracking gekoppelt mit Versionierung
- Versionierung mit Rechteverwaltung
- Nightly Builds und autom. Regressionstests nach Modelländerungen zur Qualitätsverbesserung
- Vorkompilierte Modellteile verkürzen Modelbuild-Dauer
- Freigabe/Releases von HiL-Modellen zur besseren Traceability und Dokumentation

Abteilungsübergreifender Austausch von Modellteilen

- durch definierte Schnittstellen und zentrale Datenablage

Training method of oil and gas industry operators on the example of the automaton simulation usage

Eugeniy Gromakov, National Research Tomsk Polytechnic University

gromakov@tpu.ru

Maria Sotnikova, Hochschule Magdeburg-Stendal

mnsotnikova@vsibiri.info

Abstract

At the present time, in the oil and gas industries centers of operators training simulate situational problems like alarm and pre-alarm events. Recent studies in this field allow us to consider only the most likely emergency scenarios. In order to identify unlikely and untypical situations the HAZOP-methodology is being used. Its usage is becoming problematic due to the significant growth of low-probability "a fault tree leaves." In this paper we propose to use the automaton approach with the use of advanced automaton cuts for the effective use of failures models in practice, training of operators working in emergency situations. This process simulation approach will allow taking into account highly unlikely and untypical situations.

1 Introduction

Accidents at various facilities in the oil and gas sector can be divided into three groups of interrelated causes that contribute to the emergence of the accident, as analyses have shown:

- equipment failures,
- human error,
- underexpanded external effects of natural and manmade.

Through training and recurrent training of operators, we can reduce the severity of the accidents.

At the present time, in industrial centers of operators training situational problems like typical alarm and pre-alarm events are considered.

2 Formulation of the problem

According to the Pareto analysis only 20% of all incidents are responsible for 80% of damages.

As situational problems 20% of the best reasons of accident scenarios are detected and considered. In this case consideration of these situational problems reduces the likelihood of emergency events up to 80%. Thus it turns out that 80% of the scenarios fall out of consideration in the virtual training places, and operators are not prepared for 80% of the

unlikely system failures causes. However, international experience indicates greater risk of unlikely and untypical situations which operators are not prepared to control.

Emergency situation in the oil industry can paralyze the work of the oil and gas facilities until the elimination of the causes, and if unsuccessful can result in fire, possible detonation, and others. That may lead to the death of more than one dozen people.

Recent studies of the fault tree construction allow us to consider only the most likely emergency scenarios. That makes it necessary to develop the methods to detect more untypical events. It will allow operators to be ready to manage a lot of emergency events. Thereby it will significantly improve the quality of their training.

In Fig. 1. there is the likelihood dependence of emergency situations, depending on the number of possible causes of the accident scenarios.

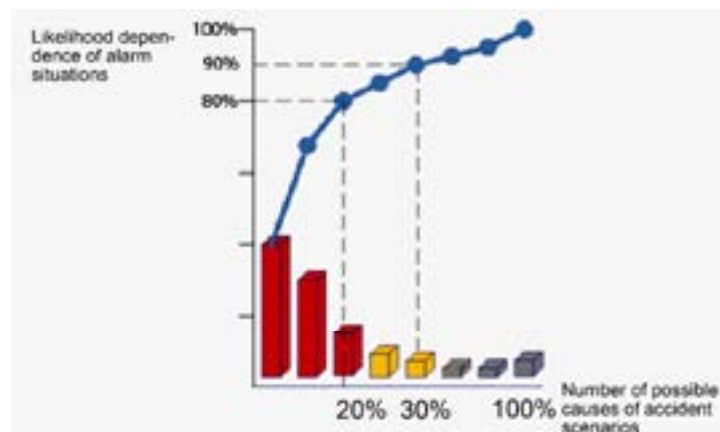


Figure 1: Likelihood of alarm situation, depending on the number of possible causes of the accident scenarios

3 The method of finite automaton as a tool for studying of a typical emergency scenarios

3.1 Disadvantages of the HAZOP-methodology used to identify emergency causes

In order to identify emergency causes at the present time the HAZOP-methodology is used. [1]. But the HAZOP-methodology considers only the most dangerous and most likely emergency situations. Its usage is becoming problematic to identify unlikely and untypical situations due to the significant growth of low-probability "a fault tree leaves." In addition, the HAZOP-methodology has a number of other shortcomings. In the process of the object analyzing on the availability of emergency situations consideration of relationships, regularities of accident scenarios, co-occurrence of multiple emergency scenarios are not included. [2].

3.2 Usage of the extended automaton cuts

In the practice of operators training of working in emergency situations automaton approach is proposed using "cuts" of "the extended machine". This approach will reduce costs for expensive prevention of emergency situations. Such an automaton approach allows creating virtual field automated place. That allows regularly training of operators for emergency scenarios and also gives intelligence prompts information of appropriate action. This approach

has considerable practical importance. Enterprises can get the simulation methods of technological processes to the workplace of the operators that will allow taking into account not unlikely typical situations. This account will allow a more carefully prepare operators of the oil and gas industry. This in turn will significantly reduce the probability of accidents and prevent them. That will reduce the costs of emergency response and resulting consequences.

It becomes cost effective to prepare the operators to the emergence of a larger number of alarm events, than to technically realize the lack of all contingencies.

StateFlow is used as software package for automated device modeling. StateFlow is a simulation tool for event-driven systems, for designing of complex control systems. StateFlow allows to simulate the behavior of complex event-driven systems, based on the theory of finite automats. The StateFlow diagram is a graphical representation of a finite automaton, where states and transitions form the basic building blocks of the system. Fig. 2 shows the finite state machine model of technological process section – the simulator of accidents processing. This model consists of a console, the control part (controller), the physical model of the reservoir and visualization tools displaying values of interest variables.

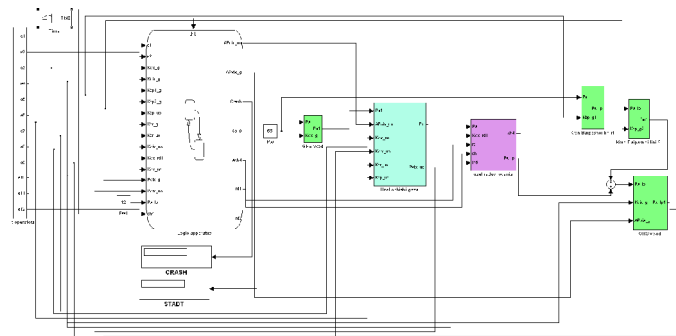


Figure 2: Finite state machine model of technological process section

Fig. 3 shows state model of the technological process.



Figure 3: StateFlow-state model of the technological process

Here is a hierarchical method of model building for automaton programs of simulator's control system. This method is used in modern enterprises.

In order to realize the most danger in terms of safety and security scenarios components and contained within them control objects are considered as the subsystems. Considered control objects are divided into several sub-systems of finite automaton. Each subsystem can be considered as finite state machines of the Moore-Mealy, arranged in a hierarchy. As a result, the logical part of the reservoir control system is given as a system of interacting automats. An automaton located higher in the hierarchy controls its imbedded automats by events generating

and transferring to them its control of the event. In addition, the automaton monitors the nested automatons, since its state transitions may depend on them also.

In Fig. 4. There is a diagram of the interaction between machines.



Figure 4: Diagram of machines interaction

4 Interface of simulation models

HMI LabVIEW is used as software package to create screen forms of the oil and gas facilities. It is the most suitable software to simulate the process of any physical nature. It has familiar human-machine interface to the operators of oil and gas facilities. LabVIEW has a wide range of built-in functions that allow to interact the virtual control object with Simulink-diagram.

5 Conclusion

At the present stage of the oil and gas operators' training, carried out in training centers, only typical emergency situations are considered. The importance of playing out such scenarios is significant, but we thus disregard most of the causes of other accident scenarios. The usage of "advanced finite automaton" will help us to consider significantly more events. We will limit the amount of the low-probability "leaves" of the "fault tree" by the "cuts" usage of "extended automata".

Field automated virtual places will not only generate motor-reflex skills of action in dangerous situations, but also will demonstrate to the operators the processes taking place in the equipment during the technological processes and their interdependence.

References

- [1] IEC 61882:2001: Hazard and operability studies (HAZOP studies) – Application guide, British standard, 2003
- [2] AER-5437 TSI 41-018: Guidelines for conducting HAZOP studies, SAUDI ARAMCO, loss prevention department.

Identifikation und Simulation des Nachgiebigkeitsverhaltens eines Fräsroboters

Oliver Rösch, Institut für Werkzeugmaschinen und Betriebswissenschaften
oliver.roesch@iwb.tum.de

Michael F. Zäh, Institut für Werkzeugmaschinen und
Betriebswissenschaften
michael.zaeh@iwb.tum.de

Kurzfassung

In diesem Beitrag wird ein Verfahren vorgestellt, mit dessen Hilfe die Steifigkeitsparameter eines Industrieroboters durch Messungen mit einem 3D-Scanning-Laser-Doppler-Vibrometer identifiziert werden können. Dabei werden sowohl die Kippsteifigkeiten der Getriebe als auch der Lager berechnet. Die Verarbeitung dieser Parameter in einem recheneffizienten Simulationsmodell ermöglicht die Bestimmung des Nachgiebigkeitsverhaltens des Roboters in Echtzeit. Um auch die aktuelle Achsstellung des Roboters zu berücksichtigen, wovon die Hebelverhältnisse und somit auch die Steifigkeit am Tool-Center-Point (TCP) abhängig sind, wird das Modell mit der Robotersteuerung gekoppelt. Somit wird es möglich, kraftbedingte Verlagerungen des Roboters, wie sie beispielsweise beim Fräsen auftreten, steuerungstechnisch zu kompensieren, sofern die am Tool-Center-Point (TCP) wirkenden Kräfte bekannt sind.

1 Einleitung

Für die spanende Bearbeitung werden derzeit fast ausschließlich Werkzeugmaschinen eingesetzt. 6-Achs-Knickarmroboter bieten durch ihren großen Arbeitsraum, die hohe Flexibilität und die vergleichsweise geringen Investitionskosten gegenüber Fräsmaschinen und Bearbeitungszentren großes Einsparpotenzial. Trotz dieser Vorteile werden Fräsroboter aufgrund der hohen Nachgiebigkeit der Roboterstruktur lediglich bei Bearbeitungsaufgaben mit geringen Genauigkeitsanforderungen und niedrigen Zerspankräften eingesetzt. Die hohe Nachgiebigkeit führt dazu, dass der Roboter bei der Zerspanung aufgrund der wirkenden Kräfte stark von seiner Sollbahn abgedrängt wird. Diese Abdrängung wird von den Sensorsystemen des Roboters nicht erfasst und kann deshalb nicht von der Steuerung korrigiert werden. Um die erzielbare Genauigkeit bei der Fräsbearbeitung mit Industrierobotern zu erhöhen, wird am Institut für Werkzeugmaschinen und Betriebswissenschaften der TU München (*iwb*) ein modellbasiertes Regelungssystem zur Kompensation der kraftbedingten statischen Verlagerungen entwickelt. Während des

Bearbeitungsprozesses werden die wirkenden Zerspankräfte gemessen und der resultierende Verlagerungsvektor mit Hilfe eines Steifigkeitsmodells berechnet. Dabei ist der Regler mit der Steuerung des Roboters verbunden, wodurch die Möglichkeit besteht, die aktuellen Achswinkel abzufragen und die programmierte Bahn durch Offset-Signale zu beeinflussen. Die Güte, mit der auf diese Weise der Bearbeitungsfehler reduziert werden kann, wird im Wesentlichen von der Qualität des verwendeten Simulationsmodells bestimmt. Die Herausforderung beim Aufbau dieses Modells besteht neben der zwingenden Echtzeitfähigkeit in der präzisen Bestimmung der Nachgiebigkeitsparameter des Roboters. Dabei sollen die Kippsteifigkeiten der Getriebe und der Lager berücksichtigt werden.

2 Stand der Wissenschaft und Technik

2.1 Modellierung des Strukturverhaltens von Robotersystemen

Um das statische Verhalten eines 6-Achs-Industrieroboters zu beschreiben, wird die Roboterstruktur häufig in Form eines Mehrkörpersystems (MKS) modelliert, wobei verschiedene Detaillierungsstufen möglich sind. Die einfachste Stufe sieht vor, die Strukturkomponenten als ideal steif zu betrachten und lediglich die Rotationssteifigkeit der Getriebe als lineare Kennlinie zu integrieren [1]. Eine komplexere Möglichkeit findet in [2] und [3] Verwendung. Hier wird die Elastizität des Antriebsstranges durch eine experimentell ermittelte nichtlineare Kennlinie dargestellt, wobei auch die Übersetzungsfehler durch Ungenauigkeiten in der Fertigung berücksichtigt werden. Der nächste Schritt in der Modellbildung sieht neben der Darstellung von nichtlinearen Steifigkeiten auch die Modellierung der Hysterese zwischen dem wirkenden Getriebemoment und der sich einstellenden Verdrehung vor [4]. In [5] wird nicht nur die Elastizität der Getriebe, sondern auch die Verkippung der Achslager berücksichtigt. Diese Drehfedern werden als virtuelle Gelenke bezeichnet, da sie dem modellierten Rotationsgelenk zwei zusätzliche Freiheitsgrade verleihen. Soll neben der Nachgiebigkeit der Gelenke auch die Elastizität der Strukturkomponenten berücksichtigt werden, bietet sich der Einsatz eines flexiblen Mehrkörpersystems an. Hierbei können einzelne Bauteile nach der Methode der Finiten Elemente vernetzt und in das Simulationsmodell integriert werden. Um die Recheneffizienz solcher Modelle zu steigern, kann die Anzahl der Freiheitsgrade der flexiblen Komponenten durch modale Reduktion [6] verringert werden [7].

Bei der Modellierung des Strukturverhaltens eines Roboters mit Hilfe von Mehrkörpersystemen besteht neben der Darstellung eines stationären Zustandes auch die Möglichkeit, Verfahrbewegungen der einzelnen Achsen abzubilden. Dies kann in kommerziellen MKS-Programmen wie ADAMS[®] oder RecurDyn[®] durch die Modellierung eines ideal steifen Rotationsgelenks erfolgen, auf dem die eigentliche Drehfeder des Robotergelenkes angreift [8, 9]. Somit kann das Verhalten des Roboters auch während Verfahrbewegungen mit hoher Genauigkeit prognostiziert werden. Der Nachteil bei einer derart detaillierten Modellierung liegt, auch bei der Verwendung modal reduzierter Körper, in

dem hohen Rechenaufwand, der bei der Simulation entsteht. Selbst beim Einsatz von leistungstarken Rechnern¹ ist eine echtzeitfähige Simulation unmöglich.

2.2 Identifikation der Steifigkeitsparameter der Gelenke

Neben der zuvor beschriebenen Art der Modellierung ist die präzise Bestimmung der erforderlichen Steifigkeitsparameter maßgeblich für die Validität des Simulationsmodells. Es existiert eine Reihe von Ansätzen, um die Steifigkeiten der Getriebe, der Lager und der Strukturkomponenten von 6-Achs-Knickarmrobotern zu identifizieren.

Ein direktes Verfahren zur Messung der Getriebesteifigkeit kommt in [10] zum Einsatz. Hierbei wird vorausgesetzt, dass jeder einzelne Antrieb abmontiert und das Getriebe ausgebaut wird. Dieses wird an einen Prüfstand angeschlossen und über einen Schwingungserreger belastet, wobei der Verdrehwinkel über einen Encoder gemessen wird. Auf diese Weise können durch eine Reihe von statischen und dynamischen Versuchen neben den Steifigkeiten auch die Reibungs- und Dämpfungsparameter des Getriebes ermittelt werden.

Die in [2] und [11] verwendete Methode zur Ermittlung der Gelenksteifigkeiten basiert auf einer Reihe von statischen Zugversuchen am Roboter. Der statische Zustand wird durch das Anziehen der in allen Antrieben integrierten Bremsen realisiert. Zur Messung der Getriebesteifigkeit der n -ten Achse wird im Versuch eine definierte Kraft am TCP aufgebracht, wodurch diese Achse durch ein Moment belastet wird. Die resultierende Verlagerung wird am TCP beispielsweise über ein Kamerasystem oder Messuhren bestimmt. Daraus werden der Verdrehwinkel der Achse und die Steifigkeitskennlinie des Getriebes berechnet. Nachteilig wirken sich bei dieser Messmethode der Einfluss der Haftreibung sowie die Elastizität der übrigen im Kraftfluss liegenden Gelenke aus, wodurch die Ergebnisgüte reduziert wird.

Ein weiterer Ansatz zur Bestimmung der Getriebesteifigkeiten wird in [12] vorgestellt. Hierbei wird der Roboter zur Vermessung der n -ten Achse in eine Pose (Position und Orientierung) gefahren, die es erlaubt, die $(n + 1)$ -te Achse extern abzustützen. Auf diese Weise werden die Einflüsse aus der Nachgiebigkeit der übrigen Gelenke abschnittsweise ausgeschlossen, so dass das Verhalten der Achse bei der Auswertung als Einmassenschwinger behandelt werden kann. Bei der Messung erfolgt eine harmonische Anregung durch den Motor der n -ten Achse, wobei das Achsmoment mit Hilfe eines Kraftsensors in der Abstütz-Vorrichtung bestimmt wird. Durch die Identifikation der Eigenkreisfrequenz ω ergibt sich die Steifigkeit C zu

$$C = \omega^2 * M. \quad (1)$$

Die bislang vorgestellten Ansätze beschränken sich auf die Identifikation der Nachgiebigkeiten der Robotergetriebe. Das Verhalten der Strukturkomponenten bzw. der Achslagerungen wird meist simulativ durch Parametervariation bestimmt. Mit Hilfe der so genannten Grey-Box-Identifikation können diese Parameter auf Basis von Messdaten

¹ 2 x Intel® Xeon® CPU X5550, 48 GB RAM

berechnet werden [4]. Dabei werden stoßartige Bewegungen mit dem Roboter durchgeführt, die entweder durch ruckartige Verfahrbewegungen oder Hammerschläge realisiert werden. Das resultierende Ausschwingverhalten kann mittels Beschleunigungssensoren, Drehratensensoren, einer 6D-Kamera oder der achsinternen Sensorik (für Motorstrom, -winkel, und -temperatur) erfasst werden. Die gesuchten Parameter werden in einem rechenintensiven² Optimierungsalgorithmus nach dem Prinzip der Multi-Objective Parameter Synthesis (MOPS) [4] bestimmt. Ein Nachteil dieser Methode liegt darin, dass die mathematisch verschiedenen Parameterkonfigurationen aus den Messwerten ableitbar sind. Es können somit bei der Optimierungsrechnung schwer erkennbare Fehler auftreten, welche die Ergebnisgüte negativ beeinflussen.

Die vorgestellten Methoden zur Bestimmung der Steifigkeitsparameter eines Roboters unterscheiden sich in der maximal erreichbaren Ergebnisgüte, der Anzahl der identifizierbaren Parameter und in dem erforderlichen Mess- und Berechnungsaufwand. Es existiert bislang kein Verfahren, das es erlaubt, mit geringem messtechnischen Aufwand das statische Nachgiebigkeitsverhalten der Getriebe, der Lager und der Strukturkomponenten mit hoher Genauigkeit zu identifizieren. Um dies zu erreichen, wird im Folgenden eine Methode vorgestellt, mit deren Hilfe die Verlagerungen einzelner Punkte der Roboterstruktur hochgenau bestimmt und daraus die gesuchten Steifigkeitsparameter berechnet werden können. Hierbei kommt ein 3D-Scanning-Laser-Doppler-Vibrometer zum Einsatz.

3 Identifikation der Steifigkeitsparameter des Roboters

3.1 Messaufbau

Die Messungen werden an einem Industrieroboter vom Typ KR 240 R2500 prime der KUKA Roboter GmbH durchgeführt. Am Flansch des Roboters ist eine Frässpindel der HSD Deutschland GmbH montiert, an deren Gehäuse über einen doppelt wirkenden Pneumatik-Zylinder eine Kraft im Bereich von mehreren Kilonewton aufgebracht werden kann (Abbildung 1, links). Der Belastungszyklus sieht sowohl eine Zug- als auch eine Druck-Beanspruchung des Roboters vor, wobei die jeweilige Kraft langsam aufgebaut und vor der Entlastung einige Sekunden gehalten wird. Auf diese Weise besteht während der Belastungsphasen ein quasistatisches Gleichgewicht zwischen der Verlagerung des Roboters und der wirkenden Last. Um die Signalqualität des eingesetzten optischen Messsystems zu verbessern, werden sämtliche Messpunkte mit retroreflektierender Folie beklebt. Zudem wird hinter dem Roboter ein Spiegel aufgestellt, um auch Punkte auf der abgewandten Seite erfassen zu können.

² In [4] wird ein Rechencluster mit 30 CPUs eingesetzt.

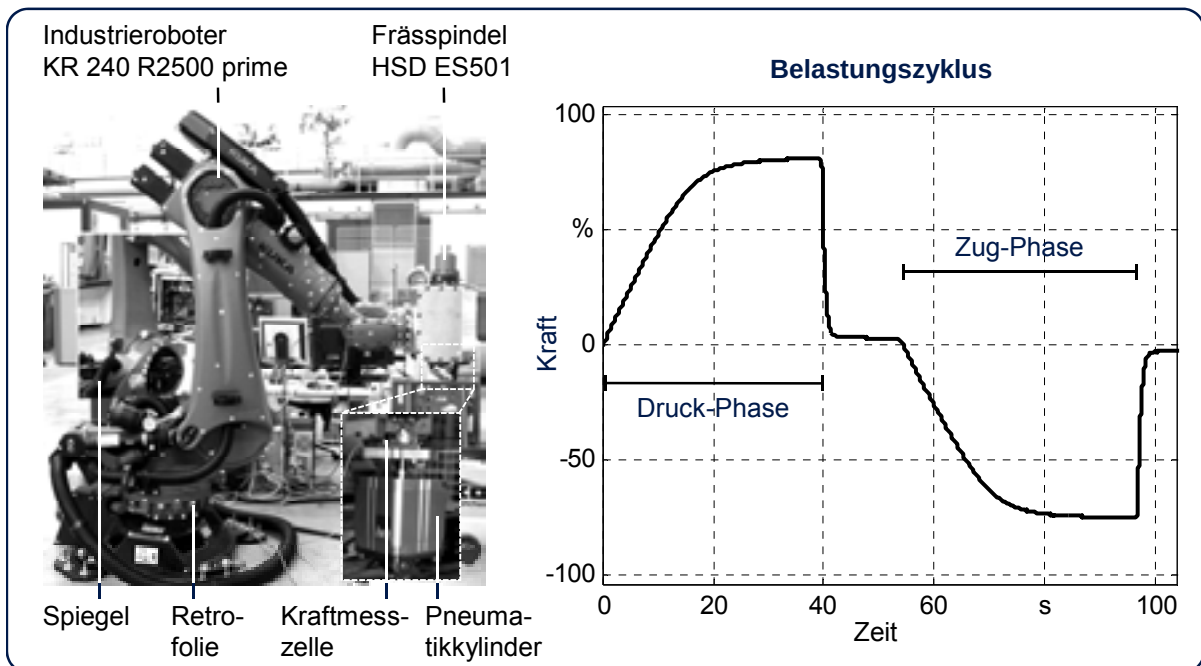


Abbildung 1: Messaufbau (links) und qualitativer Verlauf der wirkenden Kraft (rechts)

3.2 Aufbau des 3D-Scanning-Laser-Doppler-Vibrometers

Das verwendete 3D-Scanning-Laser-Doppler-Vibrometer (LDV) der Firma Polytec besteht aus drei Scanköpfen, die an einem Stativ angebracht sind (Abbildung 2, links). Jeder Scankopf verfügt über ein Laser-Interferometer, eine Videokamera sowie eine Scanner-Einheit, wodurch der Laserstrahl horizontal und vertikal um $\pm 20^\circ$ ausgelenkt werden kann. Der obere Messkopf enthält zusätzlich eine Geometriescan-Einheit zur Entfernungsmessung. Nach der Justage des Systems, bei der das Messkoordinatensystem festgelegt und die Position der Scanköpfe zueinander berechnet werden, ist es möglich, die zu messenden Punkte direkt auf dem Videobild des Messobjekts zu definieren. Über die Geometriescan-Einheit wird im Anschluss die Distanz der Messpunkte zum oberen Scankopf gemessen, so dass ein dreidimensionales Messmodell entsteht. Abbildung 2 (rechts) zeigt ein Messmodell des untersuchten Roboters, welches aus 76 Punkten besteht.

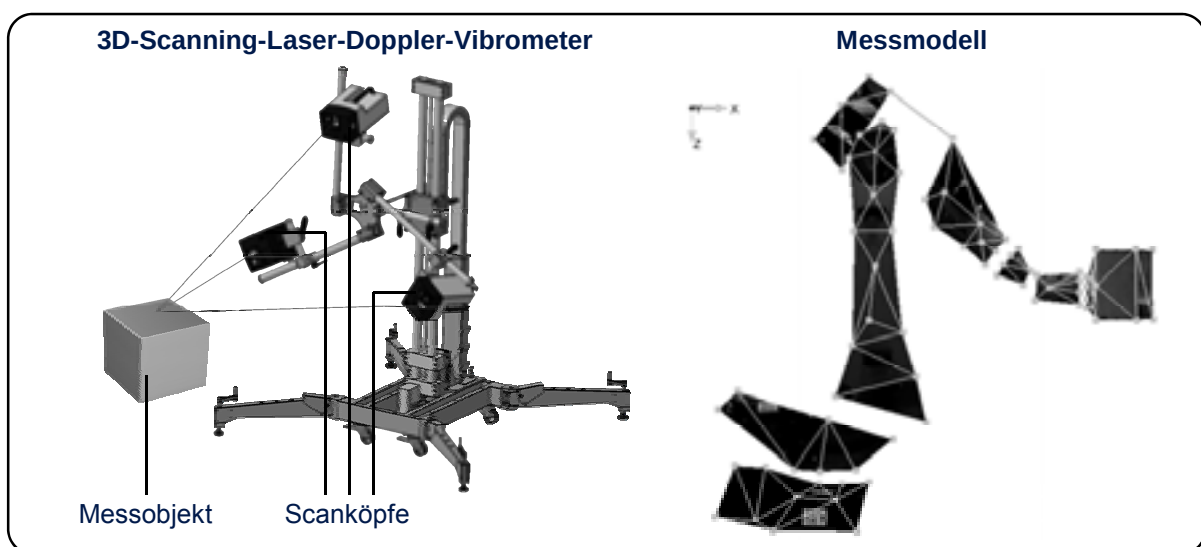


Abbildung 2: Schematischer Aufbau des LDV (links) und Messmodell des Roboters (rechts)

Bei der Messung werden die Laserstrahlen aller Scanköpfe auf einen Punkt gelenkt und die Schwinggeschwindigkeit entlang der drei Laserstrahlen auf Basis des Prinzips der optischen Interferenz bestimmt. Durch die Kenntnis der aktuellen Stellungen der Scannerspiegel und der Positionen der Messköpfe im Raum kann durch trigonometrische Transformation der Rohdaten die dreidimensionale Bewegung des aktuellen Punktes berechnet werden. Auf diese Weise wird für jeden Messpunkt der Geschwindigkeitsvektor während eines Belastungszyklus des Roboters aufgezeichnet. Durch eine numerische Integration der Geschwindigkeitsdaten wird die Verlagerung aller Messpunkte bestimmt.

3.3 Berechnung der Steifigkeitsparameter aus den Messergebnissen

Abbildung 3 zeigt die resultierenden Verlagerungsbilder einer Messung, bei welcher der Roboter ausschließlich in Richtung der z-Achse belastet wurde.

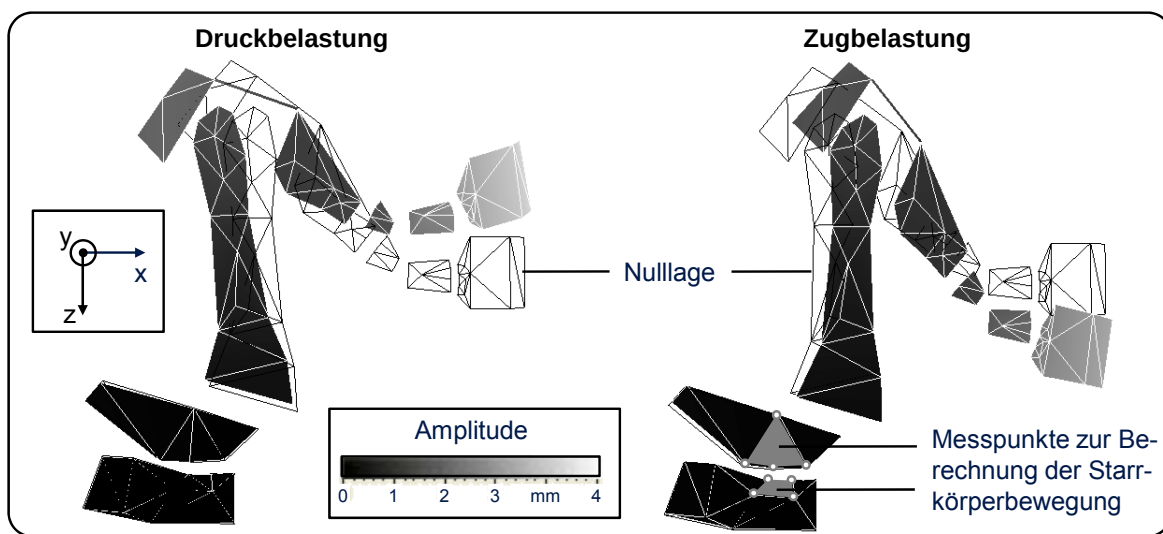


Abbildung 3: Auslenkung des Roboters bei Zug- und Druckbelastung

Es ist eine Verkipfung um die Getriebeachse des zweiten und dritten Gelenks erkennbar. Zudem zeigt sich eine Verdrehung des ersten Gelenkes um die y-Achse, welche sich aufgrund der Hebelverhältnisse deutlich auf die Verlagerung des TCP auswirkt. Dieses Beispiel verdeutlicht die Notwendigkeit, auch die Lagersteifigkeit der einzelnen Achsen bei der Modellierung des Nachgiebigkeitsverhaltens des Roboters zu berücksichtigen.

Um die Steifigkeitsparameter eines Gelenks bestimmen zu können, muss aus den Messdaten die Verdrehung der angrenzenden Strukturkomponenten bestimmt werden. Da für jeden Messpunkt nur translatorische Informationen vorliegen, müssen hierfür mehrerer Messpunkte gemeinsam analysiert werden. In Abbildung 3 (rechts unten) ist dies anhand des ersten Gelenks dargestellt. Die relative Verdrehung zwischen der Roboter-Basis und dem Karussell wird bestimmt, indem die Starrkörperbewegung von jeweils vier Messpunkten auf diesen Komponenten wie folgt berechnet wird.

Die Drehung eines Starrkörpers, welcher durch vier Messpunkte M1 bis M4 beschrieben wird, kann durch die Bewegung eines körperfesten Koordinatensystems B abgebildet werden.

Dieses Koordinatensystem besitzt einen Ursprung B_0 , sowie drei orthonormierte³ Basisvektoren $\mathbf{b}_x$, $\mathbf{b}_y$ und $\mathbf{b}_z$, die in den Punkten B_x , B_y und B_z enden (Abbildung 4, links).

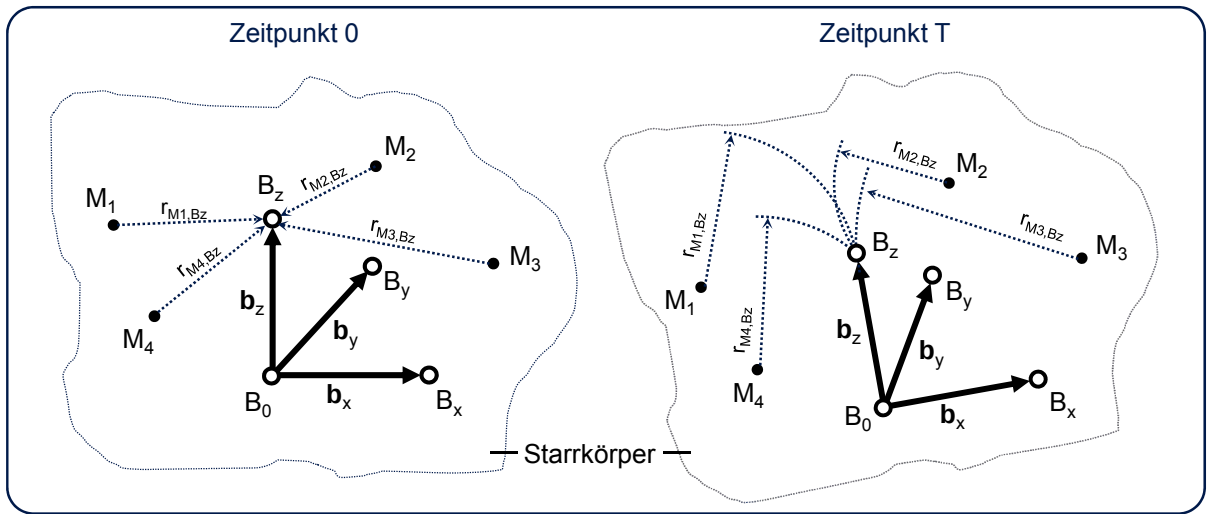


Abbildung 4: Starrkörperdrehung zum Zeitpunkt 0 (links) und T (rechts)

Zum Zeitpunkt 0 kann für jeden der Messpunkte der Abstand r zu den Punkten B_0 , B_x , B_y und B_z bestimmt werden. In Abbildung 4 sind aus Gründen der Übersichtlichkeit nur die Abstände zum Punkt B_z eingezeichnet. Diese Abstände werden sich bei einer ideal erfüllten Starrkörperbedingung nie ändern, wodurch die Position von B_z zu jedem beliebigem Zeitpunkt rekonstruiert werden kann. Geometrisch interpretiert ergibt sich B_z zum Zeitpunkt T durch den Schnittpunkt von vier Kugeln, die um die Punkte M_1 bis M_4 mit den zuvor bestimmten Radien gezogen werden (Abbildung 4, rechts). Für alle Punkte auf der Oberfläche der Kugel um den Punkt M_1 mit dem Radius $r_{M1,Bz}$ gilt

$$(x - M_{1x})^2 + (y - M_{1y})^2 + (z - M_{1z})^2 - r_{M1,Bz}^2 = 0. \quad (2)$$

Analoge Gleichungen lassen sich auch für die Kugeln um die Messpunkte M_2 , M_3 und M_4 aufstellen. Es muss im Folgenden ein Punkt identifiziert werden, dessen x -, y - und z -Koordinaten diese vier Gleichungen erfüllen. Um hieraus ein lineares Gleichungssystem zu bilden, werden die Gleichungen nach dem Kugelradius aufgelöst und paarweise voneinander subtrahiert.

³ Vektoren sind paarweise zueinander orthogonal und besitzen die Länge 1

$$\begin{aligned}
\text{I} \quad r_{M1,Bz}^2 - r_{M2,Bz}^2 &= 2(M_{2x} - M_{1x}) * x + 2(M_{2y} - M_{1y}) * y + 2(M_{2z} - M_{1z}) * z + \\
&\quad + M_{1x}^2 + M_{1y}^2 + M_{1z}^2 - M_{2x}^2 - M_{2y}^2 - M_{2z}^2 \\
\text{II} \quad r_{M2,Bz}^2 - r_{M3,Bz}^2 &= 2(M_{3x} - M_{2x}) * x + 2(M_{3y} - M_{2y}) * y + 2(M_{3z} - M_{2z}) * z + \\
&\quad + M_{2x}^2 + M_{2y}^2 + M_{2z}^2 - M_{3x}^2 - M_{3y}^2 - M_{3z}^2 \\
\text{III} \quad r_{M3,Bz}^2 - r_{M1,Bz}^2 &= 2(M_{1x} - M_{3x}) * x + 2(M_{1y} - M_{3y}) * y + 2(M_{1z} - M_{3z}) * z + \\
&\quad + M_{3x}^2 + M_{3y}^2 + M_{3z}^2 - M_{1x}^2 - M_{1y}^2 - M_{1z}^2 \\
\text{IV} \quad r_{M4,Bz}^2 - r_{M3,Bz}^2 &= 2(M_{3x} - M_{4x}) * x + 2(M_{3y} - M_{4y}) * y + 2(M_{3z} - M_{4z}) * z + \\
&\quad + M_{4x}^2 + M_{4y}^2 + M_{4z}^2 - M_{3x}^2 - M_{3y}^2 - M_{3z}^2
\end{aligned} \tag{3}$$

Dieses System lässt sich formell in die vektorielle Darstellung überführen.

$$\mathbf{A} * \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \mathbf{b} \tag{4}$$

In dieser Form beinhaltet der Vektor $\mathbf{b}$ mit den Dimensionen 4×1 alle in dem Gleichungssystem auftretenden additiven Konstanten. Die Koeffizientenmatrix $\mathbf{A}$ der Dimension 4×3 beinhaltet die multiplikativen Konstanten und x , y und z stellen die Koordinaten des gesuchten Schnittpunkts der Kugeln dar. Für das Lösen eines Gleichungssystems dieser Art steht in der Literatur eine Vielzahl von Algorithmen zur Verfügung. Um Singularitätsprobleme zu vermeiden, wie sie bei der numerischen Invertierung von Matrizen auftreten können, wird hier die LU-Zerlegung verwendet. Diese Zerlegung teilt die zu invertierende Matrix in drei Komponenten auf. Die Komponenten werden anstelle der eigentlichen Matrix zur Lösungsfindung verwendet und deren Teilergebnisse rekursiv miteinander verrechnet. Eine genaue Beschreibung dieses Verfahren ist in [13] zu finden.

Auf diese Weise lassen sich für jeden Zeitpunkt des Messdatensatzes die Koordinaten der Punkte B_0 , B_x , B_y und B_z bestimmen, wodurch auch die Basisvektoren $\mathbf{b}_x$, $\mathbf{b}_y$ und $\mathbf{b}_z$ (vgl. Abbildung 4) berechnet werden können. Mit Hilfe der Basisvektoren lässt sich die allgemeine Transformationsvorschrift für die Abbildung $(x_0, y_0, z_0) \rightarrow (x_1, y_1, z_1)$ aufstellen.

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} b_{xx} & b_{yx} & b_{zx} \\ b_{xy} & b_{yy} & b_{zy} \\ b_{xz} & b_{yz} & b_{zz} \end{bmatrix} * \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} \tag{5}$$

Diese allgemeine Rotation kann in drei Elementardrehungen um die x -, die y - und die z -Achse zerlegt werden. Eine solche Zerlegung ist nicht eindeutig, da sie von der Reihenfolge der einzelnen Rotationen abhängig ist. Um die Reproduzierbarkeit der gesamten Rotation zu gewährleisten, haben sich bei der Zerlegung verschiedene Konventionen etabliert. Im Bereich der Mehrkörpersysteme hat sich die Konvention der so genannten Kardanwinkel durchgesetzt. Die Rotationsmatrix $\mathbf{R}$ nach dieser Konvention lautet

$$R = \begin{bmatrix} c\gamma c\beta & c\gamma s\beta s\alpha - s\gamma c\alpha & c\gamma c\beta c\alpha + s\gamma s\alpha \\ s\gamma c\beta & s\gamma s\beta s\alpha + c\gamma c\alpha & -s\gamma c\beta \\ -s\beta & c\beta s\alpha & c\beta c\alpha \end{bmatrix}, \quad (6)$$

mit $c \triangleq \text{Kosinus}$
 $s \triangleq \text{Sinus}$,

wobei α , β und γ die Drehwinkel um die x-, die y- bzw. die z-Achse bilden. Durch den elementweisen Vergleich der Gesamtrationsmatrix R und der Transformationsmatrix, die aus den Basisvektoren $\mathbf{b}_x$, $\mathbf{b}_y$ und $\mathbf{b}_z$ aufgestellt wurde (Gleichung 5) können neun Gleichungen aufgestellt werden, um die drei gesuchten Winkel zu berechnen. Hieraus werden drei Gleichungen ausgewählt und die Winkel beispielsweise wie folgt bestimmt:

$$\begin{aligned} \text{I} \quad & b_{xz} = -\sin \beta \quad \rightarrow \quad \beta = -\sin^{-1}(b_{xz}) \\ \text{II} \quad & b_{yz} = \cos \beta \sin \alpha \quad \rightarrow \quad \alpha = \sin^{-1}\left(\frac{b_{yz}}{\cos \beta}\right) \\ \text{III} \quad & b_{xy} = \sin \gamma \cos \beta \quad \rightarrow \quad \gamma = \sin^{-1}\left(\frac{b_{xy}}{\cos \beta}\right) \end{aligned} \quad (7)$$

Durch den in diesem Abschnitt vorgestellten Berechnungsalgorithmus können die Verdrehwinkel einzelner Strukturbereiche des Roboters quantifiziert werden. Im Beispiel aus Abbildung 3 kann die Verdrehung des Karussells gegenüber der Basis des Roboters aufgrund der wirkenden Kraft berechnet werden. Da der Kraftangriffspunkt und die -richtung bekannt sind, können unter Verwendung der Kardankonvention die resultierenden Momente (M_{x1} , M_{y1} und M_{z1}) um die drei Raumachsen bestimmt und die Steifigkeiten des Getriebes (C_{z1}) und des Achslagers (C_{x1} und C_{y1}) berechnet werden:

$$C_{x1} = \frac{M_{x1}}{\alpha} \quad C_{y1} = \frac{M_{y1}}{\beta} \quad C_{z1} = \frac{M_{z1}}{\gamma} \quad (8)$$

4 Implementierung eines echtzeitfähigen Simulationsmodells

Wie bereits erwähnt, muss das Simulationsmodell zur Bestimmung der kraftbedingten TCP-Verlagerung echtzeitfähig sein, um in einem modellbasierten Regler Einsatz zu finden. Die Kommunikation zwischen der Robotersteuerung und dem Regelungs-PC erfolgt mit einer Zykluszeit von 4 ms. Die Echtzeitbedingung ist also erfüllt, wenn die Rechenzeit des Modells unter dieser Zeitspanne liegt. Mit kommerziellen Mehrkörperprogrammen ist dies nicht erreichbar, so dass ein analytisches Modell zur Berechnung der Verlagerung am TCP aufgrund einer wirkenden Kraft entwickelt werden muss. Das analytische Modell basiert auf einer konventionellen Vorwärtskinematik, die zur Bestimmung der Pose des TCP eines Roboters verwendet wird. Hierbei wird auf jedem Robotergelenk ein Koordinatensystem definiert, wobei in der Regel die Z-Achse mit der jeweiligen Gelenk-Achse zusammenfällt (Denavit-Hartenberg-Konvention). Die Rotation θ einer Achse n wird durch eine

Transformationsmatrix T_n beschrieben, welche auch die (konstanten) Abmessungen p der Roboterarme berücksichtigt:

$$T_n = \begin{pmatrix} c\theta & -s\theta & 0 & p_{xn} \\ s\theta & c\theta & 0 & p_{yn} \\ 0 & 0 & 1 & p_{zn} \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (9)$$

Diese Beschreibung enthält als einzigen Freiheitsgrad den Gelenkwinkel θ der Achse und wird um zusätzliche Freiheitsgrade erweitert, um auch die Nachgiebigkeit des Getriebes und des Gelenklagers abzubilden. Die erweiterte Transformationsmatrix N_n ergibt sich zu

$$N_n = \begin{pmatrix} \mathbf{ROT}_n & \begin{matrix} p_{xn} \\ p_{yn} \\ p_{zn} \end{matrix} \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad (10)$$

wobei sich die Rotationsmatrix $\mathbf{ROT}_n$ aus dem Produkt der drei orthogonalen Einzelrotationen zusammensetzt:

$$\mathbf{ROT}_n = \underbrace{\begin{pmatrix} c(\theta + \gamma_n) & -s(\theta + \gamma_n) & 0 \\ s(\theta + \gamma_n) & c(\theta + \gamma_n) & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\text{Achswinkel und Getriebeverdr\u00f6hung}} \underbrace{\begin{pmatrix} c\beta_n & 0 & s\beta_n \\ 0 & 1 & 0 \\ -s\beta_n & 0 & c\beta_n \end{pmatrix}}_{\text{Lagerverdr\u00f6hung}} \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & c\alpha_n & -s\alpha_n \\ 0 & s\alpha_n & c\alpha_n \end{pmatrix}}_{\text{Lagerverdr\u00f6hung}} \quad (11)$$

Die Gr\u00f6\u00dfen α_n , β_n und γ_n beschreiben die Verdrehungen des Gelenks um das Achslager bzw. das Getriebe. Sie k\u00f6nnen mit Hilfe der oben identifizierten Steifigkeitsparameter durch die Aufl\u00f6sung der Gleichung 8 nach dem entsprechenden Winkel bestimmt werden. Die Momente um die jeweiligen Koordinatenachsen werden zuvor durch die Projektion der am TCP wirkenden Last gem\u00e4\u00df der Kardankonvention bestimmt. In Gleichung 10 wird also nicht nur die reine Drehung einer Roboterachse abgebildet, sondern zus\u00e4tzlich deren Nachgiebigkeit ber\u00fccksichtigt. Zur Bestimmung der Position und Orientierung des TCP bei gegebenen Achswinkeln unter Ber\u00fccksichtigung einer Last wird die Gesamt-Transformationsmatrix gebildet:

$$N_{TCP} = N_1 N_2 N_3 N_4 N_5 N_6 \quad (12)$$

Wird zus\u00e4tzlich die konventionelle Vorw\u00e4rts-Kinematik berechnet ($\alpha_n = \beta_n = \gamma_n = 0$), so kann durch die Subtraktion der beiden Ergebnisse die Abweichung der Roboter-Pose gegen\u00fcber dem unbelasteten Zustand bestimmt werden.

Die Rechenzeit eines derart aufgebauten analytischen Modells zur Bestimmung der Verlagerung am TCP aufgrund einer wirkenden Kraft liegt im Bereich von 1 bis 2 ms. Somit kann die oben beschriebene Echtzeitbedingung erf\u00fcllt werden. Prinzipiell w\u00e4re es zudem m\u00f6glich, die Nachgiebigkeit der Strukturkomponenten des Roboters durch weitere Rotationsmatrizen zu ber\u00fccksichtigen.

5 Validierung

Das dargestellte Vorgehen wurde zunächst anhand eines Mehrkörpersystems verifiziert. Dies bietet den Vorteil, dass das System vollständig beobachtbar ist und alle Steifigkeitsparameter bekannt sind. Der Roboter wurde im MKS-Programm RecurDyn® modelliert, wobei die Strukturkomponenten durch Starrkörper und die Gelenke durch lineare Federn abgebildet wurden. Mit diesem Modell wird der in Abbildung 1 gezeigte Messaufbau virtuell nachgestellt, indem der Roboter durch eine definierte Kraft belastet wird. An Stelle der Messpunkte werden so genannte Marker auf dem Simulationsmodell definiert, deren Positionen zu jedem Zeitpunkt ausgegeben werden können. Aus den Bewegungen der Marker werden gemäß dem Vorgehen aus Abschnitt 3.3 die Verdrehungen der einzelnen Achsen und daraus die entsprechenden Steifigkeiten bestimmt. Ein Vergleich der im Simulationsmodell definierten Steifigkeiten mit den berechneten zeigte für alle sechs Robotergelenke eine nahezu exakte Übereinstimmung. Somit ist sichergestellt, dass sämtliche Berechnungsalgorithmen in der Lage sind, die gesuchten Größen zu identifizieren und dass sie korrekt implementiert sind.

Die einzige Möglichkeit, die Validität der aufgezeigten Methode nachzuweisen, besteht darin, gemessene und berechnete Verlagerungen des Roboters miteinander zu vergleichen. Hierfür erfolgte eine Messung am Roboter (Abschnitt 3.1) und die Berechnung der Steifigkeitsparameter (Abschnitt 3.3) sowie der Aufbau des analytischen Steifigkeitsmodells auf Basis der identifizierten Parameter (Abschnitt 4). Im Punkt der maximalen Auslenkung des Roboters (Abbildung 3, links) wurde mit Hilfe des Steifigkeitsmodells die Verschiebung des Kraftangriffspunkts auf Basis der wirkenden Kraft berechnet. Ein Vergleich der berechneten und der gemessenen Verschiebung ist in der folgenden Tabelle dargestellt.

	Messung	Berechnung
Δx	0,84 mm	0,75 mm
Δy	-0,34 mm	-0,05 mm
Δz	3,33 mm	3,16 mm

Tabelle 1: Verschiebung am Kraftangriffspunkt

Es zeigt sich, dass in Richtung des Kraftangriffs, in der auch die maximale Auslenkung vorliegt, eine gute Übereinstimmung zwischen der gemessenen und der berechneten Verlagerung besteht. Insbesondere in y-Richtung ist allerdings eine deutliche Diskrepanz erkennbar, welche auf mehrere Ursachen zurückzuführen ist. Zum einen unterliegen die Strukturkomponenten des Roboters während der Messung einer gewissen Verformung, so dass die Berechnung der Verdrehwinkel, bei denen diese als Starrkörper betrachtet werden, nicht zutreffend ist. Außerdem sind die Messdaten sämtlicher Punkte mit Rauschen behaftet, was ebenfalls zu Fehlern bei der Berechnung der Verdrehwinkel führt. Künftig sollen die Auswirkungen dieser Effekte durch einen Optimierungsalgorithmus zur Nachbearbeitung der Messdaten vermindert werden, um die Genauigkeit des Steifigkeitsmodells weiter zu verbessern.

6 Zusammenfassung und Ausblick

Bei der Fräsbearbeitung mit Industrierobotern kommt es aufgrund der wirkenden Zerspankräfte zu einer starken Abdrängung des Roboters. Aus diesem Grund ist der Einsatz von Fräsrobotern auf Anwendungsgebiete mit geringen Genauigkeitsanforderungen und niedrigen Zerspankräften beschränkt. Ein Ansatz zur Reduktion der auftretenden Verlagerung besteht in deren steuerungstechnischer Kompensation, wofür ein echtzeitfähiges Steifigkeitsmodell des Roboters erforderlich ist. In diesem Beitrag wurde eine Methode zum Aufbau eines solchen Modells aufgezeigt. Im ersten Schritt wird mit Hilfe eines 3D-Scanning-Laser-Doppler-Vibrometers die Auslenkung des Roboters bei einer Belastung mit definierter Kraft erfasst. Im Anschluss werden aus den gewonnenen Messdaten die Verdrehwinkel aller Robotergelenke bestimmt, woraus die Steifigkeiten der Getriebe und der Lager berechnet werden. Diese werden in einem analytischen Nachgiebigkeitsmodell des Roboters, welches eine Erweiterung der konventionellen Vorwärtskinematik darstellt, eingesetzt. Es konnte gezeigt werden, dass auf diese Weise eine echtzeitfähige Prognose der Verlagerung des TCP aufgrund der wirkenden Last möglich ist.

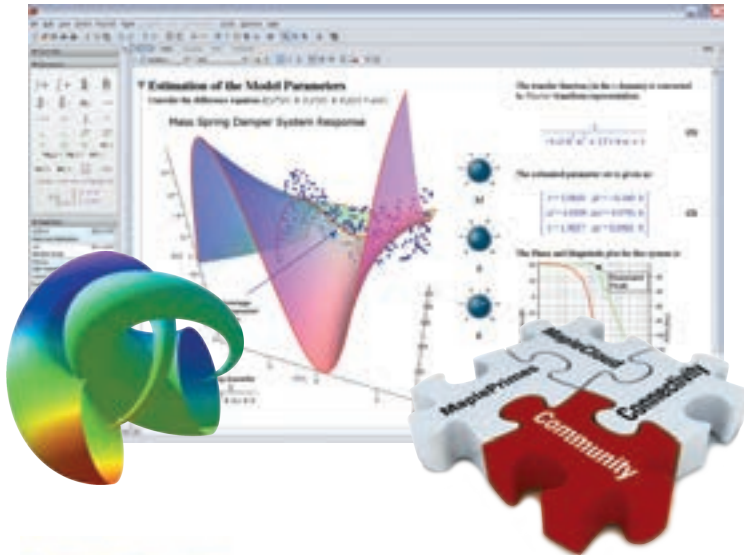
Im nächsten Schritt wird ein Algorithmus zur Nachbearbeitung der Messdaten entwickelt, um die Genauigkeit bei der Bestimmung der Gelenk-Verdrehwinkel noch weiter zu verbessern. Hiermit soll es möglich sein, auch die Nachgiebigkeit der Strukturkomponenten des Roboters zu erfassen und in das Nachgiebigkeitsmodell zu integrieren.

Literatur

- [1] Hölzl, J.: Modellierung, Identifikation und Simulation der Dynamik von Industrierobotern. Düsseldorf: VDI 1994.
- [2] Thümmel, M.: Modellbasierte Regelung mit nichtlinearen inversen Systemen und Beobachtern von Robotern mit elastischen Gelenken. Diss. TU München (2006).
- [3] Kurze, M.: Modellbasierte Regelung von Robotern mit elastischen Gelenken ohne abtriebsseitige Sensorik: Modellbasierte Regelung von Robotern mit elastischen Gelenken ohne abtriebsseitige Sensorik. Diss. TU München (2008).
- [4] Reiner, M.: Modellierung und Steuerung von strukturelastischen Robotern. Diss. TU München (2010).
- [5] Abele, E.; Rothenbücher, S.; Weigold, M.: Cartesian compliance model for industrial robots using virtual joints. *Production Engineering* 2 (2008) 3, S. 339-343.
- [6] Craig, R.; Bampton, M.: Coupling of Substructures for Dynamic Analyses. In: AIAA (Hrsg.): AIAA JOURNAL 1968 vol. 6 no. 7, S. 1313–1319.
- [7] Zaeh, M. F.; Bonin, T.; Roesch, O.: Simulation of operating deflection shapes of a gear shaping machine with flexible multi-body systems. *Interaction of Simulation and Testing: New Requirements and New Opportunities in Structural Dynamics*. Wiesbaden 12.-13. November 2008.
- [8] Abele, E.; Bertsch, C.; Laurischkat, R.; Meier, H.; Reese, S.; Stelzer, M.; Stryk, O.: Comparison of Implementations of a Flexible Joint Multibody Dynamics System Model of an Industrial Robot. In: Teti, R. (Hrsg.): 6th CIRP International Conference on Intelligent Computation in Manufacturing Engineering. Napel (Italien), 23.-25. Juli 2008.

- [9] Zäh, M. F.; Rösch, O.: Steigerung der Arbeitsgenauigkeit bei der Fräsbearbeitung mit Industrierobotern. ZWF Zeitschrift für wirtschaftlichen Fabrikbetrieb 106 (2011) 09/2011, S. 658-662.
- [10] W. Seyffferth, A. J.: Nonlinear Modeling and Parameter Identification of Harmonic Drive Robotic Transmissions. In: Robotics and Automation Society; Nihon-Gakujutsu-Kaigi (Hrsg.): Proceedings of 1995 IEEE International Conference on Robotics and Automation. Piscataway, NJ: IEEE Service Center 1995. S. 3027-3032.
- [11] Weigold, M.: Kompensation der Werkzeugabdrängung bei der spanenden Bearbeitung mit Industrierobotern. Aachen: Shaker 2008. (Schriftenreihe des PTW).
- [12] F. Pfeiffer; J. Hölzl: Parameter Identification for Industrial Robots. In: Robotics and Automation Society; Nihon-Gakujutsu-Kaigi (Hrsg.): Proceedings of 1995 IEEE International Conference on Robotics and Automation. Piscataway, NJ: IEEE Service Center 1995. S. 1468-1476.
- [13] Meyberg, K.; Vachenauer, P.: Höhere Mathematik. 6 Aufl. Berlin: Springer 2001.

Neue Herausforderungen in der Wissenschaft brauchen leistungsfähige Werkzeuge



Maple™ 15

Maple, das Ergebnis von über 25 Jahren Forschung und Entwicklung, kombiniert die weltweit leistungsfähigste mathematische Engine mit einem intuitiven "Clickable" User Interface.



MapleSim™ 5

MapleSim ist ein einzigartiges Werkzeug zur physikalischen Multi-Domain-Modellierung und Simulation.

Preisangebot anfordern:
www.maplesoft.com/angebot

Kontakt aufnehmen:

 +49 (0)241/980919-30



Beitrag zur Rückfederungsreduzierung hochfester Strukturblechteile

Patrick Sacher, Ostfalia HaW

Martin Rambke, Ostfalia HaW

pat.sacher@ostfalia.de

m.rambke@ostfalia.de

Zusammenfassung

Die Rückfederungsproblematik bei der Umformung von hochfesten Blechen ist in der Automobilindustrie ein prägnantes Thema. Die große Aufsprungneigung der hochfesten Güten erschwert die Beherrschung von Umform- und Folgeprozessen.

Ein Beitrag zur Rückfederungsreduzierung soll durch das Forschungsprojekt OPTISTUF entstehen. Hierbei werden durch Modifikation der Vorziehstufe mit Hilfe der numerischen Simulation verschiedene Variantenberechnungen durchgeführt und interpretiert. Ziel ist es hierbei, einen Zusammenhang zwischen Spannungen/Dehnungen und den daraus resultierenden Rückfederungen zu erlangen. Es soll eine Methode zur optimalen Gestaltung der Vorziehstufe hinsichtlich minimaler Rückfederung nach der Umformung entwickelt werden.

Eine Voruntersuchung zeigt auf, dass es möglich ist, Spannungszustände durch einen mehrstufigen Prozess bewusst in das Bauteil einzuleiten und somit die Rückfederung zu beeinflussen. Verwendet man andere Radien in der Vorziehstufe als in der letzten Formgebungsstufe, kann zusätzliches Strecken oder Stauchen der Radienbereiche die Richtung der rückfederungsbedingten Maß- und Formabweichungen ändern.

Ferner erfolgt ein Vergleich unterschiedlicher numerischer Methoden. Hierbei wird geprüft, ob die Rückfederungsberechnung nach der letzten Formgebungsstufe ausreichend ist, oder ob diese nach jeder Umformoperation erfolgen muss.

1 Forschungsprojekt OPTISTUF

Um das Fahrzeuggewicht zu senken und somit einen Beitrag zur verringerten CO₂-Emission zu leisten, unternimmt die Automobilindustrie enorme Anstrengungen. Ein Ansatz besteht darin, hochfeste Strukturblechteile zu verarbeiten. Somit können bei gleichen Festigkeitsanforderungen die Blechdicken reduziert werden.

Problematisch hierbei ist, dass Stahlbleche mit höheren Festigkeitsgüten und geringeren Blechdicken nach der Umformung eine starke Rückfederungsneigung aufweisen. Da das Tiefziehen solcher Strukturblechteile am Anfang der Prozesskette steht und diese nach der Umformung in Vorrichtungen gefügt werden müssen, sind nur geringe Maß- und

Formabweichungen tolerierbar. Folglich müssen verschiedene Maßnahmen im industriellen Einsatz durchgeführt werden, damit das starke Aufsprungverhalten reduziert werden kann.

Beispielsweise können Bauteilmodifikationen Anwendung finden, bei denen Werkzeugradialen verkleinert und zusätzliche Versteifungsrippen eingebracht werden. Nachteilig ist, dass Designänderungen erforderlich sind und die Reierwahrscheinlichkeit erhht wird. Eine weitere Mglichkeit ist die Verwendung von Ziehstben und die Erhhung der Niederhalterkraft. So ist es mglich, mit der Variation von Prozessparametern, die Ausstreckung im Blechteil zu vergrern und die rckfederungsbedingten Ma- und Formabweichungen zu reduzieren. Auch hier ist eine erneute Herstellbarkeitsprfung notwendig, da Reier auftreten knnen. Ferner wird die Rckfederungskompensation durchgefhrt. Hierbei wird mit Hilfe der Umformsimulation das Ma des Aufsprunges ermittelt und in einem iterativen Prozess eine neue Werkzeugkontur ermittelt. Somit ist es mglich, dass das Bauteil in die Soll-Geometrie aufspringt. Nachteilig bei dieser Methode ist, dass zur Herstellung einiger Geometrien Hinterschnitte erforderlich sind, die werkzeugtechnisch nicht oder nur mit kostenintensiven Bemhungen umsetzbar sind. [1]

Diese Manahmen werden im industriellen Tagesgeschft kombiniert angewendet, um Toleranzvorgaben einhalten zu knnen. Da diese Methoden jedoch einige Nachteile aufweisen, soll ein neuer innovativer Ansatz zur Rckfederungsreduzierung erforscht werden.

Hierzu wurde ein Strukturbauteil aus der Praxis ausgewhlt. Dieses Bauteil ist in der Abbildung 1 zu erkennen. Es handelt sich um einen Lehnenseitenholm der Sitzstruktur. Da das Bauteil eine hohe Komplexitt aufweist und somit fr Forschungszwecke ungeeignet ist, wurden die wesentlichen Charaktereigenschaften auf einen Demonstrator transferiert, der sich fr diese Untersuchung besser eignet. bertragene Merkmale sind die asymmetrische gekrmmte Geometrie, die Umformbarkeit in einem zweistufigen Prozess, Aufnahmemglichkeit durch RPS¹ und das hochfeste Material (DP 1000).



Abbildung 1: Demonstrator und Praxisbauteil aus Sitzstruktur

¹ RPS: Referenz-Punkt-System

Im Forschungsvorhaben OPTISTUF (gefördert von BMBF/AIF, Förderlinie FH profUnt, Projektleiter Prof. Dr.-Ing. Martin Rambke) werden Ansätze zur Ermittlung optimaler Vorziehstufen zur Erhöhung der Maßhaltigkeit hochfester Strukturblechteile untersucht. Hierzu erfolgte eine Orientierung an der Methode des Lehnenseitenholms, um die Methode des Demonstrators zu definieren. Diese ist in der Abbildung 2 ersichtlich. Ferner ist in der Darstellung die grundsätzliche Vorgehensweise bei der Untersuchung erkennbar.

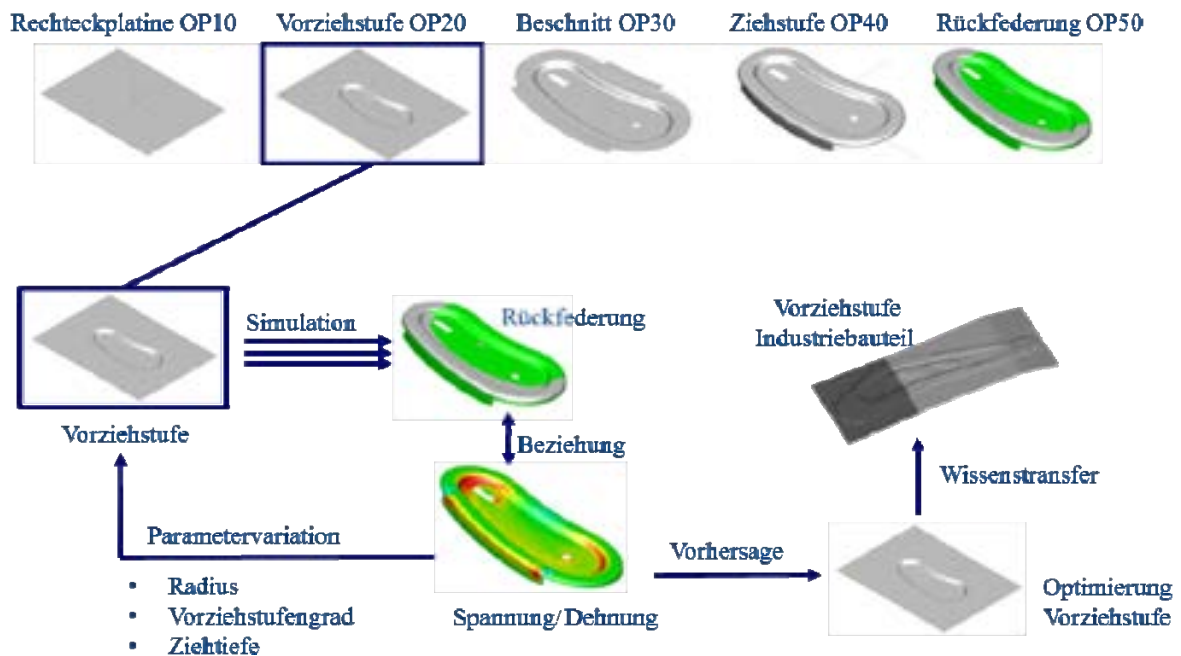


Abbildung 2: Methode und Vorgehensweise [2]

Aus der Methode geht hervor, dass aus einer Rechteckplatte in der Vorziehstufe die Innenkontur gezogen wird. Nach einer Beschnittoperation erfolgt die Formgebung der zweiten Ziehstufe. Wird der Demonstrator aus dem Werkzeug entnommen, federt dieser auf.

Mit Hilfe der Umformsimulationssoftware PAM-STAMP werden die Umformstufen explizit und die Rückfederungsberechnung implizit durchgeführt. Da eine Variation der Vorziehstufe vorhanden ist, werden verschiedene Parameter betrachtet. Beispielsweise werden Radien in der Vorziehstufe anders gezogen als in der letzten Formgebungsstufe. Weiterhin werden die Ziehtiefe und der Grad der Vorziehstufe variiert. Folglich erhält man nach der Simulation verschiedene Rückfederungsergebnisse, die miteinander verglichen werden können. Weiterhin lassen sich mit der Umformsimulation Spannungen und Dehnungen des Bauteils ausgeben, die mit der Rückfederung in Beziehung gesetzt werden können. Mit diesen Erkenntnissen soll eine optimale Vorziehstufe des Demonstrators gefunden werden, mit der nach der letzten Formgebungsstufe geringe Maß- und Formabweichungen erzielt werden können. Wenn dieses Wissen zu einer Methodik optimaler Vorziehstufen formuliert werden kann, ist es möglich, für die Vorziehstufen von Praxisbauteilen eine optimale Gestaltung hinsichtlich rückfederungsreduzierender Wirkung zu finden.

2 Ergebnisse der Voruntersuchung

Zur Untersuchung von rückfederungsbedingten Formabweichungen wurden bisher oftmals einfachgezogene Hutprofile verwendet. Hierbei konnten allgemeingültige Aussagen zum Rückfederungsverhalten von primitiven Bauteilgeometrien erlangt werden. [3][4][5]

Um auch generelle Aussagen zum Rückfederungsverhalten von mehrstufig umgeformten Bauteilen zu erhalten, wurde diese Voruntersuchung initiiert. Hierbei wird die zu untersuchende Geometrie weiterhin vereinfacht, sodass das Rückfederungsverhalten eines offenen Strukturteils erforscht wird. Dieser Sachverhalt wird in der Abbildung 3 ersichtlich.

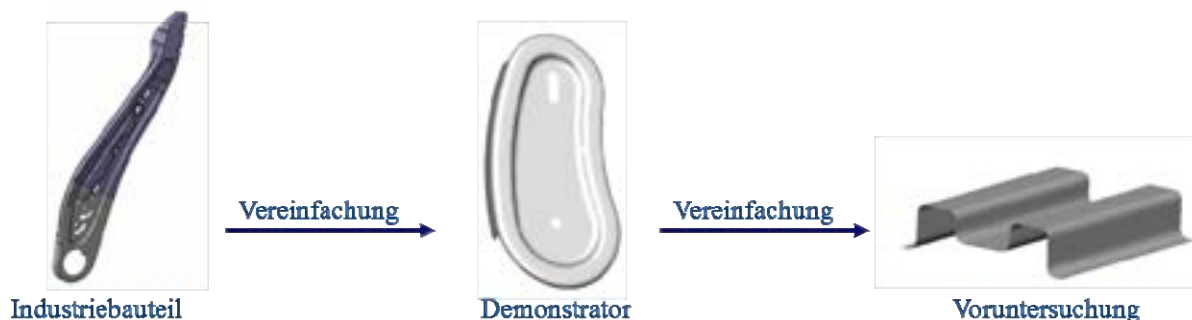


Abbildung 3: Entwicklung der Geometrie für Voruntersuchung

Es handelt sich im Gegensatz zum Demonstrator um ein einfaches symmetrisches Profil. Eine weitere Vereinfachung ist die offene Geometrie. Vorteilhaft ist hierbei, dass in dem Bauteil während und nach der Umformung vergleichbar einfachere Spannungszustände entstehen. Somit ist eine schnelle Interpretation der Ergebnisse möglich.

Aus der Bauteilgeometrie kann die Werkzeugwirkfläche der zweiten Formgebungsstufe durch Ankonstruktion abgeleitet werden. Die Werkzeugwirkfläche der ersten Ziehstufe ergibt sich durch Abwicklung des Bauteils. Es wird definiert, dass in der ersten Ziehstufe die Innenkontur gezogen wird.

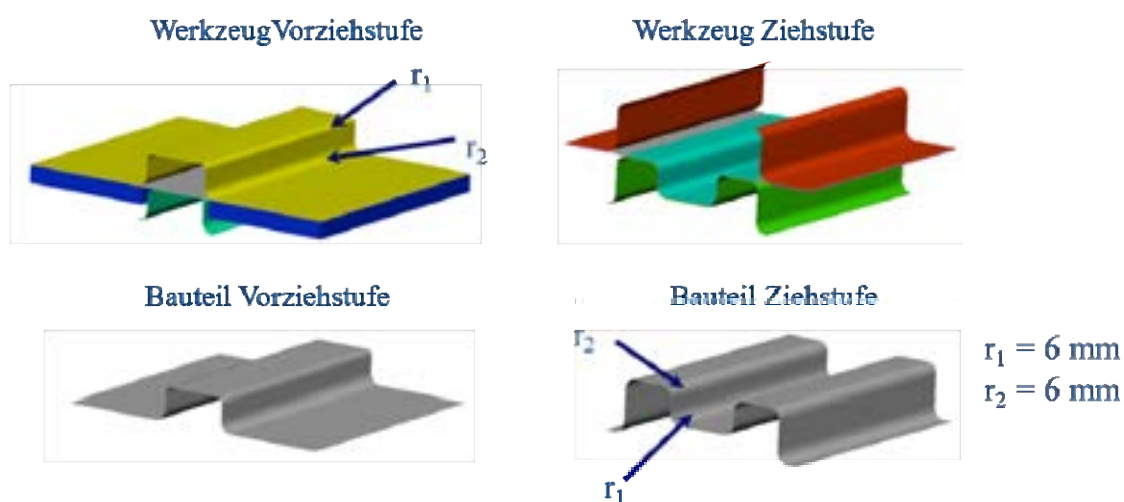


Abbildung 4: Variation Radius Voruntersuchung

Die Geometrie der zweiten Ziehstufe ist mit den Radien $r_1=6\text{ mm}$ und $r_2=6\text{ mm}$ festgelegt. Das Werkzeug der Vorziehstufe wird hingegen in den Radien r_1 und r_2 variiert. Werden nun mittels Umformsimulation mehrere Varianten durchgerechnet, erhält man bei jeder Modifikation der ersten Ziehstufe auch unterschiedliche Rückfederungsergebnisse nach der zweiten Ziehstufe. Diese Geometrien wurden in eine Datei importiert und durch einen Bauteilschnitt verglichen. Es wurden mehrere Messpunkte und die Richtung der Rückfederung definiert. Dieser Zusammenhang wird in Abbildung 5 verdeutlicht.

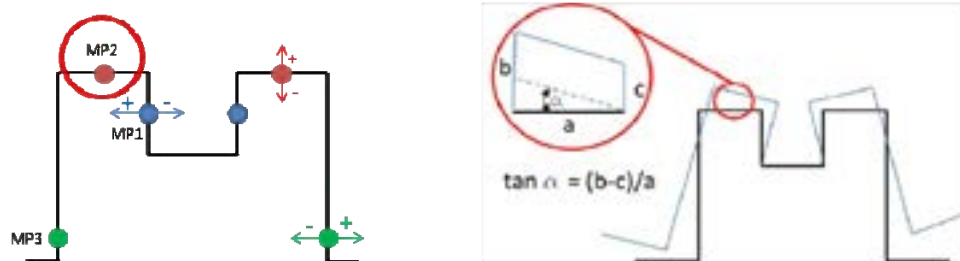
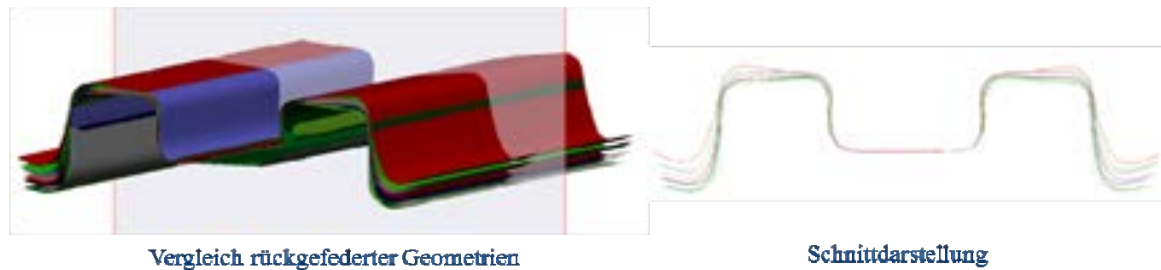


Abbildung 5: Auswertung der Rückfederungsergebnisse der Voruntersuchung

Folgend wird ein Auszug der Messergebnisse vorgestellt. Dazu betrachten wir den Messpunkt 2 (MP2) in der Abbildung 5 genauer. In diesem Messbereich wurden über trigonometrische Beziehungen die Winkeldifferenz zwischen rückgefederter und Soll-Kontur ermittelt. Das Ergebnis ist in der Abbildung 6 ersichtlich.

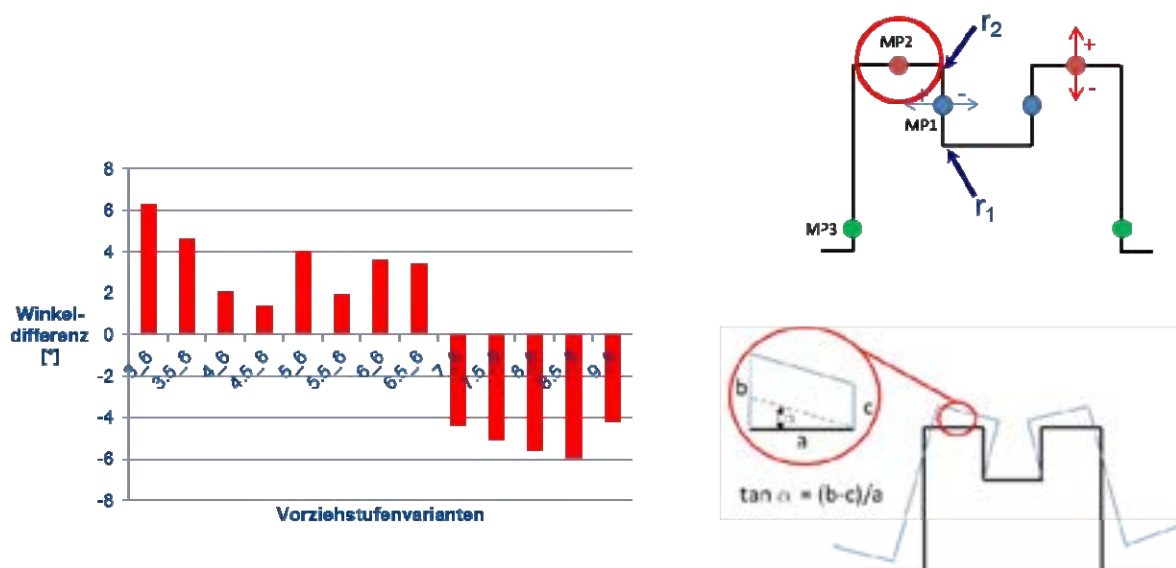


Abbildung 6: Messergebnisse MP2

Auf der Ordinate wird die Winkeldifferenz [°] angegeben. Hierbei ist zu beachten, dass die Rückfederung positiv oder negativ sein kann. Auf der Abszisse wird die Vorziehstufenvariante dargestellt. Es ist ein Benennungsschlüssel erkennbar. Der erste Wert entspricht dem Radius r_1 und der zweite Wert entspricht dem Radius r_2 der Vorziehstufe. Beispielsweise bedeutet 4_6, dass in der Vorziehstufe der Radius $r_1=4$ mm kleiner vorgezogen wurde als in der zweiten Ziehstufe (6 mm). Der Radius r_2 hingegen entspricht in der gesamten Versuchsreihe auch dem Radius der zweiten Ziehstufe (6 mm).

Bemerkenswert ist, dass man nach der zweiten Ziehstufe eine positive Rückfederung erhält wenn man in der Vorziehstufe einen kleineren Radius wählt. Verwendet man hingegen einen größeren Radius für r_1 als in der zweiten Ziehstufe, ändert sich das Rückfederungsvorzeichen in die negative Richtung. Im nächsten Schritt wird versucht eine Beziehung zwischen den Spannungen und dem beobachteten Wechsel der Rückfederungsrichtung zu ermitteln (Abbildung 7).

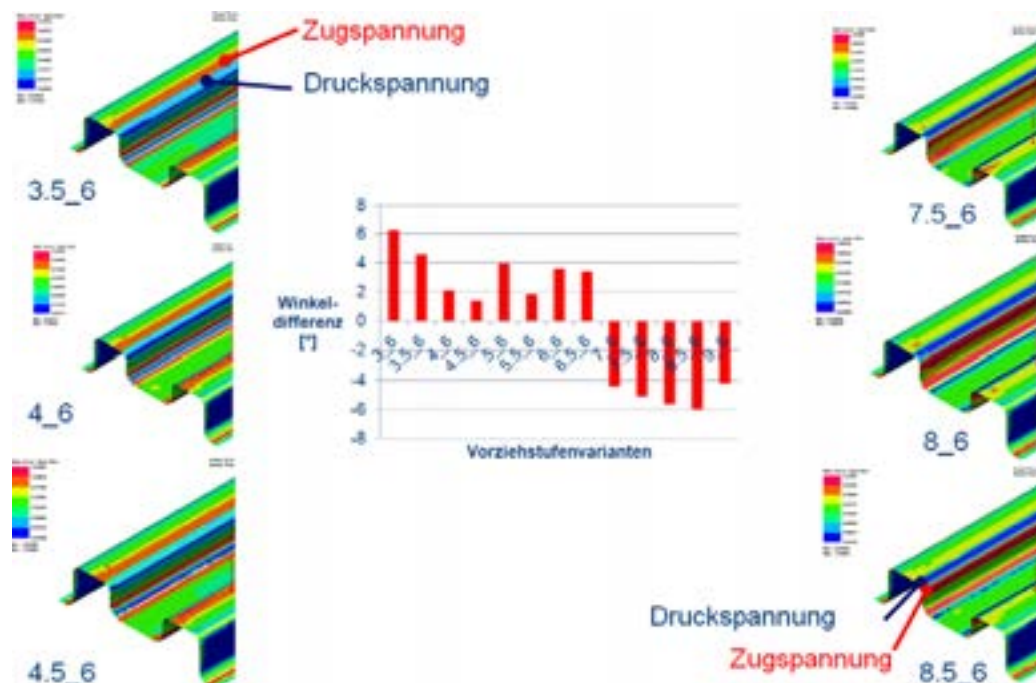


Abbildung 7: Beziehung zwischen Spannung und Rückfederung

In der Voruntersuchung wurden die Hauptspannungen in der oberen Membranebene ausgewertet und verglichen. Im Messpunkt 2 hat die Auswertung der Rückfederung ergeben, dass bei den Vorziehstufenvarianten $r_1=3$ mm-6,5mm eine positive Rückfederung nach dem Fertigzug vorhanden war. Größere Radien ($r_1=7$ mm-9mm) führten zu einer negativen Rückfederung am MP2.

Betrachtet man nun die Hauptspannungen der oberen Membran (Abbildung 7), so wird deutlich, dass für die Vorziehstufenvarianten $r_1 = 3,5$ mm/4mm/4,5mm Zug- und Druckspannung dieselbe Anordnung aufweisen. Für $r_1 = 7,5$ mm/8mm/8,5mm hingegen ist die Anordnung von Zug- und Druckspannung umgekehrt. Es ist ein Wechsel der Spannungsanordnung im betrachteten Radienbereich zu erkennen.

Diese Spannungsumkehr korreliert mit der Umkehr der Rückfederungsrichtung. Diese Ergebnisse zeigen, dass durch die Modifikation der Radien in der Vorziehstufe die Möglichkeit besteht, einen gezielten Spannungszustand herzustellen und somit die Rückfederung nach dem Fertigzug zu beeinflussen.

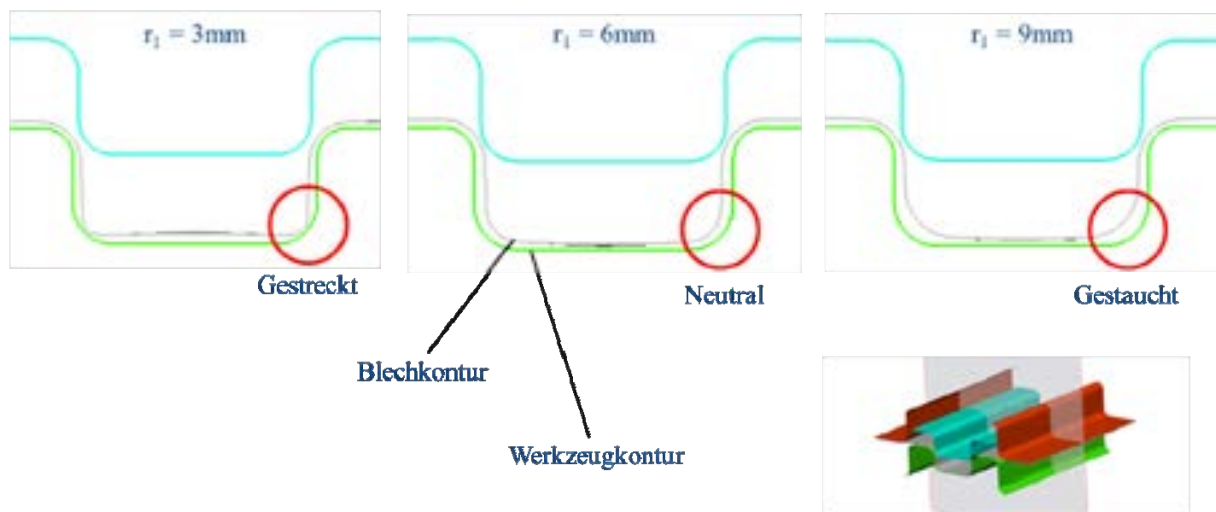


Abbildung 8: Beeinflussung der Spannungszustände durch Radienmodifikation [2]

In der Abbildung 8 wird ersichtlich, wie es möglich ist, mit Radienvariation in der Vorziehstufe gezielt den Spannungszustand nach der zweiten Ziehstufe zu beeinflussen. Es ist eine Schnittdarstellung durch die Mitte des Werkzeuges dargestellt. Die Blechkontur ist grau, der Support (matrizenseitiger Niederhalter) ist blau und der Stempel weist eine grüne Farbe auf. In der mittleren Position erkennt man, dass die Blechkontur der idealen Werkzeugkontur entspricht, da Vorziehstufenradius und der Radius der zweiten Ziehstufe übereinstimmen. Betrachtet man hingegen den linken Bildausschnitt, sieht man, dass die Blechkontur nicht mit der Werkzeugkontur übereinstimmt. In der Vorziehstufe erfolgte die Umformung mit einem kleinen Radius (3mm). Das bedeutet, dass das Material eine zusätzliche Streckung erfährt, wenn der Support während der zweiten Ziehstufe in die Halteposition geht. In der rechten Bildhälfte erkennt man hingegen, dass das Material bei einem größeren Radius (aus der Vorziehstufe) während der zweiten Stufe gestaucht wird.

Somit ist es möglich durch eine Radienvariation in der ersten Ziehstufe gezielt die Spannungsanordnung nach der zweiten Ziehstufe und damit die Rückfederung zu beeinflussen.

3 Numerische Methode

Um die Rückfederung mit Hilfe der Simulation korrekt vorherzusagen, ist es wichtig, dass die reale Methode möglichst mit der numerischen Methode übereinstimmt. Rückfederungsberechnungen werden jedoch nicht in der Regel nach jeder Operation durchgeführt. In der betrieblichen Praxis erfolgt eine Rückfederungsberechnung nur nach ausgewählten umformtechnischen Prozessschritten. Analysen zur durchgängigen numerischen

Betrachtung der Methode mit dem Fokus auf rückfederungsbedingte Maß- und Formabweichungen erfolgten auch bei komplexen Karosseriebauteilen [6][7].

Bei zweistufigen Umformprozessen ist die Frage berechtigt, ob eine Rückfederungsberechnung nur nach der zweiten Ziehstufe erfolgen muss, oder ob diese nach der Vorziehstufe und nach der letzten Formgebungsstufe sinnvoll ist. Theoretisch ist es ausreichend, nach der zweiten Ziehstufe die Rückfederungsberechnung durchzuführen, da die Spannungszustände aus der ersten Ziehstufe in die zweite übergeben werden. Eine Überprüfung des Sachverhaltes wurde an Hand der offenen Strukturgeometrie aus der Voruntersuchung vorgenommen. Das Ergebnis des Vergleichs der numerischen Methode ist in der Abbildung 9 vorhanden.

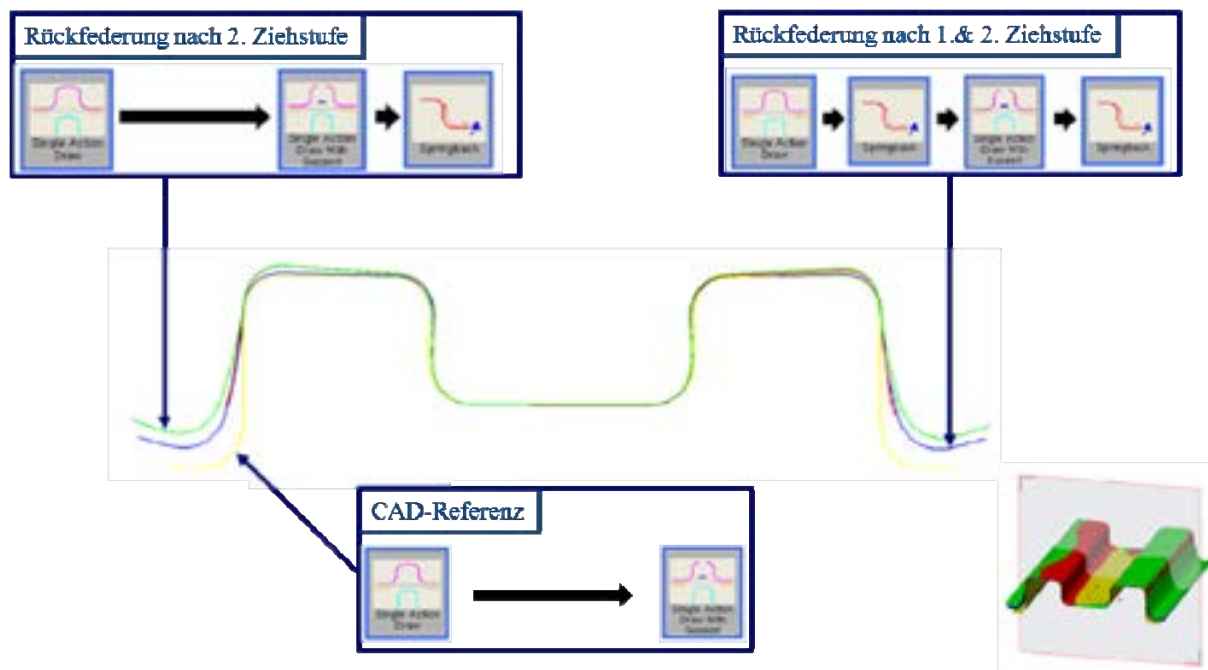


Abbildung 9: Rückfederungsvergleich unterschiedlicher numerischer Methoden [2]

Die CAD-Referenz ist gelb dargestellt. Das bedeutet, dass lediglich die erste und zweite Umformstufe ohne Rückfederungsberechnung ermittelt wurde. Dies entspricht der Soll-Kontur und dient als Geometrievergleich. Grün hingegen ist die aufgesprungene Geometrie, wobei die Rückfederungsberechnung nur nach der zweiten Formgebungsstufe stattgefunden hat. Dies entspricht der numerischen Methode, die heute Stand der Technik ist. Betrachtet man hingegen die blaue Kontur, stellt man fest, dass der Aufsprung ein kleineres Maß aufweist. Hier wurde eine Rückfederungsberechnung nach der ersten und nach der zweiten Umformstufe durchgeführt. Die Erklärung für die geringere Rückfederung ist in Abbildung 10 dargestellt.

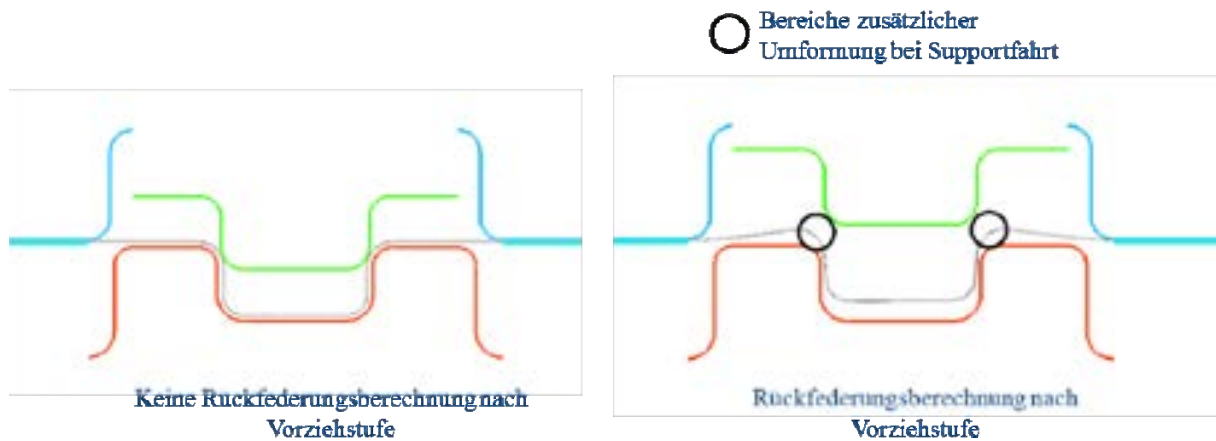


Abbildung 10: Begründung unterschiedlicher Rückfederungsergebnisse [2]

Es ist ein Mittelschnitt durch das Werkzeug der zweiten Ziehstufe dargestellt. Im linken Bildausschnitt erkennt man, dass das Blech (grau) der Sollkontur des Werkzeuges entspricht. Dies ist der Fall, da keine Rückfederungsberechnung nach der ersten Ziehstufe durchgeführt wurde. Da es sich jedoch um einen hochfesten Stahl handelt, treten schon nach der Vorziehstufe beachtliche Rückfederungseffekte auf, die beachtet werden müssen. Im rechten Bildausschnitt erkennt man, dass das Blech nicht der Werkzeugkontur entspricht und deshalb in der zweiten Stufe eine zusätzliche Umformung stattfindet, wenn der Support (matrizenseitiger Niederhalter) in die vorgegebene Position verfährt. Durch die zusätzliche Plastifizierung ergibt sich eine geringere Gesamtrückfederung nach der Umformung. Hierbei muss beachtet werden, dass das Werkstück in der Realität ebenfalls nach der ersten und nach der zweiten Ziehstufe aufspringt und deshalb sollte die Rückfederungsberechnung nach jeder Operation berechnet werden.

Um dieses Ergebnis zu prüfen, wurde derselbe numerische Methodenvergleich am Demonstrator wiederholt. Das Resultat ist in Abbildung 11 dargestellt.

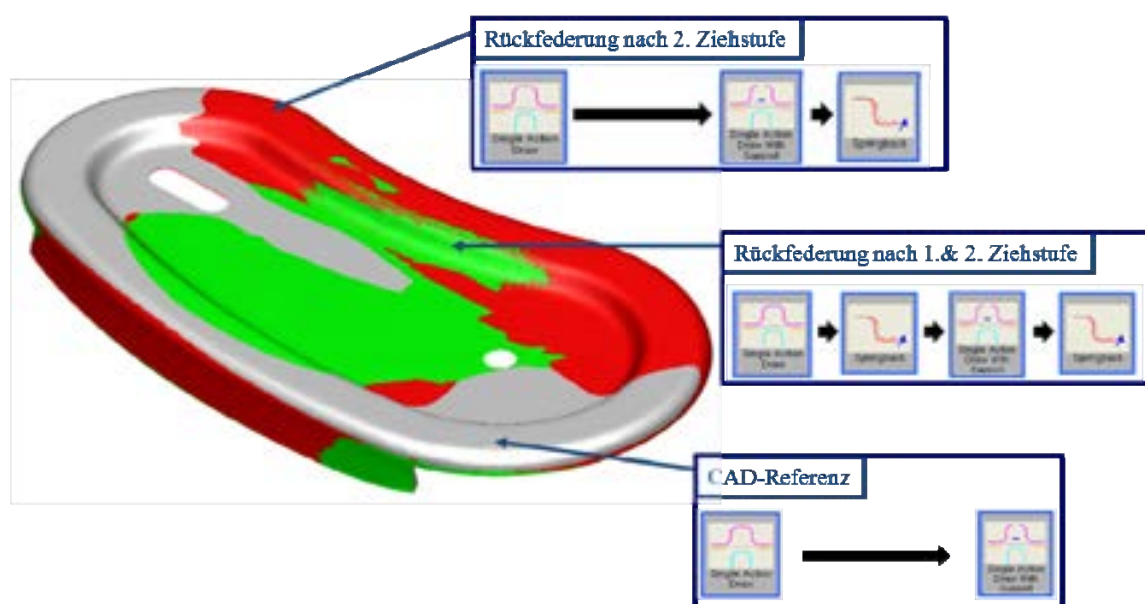


Abbildung 11: Numerischer Rückfederungsvergleich Demonstrator

Die graue Kontur entspricht der Demonstratorgeometrie, da hier keine Rückfederungsberechnung stattgefunden hat. Der rote Demonstrator entspricht der Geometrie, die man in der Simulation erhält, wenn die Rückfederungsberechnung nur nach der zweiten Ziehstufe durchgeführt wurde. Grün hingegen ist die Geometrie mit zwei Rückfederungsberechnungen nach jeder Formgebungsstufe. Man erkennt, dass auch hier Unterschiede vorhanden sind und die Ergebnisse nicht übereinstimmen. Um einen besseren Vergleich der numerischen Methoden durchführen zu können, wurden drei Bauteilschnitte definiert. Das Resultat ist in der Abbildung 12 dargestellt.

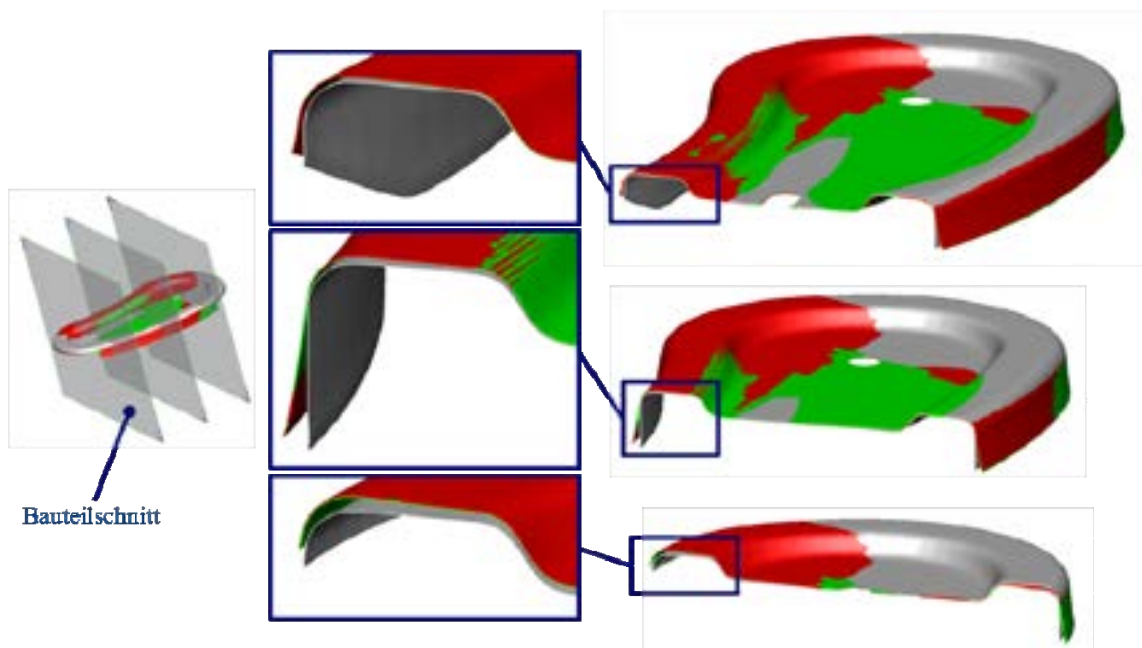


Abbildung 12: Analyse durch Bauteilschnitt Demonstrator

Im jeweiligen Detailbereich erkennt man, dass in allen drei Schnitten der Aufsprung zur Soll-Kontur mit nahezu gleichem Betrag ermittelt wurde. Bei diesem geschlossenem Profil, welches eine höhere Komplexität aufweist als das offene Strukturbauteil, hat die Wahl der numerischen Methode einen geringeren Einfluss.

Es wird vermutet, dass der Einfluss der Methode von der Rückfederung nach der ersten Ziehstufe abhängt. Ist dieser Aufsprung groß, sodass das Bauteil nicht mit der Werkzeugkontur übereinstimmt, steigt auch die Wahrscheinlichkeit, dass zusätzliche Umformung stattfindet und somit das Rückfederungsergebnis nur richtig vorhergesagt werden kann, wenn eine Berechnung nach jeder Formgebungsstufe erfolgt ist. Da durch den vermehrten Einsatz hochfester Stähle die Rückfederungsproblematik auch in der Vorziehstufe prägnant ist, wird grundsätzlich empfohlen, die Berechnung der Maß- und Formabweichung nach jeder Operation durchzuführen.

Literatur

- [1] Roll, Karl: *Simulationsgestützte Kompensation der Rückfederung*, LS-DYNA Anwenderforum ,Bamberg, 2004
- [2] Rambke, M.: *Contribution to reducing spring-back of high-strength sheet metal parts*, Automotive Circle International, Bad Nauheim, 2011
- [3] Hütte, Holger: *Untersuchung zum Rückfederungsverhalten hochfester Stahlblechwerkstoffe beim Tiefziehen*, Dissertation, 2001
- [4] Boinski, Frank: *Auslegung von Ziehteilen und Preßwerkzeugen mit elementaren Methoden unter besonderer Berücksichtigung der Rückfederung*, Dissertation, 1997
- [5] Weigert, Philipp: *Berücksichtigung formänderungsbedingter Effekte (Rückfederung) im Entwicklungsprozeß der Methodenplanung von tiefgezogenen Karosseriebauteilen*, Dissertation, 2009
- [6] Rohleder, Martin: *Simulation rückfederungsbedingter Formabweichungen im Produktentstehungsprozess von Blechformteilen*, Dissertation, Shaker Verlag, Aachen 2002
- [7] Fleischer, Michael: *Absicherung der virtuellen Prozesskette für Folgeoperationen in der Umformtechnik*, Dissertation, Shaker Verlag, Aachen 2009

Ein kalibrierbares Modell zur Beschreibung des Spritzgießprozesses elektrisch und wärmeleitfähiger Thermoplaste

Dipl.-Ing. (FH) Jens Dörner

jens.doerner@uni-due.de

Prof. Dr.-Ing. Johannes Wortberg

johannes.wortberg@uni-due.de

Universität Duisburg-Essen, Institut für Produkt Engineering,
Lehrstuhl für Konstruktion und Kunststoffmaschinen

Zusammenfassung

Die Spritzgießsimulation ermöglicht in vielen Entwicklungsstufen Zeit und damit auch Kosten einzusparen. Für übliche thermoplastische Kunststoffe sind die Simulationsmethoden seit langer Zeit Stand der Technik und auch die Genauigkeit der Resultate ist in aller Regel ausreichend um Prognosen über die Formfüllung und Entstehung von Bindenähten treffen zu können. Funktionelle Kunststoffe wie z.B. elektrisch leitfähige Thermoplaste jedoch haben derart veränderte Materialeigenschaften, sodass die exakte Beschreibung ohne Anpassungen im Simulationsprozess nicht möglich ist. Die wichtigsten Stoffdaten sind die Scherviskosität, das pvT-Verhalten, die spezifische Wärmekapazität sowie die Wärmeleitfähigkeit.

Motiviert durch die Anwendung Bipolarplatte, einer Wiederholkomponente von PEM-Brennstoffzellen, befasst sich diese Arbeit mit elektrisch leitfähigen Thermoplasten und dem Verarbeitungsprozess bzw. der Prozesssimulation. Von Interesse sind dabei die Konstruktion neuer Formteile unter Berücksichtigung der vollständigen Formteillfüllung und die Ausbildung guter elektrischer Leitfähigkeit.

In dieser Arbeit werden die Vor- und Nachteile verschiedener Modellierungsansätze beleuchtet sowie ein Ansatz für ein kalibrierbares Materialmodell vorgestellt. Unter einem kalibrierbaren Modell wird ein Ansatz verstanden, der auf einen bestimmten Zielbereich trainiert werden kann. Damit ist bspw. die Art und Geometrie des Formteils z.B. über die charakteristische Formteillwandstärke oder für bestimmte Zykluseinstellungen gemeint. Zum Abgleich dienen Kenngrößen aus Messungen an einem Spritzgießprozess. Im Fokus stehen hierbei der Einspritz- und der Werkzeuginnendruckverlauf.

1 Einleitung

1.1 Elektrisch leitfähige Thermoplaste

Die elektrische Leitfähigkeit verhilft Kunststoffen zu neuen und wirtschaftlich interessanten Eigenschaften. Für die Ausbildung elektrischer Leitfähigkeit gibt es verschiedene Ansätze. Auf der einen Seite gibt es intrinsisch leitfähige Kunststoffe, die ohne Hinzugabe von Füllstoffen sehr gute elektrische Eigenschaften aufweisen, jedoch noch viele Schwachstellen wie bspw. die fehlende thermoplastische Verarbeitungsmöglichkeit aufweisen [6]. Aus diesem Grund liegt in dieser Arbeit der Fokus bei den gefüllten Thermoplasten, die im Compoundierprozess hergestellt werden. Hochgefüllte Thermoplast-Compounds sind i.d.R. binäre oder ternäre Compounds, basierend auf einem thermoplastischen Kunststoff. Anhand der Verarbeitungstemperatur wird zusätzlich zwischen Niedertemperatur-Thermoplasten (NT) und Hochtemperatur-Thermoplasten unterschieden (HT) [10]. Häufig eingesetzte Thermoplaste sind z.B. Polypropylen (PP) für den NT-Bereich und Polyphenylensulfid (PPS) für den HT-Bereich. In binären Systemen ist die zweite Komponente, der sogenannte Füllstoff, in der Regel ein elektrisch leitfähiges Material wie Kupfer, Graphit oder Carbon Nanotubes (CNT).

Bei ternären Systemen werden zusätzliche Füllstoffe eingesetzt, um die Perkolationsschwelle zu reduzieren. Diese Schwelle beschreibt einen starken Anstieg der elektrischen Leitfähigkeit ab einer bestimmten Füllstoffkonzentration. In Bereichen niedriger Füllstoffkonzentrationen dominiert der isolierende Kunststoff. Erst ab einem Grenzwert, der Perkolationsschwelle, steigt die elektrische Leitfähigkeit drastisch an. In diesem Fall liegen die eingebrachten Füllstoffe nahe genug beisammen, sodass sie eine leitfähige Netzwerkstruktur ausbilden können. Ein weiterer Anstieg der Füllstoffkonzentration bewirkt nach Erreichen des Grenzwertes nur noch einen moderaten Anstieg der elektrischen Leitfähigkeit. Das Perkulationsverhalten ist von vielen Faktoren abhängig, weshalb die Bestimmung des Grenzwertes eine wissenschaftliche Herausforderung darstellt [2], [11]. Detailinformationen zu den thermodynamischen Eigenschaften hochgefüllter und elektrisch leitfähiger Thermoplaste können aus Vorveröffentlichungen entnommen werden [7].

1.2 Spritzgießsimulation für funktionelle Kunststoffe

Die Spritzgießsimulation, insbesondere bei der Verwendung funktioneller Kunststoffe, ist charakterisiert durch eine erschwerte Bestimmung der Stoffdaten. Fehlerpotential birgt dabei bspw. die Messung der Scherviskosität mittels Hochdruckkapillarrheometrie, da der üblicherweise radial im Messzylinder eingesetzte Sensor mit hochviskosen Kunststoffen nur schwer erreichbar ist. Eine Alternativlösung stellt dabei eine Kraftmessdose in der Stempelachse dar, ist jedoch auch von Messfehlern nicht befreit. Bei der Kraftmessode wird die Reibung mitgemessen, die das Messergebnis verfälschen kann.

Die Ermittlung des pvT-Verhaltens und der spezifischen Wärmekapazität stellen ebenfalls hohe Anforderungen hinsichtlich der Datenbereitstellung dar. Selbst bei der Nutzung

identischer Abkühlraten ergeben sich Differenzen in der gemessenen Kristallisationstemperatur. Diese Differenz wird u.a. durch die deutlich abweichenden Probenmassen hervorgerufen und ist dem jeweiligen Messprinzip geschuldet.

Neben den Stoffdaten ist auch die Bestimmung der Prozessdaten wie Druck und Temperatur nicht trivial. Die Standardausrüstung einer Spritzgießmaschine verfügt nicht über die Möglichkeit, die wahre Schmelzetemperatur zu ermitteln. Lediglich die Heizbandtemperatur oder eine Oberflächentemperatur kann mittels Temperaturfühler ermittelt werden.

Fortlaufend sind Kenntnisse über die Temperatur in den Strömungskanälen (Angussbuchse und Kavität) nur mit zusätzlicher Messtechnik zu erreichen. Darüber hinaus muss von einer über den Durchmesser inhomogenen Temperatur der Schmelze ausgegangen werden [1], [12]. Die bislang genannten hohen Ansprüche an die Messaufgaben für die Simulation sind nur Beispiele und stellen keine vollständige Auflistung der anspruchsvollen Simulationsvorbereitungen dar. Von *Chen et al.* wird die Relevanz des Wärmeübergangskoeffizienten beschrieben [4]. Man kommt zu dem Ergebnis, dass dieser Koeffizient kein statischer Standardwert, sondern ein abhängiger, dynamischer Wert ist und als dieser in der Simulation berücksichtigt werden muss. Die größten Abhängigkeiten bestehen bzgl. der Einspritzgeschwindigkeit und der charakteristischen Formteildicke.

In Simulationen mit flächigen Kupferfüllstoffen beschreiben *Heinle et al.* außerdem eine Quell- und Dehnströmung im Kern des Formteils wodurch sich die Füllstoffe quer zur Fließrichtung ausrichten und aufgrund vorhandener Scher- und Dehnströmung im Randbereich eine Ausrichtung in Strömungsrichtung stattfindet. Diese anisotropen Werkstoffeigenschaften sind für die Spritzgießsimulation ebenfalls eine besondere Herausforderung [5], [8], [13]. An Mikroskopieaufnahmen entlang der Strömungsrichtung konnte am abgekühlten Formteil ein parabelförmiges Profil im Kern festgestellt werden. Dies lässt auf eine ausgeprägte Kernströmung während der Füllphase schließen (Abbildung 1).

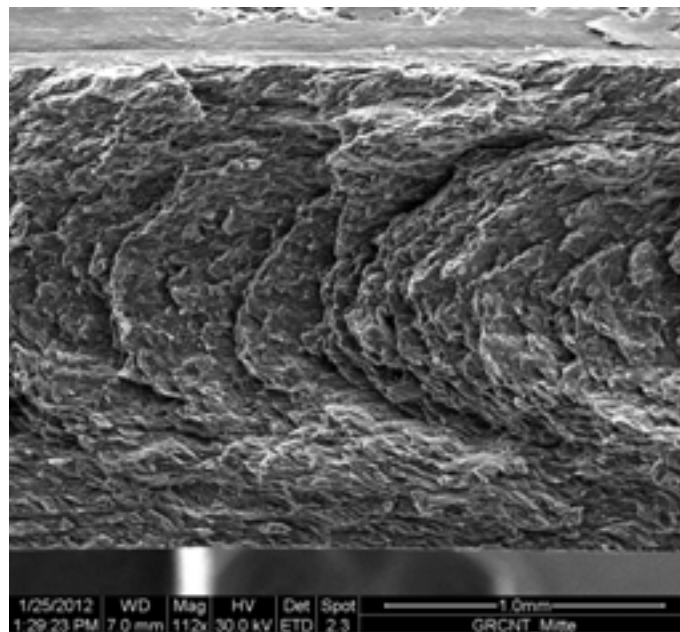


Abbildung 1: REM-Aufnahmen eines 2 mm Versuchsformteils (Strömungsrichtung von rechts nach links)

Strategien für die integrative Simulation werden am IKV und bei BASF entwickelt. Der Fokus liegt dabei bei auf einer möglichst exakten Bestimmung der mechanischen Eigenschaften unter Berücksichtigung von Mikrostrukturen, der Anisotropie und weiterer Details [9], [15].

1.3 Ziele

Langfristiges Ziel der Spritzgießsimulation ist die Prognose von Bauteileigenschaften, auch die der elektrischen Eigenschaften funktionaler Kunststoffe. Um diesem Ziel näher zu kommen sind jedoch einige Stufen vorher zu validieren. Eine qualitativ hochwertige Abbildung des Einspritz-, Nachdruck- und Abkühlprozesses ist bislang in der einschlägigen Literatur nicht publiziert. Der Lösungsweg besteht in einem hybriden, kalibrierbaren Modellansatz und dieser soll in den folgenden Kapiteln näher erläutert werden.

2 Modellbildung

Zur modellhaften Abbildung des Materialverhaltens für den Spritzgießprozess können unterschiedliche Ansätze konstruiert werden. Für ein rein physikalisches Modell werden bestehende Prozesse in hohem Detaillierungsgrad nachgebildet und dienen dem Materialmodell als Vorlage. Für den Spritzgießprozess bedeutet dies jedoch einen sehr hohen Aufwand bzgl. Zeit und Kosten, worunter auch nennenswerte Investitionen zählen. Die Charakterisierung der Stoffdaten wurde bereits kurz beschrieben. Zusätzliche Sensorik an der Maschine und im Werkzeug sind zur exakten Temperaturbestimmung und zur Abbildung anderer Größen wie z.B. dem Druckverlauf notwendig. Bei der physikalischen Beschreibung muss ebenfalls auf die prozessrealistische Stoffdatenermittlung geachtet werden. Die für Standardmessungen vorgegebenen Abkühlraten oder Messtoleranzen sind aufgrund der Materialunterschiede gegenüber ungefüllten Kunststoffen ungeeignet. Hochgefüllte und elektrisch leitfähige Thermoplaste erreichen im Spritzgießwerkzeug Abkühlraten oberhalb 50 K/s [3].

Mit herkömmlichen DSC-Messgeräten sind Abkühlraten dieser Größenordnung nicht erreichbar. Neuentwicklungen im Bereich der DSC-Messverfahren (heute als Sonderverfahren zu bezeichnen) ermöglichen mittlerweile zwar enorme Aufheiz- und Abkühlraten, sind aufgrund der Probenmasse jedoch nicht mehr prozessrealistisch. Gelingt eine vollständige Beschreibung, erreicht man einen hohen Gültigkeitsbereich hinsichtlich Prozessveränderungen wie Schmelze- bzw. Werkzeugtemperatur oder auch bei einem Werkzeugwechsel. Eine Allgemeingültigkeit ist jedoch nicht gegeben. Der Ermittlungsaufwand der Messwerte muss für jede Materialänderung wie z.B. der Austausch einen Füllstoffes (Ruß zu CNT) oder eine Änderung des Füllstoffanteils wiederholt werden. Abbildung 2 beschreibt diese Vorgehensweise schematisch.

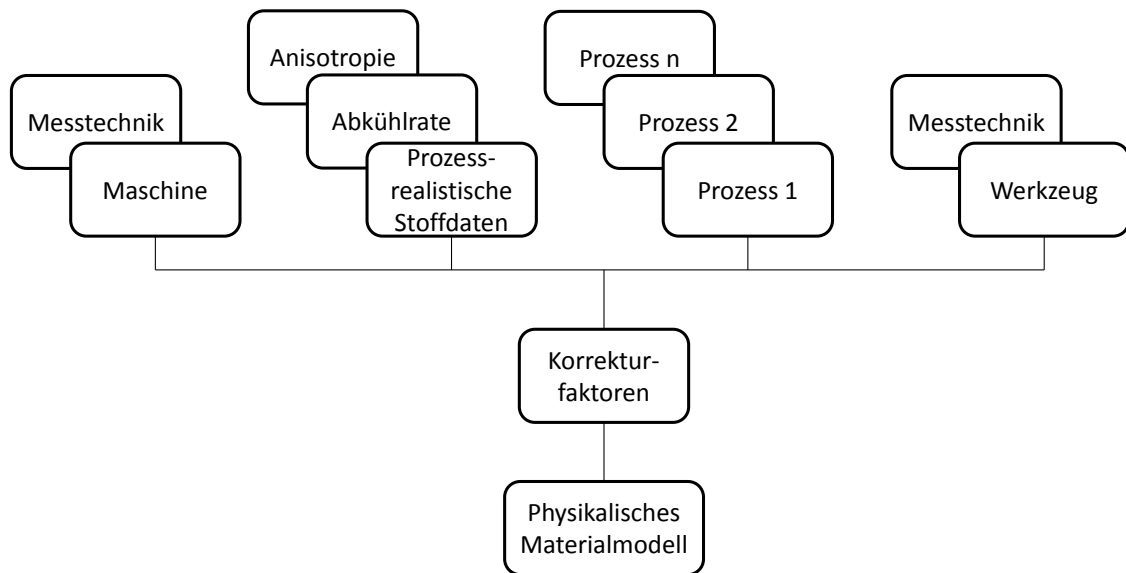


Abbildung 2: Schematischer Ablauf der rein physikalischen Materialmodellbildung

Im Folgenden soll nun ein Modellansatz beschrieben werden, der die Vorteile eines physikalischen Modells hervorhebt und die Nachteile durch eine intelligente Simulationsroutine behebt. Abbildung 3 zeigt schematisch die Zusammenhänge der kalibrierbaren Materialmodellbildung.

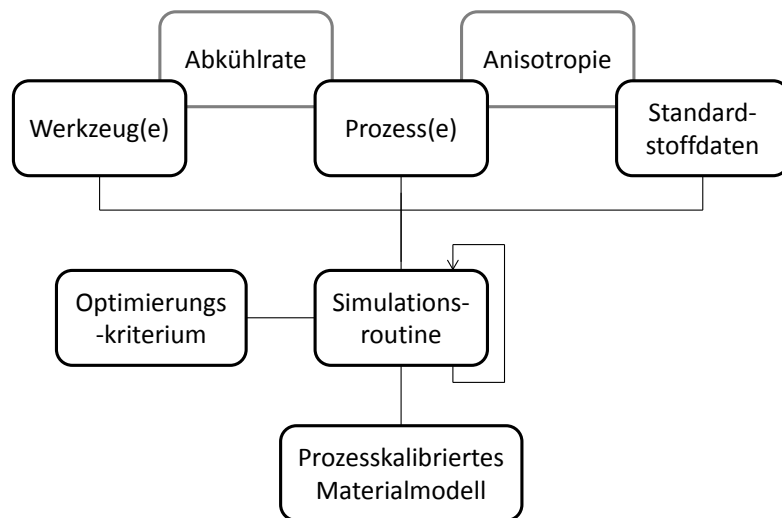


Abbildung 3: Schematischer Ablauf der kalibrierbaren Materialmodellbildung

Das in dieser Arbeit aufgebaute kalibrierbare Materialmodell benötigt als Input Daten zur Geometrie (Werkzeug, Kavität), vom Spritzgießprozess (z.B. Einspritz- und Werkzeuginnendruck) sowie Stoffdaten, die größtenteils lediglich unter Berücksichtigung des Abkühlverhaltens charakterisiert werden müssen. Anschließend werden diese Daten für die Simulationssoftware aufbereitet und in typische Modelle überführt. Für die Beschreibung der Scherviskosität kann hier bspw. der Carreau- oder Cross-Arrhenius-Ansatz gewählt werden. Andere Stoffdaten wie z.B. die spez. Wärmekapazität und die Wärmeleitfähigkeit können tabellarisch implementiert werden. Für die Beschreibung des pvT-Verhaltens wird auf das Modified-Tait-Modell zurückgegriffen [14].

Das Pre-Processing ist somit ausreichend beschrieben und es können Berechnungen erfolgen. Zur Ergebnisoptimierung wird nun mit Materialvariationen gearbeitet, um das Zielkriterium zu erreichen. Dieses Kriterium kann bspw. der Einspritz- und der Werkzeuginnendruck sein. Als freie Variablen dienen hierbei Material- und Prozessparameter, die im Vorfeld bei der Bestimmung größtenteils als unsichere Messwerte angesehen wurden. Darunter fallen die Viskosität, die durch die Referenztemperatur verändert werden kann und u.a. die Prozessdaten Schmelze- und Werkzeugtemperatur. Beide Temperaturen sind schwierig zu ermitteln, weshalb diese als freie Parameter in abgesteckten Grenzen variiert werden. Insgesamt werden fünf Parameter als freie Variablen innerhalb physikalisch plausibler Grenzen variiert.

Die Kalibrierparameter sind die Bezugstemperatur T_b zur Anpassung der Viskosität, die Kristallisationstemperatur des spezifischen Volumens und der spezifischen Wärmekapazität, die Wärmeleitfähigkeit und der Wärmeübergangskoeffizient. Die Kristallisationstemperatur wurde aufgrund von Messabweichungen zwischen pvT und spezifischer Wärmekapazität aufeinander abgestimmt.

3 Referenzprozess

Zum Abgleich mit der Simulation muss ein Referenzprozess herangezogen werden. Die Einstellungen des Referenzprozesses werden in Tabelle 1 dargestellt. Auffällig gegenüber konventionellen Thermoplasten ist die hohe Schmelze- und Werkzeugtemperatur bei der Verarbeitung eines Polypropylens. Zusätzlich müssen erhebliche Fließwiderstände mit hohen Drücken (Einspritz-, Umschalt- und Nachdruck) überwunden werden, damit die Schmelze nicht während des Fließens einfriert. Der Staudruck ist relativ niedrig eingestellt, da je nach Schneckendesign Temperaturprobleme im Plastifizierzylinder auftreten können.

Prozessparameter	Größe
Max. Einspritzdruck	2500 bar
Nachdruck (Soll)	2200 bar
Schmelzetemperatur	320 °C
Werkzeugtemperatur	130 °C
Plastifizierdrehzahl	80 1/min
Einspritzzeit	0,45 s
Nachdruckzeit	5,00 s
Umschaltdruck	2200 bar
Staudruck	15 bar
Max. Einspritzgeschwindigkeit	160 mm/s

Tabelle 1: Zyklusdaten des Referenz-Spritzgießprozesses

Zum besseren Verständnis werden in Tabelle 2 Informationen zum verwendeten Material angegeben. Der Gesamtfüllgrad ist als sehr hoch einzuschätzen. Steigerungen des Gesamtfüllgrades sind bei der Verwendung der dargestellten Füllstoffe nur noch bedingt möglich. Die Füllstoffe sind beide elektrisch leitfähig und werden zu unterschiedlichen Anteilen im Compoundierprozess eingearbeitet. Ruß ist im Vergleich zum Graphit deutlich kleiner, wodurch eine Erhöhung der elektrisch leitfähigen Strukturen im Material gewonnen werden kann. Die Zykluszeit betrug 88 s, wovon 60 s Kühlzeit als größter Bestandteil einzurechnen sind. Auch wenn das Compound, aufgrund der hohen elektrischen wie auch Wärmeleitfähigkeit, sehr schnell abkühlt, wurde zur besseren Entformbarkeit eine längere Kühlzeit gewählt. Der Spritzgießprozess wurde an einer elektrischen Standard-Spritzgießmaschine durchgeführt.

Gesamtfüllgrad	> 80 % gew. / > 60 % vol.
Matrix	Leichtfließendes Polypropylen
Füllstoff 1	Synthetischer Graphit
Füllstoff 2	Leitfähigkeitsruß

Tabelle 2: Materialdaten des betrachteten Compounds

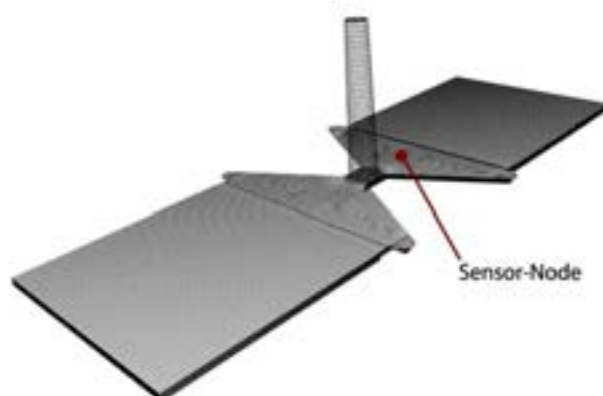


Abbildung 4: Verwendetes Formteil bzw. verwendete Simulationsgeometrie

Zur Vollständigkeit zeigt Abbildung 4 das verwendete Werkzeug und demnach auch die Simulationsgeometrie mit Darstellung der Drucksensorposition. Die Wandstärke der beiden Quader beträgt 2 mm ($l \times b = 60 \text{ mm} \times 60 \text{ mm}$).

4 Spritzgießsimulation

In diesem Kapitel werden die Resultate der Simulationsberechnungen dargestellt. Zunächst wird das Ergebnis unter reiner Verwendung der Messdaten präsentiert und im weiteren Verlauf die Effekte aufgrund der Parameterveränderungen.

4.1 Ohne Kalibrierung

Ohne weitere Anpassungen kritischer Material- und Prozessdaten ist es nicht möglich den Werkzeuginnendruck korrekt abzubilden. Abbildung 5 zeigt anhand der roten Kurven die Diskrepanz zwischen Simulation und Messung.

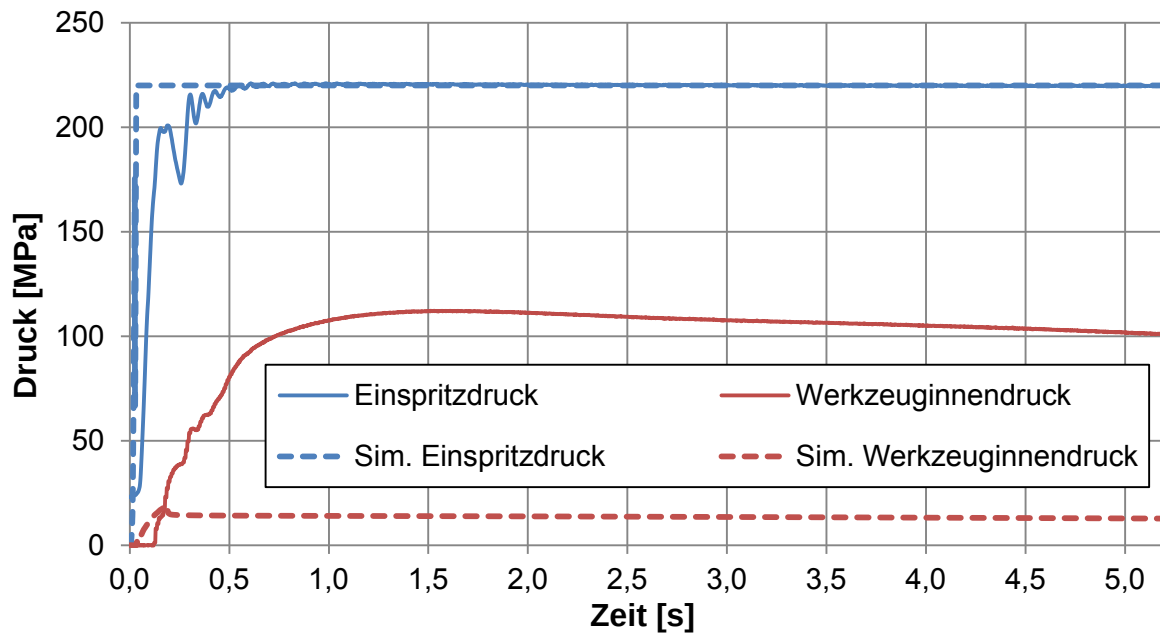


Abbildung 5: Diskrepanz Standardsimulation und Prozess

Abweichungen treten in der Druckhöhe und auch im qualitativen Verlauf auf. Um das Materialmodell hinsichtlich einer besseren Beschreibung des Werkzeuginnendruckes zu verändern werden in den folgenden Kapiteln gezielte Parameter verändert um deren Wirkung auf den Werkzeuginnendruck zu bestimmen. Erst im Anschluss kann eine Optimierung erfolgen.

4.2 Effektstudien

Exemplarisch für die durchgeführten Effektstudien werden in den Abbildungen 6 und 7 zwei Materialparameter näher beleuchtet. In Abbildung 6 ist der simulierte Werkzeuginnendruck abhängig von der Bezugstemperatur T_b (Parameter der Viskosität) in drei Stufen abgebildet (- = gering, o = mittel, + = hoch). Bei Erhöhung der Bezugstemperatur nimmt der Werkzeuginnendruck tendenziell ab. Die Reduktion resultiert aus einer erhöhten Viskosität des Materials und einem dadurch bedingten größeren Druckverlust zwischen Einspritz- und Werkzeuginnendruck.

Die drei dargestellten Kurven sind abhängig von weiteren Einstellungen, wodurch die Kurven weiter nach oben oder unten verschoben werden können. Durch eine höhere Schmelze- oder Werkzeugtemperatur verschieben sich die Kurven gemeinsam zu höheren Druckwerten.

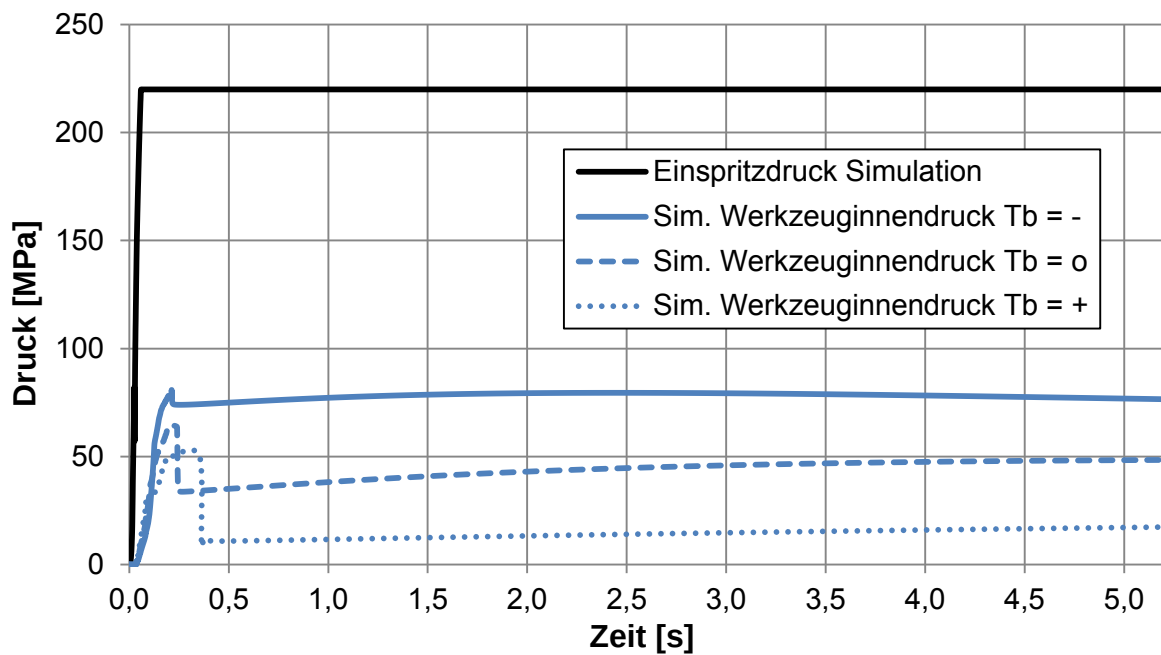


Abbildung 6: Einflussfaktor Bezugstemperatur der Viskosität

Abbildung 7 zeigt separat den Effekt der Wärmeleitfähigkeit des Materialmodells. Durch die Referenztemperatur der Viskosität konnte größtenteils Einfluss auf die vertikale Lage der Kurve genommen werden. Durch die Wärmeleitfähigkeit lässt sich auch Einfluss auf die Charakteristik des Werkzeuginnendruckverlaufs nehmen. Die Kurven zeigen unterschiedlich große Druckverluste während der Nachdruckphase bei Variation der Wärmeleitfähigkeit. Mit Erhöhung der Wärmeleitfähigkeit erhöht sich der Druckverlust woraus ein geringerer Werkzeuginnendruck resultiert.

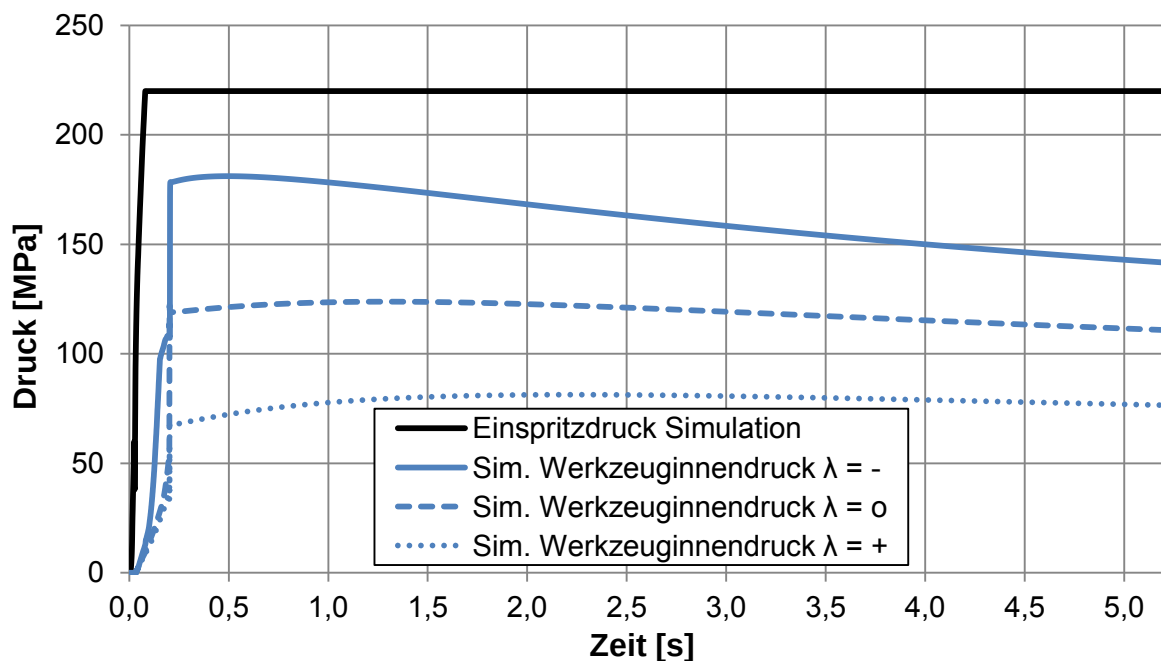


Abbildung 7: Einflussfaktor Wärmeleitfähigkeit

4.3 Nach Kalibrierung

Nach Ermittlung der einzelnen Wirkungen der untersuchten Parameter und deren Wechselwirkungen ergibt sich die Möglichkeit den Prozess optimiert abzubilden. Abbildung 8 zeigt das beste Resultat zur Beschreibung der Nachdruckphase. Die vorhandene Diskrepanz zwischen 0,25 s und 1,00 s ist auf die Kompressibilität des Materials im realen Versuch zurückzuführen. Die Simulationen wurden inkompressibel berechnet, weshalb ein steiler Anstieg in der Einspritzphase und ein schroffer Übergang am Umschaltzeitpunkt resultieren.

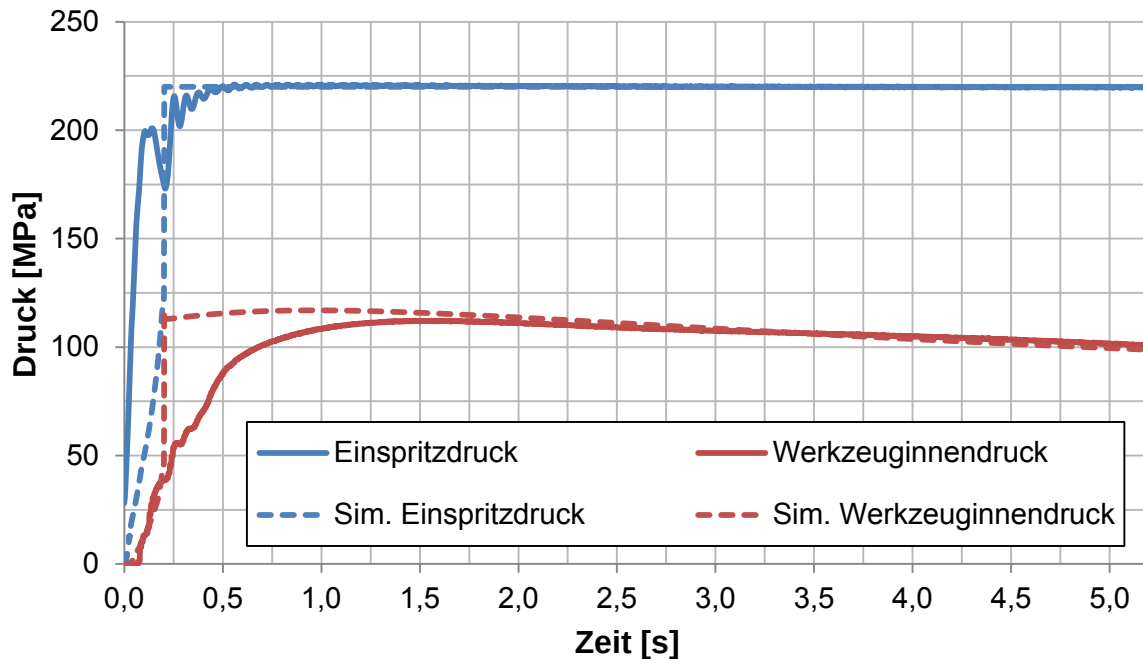


Abbildung 8: Resultat aus optimiertem Materialmodell

5 Fazit

Eine Simulation anhand der gemessenen thermodynamischen Stoffdaten und der Prozessdaten allein ist nicht ausreichend zur Beschreibung der Formteilverfüllung und der Darstellung des Werkzeuginnendruckes. Durch geschickte, manuelle Parametervariationen ist es aktuell möglich mit einer begrenzten Anzahl freier Parameter die Lösung schrittweise zu verbessern. Die zeitliche Abbildung der Einspritzphase bietet noch Verbesserungspotential, die Abbildung des Nachdruckverlaufes hingegen ist mit dieser Methodik sehr gut abbildbar.

6 Ausblick

Um einen größeren Gültigkeitsbereich zu erreichen, soll das Materialmodell anhand weiterer Formteile und Prozesse kalibriert werden. Dafür kann z.B. ein Werkzeug mit größerer Wandstärke oder eine abweichende Schmelztemperatur gewählt werden. Die Erweiterungen werden anhand sinnvoll und realistisch gewählter Parameteränderungen durchgeführt. Aufgrund der tatsächlich kleinen Verarbeitungsfenster von hochgefüllten Thermoplasten reicht bspw. eine Kalibrierung der Schmelztemperatur zwischen 320 °C und 340 °C aus.

Auch die für dieses Material in Frage kommenden Werkzeuge halten sich in engen Grenzen, weshalb keine Allgemeingültigkeit erforderlich ist.

Weitere Arbeiten werden durch eine Überarbeitung der Simulationsroutine zu einer Verringerung der Kalibrierzeiten führen. Dazu sollen bspw. Automatisierungsstrategien eingesetzt.

7 Anmerkungen

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 03X0048C gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor. Die Autoren bedanken sich für die Unterstützung im Projekt CarboPlate bei der Bayer Technology Services GmbH, Clariant Masterbatches GmbH, Evonik Degussa GmbH und dem Zentrum für BrennstoffzellenTechnik GmbH.



Literatur

- [1] Amberg, J.; Guttman, M. et al.: *Experimentelle Bestimmung thermischer Material- und Prozessdaten für die Spritzgießsimulation unter Verbesserung der Messtechnik*, Abschlussbericht AiF-Projekt 12544 N, 2003
- [2] Amesöder, S.: *Wärmeleitende Kunststoffe für das Spritzgießen*, Dissertation, Universität Erlangen-Nürnberg, 2009
- [3] Bosse, M.: *Darstellung der kunststofftechnischen Verarbeitung und funktionellen Leistungsfähigkeit weichmagnetisch hoch gefüllter Thermoplaste*, Dissertation, TU Clausthal, 2005
- [4] Chen, S. C.; Lin, Y. C. et al.: *Simulating the flow length to thickness ratio in ultra high-speed injection molding by moldex3D analysis*, ANTEC 2010, Orlando, Florida, USA, S. 1345-1348, 2010
- [5] Chen, S. C.; Shih, M. Y. et al.: *Effects of molding conditions on the conductivity properties of injection molded bipolar plate used for fuel cell*, ANTEC 2006, Charlotte, North Carolina, USA, S. 1168-1171, 2006

- [6] Chiang, C. K.; Park, Y. W. et al.: Conducting polymers: Halogen doped polyacetylene, J. Chem. Phys., H. 69, S. 5098-5104, 1978
- [7] Dörner, J.; Wortberg, J.: *Simulation des Formfüllvorgangs beim Spritzgießen elektrisch und thermisch leitfähiger Thermoplaste*, ASIM-Konferenz STS/GMMS 2011, Krefeld, S. 227-238, Shaker, 2011
- [8] Heinle, C.; Ehrenstein, G. W.: *Wärmeleitfähig modifizierte Kunststoffe – Teil1: experimentelle Analyse und Modellbeschreibung einer bauteil- und prozessabhängigen Materialeigenschaft*, Zeitschrift Kunststofftechnik / Journal of Plastics Technology 05/2009, Seite 338-358
- [9] Hopmann, C.; Arping, T. et al.: *Bauteileigenschaften präzise vorhersagen*, Kunststoffe, H. 07, S. 44-49, Hanser, 2011
- [10] Kreuz, C.: *PEM-Brennstoffzellen mit spritzgegossenen Bipolarplatten aus hochgefülltem Graphit-Compound*, Dissertation, Universität Duisburg-Essen, 2008
- [11] Lux, F.: *Review - Models proposed to explain the electrical conductivity of mixtures made of conductive and insulating materials*, Journal of Materials Science, H. 28, S. 285-301, 1993
- [12] Moneke, M.; Amberg, J.: *Temperatur- und orientierungsabhängige Stoffwerte für die Spritzgießsimulation*, Werkstoffprüfung – Konstruktion, Qualitätssicherung und Schadensanalyse, S. 315-320, 2004
- [13] Schmachtenberg, E.; Brandt, M. et al.: *Durchgängige CAE-Lösung: Faserverstärkung richtig simulieren*, Kunststoffe, H. 05, S. 94-99, 2004
- [14] Tait, P. G.: *Physics and Chemistry*, Vol. 2. Part IV, 1889
- [15] Wüst, A., Hensel, T. et al.: *Integrative Optimierung – Spritzgießsimulation als integraler Bestandteil numerischer Bauteiloptimierung*, VDI-Spritzgießen 2011, S. 197-206, 2011

Simulation hydraulischer Bremskrafterzeuger für Schienenfahrzeuge

Marco Bräunlich, Knorr-Bremse Systeme für Schienenfahrzeuge GmbH
braeunlich.marco@knorr-bremse.com

Zusammenfassung

Zur besseren Vorhersage des Verhaltens hydraulischer Bremssysteme wurde ein Teilmodell eines passiven Bremskrafterzeugers entwickelt und mit Versuchen mit realen Krafterzeugern abgeglichen. Das Modell wurde mit der Modellierungssprache Modelica geschrieben. Neben den Elementen der Standard-Bibliotheken wurden neue Modelle z.B. für Reibungskontakte implementiert. Im Ergebnis zeigt sich, dass die Simulation sehr gut Übereinstimmung mit dem Versuch zeigt.

1 Einführung

Um während einer frühen Phase der Entwicklung hydraulischer Bremssysteme für Schienenfahrzeuge das Verhalten der Systeme und Systemkomponenten unter unterschiedlichen Szenarien besser abschätzen zu können sollen verstärkt Simulationen zum Einsatz kommen. Im ersten Schritt wurden dazu Modelle der Bremskrafterzeuger entwickelt. Ziel war es dabei mit möglichst wenig unterschiedlichen Modellen eine große Anzahl verschiedener Krafterzeugervarianten zu simulieren. Das bedeutet, es waren gut parametrisierbare Systemmodelle zu entwickeln. Die Modelle sollen zunächst als Einzelkomponenten verwendbar sein aber auch in nachfolgenden Simulationen des gesamten Bremssystems eingesetzt werden können.

Zur Modellierung wurde Modelica verwendet, da diese Sprache es ermöglicht die im System vorhandenen physikalischen Bereiche Mechanik, Hydraulik und Elektronik miteinander zu verbinden.

2 Hydraulische Bremssysteme für Schienenfahrzeuge

Das Einsatzgebiet hydraulischer Bremssysteme ist vorwiegend im Bereich von Niederflurfahrzeugen im Nahverkehr (Straßenbahnen) zu finden. Dabei kommt dem, gegenüber pneumatischen Systemen, sehr geringer Platzbedarf der hydraulischen Bremsen große Bedeutung zu. Ein hydraulisches Bremssystem setzt sich aus den Komponenten Bremssteuergerät, Hydrogerät und bis zu vier Bremssätteln zusammen. Das Steuergerät ist eine elektronische Regeleinheit, welche die Regelung des Drucks mit Hilfe des Hydrogerätes

übernimmt. Im Hydrogerät selbst befindet sich die hydraulische Schaltung zur Betätigung der Bremssättel. Die Bremssättel beinhalten die, in diesem Beitrag behandelten Krafterzeuger. Krafterzeuger generieren die Kraft, welche die Bremsbeläge auf die Bremsscheibe drücken, und somit zur Verzögerung des Fahrzeugs führt.

Im Schienenfahrzeugbereich werden hydraulische Krafterzeuger in die zwei Gruppen „passive Bremskrafterzeuger“ und „aktive Bremskrafterzeuger“ gegliedert.

Um den Rahmen dieses Beitrags nicht zu überschreiten, wird im Folgenden nur auf das Modell eines passiven Krafterzeugers eingegangen.

2.1 Passive Bremskrafterzeuger

Bei der passiven Bremskrafterzeugung wird die Bremskraft über einen Federspeicher aufgebracht. Durch diesen Federspeicher, in der Regel eine Tellerfedersäule, wird der Bremskolben gegen den Bremsbelag gedrückt. Zum Lösen der Bremse wird Druck in einem Zylinder aufgebaut. Der Kolben des Zylinders drückt, in der Bremskraft entgegengesetzten Orientierung, auf den Federspeicher. Zur Verdeutlichung sind die idealisierte Kennlinie und eine Prinzipdarstellung eines passiven Bremskrafterzeugers in Abbildung 1 dargestellt.

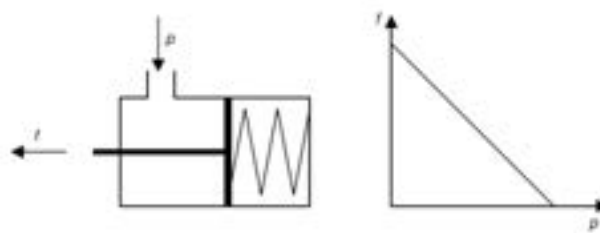


Abbildung 1: Prinzipdarstellung, idealisierte Kennlinie passiver Krafterzeuger.

3 Funktionsschema eines passiven Bremskrafterzeugers

In Abbildung 2 ist beispielhaft ein schematischer Aufbau eines passiven Bremskrafterzeugers der Knorr-Bremse zu sehen. Dieses Schema trifft auf eine Vielzahl von Krafterzeugern zu.

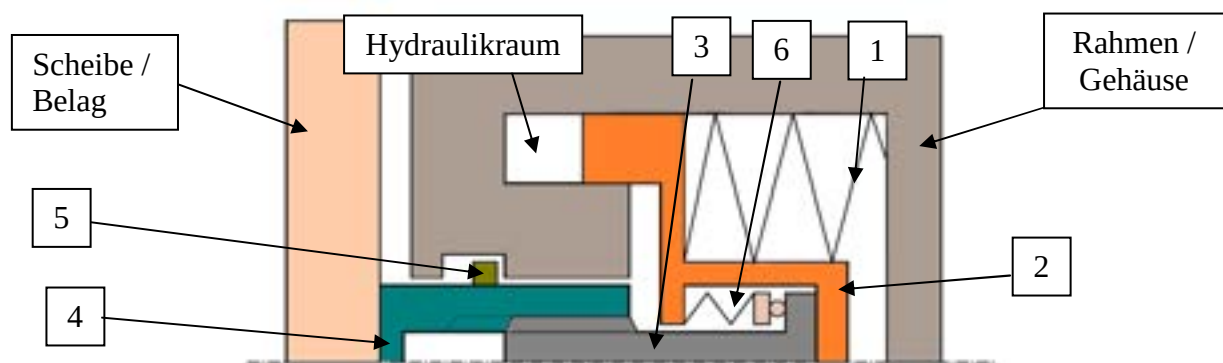


Abbildung 2: Schematischer Aufbau passiver Krafterzeuger (Halbschnitt).

Die Funktion des Krafterzeugers lässt sich wie folgt zusammenfassen. Ausgangspunkt ist der eingebremste Zustand. In diesem Zustand ist der hydraulische Druck sehr gering oder =0. Die gesamte Tellerfederkraft (1) wird über den Tellerfederkolben (2) auf die Spindel (3)

übertragen. Die Spindel leitet über ein Bewegungsgewinde die Kraft auf die Rohrmutter mit Druckstück (4), welches auf den Bremsbelag drückt, weiter. Eine Verzahnung im Kontaktbereich der Spindel mit dem Tellerfederkolben und eine verdrehsichere Lagerung der Rohrmutter (4) sorgen dafür, dass in diesem Zustand keine relative Rotation zwischen Rohrmutter und Spindel (Bewegungsgewinde) möglich ist. Auf der Rohrmutter ist eine Klemmfeder (5) platziert. Überschreitet der Hub der Rohrmutter den, durch einen festen Anschlag begrenzten Hub der Klemmfeder kommt es zu einer Verschiebung der Klemmfeder auf der Rohrmutter.

Zum Lösen der Bremse wird nun der hydraulische Druck erhöht. Dadurch wird der Tellerfederkolben (2) nach rechts verschoben und der Federspeicher (1) weiter gespannt. Über die Stellerfeder (6) wird die Spindel ebenfalls nach rechts bewegt. Die zusätzliche Feder sorgt dafür, dass die Kupplung zwischen Spindel und Kupplungshülse zunächst geschlossen bleibt. Erst wenn der Hub der Rohrmutter (4) nach rechts größer als der freie Hubweg der Klemmfeder (5) wird, bewirkt diese eine Gegenkraft. Dadurch wird die Feder (6) weiter gespannt und verkürzt sich. Somit löst sich die Kupplung und die Spindel kann sich aus der Rohrmutter heraus drehen, bis die Kupplung wieder geschlossen ist. Dadurch verlängert sich die Paarung Rohrmutter – Spindel. Kommt es beim Lösen der Bremse zu diesem Herausdrehen der Spindel, spricht man von der Verschleißnachstellung.

4 Modell des Krafterzeugers

Für die Modellierung wurde das System in die Teilsysteme Rahmen/Gehäuse, Hydraulikzylinder, Tellerfedersäule, Stellerfeder, Kupplung, Nachstellmechanismus, Klemmfeder und die Starrkörper Tellerfederkolben, Spindel und Rohrmutter gegliedert. Das vorwiegend mechanische Modell wurde als Mehrkörpersystem auf Basis der Modelica-Mechanics-Bibliothek aufgebaut. Das Gesamtmodell ist in Abbildung 3 zu sehen.

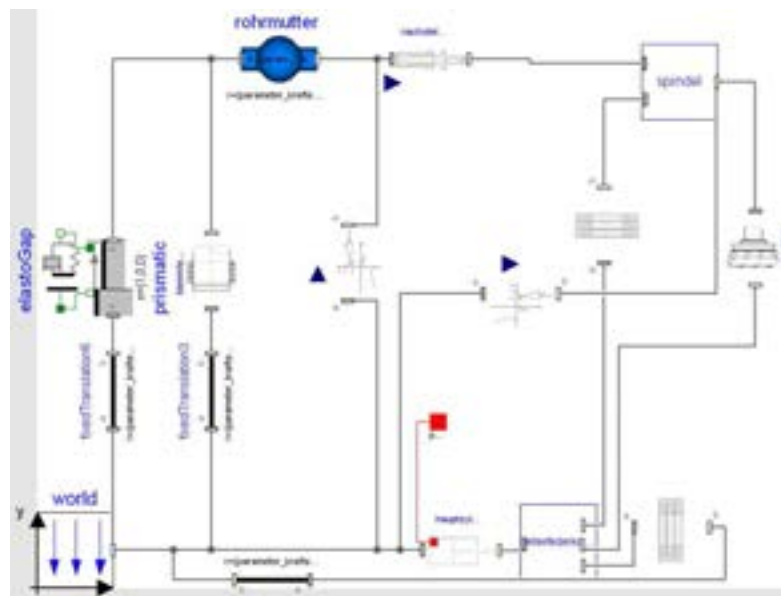


Abbildung 3: Gesamtmodell passiver Krafterzeuger.

4.1 Rahmen und Gehäuse

Der Rahmen bzw. das Gehäuse repräsentieren im betrachteten System alle nicht bewegten Komponenten und Stellen. Zudem beinhaltet es den Bremsbelag als elastischen Anschlag und die Reibungselemente zur Nachbildung der Reibung zwischen Rahmen und Spindel bzw. der Rohrmutter. Der elastische Anschlag dient der Modellierung der Steifigkeit des gesamten Bremssattels.

4.2 Hydraulikzylinder

Der Hydraulikzylinder setzt sich aus einem neu implementierten einseitig wirkenden Zylinder, einem Translationsgelenk zur Einschränkung der Bewegungsrichtung, zwei Anschlägen als Endlagenbegrenzung des Kolbens, einem Starrkörper (Kolben) und einem neu implementierten Reibmodell zusammen.

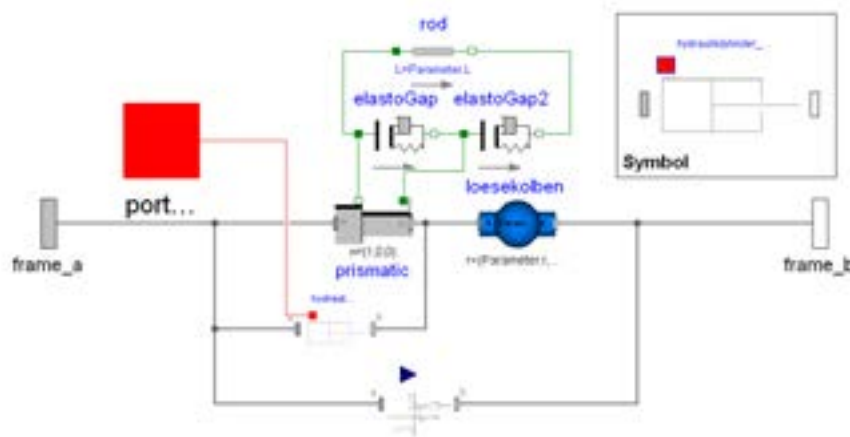


Abbildung 4: Sub model Hydraulikzylinder.

Der einseitig wirkende Zylinder wurde als Mehrkörperkomponente beschrieben. Als Basis wurde das Modell „PartialForces“ aus der Modelica-Bibliothek verwendet. Dieses Basismodell beinhaltet das Kraft- und Momentengleichgewicht des Modells und bestimmt den Vektor zwischen den zwei lokalen Koordinatensystemen in den Anschlusspunkten des Modells.

$$\underline{0} = \underline{f}_a + \underline{T}_a^T \times \underline{T}_b \times \underline{f}_b \quad (1)$$

$$\underline{0} = \underline{M}_a + \underline{T}_a^T \times \underline{T}_b \times (\underline{M}_b + \underline{r}_{relb} \times \underline{f}_b) \quad (2)$$

$$\underline{r}_{relb} = \underline{T}_b^T \times (\underline{r}_{b0} - \underline{r}_{a0}) \quad (3)$$

Neben den Gleichungen des Grundmodells wurden die Modellgleichungen zur Beschreibung des Verhaltens eines Zylinders wie folgt aufgestellt.

$$\frac{dV}{dt} = \frac{\dot{m}}{\rho} \quad (4)$$

$$V = A \times s \quad (5)$$

$$f = A \times p \quad (6)$$

Neben diesen Gleichungen sind noch einige Zuweisungen zur Komplettierung des Gleichungssystems notwendig.

A – Kolbenfläche

f – Kraft

$\underline{f}_a, \underline{f}_b$ – Kraftvektor in „frame_a“ bzw. „frame_b“

m – Masse

$\underline{M}_a, \underline{M}_b$ – Vektor der Momente in „frame_a“ bzw. „frame_b“

$\underline{r}_{relb}$ – Vektor von „frame_b“ zu „frame_a“ im KS von „frame_b“

s – Weg/Position

$\underline{T}_a, \underline{T}_b$ – Transformationsmatrix von „frame_a“ bzw. „frame_b“

V – Volumen

ρ – Dichte

4.3 Reibmodell

Das Standardreibmodell der Modelica Bibliothek bietet ausschließlich die Möglichkeit, geschwindigkeitsabhängige Reibkräfte ohne Bezug zur wirkenden Normalkraft zu simulieren. Mit Blick auf die Dichtungsreibung am Lösekolben genügt ein solches Modell jedoch nicht. Aus diesem Grund wurde ein neues Reibmodell entwickelt. Dieses Modell kann sowohl den Geschwindigkeitseinfluss als auch veränderliche Normalkräfte, durch z.B. Druckänderungen im Zylinder, abbilden. Grundlage ist das in [1] vorgestellte Modell zur Beschreibung der Reibkraft f_{fr} mit Hilfe einer Stribeck-Kurve ohne verzögertes Reibverhalten.

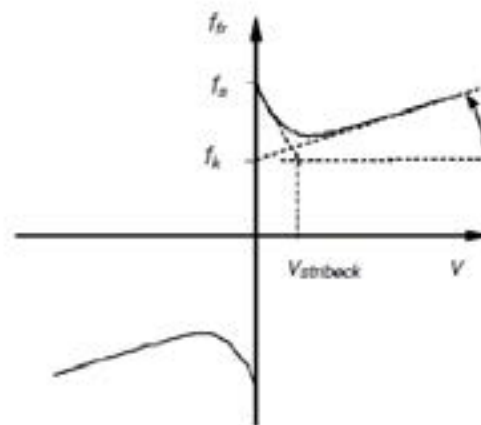


Abbildung 5: Stribeck-Reibkennlinie [1].

Als Basismodelle dient hierbei einerseits das bereits erwähnte „PartialForces“-Modell andererseits das „PartialFriction“-Modell der Modelica-Bibliothek. Das Modell „PartialFriction“ dient der Fallunterscheidung des Reibungszustandes und definiert die folgenden Zustände: Haften (locked), Start vorwärts (startForward), Start rückwärts (startBackward), Vorwärtsbewegung (Forward) und Rückwärtsbewegung (Backward).

Da das Reibmodell nach [1] außerhalb des Bereichs Haften einen kontinuierlichen Verlauf abbildet können einige Zustände zusammengefasst werden. Es ergibt sich das folgende Gleichungssystem.

$$f_{fr} = \begin{cases} f_{ext} & \text{Haften} \\ f_{Stribeck} (|v|) + b \times |v| & \text{für Start vorwärts} \vee \text{Vorwärtsbewegung} \\ -f_{Stribeck} (|v|) + b \times |v| & \text{Start rückwärts} \vee \text{Rückwärtsbewegung} \end{cases} \quad (7)$$

$$f_{Stribeck} = f_k + (f_s - f_k) \times e^{-\frac{v}{v_{Stribeck}}} \quad (8)$$

$$f_s = \mu_h \times f_n \quad (9)$$

$$f_k = \mu_g \times f_n \quad (10)$$

Unbekannt ist an dieser Stelle noch die Normalkraft. Die Normalkraft, die durch Dichtungen im Zylinder hervorgerufen wird, besteht aus einem konstanten Kraftanteil durch die Verpressung der Dichtung und aus einem variablen Anteil, der durch den Hydraulikdruck erzeugt wird. Eine hyperelastische Finite-Elemente-Analyse des Dichtungsbereichs zeigt, dass die Normalkraft in guter Näherung durch ein Polynom in Abhängigkeit des Drucks beschrieben werden kann.

$$f_n = a_1 + a_2 \times p + a_3 \times p^2 + a_4 \times p^3 + \dots + a_i \times p^{i+1} \quad (11)$$

Hinzu kommen einige Gleichungen zur Zuweisung der Konektorvariablen um das System zu vervollständigen.

- a_i – Polynomkoeffizienten
- b – Zähigkeitsfaktor für viskose Reibung
- f_{ext} – äußere Kraft
- f_{fr} – Reibkraft
- f_k – kinetische Reibkraft
- f_n – Normalkraft
- f_s – statische Reibkraft
- $f_{Stribeck}$ – Stribeckkraft
- p – Druck
- v – Geschwindigkeit
- $v_{Stribeck}$ – Stribeckgeschwindigkeit
- μ_g – Gleitreibkoeffizient
- μ_h – Haftreibkoeffizient

4.4 Tellerfedersäule und Stellerfeder

Zur Simulation einer Tellerfedersäule wurde ebenfalls ein neues Submodell implementiert. Dieses Modell beruht auf den in [2] beschriebenen Gleichungen zur Bestimmung der Federkraft.

$$f = \frac{4 \times E}{1 - \nu^2} \times \frac{t_d^4}{K_1 \times D_e^2} \times K_4^2 \times \frac{s_f}{t_d} \times \left[K_4^2 \times \left(\frac{h_0}{t_d} - \frac{s_f}{t_d} \right) \times \left(\frac{h_0}{t_d} - \frac{s_f}{2 \times t_d} \right) + 1 \right] \quad (12)$$

$$K_4 = \sqrt{-\frac{C_1}{2} + \sqrt{\left(\frac{C_1}{2}\right)^2 + C_2}} \quad (13)$$

$$C_1 = \frac{\left(\frac{t_1}{t_d}\right)^2}{\left(\frac{1}{4} \times \frac{h}{t_d} - \frac{t_1}{t_d} + \frac{3}{4}\right) \times \left(\frac{5}{8} \times \frac{h}{t_d} - \frac{t_1}{t_d} + \frac{3}{8}\right)} \quad (14)$$

$$C_2 = \frac{C_1}{\left(\frac{t_1}{t_d}\right)^3} \times \left[\frac{5}{32} \times \left(\frac{h}{t_d} - 1\right)^2 + 1 \right] \quad (15)$$

$$K_1 = \frac{1}{\pi} \times \frac{\left(\frac{\delta-1}{\delta}\right)^2}{\frac{\delta+1}{\delta-1} - \frac{2}{\ln \delta}} \quad (16)$$

$$s_f = s_{le} = \frac{s_{Säule}}{i} \quad (17)$$

Mit Hilfe der theoretischen Federkraft f kann die reibungsbehaftete Federkraft wie folgt bestimmt werden.

$$f_{gesR} = f \times \frac{n_f}{1 \mp \mu_M \times (n_f - 1) \mp \mu_R} \quad (18)$$

Da in den meisten Geräten ausschließlich Tellerfedersäulen und keine Pakete verwendet werden, kann der mittlere Summand des Nenners vernachlässigt werden. Zur Bestimmung des Vorzeichens des Reibkoeffizienten kann das Vorzeichen der Geschwindigkeit, mit der sich die Länge der Feder ändert verwendet werden. Hierbei kann die Signum-Funktion herangezogen werden. Aufgrund numerischer Probleme wurde diese Funktion durch tanh angenähert. Es ergibt sich ein kontinuierlicher Verlauf. Der Bereich um $v=0$ ist jedoch dadurch nicht exakt modelliert.

$$f_{gesR} = f \times \frac{1}{1 - \mu_R \times \tanh(v)} \quad (19)$$

Das erläuterte Modell wird sowohl für die Tellerfedersäule als auch für die Stellerfeder verwendet.

D_e	– Außendurchmesser der Tellerfeder
D_i	– Innendurchmesser der Tellerfeder
E	– E-Modul der Federn
f_{gesR}	– Gesamtkraft der Tellerfedersäule mit Reibungseinfluss
h	– Höhe der unbelasteten Feder
h_0	– Federweg bis zu Planlage der Feder
i	– Anzahl der Einzelfedern
$K_1/K_4/C_1/C_2$	– Berechnungsfaktoren
n_f	– Anzahl gleichsinnig geschichteter Federn

s_f	– Federweg
s_{le}	– Federweg einer Einzelfeder der Tellerfedersäule
$s_{Säule}$	– Federweg der Tellerfedersäule
t_d	– Tellerfederdicke
μ_M	– Reibkoeffizient für die Reibung zwischen den Mantelflächen gleichsinnig geschichteter Federn
μ_R	– Reibkoeffizient für die Reibung zwischen den Auflagestellen der Federn

4.5 Nachsteller

Der Nachsteller wird als Verbindung eines Translationsgelenks mit einem Rotationsgelenk nachgebildet. Der Zusammenhang zwischen Translation und Rotation lässt sich über das Gleichungssystem eines Gewindetriebs wie folgt beschreiben.

$$\omega_g = 2 \times \pi \times n_g \quad (20)$$

$$n_g = \frac{v}{P_g} \quad (21)$$

$$M = \frac{f \times d_m}{2} \times \tan(\xi + \beta_m) \quad (22)$$

$$\tan \xi = \frac{\mu}{\cos\left(\frac{\beta_f}{2}\right)} \quad (23)$$

$$\tan \beta_m = \frac{P_g}{\pi \times d_m} \quad (24)$$

$$l_{eges} = l_{e0} + l_e \quad (25)$$

Neben dem Modell des Gewindetriebs wurde ein Submodell zur Unterbindung der Rotation bei geschlossener Kupplung benötigt. Für dieses Teilmodell wurde auf die allgemeine Gleichung eines Rotationsschwingers zurückgegriffen.

$$I_r \times \alpha_r + b_r \times \omega + c_r \times \varphi = M \quad (26)$$

Hieraus ist jedoch nur der geschwindigkeitsabhängige Teil von Bedeutung.

$$B = b_r \times u \quad (27)$$

Durch eine große Dämpferkonstante kann ein Moment erzeugt werden, welches durch den Gewindetrieb nicht überwunden werden kann. Durch Hinzufügen der Steuergleichung

$$B \times \omega = M \quad (28)$$

kann das Gegenmoment an- und abgeschaltet werden. Als Eingangsgröße u wird dabei der Zustand der Kupplung verwendet.

- B – wirkende Dämpferkonstante
- b_r – Rotationsdämpferkonstante
- c_r – Rotationsfederkonstante

- d_m – mittlerer Gewindedurchmesser
- I_r – Massenträgheitsmoment bzgl. der Rotationsachse
- l_e – Einschraubtiefe
- l_{e0} – Einschraubtiefe zum Zeitpunkt $t=0$
- l_{eges} – Gesamteinschraubtiefe
- M – Moment
- n_g – Drehzahl
- P_g – Gewindesteigung
- u – Eingangssignal
- α_r – Winkelbeschleunigung
- β_f – Flankenwinkel des Gewindes
- β_m – Steigungswinkel des Gewindes
- φ – Drehwinkel
- ω_g – relative Winkelgeschwindigkeit zwischen Mutter und Spindel
- ω – Winkelgeschwindigkeit
- ζ – Reibwinkel
- μ – Reibkoeffizient für die Gewindereibung

4.7 Klemmfedermechanismus

Das Modell des Klemmfedermechanismus wurde, mit Ausnahme des Reibmodells, ausschließlich aus Standard-Komponenten der Modelica-Bibliothek zusammengesetzt. Es handelt sich dabei um einen Starrkörper, dessen Bewegungsrichtung durch ein Translationsgelenk eingeschränkt wird. Das Maß der möglichen Verschiebung wird dabei durch zwei Anschläge beschränkt. Die reibungsbehaftete Verbindung zur Rohrmutter wird mit Hilfe des bereits vorgestellten Reibmodells nachgebildet.

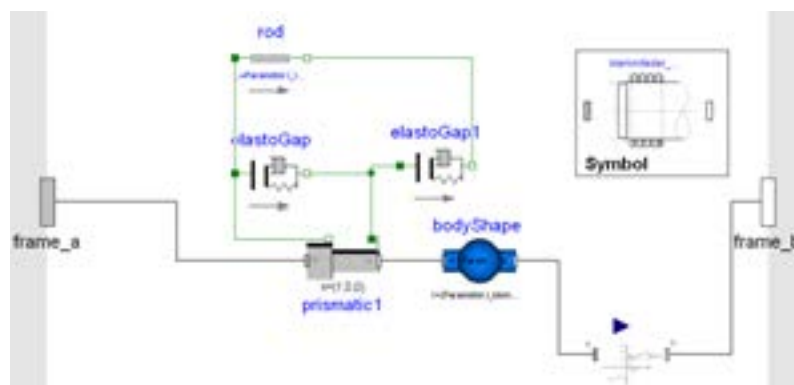


Abbildung 6: Sub model Klemmfedermechanismus.

5 Versuchsabgleich

Zur Verifikation des Modells wurden in einem Versuch die Größen Druck, Spindelweg und Druckplattenkraft aufgenommen. Als Eingangsgröße dient der Druck. Hierbei wurden der Druckaufbau und die Druckreduzierung mit unterschiedlichen Gradienten aufgenommen.

Die Bremsscheibe bzw. der Bremsbelag wurden im Versuch durch einen Ersatzkörper nachgebildet. Um einen definierten Zustand des Krafterzeugers in Hinblick auf die Lage der Klemmfeder zu erzeugen, wurde im Versuch zunächst das Nachstellen nachgebildet. Erst nach diesem ersten Bremszyklus wurden die Kennlinien in einem zweiten Bremszyklus aufgenommen.

Das Modell wurde entsprechend den Daten des im Versuch verwendeten Bremsstatts parametrisiert und als Eingangsgröße wurde der aufgenommene Druckverlauf an das Modell übergeben. Wie im Versuch auch wurde das Modell so initialisiert, dass es beim ersten Zyklus zum Nachstellen kommt, um im zweiten Bremszyklus die gleichen Zustand wie im Versuch zu garantieren. In Abbildung 7 ist der Druckverlauf, der als Eingangsgröße an das Modell übergeben wird, dargestellt.

Im Bereich bis 30s wurde ein idealisierter Druckverlauf generiert. In diesem Zeitraum findet der Nachstellvorgang statt. Ab $t=30s$ entspricht der Druckverlauf dem im Versuch aufgezeichneten Verlauf.

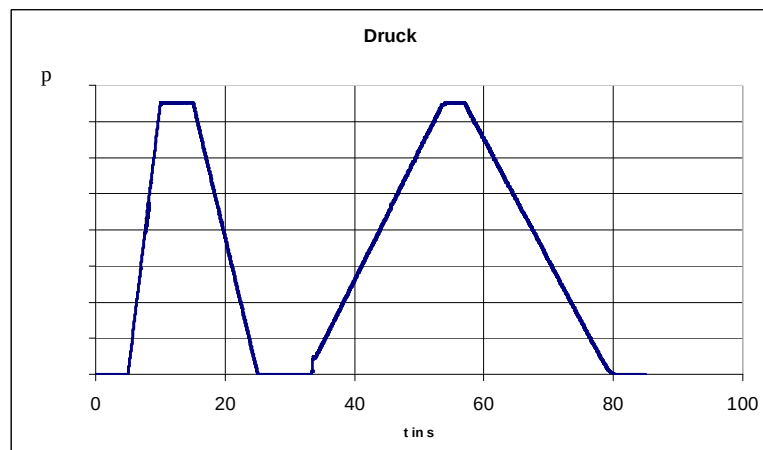


Abbildung 7: Druckverlauf Simulation.

Die folgenden drei Abbildungen zeigen den Vergleich zwischen Versuch und Simulation. Die ersten zwei Abbildungen zeigen die zeitlichen Verläufe der Kenngrößen Kraft und Weg. In der letzten Abbildung sind diese zwei Kenngrößen über den Druck dargestellt. In dieser Abbildung ist das Hystereseverhalten des Bremsstatts besonders gut zu erkennen.

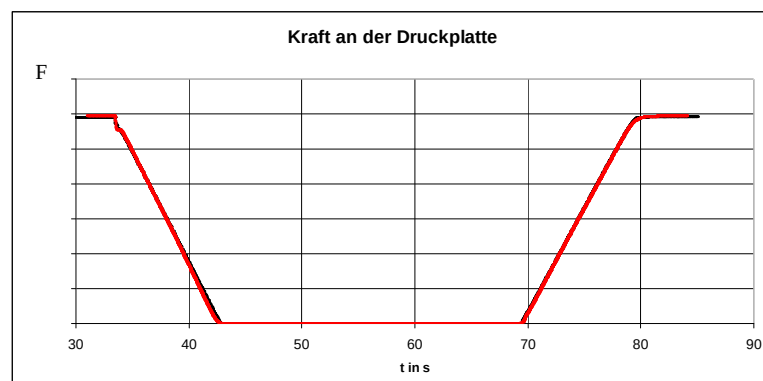


Abbildung 8: Zeitlicher Verlauf der Druckplattenkraft (rot: Versuch / schwarz: Simulation).

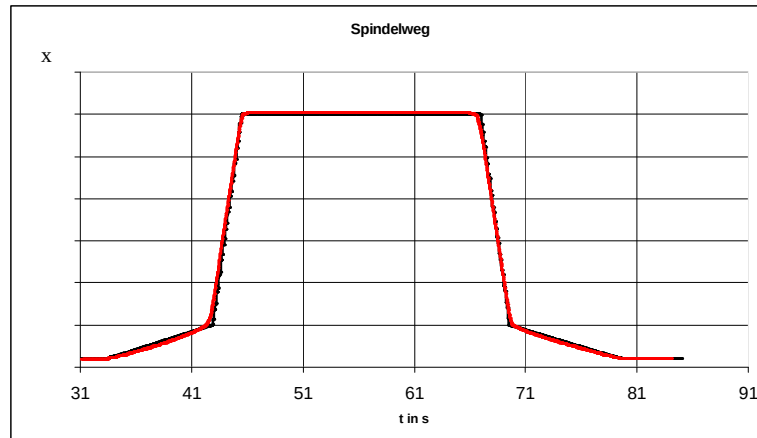


Abbildung 9: Zeitlicher Verlauf Spindelwegs (rot: Versuch / schwarz: Simulation)..

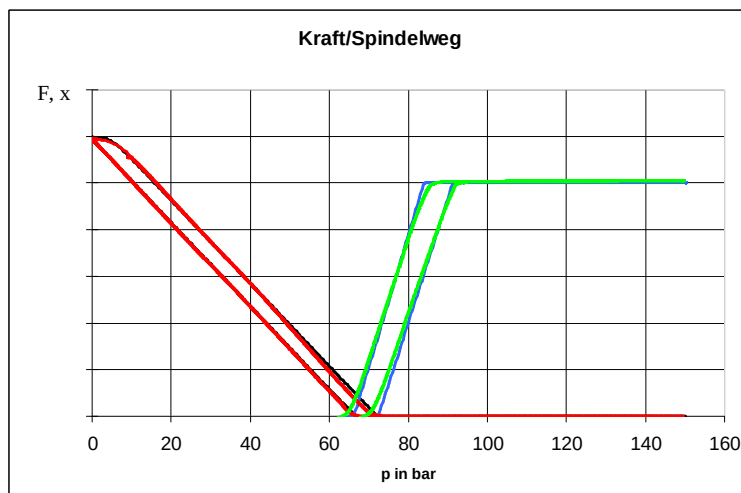


Abbildung 10: Kraft und Weg über Druck (Schwarz: Kraft Simulation / Rot: Kraft Versuch / Blau: Spindelweg Simulation / Grün: Spindelweg Versuch).

6 Fazit

Die simulierten Kennlinien zeigen sehr gute Übereinstimmungen mit den Daten aus dem realen Versuch. Kleinere Ungenauigkeiten zeigen sich in den „Knickpunkten“ der Kennlinien. Diese Unterschiede sind jedoch für ein Modell, welches in frühen Phasen des Entwicklungsprozesses als Auslegungstool verwendet werden soll, unbedeutend. Der Versuchsabgleich zeigt, dass das Modell gut als eigenständiges Element simulierbar ist. Erste Versuche mit vereinfachten Modellen des gesamten Bremssystems haben gezeigt, dass das Krafterzeugermodell auch im Systemzusammenhang gut verwendbar ist.

Zukünftig werden alle übrigen Baugruppen und Elemente des Bremssystems mit hohem Detailgrad modelliert. Ziel ist es, die hydraulische Bremseinrichtung einer kompletten Straßenbahn in verschiedenen Szenarien zu simulieren und so den Entwicklungsprozess zu unterstützen bzw. Bremssysteme zu optimieren.

Literatur

- [1] S. Klotzbach/ H. Henrichfreise, Ein nichtlineares Reibmodell für die numerische Simulation reibungsbehafteter mechatronischer Systeme, Gekürzte Fassung veröffentlicht zur ASIM 2002, Rostock 2002.
- [2] DIN 2092:2006-03

Aussteller



BAUSCH-GALL GmbH

bertrandt

cenit
DRIVEN BY YOUR VISION.

dSPACE



automotive
engineering **iaV**

Maplesoft
Mathematik • Modellierung • Simulation

tlk **TLK-Thermo GmbH**
Engineering Services and Software
for Thermal Systems

XRG
XRG Simulation GmbH

ISBN 978-3-901608-39-1