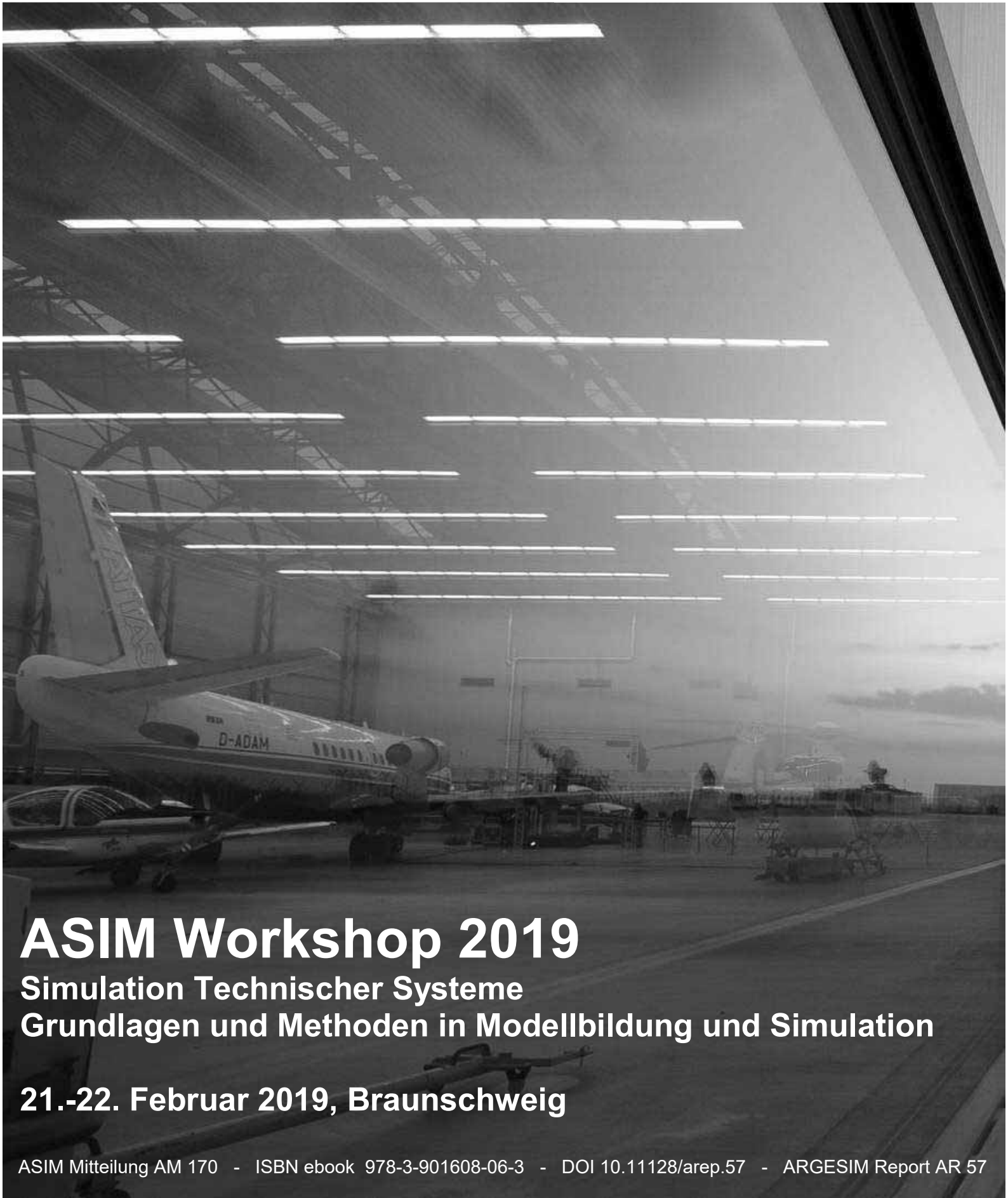




ASIM Arbeitsgemeinschaft
Simulation

**Deutsches Zentrum
für Luft- und Raumfahrt**



ASIM Workshop 2019

Simulation Technischer Systeme

Grundlagen und Methoden in Modellbildung und Simulation

21.-22. Februar 2019, Braunschweig

ASIM Mitteilung AM 170 - ISBN ebook 978-3-901608-06-3 - DOI 10.11128/arep.57 - ARGESIM Report AR 57

TAGUNGSBAND



ARGESIM

ARGESIM Reports

Published by **ARGESIM** and **ASIM**, Arbeitsgemeinschaft Simulation,
Fachausschuss Simulation in der GI - Gesellschaft für Informatik

Series Editors:

Felix Breitenecker (ARGESIM / ASIM)
Math. Modelling and Simulation Group,
TU Wien
Wiedner Hauptstrasse 8 - 10
1040 Vienna, Austria

Thorsten Pawletta (ASIM)
CEA - Computational Engineering and
Automation, HS Wismar
PF 1210
23952 Wismar, Germany

ARGESIM Report 57 ASIM Mitteilung AM 170

Titel: Tagungsband
ASIM Workshop 2019
Simulation Technischer Systeme
Grundlagen und Methoden in Modellbildung und Simulation

Herausgeber: Umut Durak
Christina Deatcu
Jan Hettwer
Email: umut.durak@dlr.de

ISBN ebook 978-3-901608-06-3

DOI 10.11128/arep.57

Das Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, der Entnahme von Abbildungen, der Funksendung, der Wiedergabe auf photomechanischem oder ähnlichem Weg und der Speicherung in Datenverarbeitungsanlagen bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten.

© by ARGESIM / ASIM, Wien, 2019

ARGE Simulation News (ARGESIM)
c/o F. Breitenecker, Math. Modelling and Simulation Group, TU Wien
Wiedner Hauptstrasse 8 - 10, A - 1040 Vienna, Austria
Tel: +43-1-58801-10115, Fax: +43-1-58801-910115
Email: info@argesim.org; WWW: www.argesim.org

Tagungsband

**ASIM Workshop 2019
Simulation Technischer Systeme
Grundlagen und Methoden in
Modellbildung und Simulation**

**21. bis 22. Februar 2019
DLR Braunschweig**

Zusammenfassung der Beiträge
Umut Durak (Hrsg.)
Christina Deatcu (Hrsg.)
Jan Hettwer (Hrsg.)



Arbeitsgemeinschaft Simulation ASIM in der Gesellschaft für Informatik GI

Tagungsleitung:

Umut Durak, Deutsches Zentrum für Luft-und Raumfahrt (DLR)

Programmkomitee:

Robert Buchta, Volkswagen

Walter Commerell, Hochschule Ulm

Christina Deatcu, Hochschule Wismar

Umut Durak, DLR Braunschweig

Leo Gall, LTX Simulation GmbH, München

Joachim Haase, Fraunhofer IIS/EAS Dresden

Andreas Körner, TU Wien

Xiaobo Liu-Henke, Ostfalia HAW

Daniel Lückerath, Fraunhofer IAIS

Heinz-Theo Mammen, Hella KGaA Hueck & Co., Lippstadt

Klaus Panreck, Fachhochschule Bielefeld

Thorsten Pawletta, Hochschule Wismar

Nikolas Popper, dwh GmbH, Wien

Michael Striebel, HTWG Konstanz

Siegfried Wassertheurer, AIT

Tagungsort:

DLR Braunschweig

Lilienthalplatz 7

38108 Braunschweig

**ASIM Workshop 2019
Simulation Technischer Systeme
Grundlagen und Methoden in
Modellbildung und Simulation
Tagungsband**

INHALT

INHALT

Editorial	1
Plenarvorträge	
Simulation in der Tiefbohrtechnik – Modellierung, Verifikation und Validierung <i>Gunther Brenner</i>	3
Kabinensimulation mit AVES - Herausforderungen und neue Möglichkeiten <i>Holger Duda</i>	5
Modellbasierte Entwicklung	
Cross-Layer Behavioral Modeling and Simulation of E/E-Architectures using PREEvision and Ptolemy II <i>Harald Bucher, Simon Kamm, Jürgen Becker</i>	7
Ein HiL-Prüfstand zur Funktionsabsicherung von Batteriemanagementsystemen mit Low-Cost Entwicklungsplattform <i>Sven Jacobitz, Xiaobo Liu-Henke</i>	13
Modellbasierte Entwicklung zukünftiger Avioniksysteme unter Verwendung von Scilab/Xcos (Kurzfassung) <i>David Müller, Umut Durak</i>	19
Simulation in der Energietechnik	
Python-Wrapper zur automatisierten Datenverarbeitung, Visualisierung und Simulation von Stromnetzen <i>Shuo Chen, Basem Idbi, David Stacic, Gerd Heilscher</i>	21
Simulation der Zwischenkreisregelung eines DC-gekoppelten Erzeuger-Speicher-Systems mit Schnelllademöglichkeit <i>Tobias Fricke, Günter Tareilus, Regine Mallwitz</i>	27
Charakterisierung von gestörten Geschwindigkeitsprofilen im in runden, rechteckigen und quadratischen Strömungsgeometrien <i>Konstantin Zacharias, Wolfgang Schlüter</i>	35
Maschinelles Lernen	
Realisierung einer Datenfusionsstruktur für die Umfeldperzeption autonomer Fahrzeuge <i>Marian Göllner, Xiaobo Liu-Henke</i>	41
Beschleunigung eines Reinforcement-Learning-Algorithmus durch Parallelverarbeitung für Robotikanwendungen <i>David Jammer, Sven Pawletta, Georg Kunert, Thorsten Pawletta</i>	49

Simulation thermischer Systeme

Simulative Untersuchung von Betriebserweiterungen in einem Aluminium- Schmelz- und Druckgussbetrieb anhand von Modellen mit unterschiedlichem Detaillierungsgrad	
<i>Johannes Dettelbacher, Wolfgang Schlüter</i>	53
Approach for synthesis and optimization of complex thermal systems for supermarkets	
<i>Jonathan Kistner, Wilhelm Tegethoff, Nicolas Fidorra, Jürgen Köhler</i>	59

Methoden & deren Anwendung I

A Python Framework for Model Specification and Automatic Model Generation for Multiple Simulators	
<i>Hendrik Folkerts, Thorsten Pawletta, Christina Deatcu, Sven Hartmann</i>	69
MATLAB/Simulink's Variant Manager vs SESToPy	
<i>Christina Deatcu, Thorsten Pawletta, Hendrik Folkerts</i>	77
Automatische Modellbildung mittels SES/MB Framework und einer Template Erweiterung	
<i>Alexander Martens, Christian Bock, Olaf Simanski, Olaf Hagendorf</i>	81

Industrie 4.0 und Digitaler Zwilling

Modellierung und Simulation vernetzter mechatronischer Komponenten einer Fertigungsstraße im Kontext der Industrie 4.0	
<i>Xiaobo Liu-Henke, Sören Scherler, Or Aviv Yarom, Jie Zhang</i>	85
Development of a Modular Configuration Framework for Digital Twins in Virtual Testbeds	
<i>Ulrich Dahmen, Tobias Osterloh, Jürgen Roßmann</i>	91
Der digitale Zwilling - Anwenderbeispiel einer Physics-Based-Simulation in der Produktion	
<i>Nanno Peters, Udo Triltsch, Dennis Rein</i>	99

Simulation in der Luft- und Raumfahrt

Inside the Virtual Test Aircraft (VIRTAC) Benchmark Model: Simulation Architecture	
<i>Nicolas Fezans, Christoph Deiler</i>	105
An Adaptable Full-Scale Aircraft Cabin in an Interconnected Simulation Environment	
<i>Mario Kallenbach, Stephan Kocks, Paul Frost, Ingo Voissel, Peter Hecker</i>	117
Instructor Operator Services for Real-time Flight Simulators	
<i>Md Nizam Uddin, Umut Durak, Jürgen Gottschlich, Sven Hartmann</i>	123

Methoden & deren Anwendung II

Simulation of RPDEVS Models of Logic Gates	
<i>Christian Fiedler, Franz J. Preyser, Wolfgang Kastner</i>	129

Probabilistic state space models - A theoretical framework with practical relevance	
<i>Peter Junglas</i>	135
RPDEVS Abstract Simulator	
<i>Franz Josef Preyser, Bernhard Heinzl, Wolfgang Kastner</i>	141
 Simulation mechatronischer Systeme	
Simulation von Partikelflugbahnen zur Auslegungshilfe von elektrostatischen Luftfiltern	
<i>Sebastian Beckers, Julian Pawlik, Jürgen Kiel, Wolfgang Grote</i>	147
Simulationsansätze zur Funktionsintegration bei elektrohydraulischen Linearachsen	
<i>Florian Meyer, Andreas Ligocki</i>	153
Model-based time varying predictive vibration control of a beam structure subjected to moving masses	
<i>Lukas Sievert, Dan Stancioiu</i>	157
Virtual Coupling of Powertrain Components: New Applications in Testing	
<i>Thomas Gwosch, Michael Steck, Sven Matthiesen</i>	163
 Simulation von Elektro- und Hybridfahrzeugen	
Modellbildung und Systemidentifikation einer PEM-Brennstoffzelle	
<i>Sören Scherler, Xiaobo Liu-Henke, Markus Henke</i>	169
Simulation und Modellbildung einer integrierten Positioniereinrichtung für induktive Fahrzeug-Batterieladesysteme	
<i>Lyucheng Zhu, Oleg Schäfer, Markus Henke, Jürgen Meins</i>	175
Untersuchung des kognitiven menschlichen Verhaltens bei der Personalisierung fahrzeugmechatronischer Systeme	
<i>Haoqi Tao, Xiaobo Liu-Henke, Thomas Vietor</i>	181
Modellbasierte Entwicklung einer Spurfolgeregelung mittels zeitdiskrettem modellprädiktivem Regler (DMPC) für unteraktuierte Systeme	
<i>Jie Zhang, Marian Göllner, Xiaobo Liu-Henke</i>	189
 Simulation technischer Systeme	
Simulation und Kompensation der Eigenspannungen in der additiven Fertigung	
<i>Tobias Mussehl, Martin Rambke</i>	195
Expansion of Models for Heart Rate Variability beyond the Autonomic Nervous System (Kurzfassung)	
<i>Jennifer Straub, Martin Bachler</i>	201
Semantisches Matching für die Konfiguration von Komponenten in Cyber-physischen Systemen	
<i>Daning Wang, Christoph Knieke, Sebastian Lawrenz, Andreas Rausch</i>	203

Methoden & deren Anwendung III

Leichtbauoptimierung von Crashstrukturen unter Einsatz einer automatisierten CAx-Prozesskette

Alexander Schülke, Andree Kafurke, Igor Sokrut, Jürgen Hillmann, Martin Müller. 211

Simulation Case Study: Using Simscape for Human Knee Joint Models

Ruth Leskovar, Andreas Körner, Felix Breiteneker 217

A Simple Simulation Model to Test Detection Methods for Periodic Effects on Transit Traversal Times

Oliver Ullrich, Daniel Lückerath 221

**ASIM Workshop 2019
Simulation Technischer Systemen
Grundlagen und Methoden in
Modellbildung und Simulation
Tagungsband**

Editorial

Wir leben im Digitalisierungszeitalter. Die Technologieentwicklung in der heutigen modernen Welt wird weitgehend von der Informatik bestimmt. Als interdisziplinäres Fachgebiet der Informatik, Simulation ist einer der wichtigsten Bausteine unserer computergestützten Zukunft. In ihrem neuen Buch¹ stellen Mittal, Durak und Ören simulationsbasierte Disziplinen vor, die die Beiträge der Simulation zu anderen Disziplinen betonen. Die Synergie zwischen Simulation und den Cyber Physical Systems (CPS), künstlicher Intelligenz (KI) und Big Data Technologien gestaltet die Zukunft der technischen Systeme.

Die ASIM - Arbeitsgemeinschaft Simulation - ist ein Fachausschuss in der GI - Gesellschaft für Informatik - zur Förderung und Weiterentwicklung von Modellbildung und Simulation im deutschsprachigen Raum.

Der Begriff technische Systeme bezieht sich auf alle vom Menschen geschaffenen Artefakte, Objekte, Produkte, Werkzeuge und technische Arbeiten, die das Ergebnis einer Produktionstätigkeit sind². Die ASIM-Fachgruppe "Simulation Technischer Systeme" (STS)³ befasst sich mit der Modellbildung und Simulation von derartigen Systemen. Über die dabei verwendeten und benötigten Modellierungs- und Simulationsmethoden hinaus, arbeitet die STS eng mit der ASIM-Fachgruppe "Grundlagen und Methoden in Modellbildung und Simulation" (GMMS)⁴ zusammen. GMMS⁴ beschäftigt sich mit neuen methodischen Entwicklungen zu Modellierungsansätzen, numerischen und softwaretechnischen Verfahren, Algorithmen sowie Simulationswerkzeugen.

Der seit vielen Jahren erfolgreiche ASIM Workshop STS/GMMS ist eines der Ergebnisse dieser Zusammenarbeit. Er umfasst die ganze Bandbreite aktueller Simulationsforschung und bietet ein Forum zur Diskussion methodischer Ansätze sowie praktischer Anwendungen auf den Gebieten der Simulation und der Modellbildung.

Mit 2 Plenarvorträgen, 39 Fachbeiträgen und 4 Tutorials findet der ASIM Workshop 2019 STS/GMMS am Deutschen Zentrum für Luft- und Raumfahrt (DLR), Standort Braunschweig, statt. Inspiriert durch den Tagungsort erreichten uns in diesem Jahr besonders viele Beiträge zum Schwerpunkt Simulation in der Luft- und Raumfahrt sowie zur Simulation mechatronischer Systeme.

Die 34 in diesem Tagungsband veröffentlichten Beiträge bilden dennoch das gesamte Spektrum der Arbeit in den ASIM Fachgruppen STS und GMMS ab. Neben den Themen mit Bezug zu grundlegenden Methoden sowie deren Anwendung, wie z.B. der modellbasierten Entwicklung, Simulationsalgorithmen und maschinellem Lernen, finden sich Beiträge zu Simulation in der Energietechnik, zur Simulation thermischer Systeme sowie zur Industrie 4.0 und Digitalem Zwilling. Mit Beiträgen zur Simulation von Elektro- und Hybridfahrzeugen ist auch in diesem Jahr der Automobilbereich stark vertreten.

Wir präsentieren Ihnen mit diesem Tagungsband ein breites Spektrum an Ideen, Ansätzen und Anwendungsbeispielen und freuen uns auf einen erfolgreichen Workshop mit fruchtbarem Austausch.

Februar 2019

Umut Durak
Christina Deatcu
Jan Hettwer

¹ S. Mittal, U. Durak, and T. Ören, eds., Guide to Simulation-Based Disciplines: Advancing Our Computational Future. (Springer, Cham, 2017).

² V. Hubka, W.E. Eder, Theory of Technical Systems: A Total Concept Theory for Engineering Design (Springer, Berlin, 2012)

³ <https://www.asim-gi.org/fachgruppen/technische-systeme/>

⁴ <https://www.asim-gi.org/fachgruppen/grundlagen-und-methoden/>

Simulation in der Tiefbohrtechnik – Modellierung, Verifikation und Validierung

Gunther Brenner

Institut für Technische Mechanik, TU Clausthal, Adolph-Roemer-Straße 2A,
38678 Clausthal-Zellerfeld, Deutschland; gunther.brenner@tu-clausthal.de

Abstract. Zur Sicherung der Energieversorgung, vor allem mit Gas, Erdöl und Tiefengeothermie, kommt der Tiefbohrtechnik eine herausragende Bedeutung zu. Angesichts steigender Anforderungen an die zu gewährleistende Sicherheit beim Tiefbohrprozess, ist die effiziente und sichere Reinigung des Bohrlochs vom erzeugten Schnittgut sowie die Erhaltung der Bohrlochintegrität durch Einhalten eines engen Druckfensters zwischen Gesteins- und Reservoirdruck von besonderer Wichtigkeit. Daher existiert aktuell in der Industrie ein großer Bedarf an effizienten, und validen Berechnungsmodellen für die mit dem Bohrklein beladenen Strömung zwischen Bohrgestänge und gebohrtem Gestein. Hierbei muss neben der komplexen Rheologie der Spülflüssigkeit auch der Einfluss von sekundären Strömungen aufgrund der Rotation des Bohrgestänges auf den Partikeltransport mit hinreichender Genauigkeit abgebildet werden. Die reduzierten Berechnungsmodelle dienen zum einen der Optimierung und Steuerung des Bohrprozesses und zum anderen der simulationsgestützten Ausbildung des Bohrpersonals. Vor dem Hintergrund von mehreren Kilometer langen Bohrstrecken kommt der Validierung der zu entwickelnden Modelle eine besondere Bedeutung zu, denn selbst kleine Unsicherheiten können zu extremen Fehleinschätzungen mit katastrophalem Ausgang führen. Im vorliegenden Beitrag wird der aktuelle Stand der unterschiedlichen numerischen Modellierungsansätze für mehrphasige Strömungen im Kontext der Tiefbohrtechnik diskutiert. Hierzu gehören auf der einen Seite, neben dem klassischen Euler-Lagrange-Ansatz, auch skalenauflösende Mehrphasenmodelle die auf einer Kombination von DEM und CFD beruhen. Auf der anderen Seite werden aggregierte Modelle (diff-flux, two- und threelayer) sowie Modelle auf Basis der sog. Populationsmethode vorgestellt. Ferner werden die Möglichkeiten und Herausforderungen zur Aggregation von numerischen Ergebnissen in reduzierte Modelle für den industriellen Einsatz diskutiert.

Vita. Gunther Brenner, Jahrgang 1962, studierte Maschinenbau an der Technischen Universität in Karlsruhe. 1988 begann er seine Tätigkeit als wissenschaftlicher Mitarbeiter am damaligen Institut für Theoretische Strömungsmechanik (jetzt Institut für Institut für Aerodynamik und Strömungstechnik) des Deutschen Zentrums für Luft- und Raumfahrt (DLR) in Göttingen. 1994 promovierte er zum Doktor-Ingenieur an der RWTH Aachen mit dem Thema „Numerische Simulation der Wechselwirkung zwischen Stößen und Grenzschichten in reagierenden Hyperschallströmungen“. Nach einem Aufenthalt als Post-Doc in Toulouse/Frankreich von 1994 bis Ende 1995 übernahm er die Leitung einer Forschergruppe am Lehrstuhl für Strömungsmechanik der Universität Erlangen. 2002 habilitierte er sich an der Uni Erlangen auf dem Gebiet der Strömungsmechanik. 2003 nahm er einen Ruf an die TU Clausthal an und vertritt dort am Institut für Technische Mechanik das Fachgebiet Strömungsmechanik. Er ist Vorsitzender der Simulationswissenschaftlichen Zentrums, einer gemeinsamen Forschungseinrichtung der Universität Göttingen und der TU Clausthal. Seit 2015 ist er zugleich Vizepräsident für Studium und Lehre.

Kabinensimulation mit AVES - Herausforderungen und neue Möglichkeiten

Holger Duda

Institut für Flugsystemtechnik, Deutsches Zentrum für Luft- und Raumfahrt (DLR), Lilienthalplatz 7,
38108 Braunschweig, Deutschland; *Holger.Duda@dlr.de*

Abstract. Der Vortrag gibt einen Überblick über den Einsatz des Forschungssimulators AVES (Air Vehicle Simulator) beim DLR in Braunschweig. AVES ist ausgelegt als modulare, flexible Plattform unter Einsatz modernster Technologien zur ganzheitlichen Erforschung des Fliegens. Zentrales Forschungsgebiet ist die dynamische Interaktion zwischen Mensch und Luftfahrzeug, insbesondere für Verkehrsflugzeuge und Hubschrauber und neuerdings für Passagierkabinen. Der Aufbau der neuen Passagierkabine stellte das AVES-Team vor neue Herausforderungen, z. B. wie in dem begrenzten Bauraum ein möglichst gutes Kabinengefühl bei den Probanden geschaffen werden kann.

Die Kabine kann bis zu 16 Passagiere aufnehmen und alle Flugbewegungen, wie Start, Landung und Turbulenzen mit dem Bewegungssystem des AVES nachbilden. Die Klimatisierung erzeugt zusammen mit den Fluggeräuschen und Ansagen eine authentische Kabinenatmosphäre, so dass die Passagiere sich wie bei einem richtigen Linienflug fühlen.

Eine der ersten Studien mit der Passagierkabine beschäftigte sich im November 2018 mit Fragen zur Akzeptanz einer fensterlosen Kabine. Die Außensicht wird den Passagieren über acht Monitore dargestellt, die als virtuelle Fenster fungieren. Forschungsgegenstand ist dabei der Einfluss von Latenzen zwischen Bewegung und Sicht. Zukünftig sind weitere Forschungen zur individualisierten Kabine, zum Passagierkomfort oder zur Minderung der Flugangst möglich.

Vita. Holger Duda, 1965 in Emden geboren, studierte Luft- und Raumfahrttechnik an der Technischen Universität in Braunschweig. 1991 begann er seine Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Flugmechanik des Deutschen Zentrums für Luft- und Raumfahrt (DLR) auf dem Gebiet der Flugeigenschaften und Flugsteuerung. Im Zeitraum von 1995 bis 1996 arbeitete er in Manching an der Flugerprobung von Tornado und EF2000. 1997 promovierte er zum Doktor-Ingenieur an der TU Braunschweig mit dem Thema „Fliegbarkriterien bei begrenzter Stellgeschwindigkeit“. 1999 wechselte er zur Volkswagen-Konzernforschung nach Wolfsburg und übernahm dort Aufgaben im Bereich aktiver Fahrdynamiksysteme. Im Juni 2005 kehrte er als Leiter der Abteilung Simulationstechnik zurück an das DLR-Institut für Flugsystemtechnik. Von 2010 bis 2013 war er verantwortlich für den Aufbau des Simulatorzentrums AVES (Air VEhicle Simulator). Seit 2007 ist er in der Lehre an der TU Braunschweig im Bereich Flugsimulation tätig.

Cross-Layer Behavioral Modeling and Simulation of E/E-Architectures using PREEvision and Ptolemy II

Harald Bucher^{1*}, Simon Kamm², Jürgen Becker¹

¹Karlsruhe Institute of Technology – Institute for Information Processing Technologies, Engesserstr. 5, 76131 Karlsruhe; **bucher@kit.edu*

²Vector Informatik GmbH, Philipp-Reis-Str. 1, 76137 Karlsruhe

Abstract. In this paper an approach for integrated behavior modeling and simulation within model-based electric/electronic-architecture (EEA) descriptions is presented. It leverages actor-oriented and UML state chart behavior modeling to address complex reactive systems. A key contribution is the aggregation of cross-layer behavior specified at the logical function architecture layer and at the hardware layer together with further properties of the EEA like the current consumption of electronic control units (ECUs) and the underlying network topology. The EEA and behavior modeling is done in the common industry tool PREEvision. Using these static descriptions, a unified simulation model is synthesized and executed using Ptolemy II, which extends a previously developed approach. In addition, a concept to feed back the simulation data into PREEvision is briefly described e.g., to further evaluate the gained results. Finally, a proof-of-concept is presented using an Adaptive Cruise Control application.

Introduction

Automotive electric/electronic-architectures (EEAs) are steadily growing in complexity due to the integration of evermore functions [1]. To cope with that complexity at system level, model-based architecture description languages (ADLs) and tools have been established in recent years such as the EAST-ADL [2], EEA-ADL [3] (realized in the tool PREEvision [4,1]) and Vehicle Systems Architect [5], each of which are compliant to the AUTOSAR [6] standard. Each of them offer sophisticated static modeling capabilities from several viewpoints such as requirements, functional network, hardware/software architecture, wiring harness and topology.

A common process is to start with the realization-independent and early stage modeling of the logical function architecture which typically stays stable for years

and thus is the basis for further refinements in the development life cycle [1]. Another trend is the architecture-centric modeling of behavior integrated within the model-based EEA descriptions in order to have a common formal format for exchange and subsequent simulation analysis. The trend to amend this is underlined e.g., by the behavioral annexes of the EAST-ADL [2] and the AADL [7] or the integration of UML-compliant state charts into the latest PREEvision release v9.0 [4]. Therefore, recent research is focused on the generation of executable behavior from these static descriptions [8,9,10,11,12,13].

A downside of the behavioral annexes is that they only support the association of architectural components with simple, flat finite state machines (FSMs) which result in state and transition explosion with more complex systems. The mentioned approaches therefore often delegate detailed behavior to external descriptions which results in the loss of the integrated characteristics. In addition, it elicits inconsistencies between the architecture and behavior models and prevents the consideration of lower abstraction layers.

An approach which faces this challenge is presented in [8] by synthesizing and executing a cross-domain simulation from static EEA descriptions designed in PREEvision. In this work we extend that approach to support both actor-oriented and state chart behavior modeling to address complex reactive systems. Concerning state charts the UML subset provided by PREEvision is leveraged and enhanced to support extended state machines to further handle complexity. In addition, cross-layer behavior specifications and further EEA properties from lower abstraction layers are synthesized into a unified Ptolemy II (PtII) simulation model. A concept to feed back the simulation data into PREEvision is extended and completes the contributions.

1 Background

The overall baseline approach as proposed in [8] and the extensions addressed in this work are shown in **Figure 1**. Starting point is a data model e.g., as provided by PREEvision, which captures all relevant abstraction layers of an EEA. For modeling *executable* behavior integrated within the EEA model, a new layer called *Behavioral Logical Architecture (BLA)* is introduced that refines the static logical blocks with detailed behavior by reusing actors [14] from the *PtII Actor Library*. The library contains actors of the heterogeneous modeling and simulation tool Ptolemy II [15] and is imported as a separate library of logical block types into PREEvision. These block types are used to instantiate actors at the BLA layer. In combination with mappings from the LA layer to lower layers they provide the connection of the behavioral blocks of the BLA to domain-specific information at lower layers enabling the cross-domain simulation of the underlying network communication or even electric circuits [16] in an aspect-oriented manner. A variant-sensitive synthesis is also implemented supporting the analysis of architecture variants.

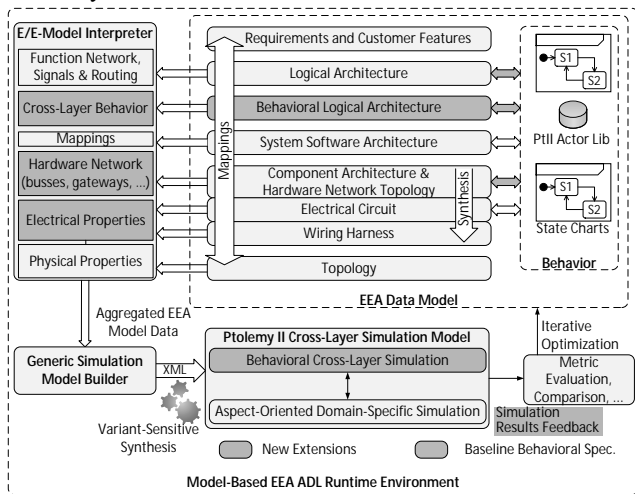


Figure 1: Approach for cross-domain simulation synthesis of model-based EEAs [8] and new extensions to combine cross-layer behavior specifications using UML state charts and actor-oriented library components.

2 Concepts

To amend the baseline actor-oriented modeling with state-based behavior we leverage the newly added capability of PREEvision v9.0 to refine architecture artifacts

with state charts across several layers including the logical architecture and components of the hardware layer such as ECUs and internal processing units.

The basic principle is to annotate a state chart as child artifact to an architecture artifact. Dependent on the abstraction layer, the interfaces to the state chart comprise different data providers and consumers. At the logical architecture, for instance, communication between functions is done via typed ports, which have attached an interface. The interface specifies the actual data exchanged e.g., in terms of data elements. This follows the AUTOSAR standard. The specified data elements of each port are then available in the state chart of the function to use them in guard and action expressions. The modeling is illustrated in **Figure 2**. A similar modeling approach applies for hardware components except that state charts are annotated at instance level and data providers differ from data elements.

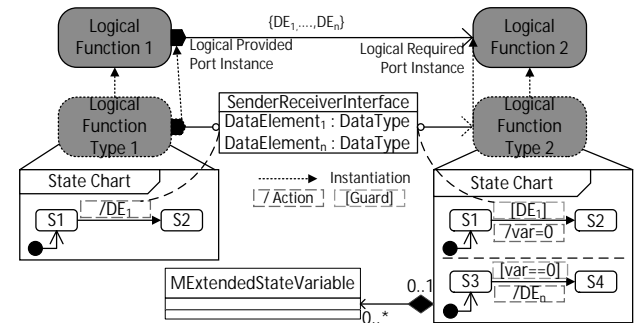


Figure 2: Modeling principle to refine logical function types with state charts. Communication is done via data elements.

2.1 Extended State Machines

A downside of the current state chart modeling capability is the missing support of extended state machines, which can significantly reduce the complexity [15]. Therefore, we propose a meta-model extension by extended state variables. This is indicated in **Figure 2** by the composition of the state chart with the proposed meta-class `MExtendedStateVariable`.

2.2 Combining Actor-oriented and State Chart Behavior Simulation

In the baseline approach in **Figure 1**, behavior is specified by mapping an atomic logical function instance to a composite building block at the BLA layer. Executable actors are instantiated within that building block. Port

prototype mappings are generated once to ensure the consistency between the interfaces of the atomic logical function and its refinement building block.

State charts are simulated using modal models [15,17] in PtII. Modal models basically represent a specialized composite actor containing a hierarchical state machine governed by an *FSMDirector*. Each state can contain another state machine refinement or even an actor-oriented sub-model following a distinct execution semantics i.e., a different model of computation (called *Director* in PtII). Modal models are also suitable to deterministically simulate *hybrid systems* [15]. Data exchange between modal models is done via ports. Therefore, a building block of type *ModalModel* is used to identify logical functions which contain a state chart description.

Additional data element sub-mappings are generated once in order to respect the interfaces of the logical functions and to connect the simulation model counterparts during simulation model synthesis. Each port of a building block represents a data element. See **Figure 3**.

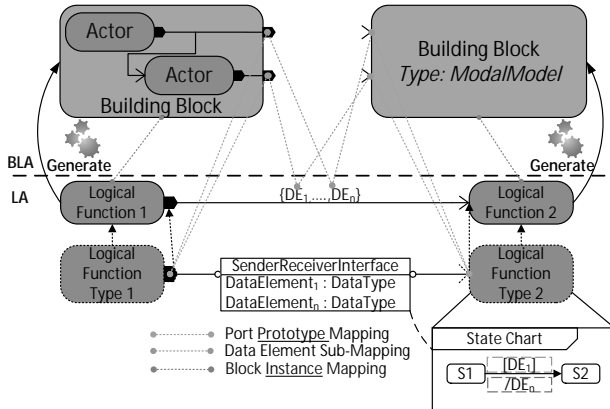


Figure 3: Generation of BLA building block stubs and mappings. State charts are encapsulated in a building block of type *ModalModel*.

2.3 Cross-Layer Behavioral Synthesis

To allow the simulation of cross-layer behavior we leverage the state chart refinements of hardware components. However, the link to higher layers i.e., the logical layer, is missing.

Therefore, we propose to use AUTOSAR-oriented *BasisServiceInterfaces* on logical ports in order to provide additional data elements or operations to communicate with state charts of mapped hardware components. In addition, we propose to reference ECU attributes as state chart variables. For instance, this enables the

modeling and simulation of mode-based cross-layer behavior, where functions can request a certain operating mode of the ECU and only perform its functional behavior if the ECU responds it is ready to run. In order to allow spontaneous FSMs [15], which not only react on input events, a *timeout* guard expression (taken from PtII) is introduced e.g., to model the startup time of the ECU. An example cross-layer model is shown in **Figure 4**.

Finally, we propose the mapping of current consumption descriptions in terms of PREEvision's meta-class *MCurrentDescriptorType* on state transitions of hardware states. Together with the timed behavior, a mode-based current consumption can be simulated.

In the synthesized Ptolemy II model, the function and hardware state charts are encapsulated in distinct modal models communicating via ports which represent the basis service interfaces. An additional output port is generated for the current consumption of the ECU.

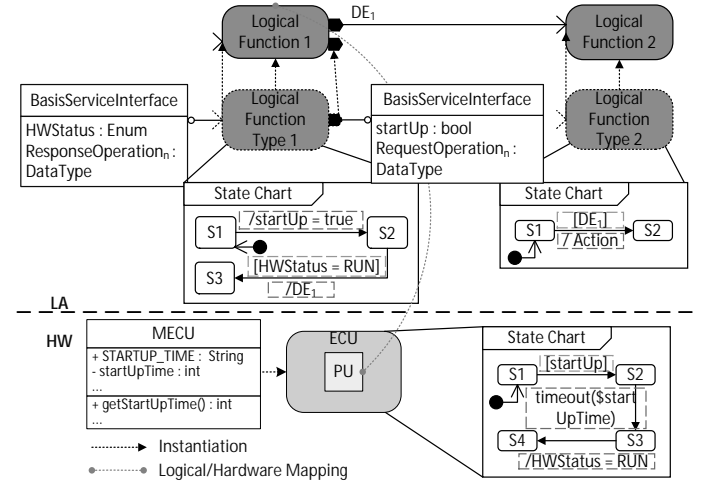


Figure 4: Cross-layer behavior specification using basis service interfaces on logical ports to communicate with the mapped hardware component state chart via additional data elements or operations.

Hardware Network. In [8], network communication between functions such as CAN is traced based on their mapping to the hardware and is considered in the resulting simulation in an aspect-oriented way. Together with the state chart refinement of ECUs and processing units, it is possible to automatically include additional behavior along the communication path such as gateways by cascaded aspect-oriented simulations. Typically gateways have no logical function counterpart, since they are dependent on the mapping.

2.4 Simulation Data Feedback

To make use of the simulated results in PREEvision in order to perform further analysis and to relate the results with the original EEA model artifacts, a feedback approach is applied. The approach relies on OSGi [18] and is further described in [19]. We reuse and extend the approach by implementing a listener for modal model controllers focusing on the feedback of information about the simulated state machines such as timestamps, current state, previous state, output and variable actions.

2.5 Transformation Rules

In **Table 1** the basic transformation rules between PREEvision's UML state chart subset and modal model artifacts in PtII are summarized.

Note that each generated PtII artifact is suffixed by the UUID of the original EEA model artifact in order to uniquely relate the artifacts and avoid name conflicts on PtII side.

UML State Chart Subset	Ptolemy II Modal Models
simple state, choice & junction pseudo-state	state
initial pseudo-state	state with property <i>isInitialState</i>
final state	state with property <i>isFinalState</i>
composite state	<i>state machine refinement</i> state
orthogonal state	<i>default refinement</i> state containing a discrete-event director and a modal model composite for

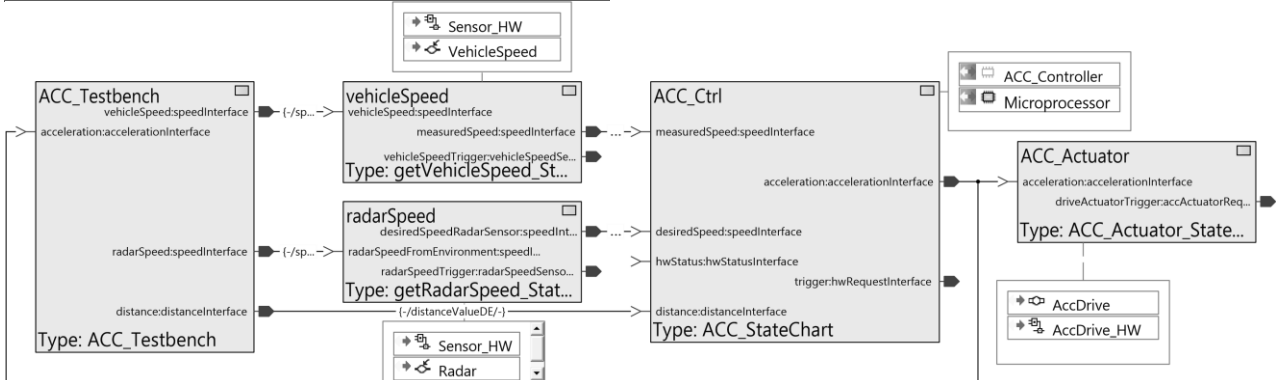


Figure 5: Logical function architecture of the ACC application. Each of the logical functions except for *ACC_Testbench* is refined by a state chart. Their mapping to the hardware layer is illustrated by the annotated text boxes. The behavior of the *ACC_Testbench* is modelled actor-oriented at the BLA layer and is not mapped to the hardware. Thus, a combined actor-oriented and state chart modeling is applied.

3.1 State Charts

The most important state charts are the one of the ACC

each parallel region. Data dependencies between regions are analyzed and communicated via ports between the affected modal models. [17]

deep history state	history transition
state transition	ordinary transition
guard condition	guard expression
IO/variable action	output/set action expression

Table 1: Basic transformation rules between PREEvision's UML state chart subset and PtII modal models.

3 Use Case Results

In this chapter, the concepts are demonstrated by means of an *Adaptive Cruise Control (ACC)* application presented in [8] which is enhanced based on **Figure 4**.

The logical function architecture is shown in **Figure 5**. The *ACC_Testbench* generates the stimuli for the vehicle speed and radar speed sensor functions as well as for the ACC controller in a closed-loop fashion based on the calculated acceleration of the ACC controller. The initial speeds and the distance are set to 15m/s and 190m respectively. Each of the functions offer *BasisServiceInterfaces* to request or retrieve a certain operating mode of the state chart of their mapped hardware component. The corresponding BLA building block stubs and mappings are generated according to **Figure 3**.

controller shown in **Figure 6** and its corresponding ECU state chart realizing an ECU Manager depicted in **Figure 7**. The remaining state charts of the sensor and actuator

functions and their hardware are modelled simple. They only forward/retrieve the speed/acceleration values and request the sensors/actuator to run as long as they receive values. The ACC is calculating the acceleration only if the ECU is ready to run. The orthogonal state *operate* limits the calculated acceleration by the state variables *aMin* and *aMax*. In the *freeRoad* state the radar detects no vehicle, the own speed reached the desired speed and the sleep mode is requested. A wakeup is triggered when the radar detects a new vehicle. A shutdown is requested when the vehicle stands still.

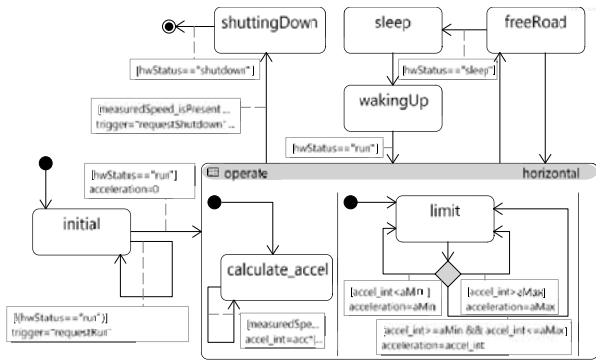


Figure 6: ACC controller state chart. Some transition actions are omitted for space reasons.

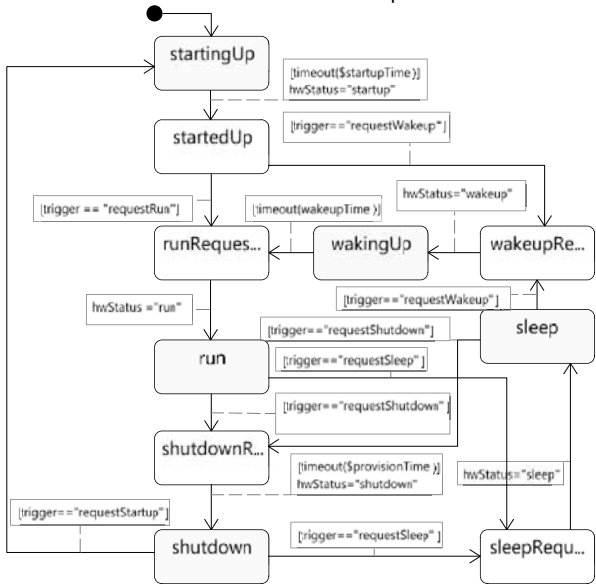


Figure 7: ACC ECU state chart realizing an ECU Manager oriented on the AUTOSAR fixed ECU Manager. Transitions to the yellow states have mapped a current descriptor type in order to simulate a mode-based current consumption.

The ACC ECU state chart represents the different operating modes which can be requested by the ACC controller state chart and sends back the current status via the BasisServiceInterfaces. In addition, the startup- and

provision time attributes of the ECU (50ms and 200s) are referenced as well as a *wakeupTime* state variable (10ms) which are used as timeouts. Provision time is the time a component stays active after its shutdown is requested.

3.2 Simulation Results

Figure 8 shows the PtII plot of the ACC simulation. Until 250s the vehicle is following the leading vehicle. Then the leading vehicle disappears and the ACC accelerates to its desired speed at free road. At 300s a new vehicle is detected at a distance of 200m. At 350s the vehicle is decelerating until it stands still at 380s. At 350s the acceleration is limited to *aMin*.

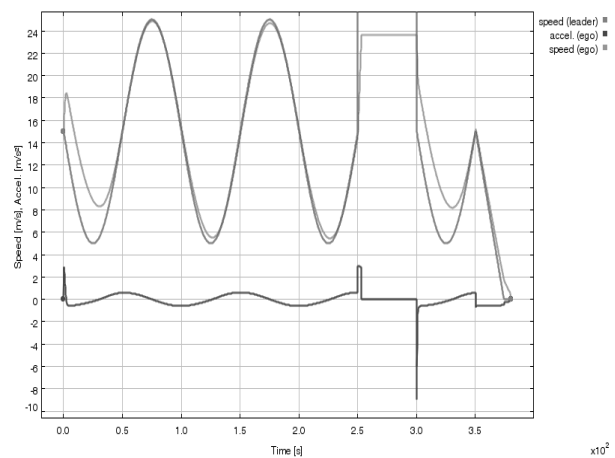


Figure 8: ACC simulation showing the speed of the leading vehicle (red) and the ego vehicle (green) in m/s as well as the acceleration calculated by the ACC (blue) in m/s².

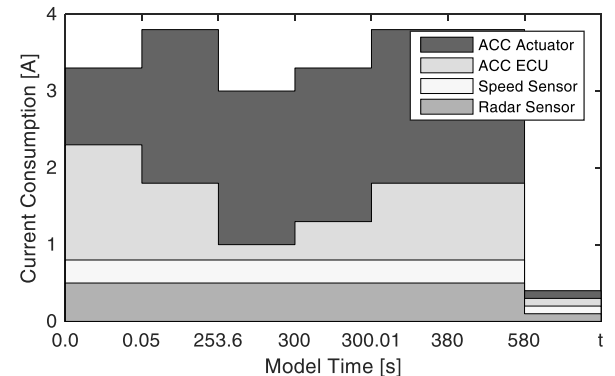


Figure 9: Current consumption of the mapped ACC hardware dependent on the operating mode. Created based on the fed back simulation data which is written to a CSV file in PREEvision.

Figure 9 shows the mode-based current consumption of the hardware components at the key time-points with

synthetic values. The sensors are operating until their shutdown. Until 50ms the ACC ECU and Actuator are starting up before they are ready to run. At 253.6s the vehicle has reached its desired speed at free road and the ACC ECU goes to sleep mode. At 300s it wakes up for 10ms. At 380s the vehicle stands still but all hardware components stay active for the same provision time before they shutdown at 580s.

4 Conclusion

In this paper, we presented a set of concepts and its evaluation by an ACC application to model and simulate behavior of model-based EEAs in an integrated manner. The key contribution is the combination of actor-oriented and state chart based behavior across several abstraction layers. This enables new possibilities to analyze model-based EEAs in early development stages dependent on architectural decisions and information.

Future work could include enhanced support of UML state charts, the integration and consideration of behavior at the AUTOSAR-compliant system software architecture layer and envisioning their code generation.

References

- [1] J. Schäuffele, "E/E Architectural Design and Optimization using PREEvision," in *SAE Technical Paper 2016-01-0016*, 2016. [Online]. <https://doi.org/10.4271/2016-01-0016>
- [2] EAST-ADL Association. (2013) EAST-ADL Domain Model Specification. [Online]. <http://www.east-adl.info/Specification>
- [3] J. Matheis, "Abstraktionsebenenübergreifende Darstellung von Elektrik/Elektronik-Architekturen in Kraftfahrzeugen zur Ableitung von Sicherheitszielen nach ISO 26262," Karlsruhe Institute of Technology, Ph.D. thesis ISBN: 978-3-8322-8968-3, 2010.
- [4] Vector Informatik GmbH, "PREEvision v9.0 Manual," 2018.
- [5] Mentor Graphics. (2018, Oct.) Volcano™ Vehicle Systems Architect. [Online]. <https://www.mentor.com/products/vnd/autosar-products/volcano-system-architect>
- [6] AUTOSAR Consortium. (2018) AUTOSAR 4.4 (Automotive Open System Architecture) Specifications. [Online]. <http://www.autosar.org>
- [7] SAE International, "SAE Architecture Analysis and Design Language (AADL) Annex Volume 2: Annex B: Data Modeling Annex, Annex D: Behavior Model Annex, Annex F: ARINC653 Annex," USA, Standard AS5506/2, Jan. 2011.
- [8] H. Bucher, C. Reichmann, and J. Becker, "An Integrated Approach Enabling Cross-Domain Simulation of Model-Based E/E-Architectures," in *SAE Technical Paper 2017-01-0006*, Mar. 2017. [Online]. <http://papers.sae.org/2017-01-0006/>
- [9] R. Weissnegger et al., "Simulation-based Verification of Automotive Safety-critical Systems Based on EAST-ADL," *Procedia Computer Science*, vol. 83, pp. 245-252, 2016.
- [10] R. Marinescu et al., "Analyzing Industrial Architectural Models by Simulation and Model-Checking," in *Formal Techniques for Safety-Critical Systems*.: Springer International Publishing, 2015, vol. 476, pp. 189-205.
- [11] MAENAD Consortium, "MAENAD Analysis Workbench," Deliverable D5.2.1 V4.0 2014. [Online]. <http://www.maenad.eu/public/Deliverables>
- [12] G. Lasnier, L. Pautet, J. Hugues, and L. Wrage, "An Implementation of the Behavior Annex in the AADL-Toolset Osate2," in *2011 16th IEEE International Conference on Engineering of Complex Computer Systems*, Apr. 2011, pp. 332-337.
- [13] P. G. Larsen et al., "Integrated Tool Chain for Model-Based Design of Cyber-Physical Systems," in *The 14th Overture Workshop: Towards Analytical Tool Chains*, vol. 4/28, Nov. 2016, pp. 63-79.
- [14] E. A. Lee, S. Neuendorffer, and M. J. Wirthlin, "Actor-Oriented Design Of Embedded Hardware And Software Systems," *Journal of Circuits, Systems, and Computers*, vol. 12, pp. 231-260, 2003.
- [15] Claudius Ptolemaeus, *System Design, Modeling, and Simulation using Ptolemy II*.: Ptolemy.org, 2014. [Online]. <http://ptolemy.org/books/Systems>
- [16] H. Bucher and J. Becker, "Electric Circuit- and Wiring Harness-Aware Behavioral Simulation of Model-Based E/E-Architectures at System Level," in *2018 IEEE International Systems Engineering Symposium (ISSE)*, 2018, pp. 1-8.
- [17] E. A. Lee and S. Tripakis, "Modal Models in Ptolemy," in *Proceedings of the 3rd International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools*, Oct. 2010, pp. 11-21.
- [18] OSGi Alliance. (2018, Oct.) The Dynamic Module System for Java. [Online]. <https://www.osgi.org>
- [19] H. Bucher, K. Neubauer, and J. Becker, "Automated Assessment of E/E-Architecture Variants using an Integrated Model- and Simulation-based Approach [in press]," in *SAE Technical Paper 2019-01-0111*, 2019.

Ein HiL-Prüfstand zur Funktionsabsicherung von Batteriemanagementsystemen mit Low-Cost Entwicklungsplattform

Sven Jacobitz^{1*}, Xiaobo Liu-Henke¹

¹Institut für Mechatronik, Ostfalia Hochschule für angewandte Wissenschaften, Salzdahlumer Str. 46/48, 38302 Wolfenbüttel; *sve.jacobitz@ostfalia.de

Abstract. In diesem Beitrag wird die Entwicklung eines Hardware-in-the-Loop-Prüfstands zur Funktionsabsicherung von Batteriemanagementsystemen (BMS) für Lition-Ionen-Batterien vorgestellt. Der Prüfstand wird im Rahmen eines Forschungsprojektes zur Funktionsvalidierung einer Low-Cost RCP-Entwicklungsplattform sowie zum effizienten Entwickeln und Testen von BMS Funktionen eingesetzt. Diese basiert auf Scilab/Xcos zur Offlinesimulation und einem Mikrocontroller als Echtzeithardware. Er teilt sich in Prüfstandsmodul, Echtzeithardware und Softwaremodul. Die funktions- und gestaltungsorientierte Auslegung des Prüfstands erfolgt unter Verwendung dynamischer Modelle, sowohl der Strecke als auch der Funktionen. Hierdurch werden Sensoren, Aktoren und ein Prozessor ausgewählt. Abschließend erfolgt exemplarisch die Darstellung von Verifikationsergebnissen in Form von Messdaten.

Einleitung

Aufgrund stetig steigender Anforderungen an Funktionalität und Vernetzung technischer Produkte wird heutzutage immer mehr Hard- und Software integriert. Kern der hierdurch entstehenden intelligenten Systeme sind Steuergeräte und die darauf implementierten, hochwertigen Reglerfunktionen inklusive der Signalverarbeitung. Zusammen mit Sensorik und Aktorik bilden diese typische mechatronische Systeme. Neben dem Funktionsumfang führt der hohe Vernetzungsgrad zu komplexer Software und starken Wechselwirkungen zwischen den Funktionen und Steuergeräten. Die harte Markt- und Wettbewerbssituation fordert zusätzlich vom Hersteller Produkte trotz rasant wachsender Komplexität in immer kürzerer Zeit kostengünstig zur Serienreife zu bringen [1]. Folglich beinhalten die Entwürfe von Soft- und Hardware sehr oft zahlreiche Fehler. Da sich aus solchen Komponenten wesentliche Eigenschaften wie Zuverlässigkeit und Sicherheit ergeben, ist die Entwicklung und Absicherung solcher eingebetteten mechatronischen Sys-

teme durch einen effektiven Entwicklungsprozess unabdingbar [2].

Zur Beherrschung der Systemkomplexität wird der strukturierte, modellbasierte, verifikationsorientierte Rapid Control Prototyping (RCP)-Entwicklungsprozess eingesetzt, welcher Model-in-the-Loop (MiL), Software-in-the-Loop (SiL), und Hardware-in-the-Loop (HiL) beinhaltet (vgl. Abbildung 1). In der Automobilentwicklung hat sich dieser etabliert [3]. Insbesondere die HiL-Simulation ist zu einem wichtigen Schritt im Freigabeprozess von Steuergerätesoftware geworden.

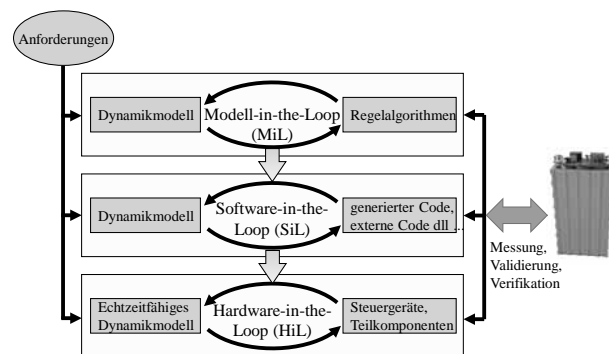


Abbildung 1: Modellbasierter, verifikationsorientierter Entwicklungsprozess [4].

In den Ebenen MiL und SiL werden die Regelfunktionen modellbasiert ausgelegt und mit definierten Schnittstellen in die Modelle der zu regelnden Strecke eingebettet. Gemäß der vorgegebenen Spezifikation und Wunschfunktionen erfolgen bereits in den frühen Entwicklungsphasen Tests. Die Simulationsergebnisse aus MiL / SiL werden in einer Echtzeitumgebung mittels HiL-Simulation weitergehend validiert und optimiert. Dabei wird stets auf die vorherige Ebene zurückgegriffen, um Fehlerquellen zu beseitigen und

gewünschten Funktionen zu realisieren. Während aller Prozessschritte erfolgen Identifikations- und Validierungsmessungen am realen System [2].

1 Motivation

Kennzeichnend für die vorgestellte Methodik sind die hohe Durchgängigkeit und der Automatisierungsgrad, von der Modellbildung, der modellbasierten Funktionsauslegung über die automatische Codegenerierung bis hin zur Echtzeitrealisierung (vgl. Abbildung 2 links). Sie wird durch eine durchgängige, voll automatisierte CAE-Plattform begleitet. Derzeit unterstützen ausschließlich kostenintensive Kombinationen aus CAE-Werkzeug und Echtzeithardware, z. B. Matlab/Simulink mit einem System der Firma dSPACE (vgl. Abbildung 2 mittig), einen durchgängigen Prozess [5].

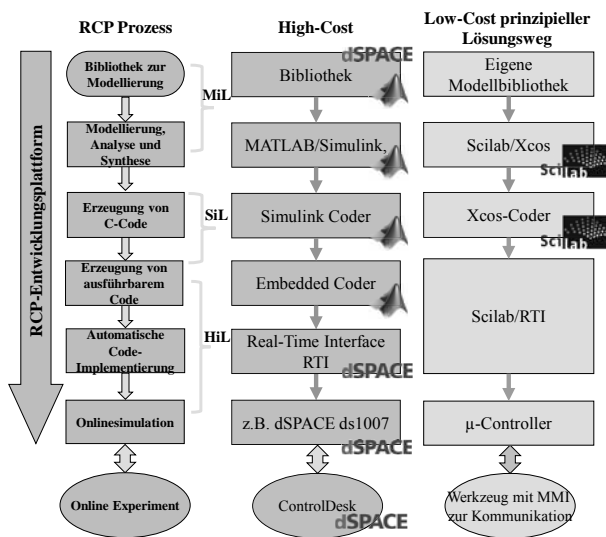


Abbildung 2: Konzept der Low-Cost Entwicklungsplattform.

Im Rahmen des durch die EU geförderten Forschungsprojektes *Low-Cost Rapid Control Prototyping-System mit Open-Source-Plattform für die Funktionsentwicklung von eingebetteten mechatronischen Systemen (Lo-CoRCP)* wird an der Ostfalia eine Plattform, welche den RCP-Prozess durchgängig unterstützt, entwickelt. Grundlage bildet das CAE-Programm Scilab/Xcos, welches ähnliche Funktionen wie Matlab/Simulink bietet. Als Echtzeithardware kommt ein Mikrocontroller inklusive Betriebssystem zum Einsatz. Abbildung 2 illustriert das Konzept der Low-Cost-Plattform

in Zusammenhang mit dem durchgängigen Entwicklungsprozess.

Die Validierung der Projektergebnisse erfolgt durchgängig anhand der modellbasierten Entwicklung der Softwarefunktionen eines Batteriemagementsystems (BMS) für Lithium-Ionen-Batterien. Der in diesem Beitrag vorgestellte Prüfstand wird hierbei auf HiL-Ebene des Entwicklungsprozesses eingesetzt.

2 Konzept des Prüfstands

Wie in Kapitel 1 beschrieben, soll der Prüfstand im Rahmen des Projektes LoCoRCP zur Validierung einer Low-Cost RCP-Entwicklungsplattform verwendet werden. Hierbei werden die Software-Funktionen eines an der Ostfalia entwickelten BMS (vgl. [4]) simuliert. Des Weiteren kann der Prüfstand auch genutzt werden, um die BMS-Funktionen effizient zu entwickeln und reproduzierbar zu testen. Der Prüfstand muss somit eine hohe Bandbreite von Entwicklungsaufgaben unterstützen [1].

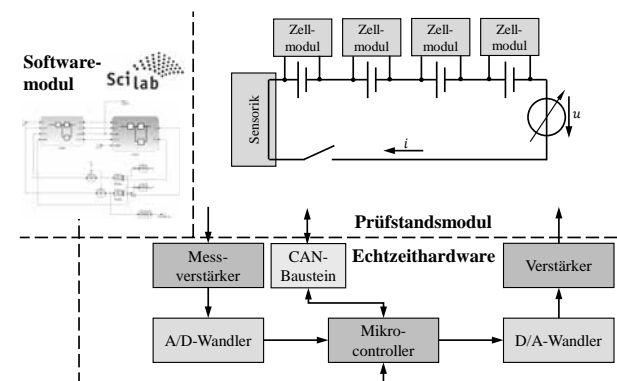


Abbildung 3: Konfiguration des HiL-Prüfstands.

Abbildung 3 stellt die Konfiguration des HiL-Prüfstands dar. Sie teilt sich in Prüfstandsmodul, Echtzeithardware und Softwaremodul auf. Das Prüfstandsmodul setzt sich aus vier LiFePO₄ Batteriezellen inklusive Zellmodulen, einer elektrischen Quelle / Last, Sensorik sowie einem Trennmechanismus zusammen. Die Realisierung wird in Kapitel 4.1 beschrieben. Als Softwaremodul kommen neben den zu testenden BMS-Funktionen umfangreiche Sicherheits- und Kommunikationsfunktionen zum Einsatz. Hierdurch werden die Anbindung einer Benutzerschnittstelle sowie die komfortable Aufzeichnung von Messdaten ermöglicht. Für die Echtzeitverarbeitung und Kommunikation wird

eine, auf einem Mikrocontroller basierende, Low-Cost Hardware verwendet (vgl. Kapitel 1). Diese wird in Kapitel 4.2 beschrieben. Die rechnergestützte Vorgehensweise zum Entwurf des gesamten HiL-Prüfstands wird im Folgenden detailliert dargestellt.

Mittels modellbasierter, funktionsorientierter Entwicklungsmethodik werden die erforderlichen Komponenten des Prüfstands neben der Dimensionierung für stationäre Betriebsfälle zusätzlich im Hinblick auf eine ausreichende Dynamik konzipiert. Dabei werden Regelungen zur Erhöhung der Bandbreite und zur Vermeidung des Totzeitverhaltens im gesamten HiL-System modellbasiert ausgelegt, sodass das Gesamtsystem über eine ausreichende Bandbreite verfügt. Das Vorgehen der funktionsorientierten Auslegung ist in Kapitel 3 beschrieben.

Anhand der gewählten Komponenten ergeben sich Anforderungen an die mechanische Tragstruktur und den Bauraum des Prüfstands. Diese werden durch den gestaltungsorientierten Entwurf, welcher hier nicht näher erläutert wird, umgesetzt.

Zuletzt erfolgt die Inbetriebnahme des Prüfstands durch Integration der Einzelkomponenten. Dabei werden die erforderlichen mechanischen, elektrischen sowie informationstechnischen Verbindungen hergestellt. Die Kalibrierung und Skalierung der Sensorik sowie die Inbetriebnahme der Echtzeithardware müssen im Vorfeld durchgeführt werden. Das Vorgehen ist in Kapitel 4.4 beschrieben.

3 Funktionsorientierte Auslegung

Im Folgenden wird die Auslegung der Prüfstandsfunktionen detailliert dargestellt. Anhand von Modellen des Batteriepakets und der Softwarefunktionen des BMS erfolgt simulationsbasiert die Bestimmung von Anforderungen an die Komponenten und Regelungen des Prüfstands.

3.1 Modellbildung

Zur Auslegung des Prüfstands wird neben dem Modell der Softwarefunktionen des BMS auch ein Modell des Batteriepaketes genutzt. Es besteht aus vier in Reihe geschalteten Modellen einer LiFePO₄-Zelle.

Das Verhalten der einzelnen Zelle ist dabei stark nichtlinear vom Ladezustand (*SoC*), der Temperatur und dem Alterungszustand abhängig [6]. Das in

diesem Beitrag beschriebene Modell vernachlässigt den Temperatur- und Alterungseinfluss.

Der Ladezustand, ausgehend vom Anfangszustand *SoC*₀, ergibt sich aus Bilanzierung des Strangstromes *i*_s unter Berücksichtigung eines stromrichtungsabhängigen Wirkungsgrades (*η*_c) sowie der Nennkapazität *C*_n:

$$SoC(t) = SoC_0 + \int \frac{\eta_c}{C_n} i_s(t) dt \quad (1)$$

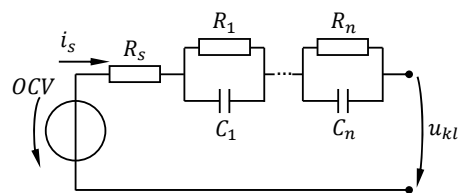


Abbildung 4: El. Ersatzschaltbild einer Batteriezelle [6].

Zur Berechnung der Klemmenspannung *u_{kl}* wird ein elektrisches Ersatzschaltbild (vgl. Abbildung 4) verwendet. Durch einen Innenwiderstand sowie mehrere RC-Glieder erfolgt die Approximation des dynamischen Verhaltens, während das stationäre Verhalten durch eine *SoC*-abhängige Spannungsquelle (*OCV*) abgebildet wird.

3.2 Struktur des Batteriemangements

Ein Ziel des BMS ist es, Lebensdauer verringernde und kritische Zustände des Batteriesystems, wie z. B. Überspannungen einzelner Zellen, zu verhindern. In diesem Zusammenhang werden diverse funktionale und nichtfunktionale Anforderungen an das System gestellt. Die Funktionen zur Erfüllung, insbesondere der funktionalen Anforderungen, sollen mittels des HiL-Prüfstands abgesichert bzw. optimiert werden.

Übergeordnet lassen sich zunächst die Aufgaben des BMS wie folgt zusammenfassen [7]:

- Schätzung und Prädiktion von Batteriezuständen
- Steuerung des Ladevorgangs / Ladeausgleichs
- Sicherheitsüberwachung
- Diagnose und Kommunikation

Eine wichtige Aufgabe des BMS ist die Schätzung von nicht messbaren Zuständen (z.B. den Ladezustand der Batterie). Auf Basis der gemessenen und geschätzten Zustände müssen zusätzlich Größen

wie die zukünftige Leistungsfähigkeit des Systems präzisiert werden.

Auch die Steuerung der verschiedenen Lademodi wie Konstantstrom und Konstantspannung gehören zu den Aufgaben des BMS. Bei abweichenden Ladeständen einzelner Zellen, welche im Betrieb aufgrund fertigungsbedingter Schwankungen auftreten, sind diese für die Nutzung der gesamten Zellkapazität durch einen Ladeausgleich anzupassen.

Zu den wichtigsten Funktionen zählt die Sicherheitsüberwachung. Neben den Lade- und Entladeschlussspannungen der einzelnen Zellen müssen auch die Temperatur- und Stromgrenzen eingehalten werden. Ein Verlassen gültiger Betriebsgrenzen ist sicher zu detektieren und es müssen vordefinierte Sicherheitsmechanismen eingeleitet werden. Es ist auch zu definieren, welches Vorgehen beim Ausfall einzelner Sensoren oder Zellmodule abzuarbeiten ist.

Zuletzt muss das BMS eine einfache Diagnose des Batteriesystems sowie die Kommunikation mit anderen Steuergeräten und dem Benutzer ermöglichen [4].

In diesem Beitrag wird der modellbasierte Entwurf des Prüfstands exemplarisch anhand der Funktion zur Ladezustandsschätzung demonstriert. Diese basiert auf einem extended Kalmanfilter [4].

3.3 Anforderungen

Sowohl zur Unterstützung der modellbasierten Entwicklung des BMS als auch zur Validierung von Funktionen der RCP-Entwicklungsplattform werden umfangreiche Anforderungen an den HiL-Prüfstand gestellt. Diese ergeben sich einerseits aus den funktionalen Anforderungen des BMS und werden andererseits anhand von Modellen und Simulationen ermittelt. Ein hierzu verwendeter Simulationsaufbau wird durch Abbildung 5 illustriert. Dieser besteht aus einer Anregungsquelle, dem Batteriemodell, dem Ladezustandsschätzermodell sowie aus zwei Sensormodellen zur Messung des Strangstroms und der Klemmspannung. Die Wahl der Sensormodelle erfolgt in Abhängigkeit des Simulationsziels. Zunächst wird mittels Approximation der Sensorbandbreite durch eine Totzeit die notwendige Sensordynamik bestimmt. Anschließend können mittels Störsimulation (z. B. in Form von weißem Rauschen) weitere Anforderungen ermittelt werden.

Exemplarisch erfolgt in diesem Kapitel die Bestimmung der maximalen Zykluszeit zur Berechnung des Ladezustandsschätzers. Hierzu stellt Abbil-

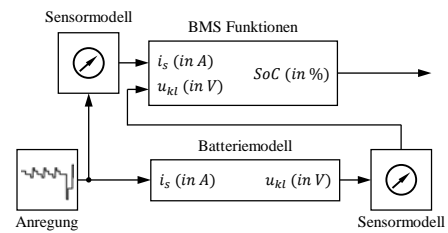


Abbildung 5: Modellbasierte Bestimmung der Anforderungen an die Sensorik.

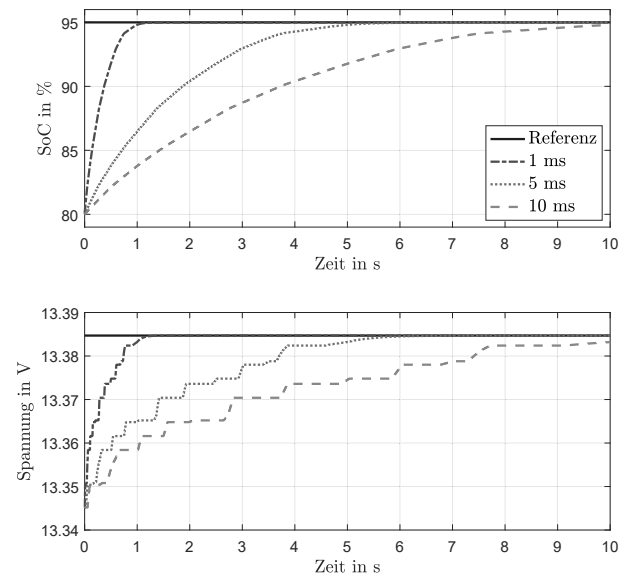


Abbildung 6: Konvergenz des Ladezustandsschätzers bei unterschiedlichen Zykluszeiten.

dung 6 vier Simulationsergebnisse zur Konvergenz des Ladezustands mit unterschiedlichen Zykluszeiten gegenüber. Der Einfluss höherer Zykluszeiten ist deutlich erkennbar. Wird eine Konvergenz innerhalb von 5 s gefordert, lässt sich anhand der Simulationsergebnisse die maximale Zykluszeit zu 5 ms bestimmen. Mittels des durch die RCP-Entwicklungsplattform generierten Funktionscodes können somit weitere Anforderungen an die Echtzeithardware abgeleitet werden.

4 Realisierung

Zur Realisierung des Prüfstands werden passende Komponenten für die drei Module gewählt sowie der gestaltungsorientierten Entwurf umgesetzt. Anschließend erfolgt die Inbetriebnahme grundlegender Module sowie

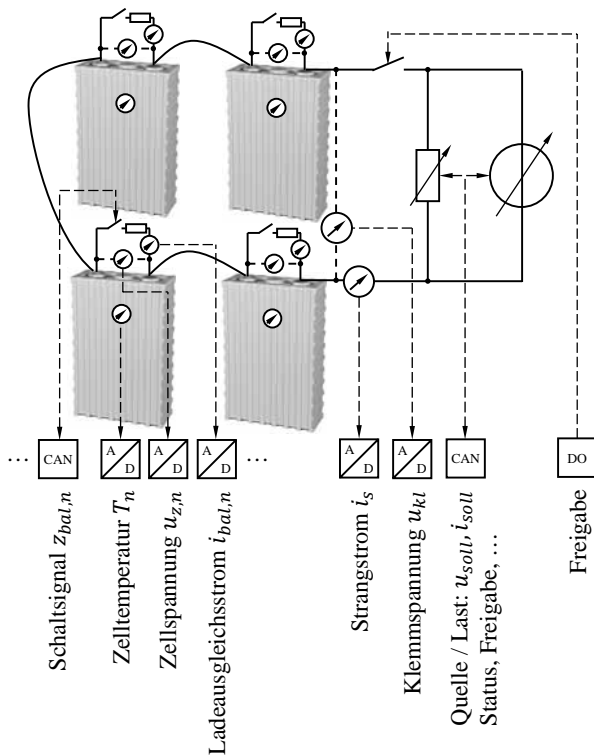


Abbildung 7: Schnittstellen zwischen Prüfstandsmodul und Echtzeithardware.

deren Integration zum HiL-Prüfstand. Zuletzt erfolgt die Verifikation der Funktionalitäten anhand von Versuchen.

4.1 Prüfstandsmodul

Das Prüfstandsmodul besteht hauptsächlich aus vier LiFePO₄-Zellen mit einer Nennkapazität von $C_n = 60Ah$, Sensorik sowie einem Schütz zur Trennung des Prüfstands vom Quell- / Lastmodul (vgl. Abbildung 7). Auf zwei frei konfigurierbaren Displayanzeigen können Daten auch ohne Anbindung an einen externen Rechner an den Benutzer ausgegeben werden.

Durch ein elektronisches Quelle- / Lastmodul lässt sich der Verlauf des Strangstromes im Rahmen der Betriebsgrenzen regeln. Durch Zellmodule erfolgt die lokale Erfassung von Messdaten der Batteriezelle zur Sicherheitsüberwachung sowie die Umsetzung des Ladeausgleichs.

4.2 Echtzeithardware

Kern der Echtzeithardware ist ein STM32F407-Mikrocontroller, auf dem die Softwaremodule ausgeführt werden. Diese Hardware verfügt über eine Vielzahl analoger und digitaler Schnittstellen, integrierte Floating Point Unit und DSP-Funktionen. Sie ist auf einer Platine zur Filterung, Skalierung und Aufbereitung von Signalen integriert. Zur weiteren Verarbeitung der Messdaten können diese sowohl in digitaler Form über einen CAN-Bus, als auch über analoge Ausgänge extern erfasst werden.

Die informationstechnische Kopplung zum Prüfstandsmodul ist durch Abbildung 7 anschaulich dargestellt. Es erfolgt die Messung der Zellspannungen und -temperaturen, der Ladeausgleichsströme sowie des Strangstroms und der Gesamtspannung. Über eine CAN-Verbindung werden der Ladeausgleich sowie das Quelle- / Lastmodul gesteuert. Über einen logischen Ausgang erfolgt die Ansteuerung des Sicherheitschützes.

4.3 Softwaremodul

Das gesamte Softwaremodul (BMS-Funktionen, Sicherheitsfunktionen, dynamische Modelle und Prüfsystemsteuerung) wird auf dem Mikrocontroller der Echtzeithardware (vgl. Kapitel 4.2) ausgeführt. Die Programmierung erfolgt in Form von Blockschaltbildern in Scilab/Xcos. Durch automatische Codegenerierung wird das Softwaremodul aus der Blockschaltbildarstellung in C-Code transformiert. Dieser wird hoch automatisiert zur Programmierung der Echtzeithardware verwendet.

4.4 Inbetriebnahme

Die Inbetriebnahme der Komponenten und Funktionen des Prüfstands erfolgt sukzessive beginnend mit der Echtzeithardware. Nicht zur Verfügung stehende Teilsysteme werden zunächst simuliert. Durch den Debugger des verwendeten Mikrocontrollers wird eine einfache Benutzerschnittstelle realisiert.

Zur Verifikation der Funktionen, Prüfung der Komponenten sowie zur Kalibrierung der Sensorik werden diverse Messungen durchgeführt. Exemplarisch stellt Abbildung 8 Simulations- und Messergebnis der Klemmspannung bei dynamischer Belastung des Batteriepakets gegenüber. Messung und Simulation stimmen sehr gut überein. Nach erfolgreicher Inbetrieb-

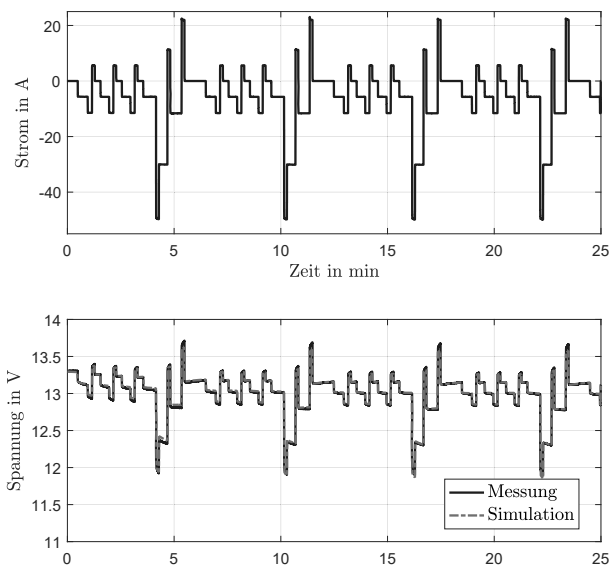


Abbildung 8: Exemplarische Messung der Klemmspannung bei dynamischer Belastung.

nahme kann der Prüfstand sowohl zur Unterstützung der Funktionsentwicklung des BMS als auch zur Validierung der Low-Cost RCP-Entwicklungsplattform eingesetzt werden.

5 Resümee und Ausblick

Der vorliegende Beitrag stellt die Entwicklung eines HiL-Prüfstands zur Funktionsabsicherung von Batteriemanagementsystemen auf Basis einer Low-Cost RCP-Entwicklungsplattform vor. Anhand eines an der Ostfalia entwickelten BMS wurden zunächst modellbasiert Anforderungen an die Funktionalitäten des Prüfstands abgeleitet und ein Konzept zur Realisierung erstellt. Dieses beinhaltet ein Prüfstandsmodul, Echtzeithardware und ein Softwaremodul. Der Prüfstand wurde sowohl funktionsorientiert als auch gestaltungsorientiert entworfen und sukzessive in Betrieb genommen. Abschließend erfolgte exemplarisch die Darstellung von Messergebnissen zur Funktionsverifikation.

Im Fokus zukünftiger Arbeiten steht im Rahmen des Forschungsprojektes LoCoRCP die Entwicklung einer geeigneten Benutzerschnittstelle. Anschließend können die Funktionalitäten der Low-Cost RCP-Plattform, von der Modellbildung über die automatische Codegenerierung bis hin zur Onlinesimulation, validiert werden.

Danksagung

Der vorliegende Beitrag wurde im Rahmen des Forschungsprojektes *Low-Cost Rapid Control Prototyping-System mit Open-Source-Plattform für die Funktionsentwicklung von eingebetteten mechatronischen Systemen* von dem EFRE Fonds der Europäischen Union unter dem Kennzeichen ZW 6-85003460 gefördert. Die Verantwortung für den Inhalt liegt bei den Autoren. Für die Förderung bedanken sich diese herzlichst.



Referenzen

- [1] Liu-Henke X, Nachtigal V. Ein integriertes Hardware-in-the-Loop-System zur Funktionsabsicherung von vernetzten Fahrwerksregelsystemen. In: *Steuerung und Regelung von Fahrzeugen und Motoren - AUTOREG 2006*, VDI-Berichte. Düsseldorf: VDI-Verl. 2006; pp. 663–677.
- [2] Liu-Henke X, Duym S. Modellgestützte Funktionsabsicherung des vernetzten mechatronischen Kraftfahrzeugs. In: *Mechatronik 2005*, VDI-Berichte. Düsseldorf: VDI-Verl. 2005; pp. 1073–1090.
- [3] Quantmeyer F, Liu-Henke X. Entwicklung eines hochflexiblen HiL-Systems zur Echtzeiterprobung des elektronischen Fahrzeugmanagements. In: *Effizienz, Präzision, Qualität: 11. Magdeburger Maschinenbau-Tage*. Magdeburg. 2013; pp. 1–10.
- [4] Liu-Henke X, Scherler S, Jacobitz S. Verification oriented development of a scalable battery management system for lithium-ion batteries. In: *Proceedings of the Twelfth International Conference on Ecological Vehicles and Renewable Energies (EVER)*. 2017; pp. 1–7.
- [5] Liu-Henke X, Jacobitz S, Göllner M, Scherler S. Systemkonzept einer durchgängigen Low-Cost RCP-Plattform. In: *Workshop der ASIM/GI-Fachgruppen STS / GMMS*. Heilbronn. 2018; pp. 163–168.
- [6] Quantmeyer F, Liu-Henke X. Modeling the Electrical Behavior of Lithium-Ion Batteries for Electric Vehicles. *Solid State Phenomena*. 2014;214:40–47.
- [7] Buccolini L, Ricci A, Scavongelli C, DeMaso-Gentile G, Orcioni S, Conti M. Battery Management System (BMS) simulation environment for electric vehicles. In: *16th International Conference on Environment and Electrical Engineering (EEEIC)*. 2016; pp. 1–6.

Modellbasierte Entwicklung zukünftiger Avioniksysteme unter Verwendung von Scilab/Xcos

David Müller^{1*}, Umut Durak¹

¹Institut für Flugsystemtechnik, Deutsches Zentrum für Luft- und Raumfahrt (DLR), Lilienthalplatz 7, 38108 Braunschweig, Deutschland; * David.Mueller@dlr.de

Abstract. Nachdem weite Bereich der Luftfahrt mittlerweile einen hohen Automatisierungsgrad erreicht haben, besteht der nächste Entwicklungsschritt in der Einführung des „intelligenten“ oder „vernetzten“ Fluges. Cyber-Physical System (CPS) Technologien gelten als unverzichtbar für die Zukunft der Luftfahrt. Mit dem Wachstum der Cyberschicht in Flugzeugen werden Anforderungen an hohen Datendurchsatz immer kritischer und dadurch entwickeln sich Multicore- bzw Multiprozessorsysteme zu einem wichtigen Thema für Forschung und Industrie. Die modellbasierte Entwicklung (oder „Model-based Design“, MBD) ist der Schlüssel zur produktiven Nutzung dieser Art von Hardware Plattformen. ARGO (WCET-Aware PaRallelization of Model-Based Applications for HeteroGeneOus Parallel Systems) ist ein Projekt, das im Rahmen des Forschungs- und Innovationsprogramms Horizon 2020 der Europäischen Union finanziert wird. Es behandelt die Parallelisierung modellbasierter Anwendungen für Multicore-Systeme. Der modellbasierte Entwicklungsprozess von ARGO beginnt mit Scilab/Xcos Anwendungsmodellen und liefert plattformspezifischen parallelen Code. Das Deutsche Zentrum für Luft- und Raumfahrt (DLR) entwickelt mit dem ARGO Workflow ein Terrain Awareness and Warning System (TAWS) als Technologiedemonstrator. Diese Präsentation wird Scilab/Xcos als modellbasiertes Design-Tool zukünftiger Avioniksysteme vorstellen, basierend auf den Erfahrungen aus dem ARGO-Projekt.

Python-Wrapper zur automatisierten Datenverarbeitung, Visualisierung und Simulation von Stromnetzen

Shuo Chen, Basem Idlbi, David Stacic, Gerd Heilscher

Hochschule Ulm, Smart Grids Forschungsgruppe, Eberhard-Finckh-Str. 11, 89075 Ulm,
chen@hs-ulm.de, idlbi@hs-ulm.de, stacic@hs-ulm.de, heilscher@hs-ulm.de

Abstract. Der Ausbau dezentraler Energiesysteme verändert die Betriebsweise der elektrischen Verteilnetze zunehmend. Dabei kommen zusätzlich und mit wachsender Tendenz, moderne Mess- und Steuerungseinrichtungen in den Verteilnetzen zum Einsatz. Für die Entwicklung und Optimierung der Betriebsweise und Planung von Verteilnetzen ist die Netzsimulation eine bewährte Methode. Dabei werden sowohl die neuen Datenquellen mit eingebunden, als auch Informationen aus bestehenden Datenbanken verarbeitet. Im Projekt ESOSEG [1] wurde ein flexibles und modular erweiterbares Analyseframework entwickelt. Dies ermöglicht die Erstellung großflächiger Verteilnetzmodelle zur Bewertung von unterschiedlichen Planungs- und Betriebsszenarien. Dazu wurden an der Hochschule Ulm für unterschiedliche Netzsimulationsprogramme zwei Wrapper mit der Skriptsprache Python entwickelt. Mit den Python-Wrappern lassen sich sowohl die Darstellung der geografisch referenzierten Netzmodelle, als auch die Modellierung und Mehrfachberechnung von Szenarien automatisiert durchführen. Dieser Beitrag soll die Konzepte der Umsetzung vorstellen, einen Vergleich der Möglichkeiten und des Aufwands aufzeigen sowie erste Ergebnisse darstellen.

Einleitung

Die voranschreitende Implementierung von stromerzeugenden Anlagen auf Basis von erneuerbaren Energien in den deutschen Verteilnetzen (VN) geht auch zunehmend einher mit einer Implementierung von intelligenten Messsystemen (iMSys).

Parallel liefert die detaillierte Datenlage in den Verteilnetzen neue Möglichkeiten für deren Planung und Betrieb. In den Übertragungsnetzen ist die Netzsimulation eine verbreitete Methode. In VN ist durch die Größe der Netze die Modellierung von umfangreichen Netzmodellen eine komplexe Aufgabe [2]. Variantenrechnungen in

der Netzplanung, das Einpflegen von Messdaten sowie Stammdaten von neuen Technologien, wie z. B. Batterien oder elektronischen Ladesäulen in bestehende Netzmodelle, sind nur mit automatisierten Software-Tools in angemessenen Zeiträumen zu bewältigen. Solche Software-Tools, die die bestehende Netzsimulationssoftware um entsprechende Funktionalitäten erweitern können, werden Wrapper genannt. Der Wrapper umgibt in der Softwarearchitektur die eigentliche Simulationssoftware und ermöglicht zum einen die Automatisierung, sowie zum anderen die Schaffung von Verbindungen zu externen Datenquellen oder anderen Programmen. In diesem Beitrag sollen Wrapper für die zwei Stromnetzsimulationsprogramme DIgSILENT PowerFactory (PowerFactory) [3] und Siemens PSS[®]SINCAL (SINCAL) [4] beschrieben und verglichen werden.

1 Simulation von Stromnetzen

Wie einleitend bereits erwähnt, bietet die Simulation eine Reihe von Vorteilen für VN. Verteilnetzbetreiber (VNB) müssen Netzkomponenten, sogenannte Assets, wie z. B. Kabel und Transformatoren, vorausschauend planen und beschaffen. Da sich mit der Systemtransformation des Stromsektors neue Technologien innerhalb von wenigen Jahren etablieren können, ist es umso wichtiger, fundierte Analysen vor dem Ausbau der Netze durchzuführen. Prominentes Beispiel für eine neue Technologie in den Stromverteilnetzen ist die Photovoltaik (PV). Für die nahe Zukunft könnte z. B. die Elektromobilität mit Ladesäulen ebenfalls eine große Unsicherheit in der Planung bringen. Aber auch bei den Assets müssen neue Technologien für neue Anwendungsgebiete, wie z. B. Regelbare Ortsnetztransformatoren (RONT) in der Planung berücksichtigt werden. Daraus ergeben sich zusammen mit den historisch gewachsenen VN eine Vielzahl von möglichen Szenarien und Variationsmöglichkeiten.

Auch im Betrieb werden durch moderne Informations- und Kommunikationstechnologien (IKT) zunehmend detaillierte Daten zu den Netzständen verfügbar. Da sich an diese, auf Grund der langfristigen Planungshorizonte und der hohen gesellschaftlichen Relevanz der Stromnetze, auch hohe Anforderungen an die Sicherheit und Zuverlässigkeit ergeben, schreitet eine großflächige Ausrollung in Deutschland bisher nur mit Verzögerungen voran [5]. Lösungen zur Implementierung dieser neuen Datenströme in Planungstools (wie beispielsweise die Stromnetzsimulation) müssen für einen reibungslosen Start mit neuer IKT bereits heute erarbeitet werden. Auch hier müssen längere Entwicklungszeiträume berücksichtigt werden. Dabei bieten diese neuen digitalen Technologien zusammen mit der Dezentralisierung des Energiesystems auch neuen Möglichkeiten für VNB. So können z. B. durch die Optimierung des Netzbetriebs Netzverluste minimiert werden, die Ausnutzung der Netze erhöht und ungewollte Netzzustände vorausschauend erkannt werden.

Die Netzsimulation in Kombination mit einem Wrapper, wie in diesem Beitrag beschrieben, bietet darüber hinaus weitere Vorteile, die tief in die bestehenden Strukturen von Daten von VNB eingreifen können. Durch die automatisierte Datenverarbeitung mit einem Wrapper kann eine systematische und effiziente Datenbereitstellung für einen Austausch zwischen einzelnen Abteilungen (Geo-Informationssysteme, Netzplanung, usw.) innerhalb eines VNB erfolgen. Dabei könnten auch Netze unterschiedlicher Sektoren gekoppelt werden. In der Praxis werden solche Ansätze zum Teil schon heute betrieben. Jedoch ist eine bidirektionale Simulation solcher Ansätze bei VNB bisher die Ausnahme. Potenziell wäre auch eine Quasi-Echtzeitsimulation möglich, welche die Auswirkungen des Anlagenbetriebs in einem Netzabschnitt detailliert bewerten könnte.

2 Systemaufbau der Simulationsumgebung

Die Konstruktion des gesamten Systems bzw. die Implementierung eines Wrappers wird an eine Simulationsumgebung angepasst. Dort sollen Datenverarbeitung und Netzberechnung automatisiert durchgeführt werden. Auf Basis der in Kapitel 1 beschriebenen Einsatzmöglichkeiten wurde mit der Programmiersprache Python eine entsprechende Simulationsumgebung aufgebaut. Der Systemaufbau ist in Abbildung 1 schematisch dargestellt.

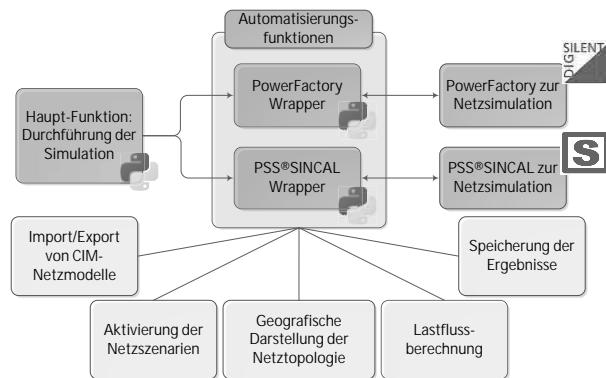


Abbildung 1: Systemaufbau der Simulationsumgebung; Die blauen Kästchen beschreiben die Software und deren Schnittstellen. Die gelben Kästchen beschreiben Automatisierungsprozesse und Funktionen.

Orientiert an den Einsatzmöglichkeiten verfügen die Wrapper zu den beiden Netzsimulationstools über folgende Fähigkeiten:

- Standardisierter Datenaustausch mittels Einlesen und Exportieren eines Netzmodells.
- Anzeigen des Netzmodells in georeferenzierter Darstellung sowie Hinterlegen der Netzdarstellung mit der zugehörigen Landschaftskarte.
- Erstellung eines Betriebsfalles durch Import von Netzszenarien (Lasten und Einspeisung werden bei der Generierung von Betriebsfällen betrachtet).
- Durchführung von Lastflussberechnungen mit Vergleich von verschiedenen Szenarien.
- Abspeicherung der Ergebnisse für weitere Analysen
- Generierung der Log-Dateien

Durch die Ausführung der Hauptfunktion wird zunächst der Wrapper initialisiert. In einem weiteren Schritt, während der laufenden Simulation, werden die Daten im Wrapper durch neue Eingaben einzelner Werte oder mit Zeitreihen aktualisiert. Solche Initialisierungen und Aktualisierungen wurden als mögliche Anschlusspunkte für externe Services wie Datenabfrage und Sollwertvorgabe umgesetzt. Dadurch wird ein Datenaustausch in Echtzeit, die Ermittlung von Ergebnissen der Netzberechnungen oder auch die Übergabe von neuen Netzszenarien im Wrapper realisiert.

3 Standardisierter Datenaustausch

Wie in Kapitel 1 bereits erwähnt, ist ein standardisierter Datenaustausch zwischen verschiedenen Fachbereichen

der Netzbetreiber eine der Herausforderungen beim Betrieb von Netzen mit hohem Anteil dezentraler Erzeugung. Allerdings haben Netzdatensoftwareprogramme üblicherweise proprietäre Datenmodelle, die spezifisch für bestimmte Programme entwickelt wurden. Für PowerFactory wurden beispielsweise der PowerFactory-Datentyp (.PFD) und DiGSILENT (.DGS) verwendet. Im Gegensatz dazu stellt das Common Information Model (CIM), basierend auf den Standards IEC 61970, IEC 61968 und IEC 62325, ein standardisiertes Austauschdatenmodell für verschiedene IT-Systeme dar. Als selbstbeschreibendes Datenmodell bietet CIM eine vereinfachte Integration von neuen Systemen. Für den Datenaustausch zwischen Übertragungsnetzbetreibern wurde basierend auf CIM das CGMES (Common Grid Model Exchange Specification) durch den Verband Europäischer Übertragungsnetzbetreiber (ENTSO-E) entwickelt. Die CIM-Standards werden ständig weiterentwickelt. Die unterschiedlichen CIM-Versionen sind jedoch nicht immer abwärtskompatibel. Dadurch ergibt sich in vielen Fällen die Notwendigkeit für einen Versionsadapter. Im Projekt ESOSEG wurde eine Datendrehscheibe für die IT-Systeme von VNB entwickelt und ein dafür entsprechendes CIM-Datenmodell erstellt [6].

4 Implementierung der Python-Wrapper

Als Vorbereitung für die Implementierung der Wrapper sind die Eigenschaften sowie Besonderheiten der Simulationsprogramme auf Basis der Dokumentationen und Handbücher zu analysieren [7], [8]. Die nachfolgende Tabelle gibt einen allgemeinen Überblick über die Funktionalitäten der Stromnetzsimulationsprogramme, die in diesem Beitrag betrachtet werden.

Eigenschaften	DiGSILENT PowerFactory	Siemens PSS®SINCAL
Schnittstellen	Python Bibliothek vorhanden	COM-Objekt (Python Bibliothek nicht vorhanden)
Import von Netzmodellen	PFD, DGS, CIM, PSS/E, PSS/U, SINCAL Database und weitere 7 Datenformate [3]	SIN, DGS, CIM, PSS/E, Excel, SINCAL Database und weitere 5 Datenformate [4]
Import von Varianten und Szenarien	in verschiedenen Formaten möglich	nur in Form einer XML-Datei möglich

Szenarien-Berechnung	pro Berechnung ein Szenario	Mehrfachberechnung von mehreren Szenarien möglich
Direkter Zugriff auf Netz-Datenbank	während der Automatisierung ausgeschlossen	Durch Datenbankabfrage möglich
Echtzeit-Simulation	Möglich, grundlegende Schnittstelle vorhanden	Möglich, höherer Aufwand für die Erweiterung

Tabelle 1: Eigenschaften der beiden Simulationsprogramme, die angesichts der Anwendungsfälle für die Implementierung der Wrapper wichtig sind

Um die Konfiguration der Netzsimulation und die Netzberechnung selbst zu automatisieren ist eine Schnittstelle in der Software notwendig über die das Netzberechnungsprogramm von einer anderen Software bedient werden kann. Generell sollte eine solche Schnittstelle vom Hersteller des Simulationsprogramms, unabhängig von Python als Programmiersprache, bereitgestellt werden, sodass ein allgemein gültiges Verfahren zur Steuerung des Programmes mit vertretbarem Aufwand realisiert werden kann. Auf Grund der sehr unterschiedlichen Möglichkeiten für die Automatisierung der beiden Netzberechnungsprogramme musste die Python-Schnittstelle in beiden Simulationsprogrammen unterschiedlich implementiert werden. Als Konsequenz wurden zwei Wrapper mit ähnlicher Architektur aber verschiedenen Funktionsblöcke als zwei separate Instanzen entwickelt.

PSS®SINCAL: Die Automatisierungsfunktionen orientieren sich hier an dem Windows COM-Objekt, welches den Verbindungsaufbau zur Datenbank, die Anpassung der Datenattribute von Netzkomponenten sowie die Parametrierung für Berechnungen ermöglicht. Insgesamt bietet SINCAL ein relativ großes Umfeld zur Umsetzung der Automatisierung. Ein Nachteil davon ist, dass viele Zusatzfunktionen wie Protokollierung und Export der Ergebnisse selbst programmiert werden müssen. Die Basisfunktionen des SINCAL-Wrappers sowie der Prozessablaufplan sind in Abbildung 2 dargestellt. Bisher befindet sich diese von Siemens zur Verfügung gestellte Python-Schnittstelle noch in einer Entwicklungsphase. Für die Umsetzung stehen lediglich für Teilbereiche einige Beispielskripte zur Verfügung. Die Skripte selbst, sowie die relevante Dokumentation wird vom Hersteller bisher nur in der Visual-Basic Skriptsprache zur Verfügung gestellt.

Diese musste während der Implementierung in die Python-Skriptsprache umgewandelt werden. Darüber hinaus wird ein direkter Zugriff auf die Netzprojekt-Datenbank, in diesem Beitrag eine MS-Access-Datenbank, für spezielle Anwendungsfälle benötigt. Bei der Implementierung wurde festgestellt, dass die Automatisierungsfunktionen des SINICAL-Wrappers in einer Multi-Thread-Umgebung, wegen Konflikten mit verzögerten Lizenzfreigaben, nicht reibungslos funktionierten. Bei der Kopplung an externe Server muss der Aufruf des SINICAL-Wrappers deshalb angepasst werden.

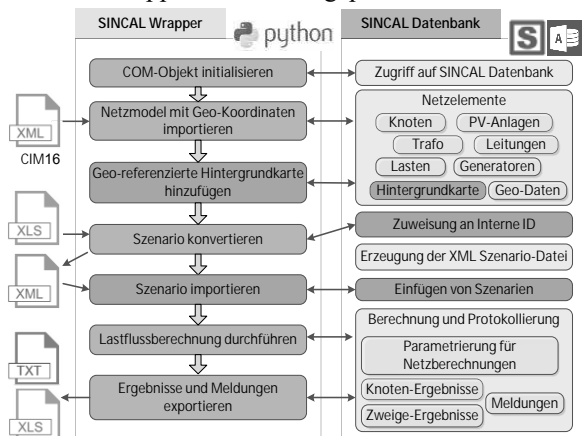


Abbildung 2: Ablauf der Automatisierung mittels SINICAL-Wrapper, Zusatzfunktionen durch direkte MS-Access Datenbankabfragen sind rot gezeichnet.

Als Input für den SINICAL-Wrapper standen ein CIM 16 (CIM Version 16) Netzmodell und Netzszenarien zur Verfügung. Nach dem Import des Netzmodells und der Durchführung der Netzberechnung wird die Datenbank des Netzprojekts entsprechend aktualisiert. Die Ergebnisse aus den Netzberechnungen sowie aus weiteren Szenarien-Berechnungen werden parallel aufgezeichnet und in Form einer Excel-Tabelle mit Zeitstempel exportiert. Diese wird für die Kopplung mit anderen Berechnungsprogrammen oder zur Rückkopplung der Netzzustände in SINICAL verwendet.

PowerFactory: Die grundlegenden skriptbasierten Automatisierungsfunktionen für PowerFactory wurden als Python-Bibliothek entwickelt und sind bei der Installation des Programms bereits einsatzfähig vorhanden. Der Funktionsaufruf ist im Rahmen der Python-Bibliothek klar definiert. Dadurch können Datenimport, -export sowie Berechnungen mit geringem Aufwand realisiert werden. Im Gegensatz zur SINICAL-Automatisierung ist

die Netzprojekt-Datenbank von PowerFactory in der laufenden Simulation von außen nicht zugänglich. Ein Prozessablaufplan für die Automatisierung von PowerFactory ist schematisch in Abbildung 3 dargestellt.

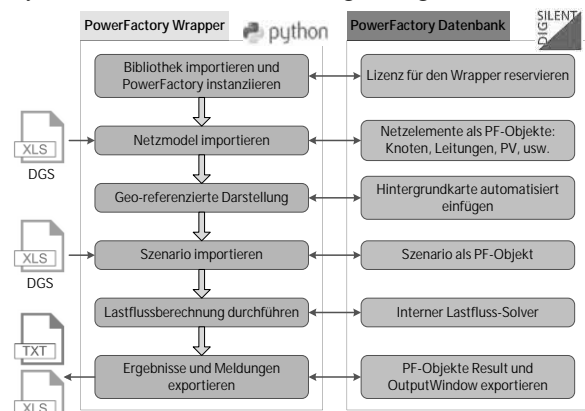


Abbildung 3: Ablauf der Automatisierung mittels PowerFactory-Wrapper; im Unterschied zum SINICAL-Wrapper sind in PowerFactory die zu importierenden Netzmodelle und Netzszenarien im DGS-Excel-Format.

5 Test und Analyse

Die beiden Wrapper wurden in mehreren Schritten sowohl auf Funktionalität, als auch auf Plausibilität getestet.

Testphase 1 - Funktionstest: Zu Beginn wurden beide Wrapper mit einem einfachen Netzmodell erprobt. Dabei wurden Basisfunktionen wie Import von Netzmodellen und Netzszenarien sowie automatische Lastflussberechnungen in einer „Stand-Alone“-Umgebung getestet. Fehler bei der Implementierung wurden sondiert und behoben. Beispielsweise wurde die geographische Darstellung eines Kerber-Landnetzes [9] auf Basis von GPS-Koordinaten jeweils in Abbildung 4 und Abbildung 5 für beide Simulationsprogramme umgesetzt. Die Geo-Koordinaten referenzierten sich auf ein exemplarisch ausgewähltes Gebiet, dasselbe Netzmodell wurde durch die zwei Wrapper in beiden Simulationsprogrammen eingelesen. SINICAL verwendet eine eigene Umrechnungsmethode für die Projektion und Verkrümmung, infolge derer die Geo-Daten nach dem Import automatisch in SINICAL korrigiert wurden. Aufgrund dessen sind „sehr“ kleine Abweichungen im Meter-Bereich auf der Karte zu erkennen. Generell wurden aber alle Knoten in beiden Programmen mit hoher Genauigkeit richtig platziert.

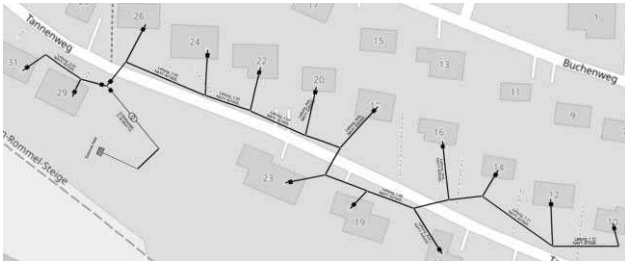


Abbildung 4: Georeferenziertes Netzmodell auf Basis des Kerber-Landnetzes in PowerFactory; die zwei Abgänge haben insgesamt 14 Hausanschlüsse, an jedem Hausanschluss ist eine PV-Anlage sowie eine Last angeschlossen.

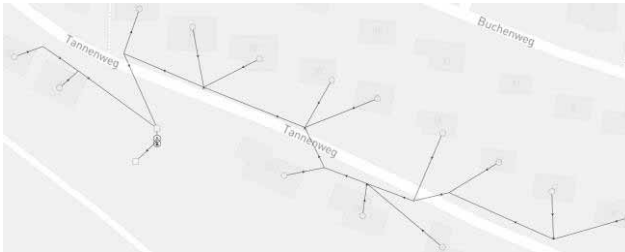


Abbildung 5: Georeferenzierte Darstellung desselben Kerber-Landnetzes in SINCAL

Testphase 2 - Plausibilitätstest: In dieser Testphase wurden die Ergebnisse aus den automatisierten Berechnungen von beiden Wrappern verglichen. Ein solcher Test war notwendig, da die zwei Simulationsprogramme das gleiche Netzmodell, allerdings mit verschiedenen Dateitypen einlesen. Dementsprechend musste das Netzmodell mindestens einmal durch einen Adapter konvertiert werden. In diesem Beitrag sollten zwei Dateien mit demselben Netzmodell, jeweils im Format DGS (.XLS) und CIM 16 (.XML) den beiden Wrappern bereitgestellt werden. Die Plausibilität der automatisiert eingelesenen Netzmodelle sowie die Lastflussberechnungen lassen sich miteinander vergleichen.

Am Beispiel des Kerber-Netzes wurden alle Knotenspannungen nach der konvergierten Lastflussberechnung verglichen. Die Abweichungen sind in Abbildung 6 dargestellt. Für diese Berechnung wurden initiale PV-Einspeisungen und Hausverbräuche als eingeschränkte Zufallszahlen festgelegt. Nach dem Vergleich lag die maximale Abweichung bei 0.00033 pu (per-Unit), der RMSE (Root Mean Square Error) bei 0.00026 pu. Da die hier beschriebenen Netzsimulationsprogramme unterschiedliche Modelle für Netzkomponenten sowie verschiedene Berechnungsverfahren haben, sind solche Abweichungen vernachlässigbar.

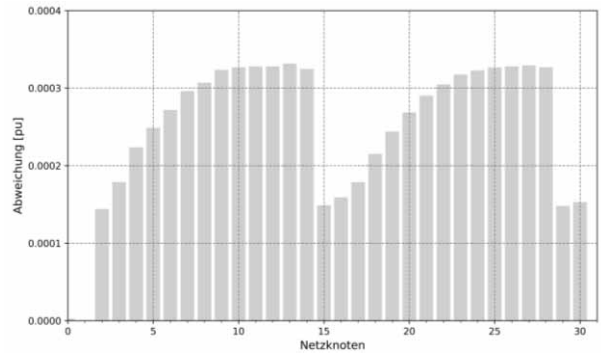


Abbildung 6: Abweichungen aller Knotenspannungen (Bezugsgröße auf Nennspannung) im Kerber-Netz; die Ergebnisse stammen aus automatisierten Lastflussberechnungen durch beide Wrapper.

Im nächsten Schritt wurden die beiden Wrapper mit einem umfangreicheren Netzmodell überprüft. Dazu stand ein Netzmodell eines realen Testgebiets mit komplexer Netztopologie zur Verfügung. Dieses stellt sich durch über 140 Hausanschlüsse, mehr als 250 Haushaltsverbraucher und 25 PV-Anlagen dar. Mit diesem Netzmodell wurde Szenario-Berechnung mit beiden Wrappern automatisiert durchgeführt. In Abbildung 7 wird die Verteilung der Abweichungen aller Knotenspannungen gezeigt. Die maximale Abweichung und der RMSE betrug jeweils 0.00028 pu und 0.000075 pu. Als Referenz ist die Messgenauigkeit $\pm 0.1\%$ der Nennspannung (0.001 pu) in der Norm IEC 61000-4-30 Class A zum Power-Quality Monitoring angegeben [10]. Somit konnten beide Wrapper mit demselben rechenbaren Netzmodell gleiche Ergebnisse automatisiert berechnen. Dadurch konnte die Anforderung an die Netzsimulation mit standardisiertem Datenaustausch erfüllt werden.

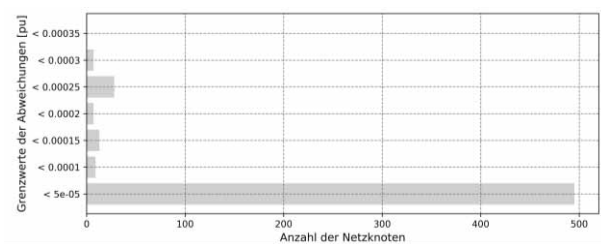


Abbildung 7: Statistische Verteilung der Spannungsabweichungen aller Knoten (Bezugsgröße auf Nennspannung) aus Lastflussberechnungen mit automatisiert importiertem Netzscenario von beiden Simulationsprogrammen.

Testphase 3 - Integrationstest: Im Rahmen des Forschungsprojekts ESOSEG wurde der SINCAL-Wrapper zur Datenbeschaffung für die Abteilung „Netzplanung“ an einen Web-Server gekoppelt. Nach einer

Eingrenzung in Netzgebiete durch weitere Softwareservices wurden die von SINCAL benötigten Eingangsdateien bereitgestellt. Dies betrifft insbesondere die CIM 16-Netzmodelle und -szenarien. Anschließend wurde der SINCAL-Wrapper von dem Web-Service aufgerufen und die Automatisierungsfunktionen konnten schrittweise durchgeführt werden. Nach der erfolgreichen Systemintegration wurden die Fähigkeiten des Wrappers getestet. Dabei wurde die komplette Automatisierungskette von der Auswahl des Netzgebiets bis zur Durchführung und Protokollierung der Berechnungen in diesem gekoppelten und modularen System einbezogen.

6 Zusammenfassung und Ausblick

Zur automatisierten Datenverarbeitung, Visualisierung und Stromnetzsimulation wurden zwei auf der Programmiersprache Python basierende Wrapper für zwei Netzsimulationsprogramme erfolgreich implementiert und für konkrete Anwendungsfälle getestet. Durch den Standard CIM kann das Netzmodell leicht von beiden Wrappern in Planungstools einbezogen werden und ermöglicht damit eine geografisch referenzierte Visualisierung für die jeweilige Netzplanungsabteilung.

Unter Betrachtung der Eigenschaften beider Simulationsprogramme wurde sowohl der Umfang der Funktionalitäten als auch die Standardkompatibilität während der Implementierung berücksichtigt. Daraus lassen sich folgende Ergebnisse zusammenfassen:

- Die Schnittstellen zur automatisierten Stromnetzsimulation werden von den Herstellern zur Verfügung gestellt, allerdings ist der Implementierungsaufwand für spezifische Anwendungen noch relativ hoch.
- Beide Simulationsprogramme können zwar Standarddatenmodelle wie z. B. CIM, jedoch unterscheiden sich Details aufgrund verschiedener CIM-Versionen oder CIM-Profile. Daraus resultiert ein zusätzlicher Aufwand für die Entwicklung von Adaptern für den Datenimport.
- Die automatisierten Berechnungen beider Wrapper können Ergebnisse mit vernachlässigbaren Abweichungen liefern. Die Abweichungen wurden durch unterschiedliche Modellierungen der Netzkomponenten sowie Berechnungsverfahren verursacht und haben keinen Einfluss auf die Simulation.
- Trotz kleiner Abweichungen bei den Plausibilitätstests sind beide Wrapper einsatzfähig. Auch der modulare

Service für die Automatisierungsanwendungen mit dem SINCAL-Wrapper konnte im ESOSEG Framework erfolgreich integriert und getestet werden.

- Mithilfe der Wrapper werden zeitaufwändige manuelle Arbeiten eingespart. Auf Grundlage systematischer und effizienter Datenbereitstellung können die VNB die Netzplanung für VN mit zunehmendem Anteil von dezentralen Energieanlagen und steuerbaren Lasten optimieren. Unnötiger Netzausbau kann so minimiert, Netzstörungen früher erkannt werden.
- Die Wrapper bieten umfangreiche Möglichkeiten für erweiterte Anwendungen in der Zukunft. Beispiel dafür sind die Quasi-Echtzeitsimulation und die Sektorenkopplung in der Netzsimulation.

Fördergeber Hinweis

Diese Veröffentlichung entstand im Rahmen des Förderprogramms ESOSEG: "Environment for Simulation, Operation and Optimization of Smart Energy Grids." (Förderkennzeichen des Hauptautors: 0325811C).

Literaturverzeichnis

- [1] ESOSEG - Environment for Simulation, Operation and Optimization of Smart Energy Grids, online: <http://esoseg.in.tum.de> (zuletzt geprüft: 04.12.2018)
- [2] Abschlussbericht vom Forschungsprojekt Nr. 44/12 „Moderne Verteilernetze für Deutschland“ (Verteilernetzstudie im Auftrag von BMWi), 12.09.2014
- [3] PowerFactory Applications, online: <https://www.dig-silent.de/en/powerfactory.html> (zuletzt geprüft: 10.01.2019)
- [4] PSS@SINCAL - Planung von Erzeugungs-, Übertragungs-, Verteilungs- und Industrienetzen, online: www.siemens.de/pss-sincal (zuletzt geprüft: 10.01.2019)
- [5] Energate, online: <https://www.energate-messenger.de/news/187056/bsi-kuendigt-rollout-fuer-ende-januar-2019-an> (zuletzt geprüft: 07.12.2018)
- [6] Dominik Ascher, Christoph Kondzialka, Towards model-driven CIM-based data exchange for DSOs, The 7th DACH+ Conference on Energy Informatics, 10.10.2018
- [7] Dokumentation „Bedienung“ des Simulationsprogramms PSS@SINCAL 15.0
- [8] DIgSILENT PowerFactory 2018, Python Function Reference
- [9] Georg Kerber. Aufnahmefähigkeit von Niederspannungsverteilsnetzen für die Einspeisung aus Photovoltaikkleinanlagen [Dissertation]. Fakultät für Elektrotechnik und Informationstechnik, TU München, 21.03.2011
- [10] IEC 61000-4-30: Testing and measurement techniques - Power quality measurement methods, International Electrotechnical Commission, 2015

Simulation der Zwischenkreisregelung eines DC-gekoppelten Erzeuger-Speicher-Systems mit Schnellademöglichkeit

Tobias Fricke^{1*}, Florian Wolgast¹, Günter Tareilus¹, Regine Mallwitz¹

¹Inst. für Elektrische Maschinen, Antriebe und Bahnen, Technische Universität Braunschweig, Hans-Sommer-Str. 66, 38106 Braunschweig, Deutschland; *tobi.fricke@tu-bs.de

Abstract. In diesem Dokument wird die Entwicklung eines DC-gekoppelten Erzeuger-Speicher-Systems mit DC-Schnellademöglichkeit beschrieben. Dabei wird besonderes Augenmerk auf die zur Unterstützung des Entwicklungsprozesses durchgeführten Simulationen gelegt. Dazu wird zunächst das genutzte Simulationsmodell diskutiert. Anschließend werden untersuchte Regelungsmethoden vorgestellt und daraufhin entwickelte Regelungstopologien verglichen. Abschließend wird das System bezüglich der Ausbildung von kapazitiven Ableitströmen untersucht. Dabei werden auch mögliche Gegenmaßnahmen berücksichtigt. Ziel ist, durch diesen Prozess eine Regelungstopologie und Teile der Hardware auswählen zu können.

Einleitung

Im Rahmen der Energiewende nimmt die Anzahl der dezentralen Erzeugereinheiten im Niederspannungsnetz beständig zu. Ist diese Zunahme erneuerbarer Energiequellen im Grunde positiv zu werten, so stellt sie jedoch das Versorgungsnetz vor neue Herausforderungen. Als ein möglicher Ansatz diesen Herausforderungen zu begegnen bieten sich PV-Speicher-Systeme in Kombination mit einem Smart-Grid an [1]. Solche Systeme sind bereits am Markt verfügbar. Oftmals bestehen sie jedoch aus mehreren, voneinander weitestgehend unabhängigen Systemen. Während Photovoltaiksysteme mittlerweile schon einige Jahre bis Jahrzehnte auf dem Markt sind, so ist die Kombination mit Speichersystemen im Einfamilienhaushalt noch vergleichsweise neu [2]. Dies führt dazu, dass die meisten Speichersysteme später als Ergänzung zum bereits vorhandenen PV-System angeschafft werden. Vorhandene PV-Systeme sind prinzipbedingt meist netzgekoppelte Systeme. Hier zeigt sich ein wesentlicher Nachteil

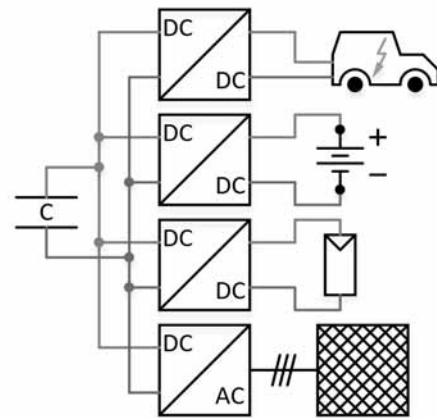


Abbildung 1: Schematische Darstellung des Gesamtsystems

solcher Anlagen. Sollen diese Energie vom PV-System in das zugekaufte, ebenfalls netzgekoppelte Speichersystem transferieren, so muss hier der Umweg über das angeschlossene Netz genommen werden. Weiterhin weichen die Spannungen von PV-Anlage und Speicherbatterien meist stark voneinander ab. Arbeiten die meisten PV-Anlagen bei Spannungen über 600 V, so bestehen die Speicherbatterien beispielsweise aus verschalteten Blei-Akkumulatoren mit Nennspannungen um 48 V [2]. Dies bedingt eine Anpassung der Spannung innerhalb der jeweiligen Geräte an das Netzniveau, bevor die Gleichspannung wechselgerichtet werden kann und sorgt daher für eine zusätzliche Wandlerstufe. Eine Möglichkeit, diesen Nachteilen zu begegnen, besteht in einer Gleichspannungskopplung der Systeme. Dabei werden die Spannungen der einzelnen Systeme über Gleichspannungswandler an eine gemeinsame Zwischenkreisspannung angepasst, von welcher aus die Energie bei Bedarf über einen Wechselrichter ins Netz gelangt. Eine Möglichkeit,

diese Anpassung bei der Batterie vorzunehmen, ist die Verwendung von 2nd-Life-Traktionsbatterien. Diese Möglichkeit wird bereits diskutiert [3]. Hier müssen für den systeminternen Energietransfer weniger Wandlerstufen passiert werden, was den Wirkungsgrad potentiell erhöht. Weiterhin kann bei entsprechender Auslegung an allen Schnittstellen der gleiche Wandler verwendet werden, was die Systemkomplexität zusätzlich verringert.

Das Verbundprojekt NetProsum2030 hat unter Anderem zum Ziel, ein solches System zu entwickeln und aufzubauen. Das Gesamtsystem wird über vier Schnittstellen verfügen. Das Drehstromnetz, eine PV-Anlage, eine Speicherbatterie bestehend aus einer 2nd-Life-Traktionsbatterie und einer DC-Schnellademöglichkeit für ein Elektrofahrzeug. Abbildung 1 zeigt das System. Auch hier ist das Ziel eine hohe Modularität des Systems zu erreichen bei gleichzeitig hohem Wirkungsgrad und verringertem Bauvolumen. Zur Auslegung des gesamten Systems wurden verschiedene Themengebiete mithilfe von Simulationen untersucht. Die Simulationen, die dabei aufgetretenen Probleme sowie die erzielten Ergebnisse werden im Weiteren thematisiert. Dabei wird zunächst das simulierte System detaillierter beschrieben.

1 Systembeschreibung

Die einzelnen Gleichspannungswandler zur Kopplung der angeschlossenen Geräte werden dabei als beidseitig gefilterte Synchronwandler aufgebaut (vgl. Abbildung 2). Durch diese Topologie wird ein bidirektionaler

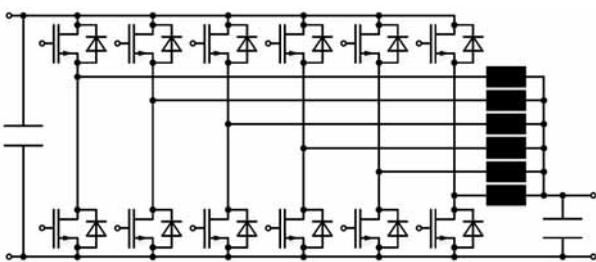


Abbildung 2: Sechsheisiger Synchronwandler mit beidseitiger Filterung

Energiefluss ermöglicht. Ein solcher Wandler wird aus sechs Phasen bestehen, die Gesamtleistung beträgt 20 kW. Eine höhere Schnittstellenleistung, beispielsweise bei Verwendung einer größeren PV-Anlage, kann

durch eine Parallelschaltung mehrerer dieser im Folgenden Leistungsmodule genannten Wandler erreicht werden. Zur Realisierung der Leistungsmodule werden Siliziumcarbid-Leistungshalbleiter verwendet. Diese ermöglichen es durch die geringeren Schaltverluste den Wirkungsgrad zu optimieren sowie das Bauvolumen zu verringern ([4]). Durch die Wahl der Zwischenkreisspannung auf 800 V sowie einer Anpassung der Schnittstellenspannungen in den Bereich zwischen 300 V und 800 V ist es möglich, für alle Schnittstellen das oben beschriebene Leistungsmodul zu verwenden. Somit wird ein hoher Grad an Modularität im Gesamtsystem erreicht.

Wie bereits beschrieben ist die Leistungselektronik als solche weitestgehend ausgelegt. Das Ziel der Simulationen ist es nun, eine regelungstechnische Systemtopologie zu finden, mit welcher sich das System möglichst optimal regeln lässt. Dabei soll die Modularität erhalten bleiben, die Systemkomplexität sowie die Kosten für das System gering gehalten werden. Weiterhin sollen Fragen bezüglich der Ableitstromproblematik bei transformatorlosen Designs untersucht werden. Dies kann zu Störungen bis hin zur Gefährdung von Personen führen und muss daher für dieses neuartige System ebenfalls betrachtet werden ([5]).

Um nun mehrere Systemtopologien miteinander vergleichen zu können, muss die verwendete Hardware bekannt sein. Diese hängt jedoch sowohl von der verwendeten Systemtopologie selbst als auch beispielsweise von der Größe des Zwischenkreises ab, da dieser dessen Zeitkonstante definiert. Die einzelnen Teilprobleme sind demnach stark voneinander abhängig. Um dennoch auf praktikablem Weg Aussagen treffen zu können, wird das Problem in mehrere Teilprobleme separiert. Dabei wird folgende Vorgehensweise genutzt:

1. Betrachtung möglicher Regelungsmethoden an einem idealisierten System
2. **Simulationsergebnis:** Auswahl einer geeigneten Methode
3. Untersuchung verschiedener Systemtopologien unter Berücksichtigung der gefundenen Regelungsmethode und einer Vorauswahl geeigneter Hardware
4. **Simulationsergebnis:** Auswahl von Systemtopologie und zugehöriger Hardware
5. Simulation des gefundenen Gesamtsystems mit

unterschiedlichen Zwischenkreiskapazitäten unter Berücksichtigung normativer Fehlerfälle

6. Simulationsergebnis: Minimal benötigte Zwischenkreiskapazität des Systems

Da die Ableitströme des Systems von der gewählten Regelungsstruktur weitgehend unabhängig sind, werden diese separat simuliert.

Für all diese Arbeitspunkte ist zunächst die nachfolgend beschriebene Erstellung eines Simulationsmodells erforderlich.

2 Simulationsmodell

Als Simulationsumgebung wird ANSYS Simplorer genutzt. Im Gegensatz zu anderen üblichen Tools wie MATLAB Simulink oder LTSpice ist es mit Simplorer möglich, sowohl die schaltungstechnischen Komponenten inklusive des Schaltverhaltens als auch die regelungstechnischen Aspekte auf systemtheoretischer Ebene zu simulieren. Dazu wird zunächst das Leistungsmodul selbst modelliert.

2.1 Leistungsmodul

Eine erste Simulation eines einzelnen sechsphasigen Leistungsmoduls zeigt stark asymmetrische Phasenströme. Dies lässt sich auf das ideale Schaltverhalten der Schalter und weitere fehlende parasitäre Komponenten in Kombination mit den Anfangszuständen des Systems zurückführen. Ein solches Verhalten führt dazu, dass sich die Ströme in einzelnen Halbbrücken im Extremfall sogar umkehren können und spiegelt daher die Realität lediglich noch im geregelten Summenstrom wieder. Um dieses simulative Problem zu lösen werden folgende Ansätze verfolgt:

- Nachbildung des realen Schaltverhaltens der Leistungshalbleiter
- Bedämpfung des Ausgangsfilters des Synchronwandlers
- Implementierung einer Einzelphasenregelung
- Näherung des Leistungsmoduls durch eine einzelne Halbbrücke

Es stellt sich heraus, dass sowohl ein realitätsnäheres Schaltverhalten als auch eine Bedämpfung zu einer Besserung führen. Ganz eliminieren lässt sich das Problem aber nur durch eine Einzelphasenregelung. Diese sorgt prinzipbedingt für gleiche Ströme in allen Phasen.

Sie steigert jedoch die Rechenzeit der Simulation auf ein unpraktikables Maß und wird somit nicht weiter verfolgt.

Mit einer Näherung des Leistungsmoduls als eine einzelne Halbbrücke lässt sich das Problem umgehen. Da das System in der Lage sein wird, einzelne Halbbrücken abzuschalten, entspricht dies auch einem realen Betriebsfall. Die Näherung besteht nun darin, dass bei Leistungen über $\frac{1}{6} \cdot P_{\text{Nenn}}$ weiterhin nur eine Halbbrücke aktiv ist. Diese Näherung hat vor allem

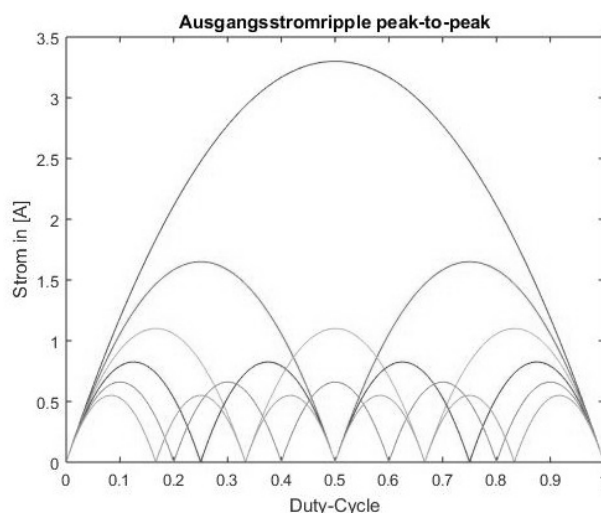


Abbildung 3: Ausgangsstromripple peak-to-peak bei m-phasigem Betrieb (m=1...6)

Einfluss auf den Eingangs- und Ausgangsstromripple der Leistungsmodule (vgl. Abbildung 3 sowie [6]). Vergleichssimulationen zeigen jedoch, dass dieser Einfluss für die hier verfolgten Zwecke vernachlässigt werden kann.

2.2 Drehstromnetz

Das Drehstromnetz hat durch seine Impedanz $\underline{X}_N(s)$ einen Einfluss auf die Spannungen am Netzanschlusspunkt. Da diese für die Gewinnung des Netzphasenwinkels $\varphi_N(t)$ benötigt werden, hat $\underline{X}_N(s)$ demnach auch einen Einfluss auf die Regelung des Systems. Da die Netzimpedanz jedoch je nach Standortbedingungen stark variieren kann ([7]), wird in den Simulationen zunächst ein Näherungswert für eine ohmsch-induktive Ersatzimpedanz $\underline{X}_{N,E}(s)$ verwendet. Der Einfluss der Netzimpedanz auf die Ableitströme wird im Kapitel 6 noch einmal erläutert.

Aus den oben beschriebenen Bestandteilen kann nun ein Gesamtsimulationsmodell erstellt werden. Die Schnittstellen wie beispielsweise die PV-Anlage werden hier durch Spannungsquellen mit ohmsch-induktivem Innenwiderstand angenähert. Weiterhin sind eventuelle Laufzeiten durch Kommunikationspfade oder Rechenzeiten noch nicht berücksichtigt. Mit diesem noch weitestgehend idealisierten Basis-Simulationsmodell können nun im Folgenden die Regelungsmethoden für das System untersucht werden.

3 Regelungsmethoden

Zur Stromregelung auf den einzelnen Leistungsmodulen werden PI-Regler verwendet. Dabei wird der Netzstromrichter mittels Strangstromregelung kontrolliert. Diese hat aufgrund des modularen Aufbaus des Netzstromrichters, welcher in Abbildung 4 dargestellt ist, Vorteile gegenüber einer Vektorregelung. So kann hier beispielsweise der gleiche Stromregelungsalgorithmus wie bei den DC-Modulen verwendet werden. Das

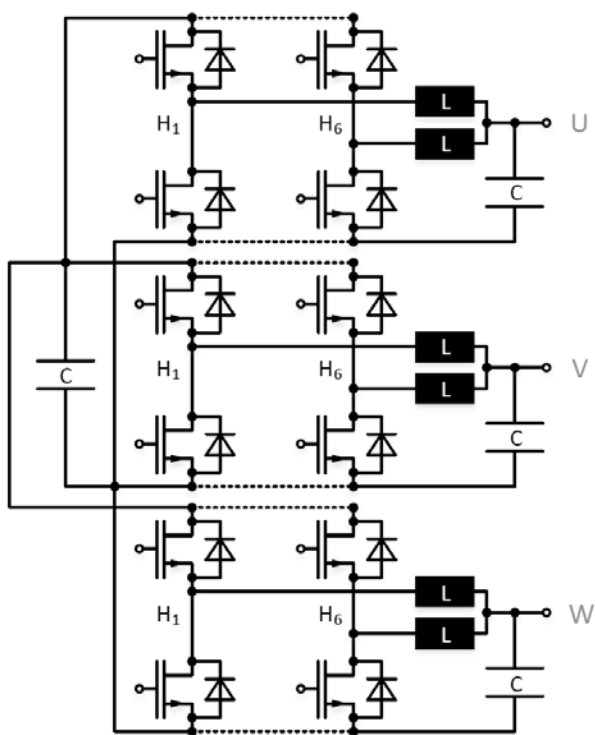


Abbildung 4: Schematischer Aufbau des Netzstromrichters

Hauptaugenmerk der Simulationen liegt hier jedoch auf der Regelung der Zwischenkreisspannung, da hier die

Regelungstopologie einen großen Einfluss hat. Einige dazu betrachtete Methoden sind:

- PI-Regler nach symmetrischem Optimum
- PI-Regler nach Führungsübertragungsfunktion
- PID-Regler nach Führungsübertragungsfunktion
- Kompensationsregler
- Störgrößenaufschaltung bei allen Reglern

Um die Regelungsmethoden untersuchen zu können, wird die Simulation um eine DC-Schnittstelle reduziert (PV-System fällt weg). Mit diesem System kann nun immer noch das Verhalten der Zwischenkreisregelung untersucht werden, die Rechenzeit einer Simulation ist jedoch geringer. Einer der DC-Wandler wird gesteuert betrieben (hier als Beispiel das Elektrofahrzeug EV), während sich der Netzstromrichter und der zweite DC-Wandler in Zwischenkreisregelung befinden. Zum Vergleich der Regelqualität der nachfolgend beschriebenen Regelungsmethoden wird ein Szenario angenommen, bei welchem positive und negative Lastsprünge nacheinander hart auf das System gegeben werden. Dabei wird die Sprungantwort des Systems, hier die Zwischenkreisspannung, bewertet.

Die Simulationen zeigen, dass alle oben beschriebenen Regler erwartungsgemäß funktionieren. Bei den Standard-Reglern erweist sich der PI-Regler ausgelegt nach der Führungsübertragungsfunktion als beste Option. Noch geringere Überschinger bei Lastwechseln sind mit dem Kompensationsregler mit PI-Zusatzbeschaltung und dem PI-Regler mit Störgrößenaufschaltung möglich. Das insgesamt beste Ergebnis erzielt jedoch der Kompensationsregler mit PI-Zusatzbeschaltung, die maximal auftretende Regelabweichung beträgt hier 0,16 %.

4 Regelungstopologien

Im nächsten Schritt werden mehrere Regelungstopologien untersucht. Da die verwendete Hardware großen Einfluss auf die Topologien hat, wird zunächst eine Vorauswahl an möglichen Mikrocontrollern wie auch Bussystemen getroffen. Auf Basis dieser Technologien sind die Regelungstopologien entstanden. Für die Bewertung dieser Topologien werden mehrere Kriterien berücksichtigt:

- Regelqualität im Normalbetrieb (aus Simulation)
- Regelqualität im Fehlerfall Netzkurzschluss (aus Simulation)

- Systemkomplexität und Implementierungsaufwand
- Systemkosten

Dazu wird zu jeder Topologie ein Simulationsmodell erstellt. Als Basis dient das in Kapitel 2 beschriebene Grundmodell. Die Rechen- und Laufzeiten werden abgeschätzt und durch Totzeitblöcke an den entsprechenden Stellen im Simulationsmodell nachgebildet. Die durchgeführten Simulationen werden im Folgenden anhand der in Abbildung 5 dargestellten Topologie erläutert.

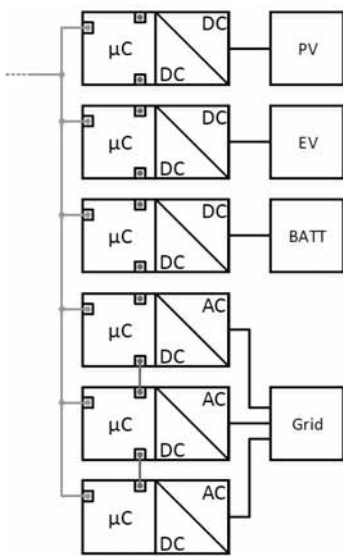


Abbildung 5: Auf CAN (grün) und UART (blau) basierende Regelungstopologie

4.1 Simulation des Normalbetriebs

Mit dieser Simulation soll zunächst die grundlegende Funktion der Topologie getestet werden. Dazu wird ein ähnliches Szenario wie in Kapitel 3 ausgewählt. Die Reaktion der gesamten Regelschleife unter Berücksichtigung der topologischen Besonderheiten kann auf diese Art getestet werden. Abbildung 6 zeigt das Ergebnis dieser Simulation. Die Spannung bricht bei einem Lastsprung auf Maximallast (20 kW) um maximal 9,5 V ein. Bei der gegebenen Zwischenkreisspannung von 800 V entspricht dies einer Regelabweichung von $\approx 1,19\%$ und liegt damit im akzeptablen Rahmen.

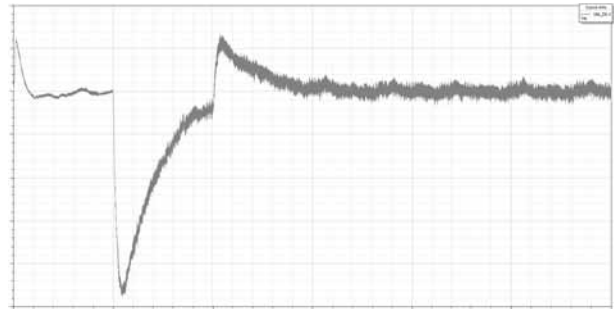


Abbildung 6: Zwischenkreisspannung bei Lastsprüngen im Normalbetrieb, max. Abweichung vom Sollwert $\approx 9,5\text{V}$

4.2 Simulation eines Netzkurzschlusses

Das System muss jedoch nicht nur in der Lage sein, den Normalbetrieb innerhalb der erwarteten Toleranzen abzudecken. In der DIN VDE-AR-N 4105 vom August 2011 wird eine dynamische Netzstützung noch nicht gefordert. Es ist jedoch denkbar, dass dies in Zukunft der Fall sein wird. Daher wird im nächsten Schritt ein dreiphasiger Netzkurzschluss simuliert. Dies bietet eine gute Möglichkeit, die Dynamik des Systems zu testen. Das System muss dabei innerhalb von möglichst kurzer Zeit reine positive oder negative Blindleistung mit $|Q_{\text{Nenn}}| = |P_{\text{Nenn}}|$ zu liefern.

Für diesen Test wird das Simulationsmodell entsprechend erweitert. Die Netzspannung aller drei Phasen sinkt nach 50 ms schlagartig auf 10% des Nennwertes ab. In diesem Moment muss der Netzstromrichter in einen gesteuerten Blindleistungsbetrieb übergehen, die Zwischenkreisregelung wird in diesem Fall der Batteriestromrichter übernehmen. Das Ergebnis dieser Simulation ist in Abbildung 7 dargestellt. Hier beträgt die maximale Abweichung vom Sollwert ebenfalls $\approx 1,19\%$. Der durch den Kurzschluss hervorgerufene Lastabwurf des Systems führt zu einer Regelabweichung von $\approx 3,25\text{V}$ (entspricht $\approx 0,41\%$) und befindet sich damit ebenfalls im akzeptablen Bereich.

4.3 Auswahl einer Topologie

Bei den Simulationen zeigt sich, dass es mehrere mögliche Varianten gibt, welche die Anforderungen erfüllen. So erweist sich beispielsweise eine auf EtherCAT basierende Variante ebenfalls als umsetzbar. Auf eine Umsetzung dieser Variante wird jedoch verzichtet,

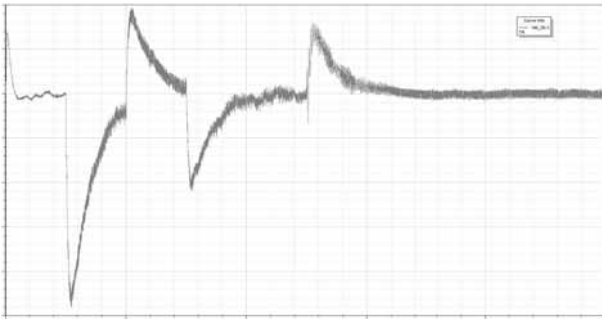


Abbildung 7: Zwischenkreisspannung bei Lastsprüngen im Normalbetrieb, dreiphasiger Netzkurzschluss bei 50ms, max. Abweichung vom Sollwert im Fehlerfall $\approx 3,25V$

da EtherCAT vorrangig für industrielle Anwendungen entwickelt wird. Daher gibt es bis jetzt nur einige wenige Mikrocontroller, die ein entsprechendes PHY implementiert haben ([8], [9]). Unter praktischen Gesichtspunkten erweist sich die in Abbildung 5 dargestellte Variante im Rahmen des Projektes NetProsum2030 als praktikabelste Lösung, daher wird diese ausgewählt. Diese nutzt eine Kombination aus CAN und UART und setzt damit auf zwei bewährte Technologien.

5 Dimensionierung des Zwischenkreises

Anstatt die Simulationen unter dem Aspekt der Regelqualität bei identischer Zwischenkreisgröße zu vergleichen ist es auch möglich, diese zu variieren. Damit lässt sich eine Aussage darüber treffen, mit welcher Topologie die minimale Zwischenkreisgröße erreicht werden kann. Voruntersuchungen an einem anderen System zeigen, dass dies grundsätzlich möglich ist. In Ergänzung zu den oben vorgestellten Simulationen wird dies aktuell noch untersucht.

6 Ableitströme

Ein weiterer wichtiger Aspekt bei der Systemauslegung ist die Erzeugung von kapazitiven Ableitströmen. Diese entstehen durch parasitäre Koppelkapazitäten gegenüber dem Erdpotential ([5]). Vor allem der Aufbau des Netzstromrichters (vgl. Abbildung 4) unterscheidet sich stark von herkömmlichen B6-Brücken. Daher soll das Ausbildungspotential der Ableitströme des hier beschriebenen Systems mittels

Simulation überprüft werden. Dazu wird das in Kapitel 2 beschriebene Grundmodell um parasitäre Erdkapazitäten erweitert. Anschließend wird das System unter verschiedenen nachfolgend beschriebenen Aspekten simuliert.

6.1 Filterwirkung der Schaltung

Im Gegensatz zu einer B6-Brücke verfügt das System durch den Aufbau aus wie in Abbildung 2 dargestellten Synchronwandlern schon über netzseitige Filter (vgl. auch Abbildung 4). Die Glättungs-drosseln und Kondensatoren wirken auf die Ableitströme als Gleichtaktfilter und führen daher zu einer Reduzierung dergleichen. Abbildung 8 zeigt den Einfluss der Kondensatoren auf das System.

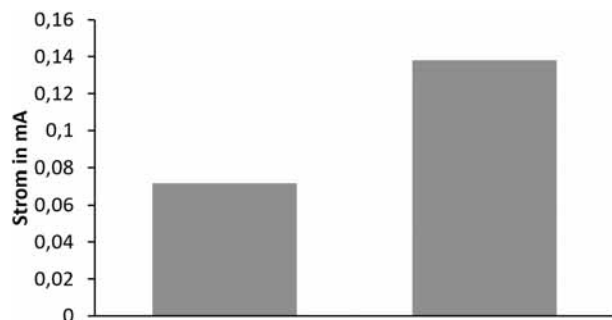


Abbildung 8: Ableitstrom des Grundsystems mit (links) und ohne (rechts) Filterkapazität $C_{Filter} = 2000nF$

6.2 Teilung des Zwischenkreises

In Bezug auf den Zwischenkreis werden zwei Varianten miteinander verglichen. Zum einen die auch in Abbildung 1 dargestellte, einfache Variante und zum Anderen eine Variante mit geteiltem Zwischenkreis. Dabei besteht der Zwischenkreis aus zwei in Reihe geschalteten Kondensatoren, der Mittelpunkt wird mit dem Neutralleiter des Netzes verbunden. Abbildung 9 zeigt die Ergebnisse der Simulation unter Verwendung des in Kapitel 2.2 beschriebenen, einfachen Netzmodells. Hier wird deutlich, dass eine Teilung des Zwischenkreises zu einer deutlichen Reduktion des Ableitstromes führt. Dies passt zu den dazu angestellten theoretischen Überlegungen.



Abbildung 9: Ableitstrom des Grundsystems ohne (links) und mit (rechts) geteiltem Zwischenkreis

6.3 Einfluss des Netzmodells

Da das Verhalten des Netzes stark vom jeweiligen Ort abhängt, soll das System auch mit verschiedenen Netzmodellen simuliert werden. Zunächst wird dazu das in Kapitel 2.2 beschriebene, einfache Netzmodell verwendet. Danach werden die Simulationen mit einem komplexeren Netzmodell wiederholt. Dieses basiert auf den Ersatzparametern einer Netznachbildung vom Typ NSLK 8128 der Firma Schwarzbeck ([10]). Bei den Untersuchungen zeigt sich, dass eine Änderung des Netzmodells starke Änderungen in den Simulationsergebnissen hervorruft. So hat eine Teilung des Zwischenkreises bei dem komplexeren Netzmodell die gegenteilige Wirkung wie bei dem einfachen Modell (vgl. Abbildung 10). Dies liegt vermutlich an der Ausbildung eines Schwingkreises zwischen Netz und dem System. Hier stellt sich die Frage, welche Netzpara-

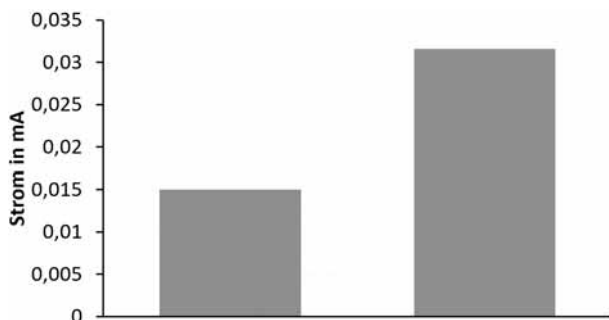


Abbildung 10: Ableitstrom des Grundsystems ohne (links) und mit (rechts) geteiltem Zwischenkreis, komplexeres Netzmodell

meter für möglichst realitätsnahe Simulationen genutzt werden sollten. Eine Messung an einem realen System zum Vergleich der Daten befindet sich aktuell in Vor-

bereitung.

Danksagung

Die hier vorgestellten Untersuchungen werden im Rahmen des vom Bundesministerium für Wirtschaft und Energie geförderten Verbundvorhabens *Kompakte modulare Wandler und optimierte Systemlösungen zur Energieflusssteuerung für netzdienlichen Prosumer 2030 mit HV-Fahrzeuggatterien »NetProsum2030«* durchgeführt.

References

- [1] Weniger J, Bergner J, Tjaden T, Quaschnig V. *Dezentrale Solarspeicher für die Energiewende*. Berlin: Berliner Wissenschafts-Verlag GmbH. 2015. 80 p.
- [2] Figgenger J, Haberschusz D, Kairies KP, Wessels O, Tepe B, Sauer DU. *Wissenschaftliches Mess- und Evaluierungsprogramm Solarspeicher 2.0 Jahresbericht 2018*. Institut für Stromrichtertechnik und Elektrische Antriebe, RWTH Aachen. 2018. 120 p.
- [3] Fischhaber S, Regett A, Schuster SF, Hesse H. *Studie: Second-Life-Konzepte für Lithium-Ionen-Batterien aus Elektrofahrzeugen*. Begleit- und Wirkungsforschung Schaufenster Elektromobilität (BuW). 2016. 123 p.
- [4] Han D, Noppakunkajorn J, Sarlioglu B. *Comprehensive Efficiency, Weight, and Volume Comparison of SiC- and Si-Based Bidirectional DC-DC Converters for Hybrid Electric Vehicles*. *IEEE Transactions on Vehicular Technology*. 2014;63(7):3001 – 3010. doi: 10.1109/TVT.2014.2323193.
- [5] SMA Solar Technology AG. *Technische Information Kapazitive Ableitströme*. <http://files.sma.de/dl/7418/Ableitstrom-TI-de-25.pdf>. Version 2.5.
- [6] Wei Chen, Linear Technology Corporation. *Application Note 77 High Efficiency, High Density, PolyPhase Converters for High Current Applications*. <http://www.linear-tech.com>. 1999.
- [7] Langkowski H, Thanh TD, Dettmann KD, Schulz D. *Grid Impedance Determination - Relevancy for Grid Integration of Renewable Energy Systems*. 2009 35th Annual Conference of IEEE Industrial Electronics. 2009. doi: 10.1109/IECON.2009.5414975.
- [8] Margit Kuther. *Kommunikation in harter Echtzeit*. <https://www.elektronikpraxis.vogel.de/kommunikation-in-harter-echtzeit-a-586680/>. 02.03.2017.

- [9] Infineon Technologies Austria AG. *Product Brief XMC4300 and XMC4800*. www.infineon.com/ethercat. 2016.
- [10] Schwarzbeck Mess-Elektronik OHG. *Netznachbildung NSLK 8128*. <http://www.schwarzbeck.de/Datenblatt/k8128.pdf>. Rev. A.

Charakterisierung von gestörten Geschwindigkeitsprofilen in runden, rechteckigen und quadratischen Strömungsgeometrien

Konstantin Zacharias^{1*}, Wolfgang Schlüter¹

¹Fakultät Technik, Hochschule Ansbach, Residenzstraße 8, 91522 Ansbach

*konstantin.zacharias@hs-ansbach.de

Abstract. Die Charakterisierung von gestörten Geschwindigkeitsprofilen mit Drall erfolgt mit dimensionslosen Kennzahlen. Das gestörte Strömungsfeld wird dabei numerisch mit Hilfe einer CFD-Simulation berechnet. Der Fokus dieser Arbeit liegt auf dem Vergleich der Relaxation der Kennzahlen in verschiedenen Strömungsgeometrien. Zunächst werden die numerischen Ergebnisse der Drall-Relaxation mit theoretischen Ansätzen validiert. Die exponentielle Abnahme des Drallwinkels wird dabei mit der Theorie von Steenbergen und Voskamp am besten beschrieben. Der Vergleich der Geometrien zeigt, ein Unterschied im Abklingverhalten des Drallwinkels für eckige und runde Querschnitte. Die Form des axialen Geschwindigkeitsprofils, beschrieben durch den Profilmfaktor, nähert sich für alle untersuchten Geometrien dem Gleichgewichtszustand in ähnlicher Form an.

Einleitung

In der technischen Anwendung ist die Strömung zumeist nicht voll ausgebildet und durch Einbauten wie z.B. Rohrkrümmer, Ventile oder Drallerzeuger gestört. Dies kann bei Anwendungen wie der Wärmeübertragung oder der Durchmischung von Mehrphasenströmungen von Vorteil oder wie in der Durchflussmess-technik unerwünscht sein. In experimentellen Untersuchungen wurden charakteristische Kennzahlen in kreisrunden Querschnitten von Yen und Mattingly [1] vorgestellt und von Müller und Dues [2] mit dem Ziel weiterentwickelt, gestörte Strömungen miteinander zu vergleichen und die Auswirkung dieser auf Messgeräte zu quantifizieren. Eichler und Lederer [3] untersuchten die Relaxation von vier dieser charakteristischen Kennzahlen: dem Profilmfaktor K_p , dem Asymmetriefaktor K_a , dem Turbulenzfaktor K_{Tu} und dem Drallwinkel ϕ . Sie stellten eine Beeinträchtigung der Strömung bis weit nach der Störung fest. Graner und Hinz [4] konnten in ihren experimentellen Versuchen die Relaxation des

Drallwinkels ϕ mit der Theorie von Steenbergen und Voskamp [5] gut beschreiben und einen linearen Anstieg des Profilmfaktors K_p mit der Lauflänge feststellen. In dieser Arbeit werden gestörte Geschwindigkeitsprofile in eckigen und runden Querschnitten untersucht. Hierfür wird die charakteristische Kennzahl des Profilmfaktors K_p leicht modifiziert und mit dem Drallwinkel ϕ bis zur Relaxation berechnet. Das komplette Strömungsfeld wird dabei numerisch simuliert. Das Ziel dieser numerischen Untersuchung ist der Vergleich von gestörten Strömungen in unterschiedlichen Kanalgeometrien.

1 Charakterisierung von Strömungsprofilen

Die Charakterisierung der gestörten Strömungsprofile erfolgt anhand des Drallwinkels ϕ und des Profilmfaktors K_p . Für den Asymmetriefaktor K_a und den Turbulenzfaktor K_{Tu} ist keine klare Abhängigkeit der Relaxation vorhanden [4]. Daher werden diese Kennzahlen in dieser Arbeit nicht betrachtet.

1.1 Drallwinkel

Zur Charakterisierung von Strömungen mit Drall wurde eine Vielzahl von Kennzahlen erarbeitet und vorgeschlagen. Eine Zusammenfassung aller Drall-Kennzahlen ist in der Arbeit von Graner und Hinz [4] aufgelistet. Im Hinblick auf standardisierte Kennzahlen [2] wird der Drallwinkel wie folgt gebildet:

$$\phi = \text{atan} \frac{|v_{yz}|_{\max}}{u_{\text{vol}}} \quad (1)$$

Dabei ist $|v_{yz}|_{max}$ der maximale Betrag der Sekundärströmung und u_{vol} die volumetrische mittlere Strömungsgeschwindigkeit. Geometrisch betrachtet ist der Drallwinkel ϕ der Winkel zwischen dem idealen Geschwindigkeitsvektor und dem tatsächlichen Geschwindigkeitsvektor mit Drall [4].

1.2 Profilkfaktor

Der Profilkfaktor K_p nach Yeh und Mattingly [1] wird aus der axialen Geschwindigkeitskomponente gebildet. Er beschreibt die Zuspitzung ($K_p > 1$) bzw. Abflachung ($K_p < 1$) des aktuellen Geschwindigkeitsprofils im Bezug auf eine voll ausgebildete Strömung. Aus der Strömungssimulation liegt die Geschwindigkeitsverteilung als Feldgröße vor. Daher wird der Profilkfaktor leicht modifiziert über den Querschnitt gebildet:

$$K_p = \frac{K_{p,num}}{K_{p,num,s}} \quad (2)$$

mit

$$K_{p,num} = \frac{1}{u_{vol}A} \int_0^R \int_{-R}^R (u_m - u) drd\phi \quad (3)$$

und

$$K_{p,num,s} = \frac{1}{u_{vol}A} \int_0^R \int_{-R}^R (u_{m,s} - u_s) drd\phi \quad (4)$$

Dabei ist u_m die axiale Geschwindigkeit in der Rohrmitte und u_s die axiale Geschwindigkeit aus dem voll ausgebildeten Geschwindigkeitsprofil. Für die nicht-runden Querschnitte erfolgte eine Koordinatentransformation. Für runde Querschnitte kann das voll ausgebildete Standardprofil nach Gersten und Herwig [6] analytisch bestimmt werden. Da eine analytische Beschreibung der voll ausgebildeten Geschwindigkeitsprofile für nicht-runde Querschnitte nicht existiert, wird auch der Profilkfaktor $K_{p,num,s}$ numerische berechnet.

2 CFD-Simulationen

Die numerischen Untersuchungen werden an geometrisch ähnlichen Profilen mit der kommerziellen CFD-Software StarCCM+ durchgeführt. Die Umrechnung findet über den hydraulischen Durchmesser d_h ,

$$d_h = \frac{4A}{U} \quad (5)$$

mit dem durchströmten Querschnitt A und des benetzten Umfangs U, statt. Für ausgebildete Strömungen können damit gute Näherungsaussagen über das Widerstandsgesetz bei beliebigen Querschnittsformen zur Verfügung gestellt werden [10]. Für die Betrachtung der geometrischen und der dynamischen Ähnlichkeit müssen die Reynoldszahlen für die unterschiedlichen Querschnitte gleich sein. Dies wird erreicht, indem die dynamische Viskosität μ für den runden und rechteckigen Querschnitt angepasst wird. In Tabelle 1 sind die geometrischen Abmessungen und die Strömungsgrößen für den Massenstrom $\dot{m} = 1.111 \text{ kg/s}$ bei geometrischer und dynamischer Ähnlichkeit aufgelistet.




		
D=12.5 mm	b = 10 mm h = 15 mm R=4x1.6 mm	b = 12 mm h = 12 mm R=4x1.6 mm
$d_h = 12.5 \text{ mm}$	$d_h = 12.5 \text{ mm}$	$d_h = 12.5 \text{ mm}$
$u_{vol} = 9.05 \frac{m}{s}$	$u_{vol} = 7.52 \frac{m}{s}$	$u_{vol} = 7.84 \frac{m}{s}$
$\rho = 1000 \frac{kg}{m^3}$	$\rho = 1000 \frac{kg}{m^3}$	$\rho = 1000 \frac{kg}{m^3}$
$\mu = 0.0012 \text{ Pas}$	$\mu = 9.6 \cdot 10^{-4} \text{ Pas}$	$\mu = 0.001 \text{ Pas}$
Re=98000	Re=98000	Re=98000

Tabelle 1: Untersuchte Auslaufprofile und die Strömungsgrößen für den Massenstrom $\dot{m} = 1.111 \text{ kg/s}$

2.1 Störkörper

Als Störkörper dient ein linksdrehender $D = 20 \text{ mm}$ Drallerzeuger (s. Abb. 1a) nach OIML 49-2:2013 [7] und EN ISO 4064-2:2014 [8], der sich in einem Rohr mit dem Innendurchmesser $D = 24 \text{ mm}$ befindet.

2.2 Geometrie und Simulationseinstellungen

Die Charakterisierung der gestörten Strömungsprofile erfolgt mit den Kennzahlen aus Kapitel 1 für die drei Rohrprofile rund, rechteckig und quadratisch. Die geometrischen Maße der Profile sind in Tabelle 1 abgebildet. Zur Überführung des Innendurchmessers $D=24 \text{ mm}$ auf die Rohrprofile wird jeweils eine Düse verwendet, die sich direkt hinter dem Drallerzeuger befindet. Der Düsenaustrittsquerschnitt wird dem jeweili-

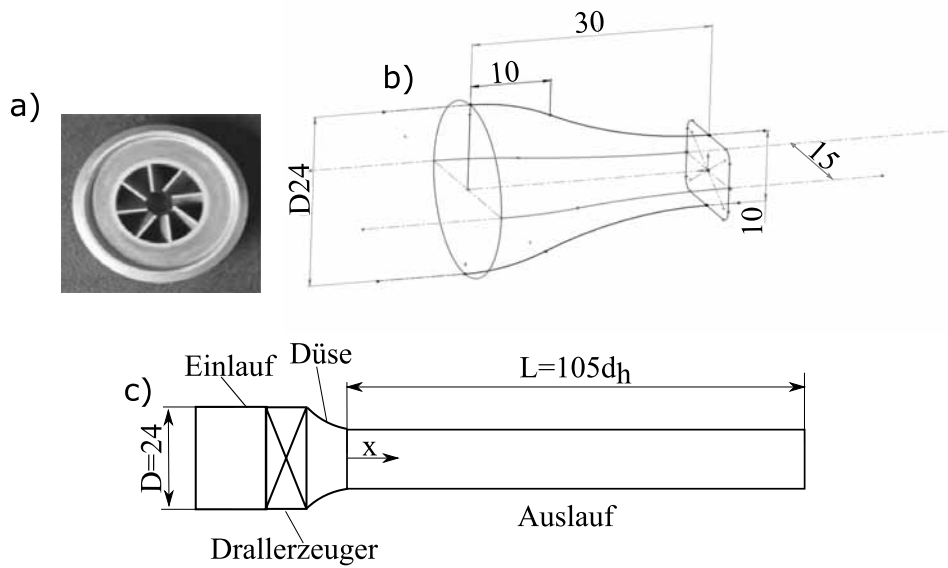


Abbildung 1: a) Drallerzeuger nach OIML 49-2:2013 [7] und EN ISO 4064-2:2014 [8], aus [4]; b) Düsengeometrie für das Rechteckprofil; c) Schematische Darstellung des gesamten Simulationsmodells

gen Rohrprofil angepasst. Die weiteren Abmessungen bleiben gleich. Abbildung 1c zeigt die schematische Darstellung des gesamten Simulationsmodells. Als Turbulenzmodell wird ein nichtlineares Reynoldsspannungsmodell (Elliptic Blending RSM) nach Mancau und Hanjalić [9] ohne Wandfunktionen verwendet. Mit diesem Modell werden alle Komponenten des Reynoldsspannungstensors berechnet und somit lassen sich auch anisotrope Turbulenzstrukturen, die in rotierenden Strömungen auftreten, erfassen. Als Randbedingung wird dem Einlauf ein voll ausgebildetes Strömungsprofil aufgeprägt. Das unstrukturierte Polyedernetz ist bis in den wandnahen Bereich aufgelöst, sodass ein y^+ -Wert von $y^+ < 1$ erreicht wird. Je nach Geometrie werden für die Vernetzung $15\text{-}20 \cdot 10^6$ Elemente verwendet.

2.3 Validierung

Die Validierung des Simulationsmodells erfolgt am $D = 20$ mm Wirbelerzeuger in einem $D = 24$ mm runden Querschnitt ohne Düse. Somit besitzt auch der Auslauf einen Innendurchmesser von $D = 24$ mm. Die Relaxation des Drallwinkels ϕ wird mit den theoretischen Ansätzen von Steenberg und Voskamp [5] und Gersten und Papenfuss [11] verglichen. Beide Theorien gehen von einer exponentiellen Abnahme des Drallwinkels (bzw. Drallintensität) in Abhängigkeit des Drall-

Abklingkoeffizienten β aus, der wiederum stark mit der Rohrreibungszahl λ und den Strömungsbedingungen korreliert. Mit der leicht modifizierten Version von Graner und Hinz [4], lässt sich der Drallwinkel wie folgt beschreiben:

$$\phi = \phi_{x_0} e^{-\beta(x-x_0)/D} \quad (6)$$

Nach Steenberg und Voskamp [5] kann der Drall-Abklingkoeffizient β in hydraulisch glatten Röhren nach der empirischen Gleichung

$$\beta_s = (1.49 \pm 0.07)\lambda \quad (7)$$

berechnet werden. Der Index s verweist dabei auf die Theorie von Steenberg und Voskamp. Die Rohrreibungszahl λ berechnet sich nach Blasius [10] für ein glattes Rohr wie folgt:

$$\lambda = \frac{0.3164}{Re^{0.25}} \quad (8)$$

Gersten und Papenfuss [11] verzichten auf den empirischen Ansatz für den speziellen Fall eines einzelnen longitudinalen Dralls. In diesem Fall kann der Drall-Abklingkoeffizient β mit der Rohrreibungszahl λ gleichgesetzt werden.

$$\beta_p = \lambda \quad (9)$$

Für den Validierungsfall mit einem Massenstrom

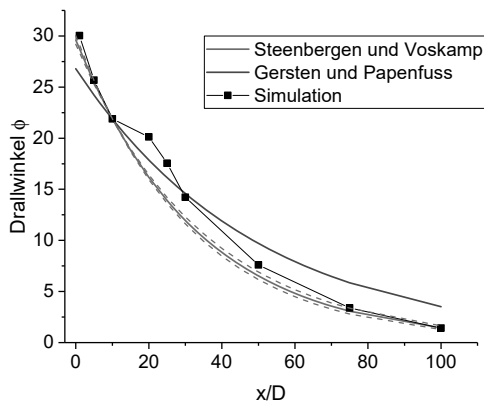


Abbildung 2: Vergleich des numerisch berechneten Drallwinkels ϕ mit den theoretischen Ansätzen

von $\dot{m}=1.111 \text{ kg/s}$ beträgt die Reynoldszahl $Re=58942$. Aufgrund der sprungartigen Erweiterung nach dem Drallerzeuger ($D=20 \text{ mm}$ auf $D=24 \text{ mm}$) wird für die Berechnung beider Ansätze der Drallwinkel an der Position $x/D=10$ mit $\phi_0=21.88$ gewählt. Abbildung 2 zeigt den Vergleich des numerisch ermittelten Drallwinkels mit den Theorien von Gersten und Papenfuss [11] und Steenbergen und Voskamp [5]. Die Relaxation des Drallwinkels ϕ wird über weite Bereiche besser mit dem Ansatz von Steenbergen und Voskamp [5] beschrieben. Zu diesem Ergebnis kommen auch Eichler und Lederer [3] und Graner und Hinz [4] in ihren experimentellen Untersuchungen. Im Bereich $x/D=10$ bis $x/D=30$ ist eine Änderung im Abklingverhalten des Drallwinkels erkennbar. Diese Änderung kann mit dem einfachen theoretischen Ansatz nicht beschrieben werden, tritt jedoch auch im Experiment von Eichler und Lederer [3] auf und ist somit nicht numerisch bedingt.

Die Validierung zeigt, dass die Relaxation des Drallwinkels numerisch mit dem nicht linearen Reynoldsspannungsmodell sehr gut beschrieben werden kann.

3 Simulationsergebnisse

In diesem Kapitel wird die Relaxation des Drallwinkels ϕ und des Profilkoeffizienten K_p für die drei Auslaufgeometrien analysiert.

Drallwinkel. Abbildung 3 zeigt die Relaxation des Drallwinkels ϕ in Abhängigkeit verschiedener stromab Positionen im Auslauf für die Geometrien

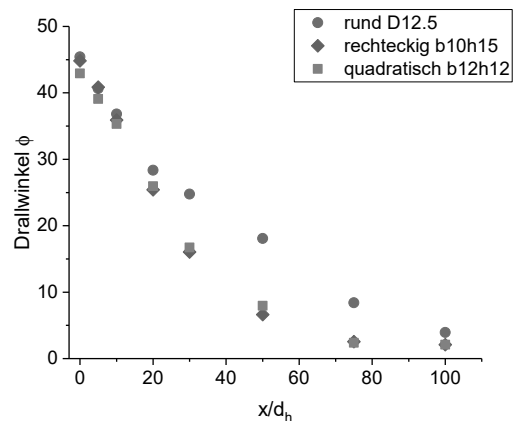


Abbildung 3: Relaxation des Drallwinkels ϕ für die untersuchten Geometrien

rund, rechteckig und quadratisch. Für die Normierung des Drallwinkels ϕ wird nach Gleichung 1 die mittlere Strömungsgeschwindigkeit am Einlauf herangezogen. Für den Bereich $x/d_h < 10$ besitzen alle drei Geometrien das gleiche Abklingverhalten. Die Unterschiede in diesem Bereich sind auf die unterschiedlichen Steigungen in der Düse zurückzuführen. Das Abklingverhalten in der rechteckigen und quadratischen Geometrie bleibt bis zur letzten Auswerteposition $x/d_h=100$ nahezu gleich. Bei dem kreisrunden Querschnitt ändert sich die Steigung der Abklingkurve ab dem Bereich $x/d_h > 20$ und nähert sich mit einem flacheren Gradienten der Relaxation. In allen drei Geometrien ist der Gleichgewichtszustand nach $x/d_h=100$ noch nicht erreicht. In der kreisrunden Geometrie ist der Drallwinkel an dieser Position noch etwa doppelt so hoch wie für die rechteckigen Geometrien. Für die runde Geometrie ist daher eine längere Auslaufstrecke nötig bis sich der Gleichgewichtszustand einstellt.

In Abbildung 4 ist der Betrag der normierten Sekundärströmung als Konturplot für vier Positionen stromab der Düse dargestellt. An der Position $x/d_h=12$ ist für den runden Querschnitt noch die Zerfallstruktur der acht Schaufeln des Drallerzeugers in der Sekundärströmung zu erkennen. Weiter stromab verlagert sich das Maximum der Sekundärströmung vom Zentrum nach außen. Diese Verlagerung ist für die Änderung des Abklingverhaltens des Drallwinkels verantwortlich, die in Abbildung 2 und 3 zu sehen sind. Bei den eckigen Profilen ist zu erkennen, dass der Drall nicht den gesamten Quer-

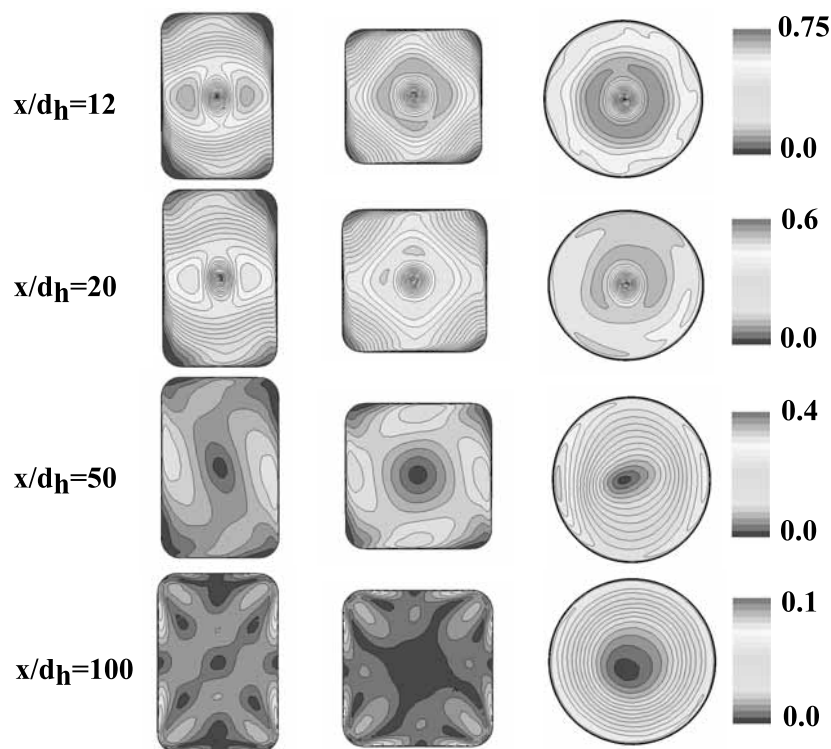


Abbildung 4: Vergleich der normierten Sekundärströmung $\frac{v_{yz}}{u_{vol}}$ für unterschiedliche stromab Positionen

schnitt einnimmt, wodurch der steilere Gradient im Abklingverhalten des Drallwinkels in Abbildung 3 zu erklären ist. In den eckigen Profilen ist der Betrag der Sekundärströmung in den Ecken minimal, bis die Wandschubspannung dort kleiner wird als an den Seitenwänden. Dadurch fließt Flüssigkeit aus dem Rohrrinnen zu den Ecken, was zu einem höheren Betrag der Sekundärströmung in den Ecken führt (vgl. Position $x/d_h=100$). Es entsteht das typische Muster der Sekundärströmung in rechteckigen Geometrien, die eine Folge der Anisotropie der Turbulenz ist.

Profilfaktor. Der Profilfaktor beschreibt eine Zuspitzung bzw. Abflachung des axialen Geschwindigkeitsprofils gegenüber der voll ausgebildeten Strömung. Abbildung 5 zeigt die Entwicklung des Profilfaktors K_p stromab der Düse für die drei unterschiedlichen Geometrien. Direkt nach der Düse besitzt das Geschwindigkeitsprofil eine Kolbenform und ist somit flacher als bei der voll ausgebildeten Strömung ($K_p < 1$). Danach ist eine lineare Zunahme des Profilfaktors K_p für alle drei Geometrien zu beobachten bis der Maximalwert bei $x/d_h=30$ erreicht wird. Ab dieser Position kommt es

wieder zu einer Abflachung des axialen Geschwindigkeitsprofils für alle drei Geometrien. Für das rechteckige Profil ist nach $x/d_h=100$ der Gleichgewichtszustand ($K_p=1$) fast erreicht. Das runde und quadratische Profil weisen an dieser Position noch einen Profilfaktor von $K_p=1.2$ auf. Für diese Profile stellt sich das voll ausgebildete Strömungsprofil weiter stromab ein. Auch in den Experimenten von Graner und Hinz [4] und Eichler und Lederer [3] ist die lineare Relaxation des Profilfaktors K_p zu erkennen. Aufgrund der vorgeschalteten Düse ist die Steigung in dieser numerischen Untersuchung deutlich größer als in den Experimenten.

4 Zusammenfassung

Es wurden numerische Strömungssimulationen mit gestörten Geschwindigkeitsprofilen in eckigen und runden Geometrien durchgeführt. Als Störkörper diente ein $D = 20$ mm Wirbelerzeuger. Die Charakterisierung der gestörten Profile erfolgte mit dem Drallwinkel ϕ und dem modifizierten Profilfaktor K_p . Die Relaxation des Drallwinkels ϕ wurde mit den theoretischen Ansätzen von Steenbergen und Voskamp [5] und Gersten und

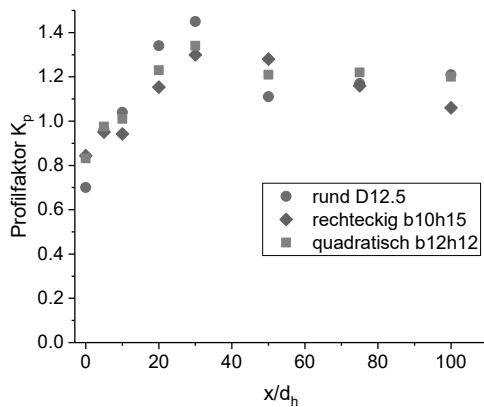


Abbildung 5: Relaxation des Profilkoeffizienten K_p für die untersuchten Geometrien

Papenfuss [11] verglichen. Es konnte eine gute Übereinstimmung mit den Simulationsergebnissen und der Theorie von Steenbergen und Voskamp nachgewiesen werden. Der Vergleich für eckige und runde Geometrien zeigte, dass bei dynamischer und geometrischer Ähnlichkeit der Drallwinkel für die rechteckige und quadratische Geometrie das gleiche Abklingverhalten besitzt. Die kreisrunde Geometrie weist eine flachere Steigung auf und benötigt daher zum Erreichen des Gleichgewichtszustandes eine längere Auslaufstrecke. Der Profilkoeffizient K_p verhält sich für die untersuchten Geometrien gleich. Es kommt zunächst zu einer linearen Steigung des Profilkoeffizienten und einer Überspitzung gegenüber der voll ausgebildeten Strömung $K_p > 1$. Anschließend relaxiert der Profilkoeffizient mit unterschiedlichen Gradienten. Die Störung im Geschwindigkeitsprofil konnte bis zur Position $x/d_h=100$ nachgewiesen werden und benötigt wohl noch eine längere Auslaufstrecke um komplett abzuklingen.

Literatur

- [1] T.T Yeh, G.E Mattingly. Pipeflow downstream of a reducer and its effects on flowmeters. *Flow Meas. Instrum.*. 1994; volume(5): 181-187.
- [2] U. Müller, M. Dues. Vollflächige Erfassung von ungestörten und gestörten Geschwindigkeitsverteilungen in Rohrleitungen mittels der Laser-Doppler-Velocimetrie. *Technisches Messen.*. 2007; volume(74): 342-352.
- [3] T. Eichler, T. Lederer. Flow development behind a swirl generator in a hot-water standard measuring facility for

large volume rates. *Flow Measurement and Instrumentation.*. 2015; volume(42): 89-97.

- [4] S. Graner, D. Hinz. Downstream relaxation of velocity profiles in pipe-flow with swirl disturbances. *Technisches Messen.*. 2016; volume(83): 696-711.
- [5] W. Steenbergen, J. Voskamp. The rate of decay of swirl in turbulent pipe flow. *Flow Measurement and Instrumentation.*. 1998; volume(9): 67-78.
- [6] K. Gersten, H. Herwig. Ausgebildete Durchströmung. *Strömungsmechanik.*. 1.Auflage. Vieweg+Teubner Verlag Wiesbaden; 1992
- [7] ISO 4064-2:2014. *Water meters for cold potable water and hot water- Part 2: Test methods* ..2014
- [8] OIML R 49-2:2013. *International recommendation: Water meters for cold potable water and hot water- Part 2: Test methods* ..2013
- [9] R. Manceau, K. Hanjalić. Elliptic blending model: A new near-wall Reynolds-stress turbulence closure *Physics of Fluids.*. 2002; volume(14): 744.
- [10] H. Herwig. Durchströmung schlanker Kanäle. *Strömungsmechanik.*. 1.Auflage. Vieweg+Teubner Verlag Wiesbaden; 2008
- [11] K. Gersten, H. Papenfuss. Decay of Disturbance in Turbulent Pipe Flow. In: W.Merzkirch. *Fluidmechanics of Flow Metering.*. Springer-Verlag Berlin Heidelberg; 2005

Realisierung einer Datenfusionsstruktur für die Umfeldperzeption autonomer Fahrzeuge

Marian Göllner^{1*}, Xiaobo Liu-Henke¹

¹Institut für Mechatronik, Ostfalia Hochschule für angewandte Wissenschaften, Salzdahlumer Str. 46/48, 38302 Wolfenbüttel; *mar.goellner@ostfalia.de

Abstract. Der Einsatz autonomer Fahrzeuge kann bis zu 90% aller derzeitigen Unfälle verhindern [1] und gleichzeitig das Aufkommen von Staus und auch den Schadstoffausstoß deutlich reduzieren. Deshalb ist es Bestrebung der Fachgruppe Regelungstechnik und Fahrzeugmechatronik der Ostfalia Hochschule an existierenden Funktionsträger und Forschungsfahrzeuge Fragestellungen des autonomen Fahrens zu erforschen. Ein wichtiger Teil des autonomen Fahrbetriebs ist dabei das Erkennen und Klassifizieren anderer Verkehrsteilnehmer sowie die Detektion von Hindernissen. Zusammen mit einer Fahrbahnerfassung und globalen Positionsdaten soll so eine Trajektorie erzeugt werden, welcher ein autonomes Fahrzeug folgen kann. Damit die an der Umfeldperzeption beteiligten Sensoren zuverlässige Daten liefern und die Objektklassifizierung möglichst fehlerfrei vollziehen, ist es notwendig mehrere Sensorergebnisse zu fusionieren. Schwerpunkt dieses Papers ist die Strukturierung der Sensordatenauswertung, die Fusionsreihenfolge und die Definition von Gütekriterien.

Einleitung

Laut einer in der FAZ veröffentlichten Studie kann der autonome Fahrbetrieb eines Personenbeförderungsfahrzeugs 90 % der sich im öffentlichen Straßenverkehr ereignenden Unfälle verhindern [1]. Dies ist zum einen auf die schnellere Reaktion und höhere Datenverarbeitungsrate, als auch zum anderen auf die umfassendere parallele sensorische Erfassung zurückzuführen. Folglich ist die Aufbereitung und Vorverarbeitung des sensorischen Inputs zur Gewährleistung fehlerfreier und exakter Detektion ein Kernthema aktueller Forschung, zumal ein Sensor nach [18] auch ein grundsätzlich störungsbehaftetes Element darstellt. Dabei ist es nicht nur wichtig, statische und dynamische Informationen über

die Kontur des umgebenden Terrains und der sich darin bewegenden Objekte zu sammeln, sondern vor allem auch diese Objekte zu klassifizieren. Durch die Klassifizierung können modellbasiert weitere Aussagen über zukünftige Bewegungen des Objektes gemacht werden. Es ist z.B. möglich, dass ein stehendes Auto als unbewegliches Objekt detektiert wird, durch die Klassifizierung jedoch präventive Szenarien berechnet werden können, welche die Weiterfahrt des Autos vorhersagen (z.B. an einer Roten Ampel). Abbildung 1 zeigt einen der Funktionsträger der Fachgruppe für Regelungstechnik und Fahrzeugmechatronik mit Sensoren zur Umfeldfassung.

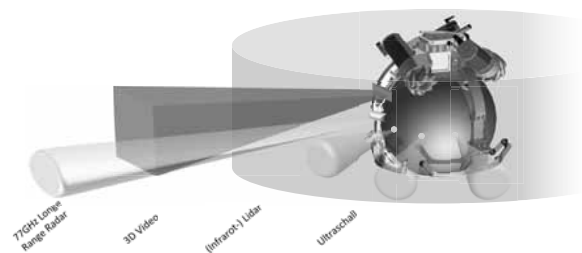


Abbildung 1: Messbereichsanordnung am Forschungsträger S-Mobile [2]

Dabei ist es von enormer Wichtigkeit sowohl die sensorische Detektion als auch die informatische Klassifizierung aus den gesammelten Sensordaten mit hoher Präzision durchzuführen. Die gewonnenen Daten sollen schließlich zur autonomen Fahrt einer Maschine verwendet werden und sind somit sicherheitskritisch. Wie das folgende Kapitel 1 beschreiben wird, sind die nutzbaren Messprinzipien der zur Umfeldfassung geeigneten Sensoren durch unterschiedliche äußere Einflüsse täuschbar. Ein Weg, die einzelnen Nachteile der Umfeldfassungssensoren zu kompensieren, ist eine Fusion, Zuordnung und Validierung der gemessenen Daten [3], [4].

1 Stand des Wissens

Um die Notwendigkeit einer Sensordatenfusion zu verdeutlichen, sollen zunächst die grundlegenden Messprinzipien üblicher Sensortypen für die Umfelderkennung erläutert und sich daraus ergebende Probleme hergeleitet werden.

1.1 LIDAR-Sensoren

LIDAR ist ein optisches Messverfahren zur Ortung und Messung der Entfernung von Objekten im Raum. Das Prinzip ähnelt dem des Radarverfahren, wobei allerdings anstelle von Mikrowellen bei Lidar verwendet werden [5]. Das Messprinzip eines Lidar beruht auf der „Time of Flight“ Messung. Hierbei werden ein oder mehrere Ultraviolett-, Infrarot- oder Strahlen aus dem Bereich des sichtbaren Lichts gepulst ausgesendet und an einem eventuell vorhandenen Objekt reflektiert. Da sich das Licht mit einer konstanten Geschwindigkeit von ca 1 c in Luft ausbreitet kann über die Zeit, die bis zum Empfangen des reflektierten Signals vergeht, primär auf die Entfernung des Objektes geschlossen werden. Zudem ist es theoretisch möglich, über die Dopplereffekte im Lichtspektrum, die Relativgeschwindigkeit zu einem Objekt (Fahrzeug) zu ermitteln. Man bedient sich praktisch aber der Differenz zwei, oder mehrerer aufeinanderfolgenden Abstandsmessungen [5].

Aufgrund diese Verfahrens ergeben sich jedoch auch Probleme. Besteht eine erhöhte Dämpfung der Atmosphäre – zum Beispiel durch Nebel bedingt – so werden einzelne Pulse an den Wassertröpfchen in der Luft reflektiert. Je nach optische Auslegung des Systems kann dies zu Sättigungsverhalten im Empfänger führen, so dass eine Messung nicht mehr möglich ist. Zudem emittiert die Sonne selbst einen erheblichen Anteil an Infrarotstrahlung. Hierdurch kann eine „Blendung“ durch das Umgebungslicht auftreten. Um diesen Effekt zu kompensieren, können durch geeignete Filtermaßnahmen der Gleichlichtanteil (verursacht durch die Sonnenstrahlung) unterdrückt werden.

Ein weiteres Problem ist die Farbe des zu detektierenden Objektes. Während die meisten Fahrzeuge für Radarsensoren viele gut reflektierende Metalloberflächen aufweisen, können Autos mit schwarzen Oberflächen nicht immer detektiert werden [6]. Elektromagnetische Wellen wie das Sonnenlicht sind energiereich. Diese Energie wird von den Molekülen eines Körpers absorbiert und in Bewegungsenergie umgewandelt. Der Anstieg der Bewegungstätigkeit der Moleküle wieder-

um erzeugt Wärme, die vom Körper an die Umgebung abgegeben wird. Bei Schwarzen Oberflächen wird fast das gesamte Lichtspektrum absorbiert, und kaum welches reflektiert.

1.2 Radar- Sensoren

Die Abstrahlung von elektromagnetischen Wellen und deren Empfang ist nur notwendige Voraussetzung für die Funktion von Radar. Damit ist aber nicht mehr als ein Träger der Informationen geschaffen. Dem Träger müssen erst noch Informationen senderseitig aufmoduliert und empfängerseitig demoduliert werden. Durch diese Modulation ist es erst möglich, Abstände zu messen [7]. Das Messverfahren basiert darauf, die empfangenen Signale bezüglich Zeit und/oder Frequenz mit dem ausgesendeten Signal zu vergleichen [8]. 1842 sagte schon Christian Doppler voraus, dass eine elektromagnetische Welle eine Frequenzverschiebung erfährt, wenn sich Beobachter und Sender relativ zueinander bewegen [9]. Bewegen sich zwei Fahrzeuge mit einer Relativgeschwindigkeit, so wird die Empfangsfrequenz wegen des Dopplereffektes sowohl in der aufsteigenden als auch in der abfallenden Rampe um einen bestimmten Betrag erhöht. Es ergeben sich zwei unterschiedliche Frequenzdifferenzen, deren Addition das Maß für den Abstand und Subtraktion das Maß für die Relativgeschwindigkeit der Fahrzeuge zueinander ist [10].

Radarstrahlen werden nicht als Kugelwelle mit in allen Raumrichtungen gleicher Intensität, sondern in einer gebündelten Weise durch die Antenne ausgesendet. Radarwellen können bei schrägen planen Oberflächen abgelenkt werden. Dies würde dazu führen, dass der Empfänger keine Reflektion detektiert [7].

Neben der Dynamik des Radarquerschnitts beeinflusst der radiale Abstand die Signalstärke am Empfänger. Bei 76,5 GHz liegt die atmosphärische Dämpfung unter 1 dB/km, damit nur 0,3 dB für den Hin- und Rückweg zu einem 150m entfernten Ziel. Ein Dämpfungsmaximum liegt bei 60 GHz mit etwa 15 dB/km vor. Starke Dämpfung entsteht auch durch Regen. Dies führt zu einer erheblichen Reichweiteeinbuße, wobei die erreichbare Sichtweite oft die für ein optisches System verbleibende überschreitet. Neben der Dämpfungswirkung führt starker Regen auch zu einem erhöhten Störpegel (Clutter). Meist wirkt sich dieser Störpegel als erhöhter Rauschpegel und senkt auf diese Weise das Signal/ Rausch- Verhältnis und damit die Reichweite. Durch Schwallwasser eines zum Beispiel entgegenkommenden LKW können Scheinziele durch starke

Reflektion generiert werden. Befindet sich Wasser vor dem Strahlenaustrittsbereich, kann dies aufgrund der hohen Dielektrizitätszahl zu ungewollten „Linseneffekten“ führen, wodurch insbesondere die Bestimmung des Azimut Winkels stark verfälscht werden kann [7].

1.3 Ultraschall- Sensoren

Ultraschallsensoren basieren auf dem physikalischen Wirkprinzip der Luftultraschallausbreitung als mechanische Dichtewelle [11] und eines reflektierten Echoimpulses. Der Abstand zum nächstgelegenen Hindernis ergibt sich aus der Laufzeit des zuerst eintreffenden Echoimpulses und der (temperaturabhängigen) Schallgeschwindigkeit c_s in Luft (ca. 340 m/s) [8]. Die Reichweite eines Ultraschallsensors ist durch den vom Sender erzeugten Schalldruck p und die Empfangsempfindlichkeit bestimmt. Das Schalldruckverhältnis ist bei fester Frequenz abhängig vom Abstandsverhältnis und von den Stoffeigenschaften des Übertragungsmediums Luft. Die Arbeitsfrequenz ist damit die bestimmende Größe für den erzielbaren Abstandsbereich und die Reichweite von Störsignalen. Ultraschallsensoren arbeiten mit einer Frequenz zwischen 30 kHz – 1000 kHz [12]. Diesen Frequenzen stehen Messbereiche von 0,1 m bis 15 m gegenüber [12], [8], [10].

Hervorzuheben ist ferner die Unabdingbarkeit der Messung von Beleuchtung und Farbe sowie die geringe Beeinträchtigung der Qualität einer Objekterkennung durch Verschmutzung, Nässe oder Nebel im Erfassungsbereich des Ultraschallsensors [13].

1.4 Kamerasysteme

Ein Kamerasystem ist im Gegensatz zu den anderen aufgeführten Sensoren nicht primär zur Messung von Abständen und Geschwindigkeiten gedacht, da die Messausgabe lediglich ein zweidimensionales Abbild der Fahrzeugumgebung ist. Nach [14] sind jedoch zwei grundlegende Verfahren bekannt, um Abstandsgrößen aus dem Kamerabild zu extrahieren.

Modellbasierte Verfahren treffen Grundannahmen über Form und Größe bekannter Objekte im Fahrzeugumfeld und erzielen die dreidimensionale Umfeldrekonstruktion über den Skalierungsfaktor der Objekte im Bild. Für die hierzu notwendige Objekterkennung wird auf Standard-Bildverarbeitungsalgorithmen zur Merkmalsextraktion und moderne Klassifikatoren zurückgegriffen. Durch Segmentierung des Bildes (Frames) anhand von Diversifikationsmerkmalen [15] (Das Histo-

gram of oriented gradients hat sich dabei als besonders effizient erwiesen [16]) werden die Regionen gefiltert welche ein interessantes Objekt enthalten könnten (Region of Interest, RoI). Die einzelnen vorgeschlagenen RoI werden durch einen Erkennungsalgorithmus geschickt (wie Aggregated Channel Features (ACF [17]) oder ein Convolutional Neural Network (CNN)) welcher auf bestimmte Objektklassen geschult wurde und mit gewisser Sicherheit diese in der RoI wiedererkennen kann. Ausgang der Detektion ist ein Rahmen, der den RoI repräsentiert, in dem das geschulte Objekt erkannt wurde und ein Wert der Rückschlüsse auf die Sicherheit der Detektion liefert. Auf dessen Basis lässt sich eine sog. Bounding Box auflegen, welche durch vorhersagbare Objekteigenschaften auch Aussagen über Größe, Entfernung und Position des lokalisierten Objektes zulassen (z.B.: Detektion: Mensch, durchschnittliche Größe des Menschen: 1,70m).

Alternativ hierzu wird im Rahmen der Stereoskopie über das Prinzip der Korrespondenz auf Basis von Aufnahmen aus verschiedenen Blickrichtungen die Lage eines Bildpunktes in drei Dimensionen trianguliert. Aktuelle Systeme streben eine möglichst dichte Tiefenkarte der Fahrzeugumgebung an. Durch Kombination mit dem optischen Fluss zeitlich folgender Aufnahmen lässt sich ferner eine kombinierte Positions- und Geschwindigkeitsschätzung der Bildpunkte vornehmen, wodurch sich Vorteile bei der nachgeschalteten Objektbildung und -verfolgung ergeben. Steht nur eine Kamera zur Verfügung, lassen sich die Ansätze der Stereoskopie bei statischem Umfeld und bewegter Fahrzeugplattform auf die Auswertung zeitlich aufeinander folgender Aufnahmen erweitern [14], [8].

1.5 Analyse und Bewertung

Die folgenden Abbildung 2 zeigt zusammenfassend die aus den vorherigen Kapiteln bekannten Rechercheergebnisse, qualitativ bewertet nach Kriterien die zur Generierung eines vollständiges Abbilds der Umgebung bei jeglichen Witterungsbedingungen notwendig sind.

Es ist ersichtlich das ein einzelner Sensortyp nicht alle Anforderungen an die Umfelderkennung für autonome Fahrzeuge erfüllen kann. Um die Detektion zu verbessern und eine Klassifizierung zu ermöglichen, müssen die gewonnenen Messwerte fusioniert werden. Das Netzdiagramm liefert dabei die Basis zur Erschaffung eines umfassenden Konzeptes welches die einzelnen Vorteile der jeweiligen Sensortypen ausnutzen und dabei deren Nachteile kompensieren kann.

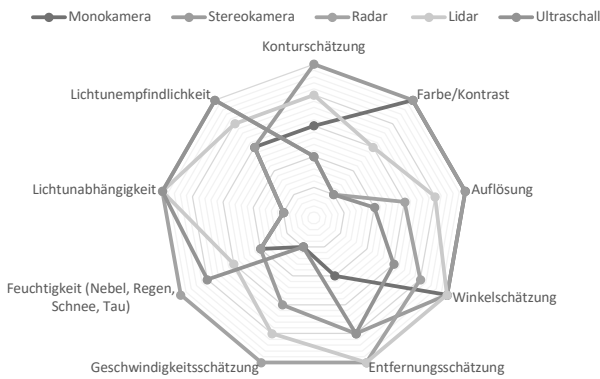


Abbildung 2: Eigenschaften der Umfoldsensoren

2 Methodik und Systemkonzept

Eine hierarchische Strukturierung der Datenverarbeitungsaufgaben bilden die grundlegende Basis- Funktionsarchitektur [19], welche auf Abbildung 3 dargestellt ist.

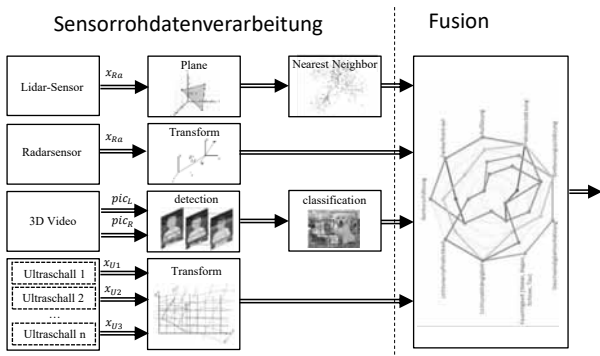


Abbildung 3: Systemkonzept

Die unterste Ebene der Struktur ist die Sensorrohdatenverarbeitung in der sensorspezifische Aufbereitungsalgorithmen implementiert werden. Während die Rohdaten eines Ultraschall oder Radarsensors im einfachsten Fall nur den Positionskoordinaten des Sensors an der Fahrzeugkarosserie zugeordnet werden müssen, sind für andere Umfoldsensoren weitgehendere Maßnahmen notwendig.

Um die Datenverarbeitung der Sensormessergebnisse des LIDAR- Sensors zu beschleunigen wird zunächst eine Selektion relevanter Datenpunkte durchgeführt. Dies umfasst die Deselektion von Eigenfahrzeugdetektionen und das Entfernen von irrelevanten Daten. Dabei ist die Deselektion der Eigenerkennung durch Pa-

rametrierung eines Koordinatenfeldes als Fahrzeugengrenzlinie durchzuführen. Ein einfacher Algorithmus kann dann aus den Rohdatenpunkten diejenigen herauslösen welche innerhalb der parametrisierten Fahrzeugengrenzlinie liegen.

Der zweite Schritt des entfernen irrelevanter Daten bezieht sich auf Datenpunkte welche z.B. auf der Straße oder in Höhen oberhalb des Sicherheitshorizonts des Fahrzeugs liegen. Zum entfernen der Fahrbahndatenpunkte aus der Umfelderkennung müssen diese zunächst als fahrbahnzugehörig erkannt werden. Dazu werden die Koordinatendaten des Sensors mittels Winkellagedaten auf das Weltkoordinatensystem umgerechnet. An diesen Punkten wird eine oder eine Schar von Ebenen angelegt und mittels Analyse der Ebenenlage eine (Fahrbahn-)Fläche erkannt. Die Detektion ist nun wieder einfach über die Höhenlage der detektierten Ebenen und ihre Relation zu weiteren Ebenen zu lösen.

Die Datenverarbeitung des Kamerasystems ist umfangreich, da die Kamera, wie im vorherigen Kapitel beschrieben, selbst gar keine messenden Eigenschaften besitzt. Hier wird zweistufig verfahren um zuverlässige Ergebnisse erhalten zu können. Zunächst werden aus dem Kameraframe Objekte extrahiert. Dieses Extraktion von Objekten geschieht auf Basis segmentierter und aggregierter Kameraeinzelframes in Framerate. Diese gewonnen Pixelfelder in der Größe 16x16 Pixel werden anschließend durch zwei unabhängige Erkennungsfiler, jeweils basierend auf der ACF- Methode, geschickt. Wurde ein Feature erkannt, so wird zu diesem eine Bildkoordinate rückgerechnet auf dessen Basis eine sog. Bounding Boxes gezeichnet und mittels Färbung klassifiziert wird. Diese Bounding Box umgrenzt das detektierte Objekte, dessen Farbe klassifiziert es.

Wird ein Objekt erkannt, so wird die Höhe der Boundig Box in Pixeln perspektivisch auf die Höhe des Objektes umgerechnet. Diese Höhe entspricht der Sichthöhe, welche über den suchenden Abstand der Höhe des realen Objektes zugeordnet werden kann. Invertiert man diesen Prozess kann durch eine Schätzung der realen Höhe des Objektes auf den Abstand zum Sensor zurückgerechnet werden. So kann man für ein Auto oder einen Fußgänger eine Durchschnittshöhe hinterlegen. Hier ist ersichtlich das eine feinere Klassifizierung (z.B. nicht Klasse: Auto sondern Klasse: SUV, Limousine, etc.) zu einem besseren Ergebnis führt. Hier sollen diese Daten aber zunächst nur zur Zuordnung dienen und müssen nicht besonders exakt sein.

Nächsthöhere Ebene ist die Fusionsebene. Diese

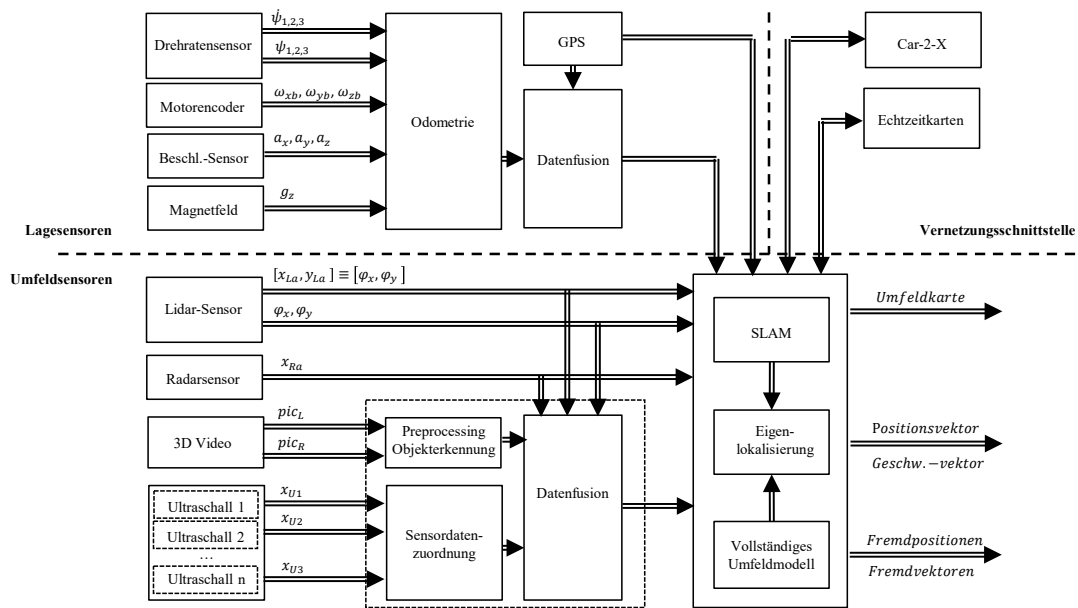


Abbildung 4: Gesamtwirkungsplan der Umfelderfassung

folgt einem Schema welches direkt aus den Sensoreigenschaften abgeleitet werden kann. Auf Basis der von der Kamera detektierten, klassifizierten und über die Merkmalschätzung grob verorteten Objekte werden Fusionsbereiche definiert. Der Radius des Fusionsbereiches wird zunächst soweit ausgedehnt bis ein Detektionspunkt des Radarsensors innerhalb liegt. Die Radardaten werden dem detektierten und klassifizierten Objekt zugeordnet. Anschließend werden innerhalb des Bereiches Lidar-Messpunkte gesammelt welche einen geringen absoluten Abstand zueinander haben. Diese Punkte sind mit hoher Wahrscheinlichkeit Messpunkte auf dem detektierten Objekt. Sind alle Informationen zugeordnet so werden diese anschließend gewichtet. Dabei wird den Geschwindigkeitsmesswerten eines Radarsensors z.B. mehr Vertrauen geschenkt als denen eines Ultraschallsensors. Anders sind die Abstandsdaten eines LIDAR-Sensors höher gewichtet als die eines Radarsensors.

3 Systemrealisierung

Die Abbildung 4 zeigt den Gesamtwirkungsplan der Umfelderfassung mit den angrenzenden Datenerfassungssystemen der Eigenbewegung über Lagesensoren und der Vernetzungsschnittstellen.

Eingänge in das System sind dabei zum einen die in

Kapitel 1 vorgestellten exterozeptiven Umfelsesensoren, aber auch Lagesensorik des Fahrzeugs, sowie indirekte und sekundäre Umfelsesensorik, welche Daten über Kommunikationsschnittstellen einspeist. Die Lagesensoren des Fahrzeugs werden vornehmlich zur Eigenlokalisierung auf der Umfeldkarte, sowie zur Umrechnung von Transformationsfaktoren der Umfelsesensoren genutzt. Dabei werden die Rohdaten aus Drehraten- und Beschleunigungssensoren, etc. über eine Odometrie dem Trägerfahrzeug zugeordnet und in das fahrzeugfeste Koordinatensystem umgerechnet.

Die Vernetzungsschnittstelle kann genutzt werden um weitere Daten des Umfeldes aus Sensoren anderer Verkehrsteilnehmer zu gewinnen, diese Daten können dabei unverarbeitet oder bereits für die Anwendung aufbereitet sein. Es ist z.B. möglich über diese Schnittstelle die Koordinate eines anderen Verkehrsteilnehmers einzuspielen.

Ausgang der Struktur sind eine Karte der Umgebung, welche die detektierten und kategorisierten Objekte, sowie Positions- und Geschwindigkeitsvektor dieser, aber auch Positions- und Geschwindigkeitsvektor des eigenen Fahrzeugs abbildet. Die gewonnenen Daten können nun einer nachgeschalteten Trajektorienplanung übergeben werden, welche die Navigation auf dieser Karte zum gewünschten Zielort übernimmt.

4 Simulationsergebnis

Die Struktur wurde mit Beispieldaten nach [21] getestet. Die folgende Abbildung 5 zeigt ein Auswertediagramm.

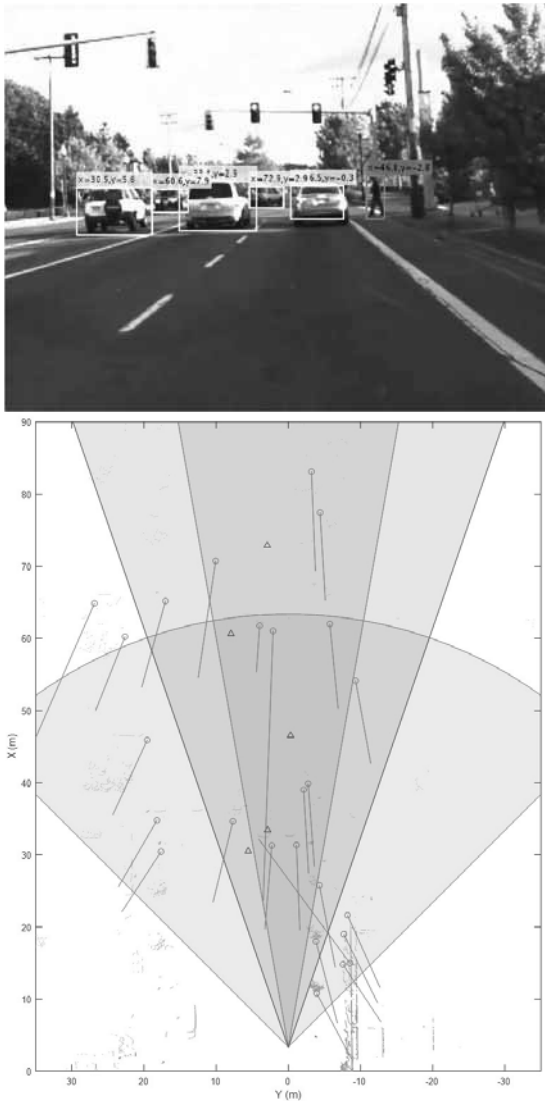


Abbildung 5: Datenauswertung nach Fusion

Die Eingänge des Systems wurden mit einem Kamerabild, der Datenwolke eines LIDAR-Sensors, Radarmesspunkten und den Daten einer inertialen Messeinheit zur Lagedetektion bespielt. Die Synchronisation erfolgte durch zeitstempelbasierte Ereignisfenster.

Das Frame im oberen Bereich der Abbildung, welches aus den Daten einer Mono-Kamera gewonnen wurde, zeigt eine typische Verkehrssituation an einer

Kreuzung mit anderen Verkehrsteilnehmern. Mithilfe der Objekterkennungsalgorithmen wurden in diesem Frame zunächst Objekte identifiziert und klassifiziert. Durch die Bounding Box gelb markierte Objekte sind dabei PKW und rot markierte Objekte Passanten.

Den Objekten wurden anschließend weitere Information zugeordnet. So sind oberhalb der Bounding Box Abstandsinformation hinterlegt, welche direkt oder indirekt gemessen und anschließend auf ihre Plausibilität geprüft und/oder mit anderen Daten fusioniert wurden. Die dafür verwendeten Daten sind im unteren Bereich des Bildes in einer kartesischen Draufsicht dargestellt, welche als Achsenmittelpunkt das Koordinatensystem des eigenen Fahrzeugs aufweist. Die roten Kreise stellen Abstandsinformationen aus Radarmessdaten dar, der an den Kreis angesetzte rote Vektor steht für Geschwindigkeit und Bewegungsrichtung des Objektes zum Fahrzeugkoordinatensystem. Da das Fahrzeug sich selbst bewegt erscheinen hier auch im Weltkoordinatensystem stillstehende Objekte so als wenn sie sich auf das Fahrzeug zubewegen. Blau markierte Dreiecke stehen für Abstandsdaten welche aus der Merkmalschätzung extrapoliert wurden. Diese beziehen sich nur auf klassifizierte Objekte, weshalb nur Fahrzeug im blauen Erfassungsbereich der Kamera dargestellt sind. Die grünen Punkte innerhalb der Darstellung sind Messpunkte eines Lidar-Sensors welche bereits in beschriebener Art und Weise gefiltert wurden. Es sind nur noch Messpunkte dargestellt welche von vorausfahrenden Autos, seitlichen Hindernissen und/oder durch den Radar-Sensor markierten Objekten reflektiert wurden.

5 Zusammenfassung

In diesem Beitrag wurden ausführlich die Eigenschaften bekannter Umfeldsensoren erläutert und über eine Sensordatenbewertung die Notwendigkeit einer Datenfusion hergeleitet. Die Datenfusion wurde grundlegend dargestellt und aus ihr wurde ein Wirkdiagramm des realen Aufbaus abgeleitet. Dieses umfasst neben der eigentlichen Sensordatenfusion auch die unabdingbare Einbindung komplementärer Systeme wie Lage Sensoren. Die Wirksamkeit der Struktur wurde in Kapitel 4 mithilfe von Beispieldaten überprüft.

Die gewonnenen Erkenntnisse sollen weiterhin in die Umsetzung von Umfeldersensoren für andere Forschungsträger der Fachgruppe für Regelungstechnik und Fahrzeugmechatronik fließen.

Referenzen

- [1] Peitsmeier H. Der Autopilot will endlich ans Steuer. *Frankfurter Allgemeine Zeitung*. 02.06.2015;2015.
- [2] Liu-Henke X, Göllner M, Tao H. Modellbasierte Reglerauslegung eines sphärischen Elektroantriebs. In: *Workshop der ASIM/GI-Fachgruppen "Simulation technischer Systeme"*, Herausgegeben durch Wahmkow C, Roßmanek P, Wendorf R, ASIM-Mitteilungen, pp. 37–46. Stralsund: Fachhochschule Stralsund. 2015;.
- [3] Klaus F. *Einführung in die Techniken und Methoden der Multisensordatenfusion*. Habilitation, Universität Siegen, Siegen. 1999.
- [4] Becker JC. *Fusion der Daten der objekterkennenden Sensoren eines autonomen Straßenfahrzeugs: Zugl.: Braunschweig, Techn. Univ., Diss., 2002*, Vol. 948 der *Fortschritt-Berichte / VDI Reihe 8, Meß-, Steuerungs- und Regelungstechnik*. Düsseldorf: VDI-Verl., als ms. gedr. Ausg. 2002.
- [5] Gotzig H, Geduld G. LIDAR-Sensorik. In: *Handbuch Fahrerassistenzsysteme*, Herausgegeben durch Winner H, Hakuli S, Lotz F, Singer C, ATZ / MTZ-Fachbuch, pp. 317–334. Wiesbaden: Springer Vieweg. 2015;.
- [6] Steinmeyer S. *Probabilistische Fahrzeugumfeldschätzung für Fahrerassistenzsysteme*. Dissertation, Technische Universität Carolo-Wilhelmina, Braunschweig. 2014.
- [7] Winner H. Radarsensorik. In: *Handbuch Fahrerassistenzsysteme*, Herausgegeben durch Winner H, Hakuli S, Lotz F, Singer C, ATZ / MTZ-Fachbuch, pp. 259–316. Wiesbaden: Springer Vieweg. 2015;.
- [8] Reif K. *Fahrstabilisierungssysteme und Fahrerassistenzsysteme*. Bosch Fachinformation Automobil. Wiesbaden: Vieweg+Teubner Verlag / GWV Fachverlage GmbH Wiesbaden. 2010.
- [9] Hering E, Schönfelder G. *Sensoren in Wissenschaft und Technik: Funktionsweise und Einsatzgebiete*. Praxis. Wiesbaden: Vieweg+Teubner Verlag, 1st Ausg. 2012.
- [10] Reif K, Hrsg. *Sensoren im Kraftfahrzeug*. Bosch Fachinformation Automobil. Wiesbaden: Springer Vieweg, 2nd Ausg. 2012.
- [11] Sorge G, Hauptmann P. *Ultraschall in Wissenschaft und Technik*, Vol. 46. Frankfurt a.M.: Harri Deutsch Thun. 1985.
- [12] Langen A. *Ein Verfahren zur Konstruktion anwendungsoptimierter Ultraschallsensoren auf der Basis von Schallkanälen*, Vol. 185 der *Fortschritt-Berichte / VDI Reihe 8*. Berlin and Heidelberg: Springer. 1993.
- [13] Tränkler HR, Reindl LM. *Sensortechnik: Handbuch für Praxis und Wissenschaft*. VDI-Buch. Berlin: Springer Vieweg, 2nd Ausg. 2014.
- [14] Effertz J. *Autonome Fahrzeugführung in urbaner Umgebung durch Kombination objekt- und kartenbasierter Umfeldmodelle*. Dissertation, Technische Universität Carolo-Wilhelmina, Braunschweig. 2009.
- [15] Uijlings JRR, van de Sande KEA, Gevers T, Smeulders AWM. Selective Search for Object Recognition. *International Journal of Computer Vision*. 2013; 104(2):154–171.
- [16] Dalal N, Triggs B. Histograms of Oriented Gradients for Human Detection. In: *CVPR 2005*, Herausgegeben durch Schmid C, Tomasi C, Soatto S. Los Alamitos, Calif: IEEE Computer Society. 2005; pp. 886–893.
- [17] Yang B, Yan J, Lei Z, Li SZ. Aggregate channel features for multi-view face detection.
- [18] Weber J. *Globale Selbstlokalisierung für mobile Service Roboter*. Dissertation, Universität Kaiserslautern, Kaiserslautern. 2002.
- [19] Liu-Henke X, Gollner M, Tao H. An intelligent control structure for highly dynamic driving of a spherical electrical drive. In: *2017 Twelfth International Conference on Ecological Vehicles and Renewable Energies (EVER)*. Piscataway, NJ: IEEE. 2017; pp. 1–10.
- [20] Darms M, Winner H. Umfelderfassung für ein Fahrerassistenzsystem zur Unfallvermeidung. In: *Steuerung und Regelung von Fahrzeugen und Motoren - AUTOREG 2006*, VDI-Berichte, pp. 207–2018. Düsseldorf: VDI-Verl. 2006;.
- [21] Mathworks. Automated Driving System Toolbox™ Reference.

Beschleunigung eines Reinforcement-Learning-Algorithmus durch Parallelverarbeitung für Robotikanwendungen

David Jammer¹, Sven Pawletta^{1*}, Georg Kunert¹, Thorsten Pawletta¹

¹Hochschule Wismar – University of Applied Science, Forschungsgruppe CEA
23966 Wismar; *sven.pawletta@hs-wismar.de

Abstract. Machine-Learning (ML) findet derzeit auch in ingenieur-technischen Anwendungen eine große Beachtung. Als problematisch erweist sich dabei häufig der erforderliche Rechenaufwand. Im Beitrag werden Beschleunigungspotentiale für ein Reinforcement-Learning-Verfahren untersucht. Neben Ansätzen der Parallelverarbeitung wird auch das Beschleunigungspotential von Laufzeitumgebungen und effizienten Suchalgorithmen betrachtet.

Einleitung

In der Forschungsgruppe *Computational Engineering & Automation* (FG CEA) der Hochschule Wismar werden seit mehr als 10 Jahren Projekte zum Einsatz von Industrierobotern in der flexiblen Fertigung und Montage durchgeführt. Konzeptionell basieren die Arbeiten auf dem Framework *Simulation Based Control* (SBC,[1]). Das SBC-Framework unterstützt die Methodik des *Rapid Control Prototypings* (RCP, [2]).

Als primäre Software-Plattform für Prototyping sowie operativen Betrieb stehen beim SBC *Scientific Computing Environments* (SCEs) im Mittelpunkt. Bei den SCEs handelt es sich um interpreter-basierte Systeme, die einen komfortablen Zugriff auf umfangreiche Bibliotheken des numerischen und symbolischen Rechnens in Verbindung mit einer matrix-orientierten Programmiersprache bieten. Der Quasistandard in diesem Bereich wird durch das kommerzielle System Matlab von The Mathworks Inc. definiert. Alternativ stehen freie Systeme wie Octave, Scilab, FreeMat und andere zur Verfügung. Höhere interpretierende Sprachen wie Python können ebenfalls zur Klasse der SCEs gezählt werden.

In [3] wurde über ein Projekt der FG CEA zur

Generierung von Robotersteuerungen unter Verwendung eines ML-Verfahrens berichtet. Konkret wurde das Lernverfahren *Reinforcement* (RL) in der Ausprägung *Q-Learning* (QL) in der Variante *Value Based Learning* (VBL) betrachtet und die Integration in das SBC-Framework dargestellt.

Die grundsätzliche Anwendbarkeit von ML-Verfahren im Rahmen des SBC-Frameworks wurde in [3] an einem Beispielproblem geringer Komplexität demonstriert und nachgewiesen (*Türme von Hanoi*, TvH). Für die Anwendung von ML-Verfahren in der Praxis stellt sich jedoch regelmäßig die Frage, welche rechen-technischen Ressourcen für Probleme realer Komplexität erforderlich sind.

Dieser Frage widmet sich der vorliegende Beitrag. Dazu wird eingangs das in [3] gewählte ML-Verfahren soweit eingeführt, dass ein Verständnis der nachfolgenden Betrachtungen möglich ist.

Anschließend wird anhand des Studienbeispiels TvH die Komplexitätsproblematik dargestellt. Im Kontext realer Anwendungen führt eine hohe Komplexität im Zusammenhang mit ML-Verfahren zwangsläufig zu einem sehr hohen Rechenaufwand und damit zu langen Laufzeiten. In der Praxis ist die maximal erlaubte Rechenzeit meist begrenzt. Wenn diese Schranke durch eine konventionelle Implementierung nicht eingehalten werden kann, wird heute häufig versucht, eine Lösung durch *Parallelverarbeitung* (PV) auf Basis der Multi-Core-Technologie zu erzielen. Tatsächlich sind die erreichbaren Beschleunigungsfaktoren auf Ein-Prozessor-Computern aber noch relativ begrenzt. Die Core-Zahl der gegenwärtig leistungsstärksten Prozessoren von AMD und Intel ist 32 beziehungsweise 28.

Im vorliegenden Beitrag werden deshalb zuerst Beschleunigungspotentiale untersucht, die einer PV

vorgelagert werden sollten. Das sind (i) der Wechsel der Laufzeitumgebung und (ii) die Implementierung einer effizienten Algorithmik. Erst zum Schluss (iii) wird auf den Einsatz einer geeigneten PV-Technologie eingegangen.

1 Reinforcement Learning

Gegenstand der Untersuchung in [3] war eine Variante des RL, die als Q-Learning bezeichnet wird. Q-Learning kann nach [4, 5] wiederum nach verschiedenen Ansätzen durchgeführt werden. Der in [3] und hier betrachtete Ansatz ist das Value Based Learning (VBL). Die Grundstruktur des Verfahrens zeigt Abbildung 1.

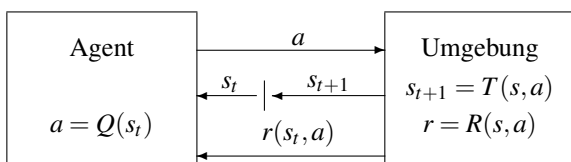


Abbildung 1: Grundstruktur des RL/QL/VBL nach [3]

Der Agent besitzt anfangs kein Wissen über seine Umgebung. Initial kommuniziert die Umgebung ihren momentanen Zustand s_t an den Agenten. Aus Sicht des Agenten kann s_t als Anfangszustand der Umgebung für eine beginnende Explorationsphase interpretiert werden. Die Abbildung $a = Q(s_t)$ steht im Agenten für das erlernte Wissen über die Umgebung. Verfügt der Agent über ausreichendes Wissen, so kann er auf Basis des momentanen Zustands s_t eine Aktion a auswählen und an die Umgebung kommunizieren, die dort zu einer sinnvollen Zustandsänderung im Sinne einer Zielvorgabe führt. In der Regel besteht die Zielvorgabe im Erreichen eines bestimmten Zielzustandes über den Weg möglichst weniger Zustandsänderungen.

Anfangs kann der Agent Aktionen a nur zufällig generieren, weil er noch kein Wissen besitzt. Die in der Umgebung auf a folgende Zustandsänderung wird durch $s_{t+1} = T(s, a)$ beschrieben. Da es sich bei den betrachteten Umgebungen jeweils um Simulationen einer realen Umgebung handelt, kann T auch als Zustandsübergangsmodell bezeichnet werden. Die auf a folgende Änderung in den Zustand s_{t+1} wird wieder an den Agenten kommuniziert. Auf diese Weise erkundet der Agent den Zustandsraum der Umgebung und lernt dabei, welche Aktionen zulässig sind. Auf unzulässige Aktionen reagiert T mit $s_{t+1} = s_t$, der Zustand bleibt also unverändert. Darüber hinaus besitzt

die Umgebung ein Belohnungsmodell $r = R(s, a)$. Die Belohnungswerte r (reward) werden ebenfalls an den Agenten kommuniziert. Welche Werte r zum Einsatz kommen, hängt vom jeweiligen Anwendungsproblem ab. Übliche Belohnungswerte sind:

- $-\infty$: Aktion a ist unzulässig
- 0: Aktion a ist zulässig. Der resultierende Zustand s_{t+1} ist noch kein Zielzustand.
- 1: Aktion a ist zulässig. Der resultierende Zustand s_{t+1} ist der gewünschte Zielzustand.
- -1: Aktion a ist zulässig. Der resultierende Zustand s_{t+1} ist aber unerwünscht.

Die erste Exploration endet, wenn der gewünschte Zustand in der Umgebung erreicht ist ($r = 1$). Ein solcher Durchlauf wird als Episode bezeichnet. Alle in einer Episode erreichten Zustände sowie die empfangenen Belohnungen werden im Agenten gespeichert.

Durch wiederholte Explorations-Episoden wird ein immer größerer Teil des Zustandsraums der Umgebung nach dem Trial-and-Error-Prinzip erkundet. Abhängig von der konkreten Anwendung besitzt der Agent nun Wissen über mindestens eine Zustandsfolge, die von s_t nach s_{Ziel} führt. Nun kann ein zweiter Mechanismus zur Wirkung kommen, der als Exploitation bezeichnet wird. Damit ist das Lernen auf Basis bereits erworbenen Wissens gemeint.

Grundsätzlich ist das Verhältnis von angemessener Exploration zu Exploitation in fortgeschrittenen Episoden nicht trivial. Für einen weitergehenden Einstieg in die Problematik sei an dieser Stelle auf [3] und darüber hinaus auf die umfangreiche Primärliteratur verwiesen.

2 Studienbeispiel TvH

Die in [3] vorgestellte Arbeit verfolgt das Ziel, den Einsatz eines RL-Verfahrens im Kontext der Industrierobotik zu untersuchen. Hier werden Gelenkarmroboter häufig für Transport-, Fertigungs- und Montageaufgaben eingesetzt. Gerade vor dem Hintergrund von Montageaufgaben ist die Wahl des Einspieler-Problems Türme von Hanoi (TvH) sehr anschaulich. In seiner minimalen Komplexität mit zwei Scheiben und drei Stäben ist es sogar vollständig grafisch darstellbar, wie Abbildung 2 zeigt.

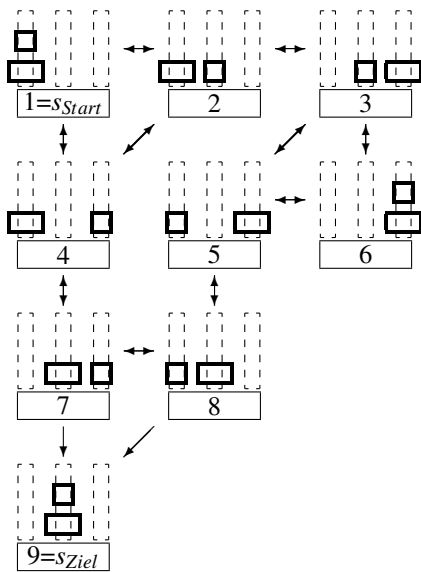


Abbildung 2: Mögliche Spielzüge bzw. Zustände und Übergänge bei TvH minimaler Komplexität nach [3]

Über die Parameter *Anzahl der Scheiben* und *Anzahl der Stäbe* lässt sich eine beliebige Problemkomplexität einstellen. Anhand der Abbildung 2 sind Start, Ziel und Regeln von TvH leicht nachvollziehbar:

1. Größere Scheiben dürfen *nie* über kleineren liegen. Dies gilt auch für den Startzustand.
2. Startzustand: Alle Scheiben liegen auf Stab 1.
3. Zielzustand: Alle Scheiben liegen auf Stab 2.
4. Bei jeder Aktion kann immer nur eine oben liegende Scheibe von einem Stab zu einem anderen Stab transportiert werden (Analogie zu *Pick&Place* in der Robotik).

In seiner minimalen Komplexität besitzt TvH 9 mögliche Zustände und 6 Aktionen.

3 Beschleunigungspotential durch Wechsel der Laufzeitumgebung

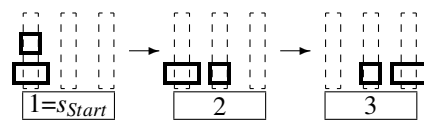
Das Ziel der in [3] publizierten Arbeit bestand im Nachweis einer prinzipiellen Machbarkeit. Der dazu erforderliche Prototyp wurde in einer SCE implementiert und für diesen Beitrag unter folgenden Bedingungen nochmals ausgemessen:

- Hardware-Plattform: Intel Core i7 8th Gen., 6 Cores plus Hyper-Threading Technology (HTT), 32 GB Hauptspeicher
- Betriebssystem: GNU/Linux 4.15.0-43-generic / Ubuntu 18.04 LTS
- SCE: Matlab 2018a
- TvH-Komplexität: 6 Scheiben, 3 Stäbe
- Laufzeit: 6,5 Sekunden im Mittel

Angesichts der noch relativ geringen Problemkomplexität ist eine mittlere Laufzeit von 6,5 Sekunden bereits bedenklich hoch, so dass Überlegungen bezüglich nutzbarer Beschleunigungspotentiale begründet sind. In einem ersten Schritt wurde der SCE-Prototyp ohne strukturelle Änderungen in C reimplementiert. Damit konnte ein Wechsel der Laufzeitumgebung von Matlab 2018a nach GCC 7.3.0 erfolgen. Die anschließend gemessene Laufzeit betrug 2,0 Sekunden, was einem Speedup-Faktor von mehr als 3 entspricht.

4 Beschleunigte Suche durch Binärbäume

Abbildung 3 zeigt die Datenstruktur DS nach zwei ausgeführten Aktionen in der initialen Explorationsepisode.



Datenstruktur DS

S_1	1	2	0	0	0	0
S_2	0	2	0	1	0	0
S_3	0	0	0	1	0	2

Abbildung 3: Speicherung explorierter Zustände im Agenten

Mittels DS werden sämtliche explorierten Zustände der Umgebung im Agenten als jeweils eine Zeile dauerhaft gespeichert. Nach jeder Aktion muss geprüft werden, ob der resultierende Zustand in DS bereits vorhanden ist oder als neue Zeile angefügt werden muss. Der bisherige Prototyp verwendete dafür eine $O(n)$ -Suche. In einer detaillierten Laufzeitanalyse wurde festgestellt, dass die $O(n)$ -Suche einen Anteil von 99% an der Gesamtlaufzeit besitzt.

Daraufhin wurde die Suche in DS unter Verwendung eines Binärbaums reimplementiert. Die notwendige Basisfunktionalität steht in der C-Bibliothek der GCC-Umgebung bereits zur Verfügung (tsearch(3)). Die dortige Implementierung basiert auf Donald Knuth [6].

Durch die Umstellung reduziert sich die Komplexität der Suche auf $O(\log_2(n))$. Die Laufzeit der bisher untersuchten TvH-Konfiguration sinkt im Mittel von 2 Sekunden auf 0,06 Sekunden, was einem Speedup-Faktor von 33 entspricht.

5 Parallelisierung mittels POSIX-Threads

POSIX-Threads [7] sind eine standardisierte, von fast allen Betriebssystemen unterstützte Technik zur Realisierung mehrerer Kontrollflüsse (Threads) innerhalb eines gemeinsamen Adressraums (Prozess). Demzufolge ist eine sehr effiziente Inter-Thread-Kommunikation nach dem Shared-Memory-Modell (SHM) möglich. Häufig werden Thread-Anwendungen im SPMD-Stil realisiert (*Single Programm Multiple Data*). SPMD-Implementierungen können sowohl sequentiell auf einem Core als auch parallel auf mehreren Cores laufen.

Thread-basierte Programme können auch unter Nutzung nur eines Cores gegenüber einer konventionellen Implementierung einen Speedup aufweisen. Dies wird bei heutigen Prozessoren durch Hyper-Threading und ausgefeilte Cache-Technologien ermöglicht. Bei der bisher untersuchten TvH-Konfiguration ergibt sich dadurch nochmals ein Speedup von 1,2.

Wegen des verwendeten SPMD-Stils kann die Thread-Implementierung ohne Änderungen auch mehrere Cores für PV benutzen. Da die mittlere Laufzeit der untersuchten TvH-Konfiguration aber nur noch 50 Millisekunden beträgt, ist ihr sogenanntes Problemgewicht inzwischen aber so gering, dass sie von PV auf Multi-Core-Technologie nicht mehr profitieren kann.

Wie Tabelle 1 zeigt, können aber komplexere TvH-Konfigurationen durchaus durch PV noch weiter beschleunigt werden.

	Speedup	Threads	Zustände
8 Scheiben 3 Stäbe	2,3	8	6.564
10 Scheiben 4 Stäbe	2,5	14	1.048.583
10 Scheiben 5 Stäbe	4,3	14	9.765.625

Tabelle 1: Speedup ausgewählter TvH-Konfigurationen

6 Zusammenfassung

In diesem Beitrag wurden verschiedene Möglichkeiten zur Beschleunigung einer ML-Anwendung untersucht. Das größte Einzelpotential wies dabei die Verbesserung der Algorithmik auf. In Kombination tragen jedoch alle Maßnahmen zur Beschleunigung bei. Für die TvH-Konfiguration 6 Scheiben, 3 Stäbe wurde ein Gesamt-Speedup von 130 erreicht. Der PV-Anteil an der Gesamtbeschleunigung wächst mit zunehmender Problemkomplexität.

In weiteren Arbeiten sollen weitere Ansätze zur Beschleunigung von ML-Verfahren wie SIMD-Technologien und NUMA-Architekturen untersucht werden.

Literatur

- [1] Freyman B, Pawletta S, Schmidt A, Pawletta T. Design, Simulation and Optimization of Task-Oriented Multi-Robot Applications with MATLAB/Stateflow. *SNE - Simulation News Europe*. 2016;26(2):83–90. doi: 10.11128/sne.26.2.1033.
- [2] Abel D, Bollig A. *Rapid Control Prototyping - Methoden und Anwendungen*. Springer Verlag. 2006.
- [3] Kunert G, Pawletta T. Generating of Task-Based Controls for Joint-Arm Robots with Simulation-Based Reinforcement Learning. *SNE - Simulation Notes Europe*. 2018;28(4):149–156. 10.11128/sne.28.tn.10442.
- [4] Sutter S R, Barton G A. Reinforcement Learning: An Introduction. *Cambridge/MA: MIT Press*. 2012;(2):324.
- [5] Akhtar S. Practical Reinforcement Learning. *Birmingham/UK: Packt Publishing Ltd*. 2017;(1):320.
- [6] Donald E Knuth. *The Art of Computer Programming Volume 3*. Addison Wesley, 2nd ed. 1998.
- [7] Wolf J, Wolf KJ. *Linux-Unix-Programmierung*. Reihnwerk Verlag. 2016.

Simulative Untersuchung von Betriebserweiterungen in einem Aluminium- Schmelz- und Druckgussbetrieb anhand von Modellen mit unterschiedlichem Detaillierungsgrad

Johannes Dettelbacher*, Wolfgang Schlüter

Hochschule Ansbach, Residenzstraße 8, 91522 Ansbach, *j.dettelbacher@hs-ansbach.de

Abstract. Investitionen für Betriebserweiterungen in der Industrie sind stets mit einem bestimmten Risiko verbunden und werden dahingehend sehr zurückhaltend getätigt. Eine Abhilfe schafft hierbei die Simulationstechnik, welche schnell Aussagen über den Ausbau eines Betriebes zulässt und somit das Risiko minimiert. Unklar ist dabei jedoch, wie detailliert das benötigte Modell den realen Betrieb abbilden muss. Im Rahmen dieser Arbeit werden Simulationsmodelle eines Aluminium- Schmelz- und Druckgussbetriebes mit unterschiedlichem Detaillierungsgrad aufgebaut. Beim vereinfachten Modell werden die jeweiligen Maschinengruppen als Flüssigaluminiumspeicher betrachtet und mithilfe von Bilanzgleichungen beschrieben, während im detaillierten Modell die Maschinen und Transporter mit den Prozessschritten einzeln abgebildet werden. Anhand der aufgebauten Modelle werden verschiedene Betriebserweiterungen mit dem Ziel untersucht, den notwendigen Detaillierungsgrad des Modells zu ermitteln. Als Betriebserweiterung werden zum einen das Hinzufügen von Schmelzöfen bzw. Materialquellen und zum anderen das Hinzufügen von Druckgussmaschinen bzw. Materialsäcken mit Hinsicht auf Energieeffizienz und Ausbringung des Betriebes analysiert.

Einleitung

Eine Erweiterung eines Betriebes um Produktionsmaschinen steigert die Produktionskapazität und kann somit langfristig die Wirtschaftlichkeit im Unternehmen erhöhen. Während die Investitionskosten der Betriebserweiterungen in der Regel kalkuliert werden können, ist der genaue Nutzen bei komplexeren Betriebsstrukturen, wie in der Nichteisen-Industrie, nur abschätzbar. Eine genauere Betrachtung, ohne die Produktion zu beeinflussen, bietet die Simulationstechnik. Es ist jedoch unklar, wie

detailliert das Simulationsmodell den realen Betrieb abbilden muss, um aussagekräftige Ergebnisse zu erhalten. Um diese Frage zu klären, werden im Rahmen dieser Untersuchung unterschiedlich detaillierte Simulationsmodelle eines Aluminium-Schmelz- und Druckgussbetriebes aufgebaut und hinsichtlich Betriebserweiterungen untersucht.

1 Aluminium Schmelz und Druckgussbetrieb

Die vorliegende Untersuchung betrachtet einen mittelgroßen und einen großen Aluminium-Schmelz- und Druckgussbetrieb. Ein typischer Aufbau eines solchen Betriebes ist in Abbildung 1 dargestellt. Der Betriebsablauf beinhaltet sowohl kontinuierliche (z. B. Schmelzen) als auch ereignisdiskrete (z. B. Staplertransport) Prozessschritte.

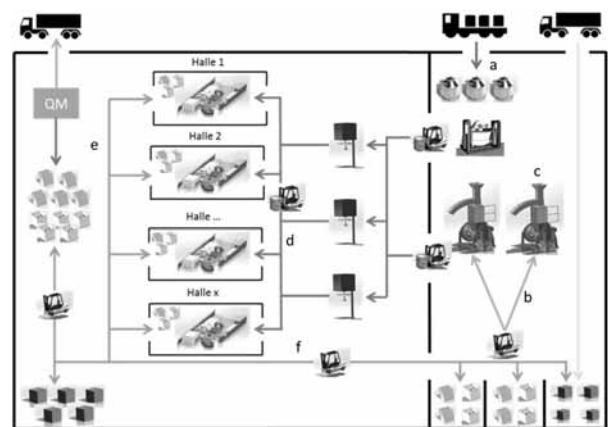


Abbildung 1: Schema eines Aluminium- Schmelz- und Druckgussbetriebs mit Prozessschritten [1]

Die zugrundeliegenden Prozesse sind:

- Anlieferung von flüssigem Aluminium (Abbildung 1, a),
- Beschickung der gasbetriebenen Schachtschmelzöfen über Stapler mit Masseln, Rücklauf- oder Ausschussmaterial (Abbildung 1, b)
- Erwärmen, Schmelzen und Überhitzen bzw. Warmhalten des Metalls (Abbildung 1, c)
- Verteilung des flüssigen Aluminiums mit Staplern auf die Dosieröfen der Druckgussmaschinen (Abbildung 1, d)
- Produktion von Gussteilen in den Druckgussanlagen und Qualitätsprüfung (Abbildung 1, e)
- Transport von Materialbehältern aus dem Druckgussbetrieb oder von Masselpaketen aus dem Lager zum Schmelzbetrieb (Abbildung 1, f)

Die Versorgung des Betriebes mit Flüssialuminium kann auf zwei Wegen realisiert werden. Zum einen werden die Schmelzöfen über Stapler mit festen Aluminium beliefert, in welchen das Aluminium in Form von Masseln, Rücklauf- oder Ausschussmaterial aufgeschmolzen und in der Ofenwanne bis zur Entnahme warmgehalten wird. Alternativ kann das Flüssialuminium aus externen Schmelzereien direkt angeliefert und bis zur Entnahme in einer Kippstation gelagert werden.

Die Schmelz- und Warmhaltevorgänge im Schmelzofen sind hierbei besonders energieintensive Prozesse und somit maßgeblich für die Energieeffizienz des Betriebes. So benötigen die Schmelzöfen etwa 50 % des betrieblichen Gesamtenergiebedarfs. Deswegen liegt bei der Berechnung des Energieverbrauchs der Fokus auf dem Schmelz- und Warmhalteprozess.

Für den Materialfluss sind hingegen alle Prozessschritte relevant. So wird das flüssige Aluminium nach dem Schmelzprozess aus der Ofenwanne des Schmelzofens entnommen und mit Hilfe von Staplern zu den Druckgussmaschinen gebracht. Die Stapler agieren hierbei nach vordefinierten Regeln: Druckgussmaschinen, die nur noch wenig Flüssialuminium zur Produktion zur Verfügung haben, werden priorisiert beliefert.

Bei den Druckgussmaschinen erfolgt die eigentliche Produktion der Teile. Hierbei werden je nach Maschine unterschiedliche Bauteile gegossen, was zu einem maschinenspezifischen Verbrauch an Flüssialuminium führt. Nach der Produktion erfolgt die Qualitätskontrolle, welche fehlerhafte Gussteile und Rücklaufmaterial wieder zu den Schmelzöfen zurückführt. Eine mögliche Erweiterung eines solchen Betriebes ist das Hinzufügen eines

Schmelzofens, einer Druckgussmaschine oder eines Staplers. In Rahmen der folgenden Untersuchungen wurden nur der Zubau von Schmelzöfen und Druckgussmaschinen untersucht, da bei den untersuchten Betrieben an den Staplern kein betrieblicher Engpass auftritt.

2 Simulationsmodelle

Für die Untersuchung wurde vom Detaillierungsgrad unterschiedliche Simulationsmodelle aufgebaut, welche den Aluminium- Schmelz- und Druckgussbetrieb abbilden. Um die Modelle, welche sowohl kontinuierliche als auch ereignisdiskrete Elemente beinhalten, mathematisch zu beschreiben, werden hybride Automaten verwendet [3]. Die Simulation der Modelle erfolgt mit den Software-Tools MatLab, Simulink und Stateflow.

2.1 Vereinfachtes Modell

Beim vereinfachten Modell werden die jeweiligen Maschinengruppen als Flüssialuminiumspeicher betrachtet. Es werden jeweils die Materialquellen (Schmelzöfen und Kippstation), die Transporter (Stapler) sowie die Materialsenken (Druckgussmaschinen) zu einem Speicher zusammengefasst (Abbildung 2).

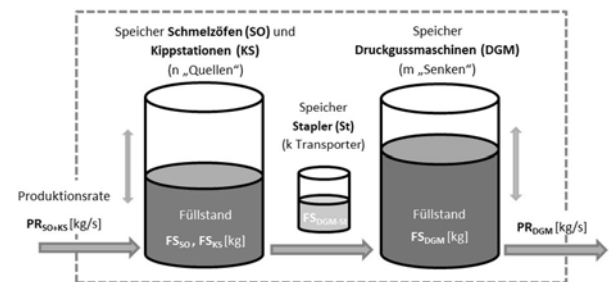
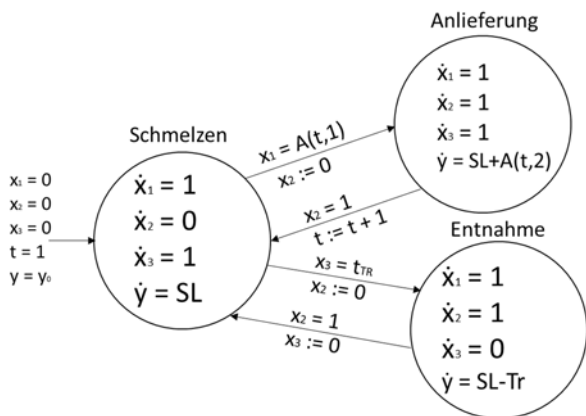


Abbildung 2: Maschinen als Flüssialuminiumspeicher [2]

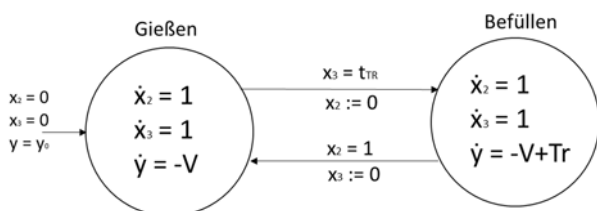
Der Materialquellenspeicher (in Abbildung 2 links) hat als Zufluss das kontinuierlich geschmolzene Aluminium und die ereignisdiskrete Anlieferung von Flüssialuminium. Die ereignisdiskrete Entnahme erfolgt durch Stapler. Das Modell der Materialquelle besteht aus den drei Zuständen Schmelzen, Anlieferung und Entnahme. Ein Übergang zwischen den Zuständen findet dann statt, wenn die Zeitvariablen die Sprungbedingung für ein Ereignis erfüllen. Der Schmelzvorgang ist in alle Zuständen aktiv. Zusätzlich wird je nach Zustand Flüssialuminium angeliefert oder entnommen.



- x_1 Uhr für die Gesamtzeit zur Bestimmung der Anlieferung von Flüssigaluminium
- x_2 Uhr für die Bestimmung der Dauer in den Zuständen Anlieferung und Entnahme
- x_3 Uhr für die Bestimmung der zyklischen Entnahme durch den Stapler
- y Füllstand des Flüssigaluminiumspeicher
- y_0 Anfangsfüllstand des Flüssigaluminiumspeicher
- SL Schmelzleistung aller Schmelzöfen
- A Array mit Ankunftszeiten und Anlieferungsmengen
- t_{TR} Zeitdauer bis zur erneuten Aluminiumentnahme bzw. Aluminiumzugabe durch den Stapler
- Tr Entnahmemenge durch Transporter
- t Position im Array A

Abbildung 3: Modell Materialquelle

Der Materialsenkenspeicher (in Abbildung 2 rechts) hat als Zufluss die ereignisdiskrete Befüllung durch den Stapler und als Abfluss den Aluminiumverbrauch durch den Gießprozess. Der Gießprozess wird vereinfacht als kontinuierlicher Vorgang angenommen, da hierbei die Intervallzeiten relativ klein sind. Es ergeben sich somit die beiden Zustände Gießen und Befüllen.



- x_2 Uhr für die Bestimmung der Dauer in den Zustand Befüllen
- x_3 Uhr für die Bestimmung der zyklischen Befüllung
- V Materialverbrauch aller Druckgussmaschinen
- Tr Befüllmenge durch den Stapler (ist gleich der Entnahmemenge)

Abbildung 4: Modell Materialsenke

Der Materialspeicher der Transporter spielt für die folgenden Untersuchungen keine Rolle und wird daher nicht weiter betrachtet. Da im vereinfachten Modell die einzelnen Schmelzöfen nicht abgebildet werden, kann der Energieverbrauch nur über die Schmelzleistung abgeschätzt werden. Hierzu wird der Energieverbrauch aus der Schmelzleistung und dem mittleren Energieverbrauch bestimmt.

2.2 Detailliertes Modell

Beim detaillierten Modell werden die einzelnen Materialquellen und Materialsenken, wie in Abbildung 5 dargestellt wird, als separate Speicher betrachtet. Durch die differenzierte Betrachtung wird ermöglicht, dass der Ausfall einzelner Maschinen aufgrund von Aluminiummangel abgebildet werden kann. Zudem können den Maschinen spezifische und zeitabhängige Schmelzleistungen bzw. Materialverbräuche sowie Energieverbräuche zugewiesen werden.

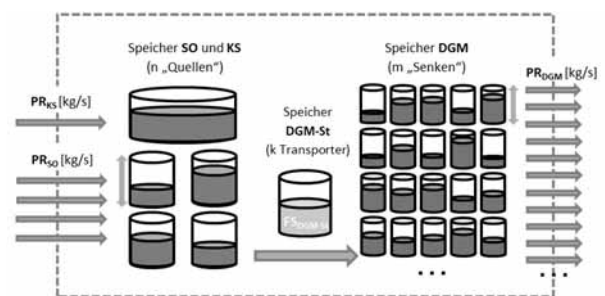


Abbildung 5: Maschinenspezifische Betrachtung der Flüssigaluminiumspeicher

Somit wird auch die Kippstation getrennt von den Schmelzöfen betrachtet. Für diese sind drei Zustände möglich: Anlieferung des Flüssigaluminiums, Entnahme des Aluminiums und ein Zustand, bei dem keine Veränderung des Füllstands eintritt.

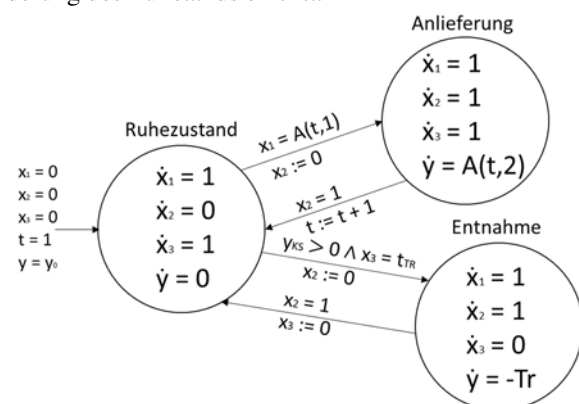
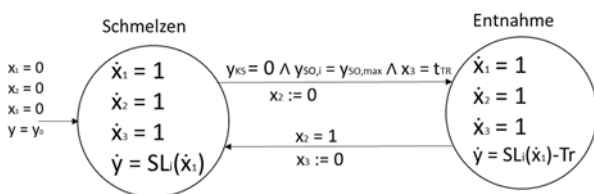


Abbildung 6: Modell Kippstation

Da im detaillierten Modell mehrere Materialquellen und -senken vorhanden sind, bedarf es einer Steuerung, welche vorgibt aus welcher Materialquelle entnommen und welche Senke beliefert werden muss. Die Steuerung der Stapler erfolgt über die Füllstände der Maschinen. Die Steuerungslogik gibt vor, dass die Kippstation (KS) als Materialquelle priorisiert genutzt wird, sofern diese Aluminium beinhaltet. Wenn die Kippstation kein Aluminium vorrätig hat, wird aus dem Schmelzofen (SO) mit dem maximalen Füllstand (Gleichung 1) entnommen.

$$y_{SO,max} = \max(y_{SO,1}, y_{SO,2}, \dots, y_{SO,m}) \quad (1)$$

Die Schmelzöfen werden einzeln beschrieben. Für die Schmelzöfen gilt jeweils:



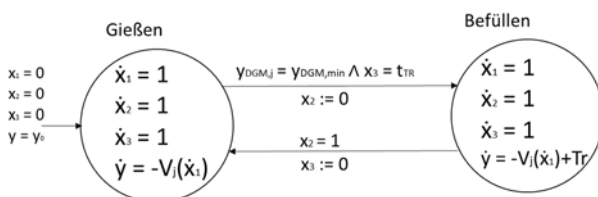
$SL_i(\dot{x}_1)$ Spezifische und zeitabhängige Schmelzleistung des Schmelzofens i

Abbildung 7: Modell Schmelzöfen

Als zu beliefernde Materialsenke bzw. Druckgussmaschine wird die ausgewählt, welche den niedrigsten Füllstand (Gleichung 2) aufweist.

$$y_{DGM,min} = \min(y_{DGM,1}, y_{DGM,2}, \dots, y_{DGM,m}) \quad (2)$$

Die Druckgussmaschinen werden jeweils folgendermaßen beschrieben:



$V_j(\dot{x}_1)$ Spezifischer und zeitabhängiger Verbrauch an Flüssigaluminium der Druckgussmaschine j

Abbildung 8: Modell Druckgussmaschinen

Auch beim detaillierten Modell werden die Schmelzschächte nicht abgebildet. Der Energieverbrauch wird über die ofenspezifische Schmelzleistung und dem mittleren ofenspezifischen Energieverbrauch ermittelt.

2.3 Hochdetailliertes Modell

Im hochdetaillierten Modell werden, bis auf die Lagerhaltung, alle Prozessschritte aus der Abbildung 1 abgebildet. Ein besonderer Fokus liegt hierbei auf den Schmelzschacht bzw. Schmelzprozess und dem damit verbundenen Energiemodell. Das hochdetaillierte Simulationsmodell umfasst noch weitere Funktionalitäten wie die Berücksichtigung von Ausfallzeiten, unterschiedlichen Legierungen und unterschiedlichen Produkten. Das Modell wird von Buswell und Schlüter in [4, 5] ausführlich beschrieben. Eine Beschreibung mit hybriden Automaten würde den Rahmen dieser Arbeit sprengen.

Der grundlegende Simulationsablauf ist in Abbildung 9 dargestellt. Das Materialflussmodell erfasst den kompletten Materialfluss innerhalb des Betriebes, während das Energieflussmodell die thermodynamischen Vorgänge innerhalb der Schmelzöfen erfasst. Die Synchronisierung erfolgt in jeden Zeitschritt mithilfe eines Schnittstellenobjekts, welches den Austausch der Daten zwischen beiden Modellen realisiert.

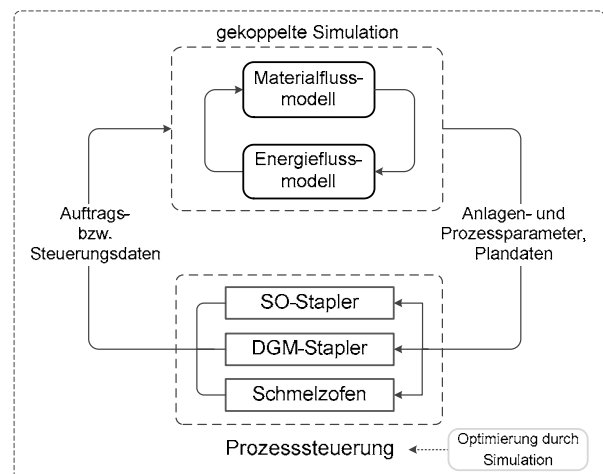


Abbildung 9: Bestandteile der Betriebssimulation [4]

Für die Steuerung der Betriebsabläufe werden in einem Steuerungsmodul anhand von Anlagen- und Prozessparametern sowie den definierten Steuerungsstrategien Aufträge für die Stapler und die Schmelzöfen erzeugt.

3 Simulationsergebnisse

Um repräsentative Ergebnisse zu erhalten, wird die Untersuchung anhand von zwei ausgewählten Betrieben durchgeführt. Betrachtet werden ein großer Betrieb mit 4

Schmelzöfen, 31 Druckgussmaschinen und zusätzlicher Anlieferung von Flüssigaluminium sowie ein mittelgroßer Betrieb mit 5 Schmelzöfen, 24 Druckgussmaschinen und ohne zusätzlicher Anlieferung von Flüssigaluminium. Für jeden Betrieb werden 3 Szenarien simuliert: Der Betrieb ohne Betriebserweiterung („normal“), der Betrieb mit zusätzlichem Schmelzofen („neuer SO“) sowie der Betrieb mit zusätzlicher Druckgussmaschine („neue DGM“). Die Szenarien werden jeweils mit den Modellen unterschiedlicher Detaillierungstiefe simuliert und die Simulationsdauer wird jeweils auf eine Betriebswoche festgelegt. Das vereinfachte Modell wird im folgenden Kapitel als Modell 1, das detaillierte Modell als Modell 2 und das hochdetaillierte Modell als Modell 3 bezeichnet. Der Fokus bei den Ergebnissen liegt auf die Ausbringung und der Energieeffizienz des Betriebes.

3.1 Ausbringung

Die Ausbringung wurde anhand der verarbeiteten Masse an den Druckgussmaschinen berechnet. Die simulierte Ausbringung der Szenarien und Modelle ist in Abbildung 10 dargestellt.

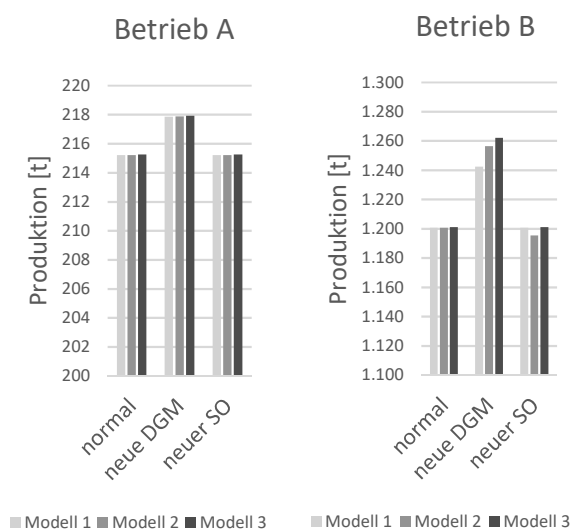


Abbildung 10: Ausbringung der Betriebe

Bei der Ausbringung zeigt sich bei allen Modellen die gleiche Tendenz. Die Erweiterung mit einem Schmelzofen hat keinen nennenswerten Einfluss, während die Erweiterung mit einer Druckgussmaschine die Ausbringung des Betriebs erhöht. Die Ergebnisse zeigen, dass jeweils Engpässe an den Druckgussmaschinen vorhanden sind. Die größte Abweichung zwischen den Modellen

ergibt sich bei der Erweiterung um eine Druckgussmaschine im Betrieb B. Die Abweichung zwischen den Modellen beträgt hier bis zu 1,6 % bezogen auf die gesamte Ausbringung bzw. bis zu 31,7 % bezogen auf die Ausbringungsdifferenz zum Betrieb ohne Betriebserweiterung.

3.2 Spezifischer Energieverbrauch

Zur Beurteilung der Energieeffizienz dient der spezifische Energieverbrauch. Hierfür wurde der Gesamtenergieverbrauch durch die Gesamtmasse des ausgebrachten Materials dividiert. Dargestellt wird der spezifische Energieverbrauch der beiden Betriebe in Abbildung 11.

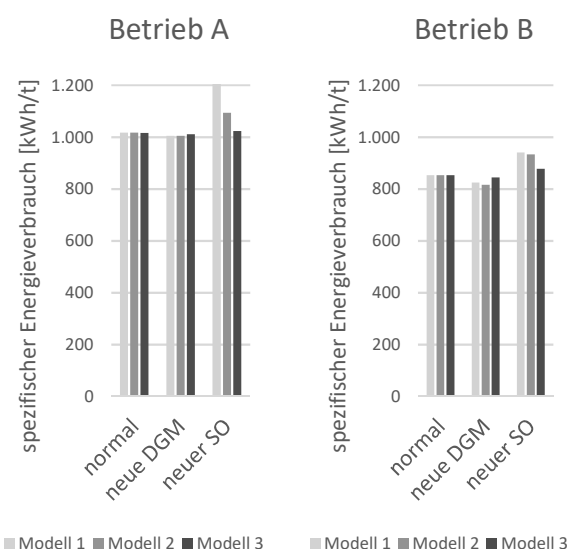


Abbildung 11: spezifischer Energieverbrauch der Betriebe

Es zeigt sich, dass das Hinzufügen einer Druckgussmaschine zu einem geringen Rückgang des spezifischen Energieverbrauches führt. Das Hinzufügen von Schmelzöfen führt hingegen zu einer Erhöhung des spezifischen Energieverbrauches. Hierbei zeigt sich jedoch eine Streuung zwischen den Modellen. Um so einfacher die Betrachtung durchgeführt wurde, umso höher der simulierte spezifische Energieverbrauch. Beim Betrieb A entsteht beim Szenario mit einem neuen Schmelzofen eine Abweichung von etwa 185 kWh/t zwischen den Modellen. Dies entspricht einer Abweichung von 18 % bezogen auf den gesamten spezifischen Energieverbrauch.

3.3 Fazit

Die Ergebnisse zeigen, dass die vereinfachten Modelle

für eine erste Abschätzung zwar schon brauchbare Ergebnisse liefern. Gerade in den interessanten Fällen der Erhöhung der Ausbringung durch eine zusätzliche Druckgussmaschine und der Erhöhung des spezifischen Energieverbrauches führen die Modelle jedoch abhängig vom Detaillierungsgrad zu signifikant abweichenden Ergebnissen. Eine Erhöhung des Detaillierungsgrades führt hier zu genaueren Ergebnissen, die helfen, die Auswirkungen einer Betriebserweiterung fundierter zu beurteilen.

4 Ausblick

Eine weitere Untersuchung von spezifischen Engpasssituationen, die durch einen erhöhten Flüssigaluminiumverbrauch der produzierenden Druckgussmaschinen erzeugt wird, durch Simulationen mit unterschiedlichem Detaillierungsgrad, bietet sich an. Die Engpasssituation kann durch die Hinzunahme von neuen Druckgussmaschinen aber auch durch Produktionsänderungen wie z. B. durch einen geänderten Produktionsplan oder weniger Maschinenausfälle durch eine verbesserte Instandhaltung verursacht werden, wodurch sich die Nachfrage nach Flüssigaluminium signifikant erhöhen kann. Hierbei interessiert vor allem die Prognosegüte der vereinfachten Modelle. Der Vorteil der Simulationen mit geringerem Detaillierungsgrad liegt in der Zeitersparnis, die komplexe Anwendungen erst ermöglicht. Dazu zählen beispielsweise Optimierungsprobleme oder auch die modellprädiktive Steuerung des Produktionsablaufes. Der Vorteil der geringeren Simulationszeit muss mit dem Nachteil der fehlenden Genauigkeit abgeglichen und bewertet werden.

Quellenangabe

- [1] D. Jeckle. *Dokumentation der Software zur Simulation des Materialflusses und Untersuchung der Energieeffizienz eines Schmelz- und Druckgussbetriebes*. Studienarbeit, Hochschule Ansbach, 2015
- [2] J. Schmidt, W. Schlüter. *Ein dynamisches Prozesssimulationsmodell für die energetische Betrachtung von Aluminium-Schmelzöfen in einer betriebsumfassenden Materialflusssimulation*. In: Workshop der ASIM/GI Fachgruppen STS und GMMS 2016, Hamm: ARGESIM Verlag Wien, 2016, pp. 29-37.
- [3] R. Atterer. *Hybride Automaten*. Technische Universität München, 2001.
- [4] A. Buswell, W. Schlüter. *E|Melt: Erweiterung einer unternehmensspezifischen Materialfluss- und Energiesimulation zur Abbildung variabler Betriebsstrukturen der*

Nichteisen- Schmelz- und Druckgussindustrie. In: Tobias Loose (Hg.): Tagungsband Workshop 2018 ASIM/GI-Fachgruppen. Hochschule Heilbronn, 8. und 9. März 2018. Wien: ARGESIM Verlag (ARGESIM Report, 54), S. 33–38.

- [5] A. Buswell, W. Schlüter. *E|Melt: A flexible material flow and energy simulation in the context of Industry 4.0*. ASIM 2018 - 24. Symposium Simulationstechnik. Hamburg, 04.10.2018.

Approach for synthesis and optimization of complex thermal systems for supermarkets

Jonathan Kistner^{1*}, Wilhelm Tegethoff¹, Nicolas Fidorra², Jürgen Köhler²

¹TLK-Thermo GmbH, Hans-Sommer-Straße 5, 38106 Braunschweig, Germany; *j.kistner@tlk-thermo.de

²Institut für Thermodynamik, Technische Universität Braunschweig, Hans-Sommer-Straße 5, 38106 Braunschweig, Germany

Abstract. The development of thermal systems for supermarkets is a challenging task. Both, heating and cooling demands at different temperature levels have to be satisfied under individual boundary conditions. In combination with a broad range of available technologies and components, a high number of possible system layouts exist. Thus, various types of refrigeration systems can be found in supermarkets: Central refrigeration systems with one or two stages and direct evaporation, central systems with a secondary loop or systems with (semi)-plug-in-cabinets. The system topology and operating strategy depend on climate conditions, building scale, customer's occupancy or evaluation criteria. In practice, established solutions based on experience are used. However, comparing all alternative concepts is difficult. Beside the consideration of investment costs, it is essential to evaluate the energy consumption. For the calculation of energy consumption, considering dynamic interactions between components is crucial.

To compare different system layouts under consideration of dynamic interactions, an optimal operating control has to be applied. Furthermore, the high number of possible topologies makes it necessary to reduce the complexity for the selection of components and their interconnections. Therefore, software based methods are needed to efficiently reduce complexity and evaluate system alternatives in a dynamic environment.

This paper presents a procedure that supports the user to find an optimal system topology under individual conditions. As an example, a secondary-loop refrigeration system with low and medium temperature cabinets is applied.

The user defines ambient conditions and requirements such as cooling load and temperature set-points. Additionally, a set of transient, non-linear models for available technical equipment is defined. The parametrized, ready-to-use models are managed

in a catalogue platform. In the catalogue, additional information is stored, like valid operational ranges, which is used during optimization. On this information basis, an algorithm deduces a reasonable refrigeration system layout. Intermediate result is a ready-to-simulate system. It contains only catalogue models that have physical reasonable interconnections. Subsequently, the system's fluid flow rate of each connection is optimized. The result of the optimization is used for evaluation of the system layout and further reduction of its topology.

The paper shows, that using simple input information, the complexity of the optimization problem can be extremely reduced. The suggested procedure is capable to deploy an optimal system topology under consideration of non-linear dependencies.

Introduction

Many engineering work is spent on developing better thermal energy systems. The effort that is being made touches all sectors of industry and science like cars with electrified drivetrains [14], busses [3][9] and supermarkets [1]. Especially for supermarkets, the energy saving potential is enormous. A broad range of different technologies is available. At the same time, many different thermal demands typically occur. In practice, established solutions based on experience are used. However, comparing all alternative concepts is difficult. The high number of possible topologies makes it necessary to reduce the complexity for the selection of components and their interconnections. Software based methods are needed to efficiently reduce complexity and evaluate system alternatives considering all relevant non-linearities.

This paper presents an approach to support the user finding an optimal system topology under individual conditions. As an example, a secondary-loop refrigeration system with low temperature (LT) and medium temperature (MT) cabinets is applied. The system is synthesized

on basis of ready parameterized catalogue models. A steady state parameter optimization of the synthesized system model is performed. Further system reductions are derived from the optimization result. The overall electrical energy consumption of system variants are statistically estimated over one year for different cities in Europe.

1 Supermarket refrigeration systems

Supermarket energy systems have big energy saving potentials. Energy savings related to the application of optimal topologies in combination with optimal operating strategies can be tentatively estimated to be in a magnitude of 20 % [2][13].

Furthermore, energy systems of supermarkets typically possess a high complexity caused by many different requirements: The salesroom has to be cooled, heated and dehumidified. Groceries have to be kept at different low temperature levels. The optimal system topology changes with climatic boundary condition, building scale, customers' frequency and evaluation criteria. Improvements of energy efficiency and reduction of emissions are tried to achieve by applying waste heat recovery, regenerative technologies or thermal storages [12][8].

Beside detailed system variants, some general system types for supermarket refrigeration exist: Widely used and under intensive research are central refrigeration systems. Those systems can be huge refrigeration cycles with several evaporators at different temperature levels satisfying both cooling and freezing demands. The cabinets directly contain the evaporators. The main disadvantage is the high complexity of controls and a high charge of refrigerant. In water-loop refrigeration systems every cabinet contains its own little refrigeration cycle. The water loop transfers the waste heat of the condensers to the ambient. For low temperature cabinets, often a secondary brine loop with an additional refrigeration cycle is used to cool the cabinet's condenser. In secondary-loop systems, the cabinets are directly cooled by brine. Different refrigeration cycles provide brine at needed temperature setpoints. Water-loop and secondary-loop systems are more and more focused in current researches. Amongst others, the reasons are small amounts of refrigerant and relatively easy to control and combine in different layouts [1]. A comparison, especially with central

refrigeration systems is of big interest in the current scientific and economic discussion [4].

The mentioned aspects encouraged the author to use a secondary-loop system as example for system synthesis and optimization. In fact, this is already a design specification. The example is chosen to be very simple to ease the evaluation of the represented procedure. This paper is a preliminary study for further methodical work to support scientists and planners to develop new system topologies for supermarkets.

2 Procedure for layout development

In this chapter, the applied procedure for layout development is represented. It contains three main topics: Firstly, set up of the information basis. Secondly, execution of a system synthesis based on the defined information. Thirdly, optimization of the synthesized system model.

Figure 1 shows the process of the layout development. In the first step, the user starts defining requirements and boundary conditions. At this point, the refrigeration loads and related temperature setpoints are defined. The boundary conditions can be connected to a specific location that is relevant for the system development. Generic boundary conditions are used in this paper. Furthermore, a set of components has to be chosen, that shall be available for the system synthesis. Most information related to the catalogued models can be assumed to be given as meta data. However, the user might have to change some data like temperature setpoints for refrigeration cycles or estimated valid inlet temperatures for cabinets. Once the system is synthesized, the user needs to find valid starting conditions for optimization.

In the present study, three main questions were of particular interest:

- How could a synthesis look like on basis of component functionalities without executing any simulations?
- How does a well-suited information basis look like to efficiently support the system synthesis and reduce complexity?
- How can a system layout efficiently be evaluated with less user interactions as possible?

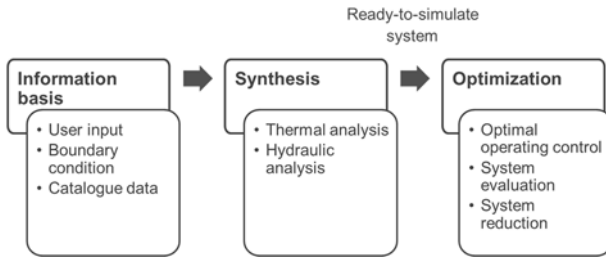


Figure 1: Employed procedure for system synthesis and optimization. The synthesis uses a well-suited information basis to find physical reasonable interconnections. The synthesized ready-to-simulate system model is optimized.

The used component models in this paper can be seen as type representatives. The procedure is not aimed to work for design tasks, which are typically characterized to have a big number of very similar components. Other procedures for system design exist, that could be applied consecutively or be integrated in the represented procedure in future work.

3 Information basis for system synthesis

As described in Chapter 2, the information basis contains requirements and specifications of the development task as well as a defined set of component models with related meta data. In this section, all basic information of the applied development task is specified and the available component models are described in detail.

3.1 Catalogue models

All available system components are modeled on basis of TIL [11]. Thus, the modelling language is MODELICA. TILMedia is used for media property calculation.

Figure 2 shows the component models. In the chosen scenario, the models for the cabinets are assumed to be part of the requirement defined by the user. The refrigeration units are inheritors from the same refrigeration cycle with modified parameters. The refrigeration cycle contains a physical based compressor model, two finite volume heat exchangers, an orifice valve and an ideal separator after the condenser. The whole cycle is scalable by one nominal cooling capacity. Additionally, it is capable to automatically switch on and off, depending on the boundary conditions at the condenser and evaporator.

The compressor is controlled to maintain a fixed temperature setpoint at the brine outlet of the evaporator. The valve controls the superheat of the refrigerant at the evaporator outlet. The two refrigeration units mainly differ in their type of refrigerant, their scale and their temperature setpoint.

The outdoor units basically are modeled as simple temperature boundaries. The second outdoor unit additionally is capable of regulating the outlet temperature by mixing the ambient temperature with the inlet flow. The cabinets are modeled as heat boundary. An early design decision is made by using cabinet models only for secondary-loop systems.

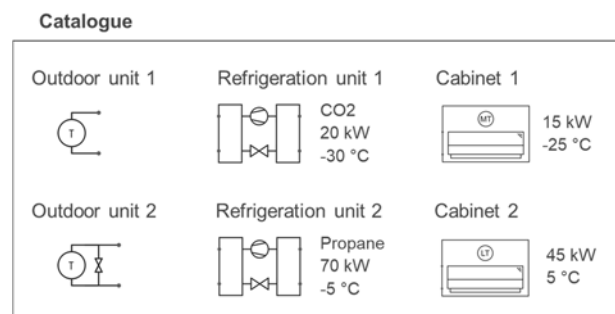


Figure 2: Catalogue models available for the system synthesis. The models for the cabinets are assumed to be part of the requirement. All other components are optional.

The catalogue models contain meta data that describe the functionalities and valid temperature ranges at all in- and outlets. Detailed examples for the meta data used for system synthesis can be seen in Chapter 4.

3.2 Requirements and specifications

The requirements and specifications are typically user inputs that characterize the overall optimization problem. As requirement, a refrigeration load with 45 kW at 5 °C and a load with 15 kW at -25 °C is specified. Furthermore, the user need to adjust the valid temperature ranges at the cabinets' inlet to complete the information basis for system synthesis. The temperature of the ambient air in a range of -10 °C to 40 °C is defined as ambient condition. The supermarket building is not considered in this study.

4 Synthesis of a secondary-loop refrigeration system

The system synthesis represented in this paper uses the

described information basis in Chapter 3 to deduce a physical reasonable system layout. In the first place, a thermal functional analysis for each inlet and outlet of the available components is executed. Catalogued temperature information and thermal functionalities of each component are used to estimate reasonable interconnections. Secondly, a hydraulic analysis integrates additional components to complete the hydraulic network and make the model executable.

4.1 Thermal analysis to find reasonable interconnections

Figure 3 shows a simplified view on the meta data that is used for thermal system synthesis. Original catalogue data, defined setpoints and deduced requirements are combined as information basis for the system synthesis.

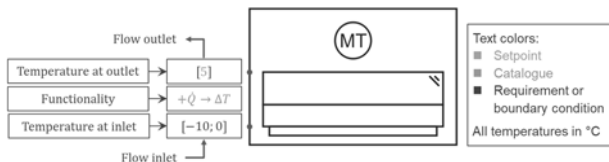


Figure 3: Simplified view of combined meta data as information basis for system synthesis.

The algorithm starts at one port and runs through all other ports of the available components in order to create valid hydraulic loops. When all options are proofed for this starting point, the algorithm restarts from another port until all possible loops are found. Note, that every component of the catalogue can be used only once in the system.

A set of rules reduce complexity and defines valid interconnections. Most important rules are described as follows:

- No interconnections between ports of the same component are allowed
- Only closed loops are allowed: The search of valid ports has to end at the same component (e.g. from evaporator outlet to inlet)
- Every loop must contain complementary functionalities (e.g. at least one positive and one negative heat flow)
- Calculated inlet temperatures have to comply to the valid range of the component port
- A port is only once connected in a loop
- No waste of generated refrigeration (e.g. evaporator to warmer outdoor unit)
- A component is allowed to have interconnections to

only one of the outdoor units (best match of temperature range is chosen)

Figure 4 shows the synthesis of one loop (see button 2) with the starting point at the outlet of the MT cabinet. This loop is valid for ambient temperatures between 0 °C and 20 °C. The direct return flow from the condenser of the LT refrigeration cycle to the MT cabinet is not a valid loop (see button 1 and dashed line). In this case, the temperature range is violated and the functionalities are not complementary. In the end, the whole system layout is built by overlaying all found valid loops. Every interconnection exists only once in the system.

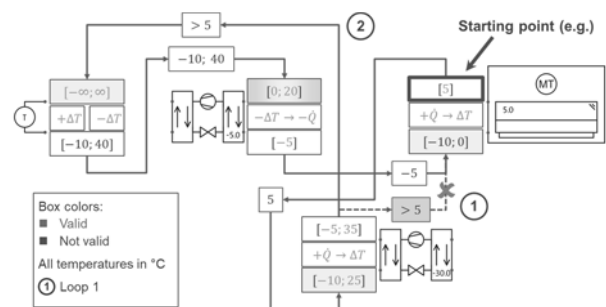


Figure 4: Example for system synthesis algorithm.

Starting at the outlet of the MT cabinet, a valid flow connection to the condenser of the LT refrigeration cycle is found. From there other valid connections are found until the loop is closed. Estimated temperatures of the stream are displayed along the blue lines.

4.2 Integration of hydraulic components

The hydraulic concept is to use pumps for every possible interconnection and control their indicated mass flow rate with the optimizer. Thus, no valves, no controllers and no junctions are needed in the hydraulic network. From port to port, pumps are added until every mass flow rate in the hydraulic system is determined. The complete synthesized system layout is illustrated in Appendix A.

5 Optimization of the synthesized system

During the applied system synthesis, no simulation is used. Therefore, the synthesized system model still has to be evaluated and maybe even further improved by re-

sults of simulation. To be able to evaluate the system layout correctly, it must operate in an optimal way while facing the defined boundary conditions. Optimal operating is defined to have minimum power consumption while observing the requirements. For finding valid starting conditions a parameter study for the control inputs of the pumps was applied. Integrations were executed in Dymola with DASSL as solver.

5.1 Optimization problem and algorithm

The applied optimizer is based on the Globalized Bounded Nelder-Mead algorithm (GBNM) [6]. It is capable of constraint handling via penalty function and probabilistic restarts for a globalized optimization. Using a global optimization is crucial for the represented optimization task. Finding good starting conditions can be very time extensive and many local minima are expected for this type of systems. Another big advantage is, even of the basic Nelder-Mead-Algorithm [7], that it allows a non-gradient evaluation of the model. The used models, especially for the refrigeration cycles, show high non-linear behaviour. Thus, sensible gradient evaluation might restrain the optimization progress or even fail at some operating points. Two other aspects appeared to be important as well: Firstly, with gradient evaluation, a precise normalization of the cost function has a strong impact on the optimization progress. A good normalization equals a good estimation of the cost function value at optimum. This is problematic in the context of topology optimization where a focus on widely automated procedures is crucial and starting conditions can be far away from the optimum. Secondly, unphysical parameter value combinations cannot be avoided. They lead to simulation failures, which impede cost function evaluation. Instead, an artificial penalty cost function value for failed simulations has to be used which can corrupt the gradient sensitivity. Comparison studies with the SLSQP from SciPy [10] based on [5] confirmed that issues.

Since the system model does not contain energy storage components, it is assumed that there is no time constant that is relevant for the power consumption. Thus, a dynamic optimal control problem can be avoided. Instead, a parameter optimization at steady state is applied with a batch of discrete ambient temperatures.

Cost function and constraints. The main ob-

ject of the optimization is to minimize the sum of all electrical power consumption in the system. Additionally, deviations to setpoint temperatures ($T_{set} - T$) are added to the cost function via penalty function. Furthermore, mass flow rates that are not directly driven by a specific pump shall be only positive for better understanding. Those dependent mass flow rates (\dot{m}_{dep}) are considered as constraints, too. Figure 5 shows how the system model evaluation during optimization is done.

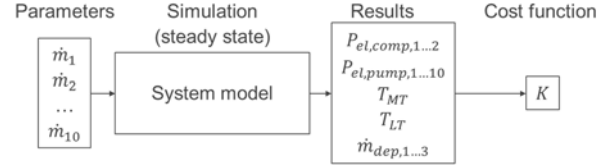


Figure 5: System evaluation during optimization.

Optimization parameters are the mass flow rate inputs (\vec{m}) for all pumps in the hydraulic system. Calculated variables for electrical power consumption (P_{el}), temperatures (T) and dependent mass flow rates (\dot{m}_{dep}) are part of the cost function.

Weighting and normalization are part of the penalty function. The optimization problem and the cost function is defined as follows:

Minimize $K(\vec{m})$, where

$$K(\vec{m}) = \sum_{n=1}^{10} P_{el,pump,n} + \sum_{n=1}^2 P_{el,comp,n} + f_c(T_{MT,set} - T_{MT}) + f_c(T_{LT,set} - T_{LT}) + \sum_{n=1}^3 f_c(\dot{m}_{dep,n}) \quad (1)$$

With

- \vec{m} – Vector with $\dot{m}_1, \dots, \dot{m}_{10}$ [kg/s]
- P_{el} – Electrical Power [W]
- f_c – Normalized constraint penalty function [W]
- T – Temperature [K]
- \dot{m}_{dep} – Dependent mass flow rate [kg/s]

5.2 Optimization results

All parallel global optimizations were limited to a maximum duration of about 40 hours, while 2000 iterations for one local search was set as maximum. During one global optimization (at constant ambient temperature) up to 30 local searches took place. In all global runs, several local searches ended up in the same optimum. Further-

more, no better set of parameters could be found by parameter studies. Thus, the optimization results seem to provide valid global minima. Integrations were numerically robust and fast: Failures only occurred due to unphysical parameter values. Speed of integration for one evaluation was about 1400 times real time.

Note that the presented results in this paper are manually simplified for better understanding: Very small mass flow rates are set to zero and some results are adjusted to show the corresponding operating mode clearer.

Figure 6 shows that under the chosen algorithm options, the optimizer is capable to change operating mode even under one local search. Under less precise truncation criteria, this local search would have stopped at about 800 evaluations and a new local run would have been initiated. With the used truncation criteria, the optimizer was capable to find a new and better minimum within the same local run. Comparing the parameter values at 800 with those at the end of the run, the operating mode changed: In the beginning, the MT cabinet and the condenser of the LT refrigeration cycle is cooled in parallel from the evaporator of the MT refrigeration cycle. At the end of the run, the working fluid that flows to the LT refrigeration cycle is completely passed on to the MT cabinet. Only less than 5 kW of the 45 kW cooling load at the MT cabinet is served directly from the MT refrigeration cycle. In Figure 7, the system simulation with optimized operating control can be seen.

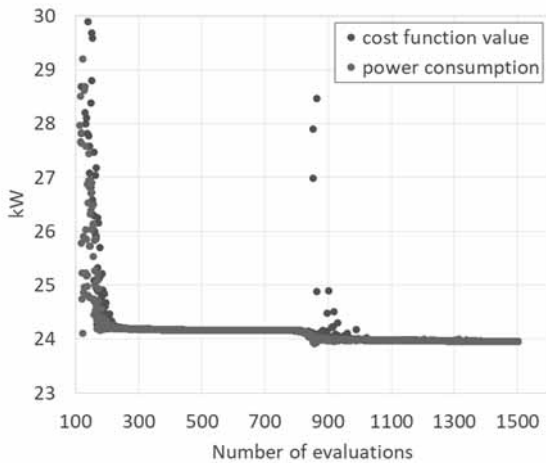


Figure 6: Detail of one local search at 20 °C ambient temperature. After elimination of the constraint penalties, the cost function value equals the power consumption. After 800 evaluations, a set of parameters is found that represents a better operating strategy.

This operating mode is found as global optimum for all ambient temperatures above 5 °C, but of course with slightly different mass flow rates.

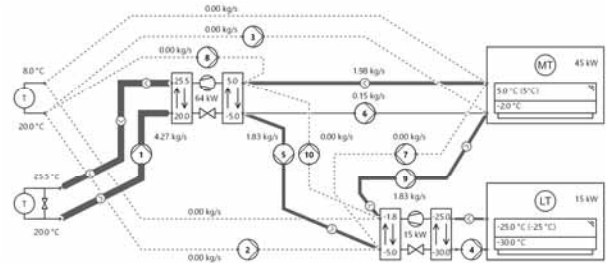


Figure 7: Optimized mass flow rates at 20 °C ambient temperature. The mass flow rate that flows from the evaporator of the MT refrigeration cycle to the condenser of the LT refrigeration cycle is completely passed on to the MT cabinet without direct return.

Below 0 °C ambient temperature, the optimal operating mode is characterized by cooling the condenser of the LT refrigeration cycle and the MT cabinet by the outdoor unit. About 55 % of the 45 kW cooling load at the MT cabinet is served by mass flow rate passed on from the condenser of the LT refrigeration cycle. Figure 8 shows the optimal mass flow rates at -5 °C ambient temperature. At lower ambient temperatures (about -7 °C), even 100% feedthrough is preferred: All cooling load at the MT cabinet is driven by the mass flow rate through the LT refrigeration cycle's condenser and no direct cooling from the outdoor unit to the MT cabinet is used.

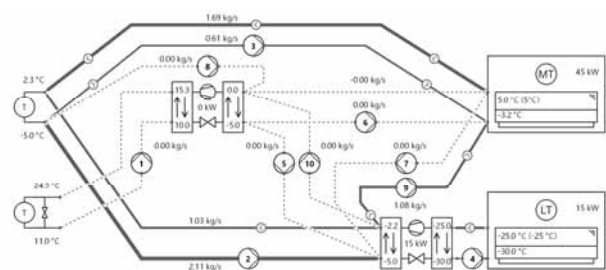


Figure 8: Optimized mass flow rates at -5 °C ambient temperature. MT refrigeration cycle is switched off. The LT refrigeration cycle and the MT cabinet are cooled via outdoor unit. A big amount of working fluid at the condenser outlet is passed on to the MT cabinet.

At ambient temperatures around 5 °C the outdoor unit covers the needed cooling at the condenser of the LT refrigeration cycle with about 60 %. The rest of about 9 kW is served by mass flow rate from the MT cabinet outlet. This additional mass flow rate through the con-

denser is passed on to the outdoor unit with a few degrees above ambient temperature, cooled down and from there passed on to the inlet of the MT refrigeration cycle. From there it flows again to the MT cabinet. There is a small range of ambient temperature in which this operating mode makes sense. Additionally, the benefit for energy consumption is very small compared to the second best result (see Figure 9 in Chapter 6).

6 Derivation of system layout reduction

In this section the capability of the presented procedure to allow topology improvements is proofed. A system topology optimization typically includes a variation of number and type of components as well as their interconnections. The main purpose of this paper is not to optimize the system topology. However, it is a preliminary study to approach topology optimization in the future. The main idea is to eliminate components and interconnections by analysing the optimized mass flow rates.

Interconnections that show zero mass flow rates at all boundary conditions could be easily eliminated. Since the optimization contains the power consumption of the pumps, in some cases the results of the present study show a splitting of mass flow rate over several interconnections in order to avoid a very high mass flow rate at only one of the interconnections. For instance, providing cooling for the MT cabinet at low ambient temperatures is not only applied by direct cooling from the outdoor unit. Instead, additional mass flows from the outdoor unit to the MT cabinet through the evaporator of the switched off MT refrigeration cycle. In these cases, it is difficult to evaluate the necessity of particular interconnections. Therefore, the known loops from the system synthesis are used to detect corresponding operating modes and hence allow the derivation of second best operating strategies that are referenced to the optimal solution.

Three variations of the synthesized system layout are analysed. Each system layout is quantified by the sum of the power consumptions weighted according to the incidence of relevant ambient temperatures over one year for Berlin, Munich, Barcelona, Madrid and Oslo. Since no component of the simplified secondary-loop refrigeration system could be eliminated, an economical evaluation is not required.

In the first place, the optimization results show never a direct return from LT refrigeration cycle (condenser) to MT refrigeration cycle (evaporator). Instead, all mass is

returned via MT cabinet. Thus, this interconnection could be simply eliminated (see Figure 9, pump 10). Secondly, looping mass from the MT cabinet back to MT refrigeration cycle (evaporator) via the condenser of the LT refrigeration cycle and outdoor unit is only used around 5 °C ambient temperature. Second-best option is a higher mass flow rate from the outdoor unit to the condenser of the LT refrigeration cycle. Figure 9 shows the original optimization results at 5 °C. The elimination of the red interconnections only leads to a higher electrical energy consumption over year of about 0.1 % for all considered cities.

The third option that has been analysed is the elimination of all two connections between the LT refrigeration cycle and the MT cabinet. This affects the operating mode for ambient temperatures below 0 °C and above 10 °C. In Appendix B, the corresponding layout reduction can be seen. Compared to the optimization results, the estimated energy consumption over year is slightly higher: Between 1.3 % for Oslo and 1.7 % for Barcelona.

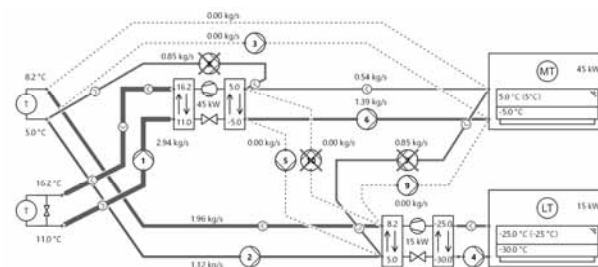


Figure 9: Optimized mass flow rates at 5 °C ambient temperature. The LT refrigeration cycle could be cooled only via outdoor unit. Thus, the red interconnections could be eliminated with almost the same electrical energy consumption. The path with pump 10 could be eliminated anyway. The reduced system layout is valid for all considered ambient temperatures.

7 Conclusion

The represented approach for system synthesis and optimization reduced the number of interconnections from potentially 64 to 10. The synthesized secondary-loop refrigeration system contains all physical reasonable interconnections. The system model covers all relevant nonlinearities. At the same time, integration was fast and numerically robust. The applied GBNM algorithm appeared

to be robust in finding global minima within an acceptable duration of 1 day.

The optimization results allowed the derivation of further system layout reductions. The catalogued thermal functionalities of the components and the found hydraulic loops during system synthesis were important to interpret optimization results and to detect corresponding operating strategies.

Acknowledgements

This work has been supported by the German Ministry BMBF in the VEOTOP project (FKZ: 01LY1809A). We are much obliged to the BMBF and the project leader DLR.



References

- [1] Fidorra N, Kistner J, Tegethoff W, Köhler J. Energetische Bewertung von Wasserkreislauf-Systemen für Supermärkte. In DKV-Tagungsberichte. *Deutsche Kälte- und Klimatagung*; 2017; Bremen.
- [2] Fidorra N, Köhler J. Energetische Untersuchung integrierter Supermarktkonzepte. In DKV-Tagungsberichte. *Deutsche Kälte- und Klimatagung*; 2016; Kassel.
- [3] Kaiser C, Schröder A, Raabe G. Entwicklung eines CO₂-Ejektorkreislaufs für eine umschaltbare Wärmepumpen-Klimaanlage für Omnibusklimaanlagen. Deutsche Bundesstiftung Umwelt - Forschungsbericht (AZ: 30270). 2015.
- [4] Karampour M, Sawalha S. Comparison of State-of-the-art CO₂ and Alternative Refrigeration Systems for Supermarkets. *Gustav Lorentzen Conference*; 2018; Valencia.
- [5] Kraft, D. A software package for sequential quadratic programming. Forschungsbericht DFVLR. 1988; DFVLR-FB 88-28.
- [6] Luersen MA, Le Riche R, Guyon F. A constrained, globalized, and bounded Nelder–Mead method for engineering optimization. *Struct Multidisc Optim*. Springer-Verlag; 2003.
- [7] Nelder JA, Mead R. A simplex for function minimization; *The Computer Journal*. 1965, Volume 7 (4): 308–313.
- [8] Nöding M, Fidorra N, Gräber M, Köhler J. Operation

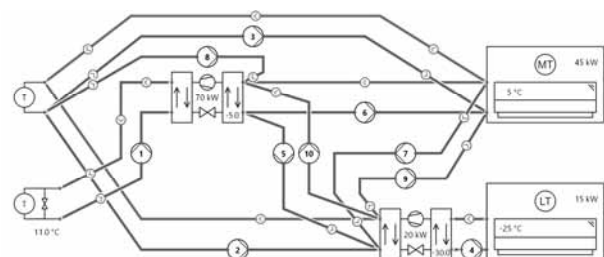
Strategy for Heat Recovery of Transcritical CO₂ Refrigeration Systems with Heat Storages. *29th International Conference on ECOS*; 2016; Portoroz.

- [9] Peteranderl C, Bernath M, Tegethoff W, Köhler J. City Bus with Modular R744 HVAC System Based on Passenger Car Components, *Thermal Management Systems Symposium*; 2018; San Diego.
- [10] SciPy, <https://docs.scipy.org/doc/scipy/reference/tutorial/optimize.html> (visited 19.12.2018)
- [11] TLK Thermo GmbH, TIL-Suite, www.tlk-thermo.de <http://www.tlk-thermo.com/index.php/de/softwareprodukte/til-suite> (visited 19.12.2018)
- [12] Titze M. Dynamische Simulationen zur Wärmerückgewinnung im Supermarkt. *Supermarkt-Symposium ZVKKW*; 2014; Darmstadt.
- [13] Titze M. Energetic Analysis and Optimisation Strategies of a modern north European Supermarket [dissertation]. TU Braunschweig; 2017.
- [14] Weustenfeld T. Heiz- und Kühlkonzept für ein batterieelektrisches Fahrzeug basierend auf Sekundärkreisläufen [dissertation]. Fakultät für Maschinenbau. TU Braunschweig; 2017.

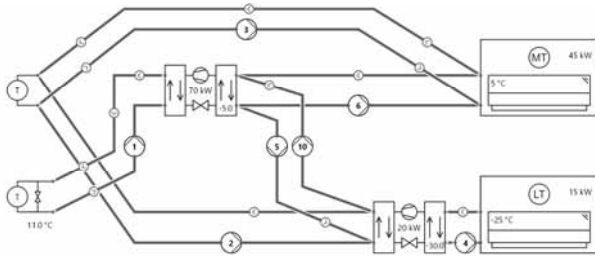
Appendix

Appendix A. The figure below shows the synthesized system model. The red interconnections have no pumps. Their mass flow rate are determined by the other mass flow rates in the hydraulic system. Temperature setpoints and refrigeration capacities are displayed as well.

The LT cabinet can only be cooled by the LT refrigeration cycle. The condenser of the MT refrigeration cycle is only be cooled by outdoor unit 2 which can hold a minimum required temperature for condensation. The MT cabinet can be cooled by MT refrigeration cycle and outdoor unit 1. The LT refrigeration cycle (condenser) can be cooled by outdoor unit 1 and MT refrigeration cycle (evaporator). Additionally, several interconnections exist for looping mass between those components at MT temperature level.



Appendix B. The figure below shows the system model of the third identified option for system reduction. Interconnections between the LT refrigeration cycle and the MT cabinet are removed. Additionally, one related interconnection is removed as well. Remaining second-best operating modes are only simple feed and return flows between the components of MT level. Therefore, the path of pump 10 has to remain in the system layout.



A Python Framework for Model Specification and Automatic Model Generation for Multiple Simulators

Hendrik Folkerts^{1*}, Thorsten Pawletta¹, Christina Deatcu¹, Sven Hartmann²

¹Research Group Computational Engineering and Automation, Hochschule Wismar, Philipp-Müller-Straße 14, 23966 Wismar, Germany; *hendrik.folkerts@cea-wismar.de

²Clausthal University of Technology, Adolph-Roemer-Straße 2A, 38678 Clausthal-Zellerfeld, Germany

Abstract. The System Entity Structure (SES) is a high level approach for variability modeling, particularly in simulation engineering. An SES describes a set of system configurations, i.e. different system structures and parameter settings of system components. In combination with a Model Base (MB), executable models can be generated from an SES. Based on an extended SES/MB approach, an enhanced software framework is introduced that supports variability modeling and automatic model generation for different simulation environments. The main focus of this paper is the presentation of open source software tools for modeling an SES and for the automated generation and execution of models derived from an SES and the belonging MB by means of an engineering application.

Introduction

In Software engineering, a classical approach on variability modeling is the use of Feature Models (FM) in combination with 150% models. Feature Models specify components and relations, which are used for modeling the variability of a system or product [1]. By selecting features a Variant Model (VM) can be generated from the FM. Thus, the VM represents a subset of all variants defined in the FM. The FM and the VM are platform-independent. For generating executable models, platform dependent dynamic models are needed, which are linked to the FM. All dynamic models are organized as an 150% model. With the help of a variant generator an executable model is generated based on an VM and the 150% model. The software pure::variants is a prominent example supporting this approach [2]. Another approach is the executable Unified Modeling

Language (xUML). The xUML is a subset of the graphical notation of the UML with executable semantics in order to generate platform dependent software [3].

Analogous to approaches coming from software engineering the systems theory community introduced methods for platform-independent variability modeling with subsequent platform-dependent model generation of specific variants. One method is the System Entity Structure (SES) and Model Base (MB) approach. In this paper the theory of an extended SES/MB approach is discussed and software tools implementing the theory are presented using an engineering application example.

1 SES/MB Theory and Implementation

This section briefly discusses the general SES/MB theory and the derived extended SES/MB (eSES/MB) infrastructure. Subsequently, an implementation of the infrastructure is presented.

1.1 SES/MB Basics and the eSES/MB Infrastructure

An SES is represented by a tree structure comprising entity nodes, descriptive nodes and attributes. Different system structures can be coded in an SES tree. In context of simulation technology *entity nodes* are linked with *basic models* organized in an MB. Attributes of an entity node correspond to the parameters of the associated basic model. *Descriptive nodes* describe the relations among at least two entities and are divided into *aspect*, *multi-aspect* and *specialization* nodes. An aspect node represents the composition of an entity. The multi-aspect node is a special aspect node and describes the decomposition of an entity in several entities of the

same type. Aspects and multi-aspects express a has-a relation. The specialization node describes the taxonomy of an entity and expresses an is-a relation. Descriptive nodes specify characteristic variation points using specific rules.

Specifying a composition of entities, which describes in the context of modeling and simulation the composition of models in a modular-hierarchical manner, requires a specification of *coupling* relations. Couplings can be specified as properties of descriptive nodes of the type aspect or multi-aspect and consist of pairs of entity names and port names, such as (*source entity*, *source port*, *sink entity*, *sink port*).

In order to derive a specific system configuration the SES needs to be pruned. All variation points are resolved by evaluating the rules at the descriptive nodes. More than one aspect or multi-aspect on the same tree level, called siblings, form a variation point with a xor-selection. A second kind of variation point is defined by specialization nodes. Besides, when selecting one child entity of the specialization, the name and the properties of the selected child entity are merged with the name and properties of the father entity of the specialization according to the SES inheritance axiom. The resulting *Pruned Entity Structure (PES)* represents exactly one system configuration. In conjunction with an MB, a fully configured and executable model can be generated from the PES.

The basic SES/MB framework introduced in [4] was extended in [5] and [6] by new modeling features, methods and components, such as an *Experiment Control (EC)* and an *Execution Unit (EU)* as shown in Figure 1. In this eSES/MB infrastructure, the EC uses an interface to the SES and its methods to derive goal-driven system configurations and to generate models, which are executed by the EU. The results returned by the EU are collected and analyzed by the EC. Thus, the derivation and generation of subsequent system configurations can be controlled reactively based on experiments already carried out.

The interface to the SES is established by variables with a global scope, called *SES variables (SESvar)*. *Semantic conditions* can be used to specify permitted value ranges and dependencies between SESvars. *SES functions (SESfcn)* are introduced for the specification of procedural knowledge. Complex variability can often be described more easily with SESfcns. Typical examples include the definition of varying coupling relations or the definition of variable parameter configurations in attributes.

For automatic pruning, *selection rules* at descriptive nodes can be defined, such as *aspectrules* for aspect and multi-aspect siblings or *specrules* at specialization nodes. A special attribute of multi-aspects is the *number of replications (numRep)*. The numRep attribute specifies the number of entities to create at a multi-aspect node during pruning. The *mb-attribute* of leaf entity nodes specifies the linkage of the entity node to a basic model in the MB. Attribute values and selection rules can be specified using SESvars or SESfcns.

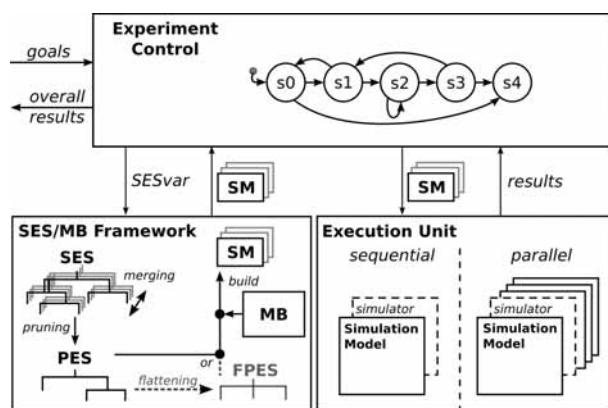


Figure 1: The eSES/MB infrastructure.

1.2 Software Tools

The infrastructure in Figure 1 is implemented as a Python framework as presented in Figure 2. The tools are called *SESToPy*, *SESMoPy*, and *SESEuPy*.

SESToPy implements a graphical editor and all SES related methods. In the editor an SES tree can be specified interactively in a file browser view and for every node specific attributes or rules can be defined. In addition to the pruning method already mentioned, SESToPy supports some more methods such as *merging* different SES and *flattening* for removing the hierarchy information. Applying the flattening method, a *Flattened Pruned Entity Structure (FPES)* is derived. The pruning and flattening methods can be used interactively or as API methods. SESToPy supports to save SES, PES, and FPES as JSON or XML files.

For generating executable models, SESMoPy was developed. SESMoPy is a model builder, which supports *build* methods for several simulation environments. Currently native models for *MATLAB/Simulink/Simscape/Simevents*, *Dymola*, *Open-*

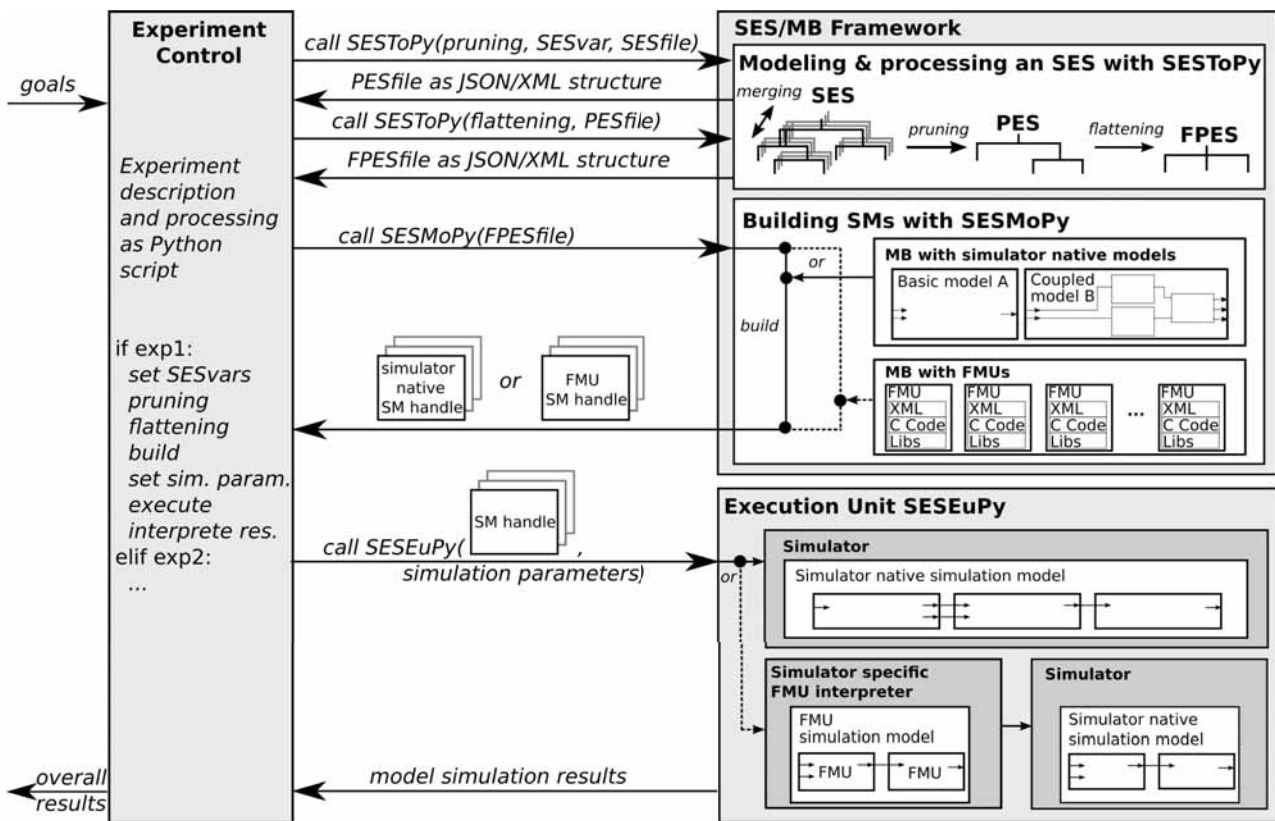


Figure 2: Python-based eSES/MB infrastructure for multiple EUs.

Modelica, the *PDEVs* for *MATLAB* toolbox and the Python-based environment *DEVSimPy* can be generated. In addition, *SESMoPy* supports model generation based on the Functional Mock-up Interface (FMI) definition. The FMI standard is defined for several uses, one of which is FMI for Model Exchange. The generalized interface FMI is supported by a number of established simulators [7]. Using this approach, *SESMoPy* builds a Functional Mock-up Unit (FMU) model that implements an FMI [8]. *SESMoPy*'s build methods generate executable simulation models (SM) using the information in the FPES and basic models from a target specific MB. Information about the way the model is created can be provided in the EC calling *SESMoPy* or at the SES level according to the SES enhancements in [9].

The Python software tool *SESEuPy* acts as a general EU. It implements a kind of wrapper for the integration of different simulation environments into the framework. Due to the modular structure and well-defined interfaces, components for generating component-based software for non-simulation-specific environments can

also be integrated into the framework.

In the next section, the components and functionality of the Python framework are explained using the example of an engineering application.

2 Engineering Application

A feedback control system can be modeled using transfer functions describing the behavior of the components in frequency domain. Controlled variables in a feedback control system are usually influenced by disturbances. A common approach for minimizing the influence of predictable disturbances is adding a feedforward control. How such a system can be designed and tested using the eSES/MB infrastructure and the introduced tools is described in the following paragraphs.

2.1 Problem Description

A process unit with a *PTI* behavior shall be controlled using a PID controller. A disturbance with a *PTI* behavior affects the output of the process unit. Different

configurations of the PID controller shall be tested. If a defined regulatory goal is met, the current configuration of the PID controller is taken. Otherwise the structure is varied by adding a feedforward control to the system and different configurations of the PID controller are analyzed. Figure 3 depicts a schematic representation of the application.

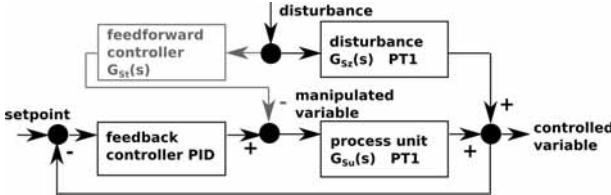


Figure 3: Structure of the feedback control system with alternative feedforward control.

The system's behavior follows the *PT1* transfer function in equation 1 and the step-shaped disturbance affects the output of the process unit with a *PT1* behavior according to equation 2. The optional feedforward control is realized by subtracting the disturbing signal calculated by equation 3 from the manipulated variable. The control goals are a settling time of 15 seconds and a maximum overshoot of 5% after a disturbance.

The system has two structure variants, either with or without the feedforward control part, and a range of different configurations for the PID controller can be applied. In the next section, the two structure variants and their possible configurations are specified as an SES.

$$G_{Su}(s) = \frac{1}{20 \cdot s + 1} \quad (1)$$

$$G_{Sz}(s) = \frac{1}{10 \cdot s + 1} \quad (2)$$

$$G_{Sf}(s) = \frac{G_{Sz}(s)}{G_{Su}(s)} = \frac{20 \cdot s + 1}{10 \cdot s + 1} \quad (3)$$

2.2 Variant Modeling with SEStoPy

The specification of the SES is done with the tool SEStoPy. The tree and all attributes are defined via a graphical user interface. During modeling the SES with SEStoPy, checks on the SES and plausibility tests are executed indicating model errors. The SES is saved as a JSON structure.

Figure 4 depicts the SES and its representation in SEStoPy. The SES uses some extensions introduced in [9]. Besides the different system configurations, es-

sential parts for the configuration of simulation experiments are defined. The root node *exp* of the SES and its subsequent aspect *expDEC* describe a set of simulation based parameter studies for different system structures. The subtree of the entity node *simModel-ctrlSys* specifies the two system structures, i.e. a variant with and a variant without feedforward controller. The other two entity nodes specify experiment related information. The entity node *simMethod* specifies a target simulation environment for performing simulation runs using the SESvar *mysim*. Other simulation execution parameters, such as the simulation period, are not specified and are set by the EC later. The entity node *expMethod* specifies the permitted value ranges of two parameters for the PID controller. Besides the different system structures, they are the subject under study.

The aspect *simModel-ctrlSysDEC* describes that each system variant consists of the entities: *sourceSys*, *feedbackSys*, *ctrlPIDSys*, *procUnitSys*, *sourceDist*, *tfDist* and *addDist*. They are mandatory system elements. The optional feedforward control is specified by the subtree of entity *feedforwardCtrl*. The coupling relations of both structure variants are defined in the attribute *cplg1* of aspect *simModel-ctrlSysDEC*, as discussed later.

According to [10], optional parts in an SES are expressed by a specialization node where one of its children is a NONE element. A NONE element means that the entity is not included at all. The selection at a specialization is defined by an attribute called *specrule*. The specrule of the specialization *feedforwardCtrlSpec* defines that either the entity *fc* or *NONE* is selected during pruning. The result of evaluating the specrule at node *feedforwardCtrlSPEC* depends on the value of the SESvar *feedforward*. The SESvar codes the two possible structure variants as values 1 or 0. Therefore, the semantic condition $feedforward \in [0, 1]$ applies to the SESvar. The entity *fc* and its subsequent aspect *fcDEC* specifies the feedforward control structure as a composition of the two entities *tfFeedforward* and *addFeedforward*.

Aspects and multi-aspects can define coupling relations as attribute. These attributes are abbreviated with *cplg* in Figure 4. Due to the varying system structures specified in the SES, the couplings in attribute *cplg1* of aspect node *simModel-ctrlSysDEC* are defined using an SESfcn. Listing 1 shows an excerpt of the SESfcn. The coupling definitions in *cplg2* at node *fcDEC* are invariable and can therefore be defined without using an

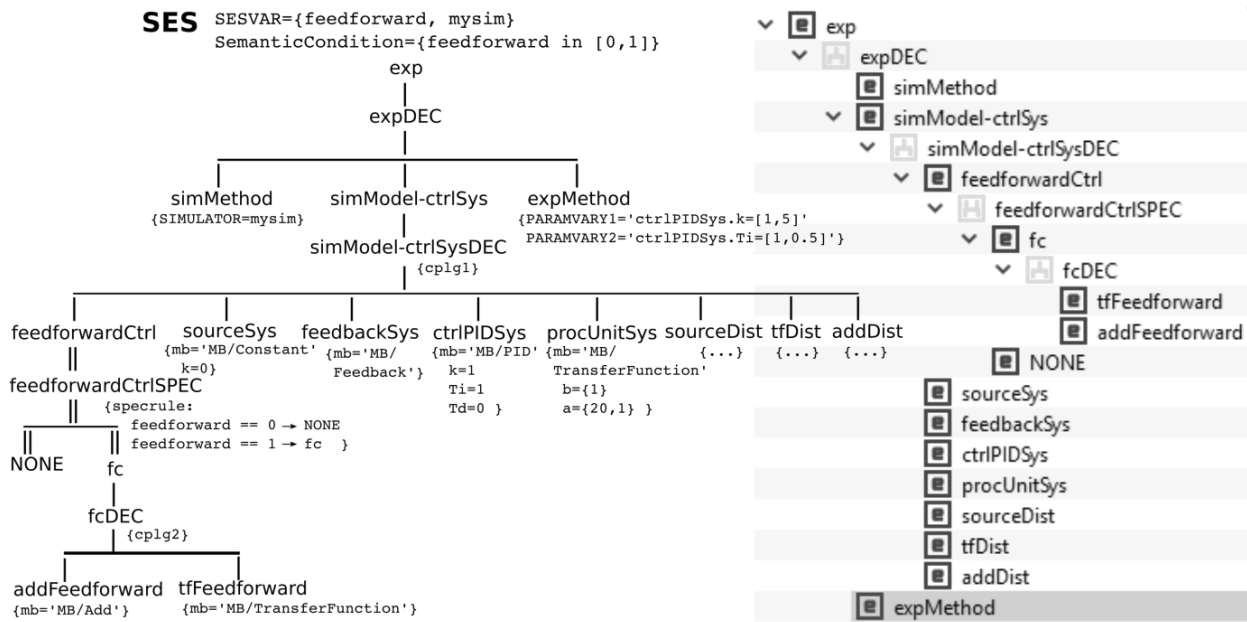


Figure 4: Left: SES specifying the feedback control system study; Right: Part of the SES representation in SESToPy.

SESfcn.

According to Section 1, each leaf node defines an mb-attribute referring to a basic model in the MB. The other attributes of the leaf nodes define properties to configure the linked basic models. The values specified at node *ctrlPIDSys* are only default values, which will be overwritten because they are parameters under study.

```

1 def cplgfcn(feedforward):
2     cplg = []
3
4     #fixed couplings
5     cplg.append(["sourceSys", "y", "
6         feedbackSys", "u1", ""])
7     cplg.append(...)
8
9     #variable couplings
10    if feedforward==0:
11        cplg.append(...)
12    elif feedforward==1:
13        cplg.append(...)
14
15    #return
16    return cplg

```

Listing 1: Excerpt of the Python code defining the SESfcn for *cplg1*.

2.3 Creating Basic Models with OpenModelica

OpenModelica defines a set of basic models for different fields of engineering. For being independent

of changes in the Modelica library, an OpenModelica *package* for the basic models used is created. The package is filled with the following basic models whose names correspond to the names in the mb-attributes of the leaf nodes in the SES:

- *Constant* as the setpoint for the controlled variable
- *Step* for stimulating the disturbance
- *Feedback* for closing the feedback control loop
- *PID* is the controller of the feedback control system
- *TransferFunction* for representing the process, the disturbance's behavior, and the feedforward
- *Add* for adding signals

Each basic model can be configured according to the attributes of the leaf node which they are linked to in the SES. The package acts as an MB for OpenModelica basic models.

2.4 Experiment Execution

For executing simulation based experiments the experiment process and its goals need to be defined in a Python script. This script implements the EC according to Figure 2. The Python framework provides some EC related template scripts. The goals of the experiment were discussed in Section 2.1. The experiment should start with the study of different PID con-

troller configurations using the control system structure without feedforward controller. In case that the objectives are not achieved by just varying the parameters k and T_i of the PID controller, the study shall be carried out with the additional feedforward control structure. Listing 2 shows a snippet of the EC script with essential steps of the experiment process.

```

1 ...
2 SESfile = ...
3 if conditions_for_experiment:
4     #prune, flatten, and build
5     SESvar = {"mysim": "OpenModelica",
6              "feedforward": 0}
7     PESfile = SESToPy("prune", SESvar,
8                      SESfile)
9     FPESfile = SESToPy("flatten", PESfile)
10    h_model = SESMoPy(FPESfile)
11    #execute
12    sim_param = ...
13    results = SESEuPy(h_model, sim_param)
14 ...
15 elif conditions_for_experiment:
16     #prune, flatten, and build
17     SESvar = {"mysim": "OpenModelica",
18              "feedforward": 1}
19     PESfile = SESToPy("prune", SESvar,
20                      SESfile)
21     FPESfile = SESToPy("flatten", PESfile)
22     h_model = SESMoPy(FPESfile)
23     #execute
24     sim_param = ...
25     results = SESEuPy(h_model, sim_param)
26 ...

```

Listing 2: Snippet of a Python-Script implementing an EC for the control system study.

According to Listing 2, the EC starts the experiment by setting the SESvar *mysim* and *feedforward*. Next, the EC calls SESToPy's API method for pruning with the current SESvar values and a reference to the file defining the SES as JSON structure. The pruning process results in a PES coded as JSON structure. Afterwards, the EC calls SESToPy's API method for flattening the PES. The created FPES is similar to the FPES shown in Figure 5, which represents the more complex FPES for the later SESvar assignment *feedforward* = 1. A reference to the file containing the FPES as a JSON structure is returned to the EC. The EC then calls SESMoPy's API method for the build method and passes the FPES file handle. SESMoPy determines the target simulator from the attribute at the node *simMethod* and the value ranges of the PID controller parameters under study from the attribute at node *expMethod* in the FPES. Based on the information in the FPES and the basic models from the MB, SESMoPy creates a file: (i) representing the model of control system that is exe-

cutable by the target simulator OpenModelica and (ii) containing the set of different parameter settings for the different PID controller configurations. The file is returned to the EC. The EC generates a further data structure *sim_param* with simulation execution related data, such as the solver to use, etc. Then, the EC calls the tool SESEuPy and passes the model file handle and the *sim_param* data structure. In collaboration with the target simulation environment OpenModelica, SESEuPy creates the fully configured simulation model and controls its execution. Figure 6 shows the structure of a fully configured OpenModelica model, but with feedforward controller, i.e. for the SESvar assignment *feedforward* = 1. Moreover, SESEuPy can create and execute parallel simulation runs. Finally, SESEuPy returns the simulation results to the EC.

In case the results meet the experiment goals, the overall results are calculated and returned by the EC. Otherwise a new model configuration and generation needs to be started. For this purpose, the *elif* part in Listing 2 defines the experiment steps for the second system structure with the additional feedforward controller by the SESvar assignment *feedforward* = 1.

The overall results of the experiment are the necessary control structure and the appropriate PID controller parameter settings, which fulfill the defined experiment goals. In the negative case, the failure to achieve the objectives may also be established.

3 Conclusion and Future Work

In this paper free software tools for variant modeling, beginning with the system specification with an SES up to automatic variant derivation, model building and execution are presented. The eSES/MB infrastructure implemented as the prototype Python tools is applicable to model and simulate engineering problems using different target simulation environments. Future works will especially cover a deeper integration of FMI in SESMoPy and test other target simulation environments.

References

- [1] Kang KC, Cohen SG, Hess JA, Novak WE, Peterson AS. Feature-Oriented Domain Analysis (FODA) Feasibility Study. *Tech. rep.*, Carnegie-Mellon University Software Engineering Institute. 1990.
- [2] pure-systems GmbH. *Technical White Paper: Variant Management with pure::variants*. 2006 (accessed

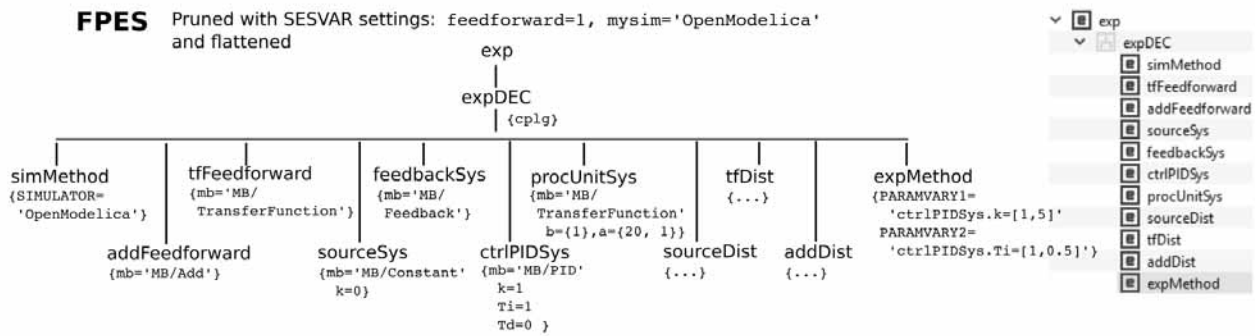


Figure 5: Left: FPES to study the feedback control system structure with feedforward; Right: FPES representation in SESToPy.

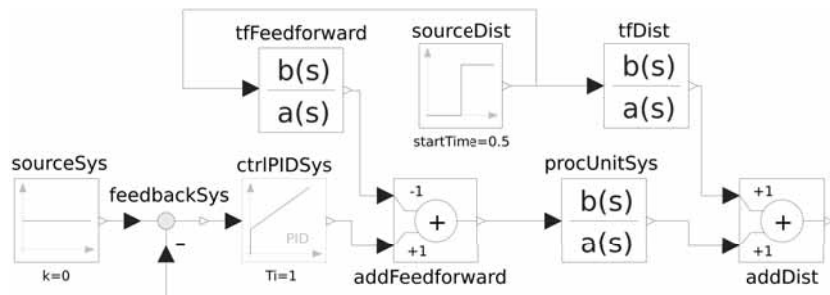


Figure 6: OpenModelica SM of the feedback control system with feedforward control.

- October 31, 2018).
 URL <https://www.pure-systems.com/mediapool/pv-whitepaper-en-04.pdf>
- [3] Starr L. *Executable Uml: How to Build Class Models*. Upper Saddle River, NJ, USA: Prentice Hall PTR. 2001.
- [4] Zeigler B, Kim T, Praehofer H. *Theory of Modeling and Simulation*. Elsevier Science. 2000.
- [5] Pawletta T, Schmidt A, Zeigler BP, Durak U. Extended Variability Modeling Using System Entity Structure Ontology Within MATLAB/Simulink. In: *Proceedings of the 49th Annual Simulation Symposium, ANSS '16*. San Diego, CA, USA: Society for Computer Simulation International. 2016; pp. 22:1–22:8.
 URL <http://dl.acm.org/citation.cfm?id=2962374.2962396>
- [6] Schmidt A, Durak U, Pawletta T. Model-based Testing Methodology Using System Entity Structures for MATLAB/Simulink Models. *SIMULATION*. 2016; 92(8):729–746.
 URL <https://doi.org/10.1177/0037549716656791>
- [7] Otter M, Elmqvist H, Blochwitz T, Mauss J, Junghanns A, Olsson H. *Functional Mockup Interface – Overview*. 2011 (accessed October 31, 2018).
 URL <https://web.archive.org/web/20110720233637/http://synchronics.inria.fr/lib/exe/fetch.php/modelica-fmi-elmqvist.pdf>
- [8] Modelica Association Project "FMI". *Functional Mock-up Interface for Model Exchange and Co-Simulation*. 2014 (accessed October 31, 2018).
 URL https://svn.modelica.org/fmi/branches/public/specifications/v2.0/FMI_for_ModelExchange_and_CoSimulation_v2.0.pdf
- [9] Schmidt A. Variant Management in Modeling and Simulation Using the SES/MB Framework. Ph.D. thesis, Rostock University. Submitted 10/2018.
- [10] Deatcu C, Folkerts H, Pawletta T, Durak U. Design Patterns for Variability Modeling Using SES Ontology. In: *Proceedings of the Model-driven Approaches for Simulation Engineering Symposium, Mod4Sim '18*. San Diego, CA, USA: Society for Computer Simulation International. 2018; pp. 3:1–3:12.
 URL <http://dl.acm.org/citation.cfm?id=3213214.3213217>

MATLAB/Simulink[®]'s Variant Manager vs SEStoPy

Christina Deatcu^{*}, Thorsten Pawletta, Hendrik Folkerts

Hochschule Wismar - University of Applied Sciences, Research Group CEA, Philipp-Müller-Straße 14,
23966 Wismar, Germany; ^{*}christina.deatcu@hs-wismar.de

Abstract. This paper describes how a complex case study for variability modeling and simulation from the documentation of MATLAB/Simulink can be remodeled with the extended System Entity Structure and Model Base (eSES/MB) approach using the Python-based tool *SEStoPy* and the accompanying modelbuilder *SESMoPy*.

Introduction

Generally, variability modeling can be seen as an approach to describe more than one system configuration. According to Capilla and Bosch [1], a software variability model has to describe the commonality and variability of a system at all stages of the software lifecycle. In simulation engineering, the problem of variability modeling is well known from the eighties. One of the first high level approaches for variability modeling in the design phase was introduced with the System Entity Structure (SES) by Zeigler in 1984 [2] and is constantly enhanced until today [3] [4]. The SES is a high level approach for variability modeling, particularly in simulation engineering. An SES describes a set of system configurations, i.e. different system structures and parameter settings of system components. In combination with a Model Base (MB), executable models can be generated from an SES.

A common tool for today's engineering applications is MATLAB/Simulink. It offers pragmatic solutions for variability modeling and can be seen as a quasi-standard in engineering. In the following section, the example of a power window control system modeled with different degrees of detail is introduced. This application is taken from MATLAB/Simulink's examples section and can be found in the online documentation [5]. Therefore, the Simulink model did not have to be created but needed to be analyzed. Remodeling the example using the extended System Entity Structure and Model Base

(eSES/MB) approach as described by Pawletta et al. [6] was successfully done.

After the problem description in Section 1, Sections 2 and 3 describe the two modeling approaches using the case study. Finally, a comparative evaluation is tried regarding: (i) the modeling effort, (ii) the clarity, (iii) the reusability, and (iv) the maintainability.

1 Problem Description

The passenger-side power window system of an automobile is modeled and simulated. The power window can be controlled from both the driver's and the passenger's side. Furthermore, closing should be stopped for security reasons, in case an obstacle is detected during upward movement of the window. The window shall be lowered by some centimeters in this case.

The system is modeled with different degrees of detail and with using different modeling concepts. From this problem specification, multiple model structures and configurations result, which are called variants. For all variants, main model parts are two switches for controlling the window, the control model, a process model of the window, and a model for 3D-animation. The occurrence of an obstacle can be controlled interactively. Furthermore, some outputs are needed to observe the window's behavior. A complete description of the *Power Window Control Project* example can be found in MATLAB's online documentation [5].

2 Implementation with Simulink's Variant Manager

Simulink as one of the most popular tools for model-based development provides special blocks for switching between model structures, the *Variant Subsystems Blocks*. Project management is facilitated by the

Simulink Project capabilities. Anyhow, that means that all variants need to be coded in only one model. Models of this kind are called 150%-models. The main Simulink model is depicted in Figure 1. The model comprises five variant subsystems: (i) the *driver_switch*, (ii) the *passenger_switch*, (iii) the model for obstacle detection, which is a submodel of the *power_window_control_system* model, (iv) the *window_system* model, and (v) the model *window_world*, that offers optional 3D-animation.

Both switches can operate either in a mode called normal mode or use a communication protocol (CP) implementation. For obstacle detection (DOE) four variants are available. The first variant is a simplistic continuous system model (Cont), the second uses power effects (PE), the third works with a visualization (Vis), and the last provides support for realistic armature and the communication protocol (RA CP). The *window_system* model subsystem (WS) comprises three variants, one simple continuous model (Cont), one reproducing power effects and additional 3D-visualization options (PE Vis), and a third variant where the realistic armature and the communication protocol is included (RA CP). The two variants for the *window_world* are a Simulink 3D animation or no animation at all.

Some of the variants use Stateflow, which is The MathWorks' implementation of state machines, and/or physical modeling, i.e. Simscape, submodels.

The active variant of the main model can be programmatically changed prior to simulation via the control variable *CV*. Table 1 lists the possible configurations and corresponding values of the control variable *CV*. Not all combinations of selected variants in

	Switches	DOE	WS
CV=1	Normal	Cont	Cont
CV=2	Normal	PE	PE Vis
CV=3	Normal	Vis	PE Vis
CV=4	Normal	RA CP	RA CP
CV=5	CP	RA CP	RA CP

Table 1: Valid variants, control variable values and selected submodels.

the subsystems are valid. If e.g. for the switches the communication protocol variant is chosen, the active variants of *window_system* and of the obstacle detec-

tion modeled in *power_window_control_system* need to be the communication protocol implementations, too. The variation in *window_world* is not addressed in Table 1, because all configuration sample scripts in the MATLAB/Simulink example implementation use the *Simulink_3D_Animation_View* variant, none uses the *No_View* submodel.

To allow only valid variant combinations, the value of *CV* is evaluated prior to simulation and mapped to specific control variables, called *variant control*, associated with the single variant subsystems. A variant choice is active, when the associated variant control evaluates to TRUE.

3 Implementation with the eSES/MB approach

For remodeling of the power window example, the platform-independent variability modeling tools *SESToPy* and *SESMoPy* were used. These Python-based tools are developed by the research group Computational Engineering and Automation (CEA) at Hochschule Wismar. Both tools and the infrastructure they are used with are described in detail in [7].

A family of systems, which in this context means all possible variants, can be defined within a System Entity Structure (SES) using *SESToPy*. An SES is represented by a directed acyclic graph with an amount of entity nodes, descriptive nodes and attributes. For usage with model generation, entity nodes are linked to basic models organized in a Model Base (MB). Attributes of an entity node correspond to the parameters of the belonging model component. The available three kinds of descriptive nodes specify the relationship between entities. Aspect nodes are used to define the composition of entities, multi-aspect nodes are a special kind of aspect nodes, where all component entities are of the same type. The third descriptive node type, the specialization node, describes the taxonomy of an entity.

Descriptive nodes can be seen as variation points of an SES. Associated with the descriptive nodes, there are rules which need to be evaluated when deriving one specific system variant. Each specific system variant comprises of a system structure and a parameter configuration.

The process of deriving a system variant from an SES is called *pruning* and the result is a Pruned Entity Structure (PES). The PES still is a directed acyclic graph like the SES, but without any variation points.

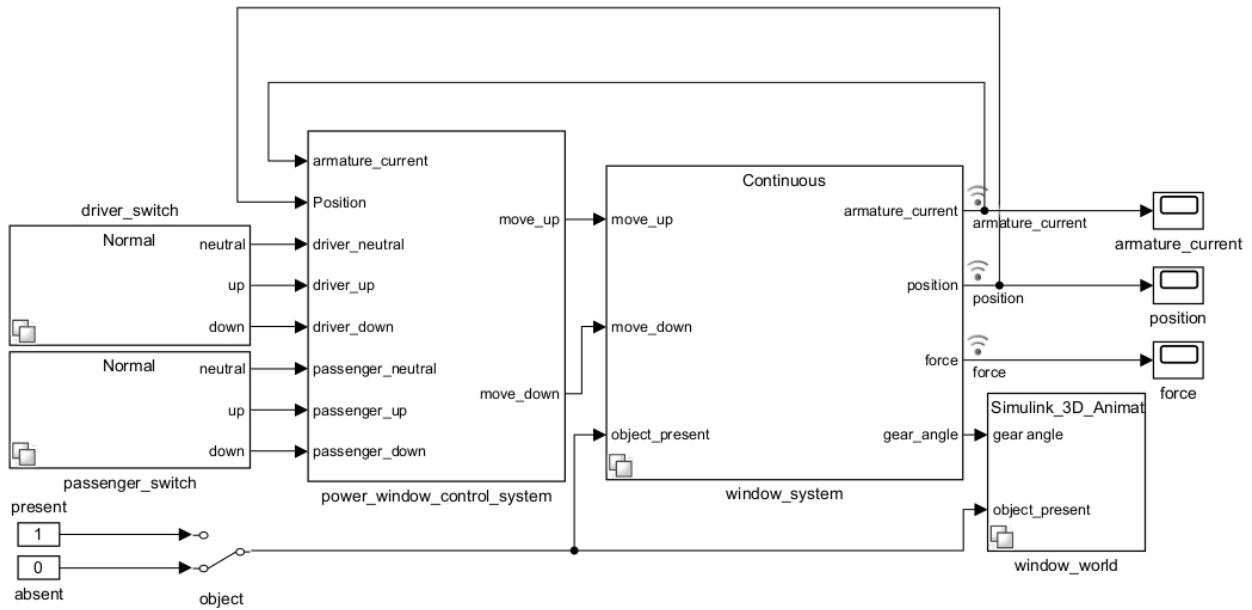


Figure 1: Overall Simulink model structure according to The MathWorks [5].

Applying the *flattening* method to a PES, all inner nodes are removed to obtain a Flattened Pruned Entity Structure (FPES). In conjunction with an MB, a fully configured and executable model can be generated from an FPES using the modelbuilder tool SESMoPy.

First step for remodeling the power window example was to identify the basic models which then needed to be organized in an MB. As the MB for the implementation of this example application the basic models were structured in four libraries. In the libraries, preconfigured blocks or submodels can be stored to reduce parametrization effort when defining the SES. The library *LibSimple* contains basic models which are blocks copied directly from Simulink's block library to allow preconfiguration and to make the MB independent from Simulink versions. The other three libraries *LibSwitches*, *LibCtrl*, and *LibWindowSys* contain more complex models and can be seen as user-defined libraries. All basic models could be extracted from the power window control project. Then all configuration variants were coded in an SES tree according to the structure of The MathWorks' 150%-model. For reasons of clarity, the SES was only detailed down to the level where variability occurs. Figure 2 shows the SES tree modeled in SESToPy.

The entire example and all variants are covered except the 3D-animation variation in *window_world*.

The SES was developed step by step making use of SESToPy's capability to combine several SES trees with the *merge* method [7]. The variation points are expressed by the specialization nodes *DriverSwitch-SPEC*, *PassengerSwitch-SPEC*, *DOE-SPEC*, and *WS-SPEC*. For each specialization node, a specialization rule is defined. During pruning the rules are evaluated and it is decided, which of the child nodes will be part of resulting PES. In analogy to the control variable *CV* and the variant control variables in The MathWorks' 150%-model, three SES variables are used. The SES variables are *driver_MODE*, *detect_O_E*, and *window_system*. Ranges of the SES variables and combinations among them are restricted by defining semantic conditions. Thus, only valid variants can be generated, i.e. the SES can be pruned only to a valid PES. Figure 3 shows the example of one possible PES. This PES corresponds to the Simulink variant, where *CV*=1. After flattening, model generation from the resulting FPES was finally successfully done with SESMoPy.

4 Conclusion

Both approaches offer the possibility to model and simulate variability systems. Regarding the modeling effort the approaches do not differ considerably. The eSES/MB approach gives more clarity during the mod-

Tree	Type	MB
PowerWindow	Entity	
PowerWindow-DEC	Aspect	
DriverSwitch	Entity	
DriverSwitch-SPEC	Spec	
Normal	Entity	'LibSwitches/Normal'
CP	Entity	'LibSwitches/Communication_Protocol'
PassengerSwitch	Entity	
PassengerSwitch-SPEC	Spec	
Normal	Entity	'LibSwitches/Normal'
CP	Entity	'LibSwitches/Communication_Protocol'
CtrlSys	Entity	
PWCS-DEC	Aspect	
RT-all-PWCS	Entity	'LibCtrl/Rate_Transitions'
DOE	Entity	
DOE-SPEC	Spec	
RA-CP	Entity	'LibCtrl/RA_CP_DOE'
Visu	Entity	'LibCtrl/Visu_DOE'
Cont	Entity	'LibCtrl/Continuous_DOE'
PE	Entity	'LibCtrl/PowerEffects_DOE'
Val-D-P-State	Entity	
Val-D-P-State-MASP	Maspect	
State-Val	Entity	'LibCtrl/StateValidation'
Ctrl	Entity	'LibCtrl/control'
Pulse-Gen	Entity	'LibCtrl/10ms'
WindowSys	Entity	
WS-SPEC	Spec	
Cont-WS	Entity	'LibWindowSys/Continuous'
PE-WS	Entity	'LibWindowSys/PowerEffects'
RA-WS	Entity	'LibWindowSys/RealisticArmature'
ManualSwitch	Entity	'LibSimple/Manual_Switch'
W-Object	Entity	'LibSimple/Constant'
Wo-Object	Entity	'LibSimple/Constant'
Output-Force	Entity	'LibSimple/Out'
Output-Position	Entity	'LibSimple/Out'
Output-Armaturecurrent	Entity	'LibSimple/Out'
Terminator-Gear-Angle	Entity	'LibSimple/Terminator'

Figure 2: SES tree of the power window control system family in SESToPy.

Tree	Type	MB
PowerWindow	Entity	
PowerWindow-DEC	Aspect	
Normal_DriverSwitch	Entity	'LibSwitches/Normal'
Normal_PassengerSwitch	Entity	'LibSwitches/Normal'
CtrlSys	Entity	
PWCS-DEC	Aspect	
RT-all-PWCS	Entity	'LibCtrl/Rate_Transitions'
Cont_DOE	Entity	'LibCtrl/Continuous_DOE'
Val-D-P-State	Entity	
Val-D-P-State-DEC	Aspect	
State-Val_1	Entity	'LibCtrl/StateValidation'
State-Val_2	Entity	'LibCtrl/StateValidation'
Ctrl	Entity	'LibCtrl/control'
Pulse-Gen	Entity	'LibCtrl/10ms'
Cont-WS_WindowSys	Entity	'LibWindowSys/Continuous'
ManualSwitch	Entity	'LibSimple/Manual_Switch'
W-Object	Entity	'LibSimple/Constant'
Wo-Object	Entity	'LibSimple/Constant'
Output-Force	Entity	'LibSimple/Out'
Output-Position	Entity	'LibSimple/Out'
Output-Armaturecurrent	Entity	'LibSimple/Out'
Terminator-Gear-Angle	Entity	'LibSimple/Terminator'

Figure 3: Resulting PES for driver_MODE=1, detect_O_E=1, and window_system=1.

eling process, because the overall structure of the model can be captured with one sight. If one uses Simulink's

Model Explorer to determine the structure, one cannot see, that and where a model contains variant subsystems until one selects a variant subsystem block. The *Variant Manager* offers a view, where the overall structure is displayed, but e.g. block properties cannot be seen then. Information about the model is distributed over several tools, which may confuse new users. A survey among our students came to the result, that eSES/MB is a lot easier to get started with. Reusability of models is ensured for both approaches but may differ in the effort. The maintainability is closely associated with the reusability. A final comparison is not possible on the basis of just one example. According to the available findings, the eSES/MB approach appears to be easier in use for beginners.

Acknowledgement

Main preliminary work is done by our student Paul Buschow who analyzed the power window example and remodeled it with the SES/MB Toolbox for MATLAB for his bachelor thesis.

References

- [1] Capilla, R., Bosch, J.. *Binding Time and Evolution*. In Systems and Software Variability Management, edited by R. Capilla, J. Bosch, and K. C. Kang, pp. 57-73. Springer 2013, Berlin Heidelberg, Germany.
- [2] Zeigler, B. P.. *Multifaceted Modelling and Discrete Event Simulation*. Cambridge 1984, Academic Press.
- [3] Zeigler, B. P., Kim, T. G., Praehofer, H.. *Theory of Modeling and Simulation*. 2nd ed. San Diego 2000, CA, USA, Academic Press.
- [4] Schmidt A. *Variant Management in Modeling and Simulation Using the SES/MB Framework*. Ph.D. thesis, Rostock University, Germany. Submitted 10/2018.
- [5] The MathWorks Website *Power Window Control Project*. <https://de.mathworks.com/help/simulink/examples/power-window-control-project.html>, accessed 2018-11-01.
- [6] Pawletta, T., Schmidt, A., Durak, U., Zeigler, B. P.. *A Framework for the Metamodeling of Multivariant Systems and Reactive Simulation Model Generation and Execution*. SNE - Simulation Notes Europe. 2018; SNE 28(1): 11-18. doi: 10.11128/sne.28.tn.10402.
- [7] Folkerts, H., Pawletta, T., Deatcu, C.. *A Python Framework for Model Specification and Automatic Model Generation for Multiple Simulators*. Submitted to this workshop.

Automatische Modellbildung mittels SES/MB Framework und einer Template-Erweiterung

Alexander Martens*, Christian Bock, Olaf Simanski, Olaf Hagendorf

Hochschule Wismar – University of Applied Sciences: Technology, Business and Design
Research Group Computational Engineering and Automation (CEA); <http://www.cea-wismar.de>
Kontakt: *alexander.martens@hs-wismar.de

Abstract. In Kooperation mit der Dr. Diestel GmbH aus Rostock, wird aktuell an einem FuE Projekt „Entwicklung eines universellen Regelungsmoduls für die Druck- und Volumenstromregelung, sowie die Bewertung der Regelung auf Busebene“ gearbeitet. Die Dr. Diestel GmbH ist im Bereich Lüftungs-, Klima, Kälte- und Reinraumtechnik tätig. Der Bedarf von kleinen bis mittelständigen Firmen an Reinraumtechnik steigt zunehmend. Die umgesetzten Kundenprojekte bestehend aus durchschnittlich mehreren hundert Metern Leitungen mit mehreren hundert angeschlossenen Teilnehmern. Die angeschlossenen Geräte werden über den LON-Bus miteinander verbunden. Eine einfache Vorhersage der Buslast ist aufgrund der Komplexität der Systeme vor einer Inbetriebnahme kaum möglich.

Ein Bestandteil des FuE Projektes ist die Entwicklung eines Bus Modells, um die Kosten möglichst exakt schätzen zu können. Hierzu wird ein Kundenprojekt in der Projektierungsphase simuliert, um eine Buslast- und Fehlerratenabschätzung durchführen zu können. Dies ermöglicht eine konkrete Aussage, ob die geplanten Busstrukturen die entstehenden Buslasten und Fehlerratenlimits einhalten. Daraus erfolgt die Aussage, ob die Kommunikationswege aufgeteilt werden müssen oder andere Geräteparameter zu wählen sind, um die entstehende Buslast zu verringern. Für ein Kundenprojekt werden in der Projektierungsphase Baugruppen mit Verbindungslisten und geschätzten Leitungslängen entworfen. Mittels automatischer Modellbildung wird ein Modell eines LON-Busses automatisch generiert und simuliert. Nach der Simulation kann die geplante Projektierung hinsichtlich Buslast und Fehlerrate bewertet werden.

Einleitung

Dr. Diestel GmbH ist im Bereich Lüftungs-, Klima-,

Kälte- und Reinraumtechnik tätig. Um die einzelnen Bereiche stetig zu verbessern, werden Forschungen unter anderem im Bereich Regelung von Lüftungsanlagen durchgeführt. [1]

Die verwendeten Sensoren, Aktoren und Regeleinrichtungen können z.B. über einen LON-Bus (Local Operating Network) verbunden werden. Nach der Installation von großen Netzwerken können aufgrund der Anzahl der Baugruppen, Typ und Kommunikationsform von Datenpunkten, sowie der Leitungslänge Probleme entstehen, da Fehlerrate und Buslast in der Projektierung kaum abzuschätzen ist.

Um die Kosten in der Projektierungsphase exakter planen zu können, wird das Kundenprojekt hinsichtlich Buslast- und Fehlerratenabschätzung simuliert. Dies ermöglicht eine Aussage, ob die geplanten Busstrukturen die entstehenden Buslasten und Fehlerratenlimits einhalten. Daraus erfolgt die Aussage, ob die Kommunikationswege aufgeteilt werden müssen oder andere Geräteparameter zu wählen sind, um die entstehende Buslast zu verringern.

Im folgenden Paper wird auf die automatische Erstellung und Ausführung von Simulationsmodellen (SM) eingegangen. Hierbei werden unter anderem System Entity Structure (SES) mit Modellbase (MB) und einer Template Erweiterung näher betrachtet.

1 SES/MB Framework

Die SES beschreibt eine Ontologie die Systemdesign, Modellierung oder auch Simulation in System- und Mengentheoretischer Form mit deren Beziehungen untereinander in Zusammenhang bringt [2,3,4]. Dies ermöglicht die Darstellung z.B. einer Aufgabe, Arbeitsschritte oder auch Aufbau eines Objektes.

Im Folgenden wird kurz auf das SES/MB Framework

im Allgemeinen eingegangen. Eine Erweiterung der MB um Templates wird vorgestellt.

1.1 Allgemein

Die SES wird in Form einer Baumstruktur dargestellt. Sie besteht aus Knoten und Kanten die parametrisiert werden können. In [3,5,6] wird der Ablauf zur automatischen Erzeugung von Simulationsmodellen dargestellt (siehe Abbildung 1).

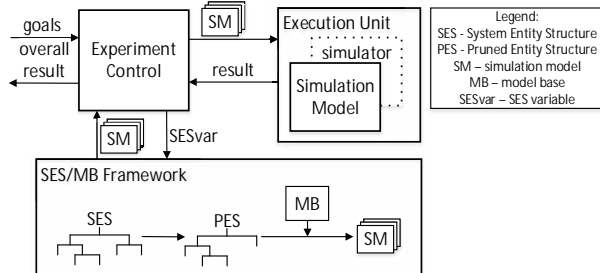


Abbildung 1: Erweiterte SES/MB Übersicht [4,5]

Die Experiment Control übernimmt die Steuerung zur Bildung und Simulation von SM aus einer vorher definierten SES. Die Knoten und Kanten können Variablen (SESvar) besitzen. Die einzelnen Werte der SES Variablen werden von der Experiment Control gesetzt. Im Anschluss werden mittels Beschneidung (engl. pruning) der SES nicht benötigte Knoten und Kanten im Baum entfernt. Mit der so entstandenen Pruned Entity Structure (PES) und der MB, welche Referenzen auf Basismodelle enthält, werden SM generiert.

Nach dem Erzeugen der SM werden diese über die Experiment Control an die Execution Unit weitergegeben und dort ausgeführt. Nach der Simulation werden die Ergebnisse zur weiteren Analyse ausgegeben.

1.2 SES/MB Framework mit einer Template-Erweiterung

Die Erweiterung des SES/MB Frameworks beinhaltet die Einführung und Verwendung von Templates.

Ein Template besteht aus Parametersätzen für mehrere Kanten und Knoten. Die Templates werden nach dem Pruning für die Erzeugung der SM verwendet. Dadurch wird die MB zur Model- und Template Base (MTB) erweitert (siehe Abbildung 2).

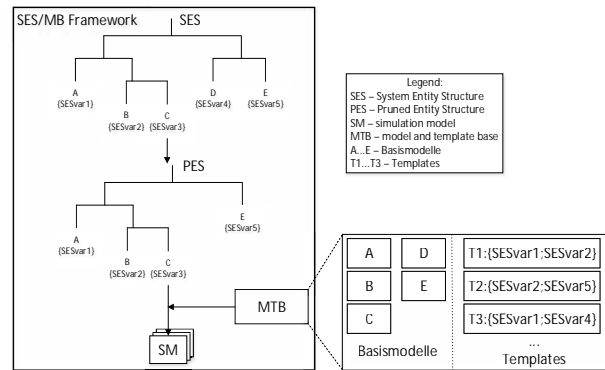


Abbildung 2: Erweiterung der MB zur MTB

2 Automatische Modellbildung

Die durch die Dr. Diestel GmbH realisierten Netzwerke in der Luft- und Reinraumtechnik setzen auf den LON-Bus als Kommunikationsmedium. Im LON-Bus werden intelligente Knoten verwendet. Diese Knoten kommunizieren mit den angeschlossenen Teilnehmern und entscheiden selbständig, entsprechend ihrer Vorgaben, über neue Zustände. Zusätzlich können beispielsweise Regelungen knotenlokal umgesetzt werden. Der Einsatz eines Leitrechners ist optional.

In [9] wurde die Entwicklung eines Simulationsmodells „LON-Bussimulation für die Busauslastung und Fehlerabschätzung mit SimEvents“ beschrieben. Um die Erstellung sowie die Ausführung der Modelle mit SimEvents automatisch aus der Projektierung zu bilden und auszuführen wurde ein SES/SM Generator realisiert.

2.1 Grundlagen

Der Modellgenerator ermöglicht es, aus einem Projektierungstool heraus Simulationsmodelle mit Hilfe des SES/MTB Ansatzes zu generieren. Dazu werden nach dem Pruning die SM mit der MTB gebildet (siehe Abbildung 3).

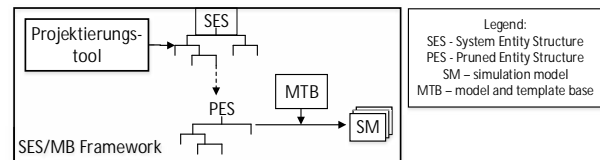


Abbildung 3: Generierung von SM aus einem Projektierungstool heraus

Zu Beginn, d.h. bei der Projektierung, werden alle auf

dem Bus befindlichen Geräte in Anzahl, Position im Bus und gewünschte Templates festgelegt. Nach Bildung der SES mit allen Knoten und Kanten wird das Pruning durchgeführt. Nach dem Pruning wird zusammen mit der MTB das SM gebildet und simuliert. Nach der Simulation können die Ergebnisse vom Benutzer ausgewertet werden und der Vorgang ggf. mit angepassten Geräten und Templates wiederholt werden (siehe Abbildung 4).

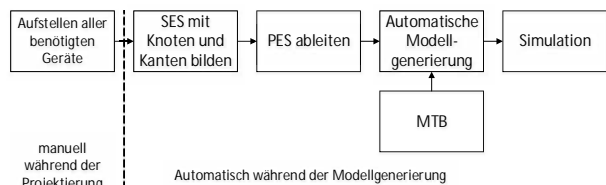


Abbildung 4: Allgemeiner Ablauf automatische Modellbildung

2.2 Automatische Modellbildung am Beispiel LON-Bus mittels SES/MTB

Im Folgenden wird die automatische Modellgenerierung am Beispiel des LON-Busses mit SimEvents näher betrachtet.

Zu Beginn werden in der Projektierungsphase aus einem Kundenprojekt heraus alle benötigten Teilnehmer in yED-Graph [10] visualisiert. In Abbildung 5 ist ein Auszug für eine Projektierung mit Brandschutzklappen (BSK), Zuluftregler (ZU) und Abluftregler (AB) dargestellt.

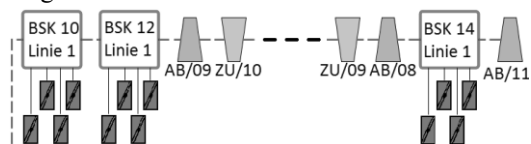


Abbildung 5: Beispiel yED-Graph (Auszug)

Die Teilnehmer werden mit unterschiedlichen Formen und Farben, sowie unterschiedlicher Beschriftung dargestellt. Diese Unterschiede werden mit MATLAB analysiert und in sequentieller Reihenfolge mit eindeutigen Bezeichnungen in einer CSV-Datei (engl. Comma-separated values) abgelegt. In Tabelle 1 ist eine Linie auszugsweise dargestellt. Die Kabellängen zwischen den Knoten wurden anhand von Leitungsplänen des Gebäudes festgelegt. Weiter wurden die Templates der einzelnen Knoten passend ihrer späteren Funktion gesetzt.

L-IP

Linie 1 UG LINX Port			
von	nach	Kabellänge	Template von Spalte 1
Bauteil			
ISP B2 01	BSK 10 Linie 1	4m	Template_13
BSK 10 Linie 1	BSK 12 Linie 1	15m	Template_04
BSK 12 Linie 1	AB/09	4m	Template_04
AB/09	ZU/10	6m	Template_03
ZU/10	...	4m	Template_09
...
...	ZU/09	4m	Template_03
ZU/09	AB/08	8m	Template_02
AB/08	BSK 14 Linie 1	3m	Template_06
BSK 14 Linie 1	AB/11	25m	Template_03
AB/11	0	8m	Template_08
		109m	

Tabelle 1: Leitungslängen (Auszug)

Templates beschreiben in der Anwendung des LON-Busses die Spezifikation Teilnehmerübergreifend, speziell das Sendeverhalten der Teilnehmer. Durch die Templates werden Sendintervalle, Sendegrößen und Acknowledgements gesetzt.

Nach Auswahl der Templates wird ein Simulink-Modell erzeugt und alle benötigten Blöcke automatisch aus der MTB (siehe Abbildung 6) in das Modell kopiert und die Knoten Variablen auf Basis der gewählten Templates gesetzt. Die Blöcke werden nach dem Platzieren automatisch miteinander verbunden (siehe Abbildung 7).

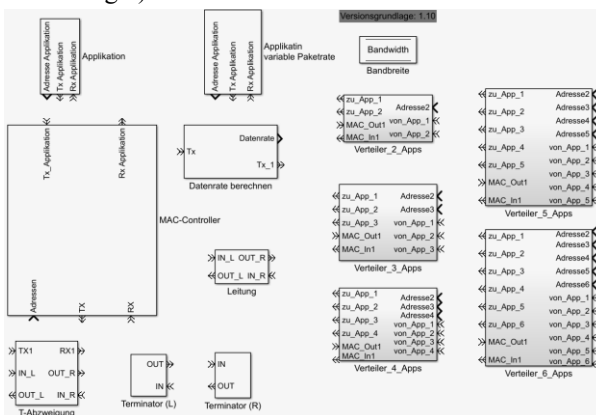


Abbildung 6: MTB

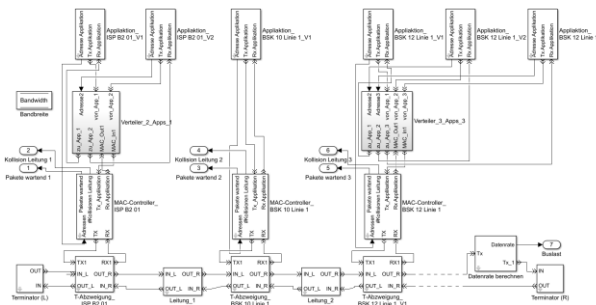


Abbildung 7: Modell (Auszug)

Damit der Benutzer die Buslast und erzeugten Fehler nach der Simulation auswerten kann, werden zusätzlich

„to Workspace“-Blöcke in das SM eingefügt, um die Daten in einer Struktur absichern zu können. Weiterhin werden globale Einstellungen im Modell wie die Simulationszeit gesetzt. Nach der Simulation werden die Daten visualisiert und können vom Benutzer ausgewertet werden.

3 Fazit und Ausblick

Die automatische Modellbildung wurde erfolgreich umgesetzt und ermöglicht so, eine Generierung aus der Projektierung heraus. Damit können Netzwerke hinsichtlich der Buslast und Fehlerrate simuliert und analysiert werden. Der Anwender fügt zu Beginn die benötigten Teilnehmer mittels Projektierungstool hinzu und schätzt die Leitungslängen. Danach wird vom Nutzer festgelegt, welche Templates auf die jeweiligen Teilnehmer angewendet werden. Nach dem Durchlauf der automatischen Modellbildung mit anschließender Simulation kann der Benutzer potentielle Probleme im Netzwerk erkennen, die Netzstruktur oder die verwendeten Templates ändern und das Modell ggf. erneut simulieren.

Durch die automatische Modellbildung mit anschließender Simulation ist es nun möglich, unterschiedliche Aufbauten eines Netzes im Vorfeld zu testen und zu bewerten, ohne erst nach der Realisierung Engpässe feststellen zu müssen.

Das Projekt wird gefördert durch das TBI MV, Förderkennzeichen: TBI-V-1-135-VBW-048.

References

- [1] Dr. Diestel GmbH, *Unternehmensprofil*, <https://www.dr-diestel.de/unternehmensprofil/>, Abruf: 01.08.2018
- [2] Zeigler B. P., Hammonds P.E., *Modeling and Simulation-Based Data Engineering*. Elsevier Academic Press 2007
- [3] Hagendorf, O. *Simulation Based Parameter and Structure Optimisation of Discrete Event Systems*. ARGESIM/ASIM – Verlag, Wien, 2010
- [4] Zeigler B. P., Praehofer H., Kim T. G., *Theory of Modeling and Simulation*. Academic Press 2000
- [5] Deatcu C, Folkerts H, Pawletta T, Durak U, Design Patterns For Variability Modeling Using SES Ontology. *SpringSim-Mod4Sim*, 2018 April, USA
- [6] Research Group CEA, *Documentation SEStoPy – SES Tools Python*. 2018 Sep
- [7] Pascheka D, Pawletta T, System Entity Structure Ontology Toolbox for MATLAB/Simulink: Used for Variant Modelling, *MATHMOD 2015 - 8th Vienna International Conference on Mathematical Modelling*, 2015 Feb, Austria
- [8] Deatcu C, Folkerts H, Pawletta T, Durak U, How to Define SES Trees for Variability Modeling. *ASIM 2018 – 24. Symposium Simulationstechnik*, 2018 Oktober, Germany, pp. 35-44
- [9] Martens A. LON-Bussimulation mit SimEvents zur Auslastungs- und Fehlerabschätzung, *ASIM 2018 – 24. Symposium Simulationstechnik*, 2018 Oktober, Germany, pp. 225-233
- [10] yWorks, *yED-Graph*, <https://www.yworks.com/products/yed> Abruf: 12/2018

Modellierung und Simulation vernetzter mechatronischer Komponenten einer Fertigungsstraße im Kontext der Industrie 4.0

Xiaobo Liu-Henke*, Sören Scherler, Or Aviv Yarom, Jie Zhang

Ostfalia Hochschule für angewandte Wissenschaften, Fakultät Maschinenbau, Institut für Mechatronik (IMEC),
Salzdahlumer Str. 46/48, 38302 Wolfenbüttel, Deutschland; *x.liu-henke@ostfalia.de

Abstract. Im vorliegenden Beitrag wird ein Ansatz zur Modellierung und Simulation hochautomatisierter Fertigungsstraßen sowie deren Erweiterung um Industrie 4.0-Lösungen (I4.0-Lösungen) zum Aufbau einer Modellbibliothek dargestellt. Die Simulationen sollen als fundierte Grundlage dienen, um Unternehmen bei der Einführung von Industrie 4.0-Lösungen zu unterstützen.

Einleitung

Im Rahmen des vom Europäischen Fonds für regionale Entwicklung (EFRE) geförderten Verbundprojekts *Methoden und Werkzeuge für die synergetische Konzipierung und Bewertung von Industrie 4.0-Lösungen* (Synus) ist die Entwicklung wissenschaftlicher Methoden und Werkzeuge zur simulationsgestützten, vorausschauenden Bewertung von I4.0-Lösungen in unterschiedlichen Unternehmensbereichen wie der Produktentwicklung, der Produktionsplanung und der Produktion, um insbesondere KMU bei der Einführung von Industrie 4.0-Lösungen zu unterstützen.

Wesentliche Thematik des Teilprojekts *Modellbasierte Konzeption und Bewertung von Industrie 4.0-Lösungen zu Vernetzung mechatronischer Komponenten in Produktionsanlagen durch Digitalisierung* (MiMec) ist die Modellierung und Simulation vernetzter mechatronischer Komponenten bestehender Industrieanlagen und verfügbarer I4.0-Lösungen sowie die systematische Integration autonomer Transportsysteme durch vollständige digitale Vernetzung in intelligenten, cyber-physischen Produktionsanlagen.

Ein wesentliches Teilziel ist herbei die Entwicklung einer Modellbibliothek zur Abbildung bestehender Produktionsanlagen und zur Erweiterung dieser um I4.0-Lösungen. Anhand von Simulationen und einer erarbeiteten Bewertungsmethodik sollen Handlungsempfehlungen

zur Einführung von I4.0-Lösungen gegeben werden.

Im vorliegenden Beitrag soll ein erster Ansatz zur Modellierung und Simulation hochautomatisierter und vernetzter Produktionsanlagen dargestellt werden. Anhand zweier exemplarischer Fallbeispiele werden die konkreten Einflüsse digitaler Vernetzung intelligenter Fertigungsanlagen auf die Gesamtdurchlaufzeit einer individualisierten Einzelstückfertigung aufgezeigt.

1 Motivation

Die Komplexität in der Fertigung steigt insbesondere durch immer stärker individualisierbare Produkte mit sehr kleinen Losgrößen bis hin zur Einzelstückfertigung. Die Fertigung muss flexibel auf diese kleinen Losgrößen reagieren können und in der Lage sein, produktspezifische Eigenschaften wie Entwicklungs- und Fertigungsdaten handhaben und auch verarbeiten zu können. Die mit kleinen Losgrößen verbundenen langen Wechsel- und Stillstandszeiten sowie der hohe Personalkosteneinsatz stellen große Herausforderungen für Unternehmen mit konventionellen bzw. hochautomatisierten Produktionsanlagen dar.

Im Zuge der vierten industriellen Revolution wird die Fertigung zum einen zunehmend intelligenter und kann bspw. mithilfe der Informationen von RFID-Chips selbstorganisiert Tätigkeiten wie Werkzeugwechsel oder das Beschaffen von Werkzeugen durch ein fahrerloses Transportsystem veranlassen. Zum anderen werden Maschinen digital mit anderen Maschinen vernetzt und können mit diesen Informationen austauschen. Dieser Informationsaustausch intelligenter Maschinen, Lagersystemen und Betriebsmitteln in cyber-physischen Systemen (CPS) ist also eine Schlüsseltechnologie zur Effizienzsteigerung hinsichtlich verschiedener Zielkriterien wie Auslastung, Durchlaufzeit oder Individualisierung [1].

Die gezielte Nutzung des Internet of Things (IoT) im Sinne industrieller Anwendungen stellt dabei die technische Grundlage für I4.0 und CPS dar [2] [3]. Trotz vielversprechender Vorteile sind Hemmnisse wie mangelnde Kenntnisse über verfügbare I4.0-Lösungen oder hohe Investitionen insbesondere für kleine und mittlere Unternehmen (KMU) Gründe für Zurückhaltung [4].

Zusätzlich führt die hohe Komplexität der I4.0-Technologien, insbesondere die Vernetzung von Systemen, zu einer schwierigen Einschätzung der tatsächlichen Auswirkungen in Betracht gezogener I4.0-Lösungen.

2 Methode und Konzept

Wegen der stetig steigenden Komplexität mechatronischer, vernetzter Systeme ist eine strukturierte, ganzheitliche Entwurfsmethodik unausweichlich.

In einem Top-Down-Verfahren erfolgt zunächst eine Modularisierung und hierarchische Strukturierung eines komplexen Gesamtsystems in intelligente, gekapselte Teilsysteme bestehend aus mechatronischen Komponenten mit definierten Schnittstellen. Abbildung 1 zeigt beispielhaft die mechatronische Struktur einer I4.0-Produktionsanlage mit vier Hierarchieebenen nach [5].

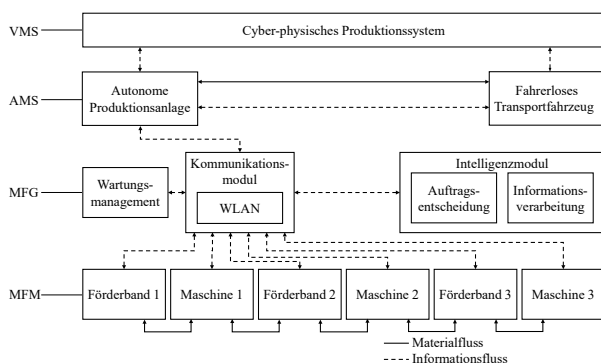


Abbildung 1 Mechatronische Strukturierung einer Produktionsanlage.

Die unterste Hierarchieebene bilden die mechatronischen Funktionsmodule (MFM) bestehend aus nicht weiter teilbaren mechatronischen Systemen inklusive Grundaufbau, Sensorik, Aktorik und Informationsverarbeitung. Jedes gekapselte MFM besitzt eine definierte Funktionalität und beschreibt die Dynamik des Systems. Durch Kopplung mehrerer MFM und Hinzufügen einer Informationsverarbeitung entstehen mechatronische Funktionsgruppen (MFG). MFG ermöglichen durch Nutzung der unterlagerten MFM die Realisierung höherwertiger

Funktionen. Die Kombination von MFG stellen autonome Mechatronische Systeme (AMS) dar, welche in diesem Beispiel durch die autonome Produktionsanlage selbst oder autonome Transportfahrzeuge repräsentiert werden. Vernetzte AMS bilden schließlich ein vernetztes mechatronisches System (VMS) bzw. ein cyber-physisches Produktionssystem, welches bspw. einer I4.0 Fertigungsstraße oder einer I4.0-Fabrik entspricht.

Ausgehend vom Produkt als zentrales Element der Fertigung werden zunächst Produkteigenschaften definiert, welche von den Komponenten zur Auftragsbearbeitung benötigt werden. Im Hinblick auf die individualisierte Einzelstückfertigung als eines der Teilziele von I4.0, wurden eine Sensor- und Produkt-ID eingeführt. Die Sensor-ID ist eine eindeutige Ziffernkombination, die eine Zuordnung zu einer Produkt-ID, welche den Produkttyp beschreibt, ermöglicht.

Zur Abbildung einer Fertigungsstraße werden in einem ersten Ansatz folgende Module mit spezifischen Funktionalitäten und Schnittstellen konzipiert.

- **Maschinenmodul:** Es deckt Funktionalitäten beliebiger Fertigungsanlagen ab. Unter der Annahme, dass ein Maschinenmodul anhand von Sensor- und Produkt-ID einem Werkstück ein Fertigungsprogramm zuordnen kann, werden die entsprechenden Parameter in Form eines produktspezifischen Maschinenkatalogs hinterlegt. Der Maschinenkatalog enthält in jeder Maschine die für jeden Produkttyp benötigten Fertigungsdaten, beispielsweise die Fertigungszeit.
- **Förderbandmodul:** Transport des Werkstücks/Produkts zwischen einzelnen Maschinenmodulen oder anderen Komponenten. Parameter wie Förderstrecke oder -geschwindigkeit sind dabei für jedes Förderbandmodul einstellbar.
- **Fahrerloses Transportfahrzeug (FTF):** Erledigung von Besorgungsfahrten auf Abruf, z.B. zur Beschaffung von Ersatzteilen oder Werkzeugen. Dabei kann die Fahrgeschwindigkeit jedes FTF beliebig festgelegt werden.
- **Kommunikationsmodul:** Informationsschnittstelle zwischen mehreren gleichen oder unterschiedlichen Modulen. So kann bspw. eine Maschine einem FTF direkt eine Ersatzteilbestellung erteilen.
- **Intelligenzmodul:** Künstliche Intelligenz im Sinne der Entscheidungsfindung und Informationsverarbeitung. Das mit Kommunikations- und Intelligenzmodul erweiterte FTF aus dem obigen Beispiel

könnte die Besorgungsfahrt selbstständig planen.

- **Cyberphysisches Produktionssystem:** Verwaltung von Informationen wie Sensor- und Produkt-ID der Produkte und Übermittlung dieser Informationen, bspw. durch WLAN, an alle Komponenten der Fertigungsstraße. Das übergeordnete IoT-basierte Modul koordiniert die Abläufe im gesamten Produktionssystem.

Der modulare Modellierungsansatz in Form einer Modellbibliothek ermöglicht die in Anzahl und Art nahezu beliebige Kombination verschiedener Module zu komplexen Gesamtsystemen. Um die Wiederverwendbarkeit und Kompatibilität der Module untereinander zu gewährleisten, ist die Wahl der Schnittstellen elementar.

Nach der mechatronischen Strukturierung erfolgt die mechatronische Komposition. Dabei wird in einem Bottom-Up-Verfahren jedes Modul, begonnen mit der tiefsten Hierarchieebene, in einem durchgängig modellbasierten, verifikationsorientierten Prozess ausgelegt, validiert und sukzessive zum Gesamtsystem integriert.

3 Modellbildung

3.1 Maschinenmodul als Grundmodul

Die Modellierung der Module und ihrer hinterlegten Algorithmen wird im Folgenden exemplarisch für das Maschinenmodul dargestellt (Abbildung 2).

Ein Maschinenmodul besitzt die vier Zustände „Ruhezustand“, „Fertigungszustand“, „Anpassungszustand“ und „Fehlerzustand“. Eine Anpassung ist als ein Prozess definiert, den die Maschine selbstständig durchführen kann, bspw. ein Update des Fertigungsprogramms oder die Erwärmung des Arbeitsraumes. Fehlerzustand bedeutet, dass eine Anpassung vorgenommen werden muss, die von einem äußeren Einfluss abhängig ist, wie die Beschaffung von Ersatzteilen mit einem FTF oder ein Werkzeugwechsel durch einen Menschen.

Nach Erhalt der Produktinformationen wird die aktuelle Maschinenkonfiguration, z.B. das aktuelle Werkzeug oder Fertigungsprogramm, anhand des in Kapitel 2 beschriebenen Maschinenkatalogs für das zu fertigende Produkt geprüft. Stimmen Soll- und Ist-Konfiguration überein, kann nach Eintreffen des Produkts vom Ruhe- in den Fertigungszustand gewechselt werden. Ist zur Fertigung eine Anpassung nötig, wird geprüft, ob diese selbst-

ständig durchgeführt werden kann oder externe Unterstützung benötigt wird. Die Maschine wechselt in den entsprechenden Zustand „Anpassung“ oder „Fehler“ und gibt ggf. ein Fehlersignal mit Fehlercode aus. Nachdem die Anpassung von einer anderen Komponente quittiert wurde, wird die Konfiguration erneut geprüft.

Der Zustand „Fertigung“ wird durch den Block „Produktion“ aus Abbildung 2 repräsentiert. Innerhalb dieses Blocks sind verschiedene Aktionen, wie die Berechnung relevanter Prozessgrößen (Temperaturen, Leistungsaufnahme, ...) anhand eines hinterlegten Dynamikmodells, oder der schlichte Ablauf einer Bearbeitungszeit möglich. Der aktuelle Maschinenstatus sowie die berechneten Prozessgrößen werden durchgehend ausgegeben.

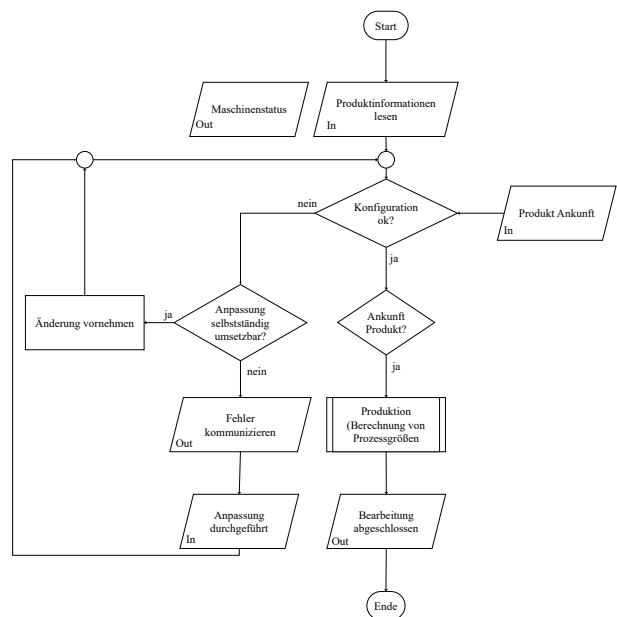


Abbildung 2 Ablaufplan des Maschinenmodells.

Im Rahmen dieses Beitrags wird durch die Kombination der zuvor beschriebenen Module eine Fertigungsstraße mit und ohne I4.0-Komponenten modelliert. Anhand einer späteren Simulation soll der Einfluss der eingesetzten I4.0-Module analysiert werden.

3.2 Fertigungsstraße als Gesamtsystem

Zur Veranschaulichung der beschriebenen Module und deren Verwendung wird in diesem Abschnitt die Zusammenstellung mehrerer Module gleicher und unterschiedlicher Arten zu einer konkreten Fertigungsstraße aufgezeigt. Die Auswahl und Kombination der im Folgenden beschriebenen Module ist als exemplarisch zu betrachten.

Hochautomatisierte Fertigungsstraße

Abbildung 3 zeigt den Aufbau der beschriebenen Produktionsstraße schematisch.

Modelliert wird eine automatisierte Einzelstückfertigung mit drei beteiligten Maschinen. Die erste Maschine führt eine standardisierte mechanische Bearbeitung aus. Standardisiert bedeutet in diesem Fall, dass die Arbeitsschritte für alle Produkttypen gleich ablaufen. In der zweiten Maschine wird ein individuelles Muster in die Oberfläche des Produkts graviert. Dabei muss für jedes Werkstück ein neuer CNC-Code geladen werden. Die letzte Maschine führt mit einer abschließenden Lackierung wieder einen Standard-Arbeitsschritt durch.

Vor jeder Maschine befindet sich ein Förderband zum Transport der Werkstücke innerhalb der Produktionsstraße. Hinter der letzten Maschine fallen die Werkstücke in einen Sammelbehälter, welcher im Folgenden nicht explizit berücksichtigt wird.

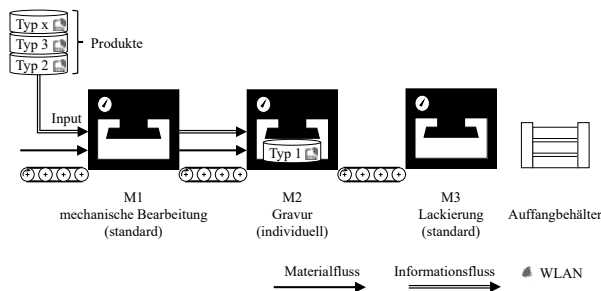


Abbildung 3 Schematischer Aufbau der konventionellen Fertigungsstraße.

In einem ersten Schritt wird eine konventionelle, hochautomatisierte Fertigungsstraße betrachtet. In diesem Fall kann M2 die individualisierten Produktionsdaten nur über das Produkt selbst mittels eines RFID-Chips erfassen.

Teilautonome Fertigungsstraße

In einem zweiten Schritt wird die konventionelle, hochautomatisierte Fertigungsstraße um Vernetzungs- und Intelligenzmodule erweitert. Der prinzipielle Aufbau ist in Abbildung 4 dargestellt. Somit wird die Kopplung von Informations- und Materialfluss (Abbildung 3) in der teilautonomen Fertigungsstraße aus Abbildung 4 aufgelöst. Diese Trennung eröffnet zahlreiche Möglichkeiten zur Optimierung der Produktion, wie die Verkürzung der Durchlaufzeiten durch intelligente Maschinenalgorithmen. Darüber hinaus sind Verbesserungspotentiale in

weiteren Bereichen, z.B. der verbrauchsgesteuerten Ersatzteilbestellungen oder dem prädiktiven Instandhaltungsmanagement durch Fernüberwachung denkbar.

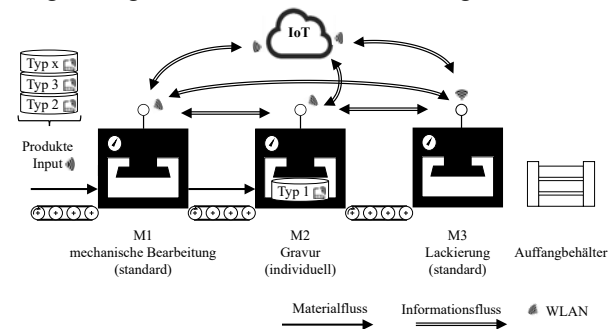


Abbildung 4 Schematischer Aufbau der teilautonomen I4.0-Fertigungsstraße.

Durch Kommunikation untereinander und einer übergeordneten Cloud im Sinne des IoT, werden Informationen über die zu fertigenden Werkstücke frühzeitig übermittelt. In Verbindung mit künstlicher Intelligenz kann der Wechsel des CNC-Codes frühzeitig initiiert werden, wodurch sich die Durchlaufzeit von M2 auf die tatsächliche Bearbeitungsdauer verkürzt (Abschnitt 4.2).

Die modellierten Fertigungsstraßen bilden die Grundlage für weitere Untersuchungen durch Simulation.

4 Simulation und Auswertung

Die in Kapitel 2 entwickelten Modelle stellen einen ersten Ansatz zur Simulation von Fertigungsstraßen dar. Zur Untersuchung der Auswirkungen auf die Durchlaufzeit der implementierten I4.0-Lösungen werden die in Kapitel 2.2 beschriebenen Fertigungsstraßen in zwei konkreten Anwendungsszenarien simuliert.

Um die Produktinformationen für die Simulation zu erzeugen, wird neben den beschriebenen Modulen ein Produktgenerator angelegt (Abbildung 4). Dieser erzeugt mit einer einstellbaren Frequenz zufällige Sensor- und Produkt-IDs verschiedener Produkttypen.

4.1 Fertigung eines Standardprodukts

Im ersten Szenario wird ein regulärer Maschinenablauf simuliert, bei dem jede Maschine mit der Bearbeitung eines neuen Teils beginnen kann, sobald sie den vorherigen Fertigungsprozess abgeschlossen hat. Die Simulationsparameter sind in Tabelle 1 zusammengefasst.

Tabelle 1: Simulationsparameter – Szenario 1 und 2.

Modul	Arbeitsschritt	Dauer [s]
M1	Mechanische Bearbeitung	120
M2	CNC-Code laden	60
M2	Gravur	60
M3	Lackierung	30
Förderband	Transportieren	10

Abbildung 5 zeigt das Simulationsergebnis der Standardproduktfertigung. In Abbildung 5a) ist das Verhalten der konventionellen Fertigungsstraße als Referenz dargestellt. Die Durchlaufzeit eines Werkstücks inklusive aller Transporte beträgt $T_K = 300s$.

Abbildung 5b) stellt die Simulationsergebnisse der teilautonomen Anlagen dar. Es ist zu erkennen, dass die Anpassung in M2 wie erwartet stets frühzeitig stattfindet, sodass sich die Durchlaufzeit jedes Werkstücks um $\Delta T_{abs} = 60s$ auf $T_{TA} = 240s$ verkürzt.

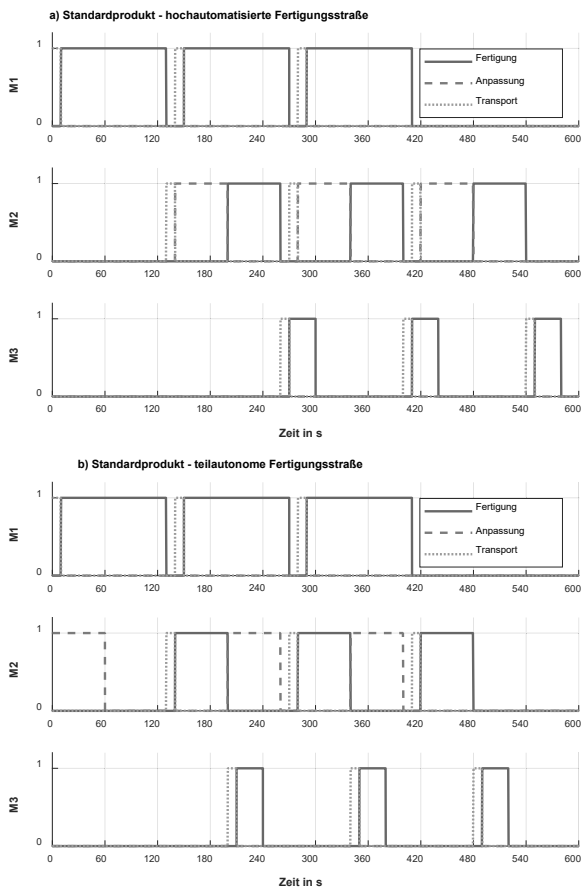


Abbildung 5: Simulationsergebnis des Produktionsablaufs in der Standardproduktfertigung.

Dadurch, dass ein neuer Fertigungszyklus in beiden Fällen immer zum gleichen Zeitpunkt beginnt, kann sich die Zeitersparnis mit der Stückzahl nicht aufsummieren. Die getroffene Maßnahme verliert folglich mit steigender Stückzahl n , gemessen an der relativen Zeitersparnis ΔT_{rel} , an Wirksamkeit.

$$\Delta T_{rel} = \frac{\Delta T_{abs}}{n \cdot T_K} \quad (1)$$

$$\lim_{n \rightarrow \infty} \Delta T_{rel} = 0 \quad (2)$$

4.2 Hochpräzise Fertigung

In diesem Szenario sollen auf M1 hochpräzise Bauteile gefertigt werden. Die Vibrationen der beiden anderen Maschinen beeinflussen dabei die Bauteilqualität negativ. Deshalb darf M1 nur dann fertigen, wenn sich alle anderen Module (M2, M3 und Förderbänder) im Ruhe- oder Anpassungszustand befinden. Es ergibt sich ein neuer Ablauf, bei dem erst mit einem erneuten Fertigungszyklus begonnen wird, sobald das Werkstück M3 passiert hat. Zur besseren Vergleichbarkeit bleiben die Simulationsparameter aus Tabelle 1 bestehen.

Auch in diesem Anwendungsfall benötigt ein Durchlauf der konventionellen Produktionsstraße $T_K = 300s$. Mangels Kommunikationsmöglichkeiten startet der Transport zu M1 erst nachdem M3 das Werkstück tatsächlich fertiggestellt hat (Abbildung 6a)). Abbildung 6b) illustriert die Ergebnisse der I4.0-Variante.

In diesem Fall kennt jede Maschine den Status der jeweils anderen. Dadurch wird ab dem zweiten Durchlauf der Transport des Halbzeugs zu M1 eingeleitet während M3 noch fertigt. M1 kann anschließend direkt mit der Fertigung beginnen. Zusätzlich wird die Anpassung des CNC-Programms von M2 wie erwartet vorgezogen, wodurch sich in diesem Beispiel der Hauptteil der Zeitreduzierung ergibt.

Der veränderte Ablauf des zweiten Szenarios führt zu einer Kumulierung der absoluten Zeitersparnis ΔT_{abs} . Im ersten Durchlauf beträgt die absolute Zeitersparnis $\Delta T_{abs,1} = 60s$, ab dem zweiten Werkstück $\Delta T_{abs,n>1} = 70s$. Daraus folgt für die relative Zeitersparnis:

$$\Delta T_{rel} = \frac{(n-1) \cdot \Delta T_{abs,n>1} + \Delta T_{abs,1}}{n \cdot T_K} \quad (3)$$

$$\lim_{n \rightarrow \infty} \Delta T_{rel} = \frac{\Delta T_{abs,n>1}}{T_K} \quad (4)$$

In diesem Anwendungsszenario ergibt sich nach Gleichung (4) eine relative Zeitersparnis $\Delta T_{rel} = 23,3\%$.

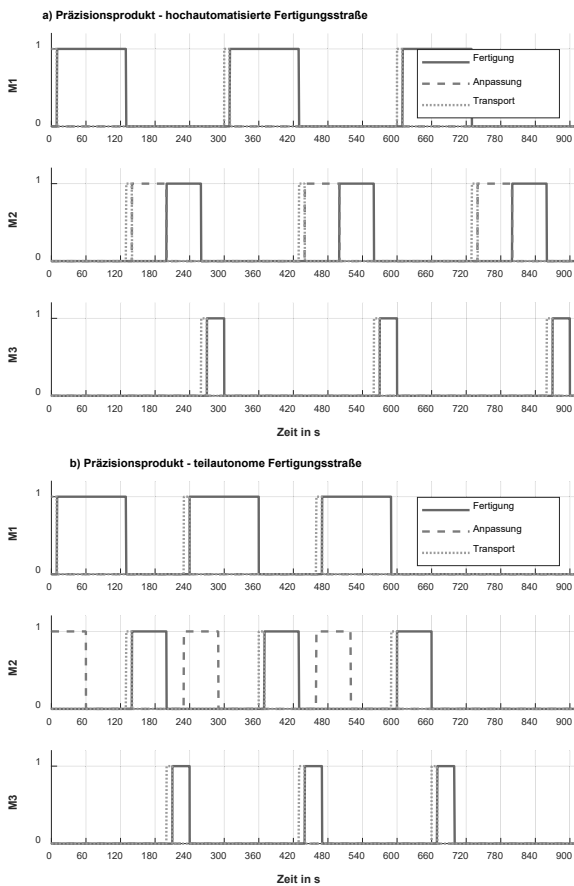


Abbildung 6: Simulationsergebnis des Produktionsablaufs in der Präzisionsproduktfertigung.

Obwohl sich die Standard- und Präzisionsfertigung nur durch eine kleine Änderung im Prozessablauf unterscheiden, hat die Nutzung derselben I4.0-Lösungen im zweiten Fall eine deutliche Produktivitätssteigerung ergeben. Ist die Durchlaufzeit das einzige Bewertungskriterium, liefert das erste Szenario keine Verbesserung, welche die Einführung der I4.0-Lösungen rechtfertigen. Für andere Bewertungskriterien kann sich ein Nutzen aus weiteren Vorteilen der I4.0 (Kapitel 3.2) ergeben.

5 Fazit und Ausblick

Der vorliegende Beitrag zeigt einen ersten modularen Modellierungsansatz für Produktionsstraßen mit und ohne I4.0-Lösungen zur Bewertung der getroffenen Maßnahmen im Rahmen der Forschungsprojekte MiMec und Synus.

Nach einer Beschreibung der zugrundeliegenden Problematik und des Wissensstandes folgte die Vorstel-

lung und Durchführung der Entwurfsmethodik der hierarchischen Strukturierung. Im Anschluss wurde die Modellbildung exemplarisch sowohl für ein Basismodul, als auch für eine gesamte Fertigungsstraße dargestellt.

Die grundverschiedenen Simulationsergebnisse der gewählten Beispiele zeigte die Vielfältigkeit und Komplexität moderner Produktionssysteme. Sie heben die Herausforderung bei der Auswahl geeigneter I4.0-Lösungen für jede individuelle Produktionsanlage hervor.

Die nächsten Arbeitsschritte umfassen unter anderem die Erweiterung der Modellbibliothek sowie die modellgestützte Untersuchung weiterer Bewertungskriterien.

6 Danksagung

Dieser Beitrag wurde im Rahmen des Teilprojekts *Modellbasierte Konzeption und Bewertung von Industrie 4.0-Lösungen zur Vernetzung mechatronischer Komponenten in Produktionsanlagen durch Digitalisierung (Mi-Mec)* des Verbundprojekts *Methoden und Werkzeuge für die synergetische Konzipierung und Bewertung von Industrie 4.0-Lösungen (Synus)* durch den Europäischen Fonds für regionale Entwicklung (EFRE) unter dem Förderkennzeichen ZW 6 85012454 gefördert. Die Verantwortung für den Inhalt liegt bei den Autoren.



EUROPÄISCHE UNION
Europäischer Fonds für
regionale Entwicklung



Literatur

- [1] Promotorengruppe Kommunikation der Forschungsunion Wirtschaft - Wissenschaft. *Umsetzungsempfehlungen für das Zukunftsprojekt Industrie 4.0*. Abschlussbericht. 2013
- [2] Audi AG. *Audi Geschäftsbericht 2014*. Ingolstadt; 2014. S. 34-40
- [3] Jaspersmitte, J. *Was hinter Begriffen wie Industrie 4.0 steckt*. Computer & Automation. 2012
- [4] McKinsey&Company. *Industry 4.0. How to navigate digitization of the manufacturing sector*. 2015
- [5] Liu-Henke, X. *Mechatronische Entwicklung der aktiven Feder-/Neigetechnik für das Schienenfahrzeug RailCab*. VDI-Fortschritt-Berichte, Reihe 12, Nr. 589, VDI-Verlage. Düsseldorf, 2004

Development of a Modular Configuration Framework for Digital Twins in Virtual Testbeds

Ulrich Dahmen^{1*}, Tobias Osterloh¹, Jürgen Roßmann¹

¹Institute for Man-Machine Interaction (MMI), RWTH Aachen University, Ahornstraße 55, 52074 Aachen, Germany; *dahmen@mmi.rwth-aachen.de

Abstract. The tremendous progress in computer technologies within the last decades enabled an increasing use and importance of modeling and simulation for all engineering and decision-making processes. However, the inherent complexity of mechatronic systems increases significantly, causing a steep rise in complexity of the corresponding simulation models. This paper introduces a concept for a modular configuration framework on basis of Digital Twins in Virtual Testbeds to handle the complexity of creating reliable system simulations.

Introduction

The steep rise in interconnection and multidisciplinary leads to a non-linear increase in the level of complexity of modern technical systems. All engineers have to manage this ever-increasing inherent complexity regarding design, production, and timely marketing. It is therefore necessary to understand all individual components, and the overall system itself at all stages of the development process. Since modeling and simulation allows earlier testing, verification and validation and is now widely accepted as the third pillar of science and engineering along with theory and experiment [1], its use and importance for all engineering and decision-making processes increases as well. Although final validation can only be achieved by acceptance tests on the real product, a significant part of the test results can be obtained in advance using comprehensive system simulations.

First attempts to provide a flexible and expandable environment for the simulation of complex systems were made by the eRobotics community, providing software frameworks with various interfaces for the integration of arbitrary simulation and control algorithms [2]. Modern simulation infrastructures (like BASILES [3] and VTi [4]) integrate various algorithms to a comprehensive overall system simulator, trying to reduce the amount of required tools. These kind of simulation infrastructures are called Virtual Testbeds (VTB).

At the same time, several approaches (especially in the context of Industry 4.0) try to mirror all relevant aspects of a real-world asset into a virtual substitute summarized by the term Digital Twin (DT). Digital Twins are virtual images of real world assets, describing their physical properties, behavior, as well as their communication interfaces to other Digital Twins or real-world assets [5].

However, modeling and simulating complex systems to support system design and realization remain challenging tasks, since complex systems require complex simulation models. Consequently, the need for sophisticated methodologies in order to handle the complexity of creating reliable system simulations persists. For that reason this paper introduces a concept for a modular configuration framework on basis of Digital Twins in Virtual Testbeds. It helps to structure and configure complex virtual experiments depending on corresponding simulation purposes to satisfy specific information needs. Thus, the simulation model becomes adjustable and scalable which is a central aspect of the presented approach. The following sections will present the configuration formalism for block-oriented simulations in detail and discuss the benefits of this approach. Afterwards an application example, based on a case study from a space project is provided, followed by a summary and outlines for future work at the end of the paper.

1 Concept of the Modular Configuration Framework

1.1 Overview

The great challenge with comprehensive and cross-domain system simulations is the evolving complexity of the simulation model itself. Collecting engineering data and simulation algorithms forming large models

without consistent structures to manage relations between different simulation domains and without considering different and usually over time changing level of detail and integration makes it nearly impossible to apply, maintain, extend, reuse and verify those simulation models and its results. This is the starting point for the development of the configuration framework that should formalize and structure the modeling process for complex systems to handle the complexity of cross-domain system simulation models.

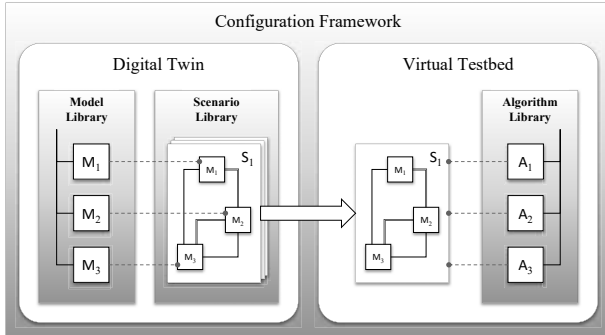


Figure 1: Basic structure of the configurable framework, based on Digital Twins and Virtual Testbeds

As illustrated in figure 1 the concept distinguishes between two main entities: The Digital Twin of the specific system of interest (SoI) and a suitable Virtual Testbed. The Digital Twin provides a collection of block-oriented model elements forming the DT's Model Library, and a collection of simulation scenarios collected in the DT's Scenario Library. Each of these scenarios can be executed ("running the simulation") inside the Virtual Testbed that provides the Algorithm Library, containing all relevant simulation algorithms that can be applied to the Digital Twin. This basic structure is extended by formalized processes that help to specify, execute and verify various system simulations. The next sections will present the elements more in detail.

1.2 The Model Library

The Model Library is hierarchically structured and mirrors the system's physical structure (representation of the structural decomposition respectively the product tree, combined with the system context) derived from the system model (part of Systems Engineering). It contains basic *model elements* M_i on each hierarchical layer of the structural system decomposition (from overall system, subsystems, assemblies to components). Each

model element provides a set of static element properties m_1, \dots, m_i , derived from corresponding physical properties (e.g. masses, CAD geometry data, etc.) in the part specifications inside the product model (part of Domain Engineering)¹.

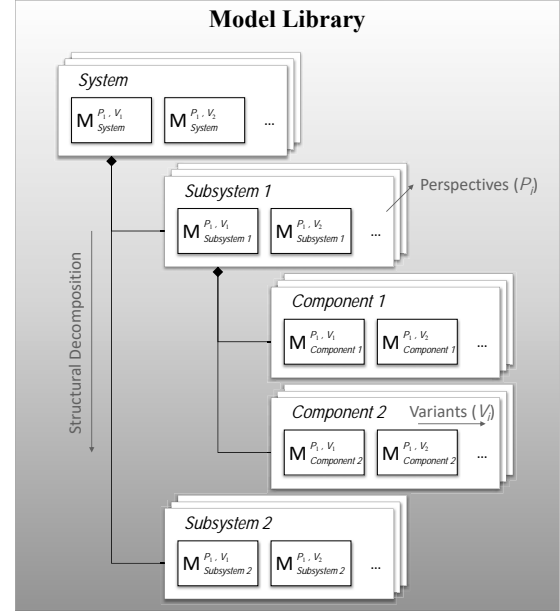


Figure 2: Detailed structure of the DT's Model Library

As illustrated in figure 2 each layer of the Model Library can provide different types of the same model element, considering different simulation aspects (also called simulation domains, see section 1.4). Those types form different *perspectives* on the system and refer to corresponding simulation domains provided by the Algorithm Library. For example:

- $M_{Component2}^{Perspective1} = M_{Component2}^{RigidBodyDynamics} \rightarrow \text{rb-substitute}^2$
- $M_{Component2}^{Perspective2} = M_{Component2}^{ThermalRadiance} \rightarrow \text{nodal model}$
- $M_{Component2}^{Perspective3} = M_{Component2}^{SensorSimulation} \rightarrow \text{mech. CAD}$

Thus, a specific model element contains all static parameters that properly abstract the corresponding system element from the perspective of the associated simulation domain (a concrete example is presented in section 1.6).

¹The indicated relations to other engineering disciplines like (Model-based) Systems Engineering and Domain Engineering base on a corresponding classification approach presented in [6].

²Abbreviation for rigid body substitute.

Furthermore, each system element may contain several different implementations of the same model type, defined as model *variants*. These model variants allow to model the system with different level of detail, which is necessary for the application of the Digital Twin during the whole development process because the appropriate level of detail depends on the current progress of the project and additionally on the structure of the envisaged simulation scenario³. Figure 3 illustrates an example for a component of the system from the application example in section 2:

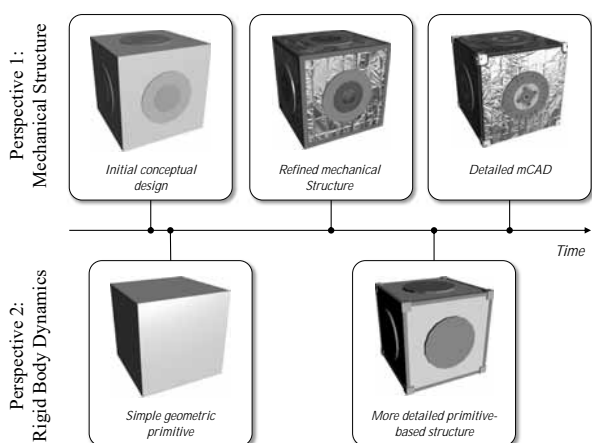


Figure 3: Model refinement with model variants

The lower part presents two variants of the model element for the system perspective "rigid body dynamics" with different levels of detail. The upper part illustrates three variants of the same model element (likewise with an increasing level of detail), but for a different system perspective. The time line indicates how the model refinement follows the growth of detail knowledge regarding the component. It shows further that model variants from different perspectives may evolve with different speed.

1.3 The Scenario Library

In contrast to the Model Library the Scenario Library contains specific simulation experiments, which are meaningfully executable simulation models build up from the model elements of the Model Library. A specific *simulation scenario* S_i of the Scenario Library can be designed with the help of a formalized process: According to a specific question that should be answered

³Sometimes a model with a lower level of detail is more suitable for a specific simulation than one with a higher level.

by simulation, the simulation engineer defines an appropriate scenario using a *configuration table* as presented in figure 4. The first column contains the system's structural decomposition, while the following columns offer system perspectives (according to simulation domains available in the Virtual Testbed). Each table cell allows to either select an existing model element of the DT's Model Library (according to the corresponding domain of its column) or to leave it unselected ("not assigned"). If there is no selectable model element for a specific module and perspective it can be an indication of a lack in the Model Library. But there are many cases where entries intentionally remain empty (especially in early project phases). The availability of model elements from different hierarchical layers now allows to create simulation scenarios with different levels of integration.

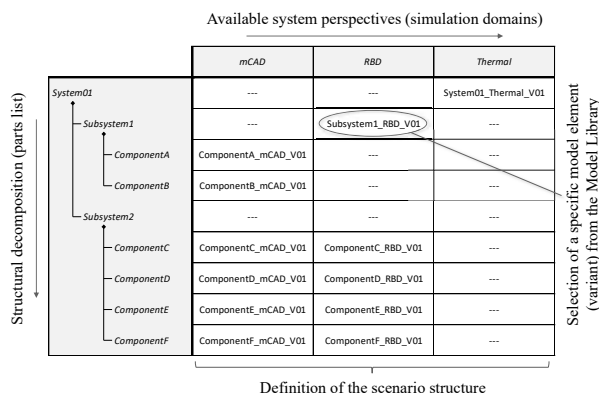


Figure 4: Configuration table for specifying a specific simulation scenario

Based on the configuration table that specifies which model elements compose the simulation scenario, an *interconnection table* as presented in figure 5 defines relations between those model elements. Each row and column of the interconnection table refers to a specific model element selected in the configuration table, forming a quadratic matrix. Thus, a specific cell entry defines and parameterizes an interconnection C_i between the corresponding model elements⁴. All of these interconnections can be classified as *inner-domain* or *inter-domain interconnection*. The former describe connections between model elements that refer to the same simulation domain and the latter between model elements from different simulation domains.

⁴Note: The interconnection table only specifies *explicit* interconnections, *implicit* interconnections will be presented in section 1.4.

The arrangement of rows and columns in the interconnection table causes a triangular structure where the lower half can be omitted for redundancy reasons (grayed out in figure 5). To handle larger systems with a great number of model elements it is recommendable to divide the interconnection table into subtables according to the corresponding simulation algorithms. Those subtables that form the leading diagonal (colored yellow) contain inner-domain interconnections, and the remaining subtables (green ones) contain inter-domain interconnections.

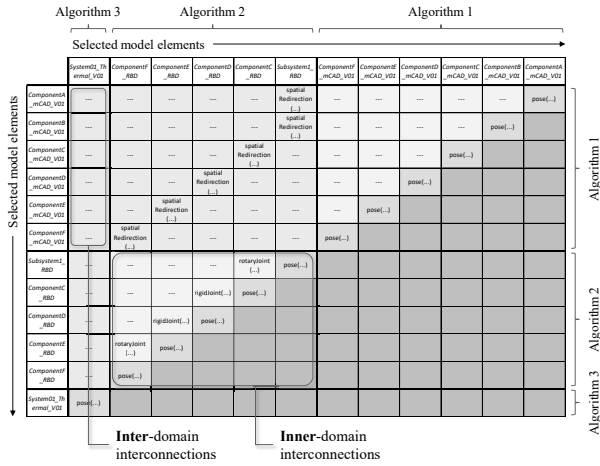


Figure 5: Interconnection table for specifying a specific simulation scenario

The cells of the leading diagonal inside each inner-domain subtable (colored in a darker yellow) are special since they map model elements to themselves. These cells are used to set the initial state \vec{x}_0 of the corresponding model element (necessary for solving the scenario as initial value problem). Thus, the configuration table and the interconnection table define the scenario S as a set of involved model elements, explicit interconnections, as well as their initial state. This way the scenario becomes the simulation's conceptual model and it becomes possible to automatically generate the executable simulation code for a suitable simulation tool.

1.4 The Algorithm Library

Finally, the Algorithm Library provides *simulation algorithms* A_i . As already mentioned a specific simulation scenario (to be exact the generated scenario model code) can be executed ("running the simulation") inside the Virtual Testbed by applying the corresponding sim-

ulation algorithms. Therefore, the Virtual Testbed provides a runtime environment that loads the model code into a comprehensive simulation database and schedules the application of the simulation algorithms. At each simulation step the simulation algorithms analyze the current state $\vec{x}(t)$, that provides all dynamic properties of the model elements and interconnections, and calculate the consequent state $\vec{x}(t + dt)$ considering static properties given by the scenario S , and the current time t (see equation (1)). Therefore, a simulation algorithm may describe a differential equation system, but does not necessarily need to.

$$\vec{x}(t + dt) = A_i(S, \vec{x}(t), t) \quad (1)$$

Typically, an algorithm A_i executes a specific *simulation domain* referring to a corresponding system perspective (e.g. i equals the rigid body dynamics that manages the dynamic properties of rigid body substitute model elements). But there are also simulation algorithms that consider multiple perspectives. Those algorithms usually realize inter-domain interconnections.

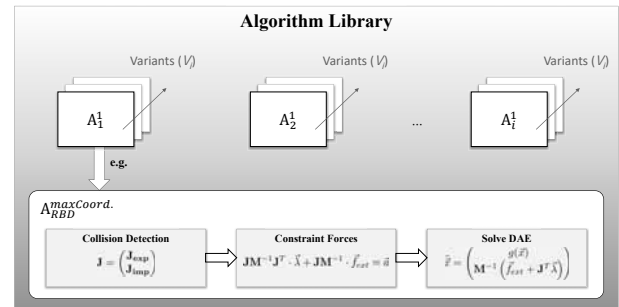


Figure 6: Schematic illustration of the Algorithm Library

For practical reasons sometimes it is recommended to automate special kinds of interconnections. An example from the rigid body dynamics simulation domain is the management of collisions between rigid bodies. To allow two arbitrary rigid bodies to collide with each other, an interconnection between the corresponding model elements has to be defined. The effort to model all these collision interconnections ("everyone to everyone") quickly becomes not manageable, especially in case of large scenarios with a multitude of separate model elements. For that reason the corresponding simulation algorithm (in the given example the rigid body dynamics algorithm) is extended by a collision handler that automatically detects and resolves collisions between any rigid bodies in the scenario. Doing this, the

collision becomes an *implicit* interconnection that does not have to be modeled explicitly within the scenario. As a consequence the formalism distinguishes between *explicit* interconnections C_{exp} (compare section 1.3) and *implicit* interconnections C_{imp} .

Analogous to the model elements in the Model Library each simulation algorithm may exist in different *variants*. For example the variants A_i^j of the rigid body dynamics simulation algorithm could be a maximal and minimal coordinate formulation.

$$A_i \in \{A_i^1, \dots, A_i^j\} \quad (2)$$

1.5 Execution of a Specific Scenario

The last step in the configuration process is the selection and execution of a specific scenario S from the Scenario Library. Additionally, the Virtual Testbed has to be configured for the execution of the scenario S . Therefore, all required simulation algorithms A_i have to be selected from the available variants of the simulation algorithms. Furthermore, each algorithm A_i has to be configured regarding the requirements of the chosen scenario S . In general, these configuration capabilities are any properties or settings the simulation algorithms provides, e.g. the configuration of the integrator, selection of solvers or selection of numerical stabilization techniques.

To get the heart of this, a closer look at figure 1 clarifies that the Digital Twin specifies *what* has to be simulated and the configuration of the Virtual Testbed defines *how* it has to be simulated.

1.6 Academic Example for the Formalism

To demonstrate the basics of the proposed formalism a simple example with only one perspective should be presented here. A good example is a rigid body dynamics simulation of a rod pendulum. A rigid body i is described by its model M_i . The model of the rigid body i itself consist of the center of mass \vec{c}_i , its inertia tensor θ_i and its mass m_i . Additionally, for interactive rigid body dynamics simulations the model M_i is complemented by a collision geometry G_i . All quantities are given regarding a reference frame \mathbf{R}_i .

$$M_i = \{\mathbf{R}_i, \vec{c}_i, \theta_i, m_i, G_i\} \quad (3)$$

The state \vec{x} of a rigid body i is described by its position \vec{p}_i , its orientation \vec{q}_i (as quaternion), its linear velocity \vec{v}_i and its angular velocity $\vec{\omega}_i$.

$$\vec{x}_i = (\vec{p}_i, \vec{q}_i, \vec{v}_i, \vec{\omega}_i)^T \quad (4)$$

Rigid bodies can be coupled explicitly, describing kinematic constraints between two rigid bodies. For maximal coordinate rigid body simulation, these constraints can be described by a Jacobian $\mathbf{J}_{exp,i,j}$. A rigid connection between two rigid bodies i and j is realized by an entry $\mathbf{J}_{i,j}$ in the Jacobian \mathbf{J}_{exp} [7]. Thereby, an explicit connection between the models M_i and M_j is realized.

$$\mathbf{J}_{exp,i,j} = \begin{pmatrix} \mathbf{1}_{3 \times 3} & \mathbf{R}_{\times,i} & \cdots & -\mathbf{1}_{3 \times 3} & -\mathbf{R}_{\times,j} \\ \mathbf{0}_{2 \times 3} & \mathbf{E}_i & \cdots & \mathbf{0}_{2 \times 3} & -\mathbf{E}_j \end{pmatrix} \quad (5)$$

So far, the simulation engineer specifies the explicit interconnections between the models M_i . The interactive behavior is covered by implicit interconnections (e.g. contacts or friction between rigid bodies). Analogous to the explicit interconnections the implicit interconnections can also be formulated on a constrained basis \mathbf{J}_{imp} (a detailed formulation of equation (5) and (6) is given in [7]).

$$\mathbf{J}_{imp,i,j} = \begin{pmatrix} -\vec{n}^T & (\vec{n} \times \vec{r}_i)^T & \cdots & \vec{n}^T & -(\vec{n} \times \vec{r}_j)^T \end{pmatrix} \quad (6)$$

Based on this formalism, a scenario S can be composed from various models M_i , like depicted in figure 7 (each color encodes a single model M_i). The connections within the scenario S are described by the explicit connections C_{exp} realized by the Jacobian \mathbf{J}_{exp} .

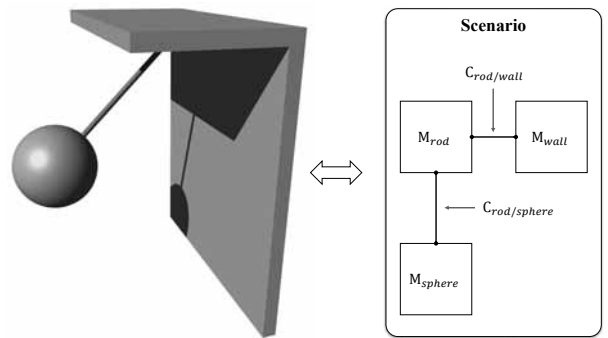


Figure 7: Academic example of a rod pendulum, illustrating the formalism

$$S = \{M_{wall}, M_{rod}, M_{sphere}, C_{rod/sphere}, C_{rod/wall}\} \quad (7)$$

The initial configuration of the composed model is described by the state vector \vec{x} .

$$\vec{x}_0 = (\vec{x}_{wall}, \vec{x}_{rod}, \vec{x}_{sphere})^T \quad (8)$$

In order to execute the scenario, a simulation algorithm $A_{RBD}^{maxCoordinates}$ is applied. The simulation algorithm can be divided into three parts:

1. Calculation of the Jacobian \mathbf{J} from the explicit interconnections \mathbf{J}_{exp} and implicit interconnections \mathbf{J}_{imp} . This requires a collision detection algorithm.

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}_{exp} \\ \mathbf{J}_{imp} \end{pmatrix} \quad (9)$$

2. Calculation of the constrained forces $\vec{\lambda}$ enforcing the formulated constraints \mathbf{J} , by solving a Linear Complementarity Problem (LCP) [8].

$$\begin{aligned} \mathbf{J} \cdot \mathbf{M}^{-1} \cdot \mathbf{J}^T \cdot \vec{\lambda} + \mathbf{J} \cdot \mathbf{M}^{-1} \cdot \vec{f}_{ext} &= \vec{a} \\ \text{with } \vec{\lambda}, \vec{a} &\geq \vec{0}, \quad \lambda_i a_i = 0 \quad \forall i \end{aligned} \quad (10)$$

3. Solving the differential equation $f(\vec{x}, t)$, where the function $g(\vec{x})$ gives the derivative of the position and orientation from the current state vector \vec{x} [7].

$$\ddot{\vec{x}} = f(\vec{x}, t) = \begin{pmatrix} g(\vec{x}) \\ \mathbf{M}^{-1} \cdot (\vec{f}_{ext} + \mathbf{J}^T \vec{\lambda}) \end{pmatrix} \quad (11)$$

An analysis of the execution of the example scenario S is shown in figure 8. The pendulum moves according to its explicit interconnection. The spikes in the deflection indicate the contact of the sphere with the wall, constituting the implicit interconnection, being realized by the simulation algorithm $A_{RBD}^{maxCoordinates}$.

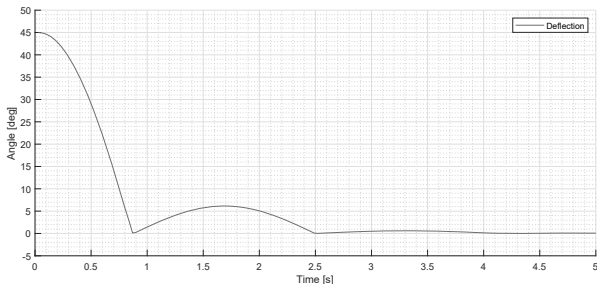


Figure 8: Analysis of pendulum example

2 Application Example

This section presents a more complex example for the application of the introduced concept in the context of the emerging technology of modular satellite systems. Since traditional satellite development focuses on unique solutions with high development costs and long lead periods, novel approaches try to shift those monolithic architectures to a modular design with standardized and reusable building blocks (BB).

The following examples base on the upcoming iSAT-1 space mission, that should combine a modular satellite platform with a tracking payload to demonstrate the operability of modular satellites and extend the operational small object tracking service from the international cooperative for animal research using space (ICARUS) [9]. During an initial ECSS Phase 0 study, a Digital Twin was developed to simulate and investigate early system designs in various operational scenarios.

The upper part of figure 9 shows an excerpt of the satellite's structural decomposition (from overall-system to sub-assembly level), created during system design. Accordingly, the iSAT-1 consists of 11 building blocks (BB1 to BB11), two customized compartments, the ICARUS payload antenna and three solar arrays covering the satellite's energy demand. To setup the Digital Twin of the iSAT-1 system, initial model elements for the mentioned assemblies and sub-assemblies with respect to simulation domains that are relevant at this project stage were created and added to the Model Library (see figure 9).

In the next step a specific simulation scenario was defined to investigate the influence of the mission orbit on the space craft's thermal budget⁵. As illustrated in figure 10 model elements from two perspectives are used⁶: rigid body dynamics (covering the orbital movement) and thermal radiance (covering the thermal input and exchange). Each model element is assigned with initial values based on the test settings. These include values for the initial momentum of the rigid body substitutes (defining the mission orbit) and values

⁵In case of a satellite with a modular system bus, it becomes more complex to dimension the thermal budget because of the need to exchange and transport thermal loads through individual modules.

⁶For a better vividness a graphical representation of the underlying configuration and interconnection table is shown here.

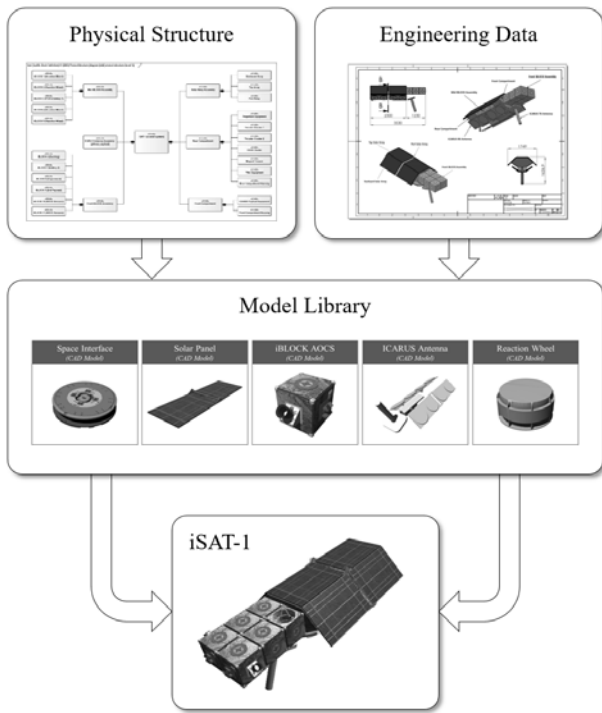


Figure 9: Set up of the DT's Model Library, based on the system break down structure and product data

for the initial temperature of the building blocks. Inside the rigid body perspective the model elements are connected by joints (explicit inner-domain interconnections, blue lines), according to the system layout specification derived from the system specification (see figure 9). Equally, thermal junctions between model elements of the thermal radiance perspective define the satellite's thermal network (orange lines). Within both perspectives the resulting system can be visualized as 3D model (see figure 10). Finally, the cross-domain couplings between orbit and thermal budget are realized by explicit inter-domain interconnection between model elements from different perspectives but same hierarchical layer (green lines)⁷.

Based on the complete scenario definition (given by the filled configuration and interconnection table) the executable simulation code was created. The next step was the configuration of the Virtual Testbed. During this step appropriate variants of the involved simula-

⁷In the case at issue this inter-domain interconnection describes a spatial redirection transformation. As a result the substitute geometries of the thermal model elements will follow the movement of the rigid body substitutes and thus be exposed to changing conditions of radiation, according to the orbital movement.

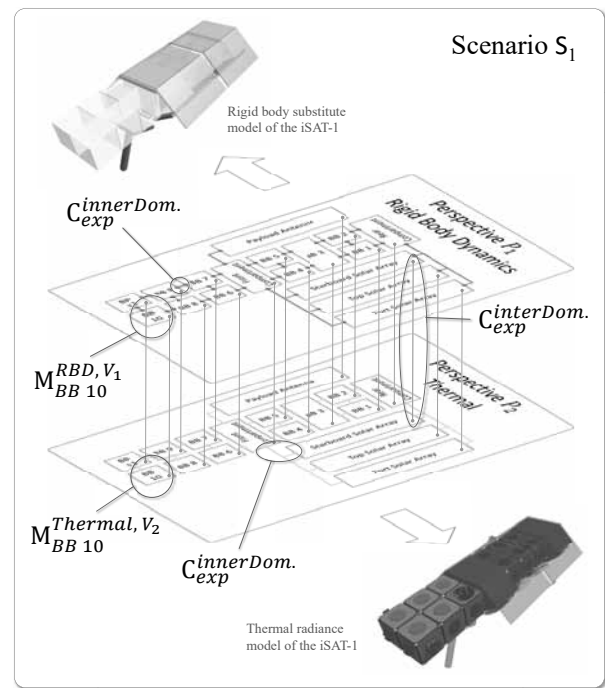


Figure 10: Visualization of the scenario definition with all explicit interconnections

tion algorithms were selected and parameterized. For the presented scenario the following algorithm variants were selected:

- For the execution of the rigid body dynamics perspective a rigid body dynamics algorithm in maximal coordinates $A_{RBD}^{maxCoordinates, newtonGravitation}$ is selected. Additionally, the algorithm takes Newton's gravitational law into account, enabling the simulation of orbital dynamics.
- For the execution of the thermal perspective a rendering-based algorithm is applied to determine the radiance on the satellite's surface. This radiance calculation is complemented by a thermal nodal model analyzer, simulating the thermal flow inside the satellite.

Within this configuration suitable settings for the execution of the selected simulation algorithms were defined. These settings include among others the integration step size dt and parameters for the numerical stabilization. For the thermal simulation, a reference date was declared, providing the base for the calculation of the relative earth-sun position by the thermal simulation

algorithm. Finally, after all configurations were performed the scenario was executed and the gathered data was analyzed. Figure 11 exemplarily shows the execution of an additional simulation scenario, that includes a perspective with detailed mechanical CAD geometries for a camera-based sensor simulation.

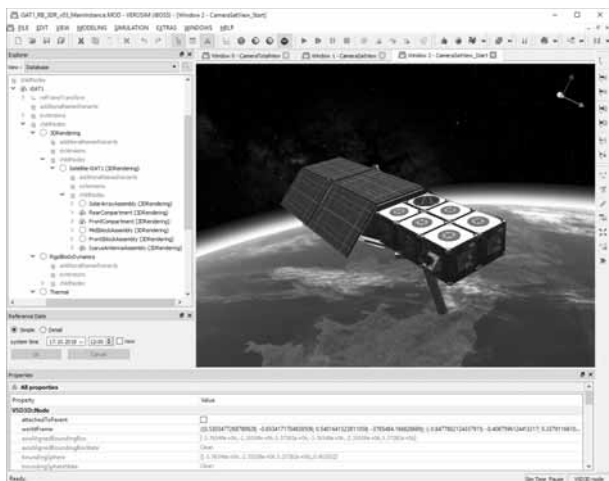


Figure 11: Execution of a specific simulation scenario inside the Virtual Testbed

3 Summary and Outlook

This paper introduces a concept for a modular configuration framework that helps to structure and configure complex virtual experiments. The basis are Digital Twins that specify *what* to simulate, and Virtual Testbeds defining *how* to simulate. The developed formalism manages different simulation aspects in a flexible way and enables the application of formalized processes that help to specify (configure), execute and verify specific simulations. The concept makes the simulation model adjustable and scalable in order to satisfy specific information needs.

Up to now the developed concept focuses on the structural view on passive systems. To take behavior aspects of an actuated system into account, it is necessary to extend the formalism by the integration of an operational view. Due to its structured design the concept allows the development and application of formalized verification and validation processes on different layers (model elements, scenario structure, scenario execution), which is one of the next development steps.

Furthermore, the parameterization of algorithms for the execution of a specific scenario will be detailed in the future, too.

Acknowledgement

This work is part of the project "ViTOS-II", supported by the German Aerospace Center (DLR) with funds of the German Federal Ministry of Economics and Technology (BMWi), support code 50 RA 1810.

References

- [1] Oberkampf W, Roy C. *Verification and validation in scientific computing*. Cambridge: Cambridge Univ. Press. 2012.
- [2] Rossmann J, Schluse M. Virtual Robotic Testbeds: A Foundation for e-Robotics in Space, in Industry – and in the Woods. In: *Proceedings of the 4th International Conference on Developments in eSystems Engineering (DeSE)*, edited by Saad, Abir, et al. The British University in Dubai, UAE. 2011; pp. 496–501.
- [3] Salas Solano S, et al. BASILES: A common simulation platform to promote models and simulation reuse. In: *Proceedings of 13th International Conference on Space Operations*. Pasadena, California, United States of America. 2014; .
- [4] Rast M, Kupetz A, Schluse M, Stern O, Rossmann J. Virtual Testbed for Development, Test and Validation of Modular Satellites. In: *The 13th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2016)*. Beijing. 2016; .
- [5] Grieves M. Digital Twin: Manufacturing Excellence through Virtual Factory Replication. 2014.
- [6] Dahmen U, Rossmann J. Experimentable Digital Twins for a Modeling and Simulation-based Engineering Approach. In: *4th IEEE International Symposium on Systems Engineering (ISSE 2018), October 1-3, Rome, Italy*. IEEE. 2018; .
- [7] Jung T. Methoden der Mehrkörperdynamiksimulation als Grundlage realitätsnaher Virtueller Welten. Ph.D. thesis, RWTH Aachen University, Institute for Man-Machine-Interaction. 2011.
- [8] Baraff D. Linear-time dynamics using Lagrange multipliers. In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM. 1996; pp. 137–146.
- [9] Wikelski M, Müller U, Naumann W. Das satellitenbasierte ICARUS-Projekt: Ein neues globales Tierbeobachtungssystem. In: *Der Flake*. 2015; .

Der digitale Zwilling - Anwenderbeispiel einer Physics-Based-Simulation in der Produktion

Ein Montagesystem simuliert und programmiert mit Siemens PLM Software

Nanno Peters^{1*}, Udo Triltsch¹, Dennis Rein¹

¹Ostfalia Hochschule für angewandte Wissenschaften; Maschinenbau / Institut für Produktionstechnik; Salzdhalmmer Straße 46/48; 38302 Wolfenbüttel; n.peters@ostfalia.de, u.triltsch@ostfalia.de

Zusammenfassung. Dieser Artikel beschreibt den, mit Siemens PLM (Produktlebenszyklusmanagement) [1, 2] Software umgesetzten, digitalen Zwilling eines Montagesystems als ein Teil einer digitalen Fabrik. Ziel ist es, einen Überblick zu geben, die Stärken und Schwächen dieses Entwicklungsprozesses mit dem Mechatronic-Concept-Designer MCD (Teil von Siemens NX) aufzuzeigen und die hierfür benötigten Softwareteile zu beschreiben und zu anderen Tools abzugrenzen. Es werden einige Randbedingungen des digitalen Zwillinges aufgezeigt, welche für einen produktiven Einsatz vorliegen müssen.

1 Einleitung

Komplexer werdende Systeme und der Zwang, Entwicklungszeiten zu verkürzen, verlangen eine neue Art von parallelierter Planung und Entwicklung. Um dies zu ermöglichen, werden einzelne, früher getrennte, Softwaretools immer mehr vernetzt. Dies geschieht nicht nur auf der kaufmännischen, sondern auch auf der funktionalen und simulationstechnischen Ebene. So besitzt heutzutage jedes professionelle CAD-Programm (Computer-Aided Design) eine Möglichkeit, Finite-Elemente-Methoden (FEM) oder andere CAE-Tools (Computer-Aided Engineering) relativ problemlos einzubinden. Standard ist seit Jahren beispielsweise eine integrierte CAM-Schnittstelle (Computer-Aided Manufacturing) für eine nachgeordnete Fertigung, neuerdings auch bei additiven Fertigungsverfahren.

Die Firma Siemens setzt u.a. auf eine beschleunigte virtuelle Inbetriebnahme des Produkts durch einen digitalen Zwilling im Entwicklungsprozess (Abbildung 1). Der digitale Zwilling einer Produktionsanlage ermöglicht u.a. eine schneller realisierbare Prozessneugestaltung, die Möglichkeit einer Kollisionserkennung im Modell, eine vereinfachte SPS-Programmierung, die Minimierung des

Risikos beim Produktionsanlauf, einen schnelleren Vergleich verschiedener Konzepte und eine gute Visualisierung des Prozesses für den Vertrieb.

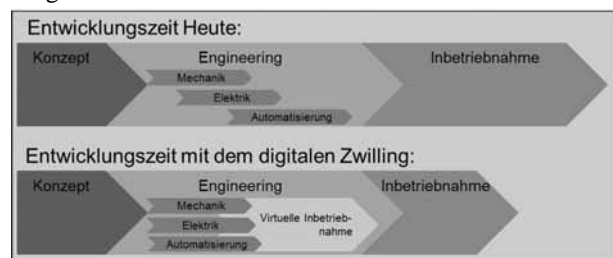


Abbildung 1: Siemens primäre Zielbeschreibung: Reduzierung der Entwicklungszeit durch Nutzung eines digitalen Zwillinges

An dieser Stelle sei ein Projekt der Firma Festo genannt, bei der 30% Entwicklungszeit bei einer Abfüllanlage eingespart wurde [3].

Ermöglicht werden soll dies mit dem Mechatronic-Concept-Designer (MCD), welcher voll in die NX-CAD Umgebung integriert ist. Grundlage für die kinematisch-physikalische Simulation von starren Körpern mit Schwerkraft, Reibung und diskrete und stetige Kollisionserkennung ist die Bullet-Physik-Engine, eine Open-Source-Software [4]. Zusätzlich ermöglicht der MCD eine virtuelle Inbetriebnahme von Systemen einschließlich der Motoren, Aktoren und Sensoren. Dies soll beispielhaft in dieser Veröffentlichung an einem Montagesystem gezeigt werden.

2 Montagesystem am IPT

Das Institut für Produktionstechnik der Ostfalia Hochschule besitzt ein Montagesystem für Lehrzwecke im Bereich der Handhabungs- und Montagetechnik. Die Abbildung 2 zeigt schematisch den Aufbau dieser Anlage, nicht dargestellt ist die übergeordnete Siemens S7-1500 Steueranlage. Es handelt sich dabei um ein Umlaufsystem von Werkstückträgern auf sieben Transportbändern der Firma Bosch aus u.a. sieben Stoppern, sechs Druckluft-Hubtischen, 16 Positionssensoren, sechs

RFID-Schreib-/Lesegeräten, zwei angebunden Robotern und einem Touchmonitor. Die Positionssensoren erkennen die Anwesenheit von Werkstückträgern an den jeweiligen Einbaustellen. Die Stopper und Hubtische werden je nach Fertigungsprogramm angesteuert/freigegeben.

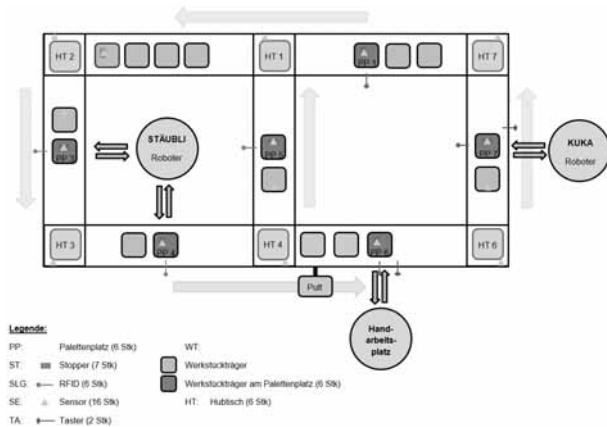


Abbildung 2: Schematische Darstellung des Montagesystems



Abbildung 3: Anlagenfoto (vorne Hubtisch 6/Hubtisch 7)

3 Erstellung des Simulationsmodells

Um das in Kapitel 2 beschriebene reale System zu einem digitalen Zwilling umzuwandeln ist es zuerst notwendig, jede einzelne Komponente des Systems zu konstruieren bzw. bestehenden CAD-Dateien in das NX-CAD-Format (.prt) zu überführen. Die fertige Montagelinie besteht aus über 600 Einzelteilen und ca. 200 Verknüpfungen.



Abbildung 4: CAD-Verhaltensmodell des Montagesystems

Teile, welche in der Simulation einen Schwerkrafteinfluss aufweisen sollen, müssen innerhalb des MCD als Starrkörper definiert werden. Für eine Interaktion von Teilen untereinander müssen diese zusätzlich als Kollisionsobjekt definiert werden. Die möglichen Kollisionskörperarten finden sich aufgelistet in Tabelle 1. Ein Kollisionskörper kann normalerweise den anderen nicht durchdringen. Die Kollisionseigenschaften können im einzelnen Bauteil. Sowie auch in der Baugruppe definiert werden. So können Baugruppen oder Teile später leicht wiederverwendet werden. Auch können über ein Kollisionsmaterial die Eigenschaften wie dynamische Reibung, Haftreibung, Rollreibung und Stoßzahl festgelegt werden.

Typ	Geometrische Genauigkeit/ Simulationsperformance	Zuverlässigkeit
Quader	Niedrig / Hoch	Hoch
Kugel	Niedrig / Hoch	Hoch
Zylinder	Niedrig / Hoch	Hoch
Kapsel	Niedrig / Hoch	Hoch
Konvex	Mittel / Mittel	Hoch
Multi-Konvex	Mittel / Mittel	Hoch
Netz	Hoch / Niedrig	Niedrig

Tabelle 1: Performance der Kollisionskörperarten in MCD [5]



Abbildung 5: drei mögliche Kollisionskörper am Werkstückträgermodell

Beispielhaft ist eine mögliche Auswahl des Kollisionskörpers am Werkstückträger in Abbildung 5 gezeigt. Für die Systemsimulation besitzt der Träger aber sechs Kollisionskörper (Abbildung 6). Vier Quader als Kollisionskörper an der Unterseite, zwei Quader für das Oberteil des Trägers. Damit werden beispielsweise die Stoppfunktion und die Führung auf den Laufbändern realisiert. Die Transportbänder, die Stopper, die Hubtische und die Führung besitzen ebenfalls passende Kollisionskörper.

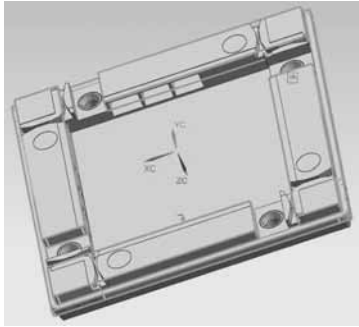


Abbildung 6: Kollisionkörper an der Unterseite des Werkstückträgers

Die Abbildung 7 zeigt die Übersicht der primären Funktionen im MCD. Die realen Positionssensoren in der Anlage werden mit Kollisionssensoren eingebunden. Die einzelnen Förderbänder sind über Geschwindigkeitsregler und die Hubtische über Positionsregler angebunden. Das Gesamtsystem ist mit allen 71 Motorik- und Sensorikkomponenten in Abbildung 8 gezeigt.

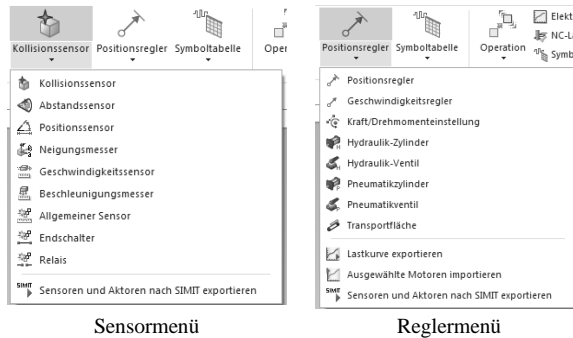


Abbildung 7: Hauptfunktionen im MCD

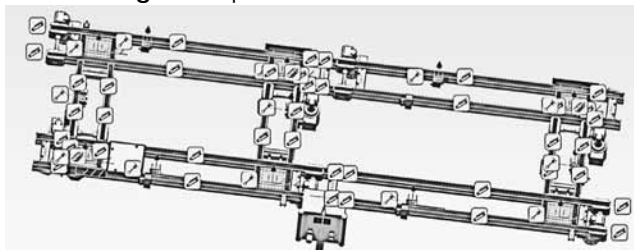


Abbildung 8: Simulationskomponenten des Gesamtsystems

Für eine einfache Model-in-the-loop (MIL) Simulation kann innerhalb des MCDs mit dem Sequenzeditor ein Umlauf eines Werkstückträgers erstellt werden (Abbildung 9). Daraus folgt das Simulationsablauf aus Abbildung 10.

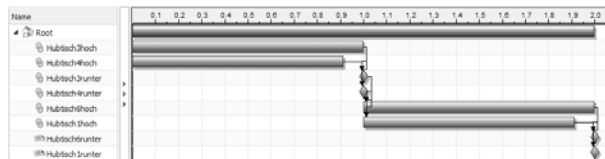


Abbildung 9: Sequenzeditor MCD – Konfiguration der Umlaufsteuerung eines Trägers

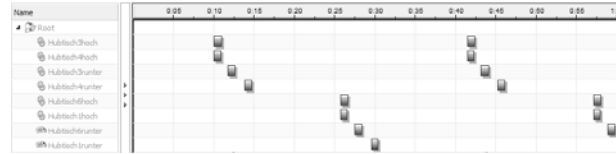


Abbildung 10: Sequenzeditor MCD – Aktive Elemente von zwei Umläufen eines Trägers
Mit diesem simplen Umlauf könnten beispielsweise Kollisionen innerhalb des Systems relativ schnell gefunden werden, auch können hiermit erste Abschätzungen bzgl. der Kräfte, Geschwindigkeiten, Beschleunigungen, etc. realisiert werden (Abbildung 11).

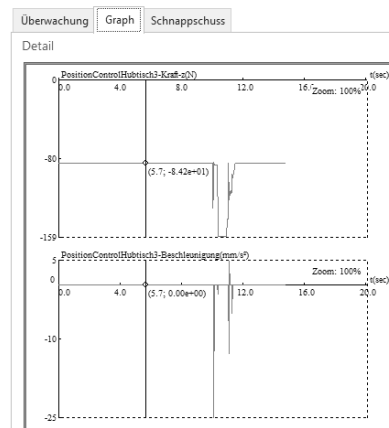


Abbildung 11: Beispielhafte Simulationsdaten vom Hubtisch 3 (Positionssensor) im MCD-Inspector

Ein kompletter Werkstückträgerumlauf mit 0,3m/s Transportgeschwindigkeit benötigt 32 Sekunden und kann in der Simulation viermal so schnell berechnet werden (System: 4x3,3Ghz; 8GB RAM). Die Systemauslastung steigt dabei von 30% auf durchschnittlich 80% und vom Arbeitsspeicher werden von NX etwa 800MB belegt.

4 MCD-Schnittstellen

Für eine Software-in-the-loop (SIL) Simulation stehen weitere Schnittstellen zur Verfügung. Hier dient der MCD dann als Ersatz für die reale Anlage.

Mögliche Schnittstellen zum MCD wären:

- OPC DA (Open Platform Communications Data Access)
- SHM (shared memory)
- Matlab
- PLCSIM
- TCP und UDP
- Profinet

So bietet beispielsweise die PLCSIM Schnittstelle mit

dem PLCSIM Advanced Tool die Möglichkeit, das Modell mit einer simulierten SPS zu verknüpfen, auf welcher das echte SPS-Programm für eine virtuelle Inbetriebnahme aufgespielt werden kann (Abbildung 12). Dies gilt aber nur in Grenzen. Einige Sensoren oder Aktoren werden nicht direkt an Ein- und Ausgänge der SPS angeschlossen, sondern auch über Bussysteme (z.B. Profibus), welche nur aufwendig simuliert werden können.

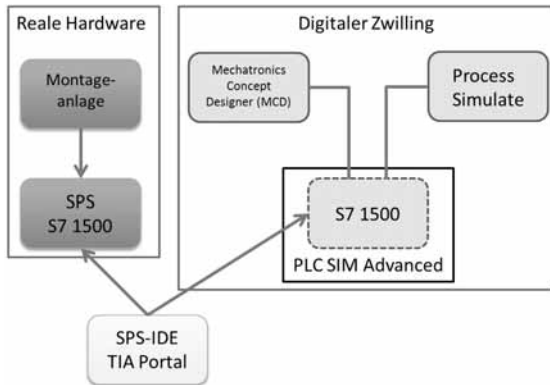


Abbildung 12: PLC SIM Advanced Einbettung in der Systemumgebung

Für dieses Problem bietet Siemens unter anderem SIMIT an. Hiermit sind Verhaltensmodelle und auch Hardware in the Loop (HIL) simulierbar. Beim HIL kann die reale SPS das Modell steuern. Weitere Informationen dazu finden sich im nächsten Kapitel.

Innerhalb des TIA-Portals (Entwicklungsumgebung für Siemens SPS) kommen die IEC 61131-3 Programmiersprachen zur Anwendung. Eigene Funktionen für die virtuelle Inbetriebnahme der Anlage wurden in Structured Control Language SCL (angelehnt an ST) geschrieben.

5 Abgrenzung zu anderen Siemens-Simulationstools

Dieses Kapitel dient der Einordnung des MCD und zur Übersicht der verschiedenen Tools der Produktionssimulation innerhalb des Siemenskonzerns (Abbildung 13).

• Produktionslinie		Tecnomatix Plant Simulation	
• Roboterzelle		Tecnomatix Process Simulate	
• Produktions Maschine		NX Mechatronic Concept Designer	
• Komponentenphysik		Simcenter Amesim	
• Komponenten & Peripherie		SIMIT	
• Automatisierung		PLCSIM Advanced & WinCC	

Abbildung 13: Tools der Produktionssimulation Siemens AG [6]

- Tecnomatix Plant Simulation hat den Fokus auf den

Materialfluss in ganzen Produktionshalle und für die Untersuchung von Engpässen in der Produktion. Es können auch bspw. Schichtpläne, Laufwege, MTBF (Mean Time Between Failures) mit berücksichtigt werden.

- Tecnomatix Process Simulate ist fokussiert auf ganzheitliche Prozessabbildung ohne Physik-Engine, beispielsweise in der Roboterzelle. Es dient u.a. für die Berechnung Erreichbarkeiten, Prozesszeiten, und Kostenabschätzungen, aber auch für die Erstellung von Ablaufsimulationen. Einfache Pick & Place Roboterprogramme könnten auch im MCD nachgebildet werden.
- SIMIT ist dazu da, sich logisch wie Profinet-Peripherie zu verhalten und die Reaktion der Komponenten auf Telegramme, etc. nachzubilden (Abbildung 14).

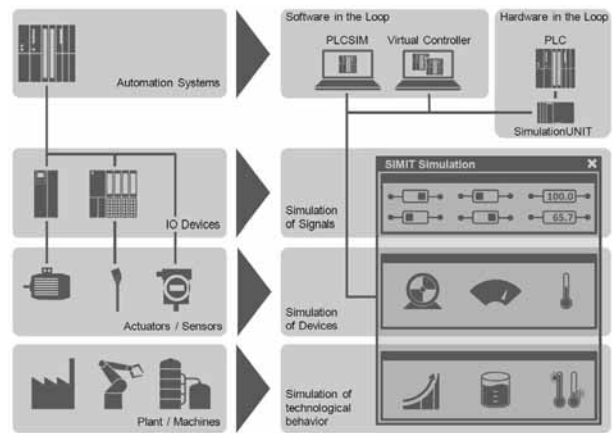


Abbildung 14: Gegenüberstellung einer realen und einer simulierten Anlage in SIMIT [7]

Es können in SIMIT auch prozessspezifische Verhalten simuliert werden, z.B. Füllstände von Tanks, Verzögerungen, Hochlauf rampen oder Limits. Optional bietet SIMIT fertige Bibliotheken mit Simulationsobjekten aus verschiedenen Bereichen an. Das Tool kommt ursprünglich aus der Prozessindustrie. Es wird nun aber zunehmend auch für die ‚diskrete‘ Industrie verwendet [6].

Als übergeordnete Software kann über die gesamte PLM-Phase für diese Tools Teamcenter genutzt werden.

6 Zusammenfassung, Bewertung und Ausblick

Das beschriebene Vorgehen anhand des Montagebands spiegelt nicht den normalen Entwicklungsprozess mit dem MCD wider. Es wurde invers angewandt, indem von einem bestehenden System ein Modell bzw. digitaler Zwilling erzeugt wurde. Auch ist eine sinnvolle Kollisionserkennung an diesem Beispiel nicht umsetzbar und kann erst bei komplexeren Anlagen Fehler finden, welche nicht so offensichtlich sind.

Um Softwareteile in der realen SPS, sowie für die Simulation nutzen zu können, ist eine andere Art der Programmierung nötig. Funktionen müssen stärker autark lauffähig sein, um in der virtuellen SPS ohne Peripherie, nur mit dem digitalen Zwilling, getestet werden zu können. Notausssysteme und Busanbindungen müssen stark separiert vorgesehen werden.

Das Zusammenwachsen der verschiedenen Teilbereiche in der Entwicklung von Fertigungsanlagen erfordert einen hohen Schulungsaufwand und eine Spezialisierung der Mitarbeiter. Zudem kommen mit jeder neuen Funktion neue Softwarekosten bei Wartung und Beschaffung hinzu.

Der Pflegeaufwand für einzelne Bauteile/Baugruppen wird steigen, da zusätzlich zu den konstruktiven Änderungen auch die physikalischen Modellanpassungen mit archiviert werden müssen. Es muss darauf geachtet werden die Baugruppen universell aufzubauen, um die Wiederverwendbarkeit von Anlagenteilen zu ermöglichen.

Die in NX vorhandenen FEM-Simulationen können nicht mit dem MCD gekoppelt werden. So gibt es beim MCD keine Wärmeentwicklung (Längenausdehnung der Bauteile) oder Steifigkeitseinflüsse (z.B. Biegung) von Teilen.

Die durchgängige Vernetzung der Siemens PLM-Software ist noch einzigartig und hat wie beschrieben viele Vorteile. Von einem Anbieter abhängig zu sein, könnte auf lange Sicht auch Nachteile haben, um dies zu bewerten fehlen Details zur Lizenzgestaltung.

Die Hochschullizenzen sind bei NX ähnlich dem der Konkurrenz, der Funktionsumfang ist mit dem MCD und enthaltenen FEM-Lizenzen aber größer.

Zurzeit wird an der Implementierung der Roboter (Stäubli/Kuka) am Montageband in den digitalen Zwilling über shared memory (SHM) gearbeitet und die SPS Programme für die Ansteuerung des Modells angepasst.

Quellen

- [1] M. Eigner, D. Roubanov, and R. Zafirov, Eds., *Modellbasierte virtuelle Produktentwicklung*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.
- [2] Friedrich Peschke, *Product Lifecycle Management (PLM)*, 1st ed. München: Hanser, 2017.
- [3] Siemens AG, *Anwenderbericht Mechatronics Concept Designer: Festo: 65121 A4 DE 07/17 loc.*
- [4] URL: <https://pybullet.org>. Jan 2019.
- [5] Siemens AG, URL: <https://training.plm.automation.siemens.com/mytraining/index.cfm>. Jan 2019.
- [6] Siemens AG, *Simulationsprodukte*. via Email; Digital Factory; Dez 2018.
- [7] Siemens AG, *SIMIT Simulation V9.0 Getting Started: SIMATIC PCS 7 V8.2, SIMIT Simulation V9.0 SP1*. Beitrags-ID: 77362399; V1.0, 2017.

Inside the Virtual Test Aircraft (VIRTTAC) Benchmark Model: Simulation Architecture

Nicolas Fezans^{1*}, Christoph Deiler^{1†}

¹DLR - German Aerospace Center, Institute of Flight Systems, Lilienthalplatz 7, 38108 Braunschweig, Germany;
*nicolas.fezans@dlr.de, †christoph.deiler@dlr.de

1 Introduction

One of the major goals of research and innovation in aviation is to enhance the overall air traffic safety and to make traveling even more comfortable for both pilots and passengers. Novel aircraft safety and control features are normally developed for distinct aircraft type due to e.g. a certain demand from the aircraft manufacturer or its availability for research facilities in terms of the existence of high-quality simulation models or flight testing capabilities. In the last years, numerous interesting and noticeable innovations to enhance aviation safety have been published for different aircraft types. For example, a very small study of developments in the field of aircraft flight envelope protection revealed that 12 different aircraft types or models were used in numerous publications [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17]. Hence, comparison of all the different developments is very difficult. Furthermore, the assessment of the applicability of a published new methodology for a different type of aircraft is very difficult as normally the reader's knowledge about the underlying system is quite small. To overcome this problem and provide a common simulation model to the research community, NASA introduced in 2011 a generic aircraft simulation model called the "Transport Class Model" (TCM) derived from a sub-scale "Generic Transport Model" (GTM) simulation [18]. It is a fully functioning aircraft simulation including realistic engine and actuator behavior, sensor models and a flight control system. Although a significant number of failure scenarios were considered, computed and tested in CFD and wind tunnels [19], only a few of them were implemented in the distributed Simulink simulation model.

Moreover, the problem of comparability between various new developments is also present within the

field of aircraft system identification. Various algorithms for parameter estimation and simulation model identification as well as related software tools have been developed during the last decades, but most of them were tested and verified for different aircraft. For example, Refs. [20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38] show results of various system identification techniques for more than 20 different aircraft types. Consequently, there is the problem to assess the quality of each methodology as there is no common base for an objective evaluation. The proposed high-quality model will be made available to all developers and will constitute a good complement to the already existing/available models.

This paper presents a new generic benchmark model that can be freely downloaded and used by the community for all kinds of investigations and in particular for defining benchmark applications to compare various approaches. This model is called VIRTTAC-Castor and is the first member of a model family called VIRTTAC, see sections 2 and 3. The way this model works internally and the main choices made for the internal architecture of the current model implementation are presented in section 4. The conditions of use and licenses are mentioned in section 5. The foreseen applications (see section 6) include but are not restricted to supporting the investigations related to the estimation of the reduced flight envelope, fault-tolerant aircraft flight control, or the development of enhanced system identification techniques. It is built based on the knowledge gained at DLR over several decades of flight research, including all aspects of model building based on wind tunnel and CFD data as well as from flight testing (see [39] and references therein). By distributing this benchmark model and scenarios the authors intend to share some of this knowledge with the community and to help building comparisons across techniques used by different research groups.

2 VIRTAC: VIRtual Test AirCRAFT – Motivation and Objectives

2.1 Motivation

The whole idea of creating VIRTAC comes from the observation of the authors that in the area of flight dynamics and flight control there is a lack of commonly available, good and realistic benchmark models close to real aircraft behavior and characteristics. Most engineers and researchers are developing and/or using proprietary models for their work, but they often cannot share these models. Very often these restrictions result from the vehicles themselves and the fact that the manufacturer consider that these models might reveal some trade secrets or that they might lose some control over the investigations made based on the models of their vehicles. Within very large companies and organizations other types of issues can often be observed: dilution of responsibilities across several sub-organizations, internal dynamics, lack of incentive for long-term actions (constant changes in the intermediary management levels), often leading individuals to the conclusion that releasing some models and information is a potential risk for their career with little to no expected personal benefit.

Apart from slowing down research and innovation, this situation is also problematic in the sense that good science thrives through comparing hypotheses with observations and through independent validation of the results by different teams. Reproducibility of the results and cross-checking have been one of the corner stones in science and will remain so. Engineering-related disciplines differ from more fundamental science in the sense that its actual goal is less to produce new knowledge than to create something of economical or strategic value from the current body of knowledge. Whilst new knowledge might be produced along the way, the context strongly drives engineering work towards a future return on investment. In this context, openness is mainly seen as a potential future loss of revenue and as potentially endangering the currently foreseeable revenues (e.g. through additional risks). In order to support research and science in their domains, the authors decided to build and provide VIRTAC to the entire community.

2.2 Objectives

VIRTAC is developed with two main objectives in mind.

1. Provide high-quality representative models to engineers and researchers who need some but do not have access to the kind of research infrastructure that the authors have access to.
2. Provide a wide range of benchmarks to the community with various complexity levels, including some which include as many real-world effects as possible. The objective for the most complex benchmarks is that it should never be possible to pass them successfully but fail in the real-world due to effects that could not be tested with VIRTAC. Whilst the objective is to build complete benchmarks, the current work focuses on the development of the dynamic aircraft model at the heart of these benchmarks.

This last element “never pass the most complex benchmark if it fails in practice” directly leads to the need for representative system architectures and for modeling of all kinds of real-world effects. Information that would not be available in practice should also be hidden from the users by VIRTAC in order to ensure that it cannot be exploited. This includes information on the internal working of the models, their exact structure, parameter values, etc. This also includes all values that are required for performing the simulations but which would not be available in practice (e.g. information for which no sensor exists or is installed/available in a real aircraft).

“VIRTAC users are basically aeronautical engineers who are confronted with a new aircraft. They can flight-test the aircraft and learn how it flies, but there is still a difficulty to predict how it behaves and they cannot access physical quantities unless there is a sensor measuring them.”

Most users have prior knowledge about flight mechanics/dynamics and control and should use it. The behavior of VIRTAC will be very familiar to flight dynamics specialists, since the vehicle behaves like an aircraft. However, no equations and no aerodynamic coefficient derivatives will be made available. Precise knowledge of the aircraft can be gained by “virtually flight-testing” it (i.e. performing simulations). Knowledge can be exchanged with the rest of the community

(e.g. through exchange of identified models) and is encouraged. The authors intend, aside from the website where VIRTAC and its updates can be downloaded, to organize with the interested parties an exchange platform for the community and gather information regarding all investigations that were performed by VIRTAC.

In the long term, the authors expect to build several models with slightly different characteristics and behaviors. For each of these models a rough description of the model will be provided. This description will include some basic description of the shape of the aircraft and its geometry and can be imagined as what a specialist would notice by looking at the aircraft. A few key technical specifications will be provided too. It is not intended for users to generate alternative data sets on the aircraft from other sources than the provided simulation model, therefore no detailed design data of any kind will be provided (no CAD geometry, structure design, etc.). The simulation is based on a nonlinear rigid-body model, which is meant to be valid for a pre-defined flight envelope and will include several high-lift configurations and additional effects like stall or ground effect in its final version. The aircraft briefly described in the present paper is the first of the VIRTAC family. As it will receive some siblings, a simple naming nomenclature is introduced.

2.3 The VIRTAC Family: Naming Conventions

VIRTAC is meant to become a family composed of several models. The idea of using a naming nomenclature for the whole VIRTAC family has been considered and rejected, at least for now, due to the difficulty of ensuring that this nomenclature will be precise enough to differentiate the models that would be integrated in the future and also stable over time, such that the references made to one or the other model stay valid over extended periods of time.

As the number of models and variants expected to be developed and shared within the community will remain relatively low (most likely below 15), it was decided to give names to these models and to maintain a directory with the corresponding information for each of them. The individual names will be chosen such that:

1. They can be relatively easily pronounced by a wide range of speakers and easily distinguished even if pronounced by a non-native speaker.
2. They can be easily found with a search engine.

For this a web search using both “VIRTAC” and the name of the configuration should lead only or mostly to documents related to that aircraft model, for instance past publications using this model.

Whilst the authors might chose other types of names in the future, these requirements should be satisfied with many star and galaxy names. The name VIRTAC-Castor is chosen for the first aircraft of the VIRTAC family introduced hereafter. This aircraft is a twin-turbofan configuration in the 100-passenger category. A twin-turboprop variant of this aircraft is foreseen and the name VIRTAC-Pollux is already reserved for it.

3 VIRTAC-Castor Model

3.1 Aircraft Geometry and Configuration

VIRTAC-Castor represents a generic short- to medium-haul transport aircraft for around 100 passengers with a high wing (small anhedral) and a T-tail configuration. This configuration has been completely created from scratch for VIRTAC. It has a configuration that remembers the Dornier 328 JET but is significantly larger. It is somewhat between a BAe 146-200 / AVRO RJ85 and a BAe 146-300 / AVRO RJ100 in terms of size, but only has two turbofan engines. Note that, even if DLR did identify models for the Dornier 328 [26] the herein proposed model was not derived from these data. As already mentioned, this configuration will receive a sibling (VIRTAC-Pollux) later that will be based on two turboprop engines, leading to different engine dynamic behavior as well as greater coupling of the engines and the aerodynamics due to the propeller slipstream.

An artistic illustration of the VIRTAC-Castor is given in figure 1 and as well in figure 2 as three-side view. This illustration is provided for a common understanding of the modeled aircraft but no CAD model and precise geometry is given/distributed (at least for now). This aircraft was not produced through a complete pre-design process but its dimensions and characteristics should correspond to a short-to-medium range commercial air transportation role with a capacity of around 100 passengers. Table 1 provides an overview on its current dimensions and characteristics, which is not complete but can give the user the necessary information for subsequent model use.

3.2 Aircraft Aerodynamics

The aircraft model’s aerodynamics contain formulations to consider nonlinear and unsteady effects of wing and empennage. The model benefits from DLR’s large experience in modeling and identifying complex aerodynamic models for different airplanes, regions of the corresponding flight envelope and distinct applications in simulation [40, 41, 26]. The aerodynamic model is primarily formulated as a derivative model but includes several specific and complex extensions to enhance the model’s capabilities. It allows for example to cover unsteady trailing edge flow separation, which allows to model the normal stall behavior for an aircraft configuration as given in figure 1. The aerodynamic model formulation further allows to easily implement failure cases of an aerodynamic degradation of various sources as defined in section 3.7. A ground effect model (ground currently always at the elevation of 0 meters) is already included.



Figure 1: Artistic illustration of the VIRTAC-Castor configuration

AC length	30.0 m
wing span	28.0 m
horizontal tail span	10.4 m
wing area	75.0 m ²
horizontal tail area	20.0 m ²
wing aspect ratio	10.4
mean aerodynamic chord	2.17 m
max. take-off weight	56 000 kg
empty weight	33 000 kg
max. fuel weight	16 000 kg
max. payload / PAX weight	12 000 kg
max. range	5 500 km
max. altitude	35 000 ft
max. operating Mach number	0.76
cruise Mach number	0.725

Table 1: Overall dimensions and characteristics of VIRTAC-Castor



(a) top view



(b) front view



(c) side view

Figure 2: Artistic illustration of the VIRTAC-Castor configuration, multiview projection

3.3 Propulsion

The VIRTAC-Castor model includes two turbofan engine models which can be controlled separately. The engine command inputs virtually correspond to a N1 (engine fan shaft rotation speed) expressed in %. The dynamic model will therefore correspond to the behavior of the engine plus the corresponding FADEC.

In the long-term quite good engine dynamic models will be integrated in VIRTAC-Castor and in all or most models in the VIRTAC family. However, a significant amount of work is still required from the authors in order to finish building up these models and to integrate them into the VIRTAC structure. As a consequence, the first versions of VIRTAC-Castor are expected to be delivered with much simpler preliminary models. These models will be representative for most scenarios, but as soon as the user-implemented flight control system will be very dependent on the en-

gines' transient response, the validity of results will have to be checked. For instance, no serious development and tuning of a control law or autopilot based only on the engines (i.e. a propulsion-controlled aircraft or PCA as in [42, 43, 44, 45]) will be possible with the preliminary model. Simple relatively low-gain autothrust/autothrottle functions would however not be too strongly affected.

3.4 Flight Controls

The simulation model of VIRTAC-Castor contains several control surfaces including various spoilers on the wing which may be more than usual for this size of airplane. In detail, the model provides:

- trimable horizontal stabilizer
- left and right elevators
- left and right ailerons
- rudder
- five spoilers on each side (four roll spoilers/airbrakes and one ground spoiler)

Actuator models are included for all control surfaces. Limits (deflection, deflection rate, and acceleration) are included. The actual control surface deflection is measured internally by the actuator and provided as output of the VIRTAC-Castor model. The commanded signal and the measured deflection can therefore be compared; for instance users might want to compare them within a flight control system fault detection logic. Numerous possible faults will be integrated in the actuator models and be added over time, see section 3.7 hereafter.

3.5 Sensor Models

Sensor models are a crucial element for VIRTAC: they are the only way to know what is happening to the aircraft during the simulation. The physical quantities measured, the sensor characteristics (e.g. calibration, noise, dynamic behavior, quantization errors) as well as all the real-world issues related to where and how they are installed on the airframe will be defined as closely as possible to the state-of-the-art regular aircraft instrumentation. Currently, the authors are considering future inclusion of better sensors, which would resemble a complementary flight test instrumentation (FTI) and could be used for system identification studies. If FTI-like sensors were included in VIRTAC in the future, these sensors should not be used for flight control, flight

control adaptation, or fault detection and isolation studies as they would not normally be available on the aircraft in regular operations.

The usual list of measurements provided by air data and inertial reference systems on Part/CS-25 airplanes is available for VIRTAC(-Castor). This includes attitude angles, rotational rates, accelerations, inertial velocity vector, static and total pressure and vertical speed, the various airspeeds, inflow angles (α , β), air temperature, etc. and many derived quantities. For each available measurement sensor characteristics and real-world effects are considered. When it is common practice to have redundancies in the regular aircraft instrumentation, several sensors will also be modeled. The relationships used in practice to derive the physical quantities that are not directly measured will be modeled such that the propagation of faulty measurements can be correctly simulated during faulty scenarios.

For now, VIRTAC-Castor contains three inertial reference units providing the corresponding measurements of accelerations, rotational rates and attitude. Furthermore, there are four air data systems measuring angle of attack, angle of sideslip, and calibrated airspeed as well as static pressure, static temperature and barometric altitude¹.

3.6 Landing Gear

The influence of the landing gear on the aerodynamics is already included, but the gear itself, the wheel and tire dynamics as well as the braking system are not implemented yet. In the future, VIRTAC will also include models for the landing gears and their systems as well as more realistic ground elevation profiles and runway characteristics.

3.7 Test Scenarios and Failure Cases

For the first version of the benchmark model several test scenarios and failure cases are already available and will be extended in the future. The currently foreseen set of scenarios mainly contains aerodynamic degradation and control surface actuator failures:

Wing Ice Case

Ice can have hazardous effects on the aircraft's flight characteristics. Large accumulations on the wing – mainly mostly near the wing leading edge – increase the

¹with VIRTAC-Castor version 0.5-alpha

drag and reduce the maximum angle of attack and consequently increase the stall speed. This has a direct influence on the safe flight envelope and poses a threat to crew and passengers. VIRTAC-Castor (and probably most future VIRTAC family models) will be capable of considering the effects of a generic wing ice accumulation in terms of the resulting aerodynamic degradation. The timely increase of degradation resp. accumulation as well as a de-icing can be triggered by the user whereas the details about the degradation itself are part of the closed model to allow a fair and realistic test of new developments like detection algorithms or robust flight controllers. The corresponding knowledge about the expectable effects and a realistic amount of degradation is derived from previous icing research at DLR [46, 47] where high-quality simulation models were identified from flight data.

Horizontal Tail Damage & Icing

The model will include changes of dynamic behavior caused by a partial loss (various levels) of one side of the HTP similar to [48] resulting in a changed controllability of the aircraft. Partial damage at the HTP or VTP leading edge as well as local icing of the empennage will probably be included in the future.

Actuator Faults

Faults in the actuators will be included in the future. Each actuator will be controlled independently (ailerons, elevators, rudder, spoilers) and possibly be subject to faults. The faults cases that will be implemented include the typical actuator faults such as hardover, runaway, frozen at a given position, change in dynamic behavior, etc.

Engine Bird Strike

Bird strike damage to engines is considered for later inclusion in VIRTAC-Castor. Simulation models were developed at DLR for the EU FP7 Man4Gen project [49, 50], which could be adapted for the VIRTAC-Castor turbofan engine, once its nominal version will be available. These models are relatively simple and the variability of the effects of bird strikes in engines makes it very difficult, if not impossible, to build a generically valid model for such events. Investigating the adverse consequences of such failures onto advanced fault detection algorithms and their robustness against them is certainly interesting. This would be possible when such engine fault models will be integrated into VIRTAC.

4 Implementation Challenges

4.1 Need for a Black-Box Implementation

The internal implementation of the VIRTAC dynamic models is hidden from the user (black-box), which permits to prevent users from accessing any information which would not be available in practice. In the MATLAB/Simulink environment this is done by encapsulating the dynamic model in a Simulink s-function. This has also the advantage of significantly speeding-up the simulations while still letting the users be able to include their part (e.g. a flight control system) around the VIRTAC flight mechanics models. At the time of writing this paper, the current implementation of the VIRTAC-Castor model runs about 50 times faster than real-time on a Desktop PC with MATLAB/Simulink R2007b 32bit under Windows 7 Enterprise SP1 and with an Intel Core i7-2600 @3.4 Ghz.

4.2 Availability Beyond MATLAB/Simulink

The authors (and designers of VIRTAC) also aim at ensuring the wide availability of the VIRTAC models and their long-term maintainability. The wide availability includes the possibility to bring VIRTAC to other environments, including other scientific software suites than MATLAB/Simulink. The internal implementation therefore makes a clear separation between the MATLAB/Simulink-specific features/APIs and the code responsible for the trim and simulation of the aircraft. The former is confined to the s-function code which is only a wrapper for the actual dynamic model (the latter). The latter is implemented in a library to which the s-function is statically linked. This decomposition should permit to create VIRTAC implementations which could be used with a large variety of other tools, such as Scilab/Xcos or even using the Python programming language. The authors want to have the possibility of offering VIRTAC for other tools than MATLAB/Simulink with a reasonably low amount of work; however, it is crucial to ensure that the capabilities provided by VIRTAC and its specific benchmarks remain the same for all these implementations. A benchmark scenario will only be officially supported by the authors if it is available for all supported implementations, or if there is a valid technical reason for not doing it. A valid technical reason might be that the benchmark scenario includes VIRTAC and another component which cannot be ported to the other tools/platforms (e.g. due to

intellectual property restrictions or if porting this component would require an unreasonable effort).

4.3 Modularity for Long-Term Maintainability

The long-term maintainability includes the need for a modular internal structure of the simulation code. This is done by using a general structure, whereby the different model parts can be defined separately and “connected” to each other. This makes the modeling paradigm used inside the VIRTAC models similar to the one used in graphical modeling tools, like Simulink or Xcos. No graphical editor has been developed to create or edit these connections but the C++ syntax used for this is as simple as the MATLAB/Simulink “add_line” command. The internal implementation concept chosen might evolve over time without the VIRTAC users noticing any change². Currently, the internal implementation is a compromise between modularity and complexity of the implementation workflow. More automated or even code-generation-based solutions have not been chosen because it has been estimated that their additional level of complexity and their restrictions³ outweigh the benefit expected for VIRTAC. Depending on the future evolution of VIRTAC and of the corresponding aircraft models and benchmark scenarios this choice might be reevaluated. One of the important features that the current implementation requires is a mechanism permitting to call the different model parts in the right order such that the simulation results remain correct, i.e. as if the model had not been split into different entities. Note that such a mechanism is also required in simulation tools like Simulink and Xcos.

The main modules that are currently defined are:

- Flight Control Actuators
- Propulsion
- Airframe Dynamics (which includes the aerodynamics and the equations of motion)
- Sensors
- Landing Gear (foreseen but not implemented yet).

A simple environment model is already included in the current version of VIRTAC but is not a separate module. It includes a standard atmosphere model and a Dryden turbulence generator. The possibility to bring some

²Beyond possibly slightly modified rounding errors, which could also be caused by changing the compiler or the compiler flags.

³They always include some assumptions on the structure of the system and/or produce a barely human-readable code.

icing conditions is currently being developed to enable icing-related envelope reduction benchmark scenarios. The implementation of the environment model is likely to be largely redesigned in the near to mid-term future.

5 Conditions of Use

5.1 Who Can Use VIRTAC? What Are the Conditions of Use?

Source files

As of now, anyone can download and use the VIRTAC models. Any part of VIRTAC provided in source form (e.g. MATLAB .m files or Simulink models) is subject to the very permissive MIT license:

MIT License

Copyright (c) 2018 Deutsches Zentrum für Luft- und Raumfahrt e.V., Christoph Deiler, Nicolas Fezans

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Other files: executable, binaries, binary data

All non-disclosed code and data files (e.g. executable, dynamic/static libraries, binary files, etc.) are licensed under the Creative Common Attribution-NoDerivs 4.0 Generic license (CC-BY-ND 4.0). A human-readable summary is provided hereafter: please refer to the official license text for the legally binding text. Note that any attempt to disassemble the binary code, binary data, executable, or dynamic/static libraries provided is hereby considered as a derivative and is consequently hereby prohibited, even if not shared. A normal use of VIRTAC for its intended purpose does not require such operations and therefore users will normally not be affected by this restriction.

CC-BY-ND 4.0 (*human-readable summary*)

• **You are free to:**

- Share – copy and redistribute the material in any medium or format for any purpose, even commercially.
- The licensor cannot revoke these freedoms as long as you follow the license terms.

• **Under the following terms:**

- Attribution – You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- NoDerivatives – If you remix, transform, or build upon the material, you may not distribute the modified material.
- No additional restrictions – You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

6 Foreseen Applications and Community Involvement

VIRTAC is expected to become a very useful tool for a quite large community of researchers and engineers working in the areas of atmospheric flight mechanics (AFM) and guidance, navigation and control (GNC). A couple of foreseen uses are listed hereafter and the authors are welcoming further development suggestions linked to any other potential application of VIRTAC.

System Identification / Machine Learning

Aircraft system identification is a rather obvious potential use of VIRTAC, as it allows its users to perform *virtual* flight tests. Very few organizations and companies worldwide possess the financial resources and the technical means to perform large flight test campaigns for system identification purposes. VIRTAC is expected to give many researchers and engineers access to a (virtual) test aircraft, who would not have this kind of possibility otherwise.

Flight Control, Flight Guidance, and Fault-Tolerant Control

The flight control and flight guidance communities can also benefit from VIRTAC models as they constitute fully working and representative aircraft models on which control and guidance concept can be developed, tested, and compared among teams that would otherwise not have common benchmarks, for instance due to intellectual property restrictions. VIRTAC is designed from the beginning to support fault-tolerant control in all its possible forms by providing representative models with many possible fault scenarios. The objective of the VIRTAC models is to provide valid models 1) which help the users (e.g. control scientist) to understand the exact consequences of these faults on the overall system and 2) which can be used to validate and demonstrate the fault-tolerance capabilities of some controllers. Very often the fault-tolerant control techniques will require specific model formulations to be designed or to be used online: VIRTAC provides no simplified model for control design, each user should build his/her own simplified model or reuse some that might have been built and shared by others.

Within a few years period, it is planned that VIRTAC models will also be ported to the DLR AVES simulator in order to permit pilot-in-the-loop evaluations of the most interesting control concepts developed for

VIRTTAC and for which a pilot-in-the-loop evaluation could be valuable. Please contact the authors for further information on VIRTTAC@AVES or to send some suggestions.

Teaching Flight Mechanics and Control

The authors expect that VIRTTAC could become very useful for teaching purposes (in aerospace engineering but also for pilot training), even though teaching is not in the focus of the current developments. Nevertheless, the authors would be happy to support such efforts, within what can reasonably be performed without limiting the usability for the other potential user groups.

Long Term Evolution of the Model and Community Involvement

The model provided with this paper is only the first step of a long-term initiative aiming at providing good and representative models to the community. Even if the task of building a flight dynamics model is a very instructive, the amount of work spend in our community to build models for the purpose of our research activities is very high. Additionally these models are often very restricted due to unavailability of the required data to the model builder or to other practical constraints. The multiplication of models and the relative lack of a common benchmark often makes it difficult to compare the proposed approaches.

The evolution of the herein proposed benchmark models and scenarios will be strongly oriented towards the needs of the community. In order to remain a good validation tool, the system will need to remain only partly observable to the end user and therefore some parts will remain undisclosed (at least for several years). For everything else (e.g. definition of new test scenarios, automatized evaluation scripts, etc.) VIRTTAC will be as open as possible and contributions from the community are very much welcome.

7 Summary

This paper presents the recently started development of a series of generic aircraft models gathered within the VIRTTAC family. These models will be available to the research community in the future, e.g. as high-quality validation benchmark models or for testing new methodologies in the fields of robust control or reduced envelope protection. The first model of this family, the 100-passenger jet airplane VIRTTAC-Castor, already

provides within its preliminary 0.5-alpha version a basis including various necessary functions to simulate respectively virtually flight test the aircraft on the one hand. On the other hand, the aircraft might already been used for its foreseen purposes. After finishing the development of VIRTTAC-Castor in the near future, it will be followed by a turboprop version of similar size called VIRTTAC-Pollux.

VIRTTAC Download and Contact Information

To provide the models of the VIRTTAC family to the community, a GitHub repository was created. This repository is located at

<https://github.com/VIRTTAC/VIRTTAC>

and will be updated if necessary due to new model developments of aircraft within the VIRTTAC family.

For any questions on VIRTTAC-Castor, the VIRTTAC family or for general support concerning the VIRTTAC simulation, please use the following VIRTTAC email address:

VIRTTAC@dlr.de.

References

- [1] Lombaerts T, Looye G, Ellerbroek J, Rodriguez y Martin M. Design and Piloted Simulator Evaluation of Adaptive Safe Flight Envelope Protection Algorithm. *Journal of Guidance, Control, and Dynamics*. 2017; 40(8):1902–1924.
- [2] Lombaerts T, Looye G, Chu P, Mulder JA. Pseudo Control Hedging and its Application for Safe Flight Envelope Protection. AIAA Guidance, Navigation, and Control Conference, Guidance, Navigation, and Control and Co-located Conferences, Paper No AIAA 2010-8280. Toronto, Ontario, Canada: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2010; .
- [3] Tomlin C, Lygeros J, Sastry S. Aerodynamic Envelope Protection Using Hybrid Control. Proceedings of the American Control Conference. Philadelphia, Pennsylvania, USA: Institute of Electrical and Electronics Engineers (IEEE). 1998; .
- [4] Shin HH, Lee SH, Kim Y, Kim ET, Sung KJ. Design of a Flight Envelope Protection System Using a Dynamic Trim Algorithm. *International Journal of Aeronautical and Space Sciences*. 2011;12(3):241–251.

- [5] Well KH. Aircraft Control Laws for Envelope Protection. AIAA Guidance, Navigation, and Control Conference and Exhibit, Guidance, Navigation, and Control and Co-located Conferences, Paper No. AIAA 2006-6055. Keystone, Colorado, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2006; .
- [6] Rafi M, Steck JE, Rokhsaz K. A Microburst Response and Recovery Scheme Using Advanced Flight Envelope Protection. AIAA Guidance, Navigation, and Control Conference, Guidance, Navigation, and Control and Co-located Conferences, Paper No. AIAA 2012-4444. Minneapolis, Minnesota, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2012; .
- [7] Wilson JM, Peters ME. Automatic flight envelope protection for light general aviation aircraft. 28th Digital Avionics Systems Conference. Institute of Electrical and Electronics Engineers (IEEE) and American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2009; .
- [8] Hossain KN, Sharma V, Bragg MB, Voulgaris PG. Envelope Protection and Control Adaptation in Icing Encounters. 41st Aerospace Sciences Meeting and Exhibit, Aerospace Sciences Meetings, Paper No. AIAA 2003-25. Reno, Nevada, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2003; .
- [9] Gingras DR, Barnhart BP, Ranuado RJ, Ratvasky TP, Morelli EA. Envelope Protection for In-Flight Ice Contamination. 47th AIAA Aerospace Sciences Meeting, Paper No. AIAA 2009-1458. Orlando, Florida: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2009; .
- [10] Tekles N, Holzapfel F, Xargay E, Choe R, Hovakimyan N, Gregory IM. Flight Envelope Protection for NASA's Transport Class Model. AIAA Guidance, Navigation, and Control Conference, AIAA SciTech Forum, Paper No. AIAA 2014-0269). National Harbor, Maryland, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2014; .
- [11] Ackerman KA, Talleur DA, Carbonari RS, Xargay E, Seefeldt BD, Kirlik A, Hovakimyan N, Trujillo AC. Automation Situation Awareness Display for a Flight. *Journal of Guidance, Control, and Dynamics*. 2017; 40(4):964–980.
- [12] Tekles N, Chongvisal J, Xargay E, Choe R, Talleur DA, Hovakimyan N, Belcastro CM. Design of a Flight Envelope Protection System for NASA's Transport Class Model. *Journal of Guidance, Control, and Dynamics*. 2017;40(4):863–877.
- [13] Tang L, Roemer M, Ge J, Crassidis A, Prasad J, Belcastro C. Methodologies for Adaptive Flight Envelope Estimation and Protection. AIAA Guidance, Navigation, and Control Conference, Guidance, Navigation, and Control and Co-located Conferences, Paper No. AIAA 2009-6260. Chicago, Illinois, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2009; .
- [14] Gingras D, Barnhart B, Ranaudo R, Martos B, Ratvasky T, Morelli E. Development and Implementation of a Model-Driven Envelope Protection System for In-Flight Ice Contamination. AIAA Guidance, Navigation, and Control Conference, Guidance, Navigation, and Control and Co-located Conferences, Paper No. AIAA 2010-8141. American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2010; .
- [15] Falkena W, Borst C, Chu Q, Mulder J. Investigation of Practical Flight Envelope Protection Systems for Small Aircraft. *Journal of Guidance, Control, and Dynamics*. 2011;34(2):976–988.
- [16] Ye H, Chen M, Wu Q. Flight Envelope Protection Control Based on Reference Governor Method in High Angle of Attack Maneuver. *Mathematical Problems in Engineering*. 2015;2015, Article ID 254975:15.
- [17] Lambregts AA. Flight Envelope Protection for Automatic and Augmented Manual Control. Proceedings of the EuroGNC 2013, 2nd CEAS Specialist Conference on Guidance, Navigation & Control. Delft, Netherlands: Council of European Aerospace Societies (CEAS). 2013; pp. 1364–1383.
- [18] Hueschen RM. *Development of the Transport Class Model (TCM) Aircraft Simulation From a Sub-Scale Generic Transport Model (GTM) Simulation*. Technical Memorandum, National Aeronautics and Space Administration, Hampton, Virginia, USA. 2011. URL <http://hdl.handle.net/2060/20110014509>
- [19] Frink NT, Pirzadeh SZ, Atkins HL, Viken SA, Morrison JH. CFD assessment of aerodynamic degradation of a subsonic transport due to airframe damage. In: *Proceedings of the 2010 AIAA Aerospace Science Meeting*, AIAA 2010-500. Orlando, FL, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2010; .
- [20] Klein V. Aircraft parameter estimation in frequency domain. 4th Atmospheric Flight Mechanics Conference, Guidance, Navigation, and Control and Co-located Conferences, Paper No. AIAA 1978-1344. Palo Alto, CA, U.S.A.: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 1978; .
- [21] Morelli EA, Klein V. Optimal input design for aircraft parameter estimation using dynamic programming

- principles. 17th Atmospheric Flight Mechanics Conference, Guidance, Navigation, and Control and Co-located Conferences, Paper No. AIAA-90-2801. Portland, Oregon, USA. 1990; .
- [22] Lichota P, Sibilski K, Ohme P. D-Optimal Simultaneous Multistep Excitations for Aircraft Parameter Estimation. *Journal of Aircraft*. 2017;54(2):747–758.
- [23] Raol J, Jategaonkar RV. Aircraft parameter estimation using recurrent neural networks - A critical appraisal. 20th Atmospheric Flight Mechanics Conference, Guidance, Navigation, and Control and Co-located Conferences, Paper No. AIAA-95-3504. Baltimore, Maryland, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 1995; .
- [24] Jategaonkar RV, Thielecke F. Aircraft parameter estimation – A tool for development of aerodynamic databases. *Sadhana*. 2000;25:119–135.
- [25] Jategaonkar RV, Plaetschke E. Algorithms for Aircraft Parameter Estimation Accounting for Process and Measurement Noise. *Journal of Aircraft*. 1989; 26(4):360–372.
- [26] Jategaonkar R, Fischenberg D, von Gruenhagen W. Aerodynamic Modeling and System Identification from Flight Data-Recent Applications at DLR. *Journal of Aircraft*. 2004;41(4):681–691.
- [27] Iliff KW. Aircraft parameter estimation. 25th AIAA Aerospace Sciences Meeting, Paper No. AIAA 1987-0623. Reno, Nevada, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 1987; .
- [28] Chandler PR, Pachter M, Mears M. System Identification for Adaptive and Reconfigurable Control. *Journal of Guidance, Control, and Dynamics*. 1995; 18(3):516–524.
- [29] Morelli EA. In-flight system identification. 23rd Atmospheric Flight Mechanics Conference, Guidance, Navigation, and Control and Co-located Conferences, Paper No. AIAA-98-4261. Boston, MA, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 1998; .
- [30] Morelli EA. Practical Aspects of the Equation-Error Method for Aircraft Parameter Estimation. AIAA Atmospheric Flight Mechanics Conference and Exhibit, Guidance, Navigation, and Control and Co-located Conferences, Paper No. AIAA 2006-6144. Keystone, Colorado, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2006; .
- [31] Peyada NK, Ghosh AK. Aircraft parameter estimation using a new filtering technique based upon a neural network and Gauss-Newton method. *The Aeronautical Journal*. 2009;113(1142):243–252.
- [32] Peyada N, Ghosh A. Aircraft Parameter Estimation Using Neural Network Based Algorithm. AIAA Atmospheric Flight Mechanics Conference, Guidance, Navigation, and Control and Co-located Conferences, Paper No. AIAA 2009-5941. Chicago, IL, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2009; .
- [33] Chowdhary G, Jategaonkar R. Aerodynamic parameter estimation from flight data applying extended and unscented Kalman filter. *Aerospace Science and Technology*. 2010;14(2):106–117.
- [34] Morelli EA. Flight Test Maneuvers for Efficient Aerodynamic Modeling. *Journal of Aircraft*. 2012; 49(6):1857–1867.
- [35] Morelli EA, Klein V. Application of System Identification to Aircraft at NASA Langley Research Center. *Journal of Aircraft*. 2005;42(1):12–25.
- [36] Dorobantu A, Murch A, Mettler B, Balas G. System Identification for Small, Low-Cost, Fixed-Wing Unmanned Aircraft. *Journal of Aircraft*. 2013; 50(4):1117–1130.
- [37] Fujimori A, Ljung L. A polytopic modeling of aircraft by using system identification. International Conference on Control and Automation. Budapest, Hungary: Institute of Electrical and Electronics Engineers (IEEE). 2005; .
- [38] Fujimori A, Ljung L. Model identification of linear parameter varying aircraft systems. *Journal of Aerospace Engineering*. 2006;220(4):337–346.
- [39] Jategaonkar RV. *Flight Vehicle System Identification: A Time-Domain Methodology, Second Edition*. Progress in Astronautics and Aeronautics. American Institute of Aeronautics and Astronautics, Inc. 2015. ISBN: 978-1-62410-278-3. eISBN: 978-1-62410-279-0. URL <https://doi.org/10.2514/4.102790>
- [40] Fischenberg D. Identification of an unsteady aerodynamic stall model from flight test data. No. AIAA 95-3438-CP in AIAA Atmospheric Flight Mechanics Conference. Baltimore, Maryland, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 1995; pp. 138–146.
- [41] Fischenberg D, Jategaonkar RV. Identification of Aircraft Stall Behavior from Flight Test Data. No. 17 in RTO Systems Concepts and Integration Panel (SCI) Symposium: System Identification for Integrated Aircraft Development and Flight Testing. Madrid, Spain: NATO Research and Technology Organisation. 1998; .
- [42] Burcham FWJ, Burken JJ, Maine TA, Fullerton CG. Development and flight test of an emergency flight

control system using only engine thrust on an MD-11 transport airplane. *Tech. rep.*, NASA. 1997. TP-97-206217.

- [43] Bull J, Mah R, Hardy G, Sullivan B, Jones J, Williams D, Soukup P, Winters J. Piloted simulation tests of propulsion control as backup to loss of primary flight control for a B747-400 jet transport. *Tech. rep.*, NASA. 1997. TM-112191.
- [44] Fezans N. Simple control law structure for the control of airplanes by means of their engines. In: *Advances in Aerospace Guidance, Navigation and Control*, edited by Holzapfel F, Theil S. Springer. 2011;ISBN: 978-3-642-19816-8.
- [45] Fezans N, Gamaleri M. Emergency propulsion-based autoland system. In: *Proceedings of the ICAS Congress 2012*. Brisbane, Australia. 2012; .
URL http://www.icas.org/ICAS_ARCHIVE/ICAS2012/ABSTRACTS/503.HTM
- [46] Deiler C. Time Domain Output Error System Identification of Iced Aircraft Aerodynamics. *CEAS Aeronautical Journal*. 2017;8(2):231–244.
- [47] Deiler C. Aerodynamic Modeling, System Identification, and Analysis of Iced Aircraft Configurations. *Journal of Aircraft*. 2018; 55(1):145–161.
- [48] Fezans N, Kappenberger C. A model of horizontal tailplane damage for use in flight dynamics. 28th International Congress of the Aeronautical Sciences. Brisbane, AUS: International Council of the Aeronautical Sciences (ICAS). 2012; p. 14.
URL http://www.icas.org/ICAS_ARCHIVE/ICAS2012/PAPERS/508.PDF
- [49] Buch JP, Niedermeier D. Das Man4Gen-Projekt: Unterstützung von Airline-Crews in unerwarteten Situationen. In: *Proceedings of the 2016 German Aerospace Congress (DLRK)*. Braunschweig, Germany. 2016; .
- [50] Buch JP, Niedermeier D, Stepniczka I. Managing the Unexpected. AIAA Modeling and Simulation Technologies (MST) Conference, AIAA AVIATION Forum, AIAA 2017-4155. Denver, CO, USA: American Institute of Aeronautics and Astronautics, Inc. (AIAA). 2017; .

An Adaptable Full-Scale Aircraft Cabin in an Interconnected Simulation Environment

Mario Kallenbach, Stephan Kocks, Paul Frost, Ingo Voissel, Peter Hecker

Institute of Flight Guidance, Technische Universität Braunschweig, Hermann-Blenk-Str. 27, 38108 Braunschweig, Germany; m.kallenbach | s.kocks | p.frost | i.voissel | p.hecker@tu-braunschweig.de

Abstract. An aircraft cabin is – apart from interactions with the cabin crew – the most important interface between an airline and its passengers. The Institute of Flight Guidance (IFF) of the Technische Universität Braunschweig developed and built a full-scale aircraft cabin section comprising the structure and electronic systems of a short- to medium-haul single-aisle commercial aircraft. In this paper, we describe the ongoing work of setting up the cabin simulation and the developed solutions to a variety of technical challenges. We present our structural hard- and software architecture focusing on high realism or plausibility while maintaining a high flexibility in the design for future developments. By means of two more detailed examples – an outside view simulation system and an interconnection network for the simulation environment – we show our approach.

Introduction

The most lasting impressions (positive or negative) of a past flight come from aspects related to the aircraft cabin interior. Apart from pre-flight experiences and crew service quality, studies (e.g., [1] [2]) show that cabin systems have a high impact on the revenue of an airline through passenger comfort and the resulting desire to re-book with the same airline and/or the same aircraft. Although the cabin is partly subject to high safety requirements, its influence on the safe execution of the flight is mainly secondary. Thus, it offers more chances for the successful introduction of new developments than other parts of the aircraft structure. These aspects are contributing to the fact that aircraft cabin systems increasingly receive attention both in research and economic contexts.

New procedures and systems aim at the sometimes contradicting aspects of increased safety, greater operational efficiency and an improvement in passenger comfort. For their validation, models and simulations are

required since a test during real-life operation is often too time- and cost-intensive. However, the multitude of cabin systems and the necessity of simulating a large number of influences on the operation in the cabin, pose a variety of challenges for the technical simulation.

In order to meet the resulting requirements of educational and research purposes, the Institute of Flight Guidance (IFF) of the Technische Universität Braunschweig developed and built a full-scale aircraft cabin section of a short- to medium-haul single-aisle commercial aircraft. The fixed-base cabin section is attached to an aircraft cockpit simulator and embedded into a superordinate simulation environment.

In this paper we describe the ongoing work of setting up the cabin simulation and the developed solutions to the mentioned technical challenges. We present our structural hard- and software architecture focusing on high realism or plausibility while maintaining a high flexibility in the design for future developments.

As a detailed example, we show the currently implemented outside view simulation and discuss the technical and perception-based challenges faced during development. Furthermore, the results of an examination with multiple test persons are featured.

Our subsequent outlook on future work contains ideas on an approach for data exchange. We envisage a publish-subscribe-based architecture to not only connect the individual components within the cabin simulator, but to also enable the interaction with other participants within the surrounding simulation environment as well as in external networks.

1 Simulator Architecture

The fixed-base aircraft cabin simulator comprises the structure and electronic systems of the entrance area including the forward galley and lavatory as well as the full lining of the the first three to four seat rows in the passenger cabin deck (cf. figure 1). It is equipped with

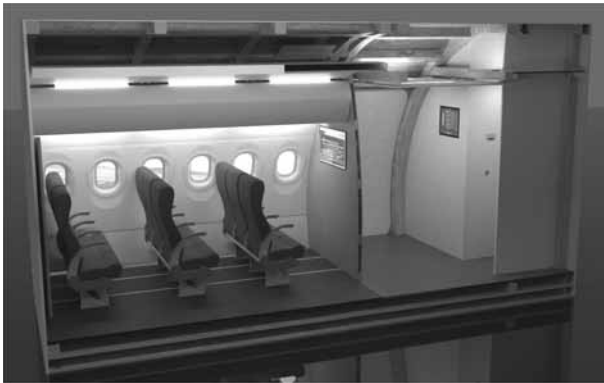


Figure 1: Computer-rendered representation of the cabin simulator cross-section

relevant cabin systems and can be flexibly adapted to test systems from current research. The focus is on networking, sensor technology and human-in-the-loop simulations for the optimization of cabin-related processes. The cabin simulator is spatially adjacent and electronically connected to the aircraft cockpit simulator of the IFF and therefore constitutes part of a full-scale airliner simulation platform, which is orientated on an Airbus A320. This platform in turn is interconnected to other simulation systems and embedded into an air traffic management simulation environment consisting of a Diamond DA42 simulator and two working positions, each of which can be operated either by a (pseudo) pilot or an air traffic controller. This enables us to not only investigate the processes and interactions between cockpit and cabin, but also to consider the cabin as a further entity within the overall air traffic transport system.

1.1 Structure

The integration and evaluation of new and further developments requires a continuous adaptation of the cabin structure. For this reason, a modular approach was chosen, which makes it possible to flexibly modify or replace cabin modules, parts and systems. Furthermore, in order to give test persons a realistic impression, great importance was attached to reproducing the look and feel of the interior as true to original as possible and in particular the inner dimensions. The outer dimensions of the cabin are slightly larger in order to provide more space behind the lining components.

The basic structure of the cabin simulator is fully composed of wooden ribs and beams in combination

with wooden panels and walls. Due to a reinforced floor structure, the cabin simulator can be easily lifted and moved. The outer shell is made of large PVC sheets. The cut-outs of the cabin doors on both sides are already in place, while the actual doors have already been designed, but not yet manufactured. In total, the cabin is approximately 5.6 m long, 4.2 m wide and 3.0 m in height and features twelve cabin windows (six on either side), which however do not have an opening to the outside.

For the lining of the passenger area, mostly used but refurbished original parts – such as window panels, ceiling panels and overhead bins – were incorporated. The total of 18 passenger seats (three groups of three seats on either side) are used originals as well and have a fold-out cocktail table to establish a business configuration. All other interior parts (e.g. the floor carpet) have been selected or replicated to resemble their counterpart in reality. Two self-developed bulkheads separate the passenger area from the entrance area. Except for the forward galley on the right-hand side, the entrance area was reconstructed for the most part. The lavatory on the left-hand side in the passage to the flight deck is only a dummy. It contains the computers and most of the other hardware components for controlling the cabin simulator as described in the following section.

1.2 Hardware

The cabin hardware offers input and output devices as human-machine interface for cabin operation on the one hand and on the other hand provides computing power for the simulated systems on board as well as for the environment simulation. In addition, it provides the basis for stimulating human senses through the visual and auditive channel; the haptic stimulus results from the structure and interior. The cabin simulator has its own power grid, which is divided into sub-grids by a main distributor. For cabin operation, five computers are used that are connected via Ethernet: two high-performance computers, each with two graphics cards for the outside view simulation and ambient noise as well as one computer each for internal audio and video streaming, for the Cabin Management System (CMS) and Flight Attendant Panel (FAP) and for the provision of various interfaces and aircraft buses (e.g. ARINC 429, RS232, RS485, CAN, etc.).

For the illumination of the cabin simulator, pairs of bright white and RGB LED strips were installed that are controlled independently via Digital Multiplex (DMX)

dimmers. Within the passenger area, the LED strips are located on both sides of the ceiling panels, below the overhead bins and on floor level; within the entrance area, multiple pairs of LED strips are mounted in the ceiling only.

The output of cabin internal audio signals takes place via structure-borne sound speakers mounted on the ceiling panels. A separated surround system hidden behind the lining is used for the aircraft and ambient sounds. To enable the outside-view simulation (see section 2), one screen is mounted behind each of the twelve cabin windows, three of which are connected via daisy chain to one graphics card of the high-performance computers. A large-scale screen is also installed in each of the two bulkheads for information and In-Flight Entertainment (IFE). Additionally, a touch-screen is available attached to the lavatory wall within the entrance area. The installation of a ventilation system for the air conditioning of the cabin is currently in progress. While the air conditioning unit and the ventilation outlets are already installed, the individual components are not yet connected.

1.3 Software

Building on the hardware and structural features, the systems and functions of the simulator are represented in a set of programs and software parts. These are classified into mock-ups, device simulation modules and auxiliary modules. Mock-ups mimic real software which is available in a real operational context. An example of a mock-up software is the FAP software implemented for the simulator. Device simulation modules imitate the behavior as well as the usage of cabin hardware – e.g. lavatory equipment. Auxiliary modules, on the contrary, do not match any real world software or device. They are necessary to provide interfaces between virtual and real hardware, provide functions for the simulation control and serve as abstraction layer between cabin systems and simulator systems.

While the mock-up software and the auxiliary modules are implemented as stand-alone binaries, interconnected via TCP/IP-based interprocess-communication, device simulation modules heavily depend on a simulation framework we named Cabin Simulation Environment (CaSE). CaSE provides such features as event-based data exchange, scenario generation and replay as well as lock-free concurrent interfaces to a variety

of data buses. This ensures that new simulation devices can be implemented as light-weight components enabling developers to focus on remodeling the real device's behavior and providing seamless integration.

Device simulation modules may both simulate the behavior of a device and the usage of cabin systems by passengers and crew. Respective scenarios, triggering actions of virtual humans in the cabin and in turn leading to reactions by device simulation modules, can be generated and replayed using the graphical user interface of CaSE. For example, modules are available for the usage of lavatories (door locks, call buttons, water tank levels) or temperature control. Thus, as shown in figure 2, CaSE expands the simulation from the first three seating rows to a full aircraft cabin layout by providing virtual additions to the real hardware.

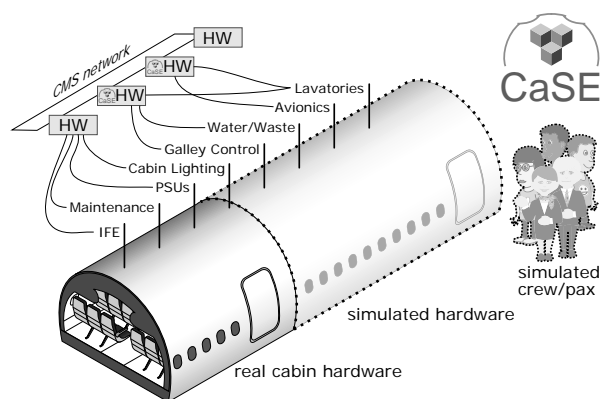


Figure 2: Schematic representation of the cabin simulator comprising the real and simulated hardware of the cabin interior, cabin systems and their usage

Central component of the whole software architecture is the mock-up software for the CMS. It provides information for and communication between systems and takes control of all connected systems in the cabin. The interface between this management system and the flight crew is the FAP. Its main purpose is to provide a graphical user interface for interacting with all systems on board and to be the link to the CMS. The FAP software is visible at the touch-screen monitor in the front section of the cabin (cf. figure 1), which is also the corresponding input device.

The auxiliary modules, generally also connected via the CMS, serve as interfaces and converters to their respective connected hardware. For example, the lighting module provides pre-defined mood lighting scenarios which can be selected at the FAP. The resulting

request is sent via the CMS to the lighting module, where it is converted into an ARTNet-compliant message which triggers the DMX dimmer hardware. Another auxiliary module serves as media streaming device for the IFE and passenger announcement system. Thus, it grants access to the bulkhead monitors and the structure-borne sound speakers (cf. section 1.2) for video and audio transmission into the cabin.

2 Outside-View Simulation

The visual perception is one of the key senses to be stimulated for a high immersion of the participants of studies. This does not only apply to the interior, where materials, forms and colors have to match the past experience of the passengers. Instead, it applies especially to the view of the outside world which is provided through the cabin windows. Because of the fixed-base character of the simulator, this is the main way to create the impression of motion of the simulated aircraft. On the other hand, errors and misconceptions of this simulation component may easily lead to discomfort or even Simulation Sickness Syndrome (SSS).

Our 12-monitor-setup and twelve independently generated views provide a realistic outside view for test persons. Apart from bandwidth concerns and image generation performance, the selection of the depicted content and the implementation of optically correct perspectives were addressed while developing the system. The main challenge the hardware setup poses is the contradiction between multiple persons looking from different positions and directions to the same fixed, 2D representation of a 3D outside view. However, we expect that deviations from reality are easily accepted if a plausible view is obtained that matches the test person's past experience. We targeted at exploiting this feature by averaging the views to minimize the perceptible deviations between the simulation and reality. We used human-in-the-loop simulations with multiple test persons to evaluate the level of immersion as well as the impact on the involved humans with respect to visual perception of movement while lacking a kinesthetic stimulus.

2.1 Architecture and Setup

The system uses the hardware architecture described in section 1.2 with twelve screens in the outer walls and can therefore access a total of two CPUs and

four GPUs for image generation. The used software strongly influences the available visual content, performance and synchronizing features. Thus, the experiments were conducted using varying software platforms, more specifically, Laminar Research's X-Plane 11, Prepar3D by Lockheed Martin and – deviant from using a flight simulator software – Google Earth Pro.

The two flight simulation programs were set up to have one master instance, calculating the flight status vector using the flight model and three to five slave instances which only generated synchronized views with different viewing angles of the environment. Both flight simulations provide configuration files for setting up the correct parameters of the outside view. In case of Prepar3D, multiple views can be specified together. For X-Plane, different instances of the software have to be launched. However, this also allows for the separate instances to use different associated GPUs, balancing the load of image generation across the available hardware. Another challenge is synchronicity and temporal resolution, which is not only crucial for holding up the illusion of steady movement. When using different software instances, information like the shape and position of clouds, ground vehicles and other traffic must be interchanged or propagated. Both simulations use TCP/IP-based proprietary protocols for that purpose.

Google Earth Pro instead was connected to the flight simulation of the adjacent cockpit simulator used as data provider and is only capable of one sole view which was stretched across the screens. Though we tested this setup, we did not consider it for the evaluation as the huge drawbacks of this approach stood out at an early stage.

2.2 Viewing Layouts

In order to consolidate and average the multitude of possible viewing directions and positions to a set of views that can be shown on the different screens, we considered different combinations of number of viewing points, number of independent views of these points and positions of the viewing points. These layouts, of which three were later used for the evaluation, are defined in table 1.

Each layout entails advantages and disadvantages concerning feature visibility, degree of realism, generation performance, etc., depending on the passengers position in the cabin and the simulated flight phase. Although the according considerations were made, the detailed discussion exceeds the scope of this paper.

	Viewing Points (VP), in total	Views per VP	VP Height
Row layout	6	2	1175 mm
Aisle layout (high)	1	12	1570 mm
Aisle layout (low)	1	12	1033 mm
Duplicated layout	1	2	1175 mm
Single layout	6	2	1175 mm

Table 1: Configurations of view layouts

In figure 3, the different viewing layouts resulting from varying the input parameters are schematically depicted. The chosen layouts determine the optical parameters of the system, which in turn can be calculated and implemented in the used software.

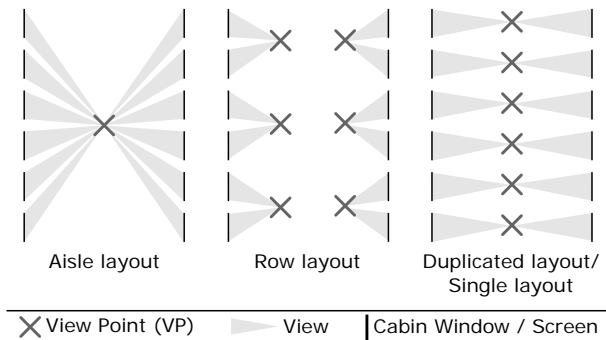


Figure 3: Schematic depiction of view layouts in the cabin simulator (top view)

2.3 Evaluation

For the evaluation of our solution, we conducted human-in-the-loop tests including four different scenarios, each with a duration of approximately 15 minutes. Each scenario contained a short taxi phase, take-off phase and a climb phase up to a virtual altitude of 12000 ft. While climbing, the aircraft performed both a left and a right turn. In the different scenarios, the viewing layout and the software platform were varied, see table 2 for reference.

The 34 test persons were asked to choose a seat and fill in a questionnaire asking for grades from 1 to 10 concerning the assessment of realism, image quality, temporal resolution, synchronicity, impression of movement and personal well-being taking the flight phase into account. Also, test persons were asked to look at the outside view from different positions in the

	Viewing layout	Software
Scenario 1	Row layout	X-Plane
Scenario 2	Aisle layout (high)	X-Plane
Scenario 3	Aisle layout (low)	X-Plane
Scenario 4	Aisle layout (low)	Prepar3D

Table 2: Input parameters for the evaluation scenarios

cabin, at least once from their seat and once from the aisle. Combined with (optional) test person specific information concerning age, gender, body height as well as frequency of flight, this produces comprehensive data for the evaluation of the implemented system.

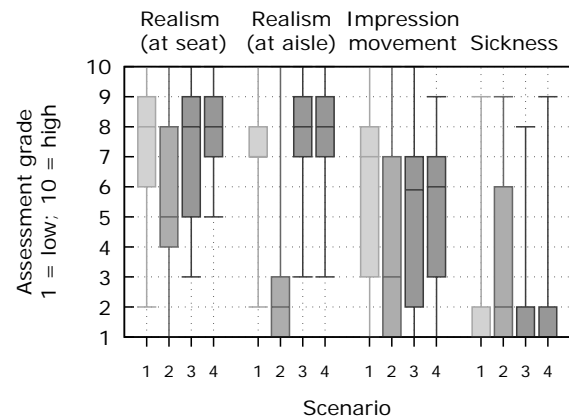


Figure 4: Grades concerning assessment of realism (while viewing from seat or aisle), of impression of movement and of individual sickness

As the excerpt from the results in figure 4 indicates, the system generally provides a good depiction of reality. The assessment concerning the impression of movement shows potential for improvement as the grades are on an average level. This feedback was to be expected since the simulator does not provide kinesthetic stimuli in the form of perceptible accelerations. Despite this missing sensation, which contradicts the test persons' observations, the experienced sickness level is low except for scenario 2. In general, scenario 2 shows the worst performance, especially when the simulation is viewed from the aisle. Test persons criticized the high visual tilt downwards in that scenario.

The overall results from the extensive study substantiated our tendency towards scenario 3. Thus, we selected this scenario for operation.

3 Interconnection Network

The interconnection of the simulators described in section 1 should enable a composite simulation. Each simulator will be considered as a single entity within the simulation structure. Nevertheless, the cabin and cockpit simulators constitute a single entity in most of the use cases. Currently, each entity operates within its own network and – if required – the intercommunication is fulfilled via dedicated interfaces.

3.1 Requirements

The aim of the next stage of development is to provide a more flexible connection between the entities and other modules. Therefore, it shall be possible to operate each entity individually and as a part of a composite simulation. For a quick integration of external modules, the interfaces have to be easily accessible. However, the interfaces must keep step with the development of systems.

3.2 Envisaged Implementation

In order to meet the requirements and to avoid the occurrence of single point of failure events, it is foreseen to use a distributed system design such as the Robot Operating System 2 (ROS 2) to connect the simulators. ROS 2 uses Fast Real Time Publish-Subscribe (RTPS) [3] which builds on the Data Distribution Service (DDS) middleware.

Using ROS 2, systems are able to publish data within self-defined topics to provide them to other entities. Participants can subscribe to the topics they require. On the application layer there is no further communication necessary between the entities and there is no central instance required for controlling the communication flow. Discovery of the participants is also handled transparently by the middleware.

Each topic holds one data type. Custom data structures have to be defined in Interface Definition Language (IDL) files provided for every system at compile time. It is envisaged to only use primitive data types as well as their corresponding arrays at first. Therefore, most of the data fields provided by our simulators are to be transferred within individual topics. Complex data types will be defined subsequently, if needed.

Since Fast RTPS internally uses UDP/TCP as transport protocol, it is mandatory that the subnets of each simulation entity will be connected within the Local

Area Network (cf. figure 5). ROS 2 uses one DDS domain where all topics are available. Each module has to be part of this domain in order to subscribe to a topic and to automatically discover other systems. A router at the top level of the topology interconnects the subnets and acts as an internet gateway.

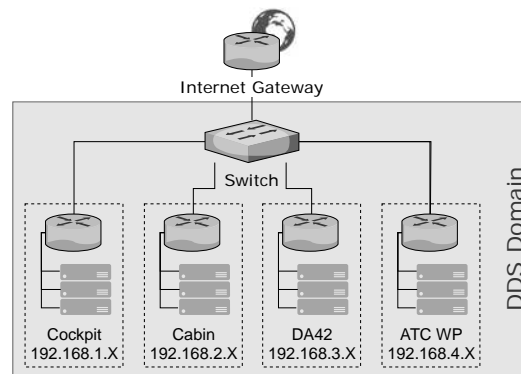


Figure 5: Envisaged network topology

4 Conclusion

We presented our full-scale aircraft cabin section comprising the structure and electronic systems of a short-to medium-haul single-aisle commercial aircraft. The development process as well as the challenges and their solutions were illustrated on the basis of examples, particularly the implementation of an outside view simulation. The subsequent evaluation showed promising results concerning the level of immersion generated by the simulator. The next step will be the development of an interconnection infrastructure between the simulator entities.

References

- [1] Richards LG. On the Psychology of Passenger Comfort. *Human Factors in Transport Research*. 1980;2:15 – 23.
- [2] Vink P, Bazley C, Kamp I, Blok M. Possibilities to improve the aircraft interior comfort experience. *Applied Ergonomics*. 2012;43(2):354 – 359.
- [3] Kwon G, Hong J, Lee T, Lee W, Park J, Tak T. Development of Real-Time Data Publish and Subscribe System Based on Fast RTPS for Image Data Transmission. In: *Proceedings, 16th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS 2017): Barcelona, Spain, October 8-13, 2017*. 2018; pp. 473 – 477.

Instructor Operator Services for Real-time Flight Simulators

Md Nizam Uddin^{1*}, Umut Durak^{1, 2}, Jürgen Gotschlich², Sven Hartmann¹

¹TU Clausthal, Department of Informatics, Julius-Albert-Str. 4 38678 Clausthal-Zellerfeld

²German Aerospace Center (DLR), Institute of Flight Systems, Lilienthalplatz 7, 38108, Braunschweig

*nizam.uddin@stud.uni-hannover.de

Abstract. Instructor Operator Stations are generally tightly coupled systems with the rest of the flight simulators and their users have limited to no means for customization, enhancement and extensions. Service-Oriented Architecture that ensures service reuseability, improved integration capabilities, scalability and better development cycle, can play an important role to draw a solid line that decouples Instructor Operator Stations from the rest of the simulation systems. In this study, Service-Oriented Architecture is adapted for real-time flight simulation applications. Instructor Operator features including user interfaces are developed as services. These services are then deployed inside containers as they provide good isolation with low overhead and easy deployment.

Overview

1 Introduction

Flight simulators are usually monolithic and proprietary, hence hard to maintain, bring changes or migrate. Non-standard communication protocols among simulator subsystems only amplifies the problem. Inevitably the maintenance effort increases. So bringing a change to the existing simulator requires more time and effort.

Instructor Operator Station (IOS) is a flight simulator subsystem that consists of the displays and controls that are used by the instructor to interact with the training system [1]. Not different from many other flight simulator subsystems, it also suffers from proprietary, non-standard, monolithic design.

This study aims at investigating an open flight simulator architecture exploiting Service-Oriented Architecture (SOA) [2] [3] and standard Representational State Transfer (REST) [4] communication protocol. Docker¹ based containerization [5] is proposed for further standardization in development and deployment.

¹<https://www.docker.com/>

The paper presents how this open architecture is implemented to provide some features regarding simulation data access and basic simulation control via user interface (UI) components [6] as services for an IOS. The challenge however is the real-time characteristics of flight simulators. The paper further explains how the real-time distributed simulation framework of German Aerospace Center (DLR), 2Simulate [7], is extended to support SOA. The proposed approach is demonstrated using DLR Air Vehicle Simulator (AVES) Software Development Kit (SDK) [8] at TU Clausthal Aeronautical Informatics Laboratory on Research Aviation Training Device (RATD) [9].

2 IOS Services

The services that provides features and functionality for IOS and controls and manages the real-time flight simulation at run-time are named as the IOS services. They can be grouped into three different categories:

- Simulation system services
- Generic UI services
- User-defined UI services

To achieve fine grained service orientation, above categorized IOS services are proposed as microservices [10]. Thereby, it would be possible to develop, deploy and test atomic features independently.

For service orientation, first the simulation system shall support web services; this is enabled by simulation system services. UI services will then be able to invoke the simulation system services for simulation control and data exchange so that they can interact with the simulator subsystems. The UI component pool will contain several types of UI components. Each UI component is available as microservice so that the composer

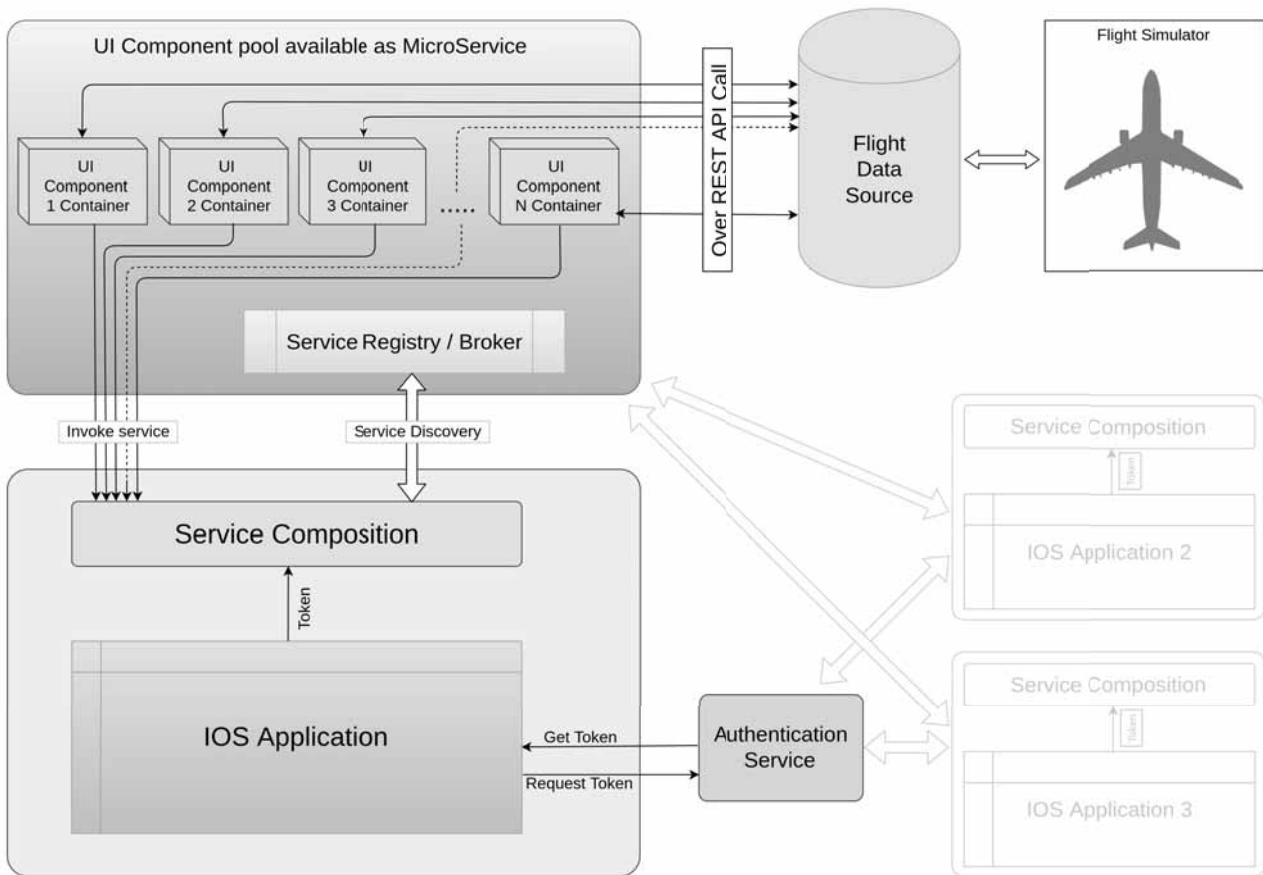


Figure 1: Service-Oriented Architecture for IOS Application

can invoke them and compose a complete IOS specific to the need.

3 Architecture Overview

Figure 1 presents the service architecture. The dark gray box on upper left represents the UI component pool available as microservice, there exists the Service Registry or so called service Broker, that tells any IOS application about the available UI components and how to invoke them as microservice. These UI components get data from the flight data source over REST API call. The light gray box in the bottom left represents a complete IOS application. The service composer checks with the service broker to know about the available services in the UI component pool and invokes them as needed for that particular IOS application. Then the service composer composes these UI components into on complete IOS application.

Figure 2 presents the work flow. It explains how authentication, services discovery and finally services are invoked to be composed into a ready to use IOS application in our approach. This is an abstract overview of each step.

4 Simulation System Services

Simulation system services are enabled by enhancing the DLR real-time distributed simulation framework 2Simulate with RESTful web service support. 2Simulate is a C++ framework to allow deterministic scheduling and controlling of real-time tasks. Real-time tasks have a wide range of requirements, e.g. control tasks as well as a wide range of data connections to external devices [7]. 2Simulate is a part AVES SDK. A typical simulator architecture that utilizes AVES SDK has a central data exchange node, an interface computer,

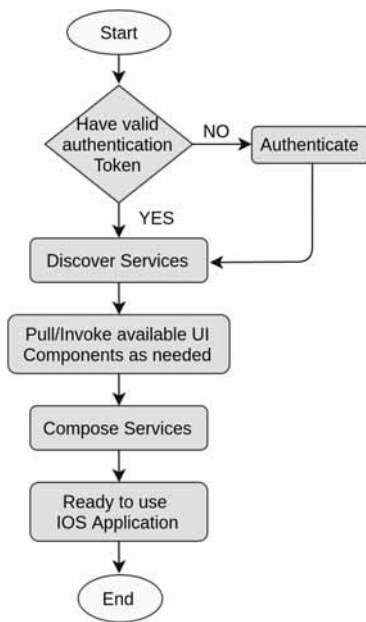


Figure 2: Flow Diagram of Service-Oriented IOS Application

which is developed using 2Simulate [8]. Enhancing 2Simulate with RESTful web services enables creating a RESTful web service capable interface computer. Thereby, simulation control and data access can be accomplished via services.

We used BOOST 1.66.0² with ASIO [11] for the REST integration into 2Simulate. Thereby an HTTP server is implemented inside 2Simulate as a utility on a non periodic thread. The clients of this server are the services of the next two category. While the server does not provide any real-time guarantees, using REST API clients can have a read/write access to the data within the real-time simulation.

5 Generic UI Services

Generic UI services provides flight simulator independent basic functions for any 2Simulate user. Two examples are simulation control and simulation status. Simulation control, as its name implies, controls the simulation system by making a POST request to the corresponding simulation system service. It can change the simulation system status into three different states: init, run and pause. Simulation status on the other hand uses a GET request and it displays the current simulation status.

²<https://www.boost.org/>

6 User-Defined UI services

While the first two categories of services are readily available, the proposed architecture also provides a mechanism to define new UI services at runtime. The UI Service Designer first gives the user an option to provide correct address and endpoint for both service registry and simulator. It then connects to the simulator to get the list of available shared variables, signal and sensor data. Based on the list, the UI Service Designer will provide the user option to select the desired data set to be able to see as a UI component in the IOS application at run-time. The user saves the UI component configuration and the UI Service Designer send the information to the service registry to add this new custom component to UI component service list. Figure 3 depict the flow diagram how the UI Service Designer works. The output of UI Service Designer is an configuration object which is then used for the next step, the UI Service Renderer. The UI Service Renderer interprets the configuration object generated by UI Service Designer.

The UI Service Renderer takes the UI Service Designer's output, the configuration object as input argument for itself. From the visual perspective, it follows a predefined dynamic component rendering template that composes the UI layout according to the configuration object dynamically at run-time.

7 Service Registry

Service registry keeps track all the components available. IOS application's service composer can ask the service registry to know about the available services and invoke them accordingly. As the user-defined UI services are created on the fly at run-time, they need to register themselves to the service registry as well. This service registry itself have two separate parts. There is a service registry manager and a service registry server. Service registry server is used to register all the services regardless of their creation time at run-time or prebuild prior before the simulation started. The service registry manager on the other hand is responsible for updating the service configuration at a later point of time if there was any mistake or needs to update because of some update in the component itself who registered. It can also do some other basic operations like setting a component as active/inactive, edit/update them, change source of

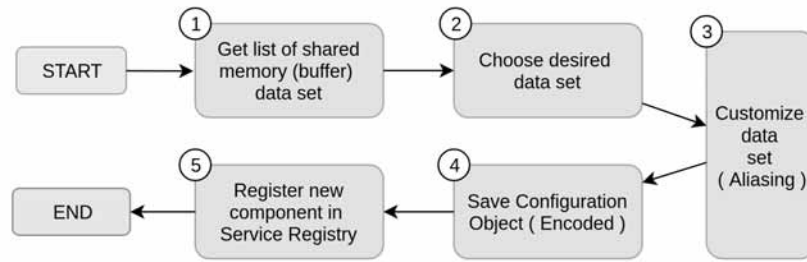


Figure 3: The UI Service Designer Flow Diagram



Figure 4: Demonstrator IOS

data or even delete them. To better understand how service registry manager and service registry server works together, below is a visual representation of how other components uses service registry in different use cases and how service registry itself works.

Any component is registered to the service registry with a configuration object and an endpoint. Then the list of available services are used by other composers to invoke these components and build their own IOS application.

8 Demonstrator

The RATD system is mainly designed and developed to provide a low-cost flight training device. The design of the RATD system architecture is based on AVES SDK [9]. For the demonstration purposes RATD Interface Computer is updated using RESTful web service capa-

ble 2Simulate. Then a sample IOS application is developed using all three categories of services. Figure 4 presents a screen shot from the demonstrator.

9 Conclusion

This research investigates the Service Oriented Architecture for flight simulators on an Instructor Operator Station use case. Proposed solution provides UI components as Instructor Operator services, covering from generic simulation control until custom user-defined on the fly UI component creation ability to add new UI components as service. IOS application development and maintenance is eased with service composition. A demonstrator is realized for Research Aviation Training Device at TU Clausthal Aeronautical Informatics Laboratory. This effort provide evidences that an open architecture flight simulators are possible utilizing service

orientation. There are readily available libraries, such as BOOST ASIO that can be well integrated to simulation frameworks to address real-time requirements. Future work is in two folds. The proposed solution will be employed at German Aerospace Center (DLR) Simulation Center AVES and the service orientation will be expanded to other simulator applications such as flight dynamics models.

References

- [1] Vazquez AA. Touch screen use on flight simulator instructor/operator stations. *Tech. rep.*, Naval Postgraduate School. 1990.
- [2] O'Brien L, Merson P, Bass L. Quality attributes for service-oriented architectures. In: *Proceedings of the international Workshop on Systems Development in SOA Environments*. IEEE Computer Society. 2007; p. 3.
- [3] Schneider S. What is real-time SOA? *Real-Time Innovations, Inc, Sunnyvale, CA*. 2010;94089.
- [4] Feng X, Shen J, Fan Y. REST: An alternative to RPC for Web services architecture. In: *First International Conference on Future Information Networks*. IEEE. 2009; pp. 7–10.
- [5] Kang ML Hui, Tao S. *Container and microservice driven design for cloud infrastructure devops*. IEEE International Conference on Cloud Engineering (IC2E). 2016;.
- [6] Brown SJ Alan, Kelly K. *Using service-oriented architecture and component-based development to build web service applications*. Rational Software Corporation 6. 2002;.
- [7] Gotschlich J, Gerlach T, Durak U. 2Simulate: A distributed real-time simulation framework. In: *ASIM STS/GMMS Workshop*. 2014; .
- [8] Gerlach T, Durak U. AVES SDK: Bridging the Gap between Simulator and Flight Systems Designer. In: *AIAA Modeling and Simulation Technologies Conference*. 2015; p. 2947.
- [9] Wang H, Chen S, Durak U, Hartmann S. Simulation infrastructure for aeronautical informatics education. In: *Proceedings of the 50th Computer Simulation Conference*. 2018; .
- [10] Thönes J. Microservices. *IEEE software*. 2015; 32(1):116–116.
- [11] Torjo J. *Boost. Asio C++ network programming*. Packt Publishing Ltd. 2013.

Simulation of RPDEVS Models of Logic Gates

Christian Fiedler^{1*}, Franz J. Preyser¹, Wolfgang Kastner¹

¹Institute of Computer Engineering, Automation Systems Group, TU Wien, Treitlstraße 1-3, 1040 Vienna, Austria
*christian.fiedler@student.tuwien.ac.at

Abstract. This paper addresses the simulation of fundamental logic gates (e.g. AND, OR, NOT) using the software *PowerRPDEVS* that is based on the *Revised Parallel Discrete Event System Specification* (RPDEVS) formalism. The formal differences of the models of a NOR gate in RPDEVS and PDEVS are analyzed. It is further shown, which possible pitfalls may occur when connecting these logic gates with feedbacks that cause algebraic loops and in which cases these algebraic loops are resolved by the RPDEVS simulation algorithm. For this purpose, a static RS flip-flop, a triggered D flip-flop and a shift register are modeled and simulated in *PowerRPDEVS*. The results are compared to previous research about the simulation of such logic circuits in *Simulink* and *Modelica*.

Introduction

In the theory of computation, the notion of Mealy and Moore automata exists which are forms of finite state automata [1]. The difference between these two types is how the output function is defined. The output function of the Moore type depends only on the internal state of the automaton, whereas for the Mealy type it also depends on the automaton's input. The formal definition of an automaton has a lot in common with the modeling formalism *Discrete Event System Specification* (DEVS) [2] and thus, also with its extension the *Parallel Discrete Event System Specification* (PDEVS), introduced by Chow and Zeigler [3]. However, since the output function in PDEVS depends only on the internal state, in principle, in PDEVS only Moore behavior is supported [4]. For Mealy behavior, a workaround including a *transitory state* (i.e. a state with zero life time) is necessary. This means that PDEVS models which immediately react to an external event with an output first have to enter a transitory state before the output function can be used to set the output.

In *Revised Parallel Discrete Event System Specification* (RPDEVS), introduced by Preyser et al. [5], the formalism was restructured to support Mealy behavior naturally. In RPDEVS immediate reactions to external

events can be modeled directly with the output function, which removes the need for transitory states in this context.

As for PDEVS, an abstract simulator for RPDEVS was defined and published in the accompanying work [6]. An implementation is provided with the program *PowerRPDEVS* that also includes a graphical model editor.

With the goal to extend the *PowerRPDEVS* model library, we created a library with combinational logic and sequential logic elements. Combinational logic gates output the result of a boolean operator applied onto the input values. Thus, when signal delays are not taken into account, their models are all of type Mealy. Sequential logic components, in practice, are usually designed by coupling combinational logic elements with storage elements [7]. As Junglas' findings in [8] show, this can be a cumbersome business in simulation tools.

In this work, first it is analysed how the RPDEVS models of logic gates differ from the corresponding PDEVS models. Afterwards, it is investigated how the RPDEVS simulation algorithm performs when creating sequential logic components and the results are compared to Junglas' work.

1 The PDEVS and RPDEVS Formalisms

PDEVS and RPDEVS are hierarchical modelling formalisms where models are composed of *atomics* and *couplings*. An atomic can receive inputs from other atomics or couplings, it has an internal state and can produce outputs. Couplings can be composed of atomics and other couplings. The formal definition of a coupling is omitted here, it can be found in [3] and [6].

1.1 Atomic PDEVS

In Equation (1) the definition of a PDEVS atomic is given as tuple.

$$A := \langle X^b, Y^b, S, \delta_{ext}, \delta_{int}, \delta_{conf}, \lambda, ta \rangle \quad (1)$$

X^b ... set of possible input bags

Y^b ... set of possible output bags

S ... set of possible (internal) states of the atomic

$\delta_{ext} : Q \times X^b \rightarrow S$... external state transition function
where $Q = \{(s, e) | s \in S, e \in [0, ta(s)]\}$

$\delta_{int} : S \rightarrow S$... internal state transition function

$\delta_{conf} : S \times X^b \rightarrow S$... confluent state transition function

$\lambda : S \rightarrow Y^b$... output function

$ta : S \rightarrow \mathbb{R}_0^+ \cup \{\infty\}$... time advance function

The time advance function ta determines the time to live $ta(s) \in [0, \infty]$ for every internal state $s \in S$. Whenever this time expires, an internal event is triggered, which first causes the execution of the output function λ and then leads to a state transition conducted by δ_{int} . However, if at the same time an input event occurs, the state transition is performed by δ_{conf} . If the atomic is not *imminent* (i.e. it has no internal event) while an input event x^b is received, δ_{ext} is called and the atomic changes into a new state $s' = \delta_{ext}(s, e, x^b)$ without producing any output. The λ function that sets the output of the atomic is only evaluated right before an internal state transition and relies on the old state of the model. Thus, when an atomic has to respond to an input with a change in output, there has to be a state transition in δ_{ext} (or δ_{conf}) into a transitory state (i.e. a state s' with $ta(s') = 0$). This way, λ can set a new output at the same point in simulation time.

1.2 Atomic RPDEVS

$$A := \langle X^b, Y^b, S, \delta, \lambda, ta \rangle \quad (2)$$

$\lambda : (Q \times X^b) \rightarrow Y^b$... output function

$\delta : (Q \times X^b) \rightarrow S$... external state transition function
where $Q = \{(s, e) | s \in S, e \in [0, ta(s)]\}$

In RPDEVS (see atomic definition in Equation (2)), the three state transition functions of PDEVS are merged to one single transition function δ which always is preceded by an evaluation of the output function λ . This evaluation of λ though, happens iteratively. That is, λ is recalculated every time the input bag has changed due to a λ computation in an influencing component. As shown in [5], this λ iteration terminates as long as the model does not contain algebraic loops. Furthermore, it still may terminate if the algebraic loop can be resolved as we will see in Section 1.4. In contrast to PDEVS, λ also depends on the current input bag. Thus, Mealy behavior can be modelled without having to change the internal state. In fact, pure functional blocks can be modeled which do not need an internal state at all, e.g. a logic NOT gate just forwards input messages inverted to the output.

As already mentioned in the introduction, a PDEVS model can be compared to a Moore machine ($\lambda(s)$), whereas an RPDEVS can be compared to a Mealy machine because its output is a function of the input and the internal state ($\lambda(s, e, x^b)$).

As already mentioned in the introduction, a PDEVS model can be compared to a Moore machine ($\lambda(s)$), whereas an RPDEVS can be compared to a Mealy machine because its output is a function of the input and the internal state ($\lambda(s, e, x^b)$).

1.3 NOR gate

A NOR gate with two inputs is constructed in PDEVS and RPDEVS to demonstrate the differences with an example in the context of logic gates.

$$NOR_{PDEVS} := \langle X, Y, S, \delta_{ext}, \delta_{int}, \delta_{conf}, \lambda, ta \rangle \quad (3)$$

$$X = \{1, 2\} \times \{0, 1\}$$

$$Y = \{0, 1\}$$

$$S = \{0, 1\}^2 \times \{0, \infty\}, \quad s = (s_1, s_2, \sigma) \in S$$

$$\delta_{ext}(s, e, x) = (a_1(s, x), a_2(s, x), 0)$$

$$\delta_{int}(s) = (s_1, s_2, \infty)$$

$$\delta_{conf}(s, x) = \delta_{ext}(s, ta(s), x)$$

$$\lambda(s) = \neg(s_1 \vee s_2)$$

$$ta(s) = \sigma$$

$$a_i(s, x^b) = \begin{cases} x_i & \text{if } (i, x_i) \in x^b \\ s_i & \text{otherwise} \end{cases}$$

Equation (3) shows a PDEVS NOR gate. As an external event could update either both or only one input, the model has to keep the last seen values of its inputs in the internal state. A helper function a_i chooses the new input value from the input bag x^b if there is any, or otherwise the saved value from the internal state. The model also keeps the value σ for ta , which is set to 0 in the case of an external event. In this way, every external event is followed by a transitory state used to create an output event.

$$NOR_{RPDEVS} := \langle X, Y, S, \delta, \lambda, ta \rangle \quad (4)$$

$$\begin{aligned}
X &= \{1, 2\} \times \{0, 1\} \\
Y &= \{0, 1\} \\
S &= \{0, 1\}^2 \dots s = (s_1, s_2) \in S \\
ta(s) &= \infty \\
\lambda(s, e, x^b) &= \neg(a_1(s, x^b) \vee a_2(s, x^b)) \\
\delta(s, e, x^b) &= (a_1(s, x^b), a_2(s, x^b))
\end{aligned}$$

When looking at Equation (4) which shows the NOR gate in RPDEVS, one can see that, because λ can access the input bag directly, the transitory state is not needed and therefore, the state space is reduced.

1.4 Static RS Flip-Flop

Flip-flops are sequential logic elements. That means, their outputs not only depend on their current inputs but can also on historic input values [7]. This implies that they have an internal state to store the historic data. A static RS flip-flop is commonly built using two NOR gates connected with their outputs fed back to the input of the other (see Figure 1).

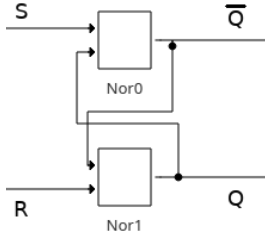


Figure 1: PowerRPDEVS model of the static RS flip-flop composed of two NOR gates.

Notably, the RS flip-flop is constructed from two *combinational* logic elements – the two NOR gates ideally have no state. Thus, deducing the mathematical model from the circuit of Figure 1 results in a system of two implicit equations:

$$Q = \neg(R \vee \bar{Q}) \quad (5)$$

$$\bar{Q} = \neg(S \vee Q) \quad (6)$$

Equations 5 and 6 can be solved as long as the inputs S and R are not both equal to 0 (see Table 1). The defined behavior for a RS flip-flop actually is to keep its previous output values in the case of $R = S = 0$. However, to know the previous value, the system needs to have a memory, i.e. an internal state. A real flip-flop is a continuous system and its signals are exposed to delays. These delays

S	R	Q	\bar{Q}
0	0	$\neg\bar{Q}$	$\neg Q$
0	1	0	1
1	0	1	0
1	1	0	0

Table 1: Solutions of Equations (5) and (6).

cause the system to still know its previous output, when the input changes.

Due to the discrete event nature, our PDEVS and RPDEVS models have to store the last seen input values and, thus, also possess an internal state. When one of the inputs S or R of the RS flip-flop changes, the affected NOR gate still has stored the previous output of the other NOR gate. Consequently, during the simulation of the PDEVS and RPDEVS models, not Equations (5) and (6) are solved, but a recurrence relation. How this recurrence relation looks like depends on the number of inputs that change concurrently and on whether the simulation algorithm works in parallel or sequentially.

Single input change. We now consider the cases in which only one input changes its value.

If input S changes, the upper NOR gate first calculates its output using the new value of S and the stored value for Q . Then, due to the change in \bar{Q} , the lower NOR gate calculates its output, already using the new value for \bar{Q} . Thus, the recurrence relation has the form:

$$Q_n = \neg(R \vee \bar{Q}_n) \quad (7)$$

$$\bar{Q}_n = \neg(S \vee Q_{n-1}) \quad (8)$$

In Table 3 the evolution of the recurrence relation in Equations (7) and (8) is depicted for all possible initial states Q_{n-1} and input values S and R . It can be seen, that in all cases a fix point is reached after at least 2 iterations ($Q_{n+1} = Q_n$). Nevertheless, the simulation of this model in PDEVS leads to an infinite loop. When processing the external event due to the change in one of the two inputs, the affected gate schedules an internal event with $ta = 0$. Then λ sets the output and triggers the other gate for an external event. Finally, δ_{int} sets $ta = \infty$. The other gate is activated though, and will do exactly the same afterwards. This again reactivates the first gate and, thus, the simulation gets stuck in a loop (the model is *illegitimate*). To

Q_{n-1}	S	R	Q_n	\overline{Q}_n	Q_{n+1}	\overline{Q}_{n+1}
0	0	0	0	1	0	1
0	0	1	0	1	0	1
0	1	0	1	0	1	0
0	1	1	0	0	0	0
1	0	0	1	0	1	0
1	0	1	0	0	0	1
1	1	0	1	0	1	0
1	1	1	0	0	0	0

Table 2: Solutions of Equations (7) and (8).

mitigate this issue, the PDEVS model must be extended, such that δ_{ext} of the NOR gates only enters a transitory state when the input bit x_i is different from the already stored bit s_i . It should be noted here, that this is an example for how reusability of PDEVS models is impaired due to the need of transitory states for modeling Mealy behavior.

In RPDEVS, the simulation terminates. The change in the input S directly leads to an execution of λ of the upper NOR gate. The produced output respectively input for the lower gate then triggers λ of the lower NOR. The output produced thereby triggers a recalculation of λ at the upper gate. However, if the newly produced output of the upper gate does not differ from its previous one, the lower gate is not triggered again. Consequently, as long as the recurrence relation reaches a fix point in a finite number of steps, the RPDEVS simulation algorithm will find that fix point and will be able to continue simulation.

The case in which the input R changes and S does not change, is completely analog and, thus, is not described.

Concurrent input change. If both inputs S and R change concurrently, it depends on the simulation algorithm, how the recurrence relation that is solved during simulation looks like. In the case of a sequential RPDEVS simulation algorithm, like implemented in PowerRPDEVS [9], first λ of the first block is calculated. Then λ of the second block is calculated, already using the new output of the first block. Thus, the recurrence relation to solve again is the one of first order discussed in the previous paragraph. Consequently, PowerRPDEVS can handle any concurrent change of both inputs S and R without getting stuck in an endless loop.

However, if the RPDEVS simulation engine works in parallel, that is, it calculates λ of both gates concurrently, the recurrence relation to be solved would be of second order (see Equations (9) and (10)).

$$Q_n = \neg(R \vee \overline{Q}_{n-1}) \quad (9)$$

$$\overline{Q}_n = \neg(S \vee Q_{n-1}) \quad (10)$$

This recurrence relation can become unstable though. When both inputs are 1 and then concurrently change to 0, the outputs start to alternate between 0 and 1.

Q_{n-1}	\overline{Q}_{n-1}	S	R	Q_n	\overline{Q}_n	Q_{n+1}	\overline{Q}_{n+1}
X	X	1	1	0	0	0	0
0	0	0	0	1	1	0	0

Table 3: The recurrence relation in Equations (7) and (8) becomes unstable if S and R change concurrently from 1 to 0.

In PDEVS, a concurrent change of both inputs first leads to an execution of δ_{ext} at both gates, storing the new input in the internal state and setting $\sigma = 0$ to enter a transitory state. The transitory state leads immediately to internal events and thus, to a execution of λ in both gates. It does not matter whether λ of the gates is executed in parallel, or consecutively, because both use the old output of the other one that is stored in the component's state. Therefore, for the PDEVS simulation algorithm the concurrent change of both inputs S and R always leads to the solution of the second order recurrence relation in Equations (7) and (8) regardless whether execution is parallel or sequential.

2 Simulation

Simulation was done in *PowerRPDEVS* which is the proof-of-concept implementation of an RPDEVS modelling environment that includes the simulation engine and a graphical model editor. It is open source and available in [9].

2.1 Static RS Flip-Flop

The model of the RS flip-flop in Figure 1 was simulated with the initial values $Q = 0$ and $\overline{Q} = 1$. The input sequence and results are shown in Figure 2. Contrary to simulation in Simulink [8], no work-arounds are needed and the outputs are not delayed.

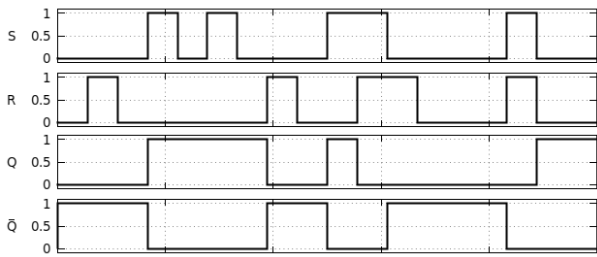


Figure 2: static RS flip-flop – Simulation results

A transition to the "forbidden" state $S = R = 1$ and back to $S = R = 0$ is included. As long as $S = R = 1$, the behaviour is actually well-defined as $Q = \bar{Q} = 0$. When S and R change to 0 simultaneously, the behaviour depends on the ordering of the λ function executions. As mentioned in Section 1.4, parallel execution of the NOR gates' λ functions would cause an infinite loop (oscillation) in the simulation, but it works in *PowerRPDEVS* because the execution is serialized.

2.2 D Flip-Flop

A (clock triggered) D flip-flop can be constructed from a triggered RS flip-flop and additional wiring at its inputs. The atomic `LogicTriggeredSampling` (LTS) was implemented for this example. It can detect edges on its second (lower) input, either triggering for rising edges, falling edges or both, and it either forwards the left ($y(t) = \lim_{\tau \nearrow t} x(\tau)$) or the right limit ($y(t) = \lim_{\tau \searrow t} x(\tau)$) of its first (upper) input when triggered by an edge. This block was placed before the inputs of a static RS flip-flop (see Figure 3). When the LTS blocks are set to take the right limit a change in the input that occurs at the same time as the clock edge is accepted by the flip-flop and it is not accepted otherwise.

It was first tried to use a different trigger detection mechanism: a *falling* block as in [8] in the Modelica model of the triggered RS flip-flop. This did not work out well in RPDEVS though, as the block has to send a 1 for an infinitesimal time frame and then switch back to 0 whenever it detects an edge. This means that because of the event-based nature of RPDEVS, the λ output at the time of the edge would be 1 and the block would need to schedule an internal event to set the output 0. If ta is set to 0 for this purpose a transitory state would be introduced which we are trying to avoid. On the other hand, if it was set to $ta = x \in \mathbb{R}^+$ this would open a time frame during which the flip-flop would accept changes in its in-

puts, although the clock edge occurred in the past. Thus, in the end the triggered RS flip-flop was modeled as depicted in Figure 3.

Figure 4 shows the D flip-flop consisting of the triggered RS flip-flop and a NOT. The results of the simulation of the D flip-flop are shown in Figure 5. \bar{Q} is omitted as it always carries exactly the opposite logic level of Q . The triggering clock edge is set to be the falling edge.

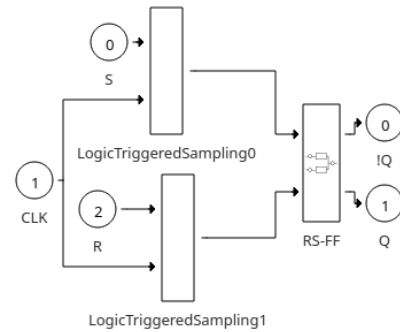


Figure 3: triggered RS flip-flop model

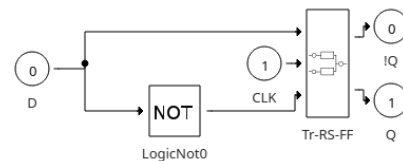


Figure 4: D flip-flop model

During the design of the LTS block we recognized that it is actually a D flip-flop in its own right. The block accepts its first input (corresponding to D) as its output only when there is an edge on its second input (corresponding to CLK) which is exactly the behaviour of a D flip-flop.

Replacing the D flip-flop with an LTS block yields exactly the same results as in Figure 5.

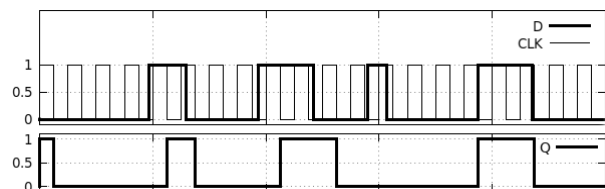


Figure 5: D flip-flop – Simulation results

2.3 Shift register

A shift register is a series of D flip-flops where the input signal is shifted through one flip-flop at a time whenever

the clock input triggers.

The shift register in Figure 6 was modelled by using three of the D flip-flops designed above. Note that for the first flip-flop the LTS blocks (that are part of the triggered RS flip-flop, see Figure 3) is set to use the right limit and for the other flip-flops it is set to use the left limit of the input.

The reason is that the input signal would otherwise travel through all the flip-flops when the first clock edge arrives, because all D flip-flops are triggered by the same clock and none of them delays the signal.

The results of the simulation are shown in Figure 7. They show the individual D flip-flops' output and how the input signal (first a 1, then a 0) propagates through the shift register one stage per clock cycle.

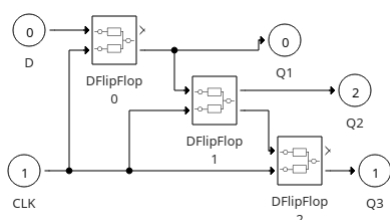


Figure 6: Shift register model

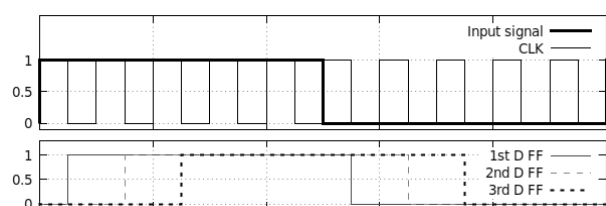


Figure 7: Shift register - Simulation results

In Junglas' tests [8], the Simulink model worked correctly without intervention, but the Modelica model seemed to show a peculiar issue that he mitigated by placing *Pre* blocks between the flip-flops which introduces an infinitesimal delay to break algebraic loops.

3 Conclusion

The *Revised Parallel DEVS* formalism offers new ways to deal with immediate outputs (Mealy behaviour of models) and algebraic loops. Specifically, we discussed a purely functional NOR gate in detail, showing that a model of it in RPDEVS can be realized with a smaller state space than in PDEVS, thus, reducing the complexity of the model. The static RS flip-flop was presented to

show the behaviour of RPDEVS models with a feedback loop with no delay. The result was that a primitive coupling of NOR gates to form an RS latch would almost in any case lead to the expected behaviour of a physical NOR gate, but a transition to the "forbidden" state can lead to oscillation if the simulation engine utilizes parallelism.

The simulation of the RS flip-flop with PowerRPDEVS shows the behaviour that is expected from an RS flip-flop, without the need to introduce delay blocks or arrange it in a special way, which is necessary in other simulators.

The triggered D flip-flop model required the creation of the LTS block that was capable of forwarding an input event exactly when a clock edge occurred which turned out to be a D flip-flop on its own. When they were put together to form a shift register, we needed to take into account the delays that actually make a shift register work and introduce them in our model as infinitesimal delays in the LTS block.

References

- [1] Sakarovitch J, Thomas R. *Elements of Automata Theory*. 2011.
- [2] Zeigler BP, Praehofer H, Kim TG. *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press. 2000.
- [3] Chow ACH, Zeigler BP. Parallel DEVS: a parallel, hierarchical, modular, modeling formalism. In: *Proc. of WSC'94*. 1994; pp. 716–722.
- [4] Joslyn C. The process theoretical approach to qualitative DEVS. 1996; .
- [5] Preyser FJ, Heinzl B, Kastner W. RPDEVS: Revising the Parallel Discrete Event System Specification. In: *Proc. of MATHMOD 2018*; pp. 269–274.
- [6] Preyser FJ, Heinzl B, Kastner W, Breitenacker F. RPDEVS Abstract Simulator. In: *Proc. of ASIM-Workshop Simulation technischer Systeme/Grundlagen und Methoden in Modellbildung und Simulation*. 2019; p. 6.
- [7] Cavanagh J. *Sequential Logic*. CRC Press. 2006.
- [8] Junglas P. Pitfalls using discrete event blocks in Simulink and Modelica. In: *Tagungsband des Workshops der ASIM/GI-Fachgruppen STS und GMMS 2016*. 2016; pp. 90–97.
- [9] PowerRPDEVS on SourceForge. <https://sourceforge.net/projects/powerrpdevs/>. [Online; accessed 27-Dec-2018].

Probabilistic state space models – A theoretical framework with practical relevance

Peter Junglas^{1*}

¹Department of Engineering “Dr. Jürgen Ulderup”, PHWT Vechta/Diepholz, Schlesierstr. 13a, 49356 Diepholz, Germany; *peter@peter-junglas.de

Abstract. Corresponding to the modeling purpose discrete models can be defined using very different approaches: For a precise description and thorough analysis one of the many different mathematical descriptions can be applied, while a working practitioner often will describe a model within a concrete simulation environment. To demonstrate that mathematical models are useful for practical purposes as well, we will present a simple state-space model for a stochastic discrete system. By means of a concrete example we will show, how the use of this model makes the practical modeling process much easier and leads to a more concise concrete implementation.

Introduction

For the description of discrete systems a large number of different mathematical models can be applied, ranging from the simple finite state machine [1] to the complex PDEVs formulation [2]. If one includes stochastic processes, the models get more complicated, well-known examples being the non-deterministic finite automaton [1] and the generalized semi-Markov process [3].

Using mathematical models brings considerable benefits: First of all it allows a complete and precise specification of a model. Secondly the whole machinery of mathematics can be used for the analysis of important properties of the models like the reachability of states, reducibility of the state space or the existence of equilibrium configurations [3].

These points are of a more theoretical nature, but there is a third advantage, often overlooked, which is important for the practitioner: A mathematical model can simplify the actual modeling and implementation in a concrete simulation environment considerably.

To illustrate this point we will present a pedagogical example and implement it in Simulink in a straight-

forward manner. This turns out to be more difficult than expected and leads to a structurally complicated solution. Next we will introduce a simple mathematical model using a probabilistic state space representation and show, how the reformulation of the example problem using this model leads to a much simpler and clearer implementation. Finally we will turn to the soundness of the basis by having a quick look at the mathematical status of some important simulation programs.

1 A pedagogical example

The model used in the following describes class sizes of a three-year third level school and the number of prospected school graduates. It is formulated in three steps of increasing detail, based on an example from [4] and extended here. The basic outputs are the class sizes $x_i(k)$, $i = 1 \dots 3$, at the beginning of year k and the number of graduates $x_g(k)$.

As a first step we assume given constant rates R_i of students, who have to repeat year i , and D_i of dropouts during year i . The model is then defined by

$$x_1(k+1) = x_{in}(k) + R_1 x_1(k) \quad (1a)$$

$$x_2(k+1) = (1 - R_1 - D_1) x_1(k) + R_2 x_2(k) \quad (1b)$$

$$x_3(k+1) = (1 - R_2 - D_2) x_2(k) + R_3 x_3(k) \quad (1c)$$

$$x_g(k+1) = (1 - R_3 - D_3) x_3(k) \quad (1d)$$

The resulting class sizes are non-integer, representing “mean values” over several years.

In the next step the mean values are replaced by integer random numbers following a binomial distribution $B(n,p)$. In the defining equations the number of repeaters and dropouts are replaced according to

$$R_i x_i(k) \rightarrow \xi_{R,i} \sim B(x_i(k), R_i)$$

$$D_i x_i(k) \rightarrow \xi_{D,i} \sim B(x_i(k), D_i)$$

leading to integer-valued class sizes.

Finally we add a rate M_i ($i = 2, 3$) of pupils who return to the previous class at midterm – a realistic possibility at German schools. The concrete numbers are again drawn from a Binomial distribution: $\xi_{M,i} \sim B(x_i(k), M_i)$. This seemingly simple extension has drastic consequences for the modeling, because it leads to changes of the class sizes at half-integer years.

2 Straightforward implementation

The implementation of the simplest model in Simulink can be done easily. One starts by introducing a generic component for a class with one input i for new pupils and three outputs: the class size c at the end of the term, the number t of pupils transferred to the next class and (for completeness) the number d of pupils that have dropped out during the current year. It is built with one `UnitDelay` block storing the class size $x(k)$ and the usual arithmetic blocks to implement the following relations derived from the model equations (1)

$$x(k+1) = i(k) + Rx(k) \quad (2a)$$

$$c(k) = i(k) + Rx(k) \quad (2b)$$

$$t(k) = (1 - R - D)x(k) \quad (2c)$$

$$d(k) = Dx(k) \quad (2d)$$

The complete model is then just a chain of three class components (cf. Fig. 1).

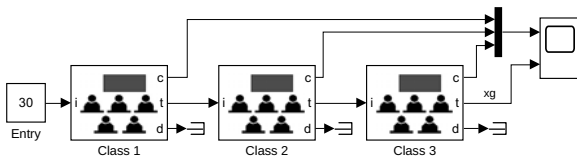


Figure 1: Simple school model

To replace the fixed rates with binomial random variables, one creates a component that outputs a random value $\xi \sim B(n, p)$, where p is given as parameter, while n comes from an input. This can be done with a simple Matlab function. The corresponding class is shown in Fig. 2.

The real challenge is of course the inclusion of the midterm downgrading. On the upper level this is simple enough: Just add another output m to the class component that gives the number of downgraders and route it

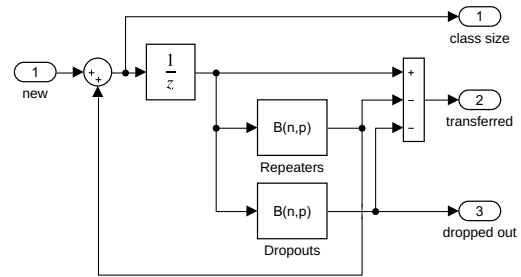


Figure 2: Class component with random values

back to the input of the lower class (cf. Fig. 3). Due to the different timing of the values, m cannot simply be added to the normal input. But a `TimeSwitch` that uses a standard `Switch` component to route its two inputs according to the time (integer or half-integer) solves this problem.

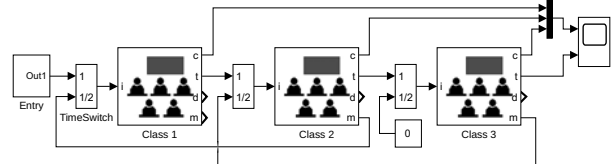


Figure 3: Complete school model

The implementation of the new class component starts by adding a binomial block for the downgraders, whose output is routed back internally to another `Time-Switch`. But the interesting question is of course, how one realises the different sample times: The class size changes every half year, the numbers of repeaters and dropouts at the beginning of a year and the number of downgraders at midterm.

Apparently Simulink offers an easy solution: Setting the `SampleTime` parameter of the `UnitDelay` to 0.5 and adding two `Rate Transition` blocks, which convert the signal to sample times $[1, 0]$ and $[1, 0.5]$, should do the trick. The corresponding class component is shown in Fig. 4, where the colors denote the different sample times.

But running the model we get the error message

Determinism of data transfer between 'school3a/Class 1/Unit Delay' and 'school3a/Class 1/Sum3' cannot be ensured because either or both blocks have non-zero sample time offset. You can resolve this by using a rate transition block whose parameter 'Ensure deterministic data transfer' is unchecked

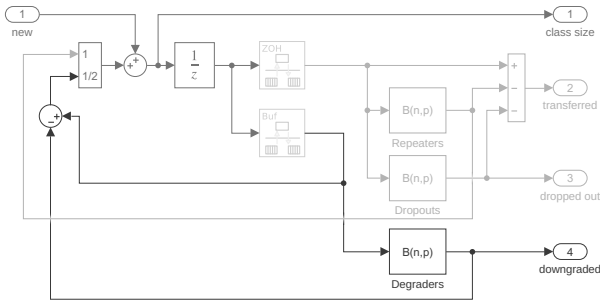


Figure 4: Complete class model

The suggested resolution actually leads to a working model: After unchecking the option `Ensure deterministic data transfer of the second Rate Transition`, the model runs without problems, and the results are as expected. But do we really know what is going on here?

Changing the sample rate is generally a non-trivial business, but a look at the relevant Simulink documentation [5] shows that there are more troubles looming around than one probably thought of, e. g. problems with timing when using multicore cpus. Without a precise understanding of the `Rate Transition` block, one cannot be sure that the `Class` component still works, when it becomes part of a very complex model – as real world components usually do.

The incremental modeling approach has lead into murky ground, since the simple structure of the model got lost on the way. Instead of relying on only half-understood remedies, we will therefore start afresh, this time with a solid foundation in the form of an underlying mathematical model.

3 A probabilistic state space model

We start with the well-known state space description

$$z(k+1) = G(z(k), v(k)) \quad (3a)$$

$$w(k) = H(z(k), v(k)) \quad (3b)$$

The integer k is the number of the time step, while z denotes the internal state, v the external input and w the output of the model, all possibly being vectors. This model can be easily implemented in Simulink by using the generic model shown in Fig. 5 and providing components for the functions $G(z, v)$ and $H(z, v)$.

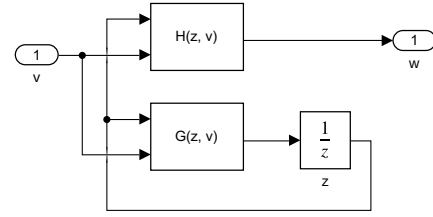


Figure 5: Generic state space model

For the stochastic examples we have to enlarge the state-space description. A common approach is the inclusion of an additional disturbance term $d(k)$ [6]:

$$z(k+1) = G(z(k), v(k), d(k))$$

$$w(k) = H(z(k), v(k), d(k))$$

In our case the disturbance is stochastic and its distribution depends on the state z . Therefore we define a random vector $\xi(z)$ and the *probabilistic state-space description*

$$z(k+1) = G(z(k), v(k), \xi(z(k))) \quad (4a)$$

$$w(k) = H(z(k), v(k), \xi(z(k))) \quad (4b)$$

Again a generic Simulink implementation can be easily given, cf. Fig. 6. The additional component `Random` computes the value of the random vector $\xi(z)$ using the current state vector $z(k)$.

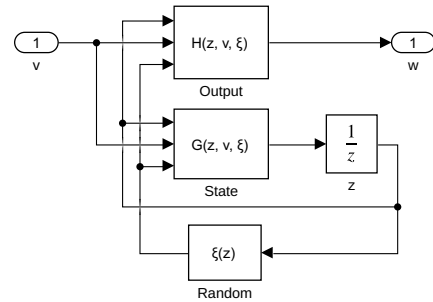


Figure 6: Generic probabilistic state space model

4 Theory-based implementation

We will now reimplement the example models using the mathematical descriptions and corresponding generic models defined above.

To reformulate the first model using eq. (3) we iden-

tify the state variable z with the class size x . Then we rewrite the defining equations (2) by setting $v \equiv i$, $w \equiv (c, t, d)'$ and get

$$\begin{aligned} G(z, v) &= v + Rz \\ H(z, v) &= \begin{pmatrix} v + Rz \\ (1 - R - D)z \\ Dz \end{pmatrix} \end{aligned}$$

The Simulink implementation of G and H is completely trivial, but so was the version we started with. Therefore we didn't gain much here expect for making contact to standard mathematical formulations.

For the implementation of the second model we now use the probabilistic state-space description eq. (4) and write

$$\begin{aligned} \xi(z) &\equiv \begin{pmatrix} \xi_R(z) \\ \xi_D(z) \end{pmatrix} \sim \begin{pmatrix} B(z, R) \\ B(z, D) \end{pmatrix} \\ G(z, v, \xi) &= v + \xi_R \\ H(z, v, \xi) &= \begin{pmatrix} v + \xi_R \\ z - \xi_R - \xi_D \\ \xi_D \end{pmatrix} \end{aligned}$$

The component `Random` simply combines the outputs of two `Binomial` blocks into a vector.

The final example model is time-dependent, since its behaviour changes between full and half years. The standard trick here is to include the time as a component of the state vector. In our case we only need to know whether we are at full or half term. Therefore we enlarge the state vector writing

$$z = \begin{pmatrix} x \\ s \end{pmatrix}, \quad z(0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

where x is again the class size and the variable s is 0 or 1 at full resp. half term. Its state equation then simply is

$$s(k+1) = 1 - s(k).$$

Now we have to define the functions $\xi(z)$, $G(z, v, \xi)$ and $H(z, v, \xi)$, especially their behaviour at $s = 0$ and $s = 1$. First we define the random vector $\xi = (\xi_R, \xi_D, \xi_M)'$. The first two components describe the numbers of repeaters and dropouts, so they should be 0 at midterm. Correspondingly the number ξ_M of midterm downgraders should be 0 at the beginning of a term. Thus

we have

$$\begin{aligned} \xi_R(z) &\sim \begin{cases} B(x, R) & | \quad s = 0 \\ 0 & | \quad s = 1 \end{cases} \\ \xi_D(z) &\sim \begin{cases} B(x, D) & | \quad s = 0 \\ 0 & | \quad s = 1 \end{cases} \\ \xi_M(z) &\sim \begin{cases} 0 & | \quad s = 0 \\ B(x, M) & | \quad s = 1 \end{cases} \end{aligned}$$

Similarly we arrive at the state equation

$$\begin{aligned} G_1(z, v, \xi) &= \begin{cases} v + \xi_R & | \quad s = 0 \\ v + x - \xi_M & | \quad s = 1 \end{cases} \\ G_2(z, v, \xi) &= 1 - s \end{aligned}$$

Finally we enlarge the output vector to include the downgraders writing $w \equiv (c, t, d, m)'$ and get the output function

$$\begin{aligned} H_1(z, v, \xi) &= \begin{cases} v + \xi_R & | \quad s = 0 \\ v + x - \xi_M & | \quad s = 1 \end{cases} \\ H_2(z, v, \xi) &= \begin{cases} x - \xi_R - \xi_D & | \quad s = 0 \\ 0 & | \quad s = 1 \end{cases} \\ H_3(z, v, \xi) &= \xi_D \\ H_4(z, v, \xi) &= \xi_M \end{aligned}$$

This formal approach guarantees a completely precise problem specification, though it may seem a bit tedious. On the upside the Simulink implementation is now merely a matter of routine: We use the generic model from Fig. 6 adding a `Demux` block to get the single output ports. Next we implement the equations for `Random` (cf. Fig. 7), `State` (cf. Fig. 8) and `Output` (along the same lines).

The auxiliary component `PhaseSwitch` (cf. Fig. 9) helps to implement the case switches. Instead of a clock (as in the first implementation) it now simply uses the value of the state variable s .

The complete `school` model looks almost like before (cf. Fig. 3), but the `TimeSwitch` components at the class inputs are replaced by simple summation blocks. This is possible now, since all signals have the same sample time 0.5 and have meaningful values at all times.

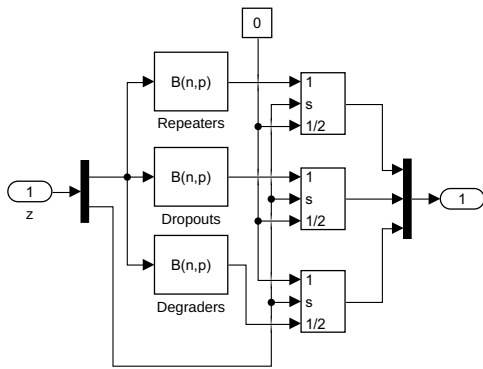


Figure 7: Implementation of the `Random` function

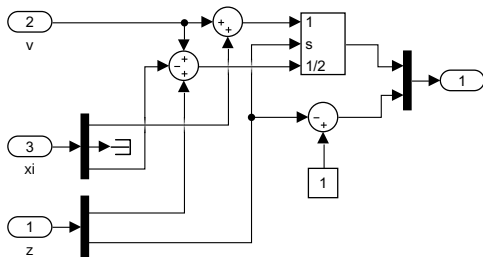


Figure 8: Implementation of the `State` function

5 Mathematical foundation of simulation tools

Even if a model is based on a sound mathematical foundation, its simulation results may be not, namely when it is implemented using a simulation environment that is itself not precisely defined. The Simulink blocks that have been used in the above model, are very simple and can be easily described mathematically. But already the example of the `Rate Transition` block has shown that this may not always be the case. Therefore lets finally have a quick look at the mathematical status of some important tools.

While the model descriptions of continuous systems often boil down to differential or differential-algebraic equations, there is no generally accepted mathematical formulation of hybrid systems containing a few discrete events or even complex state charts [7]. So even the framework, in which to formulate a mathematical model of a simulation tool, is still under construction.

For Stateflow [8], a Simulink add-on to model hierarchical state machines and flowchart diagrams, the situation has been described as follows:

However, Stateflow lacks any formal defini-

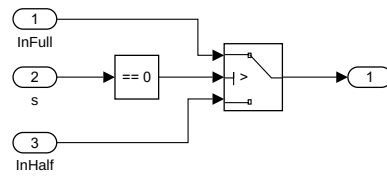


Figure 9: Implementation of `PhaseSwitch`

tion. The semantics of a program is given by the result of its simulation within the Mathworks tools. This absence of formal definition is a big obstacle to static analysis, verification, or automatic test-cases generation of Stateflow designs.[9]

Therefore the authors proceed to define operational [10] and denotational [9] semantics for Stateflow in a formal way. One immediate result is the insight that Stateflow is structurally different from other, seemingly very similar formulations of statechart diagrams. A different way to formalize a hybrid Simulink/Stateflow model has been proposed in [11] with applications to the formal verification of such complex systems as the guidance control of a lunar lander or the control system of a high-speed train.

Another simulation program in the Matlab/Simulink tool chain is SimEvents [12] that implements the transaction-based modeling paradigm for discrete event systems. Earlier releases had some serious deficiencies [13], therefore Mathworks came up with a complete redesign with version 5. A very interesting feature is that the design is based on a unifying theoretical description [14]. Unfortunately, Mathwork has chosen a new design instead of relying on the well-known PDEVS formalism [2] and doesn't provide these internal specifications in general.

The discrete-events simulation program Arena [15] from Rockwell Automation implements the process-based modeling approach. It is based internally on the simulation language SIMAN [16] and outputs a SIMAN program corresponding to the graphically constructed model. This can be useful for verification purposes, since the specification of the lower-level SIMAN constructs is much easier to understand than the complex blocks used on the upper level in Arena. And one can go down further: In [17] an essential part of the SIMAN language has been implemented using the PDEVS formalism, thereby providing a sound mathematical formulation of important parts of Arena.

6 Conclusions

Using a clear mathematical description of our example problem we arrived at a precise specification that could be implemented in Simulink in a completely routine way and is structurally simpler than the previous “straightforward implementation”. But this idea only works, if the underlying simulation tools are grounded on sound mathematical models themselves. Though some efforts have been made in this direction, a lot needs to be done to reach a satisfying, well defined environment for our models.

The practitioner often works in the context of a given very large model that changes in the course of further development. He can’t be an expert of every intricacy of the complex simulation tool he works with, and very often has not enough time to go to the bottom of every problem. This can lead to ad-hoc implementations that are potentially dangerous in the highly dynamic larger context. The only way out is to use a precise mathematical model - for the own problem as well as for the relevant features of the simulation tool.

References

- [1] Lunze J. *Ereignisdiskrete Systeme*. Berlin: de Gruyter, 3rd ed. 2017.
- [2] Zeigler BP, Praehofer H, Kim TG. *Theory of Modeling and Simulation*. San Diego: Academic Press, 2nd ed. 2000.
- [3] Cassandras CG, Lafortune S. *Introduction to Discrete Event Systems*. New York: Springer, 2nd ed. 2008.
- [4] Junglas P. *Praxis der Simulationstechnik*. Haan-Gruiten: Verlag Europa-Lehrmittel. 2014.
- [5] The Mathworks. *Handle Rate Transitions (Simulink Coder)*. <https://de.mathworks.com/help/releases/R2018a/rtw/ug/handle-rate-transitions.html>.
- [6] Ljung L, Glad T. *Modeling and Identification of Dynamic Systems*. Lund: Studentlitteratur AB. 2016.
- [7] Breitenacker F, Zauner G, Popper N, Judex F, Troch I. Structure of Simulators for Hybrid Systems - Development and New Concept of External and Internal State Events. *SNE Simulation News Europe*. 2007; 17(2):39–48.
- [8] The MathWorks. *Stateflow: Model and simulate decision logic using state machines and flow charts*. <http://www.mathworks.com/products/stateflow/>.
- [9] Hamon G. A denotational semantics for Stateflow. In: *Proceedings of the 5th ACM international conference on Embedded software*. ACM. 2005; pp. 164–172.
- [10] Hamon G, Rushby J. An operational semantics for Stateflow. *International Journal on Software Tools for Technology Transfer*. 2007;9(5-6):447–456.
- [11] Zhan N, Wang S, Zhao H. *Formal Verification of Simulink/Stateflow Diagrams: A Deductive Approach*. New York: Springer. 2017.
- [12] Clune MI, Mosterman PJ, Cassandras CG. Discrete Event and Hybrid System Simulation with SimEvents. In: *8th International Workshop on Discrete Event Systems*. Ann Arbor. 2006; p. 386–387.
- [13] Austermann L, Junglas P, Schmidt J, Tiekmann C. Conceptual problems of transaction-based modeling and its implementation in SimEvents 4.4. *SNE Simulation News Europe*. 2017;27(3):137–142.
- [14] Li W, Mani R, Mosterman PJ. Extensible discrete-event simulation framework in SimEvents. In: *Proc. 2016 Winter Simulation Conference*. Arlington: IEEE. 2016; pp. 943–954.
- [15] W David Kelton NBZ Randall P Sadowski. *Simulation with Arena*. New York: McGraw-Hill, 6th ed. 2015.
- [16] Pegden CD, Shannon RE, Sadowski RP. *Introduction to Simulation using SIMAN*. New York: McGraw-Hill, 2nd ed. 1995.
- [17] Sanz V. Hybrid System Modeling using the Parallel DEVS Formalism and the Modelica Language. Ph.D. thesis, UNED Madrid, E.T.S.I. Informatica. 2010.

RPDEVS Abstract Simulator

Franz J. Preyser^{1*}, Bernhard Heinzl¹, Wolfgang Kastner¹

¹ Institute of Computer Engineering, Automation Systems Group, TU Wien, Treitlstraße 1-3, 1040 Vienna, Austria; *franz.preyser@tuwien.ac.at

Abstract. The Revised Parallel DEVS (RPDEVS) modeling formalism enhances the Parallel Discrete Event System Specification (PDEVS) by the ability to model 'real' Mealy behavior of components. The term 'real' Mealy behavior can be summarized as immediate output response to an input event without a state transition in between. Although this enhancement simplifies model creation, especially of reusable components, it requires a more complex simulation algorithm. In this paper, we present an RPDEVS abstract simulator that describes the simulation execution of RPDEVS models.

Introduction

The Discrete Event System Specification (DEVS) [1] is a modular and hierarchical modeling formalism for systems that process input events, have an internal state, and may produce output events. Basic components can be specified as atomic DEVS which can be coupled with one other in a block diagram manner. The formal definition of an atomic DEVS is similar to a finite automaton (or sequential machine). In [2], the author describes an atomic DEVS as *DEVS Moore Automaton* embedded in additional logic that provides the necessary time events. Automata theory distinguishes between Moore and Mealy automata. The output events of Moore automata solely depend on the system's current state, whereas the output of Mealy automata may also depend on the current input. In theory, these two types of automata are equivalent in the sense that every automaton of the one type can be replaced by a corresponding automaton of the other type. However, in average the Moore model needs about twice the number of states and transitions than the corresponding Mealy model to represent the same system [3].

Both, in classic DEVS and in its most popular revision Parallel Discrete Event System Specification (PDEVS) [4], the output function λ solely depends on the internal state of the system. Thus, these two formalisms only allow the modeling of Moore behavior. If

Mealy behavior is needed, it has to be modeled with a workaround, using a *transitory state* (a state with zero lifetime). However, as discussed in [5], the use of transitory states leads to a delay of events regarding processing order, which in turn impedes reusability of components. Due to the reasons mentioned above and the experiences we made with applying both, DEVS [6] and PDEVS [7], we decided to revise PDEVS resulting in Revised Parallel DEVS (RPDEVS) published in [8]. Basically, the changes include the support of 'true' mealy behavior and the merging of the three state transition functions δ_{int} , δ_{ext} , and δ_{conf} into one generic state transition function δ . As mentioned above, a Mealy automaton needs about half the states compared to the corresponding Moore automaton. Evaluation of RPDEVS shows that formalization of Mealy models simplifies to a similar extent compared to PDEVS. Also merging the state transition functions condenses model definition, since the different transition functions often match at least in parts. However, the price for simplifying modeling is an increase in the complexity of the simulation algorithm.

In this work, we first recap the the RPDEVS formalism, before its simulation algorithm is described and presented as *abstract simulator*.

1 RPDEVS Formalism

Equally to classic DEVS and PDEVS, RPDEVS distinguishes between *atomic* and *coupled* components which can be used for modular and hierarchical structuring of complex models (see Figure 1). As shown in [8], RPDEVS also provides *closure under coupling*, which means that for every coupled component an equivalent atomic component can be designed. This assures that couplings can be used within other couplings as if they were atomics.

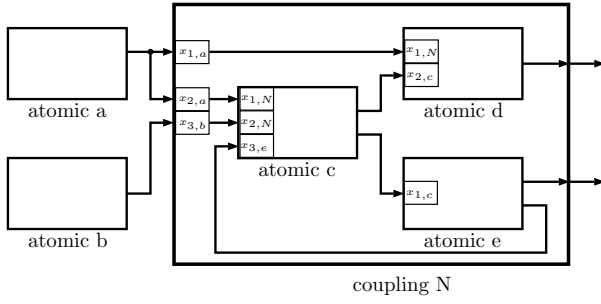


Figure 1: Modular and hierarchical decomposition of a complex model into atomic and coupled RPDEVS components.

1.1 Atomic RPDEVS

Formally, an atomic RPDEVS M is defined as

$$M = \langle X^b, S, Y^b, \delta, \lambda, ta \rangle,$$

where the single entities have the following meanings:

- X^b ... set of possible input bags
- S ... set of possible states (=state space)
- Y^b ... set of possible output bags
- $\delta : Q \times X^b \rightarrow S$... state transition function
- $\lambda : Q \times X^b \rightarrow Y^b$... output function
- $ta : S \rightarrow [0, \infty]$... time advance function
- $Q = \{(s, e) | s \in S, e \in [0, ta(s)]\}$
- e ... elapsed time since last event

Theoretically, X^b is a set of multisets with no particular structure. However, for practical implementation where it is feasible to define input ports which can be connected individually to output ports of other components, the set of possible input bags may be structured into sub-bags, one for each input port. Additionally, the sub-bags can be structured according to the source components the corresponding input messages origin from (see Figure 1). This is especially done in the RPDEVS simulation algorithm presented in Section 2, which has to remember the source component of every input message.

The differences of an atomic RPDEVS compared to PDEVS are the input dependency of the output function λ and the single state transition function δ which replaces the three separated transition functions δ_{int} , δ_{ext} , and δ_{conf} (for details about PDEVS, see [4, 1]). Furthermore, in RPDEVS, λ is called on any kind of event, external, internal, and confluent. The explicit distinc-

tion between these three event types is dropped and the behavior of an RPDEVS atomic is the same for each of them:

1. Call the output function λ .
2. Recalculate λ as long as the input bag changes due to (re)calculations of lambda at influencing components (*lambda-iteration*).
3. Conduct state transition δ once (*delta-step*).
4. Call the time advance function ta which returns the time to the next internal event.

If different treatment is necessary depending on whether the event was triggered by the arrival of an input (external event), by the expiration of the current state's lifetime (internal event), or by both happening concurrently (confluent event), this has to be incorporated into the definitions of δ and λ . External events can be recognized by a non-empty input bag ($x^b \neq \emptyset$), whereas internal events imply $e = ta(s)$.

The single transition function δ avoids having to define identical behavior multiple times in cases in which the three transition functions partly match.

According to [9], it frequently happens that calculations necessary for the output event in λ are also necessary for the computation of the next state and thus, have to be repeated in δ_{int} . In the classic DEVS simulator *DesignDEVS* [10], they even merge the two functions λ and δ_{int} to prevent unnecessary recalculations. This is not possible for RPDEVS as λ may have to be called multiple times before the state transition can be conducted. Therefore, for practical implementation, we recommend to split the internal state s of an atomic into two parts $s = (s_\delta, s_\lambda) \in S = S_\delta \times S_\lambda$ which allows to redefine λ and δ as follows:

$$\begin{aligned} \lambda &: S_\delta \times [0, \infty) \times X^b \rightarrow Y^b \times S_\lambda, (s_\delta, e, x^b) \mapsto (y^b, s_\lambda) \\ \delta &: S_\delta \times S_\lambda \times [0, \infty) \times X^b \rightarrow S_\delta, (s_\delta, s_\lambda, e, x^b) \mapsto s_\delta \end{aligned}$$

Thus, when λ already needs to calculate a new state value for generating the output y , it can be buffered into s_λ to be reused in δ .

1.2 Coupled RPDEVS

The formal definition of a coupled RPDEVS is identical to that of a coupled PDEVS (see [4]):

$$N = \langle X^b, Y^b, D, \{M_d\}_{d \in D}, \{I_d\}_{d \in D_N}, \{Z_{i,d}\}_{i,d \in D_N} \rangle$$

with $D_N = D \cup \{N\}$ and

X^b ... set of possible input bags
 Y^b ... set of possible output bags
 D ... index set
 M_d ... child component of N for each $d \in D$
 $I_d \subset D \cup \{N\}$... influencer set of d
 $Z_{i,d}$... output translation function

The output translation function $Z_{i,d}$ translates the output events of component i into input events for component d . Theoretically, $Z_{i,d}$ could also alter output events. However, in practice it just forwards events. If the destination component is a coupling, the output translation functions of that coupling further forward the events to the destinations within the coupling. This is repeated until finally the events reach atomics.

As already mentioned in Section 1.1, the multiset of possible input bags can be structured by input port and source component. In the following, we will not consider ports, but separate the input bags according to the influencers the messages originate from. Such a structuring for a component d has the form

$$X_d^b = \prod_{i \in I_d} X_{i,d}^b,$$

with $X_{i,d}^b$ being the multiset of possible input messages from component i ($Z_{i,d}: Y_i^b \rightarrow X_{i,d}^b$). Consequently, every input bag x_d^b of a component d has the form

$$x_d^b = (x_{i_1,d}^b, x_{i_2,d}^b, \dots, x_{i_l,d}^b), \quad I_d = \{i_1, i_2, \dots, i_l\}.$$

Thereby, $x_{i_k,d}^b$ is the translated result of the output function of influencer i_k : $x_{i_k,d}^b = Z_{i_k,d}(y_{i_k}^b)$, $\forall k = 1, 2, \dots, l$.

2 RPDEVS Abstract Simulator

To complete the introduction of RPDEVS started in [8], the definition of an abstract simulator is given. Like in classic DEVS and parallel DEVS, the code consists of a *simulator* part responsible for executing an atomic, a *coordinator* part responsible for executing a coupling, and a *root-coordinator* responsible for the overall model execution. Furthermore, we stick to the format known from [1], using message passing. There are five types of messages used:

i-message The initialization message is sent to every component at simulation start. It is used to ini-

tialize state variables and gather the times of the first internal events at the single components.

***-message** In PDEVS, this is the *internal state transition message* because there the output function λ is inseparably connected to the internal and confluent state transitions δ_{int} and δ_{conf} . However, in RPDEVS, λ is calculated in an iterative manner and on every kind of event. Thus, this message is solely used to trigger the λ iteration.

y-message The y-message is used to transport the output message calculated in λ to the parent coordinator where it is forwarded to the input bag of the receiving component.

x-message In RPDEVS, the x-message is used to trigger the state transition. Whenever a component receives an x-message, it executes δ and then calculates the time of its next internal event t_n .

done-message This message is used for synchronization. When the coordinator triggers child components to do their initialization or to conduct their state transition, it has to wait until all of them are done before simulation can proceed.

2.1 Simulator

The simulator of an atomic RPDEVS is nearly identical to the one of an atomic PDEVS (see [1], p. 285):

```
RPDEVS-simulator
variables:
  parent      // parent coordinator
  t1          // time of last event
  tn          // time of next event
  RPDEVS     // assoc. model with total
              // state (s,e), time advance
              // function, lambda and delta
  (s_i,e_i)  // initial total state
  y          // output message bag

when receive i-message(i,t)
  (s,e) = (s_i, e_i)
  t1 = t - e
  tn = t1 + ta(s)
  send done-message(done, tn) to parent

when receive *-message(*,x,t)
  e = t - t1
  y = lambda(s,e,x)
  send y-message(y,t) to parent

when receive x-message(x,t)
```

```

e = t - t1
s = delta(s,e,x)
t1 = t
tn = t1 + ta(s)
send done-message(done, tn) to parent
end RPDEVS-simulator

```

The most important differences compared to the PDEVS simulator are the additional parameter x of the $*$ -message, and the absence of the case distinction between *internal*, *external*, and *confluent* event when receiving an x -message. Like in [11], a done-message is used for synchronization during the potentially parallel execution to prevent the problems with Zeigler's PDEVS algorithm described in [12].

2.2 Coordinator

The more interesting part of the abstract simulator is the *coordinator*. We start with the definition of all necessary variables followed by the i -message and done-message procedures:

```

RPDEVS-coordinator
variables:
parent      // parent coordinator
t1          // time of last event
tn          // time of next event
RPDEVS     // associated coupled model
            // including index set D,
            // influencer sets I_d, and
            // output transl. fcts. Z_id
event-list // list of elements (d,tn_d),
            // sorted ascending by tn_d
IMM         // imminent children
y_coupling // output message of coupling
x_dr       // sub input bags:
            // d... sender, r... receiver
x_r        // input bag of component r
y_dN       // sub output bag of coupling
            // N, d... sender
INF        // set of influenced children
            // (with changed input bag)
INF'       // INF for next lambda-iter.
DELTA      // set of children who need to
            // conduct a state transition
CHECK      // components with withdrawn
            // input messages

when receive i-message(i,t)
DELTA = D
for-each d in D do
send i-message(i,t) to child d
wait until DELTA = {}
sort event-list according to tn_d
t1 = max{t1_d : d in D}

```

```

tn = min{tn_d : d in D}
send done-message(done, tn) to parent

```

```

when receive done-message(done, td) from d
event-list.(d,tn_d) = (d,td);
remove d from DELTA

```

At simulation start, the coordinator receives an i -message from its parent coordinator. The parent of the uppermost coordinator is the *root-coordinator* (see Section 2.3). The i -message is forwarded to all child components $d \in D$ which causes them to calculate their time of next internal event tn_d . Then, the coordinator waits until all children have sent their done-message (i.e. $DELTA = \{\}$) before the time of the next internal event tn of the coupling can be determined.

If a component is imminent (i.e. its time of next event $tn=t$), it receives a $*$ -message from its parent coordinator. This message initiates the λ iteration in the coupling. The goal of the λ iteration of a coupling is generating its output message $y_coupling$.

```

when receive *-message(*,x,t)
y_coupling = {}
for-each (d,tn_d) in event-list with tn_d=t
add d to IMM, DELTA and INF
remove (d,tn_d) from event-list
for-each r in D with N in I_r
if x_Nr != Z_Nr(x)
x_Nr = Z_Nr(x)
add r to INF and DELTA
if x_Nr={}
add r to CHECK
for-each r in INF
x_r = {x_dr : d in I_r, x_dr != {}}
while CHECK != {}
pick and remove r from CHECK
if x_r={}
if r not in IMM
remove r from INF and DELTA
for-each d in D with r in I_d
x_rd = {}
add d to CHECK
if r in I_N
y_rN = {}
INF'={}
for-each r in INF
send *-message(*,x_r,t)

```

In the $*$ -message of the coordinator first, the imminent children are determined and collected in IMM, INF, and DELTA. Then, the components' input bag has changes caused by the couplings input are calculated. All components whose input bag changed are added to INF and scheduled for state transition by adding them

to DELTA. Finally, $*$ -messages are sent to the affected child components triggering their λ execution.

These λ executions result in output messages transported via y -messages back to the coordinator. In the coordinator's y -message procedure, all output messages of all triggered child components are gathered and converted using the output translation functions Z_{dr} . Depending on the coupling relations, they are converted either into input messages for other children or into coupling output messages. Thereby, all child components whose input bag has changed are collected in INF' . After the last element in INF has responded to the coordinator with a y -message, the components in INF' are shifted into INF . If INF is not empty after that, again a $*$ -message is sent to every component in INF and their response, in form of y -messages is awaited. However, if INF is empty at the end of the y -message procedure, it means no input bag has changed during the last λ iteration, i.e. they are stable. Thus, the λ iteration of the coupling has terminated and the coordinator can send a y -messages to its parent. In [8], it is shown that for models without algebraic loops, the λ iteration always terminates after a maximum of $n = |D|$ iterations. In some cases, algebraic loops can even be solved by the simulation algorithm (see RS flip-flop in [13]).

During the course of λ iterations, it may happen that input messages for child components that were produced in previous iterations may have to be withdrawn from the respective input bag. Thereby, it may occur that the input bag becomes completely empty although it was not in the preceding iteration. These components then need to be checked separately because they may already have produced output in reaction to a non-empty input bag (Mealy behavior) and thereby may have influenced other components. This task is handled via the set CHECK.

A coordinator may represent a coupling that is used as component in a parent coupling. In this parent coupling, there is also a λ iteration in progress. Thus, the parent coordinator may send multiple $*$ -messages to its child coordinators. This is why the $*$ -message procedure of the coordinator also has to check whether formerly received coupling inputs still exist in the new iteration (using CHECK).

```
when receiving y-message(y_d,t) from d
  remove d from INF
  if d in I_N
    y_dN = Z_dN(y_d)
  for-each r in D with d in I_r
```

```
if x_dr != Z_dr(y_d)
  x_dr = Z_dr(y_d)
  if x_dr={}
    add r to CHECK
  add r to INF' and DELTA
if INF = {}
  INF = INF'
  INF' = {}
for-each r in INF
  x_r = {x_dr : d in I_r, x_dr != {}}
while CHECK != {}
  pick and remove r from CHECK
  if x_r={}
    if r not in IMM
      remove r from INF and DELTA
      for-each d in D with r in I_d
        if x_rd != {}
          x_rd = {}
          add d to INF, DELTA and CHECK
      if r in I_N
        y_rN = {}
for-each r in INF
  send *-message(*,x_r,t) to component r
if INF = {}
  y_coupling={y_dN : d in I_N, y_dN!={}}
  send y-message(y_coupling,t) to parent
```

Receiving an x -message means that the λ iteration is finished and the state transitions can be conducted. This is done by sending an x -message to every imminent child component and to every child component with non-empty input bag. These components have been gathered in DELTA during the λ iteration. After the x -messages are sent, the coordinator waits until all of them are done with their state transition. Afterwards, the time of the next internal event can be calculated and the set IMM is cleared.

```
when receive x-message(x,t)
  for-each r in DELTA
    send x-message(x_r,t) to r
  wait until DELTA = {}
  sort event-list according to tn_d
  t1 = t
  tn = min{tn_d: d in D}
  IMM = {}
  send done-message(done, tn) to parent
end RPDEVS-coordinator
```

2.3 Root Coordinator

Finally, on top of the uppermost coordinator is the root coordinator. It starts the simulation by sending an i -message to its child coordinator. Then it advances the simulation time to the time of next event, initiates the λ iteration by sending a $*$ -message, waits until the λ iteration is finished and then triggers the state

transition by sending an x-message. This is repeated until the simulation time t exceeds the final time t_{end} .

```

RPDEVS-root-coordinator
variables:
  tstart // simulation start time
  tend   // simulation end time
  t      // current simulation time
  child  // direct subordinate coordinator

t = tstart
send i-message(i,t) to child
wait for done-message(done,tn) from child
t=tn
while t < tend
  send *-message(*,t) to child
  wait for y-message(y,t) from child
  send x-message({},t) to child
  wait for done-message(done,tn) from child
  t=tn
end RPDEVS-root-coordinator

```

3 Conclusion

In this work, we first recapped RPDEVS and pointed out the parallels of PDEVS and RPDEVS to Moore and Mealy automata. Furthermore, we demonstrated how the input bags can be formally split up into sub-bags, one for each influencer. This separation is used by the abstract simulator as it makes it easier to detect input bag changes due to λ recalculations in the influencers. The structure of the abstract simulator is basically similar to the one of Zeigler's PDEVS abstract simulator [1]. For synchronization purposes though, we also added the done-message of Chow's algorithm [11].

When implementing the algorithm, especially when facilitating parallelism, aspects like consistent global variable manipulation and execution order have to be taken into account. However, this degree of detail would go beyond the scope of this paper.

Nevertheless, there already exists a proof-of-concept implementation of an RPDEVS simulator. We reprogrammed the simulation engine of the open-source classic DEVS simulator *PowerDEVS* and named it *PowerRPDEVS*. It is available on *SourceForge* [14]. In *PowerRPDEVS*, a sequential version of the algorithm is implemented. Exploitation of parallelism in the *PowerRPDEVS* engine is an issue for future work.

Acknowledgement

We thank all partners of the project ASPeCT for their contributions. The presented work is funded by the Austrian Research Promotion Agency within the program *Produktion der Zukunft*, project number 858655.

References

- [1] Zeigler BP, Praehofer H, Kim TG. *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press. 2000.
- [2] Joslyn C. The process theoretical approach to qualitative DEVS. *Proc 7th Conf on AI, Simulation, and Planning in High Autonomy Systems*. 1996;.
- [3] Klimovich AS, Solov'ev VV. Transformation of a Mealy Finite-State Machine into a Moore Finite-State Machine by Splitting Internal States. *J Comput Syst Sci Int*. 2010;49(6):900–908.
- [4] Chow ACH, Zeigler BP. Parallel DEVS: a parallel, hierarchical, modular, modeling formalism. In: *Proc. of WSC'94*. 1994; pp. 716–722.
- [5] Preyser FJ, Heinzl B, Raich P, Kastner W. Towards Extending the Parallel-DEVS Formalism to Improve Component Modularity. In: *Beiträge zum WS der ASIM/GI-Fachgruppen STS und GMMS 2016*; pp. 83–89.
- [6] Preyser FJ. An Approach to Develop a User Friendly Way of Implementing DEV&DESS Models in PowerDEVS. Masterthesis, TU Wien. 2015.
- [7] Raich P, Heinzl B, Preyser F, Kastner W. Modeling Techniques for Integrated Simulation of Industrial Systems Based on Hybrid PDEVS. In: *Proc. of MSCPES'16*, 1. 2016; pp. 1–6.
- [8] Preyser F, Heinzl B, Kastner W. RPDEVS: Revising the Parallel Discrete Event System Specification. In: *Proc. of MATHMOD 2018*; pp. 269–274.
- [9] Goldstein R, Breslav S, Khan A. Informal DEVS conventions motivated by practical considerations. In: *Proc. of DEVS 2013*; pp. 10:1–10:6.
- [10] Goldstein R, Breslav S, Khan A. Practical aspects of the DesignDEVS simulation environment. *Simulation*. 2017;94(4):301–326.
- [11] Chow AC, Zeigler BP, Kim DH. Abstract Simulator for the Parallel DEVS Formalism. In: *Proc. of the Fifth Annual Conference on AI, and Planning in High Autonomy Systems*. 1994; pp. 157–163.
- [12] Schwatinski T, Pawletta T. An advanced simulation approach for parallel DEVS with ports. In: *Proc. of SpringSim '10*. 2010; pp. 147–154.
- [13] Fiedler C, Preyser F, Kastner W. Simulation of RPDEVS Models of Logic Gates. In: *Proc. of ASIM-Workshop Simulation technischer Systeme/Grundlagen und Methoden in Modellbildung und Simulation*. 2019; p. 6.
- [14] PowerRPDEVS on sourceforge: <https://sourceforge.net/projects/powerrpdevs/>. [accessed: 2019-01-04].

Simulation von Partikelflugbahnen zur Auslegungshilfe von elektrostatischen Luftfiltern

Sebastian Beckers^{1*}, Julian Pawlik¹, Jürgen Kiel¹, Wolfgang Grote¹

¹FMDauto- Institut für Produktentwicklung und Innovation, Hochschule Düsseldorf, Münsterstr. 156, 40476 Düsseldorf, Deutschland; *sebastian.beckers@hs-duesseldorf.de

Kurzfassung. In dieser Arbeit wird ein Modell zur Simulation von Partikelflugbahnen als Auslegungshilfe von elektrostatischen Luftfiltern vorgestellt. Die Umsetzung erfordert nicht nur die Nachbildung des elektrostatischen Feldes, sondern auch die Simulation des Strömungs- und Ionenfeldes. Auf Basis dieser Feldverteilungen lassen sich die Raumkurven der Partikelbewegungen (Trajektorien) berechnen. Diese umfassen den Flugverlauf vom Eintritt der Partikel in den Elektroabscheider über die Partikelaufladung im Ionisator bis hin zur Partikelabscheidung auf den Niederschlagselektroden des Kollektors. Zur Vereinfachung des Modells werden ein zweidimensionaler Filterkanal und eine laminare Strömung angenommen. Die Rückwirkung der Partikelbewegung auf die Strömung, die Rückwirkung der Strömung auf das Ionenfeld und die Rückwirkung der Partikelladung auf das elektrostatische und Ionenfeld werden vernachlässigt wie auch der Strömungseinfluss des elektrischen Windes. Die Validierung des Modells erfolgt durch einen Vergleich der numerisch und experimentell ermittelten Partikelabscheidegrade und der Partikelabscheidungsverteilungen auf den Niederschlagselektroden.

Einleitung

Bei der Luftfilterung von Innen- und Wohnraumanwendungen kommen aufgrund ihrer hohen Energieeffizienz häufig zweistufige Elektroabscheider (EA) zum Einsatz. Eintretende Partikel werden dabei in einer Ionisator-Vorstufe zunächst durch ein Ionenfeld aufgeladen und danach in einer Kollektor-Stufe, durch die Kraftwirkung eines elektrostatischen Feldes, auf den Niederschlagselektroden (NE) abgeschieden.

Die Auslegung eines EAs basiert im Allgemeinen auf Annahmen bezüglich der elektrostatischen Feldverteilung, der Ionenverteilung sowie des Strömungsfeldes. Beispielsweise muss bei der Bestimmung der Partikelaufladung die Ionendichte sowie die mittlere elektrostatische Feldstärke abgeschätzt werden.

Somit ist zwar eine konservative Grundausslegung möglich, jedoch können die daraus resultierenden Voraussagen der Filterleistung zu ungenau sein, sodass der Entwicklungsprozess durch erforderliche Nachbearbeitungen signifikant verlängert wird (trial and error).

Dagegen erlaubt eine Nachbildung der Partikelflugbahnen des Filterprozesses eine präzisere Vorauslegung der Filterparameter und stellt demnach eine Verkürzung der Entwicklungsphase in Aussicht.

1 Simulationsmodell

1.1 Simulationsansatz

Das in dieser Arbeit vorgestellte Simulationsmodell basiert auf dem Euler-Lagrange-Ansatz. In der eulerschen Betrachtungsweise werden die für den Filterprozess relevanten Felder über ein ortsfestes Lösungsgebiet Ω abgebildet. Dazu gehören das elektrostatische Feld (E-Feld), das Ionenfeld und das Strömungsfeld. Diese bilden zusammen die kontinuierliche Phase der Simulation ab, vgl. Abb. 1.

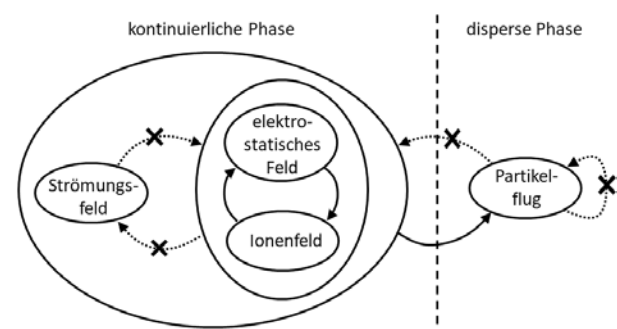


Abbildung 1: Schaubild des Simulationsansatzes

Basierend auf den oben beschriebenen Feldverteilungen lassen sich die Bahnkurven (Trajektorien) der einzelnen Partikel in der lagrangeschen Betrachtungsweise nachbilden. Dieser Programmteil stellt die disperse Phase der Zweiphasenströmungssimulation dar.

Die Wechselwirkungen zwischen dem E-Feld und dem Ionenfeld werden im Modell als eine Zwei-Wege-Kopplung berücksichtigt, da die Ionen bzw. Raumladungsverteilung im Wesentlichen durch die Wirkungsrichtung des E-Feldes sowie die Eigenladungen der Ionen bestimmt wird.

Nicht berücksichtigt werden dagegen die Wechselwirkungen zwischen dem Strömungsfeld und den elektrischen Feldern. Dies liegt zum einen daran, dass die Ionenstromstärke in der Wohnraumanwendung aufgrund der Ozonproblematik in der Regel sehr gering und infolgedessen der Impulsaustausch zwischen den driftenden Ionen und den Gasmolekülen in Form des sog. elektrischen Windes unbedeutend ist. Zum anderen liegt die durchschnittliche Ionengeschwindigkeit bei über 10 ms^{-1} [1], sodass der Strömungseinfluss, wie z. B. in Form einer Querverschiebung, durch das vergleichsweise langsame Strömungsfeld in dieser Anwendung (0.1 bis 1 ms^{-1}) zu vernachlässigen ist. Die Rückwirkungen des Partikelstroms hinsichtlich der Partikeleigenladung auf das E-Feld wie auch der Impulsaustausch mit der Strömung werden in dieser Arbeit aufgrund der Einzelflugsimulation und der geringen Stokes-Zahl $St \ll 1$ [2] ebenfalls nicht in Betracht gezogen.

Zur weiteren Vereinfachung werden die Feldverteilungen entlang der Sprühelektrode (SE) als konstant angenommen, weswegen hier mit einem zweidimensionalen Modell gerechnet werden kann. Weiterhin wird infolge der geringen Strömungsgeschwindigkeiten von einer laminaren Strömung ausgegangen.

1.2 Geometrie und Vernetzung

Die programmatische Umsetzung der Geometrieerstellung sowie die darauf basierende Vernetzung erfolgt durch die Partial Differential Equation Toolbox™ (PDE-Tool) in MATLAB™.

Als Beispiel wird im Folgenden von einer zweidimensionalen Filtergeometrie nach dem Penney-Prinzip (VDI 3678 Blatt 2) ausgegangen wie in Abb. 2 dargestellt.

Das Penney-Prinzip zeichnet sich durch die Trennung der zwei Prozessstufen, der Partikelaufladung (Ionisator) und der Partikelabscheidung (Kollektor) mit einer jeweils separaten HV-Spannungsversorgung, aus. Die SE wird in diesem Beispiel als eine runde Drahtgeometrie angenommen wie auch die Kollektorelektrode (KE) als eine rechteckige Platte zwischen den beiden Niederschlagsselektroden (NE).

Die Vernetzung erfolgt durch eine automatische Dreiecksnetzerstellung basierend auf dem Delaunay-Triangulationsverfahren [3].

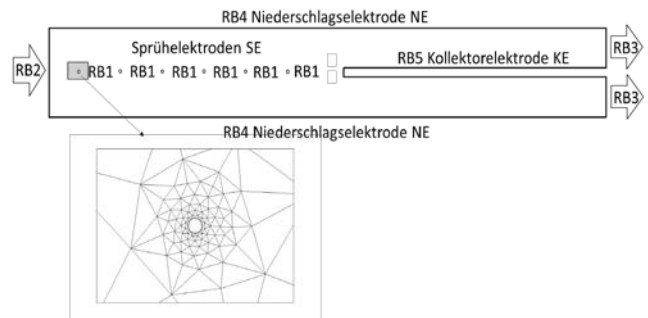


Abbildung 2: Beispiel einer zweidimensionalen Filterkanalgeometrie nach dem Penney-Prinzip

1.3 Strömungsfeld

Zur Vereinfachung des Strömungsmodells wird von einer Potentialströmung unter folgenden Annahmen ausgegangen:

- Reibungs- und rotationsfreie Strömung
- Inkompressibles Fluid
- Konstante Stoffeigenschaften des Fluides

Damit kann die Strömungsgeschwindigkeit als Gradient des Strömungspotentials ϕ dargestellt werden. In der Folge lässt sich das Strömungsfeld mit der Laplace-Gleichung mathematisch wie folgt beschreiben [2, 3]:

$$-\Delta\phi = 0 \quad (1)$$

In Bezug auf die ausgewählte Beispielgeometrie in Abb. 2 werden die nachfolgenden Randbedingungen (Rb) vorgegeben:

Rb1, Rb4, Rb5	Wand	$\Delta\phi_{Rand} \cdot \vec{n} = 0$
Rb2	Eingangsströmung	$-\Delta\phi_{Rand} = \vec{u}_{ein}$
Rb3	Ausgangsströmung	$-\Delta\phi_{Rand} = \vec{u}_{aus}$

Tabelle 1: Randbedingungen des Strömungsfeldes

Bei der Auswahl der Rb an den offenen Rändern muss beachtet werden, dass die Massenbilanz zwischen ein- und austretenden Flüssen der Kontinuitätsgleichung (Massenerhaltung) folgt.

Zur numerischen Lösung der Laplace-Gleichung (1) kommt erneut das PDE-Tool zum Einsatz, das verschiedene partielle Differentialgleichungen über die Finite-Elemente-Methode (FEM) numerisch lösen kann.

1.4 Elektrostatisches Feld

Das konservative E-Feld lässt sich mathematisch über die nachstehende Poisson-Gleichung beschreiben [4]:

$$\Delta V = -\frac{\rho_s}{\varepsilon} \quad (2)$$

Dabei kennzeichnet V die Spannung, ρ_s die Raumladungsdichte und ε die dielektrische Leitfähigkeit (Permittivität).

Da der Ionenstrom, wie oben schon erwähnt, nur sehr gering ist, kann als Randbedingung an der SE das elektrische Potential durch die Koronaeinsatzspannung V_0 angenähert werden. Dieser Näherungswert V_0 kann entweder durch den empirischen Ansatz von Peek [5] oder experimentell bestimmt werden. Die Spannung V_K an der KE wird durch den Anwender definiert und stellt einen Programmeingabeparameter dar. Des Weiteren werden die Spannung an der Gegenelektrode und die Raumladung auf null gesetzt wie auch die Feldwirkung an den Strömungsdurchlässen (vgl. Tab. 2).

Rb1	Potential an SE	$V_{Rand} = V_0$
Rb2, Rb3	Strömungsdurchlässe	$\vec{E}_{Rand} \cdot \vec{n} = 0$
Rb4	Potential an KE	$V_{Rand} = V_K$
Rb5	Potential an NE	$V_{Rand} = 0$

Tabelle 2: Randbedingungen des E-Feldes

Wie schon oben bei der Berechnung des Strömungsfeldes beschrieben, wird das PDE-Tool zur Lösung der Differentialgleichung (2) genutzt.

1.5 Ionenfeld

Ein Ionenfeld entsteht als Folge einer hohen elektrostatischen Feldstärke an der gekrümmten SE-Oberfläche. An dieser werden freie Elektronen durch die Feldwirkung beschleunigt, sodass diese mit Luftmolekülen (Sauerstoff oder Stickstoff) zusammenstoßen. Durch die Kollision entstehen weitere freie Elektronen (Stoßionisation), die wiederum beschleunigt werden und als Konsequenz einen Lawineneffekt auslösen (Avalanche-Durchbruch)[5], vgl. Abb. 3.

Die daraus resultierenden Ionen wandern aufgrund der elektrostatischen Feldwirkung zu den NE. Dadurch entstehen zwei Zonen: Zum einen die an der SE-Oberfläche liegende aktive Zone (engl. plasma region), bei der die oben beschriebenen Mikroprozesse stattfinden, zum anderen die passive Zone (engl. unipolar region), die durch die Makroprozesse des Ionentransportes (Konvektionsstrom) charakterisiert ist [4].

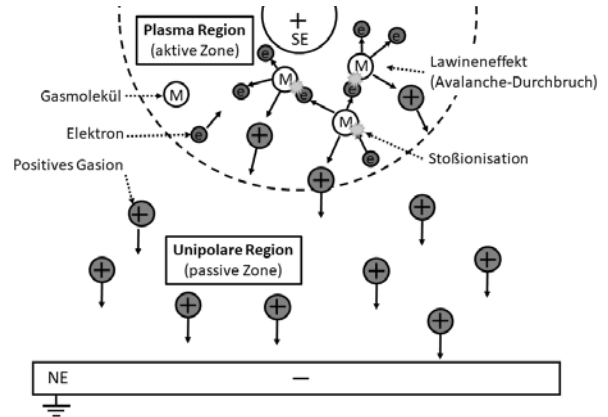


Abbildung 3: Schematische Darstellung der aktiven und der passiven Zone der positiven Koronentladung

In diesem Simulationsansatz werden die oben beschriebenen Mikroprozesse der aktiven Zone vernachlässigt und dessen Zonenbereich auf die Oberfläche der Sprühelektrode gelegt. Es wird nur der aus den Mikroprozessen entstehende Ionentransport der passiven Zone nachgebildet.

Dieser lässt sich über den Ladungserhaltungssatz in der Divergenzform, wie nachfolgend, mathematisch beschreiben:

$$\frac{\partial \rho_s}{\partial t} + \nabla \cdot \vec{j} = 0 \quad (3)$$

Dabei bildet \vec{j} in der oberen Gleichung die Stromdichte des Ionenfeldes ab und ∇ bezeichnet den Gradienten. Zur weiteren Vereinfachung werden die folgenden Annahmen bezüglich des Ionenfeldes getroffen [6, 7]:

- Die thermale Diffusion der Ionen wird vernachlässigt.
- Das magnetische Feld, das durch den Konvektionsstrom verursacht wird, wird nicht beachtet.
- Alle Ionen im elektrischen Feld sind unipolar geladen (hier positiv).
- Die Ionenbeweglichkeit wird konstant mit $\beta = 1,85 \cdot 10^{-4} \text{ m}^2 \text{ V}^{-1} \text{ s}^{-1}$ angenommen.

Infolgedessen lässt sich die Stromdichte durch die nachstehende Gleichung beschreiben [7]:

$$\vec{j} = \vec{E} \beta \rho_s \quad (4)$$

Das E-Feld in der oberen Formel setzt sich zudem aus einem äußeren \vec{E}_A und einem inneren E-Feld \vec{E}_I zusammen:

$$\vec{E} = \vec{E}_A + \vec{E}_I \quad (5)$$

In diesem Kontext repräsentiert das äußere E-Feld die Feldwirkung aufgrund des Potentialunterschieds an den Elektroden und das innere E-Feld die Feldwirkung der Ionen bzw. Raumladungen. Letztere stellen die Beziehung zwischen E-Feld und Ionenfeld dar.

Die Randbedingung an der SE wird durch die Ladungsdichte an der SE-Oberfläche definiert [6]:

$$\rho_w = \frac{I}{2\pi\beta r_i \vec{E}_K l} \quad (6)$$

Dabei wird der ionisierte Mantel r_i auf den SE-Radius gelegt [6]. Die Größe l steht für die Länge der SE und \vec{E}_K für die Feldstärke in der Korona [7]. Damit lässt sich der Konvektionsstrom I als Randbedingung für den Eingang an der SE wie folgt herleiten:

$$\vec{j} \cdot \vec{n} = \vec{E}_c \beta \rho_w \cdot \vec{n} = \frac{I}{2\pi r_i l} \quad (7)$$

Die Randbedingungen an den NE werden infolge des angewendeten Upstream-Verfahrens [3] anhand des jeweilig ortsnächsten Elementpunktes definiert. An den Strömungsdurchlässen wird der Stromdichtefluss auf null gesetzt wie auch die Startwerte der Raumladungen ρ_s . Nachfolgend sind die oben beschriebenen Randbedingungen tabellarisch zusammengefasst:

Rb1	Eingang SE	$\vec{j} \cdot \vec{n} = \frac{I}{2\pi r_i l}$
Rb4	Ausgang NE	$\vec{j} \cdot \vec{n} = \beta \rho \vec{E} \cdot \vec{n}$
Rb2, Rb3, Rb5	Wand	$\vec{j} \cdot \vec{n} = 0$

Tabelle 3: Randbedingungen des Ionenfeldes

Die Lösung der Transportgleichung (3) erfolgt über die Finite-Volumen-Methode (FVM) in MATLAB[™]. Dafür wird das Lösungsgebiet Ω in viele Teilgebiete Ω_i (finite Volumen) unterteilt (örtliche Diskretisierung) und die Stromdichte an den Grenzflächen bilanziert:

$$\int_{\Omega_i} \frac{\partial \rho_s}{\partial t} d\Omega_i + \int_{\Omega_i} \nabla \cdot \vec{j} d\Omega_i = 0 \quad (8)$$

Durch den gaußschen Integralsatz, eine zeitlichen Diskretisierung und weitere mathematische Umstellungen erhält man die nachstehende algebraische Gleichung für den Raumladungstransport:

$$\frac{\bar{\rho}_i^{n+1} - \bar{\rho}_i^n}{\Delta t} + \frac{1}{|\Omega_i|} \sum_{j=1}^{N(i)} \beta \bar{\rho}_i^n l_{j(i)} \vec{E}_{j(i)} \cdot \vec{n}_{j(i)} = 0 \quad (9)$$

$\bar{\rho}_i^n$ und $\bar{\rho}_i^{n+1}$ entsprechen der mittleren Raumladung des Elements jeweils für den Wert vor dem Zeitschritt Δt und dem Wert danach. Die Kantenlänge der Grenzfläche wird

mit $l_{j(i)}$ und die Elementfläche mit $|\Omega_i|$ gekennzeichnet. Die Anzahl der Terme $N(i)$ in der Summe entspricht der Anzahl der Grenzflächen des jeweiligen Elements i .

Die Lösung von Gleichung (9) erfolgt elementweise durch das Gauß-Jordan-Verfahren unter Berücksichtigung des Stabilitätskriteriums der Positivität [3] aus Gleichung (10), die eine obere Begrenzung für die Wahl des Zeitschritts liefert:

$$\Delta t < \frac{\Omega_i}{\sum_j^3 (l_{ij} \vec{E}_{ij} \beta)} S_F \quad (10)$$

Die Variable S_F steht in der oberen Gleichung für einen Sicherheitsfaktor (Dämpfung), der erfahrungsgemäß zwischen 1,1 und 2,0 liegen sollte.

Da das E-Feld und das Ionenfeld über die Feldwirkung der Raumladungen in Gleichung (5) miteinander gekoppelt sind, müssen beide Felder rekursiv berechnet werden. Die nachstehende numerische Lösungsfolge hat sich in dieser Arbeit bewährt:

1. Berechnung des äußeren E-Feldes \vec{E}_A durch Gleichung (2) mit $\rho_s(i) = 0$ und den Randbedingungen aus Tabelle 2.
2. Berechnung der Raumladungsverschiebungen nach Gleichung (9).
3. Berechnung des inneren E-Feldes \vec{E}_I durch Gleichung (2) mit den Raumladungen $\rho_s(i)$ als Startwerte und mit der Neumann Randbedingung gleich null für Rb1.
4. Berechnung des resultierenden E-Feldes nach Gleichung (5).
5. Wiederholung der Folge ab Punkt 2, bis das Konvergenzkriterium erfüllt ist, wie z. B. $\sum_i J_{ein} = \sum_i J_{aus}$.

1.6 Partikelflug

Auf Grundlage der oben beschriebenen Feldverteilungen können die Partikelflugbahnen (Trajektorien) mithilfe der lagrangeschen Bewegungsgleichungen berechnet werden. Die auf die Partikel einwirkenden relevanten Kräfte sind die Widerstandskraft \vec{F}_D , die den Partikelströmungsmittress charakterisiert und die coulombsche Kraft \vec{F}_C , die die Ablenkung der geladenen Partikel infolge des E-Feldes bestimmt.

Andere Kräfte wie die dynamische Auftriebskraft, die Druckkraft, die virtuelle Massenkraft, die hydrostatische Auftriebskraft, die Magnus-Kraft und die Basset (History)-Kraft bleiben u. a. aufgrund des hohen Dichteunterschieds der Partikel gegenüber des umgebenden Gases

unberücksichtigt [2]. Daraufhin lässt sich die Partikelbewegung aus dem Impulssatz (zweites Newtonsche Gesetz) wie folgt ableiten:

$$m \vec{Y} = \vec{F}_D + \vec{F}_C \quad (11)$$

Dabei steht m für die Partikelmasse und \vec{Y} für den Beschleunigungsvektor des Partikels.

Die coulombsche Kraft errechnet sich aus dem Produkt der Partikeleigenladung q und der Feldstärke $\vec{E} = \begin{bmatrix} E_x \\ E_y \end{bmatrix}$ nach:

$$\vec{F}_C = q \vec{E} \quad (12)$$

Die Größe q wird dabei im Wesentlichen durch den Aufladungsmechanismus im Ionenfeld definiert. Das Modell in dieser Arbeit bedient sich daher des kombinierten Aufladungsmodells nach Lawless [8], das sowohl die Feld- als auch Diffusionsaufladung der Partikel im Ionenfeld berücksichtigt.

Für die Widerstandskraft wird, unter der Annahme einer langsamen Strömung mit niedrigen Reynoldszahlen (schleichende Bewegung), die nachstehende Stokes-Gleichung verwendet [2]:

$$\vec{F}_D = -\frac{3\pi\eta d_p}{Cu} (\vec{Y} - \vec{U}) \quad (13)$$

Hierbei kennzeichnet \vec{Y} den Partikelgeschwindigkeitsvektor, η die dynamische Viskosität, d_p den Partikeldurchmesser und $\vec{U} = \begin{bmatrix} u \\ v \end{bmatrix}$ das Strömungsfeld. Die Cunningham-Korrektur Cu [9] berücksichtigt den molekularen Schlupf für Partikeldurchmesser von unter $0,5 \mu\text{m}$ und errechnet sich nach Gleichung (14):

$$Cu = 1 + 2 \frac{\lambda}{d_p} \left(A_1 + A_2 \cdot e^{-A_3 \frac{d_p}{2\lambda}} \right) \quad (14)$$

Die Größe λ beschreibt dabei die mittlere freie Weglänge der Gasteilchen. Für Luft gelten nach Davis die folgenden Koeffizienten $A_1 = 1,257$, $A_2 = 0,4$ und $A_3 = 1,1$ [10].

Aus den Gleichungen (11), (12) und (13) lassen sich zwei Partikelbewegungsgleichungen bezüglich der kartesischen Koordinatenrichtungen ableiten:

$$\ddot{x} = \frac{q}{m} E_x - \frac{3\pi\eta d_p}{mCu} (\dot{x} - u) \quad (15)$$

$$\ddot{y} = \frac{q}{m} E_y - \frac{3\pi\eta d_p}{mCu} (\dot{y} - v) \quad (16)$$

Dieses System gewöhnlicher Differentialgleichungen wird mit dem Runge-Kutta-Verfahren 4. Ordnung gelöst.

2 Modellvalidierung

2.1 Partikelabscheidungsrate

Zur Überprüfung bzw. Validierung des oben beschriebenen Modells wurde ein Zigarettenrauch-Experiment an einem einkanaligen EA mit dem Aufbau aus Abb. 2 durchgeführt.

Dazu wurde die Partikelabscheidungsrate bei einer Strömungsgeschwindigkeit von $1,0 \text{ ms}^{-1}$ und einer durchschnittlichen Partikeleingangskonzentration von ungefähr $\text{PM1} \approx 240 \mu\text{gm}^{-3}$ bestimmt.

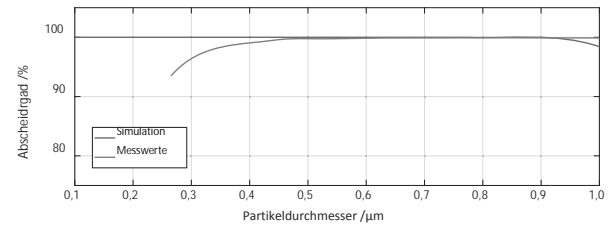


Abbildung 4: Vergleich zwischen experimentell (orange) und numerisch (blau) ermittelten Partikelabscheidungsgraden im PM1-Bereich

Der Ergebnisvergleich in Abb. 4 zeigt, dass das Simulationsmodell in einem weiten Bereich der Partikel sehr gute Übereinstimmungen mit den Messdaten aufweist. Allerdings weichen die Simulationsergebnisse im Partikelbereich von über $0,95 \mu\text{m}$ von diesen ab. Dies liegt vermutlich daran, dass größere Partikel in der Regel aus einer Agglomeration von vielen kleinen Partikeln bestehen und daher freie Stellen auf der Oberflächenstruktur aufweisen. Im Modell werden diese Partikel jedoch als blanke Kugeln mit einer experimentell ermittelten mittleren Partikeldichte von 1676 kgm^{-3} vorgegeben. Daher wird die Partikelmasse wahrscheinlich zu hoch angenommen, was in der Simulation zu einer erhöhten Trägheit der Partikelbewegung führt. Folglich erreichen die Partikel nicht die erforderliche Wanderungsgeschwindigkeit und durchdringen die Filtereinheit.

Die Abweichungen im Partikelgrößenbereich $< 0,5 \mu\text{m}$ sind durch einen hohen Ultrafeinstaubanteil im Zigarettenrauch und die demnach fehlerhaften Partikeldetektionen im unteren Messbereich des Partikelzählers (Model 1.108/1.109, Fa. Grimm) zu erklären.

2.2 Partikelabscheidungsverteilung

Die Simulation der Partikelflugbahnen ermöglicht u. a. die Bestimmung des Partikelabscheidungsortes auf den

NE. Dies lässt einen Vergleich der simulierten Abscheidungsverteilung mit den Ablagerungen an den NE aus dem Zigarettenrauch-Experiment zu:

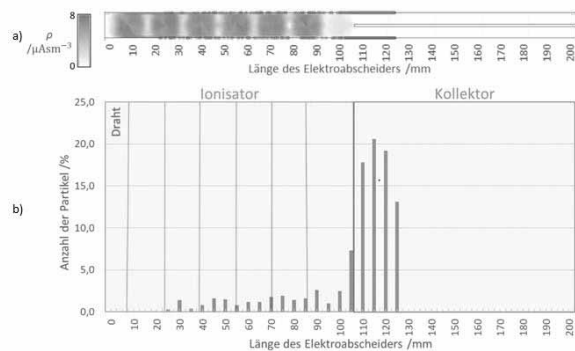


Abbildung 5: Simulationsergebnisse der Partikelabscheidungsverteilung: a) Darstellung des Ionienfeldes und der Abscheidungsorte durch die roten Punkte auf den NE, b) Partikelabscheidungsverteilung entlang der NE

Die Ergebnisse dieser Simulation sind in Abb. 5 veranschaulicht. Es zeigt sich, dass sich ein geringer Anteil der Partikel wellenförmig über den Ionisator (25 bis 105 mm) verteilt, sich jedoch der mehrheitliche Anteil am Übergang zwischen (105 bis 110 mm) und im vorderen Bereich des Kollektors (110 bis 125 mm) abscheidet.

Die Simulationsergebnisse weisen eine qualitativ gute Übereinstimmung mit den experimentell ermittelten Partikelablagerungen aus dem Experiment auf, wie die nachstehenden Bilder der NE belegen:

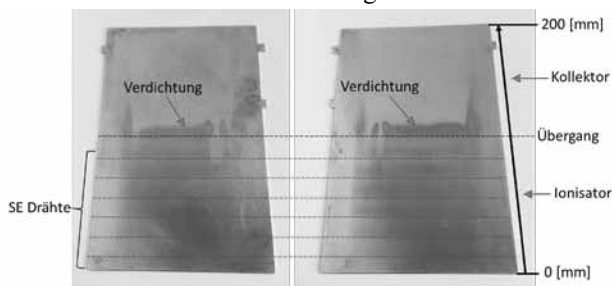


Abbildung 6: Partikelablagerungen an den NE

Auch hier finden sich die wellenförmigen Ablagerungen am Ionisator sowie eine Verdichtung der Partikelablagerung am vorderen Bereich des Kollektors. Sowohl die simulierten als auch die experimentell ermittelten Partikelabscheidungsverteilungen weisen darauf hin, dass die NE für diese Anwendung verkürzt werden könnten.

3 Fazit

Obwohl das vorgestellte Simulationsmodell vielen Vereinfachungen und Annahmen unterliegt, bildet es den Partikelflug im elektrostatischen Luftfilter hinreichend gut ab. Demnach kann das entwickelte Simulationsprogramm bei der Auslegung und der Optimierung von einfachen elektrostatischen Luftfilter-Geometrien eingesetzt werden, um die Entwicklungszeit neuer Prototypen zu verkürzen.

Für aufwändigere Geometrien und bei höheren Strömungsgeschwindigkeiten empfiehlt es sich, evtl. ein komplexeres Strömungsmodell (Navier-Stokes-Gleichungen) zu implementieren, da die Partikelflugbahnen signifikant durch das Strömungsfeld beeinflusst werden.

Der hier aufgezeigte Ansatz hat, falls er Anwendungsgültigkeit besitzt, demgegenüber aber den Vorteil, dass die Berechnungen numerisch deutlich weniger komplex und fehleranfällig sind und die Lösung damit schnell zur Verfügung steht.

Literatur

- [1] Riebel U, Lübbert C. Elektroabscheider- es gibt noch viel zu tun. Ch. Ing. Tech. 2012; 84(7): 1099-1113. DOI: 10.1002/cite.201100254
- [2] Laurien E, Oertel H. Numerische Strömungsmechanik. 5. Auflage. Wiesbaden: Springer Vieweg; 2013. 317 S.
- [3] Malcherek A. Numerische Methoden der Strömungsmechanik. 1. Auflage. London: Bockboon, 2014. 307 S.
- [4] Leuchtman, P. Einführung in die elektr. Feldtheorie. 1. Auflage. Freising: Pearson Studium, 2007. 602 S.
- [5] White HJ. Entstaubung industr. Gase mit Elektrofiltern. Leipzig: D. Verlag für Grundstoffindustrie, 1969. 336 S.
- [6] Khaled U, Eldein ZE. Experimental study of V-I characteristics of wire-plate electrostatic precipitators under clean air conditions. J. of Elec. 2013; 71(3): 228-234. DOI: 10.1016/j.elstat.2012.12.023
- [7] McDonald JR, Smith WB. A mathematical model for calculating electrical conditions in wire-duct electrostatic precipitation devices. J. of Appl. Phys. 1977; 48(6): 2231-2243. DOI: 10.1063/1.324034
- [8] Lawless PA. Particle charging bounds, symmetry relations, and an analytic charging rate model for the continuum regime. J. Aerosol Sci. 1996; 27(2): 191-215. DOI: 10.1016/0021-8502(95)00541-2
- [9] Cunningham E. On the Velocity of Steady Fall of Spherical Particles through Fluid Medium. Royal Society. 1910; 83(563): 357-365. DOI: 10.1098/rspa.1910.0024
- [10] Davies CN. Definitive equations for the fluid resistance of spheres. Proc. Phys. Soc. 1945; 57(4): 259-270. DOI: 10.1088/0959-5309/57/4/301

Simulationsansätze zur Funktionsintegration bei elektrohydraulischen Linearachsen

Florian Meyer*¹, Andreas Ligocki¹

¹Institut für Konstruktion und angewandten Maschinenbau, Ostfalia Hochschule für angewandte Wissenschaften, Salzdahlumer Straße 46/48, 38302 Wolfenbüttel; *florian.meyer@ostfalia.de

Abstract. Ein Forschungsprojekt der Ostfalia Hochschule für angewandte Wissenschaften – Fakultät Maschinenbau, welches in Kooperation mit der Berode GmbH durchgeführt wird, befasst sich mit der Entwicklung einer autarken, elektrohydraulischen Linearachse. Der Schwerpunkt des Projekts liegt auf der Integration von Elektromotor und Hydraulikpumpe zu einer „MotorPumpe“. Diese kann direkt im Arbeitszylinder untergebracht werden und versorgt diesen nach Bedarf direkt mit hydraulischer Energie. Dabei werden die hydraulischen Anschlussleitungen überflüssig, da das System nur mit elektrischer Energie zu versorgen ist.

Der Beitrag zeigt das geförderte Forschungsprojekt sowie die Ansätze für den Aufbau der Simulation des Hybridsystems unter Einsatz der Software „LMS Imagine.Lab Amesim“ und „Matlab/Simulink“ insbesondere mit dem Fokus auf die Methoden zur Modellbildung und Simulation der „MotorPumpe“.

Einleitung

Die Optimierung der Antriebssysteme mobiler Maschinen und die Hybridisierung zur Effizienzsteigerung stehen seit geraumer Zeit im Fokus von Entwicklungsprojekten. Dabei ist der gegenwärtige, verstärkte Entwicklungstrend auf das wachsende Umweltbewusstsein, die steigenden Kosten für fossile Brennstoffe sowie auf den Druck durch bestehende und zu erwartende CO₂-Regulierungen zurückzuführen. Es werden zunehmend hydraulische Komponenten und Systeme von elektrischen, flexibel handhabbaren Systemen abgelöst. Diese elektrischen Systeme können einfacher angesteuert werden, bieten im Vergleich zur konkurrierenden Hydraulik keinerlei Leckagestellen und sind in der Lage, die geforderte Leistung on Demand bereitzustellen. In vielen Fällen erreichen Sie jedoch nicht die erforderliche Leistungs-

dichte; dann werden zusätzliche ggf. mehrstufige Getriebe erforderlich.

Besonders elektrische Linearantriebe verlieren beim Einsatz solcher Wandler ihre Einfachheit sowie deutlich an Effizienz und somit an Attraktivität gegenüber den hydraulischen Systemen, welche ihre Vorteile insbesondere bei hohen Stellkräften und hoher, erforderlicher Leistungsdichte ausspielen. Diesen Entwicklungstrend haben einige Hersteller für hydraulische Systeme erkannt und bieten elektrohydraulische Linearachsen auf Basis kombinierter Elektromotoren, Hydraulikpumpen und Zylinder an. Standardkomponenten werden zu kompakt angeordneten Einheiten, meist im Verbundgehäuse, zusammengefasst (s. Abbildung 1). Die hybride Gestaltung der Linearachsen ist ein Schritt u. a. zur Verbesserung der Energieeffizienz bei hydraulischen Antrieben. Dabei bieten die bereits erhältlichen Lineareinheiten noch Integrationspotenzial.

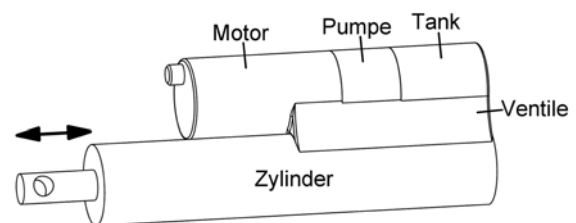


Abbildung 1: Aufbau der am Markt erhältlichen Linearachsen [1]

1 Forschungsprojekt

Dieses Integrationspotenzial greift ein Forschungsprojekt an der Ostfalia Hochschule für angewandte Wissenschaften auf. Das Ziel des Forschungsprojekts ist es, alle für den Betrieb eines Hydraulikzylinders erforderlichen Komponenten in den typischen Zylinderaufbau zu integrieren, um die charakteristische, platzsparende Zylinderform zu erhalten.

Der Schwerpunkt des Projekts liegt auf der Fusionierung von Elektromotor und Hydraulikpumpe zu einer „MotorPumpe“. Diese wird bezüglich der zum Einsatz kommenden Maschinenelemente nicht mehr wie bisher trennbar sein. Der Läufer des Motors und das Pumpenrad verschmelzen und ermöglichen so eine wesentlich höhere Funktionsintegration. Die Verfahrensgeschwindigkeit des Zylinders kann über die Drehzahl des elektromotorischen Teils variiert werden. Hydraulikzylinder und Tank sowie Hydraulikpumpe und Elektromotor werden fusioniert. Infolge dessen wird der Bauraumzuwachs, im Vergleich zu den seitlich angeordneten Versorgungseinheiten bereits erhältlicher Produkte, deutlich minimiert.

Speziell für diese Bereiche des Projekts wird ein Simulationsmodell entwickelt, das die Untersuchung verschiedener Baugrößen auf Modellebene ermöglicht und so den konstruktiven sowie den Versuchsaufwand minimiert. Zudem wird mithilfe dieses Modells eine Analyse unterschiedlicher Leistungsklassen der „MotorPumpe“ ermöglicht. Das im Projekt entwickelte Simulationsmodell wird durch die Integration der Versuchsdaten anwendungsorientiert für die Weiterentwicklung des Gesamtsystems eingesetzt.

Gliederung des Projekts. Das Forschungsprojekt gliedert sich in drei Abschnitte [1, 2]. Der erste Abschnitt ist der Aufbau eines Versuchsstands zur Analyse einzelner Elektromotoren und Hydraulikpumpen (s. Abbildung 2).

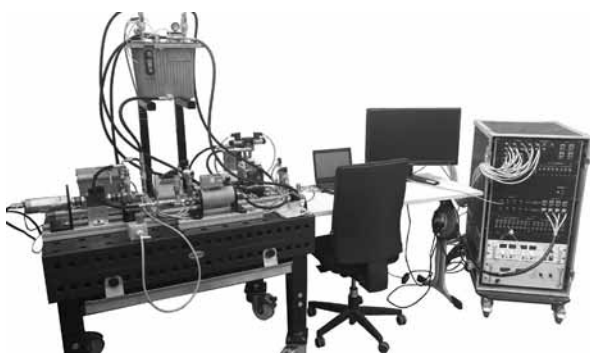


Abbildung 2: Versuchsstand zu Analyse der Einzelkomponenten [2]

Die Versuchsmessungen dienen der Ermittlung der optimalen Kombination der Einzelkomponenten für die Referenzanwendung. Parallel wird ein übergeordnetes Simulationsmodell für das Gesamtsystem entwickelt. Es begleitet kontinuierlich den Prozess der Entwicklung und

wird mithilfe der Messdaten validiert. Darauffolgend wird die Entwicklung einer elektrohydraulischen Linearachse, Integrationsstufe 1 (s. Abbildung 3), auf Basis von Kaufkomponenten zum Aufbau von Know-how durchgeführt. Entsprechend der Referenzanwendung wird die Simulation hin zur Abbildung dieser Entwicklungsstufe weiterentwickelt.

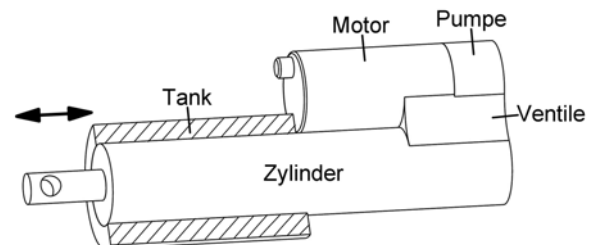


Abbildung 3: Integrationsstufe 1 [1]

Der dritte Abschnitt ist die Entwicklung einer hochintegrierten elektrohydraulischen Linearachse (s. Abbildung 4). Hierbei wird eine "MotorPumpe" entwickelt, die ein Hybrid aus Elektromotor und Hydraulikpumpe darstellt. Die Linearachse aus Integrationsstufe 1 wird weiterentwickelt um die „MotorPumpe“ zu integrieren.

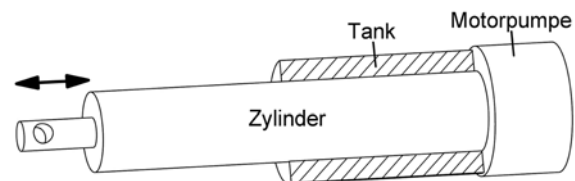


Abbildung 4: Integrationsstufe 2 [1]

2 Simulation und Methodik

Die Simulation wird als begleitendes Werkzeug während der Entwicklung in allen Projektabschnitten eingesetzt. Als Basis für die Entwicklung der Simulation werden die Software „MATLAB/Simulink“ und „LMS Imagine.Lab Amesim“ eingesetzt.

Die Ziele, die mit der Entwicklung des Simulationsmodells verfolgt werden, sind das Prüfen der Konstruktionsansätze innerhalb der Projektabschnitte und die anschließende Weiterentwicklung der Simulation zu einem Vorauslegungstool. Damit können zukünftige Entwicklungsarbeiten bei der Variantenentwicklung von elektrohydraulischen Linearachsen durch bspw. Parametervariation unterstützt werden. Die Vorhersage der Leistungs-

fähigkeit einer vorkonzipierten elektrohydraulischen Linearachse ist hierbei ein Kernaspekt, der im Fokus der Modellentwicklung steht.

Die Ausbaustufen der Simulation orientieren sich an den Entwicklungsstufen im Projekt. So wird zunächst der Versuchsstand als übergeordnetes Modell erarbeitet und anschließend zur Abbildung der Integrationsstufe 1 weiterentwickelt. Parallel entsteht ein Modell für die „MotorPumpe“ als ein Teilsystem, das anschließend in das übergeordnete Modell implementiert wird. Auf Basis der Versuchsdaten der Einzelkomponenten und den Integrationsstufen werden u.a. die Parameter für das Teilmodell der „MotorPumpe“ entsprechend modifiziert und validiert. Hierzu werden die Versuchsreihen auf die Anforderungen der Referenzanwendungen abgestimmt.

Während dieser Entwicklungsphasen werden in ausgewählten Bereichen spezialisierte Software genutzt, um Teilbereiche genauer zu analysieren. Dazu werden u. a. die Software „ANSYS“ und „FEMM“ eingesetzt. In der Software ANSYS ist bspw. die Leistungsfähigkeit eines aktiven und passiven Kühlsystems untersucht worden. Die Software FEMM wurde zur konkreten Analyse der Auslegung des elektromotorischen Teils genutzt.

Das übergeordnete Modell ermöglicht den Test der selbst programmierten Teilmodelle (MotorPumpe, Leitungssysteme, Ventile etc.) in der Systemumgebung des Gesamtmodells.

Die Software „LMS Imagine.Lab Amesim“ ist für die beschriebene Aufgabe sehr gut geeignet. Sie bietet die Möglichkeit auf verschiedenen Abstraktionsebenen Modelle aufzubauen. So ist es möglich, auf der höchsten Ebene mit Schalplanbasierten Elementen zu arbeiten (s. Abbildung 5). Diese Elemente ermöglichen bereits durch die Anpassung vorgegebener Parameter eine grobe Annäherung des Systems.



Abbildung 5: Beispiel schaltplanbasierte Elemente (links: eine Signalquelle; mittig: ein einfacher Motor; rechts: ein Rückschlagventil)

Teils sind auch komplexere Elemente verfügbar, die eine erweiterte Anpassung ermöglichen. Zur Verfeinerung des Modells ist es möglich mit dem „Component-Design“ den inneren Aufbau von Komponenten spezialisierter zu modellieren. Abbildung 6 zeigt am Beispiel ei-

nes Rückschlagventils links die Kammer mit der Vorspannfeder, die bei Bedarf mit Druck beaufschlagt werden kann. Mittig ein Masseelement, über das bspw. die Masse der Kugel auf der rechten Seite definiert werden kann. Am rechten Element können der Kugeldurchmesser, die Bohrungsdurchmesser, der maximale Öffnungsbereich und weitere Parameter definiert werden.

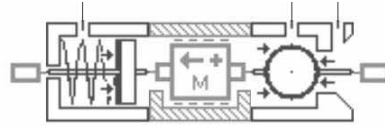


Abbildung 6: Beispiel Component-Design eines Rückschlagventils

Für die Teilelemente, bei denen mit dem Component-Design das Systemverhalten nicht ausreichend genau abgebildet werden kann, ist es möglich eine Schnittstelle zu „MATLAB/Simulink“ zu definieren. Somit können diese Elemente dort frei entwickelt werden. In diesem Fall wird eine Co-Simulation zwischen den Softwaretools aufgebaut. Dabei kann der Solver jeweils in der Hauptumgebung der Programme genutzt werden. Alternativ kann auch das Modell aus einer Software als Blackbox in die zweite Software übertragen werden. Folglich ist für jede Änderung am Modell ein erneuter Export erforderlich. Bei der eingesetzten Software kann das Modell aus „LMS Imagine.Lab Amesim“ als Blackbox in „MATLAB/Simulink“ importiert werden. Alternativ kann eine Co-Simulation aufgebaut werden, bei der die beiden Programme verlinkt sind. Hierbei fungiert „MATLAB/Simulink“ als Master.

Für die Entwicklung des Simulationsmodells der „MotorPumpe“ wurde eine Co-Simulation aufgebaut. Abbildung 7 zeigt vereinfacht die grundlegende Struktur, die für die Verknüpfung der Programme mit den einzelnen Teilmodellen entwickelt wurde. Die Elemente um die MotorPumpe sind hierfür zunächst schaltplanbasiert einfach gehalten (blaue Elemente). Die Software „LMS Imagine.Lab Amesim“ dient als Basis aus der ein verknüpftes Modell mit „MATLAB/Simulink“ erstellt wird. Zudem zeigt die Darstellung zwischen welchen Teilmodellen Daten, teils über die Softwaregrenzen hinaus, übergeben werden müssen.

Simulationsmodell MotorPumpe. Für das Simulationsmodell der MotorPumpe wurden die Modellschnittstellen innerhalb des Gesamtmodells festgelegt, damit die softwareübergreifende Simulation ermöglicht

wird. Weiter wurden die Modellstrukturen zur Abbildung der einzelnen Teilbereiche für den Elektromotor und die Hydraulikpumpe in „MATLAB/Simulink“ entwickelt. Die korrekte Definition der Schnittstellen ermöglicht dabei die disziplinübergreifende Verarbeitung der Werte zwischen den elektromotorischen und hydraulischen Teilen der Simulation.

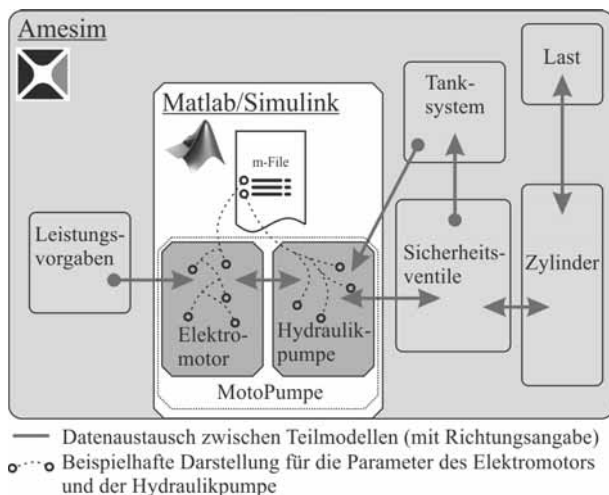


Abbildung 7: Grundstruktur der Simulation mit Verknüpfung der Software anhand der übergeordneten Teilmodelle

Die Teilmodelle werden über ein m-File parametrierbar. Mit dem m-File können die Kennwerte für die innere Mechanik bzw. die innere Fluidführung sowie für den elektromotorischen Teil individuell angepasst werden. Dabei ist es möglich sowohl die Grundkonfiguration des Elektromotors als auch die speziellen Werte für die Bewicklung und die magnetische Konfiguration zu variieren. Ergänzend zu den bereits genannten Punkten können für den hydraulischen Teil weitere Parameter und Abmessungen einzelner Pumpenelemente gezielt variiert werden. Außerdem können spezielle Modellparameter für die Spezifikation des hydraulischen Teils zur Berücksichtigung der internen Leckage angepasst werden. Abbildung 8 zeigt beispielhaft einen Ausschnitt aus dem Teilmodell der Hydraulikpumpe in „MATLAB/Simulink“.

Für die Spezifikation des elektromotorischen Teils der „MotorPumpe“ können die allgemeinen Parameter ähnlich angepasst werden. Dazu zählen die Spannungsversorgung sowie spezielle Parameter für die Anzahl der Nuten, Pole, Windungen und Leiter. Zusätzlich sind sehr spezielle Werte für die magnetische Flussdichte und den zulässigen Strangstrom vorgesehen, welche in teils sehr

aufwendigen experimentellen Versuchen für die konkrete Anwendung ermittelt wurden. Anhand der Erkenntnisse wurden das Simulationsmodell und das Funktionsmuster optimiert.

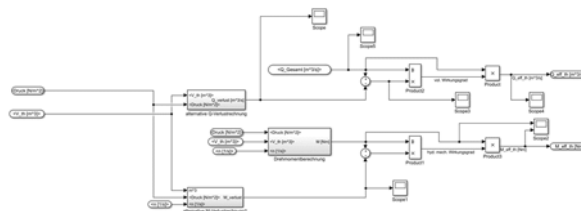


Abbildung 8: Beispiel Teilmodell Hydraulikpumpe

3 Zusammenfassung/ Ausblick

Der Beitrag zeigt die Möglichkeiten zum Aufbau der Simulation mit der Software „LMS Imagine.Lab Amesim“ und „MATLAB/Simulink“ in unterschiedlichen Abstraktionsebenen. Außerdem wird die Methodik zur Entwicklung des Simulationsmodells der Motor-Pumpe erläutert. Hierbei werden einige Parameter, die über ein m-File angepasst werden können dargestellt. Darüber hinaus wird Grundstruktur des Simulationsmodells mit der Verknüpfung der Software und ein Auszug aus dem Teilmodell der Hydraulikpumpe gezeigt.

Ein Vergleich der Simulation mit den Messdaten des Funktionsmusters befindet sich in der Vorbereitung. Hierbei sind u. a. die Messdaten der magnetischen Flussdichte in der Simulation und der Entwicklung des Funktionsmusters berücksichtigt worden. Dieser Vergleich stellt den nächsten logischen Schritt zur Validierung des Simulationsmodells dar.

References

- [1] Meyer F.; Ligocki A.; Frerichs L.: Ein Ansatz für eine neuartige elektrohydraulische Linearachse, Tagungsband - 10. Kolloquium Mobilhydraulik, Technische Universität Braunschweig, Institut für mobile Maschinen und Nutzfahrzeuge, Braunschweig, Deutschland, 2018
- [2] Meyer F.; Ligocki A.: Der autarke elektrohydraulische Linear-Aktor, Mobile Maschinen, Vereinigte Fachverlage, Mainz, Deutschland, Nr. 5 2018, S. 42-45

Förderung. Das ZIM-Projekt der Ostfalia und der Berode GmbH wird gefördert durch das Bundesministerium für Wirtschaft und Energie (BMWi) sowie Partner aus dem Bereich der Hebeteknik, der Medizintechnik und dem Werkzeugbau.



Model-based time varying predictive vibration control of a beam structure subjected to moving masses

Lukas Sievert^{1*}, Dan Stancioiu¹

¹Department of Maritime and Mechanical Engineering, Liverpool John Moores University, 3 Byrom Street, Liverpool L3 3AF, United Kingdom; **l.sievert@2017ljmu.ac.uk*

Abstract. This paper is concerned with the active vibration control of a structure subjected to moving loads, which is subject in many engineering applications. The present study presents the model of the Euler-Bernoulli simply supported beam excited by a moving mass and the design of a time varying predictive vibration controller.

One of the major challenges in designing a controller for a structure under a moving mass is that the system associated is a time-varying system, therefore linear time invariant solutions can lead to unsuitable results, especially at high travelling speeds and for certain locations of the actuators.

The approach used in this investigation is to calculate the control gains with a receding horizon model predictive control (MPC) approach during the time the mass acts upon the structure. In this way the controller takes into account the time-varying character of the problem. The results are compared numerically with a linear quadratic controller design.

Introduction

Active vibration control of structures subjected to moving loads has raised high interest in many fields of engineering, mostly in vehicle bridge interaction but also crane dynamic, robotics and aerodynamics.

In [1] a brief introduction into the problem of vibrations caused by moving loads is presented and the linear time-varying model is derived. A state space representation is used in [2,3]. Compared to the general linear time invariant vibration control approaches in [4,5] the time varying nature of the excitation leads to a more demanding control approach [6,7].

In [8,6] the time invariant quadratic optimal control approach is compared with a time varying optimal control. Especially at high traveling speeds and at a higher number of actuators the time-varying optimal control leads to better results.

The MPC approach includes future states of the system

into the optimization process. In [9] and [10] the MPC approach is studied and implemented successfully on linear time-invariant vibrating beam systems. To obtain the states and the system matrices a Kalman filter and real time system identification is used.

This paper applies the receding horizon MPC to a moving load structure, and compares it to the time variant and time invariant Linear Quadratic Regulator (LQR) control approach.

1 Problem formulation

This study is based on an Euler Bernoulli modelled beam structure, shown in Figure 1, of length L traversed by a mass m traveling at constant speed v at any time t within the time interval $[0, t_f]$ with $t_f=L/v$ assuming there is no contact loss [1,8]. The control force is defined as u .

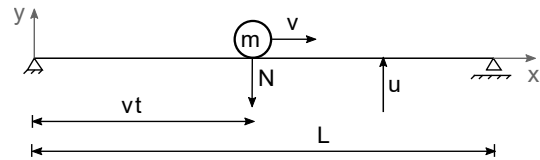


Figure 1: A model of a beam subjected to a moving mass

In modal coordinates the general system of equations is:

$$(\mathbf{M}+\Delta\mathbf{M}(t))\ddot{\mathbf{q}}+(\mathbf{D}+\Delta\mathbf{D}(t))\dot{\mathbf{q}}+(\mathbf{K}+\Delta\mathbf{K}(t))\mathbf{q} \quad (1)$$

$$=-mg\boldsymbol{\psi}(vt)-\sum_{i=1}^k\boldsymbol{\psi}(x_{ai})U_i$$

Here the constant matrices \mathbf{M} , \mathbf{D} and \mathbf{K} are expressed as functions of the modal shape vectors $\boldsymbol{\psi}(x)$, mass per unit length ρA , damping $c\rho A$ and Stiffness EI .

$$\begin{aligned}
\mathbf{M} &= \rho A \int_0^L \boldsymbol{\Psi}(x) \boldsymbol{\Psi}^T(x) dx \\
\mathbf{D} &= \rho A c \int_0^L \boldsymbol{\Psi}(x) \boldsymbol{\Psi}^T(x) dx \\
\mathbf{K} &= EI \int_0^L \boldsymbol{\Psi}(x) \boldsymbol{\Psi}^T(x) dx
\end{aligned} \tag{2}$$

The time-dependent matrices $\Delta \mathbf{M}(t)$, $\Delta \mathbf{D}(t)$ and $\Delta \mathbf{K}(t)$ are defined as

$$\begin{aligned}
\Delta \mathbf{M} &= m \boldsymbol{\Psi}(vt) \boldsymbol{\Psi}^T(vt) \\
\Delta \mathbf{D} &= 2mv \boldsymbol{\Psi}(vt) \boldsymbol{\Psi}'^T(vt) \\
\Delta \mathbf{K} &= mv^2 \boldsymbol{\Psi}(vt) \boldsymbol{\Psi}''^T(vt)
\end{aligned} \tag{3}$$

Equation (1) can be expressed in the state space representation to (4)

$$\begin{aligned}
\mathbf{A}_m(t) &= \begin{bmatrix} \mathbf{0}_{n \times n} & \mathbf{I}_{n \times n} \\ -(\mathbf{M} + \Delta \mathbf{M})^{-1}(\mathbf{K} + \Delta \mathbf{K}) & -(\mathbf{M} + \Delta \mathbf{M})^{-1}(\mathbf{D} + \Delta \mathbf{D}) \end{bmatrix} \\
\mathbf{B}_m(t) &= \begin{bmatrix} \mathbf{0}_{n \times n} \\ -(\mathbf{M} + \Delta \mathbf{M}(t))^{-1} [\boldsymbol{\Psi}(x_{a1}) \dots \boldsymbol{\Psi}(x_{ak})] \end{bmatrix}
\end{aligned} \tag{4}$$

The deflection is $w(x_s, t) = \mathbf{C}_m [\mathbf{q}(t) \mathbf{q}'(t)]^T$ at the location x_s with the output matrix is $\mathbf{C}_m = [\boldsymbol{\Psi}(x_s)' \ 0]$. After the time t_f the mass leaves, the beam vibrates free. The linear time-invariant system equation for $t > t_m$ is

$$\mathbf{M} \ddot{\mathbf{q}} + \mathbf{D} \dot{\mathbf{q}} + \mathbf{K} \mathbf{q} = - \sum_{i=1}^k \boldsymbol{\Psi}(x_{ai}) u_i \tag{5}$$

2 Control algorithms

For the linear classical optimal control algorithm with displacement-velocity feedback the gain vector \mathbf{k} is calculated by minimizing the performance index

$$J = \int_0^{\infty} (\mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{u}^T \mathbf{R} \mathbf{u}) dt \tag{6}$$

The gain vector is expressed as

$$\mathbf{k} = \mathbf{R}^{-1} \mathbf{B}^T \mathbf{P} \tag{7}$$

Where \mathbf{P} is obtained by solving the following Riccati equation.

$$\mathbf{A}^T \mathbf{P} + \mathbf{P} \mathbf{A} - \mathbf{P} \mathbf{B} \mathbf{R}^{-1} \mathbf{B}^T \mathbf{P} + \mathbf{Q} = 0 \tag{8}$$

For the time invariant system, the system matrixes \mathbf{A} and \mathbf{B} do not change, \mathbf{k} is calculated offline.

In the time-varying system, the algebraic Riccati equation is solved at every time-step t_i .

$$\mathbf{A}^T(t_i) \mathbf{P} + \mathbf{P} \mathbf{A}(t_i) - \mathbf{P} \mathbf{B}(t_i) \mathbf{R}^{-1} \mathbf{B}^T(t_i) \mathbf{P} + \mathbf{Q} = 0 \tag{9}$$

For the receding horizon MPC approach an augmented discrete state space model is used[11]:

$$\begin{aligned}
\begin{bmatrix} \Delta x_m(k+1) \\ y(k+1) \end{bmatrix} &= \begin{bmatrix} \mathbf{A}_m & \mathbf{0}_m^T \\ \mathbf{C}_m \mathbf{A}_m & 1 \end{bmatrix} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} \\
&+ \begin{bmatrix} \mathbf{B}_m \\ \mathbf{C}_m \mathbf{B}_m \end{bmatrix} \Delta u(k) \\
y(k) &= \begin{bmatrix} \mathbf{0}_m & 1 \end{bmatrix} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix}
\end{aligned} \tag{10}$$

Where Δx_m is the difference of the state variables.

$$\begin{aligned}
\Delta x_m(k+1) &= x_m(k+1) - x_m(k); \\
\Delta x_m(k) &= x_m(k) - x_m(k-1)
\end{aligned} \tag{11}$$

The future control trajectory vector $\Delta \mathbf{U}$ is denoted by:

$$\Delta \mathbf{U} = [\Delta u(k_i), \Delta u(k_i+1), \dots, \Delta u(k_i+N_c-1)]^T \tag{12}$$

Where N_c is called the control horizon, defining the number of future trajectory points. The prediction horizon N_p defines the number of future states $x(k_i+m|k_i)$ calculated with the given state vector $\mathbf{x}(k_i)$ and the length of the optimization window. The future output variables $\mathbf{Y} = [y(k_i+1|k_i), \dots, y(k_i+N_p|k_i)]^T$ is calculated by

$$\mathbf{Y} = \mathbf{F} \mathbf{x}(k_i) + \boldsymbol{\Phi} \Delta \mathbf{U} \tag{13}$$

where

$$\begin{aligned}
\mathbf{F} &= \begin{bmatrix} \mathbf{C} \mathbf{A} \\ \mathbf{C} \mathbf{A}^2 \\ \mathbf{C} \mathbf{A}^3 \\ \vdots \\ \mathbf{C} \mathbf{A}^{N_p} \end{bmatrix}; \\
\boldsymbol{\Phi} &= \begin{bmatrix} \mathbf{C} \mathbf{B} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{C} \mathbf{A} \mathbf{B} & \mathbf{C} \mathbf{B} & \dots & \mathbf{0} \\ \mathbf{C} \mathbf{A}^2 \mathbf{B} & \mathbf{C} \mathbf{A} \mathbf{B} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{C} \mathbf{A}^{N_p-1} \mathbf{B} & \mathbf{C} \mathbf{A}^{N_p-2} \mathbf{B} & \dots & \mathbf{C} \mathbf{A}^{N_p-N_c} \mathbf{B} \end{bmatrix}
\end{aligned} \tag{14}$$

The cost function J that reflect the control objective is defined as

$$J = \mathbf{Y}^T \mathbf{Y} + \Delta \mathbf{U}^T \bar{\mathbf{R}} \Delta \mathbf{U} \quad (15)$$

By minimizing equation (15) the new control vector ΔU is

$$\Delta \mathbf{U}(k_i) = (\Phi^T \Phi + \bar{\mathbf{R}})^{-1} \Phi^T (-\mathbf{F}x(k_i)), \quad (16)$$

The tuning parameter matrix $\bar{\mathbf{R}} = r_w \mathbf{I}_{N_c \times N_c}; r_w \geq 0$ is used to meet the desired closed-loop performance. Considering the time variation of the system matrixes $\mathbf{F}(k_i)$ and $\phi(k_i)$ change at every time step k_i due to time variation of the system matrixes $\mathbf{A}(k_i)$, $\mathbf{B}(k_i)$ and $\mathbf{C}(k_i)$. The change of the control force is

$$\Delta U(k_i) = (\Phi(k_i)^T \Phi(k_i) + \bar{\mathbf{R}})^{-1} \Phi(k_i)^T (-\mathbf{F}(k_i)x(k_i)), \quad (17)$$

3 Experimental results

To test the vibrational control methods, an experimental setup is installed, shown in Figure 2. A point mass m is moving at a constant speed v along the span length L . Laser displacement sensors measure the deflection response at various points x_{si} . The control force is applied at $x_a=0.5\text{m}$. The geometrical and dynamical characteristics of the beam are: span length $L = 0.6 \text{ m}$, mass per length unit $\rho A = 0.371 \text{ kg} \times \text{m}^{-1}$ and flexural rigidity $EI = 3.903 \text{ Nm}^{-2}$. A constant modal damping coefficient $\zeta = 0.2$ is assumed throughout. For a simply supported beam the i th mode $\psi_i(x) = \sin(i\pi x/L)$.

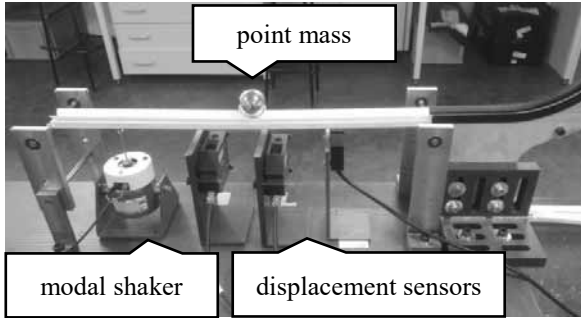


Figure 2: Experimental setup Data Physik V02 modal shaker and Micro Epsilon displacement sensors

Figure 3 shows the deflection responses measured at $x_{s1} = 0.15 \text{ m}$ and $x_{s2} = 0.25\text{m}$. A mass $m = 0.206 \text{ kg}$ moves along the beam at a speed $v=0.869\text{m/s}^2$. The numerical model shows a good match with the experimentally measured deflections.

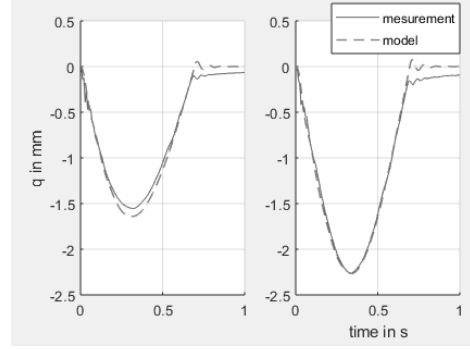


Figure 3: Deflection response at the sensor locations x_{s1} (left) and x_{s2} (right), continuous line – measurements, dashed line numerical model

The feasibility of the vibration control methods is studied numerically. For the time-invariant control the constant control matrix \mathbf{k} is determined with

$$\mathbf{A} = \begin{bmatrix} \mathbf{0}_{n \times n} & \mathbf{I}_{n \times n} \\ -\mathbf{M}^{-1}(\mathbf{K}) & -\mathbf{M}^{-1}\mathbf{D} \end{bmatrix} \quad (18)$$

$$\mathbf{B} = \begin{bmatrix} \mathbf{0}_{n \times n} \\ -\mathbf{M}^{-1}[\psi(x_{a1}) \dots \psi(x_{ak})] \end{bmatrix}$$

For the LQR control the error matrix is defined as $\mathbf{Q} = \text{diag}(1000, 100, 10, 0, 0, 0)$ and the control weighting matrix as $\mathbf{R} = 1 \times 10^{-6}$. The control weighting matrix for the MPC is $\bar{\mathbf{R}} = 1 \times 10^{-8} \mathbf{I}_{N_c \times N_c}$. The control horizon $N_c = 4$ and the prediction horizon $N_p = 8$. Relatively small values of N_c and N_p reduce the reaction time and the computation time. The sampling time is $t_s = 0.005\text{s}$.

Figure 4 shows the deflection of the system with the applied vibration controls. The MPC achieves the lowest deflection. Further reducing of \mathbf{R} dose not decrease the deflection of the LQR controls significantly.

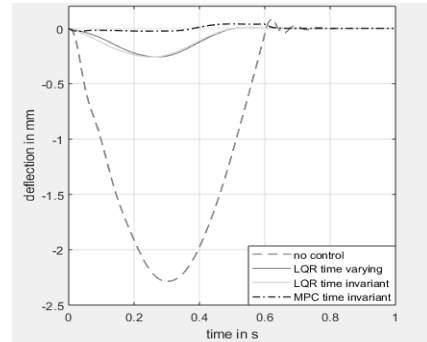


Figure 4: Time history of the mid-point deflection of the structure, quadratic control method compared with the MPC method, $m = 0.2 \text{ kg}$, $v = 1 \text{ m/s}^2$

Figure 5 shows the comparison of the input control forces of the LQR and MPC control.

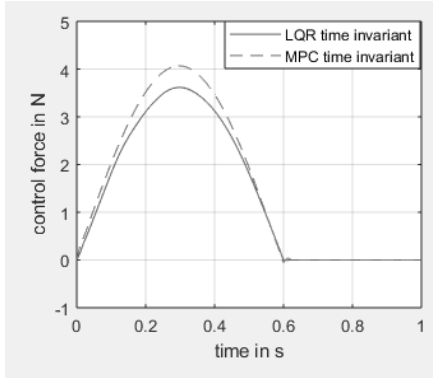


Figure 5: control input force of the MPC and LQR control

Figure 6 illustrates slightly better results for the time-invariant MPC control, whereas with a lower $\bar{\mathbf{R}}$ in Figure 7 the time-varying MPC has a lower deflection. The time-invariant MPC adds vibrations.

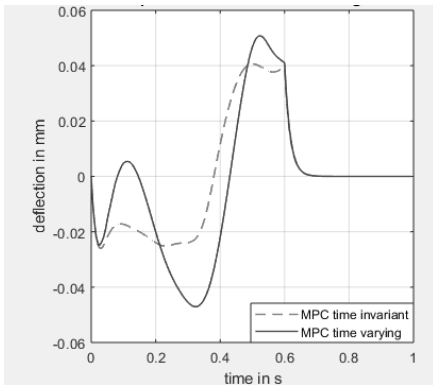


Figure 6: Time history of the mid-point deflection obtained the time invariant MPC (red dashed) and the time varying MPC (blue continuous); $m = 0.2 \text{ kg}$, $v = 1 \text{ m/s}^2$, $\bar{\mathbf{R}} = 1 \times 10^{-8} \mathbf{I}_{N_c \times N_c}$

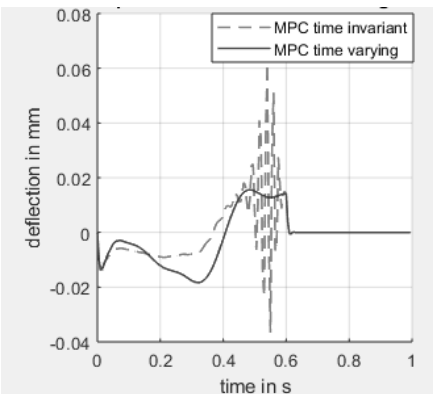


Figure 7: Time history of the mid-point deflection obtained the time invariant MPC (red dashed) and the time varying MPC (blue continuous); $m = 0.2 \text{ kg}$, $v = 1 \text{ m/s}^2$, $\bar{\mathbf{R}} = 1 \times 10^{-9} \mathbf{I}_{N_c \times N_c}$

Figure 8 compares the mid-point deflection of applying a moving mass $m = 0.2 \text{ kg}$ with a velocity $v = 2 \text{ m/s}^2$,

hereby the time-varying LQR control leads to a lower deflection of the structure.

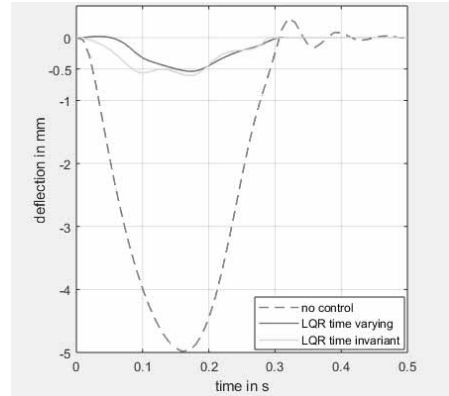


Figure 8: Time history of the mid point deflection, no control (red dotted), LQR time varying (blue) and time-invariant control (green), $v = 2 \text{ m/s}^2$, $m = 0.4 \text{ kg}$, $\mathbf{R} = 1 \times 10^{-6}$

Also the time-varying MPC shows beneficial results (Figure 9), again the time-invariant MPC adds vibrations. A increase of \mathbf{R} can reduce this added vibrations.

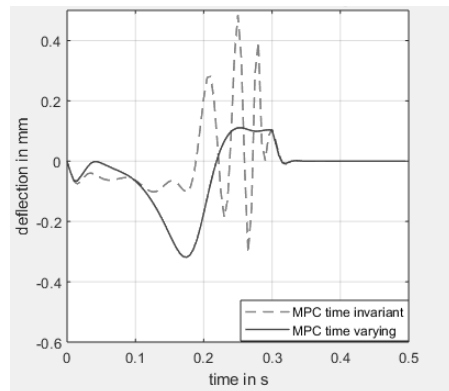


Figure 9: Time history of the mid point deflection of MPC time varying (blue) and time-invariant control (red dotted), $v = 2 \text{ m/s}^2$, $m = 0.4 \text{ kg}$, $\mathbf{R} = 1 \times 10^{-8}$

4 Conclusion and outlook

In this paper optimal control methods were studied on a time invariant moving mass structure. Hereby the receding horizon model predictive vibrational control was introduced to the moving mass problem. Compared to the classical LQR vibrational control this approach leads to a lower deflection of the structure. Especially at higher traveling speeds and with high mass the linear time-varying control approaches show better results. The system matrixes are assumed to be known, which is not always the case in reality. Further for the time invariant case the mass and the traveling speed of the moving load have to be known.

A future attempt is to include the change of the disturbance matrix in the future output variable \mathbf{Y} . To get to know the system matrixes of an unknown system real time system identification can be applied. For obtaining the states from a real system the use of an observer is thinkable. Modeling and simulation in MATLAB/Simulink is crucial throughout the development process.

References

- [1] Ouyang, H. Moving-load dynamic problems: A tutorial (with a brief overview)," *Mechanical Systems and Signal Processing*, vol. 25, pp. 2039-2060, 2011.
- [2] Stancioiu, D. Vibration of a continuous beam excited by a moving mass and experimental validation, *Journal of Physics: Conference Series*, vol. 182, 2009.
- [3] Stancioiu, D. and Ouyang, H. Experimental investigations of a multi-span flexible structure subjected to moving masses. *Journal of Sound and Vibration*, vol. 330, pp. 2004- 2016, 2010.
- [4] Balas, M. J. Active Control of Flexible Systems," *Journal of Optimization Theory and Applications*, vol. 25, 1978.
- [5] Preumont, A. *Vibration Control of Active Structures An Introduction*, Cham: Springer International Publishing AG, 2018.
- [6] Nikkhoo, A. Rofooei F. and Shadnam M., Dynamic behavior and modal control of beams under moving mass, *Journal of Sound and Vibration*, vol. 306, pp. 712-724, 207.
- [7] Pisarski D. and Bajer I., Semi- active control of 1D continuum vibrations under a travelling load, *Journal of Sound and Vibration*, vol. 349, pp. 140-149, 2010.
- [8] Stancioiu D. and Ouyang H., Optimal Vibration Control of Beams Subjected to a Mass Moving at Constant Speed, *Journal of Vibration and Control*, vol. 22, no. 14, pp. 3202- 3217, 2016.
- [9] Takács G. and Rohal'-Ilkiv B., Model predictive control algorithms for active vibration control: a study on timing, performance and implementation properties, *Journal of Vibration and Control*, vol. 20, no. 13, pp. 2061- 2080, 2014.
- [10] Takács G., Adaptive Model Predictive Vibration Control of a Cantilever Beam with Real- Time Parameter Estimation, *Shock and Vibration*, 2014.
- [11] Wang, L.: *Model Predictive Control System Design and Implementation Using MATLAB*, London: Springer- Verlag, 2010.

Virtual Coupling of Powertrain Components: New Applications in Testing

Thomas Gwosch¹, Michael Steck¹, Sven Matthiesen^{1*}

¹ IPEK – Institut für Produktentwicklung, Karlsruher Institut für Technologie (KIT), Kaiserstr. 10, 76131 Karlsruhe, Deutschland; **svn.matthiesen@kit.edu*

Abstract Testing of components in powertrain test benches is a challenge, especially if the available prototypes do not resist the required loads. One approach is the integration of the system under test via virtual coupling systems into the test bench. This paper presents a virtual coupling for connecting powertrain components with different load properties on a development test bench. The test bench integrates the remaining subsystems according to the X-in-the-Loop approach using suitable physical or virtual models. The virtual coupling is carried out by the control of the test bench motors, which are connected on both sides to the powertrain component to be tested. The control system fulfills the investigation-specific dynamic requirements for multi-closed-loop control. The verification of the virtual coupling is performed in comparison to the mechanical connection of the powertrain components.

Through the virtual coupling, the mechanically transferable power quantities can be adapted and thus opens up new application possibilities in testing.

Introduction

Test benches are used for testing variants of the product during development and for generating target values for design. The depth of the subsystem testing in prototype tests must be so extensive that sufficient information about the tested system is available and the test objective can be achieved [1]. Such test benches – also called development test benches – exist for the validation of subsystems in the automotive, aviation and power tool industries.

One challenge in the validation of powertrain components is the early availability of mechanically robust prototypes. Additively manufactured prototypes often do not achieve the desired properties with regard to mechanical load capacity in comparison to the original system.

An additional challenge is the reproduction of the relevant boundary conditions and interactions on the test bench. If as many interactions as possible are integrated,

the given uncertainties are reduced and allow a more realistic investigation in early development phases [2, p. 559]. To map these interactions, the powertrain component under test has to be integrated into the overall system [2, p. 559]. The X-in-the-Loop (abbr. XiL) approach supports this integration and enables the functional testing of a component or a subsystem (System-in-Development) in test benches by mapping the remaining subsystems through suitable physical or virtual models [2] (as Connected Systems). The subsystems of the test bench are linked via coupling systems [3] which connect the different powertrain components or the simulation models used. These coupling systems [3] have already been described in general. Virtual coupling of subsystems on the test bench is used in distributed validation for coupling several test benches (cf. [3], [4], [5], [6]).

These XiL test benches are already widely used in both the automotive and aircraft industries for the investigation of subsystems. With a fully physical coupling, direct integration of less robust prototypes into the overall system is difficult.

Challenge. There is a need for the coupling of drive components with different load capacities in a development test bench. The coupling has to enable the adaptation of the transmitted power, while the component is integrated into the overall system.

Research aim. This paper describes the virtual coupling for connecting powertrain components on a development test bench. The virtual coupling is verified with an example and allows investigations of subsystems with different load capacities.

1 Materials & Methods

1.1 Approach for Virtual Coupling

The basic idea is the decoupling of mechanical connected subsystems in a power train test bench and coupling them via virtual connections [7]. So it is possible to modify the mechanical quantities e.g. for adaption to prototypes by output scaling. It based on the X-in-the-Loop approach. The general structure of the coupling shows figure 1. Actuator 1 gets the flow quantity of actuator 2 as setpoint and actuator 2 gets the potential quantity of actuator 1 as setpoint. In general, the flow and potential quantity can be reversed.

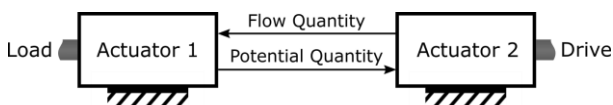


Figure 1: Virtual Coupling of mechanical subsystems

The test bench provides the simultaneous operation of mechanically detached powertrain components and their coupling in the virtual domain (multi closed-loop subsystem testing) by control of the test bench [7].

1.2 Integration of the Virtual Coupling in a Powertrain Test Bench

XiL powertrains provide the opportunity for virtual or physical subsystem integration. A XiL powertrain test bench provides a platform for subsystem testing with virtual coupling. So the approach of virtual coupling allows the extension of the possibility for the application of the powertrain test bench.

In this paper, a one-dimensional rotational powertrain with subsystems like a drive motor, a gear and a clutch is considered. One of the subsystems is the System-in-Development and the other subsystems are considered as Connected Systems. Additionally, the environment representation will be implemented with a load model and applications.

The interfaces of the physical subsystems upgraded with virtual-physical coupling systems. The coupling systems connect the subsystems in order to obtain the overall system. It is realized with sensor-actuator-systems and a central or decentral control system. The control system has to be real-time capable.

1.3 Verification

The verification is performed on the powertrain test bench (see fig. 3). For this purpose the virtual coupling between Actuator 1 and Actuator 2 is investigated.

In order to evaluate the influences of the virtual coupling, the time signals of the mechanical quantities in the interfaces of the coupling are compared. The applied synchronous motors have additional rotary encoders to measure the position. However, in order to guarantee a higher accuracy, the measuring elements are used for acquisition. For this purpose, additional measuring shafts are integrated into the drive train. These are no longer required for later investigations. The measuring shafts record the torque and rotational position in the interfaces. The input signals (Sensor 1) and the output signals (Sensor 2) are considered and compared. The test setup is shown in figure 2.

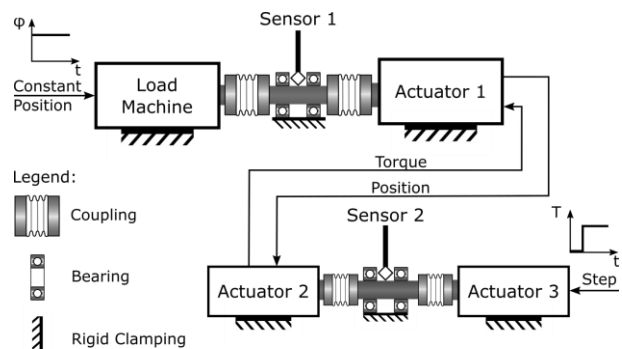


Figure 2: Verification: Test setup to evaluate the influence of virtual coupling

The load machine is operated in position control with constant position as setpoint. Actuator 3 operates as drive and is loaded with a step as desired value. The evaluation of the system response tests is done by means of parameters as used in the control engineering according to DIN IEC 60050 [8].

2 Results

First, the virtual coupling on the Scaled-Components-in-the-Loop test bench – a test bench for powertrain components – is presented. Subsequently, the results of the verification experiment are presented and the functional capability is derived from them.

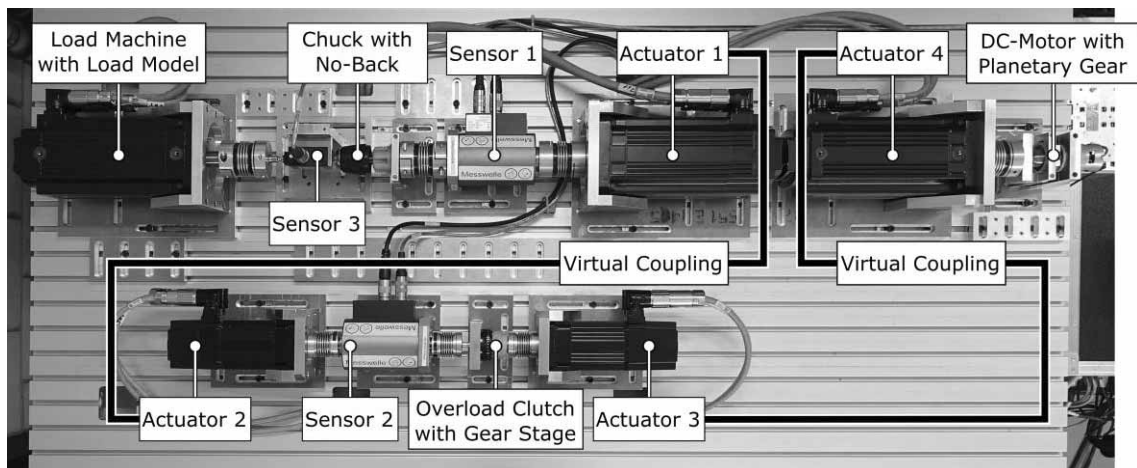


Figure 3: Scaled-Components-in-the-Loop test bench: Investigation of components of hand-held cordless screwdrivers in the entire powertrain (according to [9])

2.1 Scaled-Components-in-the-Loop Test Bench

The scaled-Components-in-the-Loop test bench, which is shown in figure 3 is a development test bench based on the IPEK-X-in-the-Loop approach [9]. With the test bench powertrain components of hand-held cordless screwdrivers can be investigated in the entire powertrain. The test bench is shown in figure 3. The development test bench contains the mechanical components of the drivetrain of a cordless screw driver. The components are a drive motor, a three-stage planetary gear, an overload clutch and a chuck with a no-back system.

The mechanical connections of the components are removed and sensor-actuator-systems are connected to the resulting interfaces as virtual connections. The sensor-actuator-systems connected via virtual coupling. Synchronous servo motors are used as actuators and DMS-based torque measurement shafts with optical incremental encoders used as sensors. The used Servo motors CMP71L (Load Machine, Actuator 1 and 4) and CMP50S (Actuator 2 and 3) were purchased from SEW-EURODRIVE. The measurement shafts (Sensor 4503A050) are from Kistler Instrumente.

Figure 3 shows on the top right the drive motor with the first and second planetary gear stage. The output shaft of the second gear stage is coupled to a test bench motor. The subsystem in top left of the figure shows the drill chuck with integrated no-back system, which is coupled

to an actuator at each of the two mechanical interfaces. On the output side (left), the drill chuck is loaded by a load model [9] with a load torque that simulates a screwing process. In the lower part of the figure, the overload clutch with the third planetary stage is coupled through the mechanical interfaces of the subsystem with two sensor-actuator-systems. For simultaneous operation of the subsystems, the actuators are coupled together in a control loop as part of the coupling system. [9]

2.2 Virtual Coupling and System Control

The coupling will be done via the connection of the actuators. The actuators are mounted on the sides of the subsystem interfaces. According to the approach, a bidirectional transmission of the signals takes place. Thus the flow quantity is transmitted in one direction – from the load to the drive. The potential quantity is transmitted in the opposite direction – from the drive to the load. The drive input is carried out via the control of the DC-Motor and the load model closes the control loop on the load side.

The bidirectional signal transmissions between the actuators is realised via a CAN bus and the controllers of the actuators. The minimal communication cycle period of the CAN bus is 1 ms. The controllers of the actuators are implemented directly on the frequency inverters due to dynamic performance of the control. Each frequency inverter has a cascaded PID-controller with feedforward control. The control parameters can be adjusted to the operating points of the system in development.

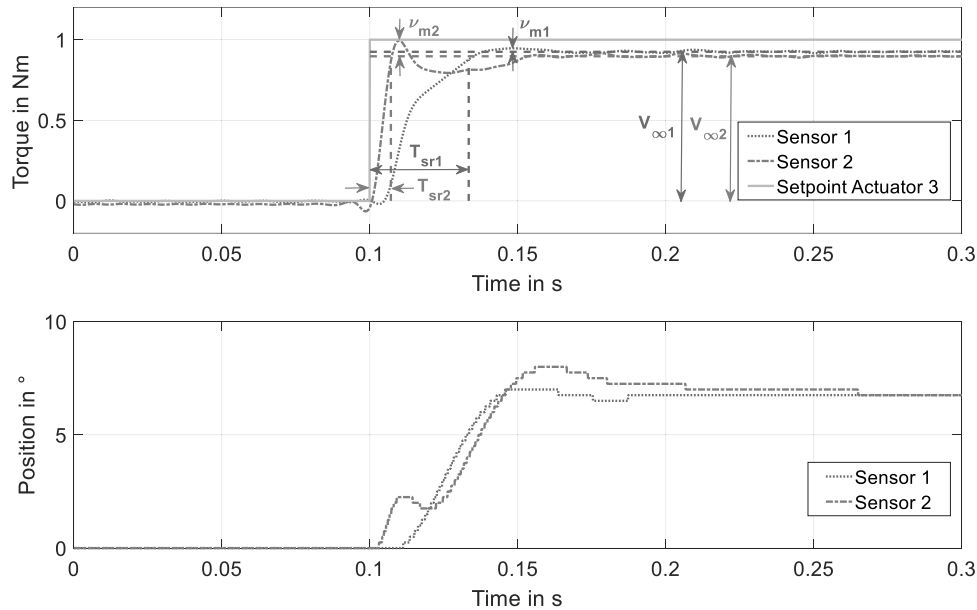


Figure 4: Results of the step response: Verification of the virtual coupling

2.3 Results of the Verification

The verification of the virtual coupling takes place in comparison to the ideal mechanical connection on the basis of the step response. An ideal mechanical connection would transmit the signals uninfluenced and thus pass on effects and interactions directly. The test setup and the measuring points used are shown in figure 2.

Figure 4 shows the results of the step response. In figure 4 (a) the setpoint sequence of the input torque (Actuator 3) shows the step with a step size of 1 Nm at the time of 0.1 s. Sensor 2 detects a steep torque rise with an overshoot of about 11%. After a short delay of a few milliseconds, the Sensor 1 detects a torque rise, which flattens out after about 60% of the final value. The step response time is 33 ms. The overshoot value is about 2%. Important characteristic values can be taken from table 1. Both sensors detects a similar increasing rotary position with small overshoot value (fig. 4 (b)). The steady-state value of the rotary position is with both sensors about 7 degree.

Measuring point	Step response time	Overshoot	Steady-state value
Sensor 1	33 ms	2.3%	0.92 Nm
Sensor 2	7 ms	10.9%	0.9 Nm

Table 1: Important characteristic values of the step response for verification of the virtual coupling

3 Discussion

The implementation of the approach leads partly to sophisticated physical powertrain test benches due to additional mechatronic systems and a closed-loop control of the interfaces for coupling function. On the other hand, there is the advantage of a simple reconfiguration of the test setup due to virtual interfaces.

The results of the system identification shows the characteristics of the coupling. The transfer characteristics consists of the signal processing time delay and the dynamic characteristics of the hardware.

The signal processing time delay is mainly caused by the communication time of the network participants. Instead of using the CAN bus the usage of an EtherCAT system can improve the performance.

The controllers and the additional mechanical components with their physical properties have an influence

on the transferable dynamics. One important aspect is the dynamic behavior of the actuators, since this makes up a large part of the rise time of the torque (fig. 4 (a)). Another important aspect is the inertia of the rotating parts of the coupling systems. This can be seen by increase of the rotational position (fig. 4 (b)). A compromise has to be found between the inertia of the system and the stability of the control.

The implementation of the approach used for powertrain test benches enables new opportunities for the investigation of systems in product development. Some of these are discussed in chapter 4.

Regarding the virtual coupling requirements, it is shown that the approach of virtual coupling can be used for a functional verification with powertrain test benches based on XiL.

4 Outlook

The virtual coupling allows new application possibilities in product development in the future. Through the virtual coupling of the development test bench, the transmitted power can be adapted to the load capacity of the prototypes.

The different application possibilities can be represented according to the X-in-the-Loop approach in an il-

lustration of the system architecture. The system architecture can be seen in figure 5.

In figure 5 the different physical hardware components of the powertrain and the necessary coupling systems are shown. The overload clutch and the other components are just connected through the virtual coupling. Within the virtual coupling, the transmitted signals can be adapted and changed in a wide range.

Scenarios. The following scenarios can be addressed with a virtual coupling:

- Virtual coupled operation offers possibilities to influence the power flow and to adapt the transmitted signals. For example, filters can be used or additional signals can be superimposed.
- The virtual coupling allows the integration of scaling models to adapt the transmitted power to components of a different size of the series.
- The virtual coupling enables the integration of prototypes that are produced additively and do not achieve the required robustness. Within the virtual coupling, the performance is adapted to the prototypes. Material models can be used to adapt differences in the material and its influence.
- The virtual coupling allows the integration of simulation models for hardware components that are not yet physically available.

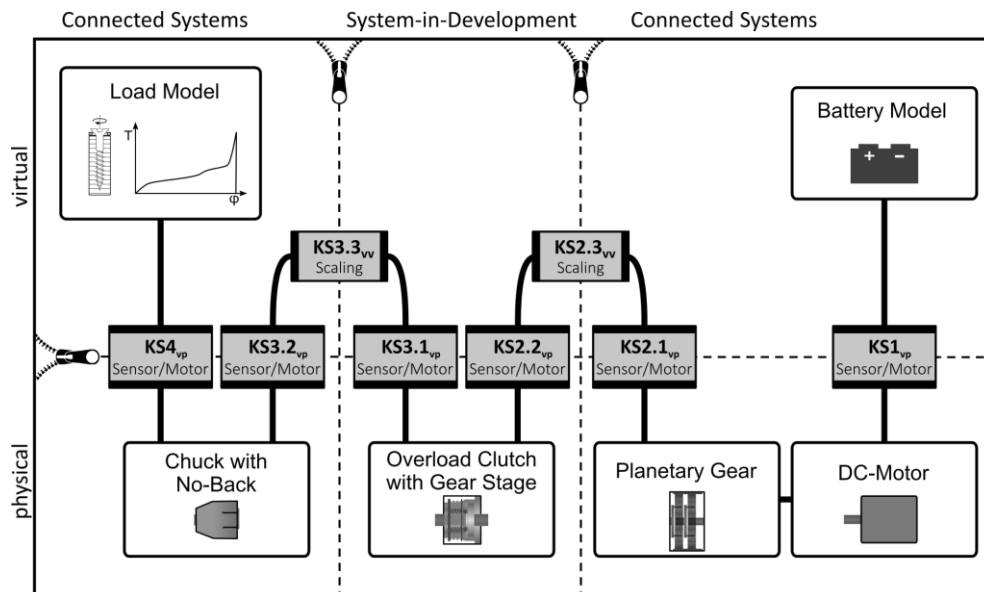


Figure 5: XiL-System-Architecture of the scaled-Components-in-the-Loop test bench (according to [3, p. 114] based on [10, p. 18])

Performance scaling. The performance scaling (see scenario 2) is described in the following. The virtual coupling allows the investigation-specific adaptation of the transmitted power by integrating scaling models within the bidirectional signal paths. [9] For the development of the scaling models, similarity indices [11] are established in a dimensional analysis [12] on the basis of the relevant system parameters. Examples for general similarities can be found in the literature [13].

If the derived similarity indices are the same in the scaled (prototype) and in the unscaled case (series part), a similar behavior is achieved on the test bench despite different properties. The findings from tests with a virtual coupling and a suitable power scaling can thus be transferred to the expected behavior of the coupling.

The knowledge gained on the test bench can be involved into the development process at an early stage. This shortens development time and supports Front-Loading in product development.

Acknowledgment. Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Wirtschaft und Energie unter dem Förderkennzeichen 20Y1509B gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

Gefördert durch:



Bundesministerium
für Wirtschaft
und Energie

aufgrund eines Beschlusses
des Deutschen Bundestages

5 References

- [1] S. Schork and E. Kirchner, "Defining Requirements in Prototyping: The Holistic Prototype and Process Development" in *Proceedings of NordDesign 2018*, 2018.
- [2] A. Albers, M. Behrendt, S. Klingler and K. Matros, "Verifikation und Validierung im Produktentstehungsprozess" in *Handbuch Produktentwicklung*, U. Lindemann, Ed., München: Hanser, 2016.
- [3] A. Albers, T. Pinner, S. Yan, R. Hettel and M. Behrendt, "Koppelsystems: Obligatory Elements within Validation Setups" in *Proceedings of DESIGN 2016*, 2016.
- [4] Y. You, „Eine Studie zur Implementierung des IPEK-X-in-the-Loop-Ansatzes in der verteilten Fahrzeugentwicklung am Beispiel Antriebsstrangentwicklung“ in *Forschungsberichte des IPEK – Institut für Produktentwicklung*, A. Albers, Ed., S. Matthiesen, Ed., Karlsruhe, Bd. 107, 2018.
- [5] D. Nickel, M. Behrendt, K. Bause and A. Albers, „Connected Testbeds – Early Validation in a Distributed Development Environment“ in *Stuttgarter Symposium für Produktentwicklung 2018*, 2018.
- [6] A. Albers, S. Yan, and M. Behrendt, "Validation Support for Distributed Vehicle Development Based on the XiL-Approach" in *FISITA 2016 World Automotive Congress*, 2016.
- [7] S. Matthiesen, T. Gwosch, S. Mangold, P. Grauberger, M. Steck and S. Cersowsky, „Frontloading in der Produktentwicklung von Power-Tools durch frühe Validierung mit Hilfe von leistungsskalierten Prototypen“, in *Stuttgarter Symposium für Produktentwicklung 2017*, 2017.
- [8] DIN IEC 60050-351:2014-09 „Internationales Elektrotechnisches Wörterbuch - Teil 351: Leittechnik“.
- [9] M. Steck, T. Gwosch and S. Matthiesen, "Frontloading in der Produktentwicklung handgehaltener Power-Tools – Lastmodelle für den Einsatz in Antriebsstrangprüfständen," in *Konstruktion*, 2019 (submitted).
- [10] M. Geier, C. Stier, T. Düser, M. Behrendt, S. Ott and A. Albers, "Simulationsgestützte Methoden IDE und XiL zur Entwicklung von Antriebsstrangkomponenten," in *ATZextra*, 2009.
- [11] W. Moog, „Ähnlichkeits- und Analogielehre“, Düsseldorf: VDI-Verlag, 1985.
- [12] J. Pawlowski, „Die Ähnlichkeitstheorie in der physikalisch-technischen Forschung“, Berlin, Heidelberg: Springer-Verlag, 1971.
- [13] J. Kuneš, "Dimensionless physical quantities in science and engineering", 1st ed. London: Elsevier, 2012.

Modellbildung und Systemidentifikation einer PEM-Brennstoffzelle

Sören Scherler^{1*}, Xiaobo Liu-Henke¹, Markus Henke²

¹Institut für Mechatronik (IMEC), Fakultät Maschinenbau, Ostfalia Hochschule für angewandte Wissenschaften, Salzdahlumer Str. 46/48, 38302 Wolfenbüttel, Deutschland; *so.scherler@ostfalia.de

²Institut für Elektrische Maschinen, Antriebe und Bahnen (IMAB), Fakultät für Elektrotechnik, Informationstechnik, Physik (EITP), Technische Universität Braunschweig, Hans-Sommer-Str. 66, 38106 Braunschweig

Abstract. Dieser Beitrag stellt die Modellbildung und Systemidentifikation der Polymer-Elektrolyt-Membran-Brennstoffzelle (PEM-Brennstoffzelle) Nexa[®]1200 (Fa. Heliocentris) dar. Dieses Modell ist notwendig für die Entwicklung intelligenter Algorithmen zur energieoptimierten Fahrt von Elektrofahrzeugen mit Brennstoffzellen als Range Extender. Nach einer Beschreibung der Modellbildungsmethodik und des eingesetzten HiL-Prüfstands werden elektrochemische Grundlagen zur Funktionsweise von Brennstoffzellen und Modellierungsansätze zur Beschreibung des Klemmenspannungsverhaltens dargestellt. Im Anschluss erfolgt eine Darstellung der Modellbildung, Systemidentifikation und Modellverifikation.

Einleitung

An der Ostfalia Hochschule wird das transdisziplinäre Verbundprojekt *Zukünftige Fahrzeugtechnologien im Open Region Lab (ZuFOR)*, gefördert vom Niedersächsischen MWK und der Volkswagenstiftung, durchgeführt.

Eine wesentliche Thematik des Teilprojekts *Intelligente Elektrofahrzeuge mit Range-Extender in Verkehrssystemen mit Fahrzeug 4.0 (iREX 4.0)* ist die modellbasierte Entwicklung eines flexibel skalierbaren, elektronischen Fahrzeugmanagements (eFZM) für Elektrofahrzeuge mit PEM-Brennstoffzelle als Range Extender. Das eFZM umfasst prädiktive Algorithmen zur autonomen, energieoptimierten Betriebs- und Zielführung, u.a. mithilfe erweiterter digitaler Karten und V2X-Kommunikation, sowie unterlagerte, für den Fahrbetrieb notwendige, Fahrzeugfunktionen wie Fahrdynamikregelung oder Energiemanagement.

In diesem Beitrag wird die Modellbildung und Systemidentifikation einer PEM-Brennstoffzelle dargestellt, womit eine notwendige Grundlage zur modellbasierten Entwicklung des eFZM geschaffen wird.

1 Methodik

Zur Durchführung einer zielgerichteten, effizienten Modellbildung werden der Modellbildungsprozess und ein flexibel konfigurierbarer Hardware-in-the-Loop-Prüfstand (HiL-Prüfstand) eingesetzt.

Der Modellbildungsprozess (Abbildung 1) beginnt mit einer Reduktion bzw. Vereinfachung des realen Systems gemäß der zu lösenden Problemstellung, aus der sich ein physikalisches Modell ergibt. Dieses wird mithilfe physikalischer Gesetzmäßigkeiten in ein mathematisches Modell überführt, welches, bspw. in Form von Signalfussplänen oder Blockschaltbildern, im Rechner abgebildet und mithilfe von CAE-Werkzeugen unter Einsatz numerischer Verfahren simuliert werden kann. Zudem umfasst der Modellbildungsprozess Messungen am realen System, um einerseits Struktur und Parameter des mathematischen Modells zu identifizieren und andererseits Modell und Simulationsergebnisse zu validieren.

Zur Durchführung dieser Messungen wird ein flexibel konfigurierbarer HiL-Prüfstand eingesetzt, welcher die energetisch relevanten Bordnetzkomponenten eines Elektrofahrzeugs mit Range Extender beinhaltet (Abbildung 2) [1]. Durch den modularen Aufbau können verschiedene Anwendungsfälle von der Systemidentifikation und Validierung von Simulationsmodellen bis zur Erprobung, Verifikation und Optimierung entwickelter Funktionen abgedeckt werden.

Der Prüfstand besteht aus einem Echthardwarezeitmodul (EZM), einem Softwaremodul (SWM) und mehreren Prüfstandsmodulen (PSM). Das EZM beinhaltet die Aufbereitung und Digitalisierung der Messsignale der PSM, die Berechnung der Regelalgorithmen auf einem digitalen Signalprozessor (DSP), die Umsetzung von Sollwerten für die PSM-Aktorik sowie die Kommunikation mit den PSM. Das SWM umfasst Modelle nicht real

vorhandener Systembestandteile, Regelalgorithmen und die Schnittstelle zwischen Nutzer und Prüfstand zur Zustandsvisualisierung und Prüfstandsbedienung. Die Modelle und Regler werden aus einem CAE-Werkzeug durch einen automatisierten Codegenerator mit allen Targetinformationen kompiliert und auf den DSP übertra-

gen. Bei den PSM handelt es sich um ein Brennstoffzellenmodul, ein Batteriemodul, ein Antriebsmodul sowie ein elektrisches Last- und Quellenmodul.

Für die Untersuchungen in diesem Beitrag wird eine Prüfstandskonfiguration mit Brennstoffzellen- sowie Last- und Quellenmodul verwendet.

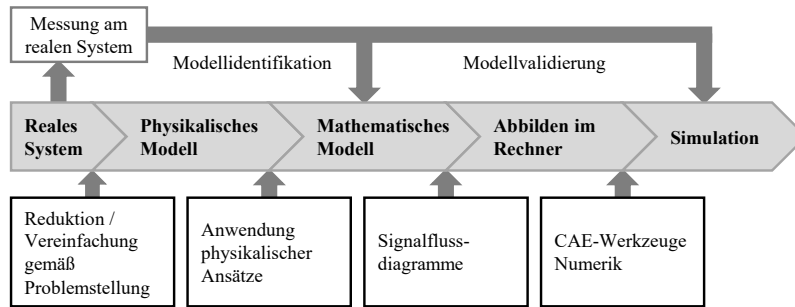


Abbildung 1: Modellbildungsprozess.

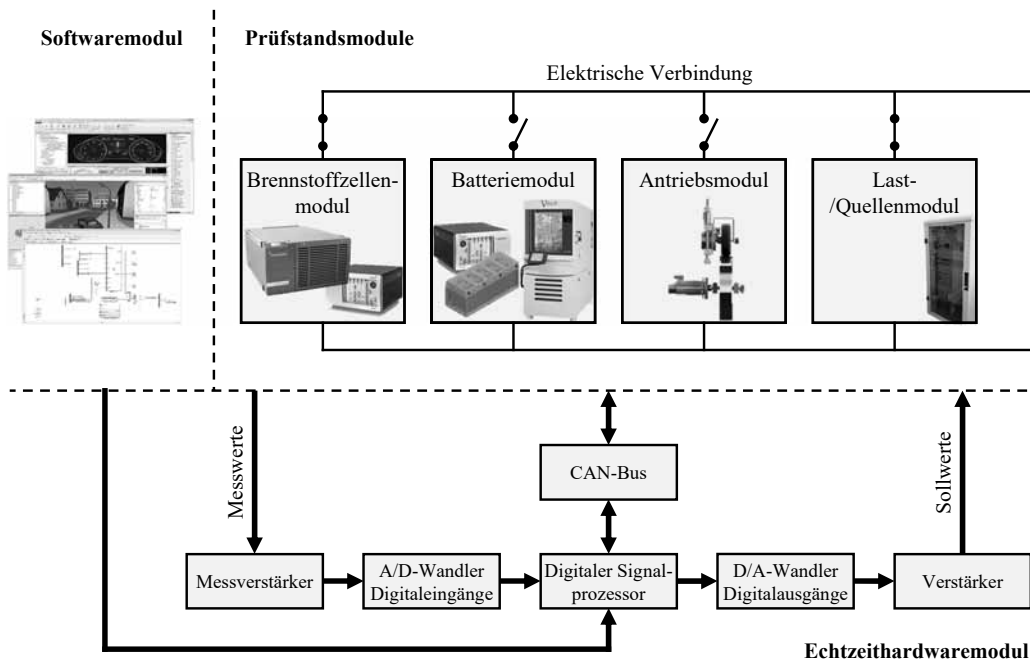


Abbildung 2: Modulare Struktur des flexibel konfigurierbaren HiL-Prüfstands.

2 Elektrochemische Grundlagen

In diesem Kapitel werden Aufbau, Funktionsprinzip und die Einflüsse auf die Klemmenspannung von PEM-Brennstoffzellen beschrieben.

In einer einzelnen Brennstoffzelle (Abbildung 3) wird eine Anode mit Wasserstoff und eine Kathode mit Sauerstoff umströmt. Die Wasserstoffmoleküle H_2 geben an der Anode zwei Elektronen e^- ab, welche von der Anode

zur Kathode fließen, sodass zwei positiv geladene Wasserstoffionen H^+ entstehen, welche wiederum durch die Protonenaustauschermembran zur Kathode fließen. An der Kathode nehmen die Sauerstoffmoleküle O_2 vier Elektronen auf und bilden doppelt negativ geladene Sauerstoffionen O^{2-} . Die durch die Membran zur Kathode gewanderten H^+ -Ionen bilden mit den O^{2-} -Ionen schließlich Wasser und komplettieren die ablaufende Redoxreaktion. [2]

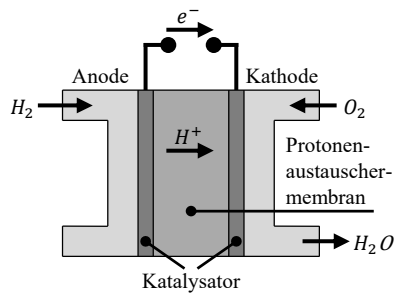


Abbildung 3: Schematische Darstellung einer PEM-Brennstoffzelle.

Die stationäre Klemmenspannung bzw. Polarisationskennlinie einer Brennstoffzelle ist in Abbildung 4 abhängig vom Laststrom dargestellt. Die Spannung der Brennstoffzelle verringert sich bei Belastung aufgrund unterschiedlicher Ladungstransporthemmungen. Es handelt sich um Spannungsabfälle durch

- Aktivierungsüberspannungen als Folge begrenzter Ladungsdurchtrittsgeschwindigkeiten zwischen Elektrode und Elektrolyt,
- Widerstandsüberspannungen, welche den linearen Spannungsabfall am Innenwiderstand abbilden,
- Diffusionsüberspannungen, welche durch eine bei hohen Stromstärken nicht mehr ausreichende Gaszu- und -abfuhr auftreten,
- und Reaktionsüberspannungen aufgrund endlicher chemischer Reaktionsgeschwindigkeiten.

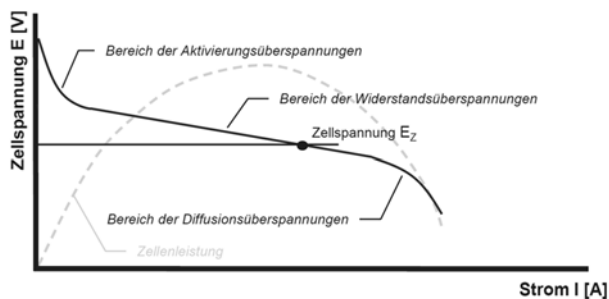


Abbildung 4: Strom-Spannungs-Kennlinie einer BZ, angelehnt an [2].

Für technische Anwendungen werden mehrere Brennstoffzellen in Serie zu Stacks verschaltet, um ein Vielfaches der Zellspannung von ca. 1 V zu erzielen. In den folgenden Ausführungen werden die Begriffe *Brennstoffzelle* und *Brennstoffzellenstack* synonym verwendet.

3 Modellierungsansätze

Zur Funktionsentwicklung sind Modelle der Klemmenspannung relevant, weshalb nachfolgend ein kurzer

Überblick über verschiedene Modellierungsansätze gegeben wird. Es existieren Ansätze zur empirischen oder physikalischen Modellierung des stationären Klemmenspannungsverhaltens und Erweiterungen zur Berücksichtigung des dynamischen Verhaltens.

Das stationäre Klemmenspannungsverhalten kann nach [3], [4] und [5] mithilfe künstlicher, neuronaler Netze beschrieben werden, wodurch allerdings der Bezug zur Physik der Brennstoffzelle verloren geht. Gleiches gilt für empirische Ansätze, welche dieses Verhalten, wie bspw. in [6] beschrieben, durch lineare oder quadratische Approximationsfunktionen annähern.

Ansätze, welche die Physik abstrahiert darstellen und zu stationären Klemmenspannungsmodellen mit relativ geringer Parameter- und Variablenzahl führen, basieren i.d.R. auf der Tafel-Gleichung und unterschiedlichen Ansätzen zur Beschreibung der verschiedenen Überspannungen. Geringfügig unterschiedliche Modellierungsbeispiele lassen sich in [7], [8] oder [9] finden.

Die stationären Modelle lassen sich durch Schaltelemente in elektrischen Ersatzschaltbildern (ESB) ergänzen, sodass sie auch das dynamische Klemmenspannungsverhalten abbilden. Hierzu werden ESB, basierend auf den auftretenden Transportvorgängen in der Brennstoffzelle, um lineare (z.B. Widerstände und Kondensatoren nach [10], [11] und [12]) oder nichtlineare Schaltelemente (z.B. Warburg-Impedanzen nach [13]) ergänzt.

3D-Modelle mit einer örtlichen Beschreibung der Brennstoffzelle oder einer Beschreibung aller Vorgänge auf Teilchenebene werden aus dieser Betrachtung ausgeschlossen, da sie aufgrund ihres sehr hohen Detaillierungsgrades im Hinblick auf den echtzeitfähigen Einsatz in Fahrzeugsteuergeräten ungeeignet sind.

4 Modellbildung

In diesem Kapitel wird der gewählte Modellierungsansatz für die Brennstoffzelle dargestellt. Es wird ein auf der Physik basierendes Modell des stationären Klemmenspannungsverhaltens mit einer Erweiterung um elektrische Bauelemente zur Approximation des dynamischen Verhaltens eingesetzt (vgl. Abbildung 5), da dieses verglichen zu 3D-Modellen einen guten Kompromiss zwischen Genauigkeit und Rechenaufwand bietet und im Gegensatz zu empirischen Ansätzen einen Bezug zur Physik aufweist.

Zur Modellierung der Polarisationskennlinie werden die Ruhespannung nach der Nernst-Gleichung und die in

Kapitel 2 beschriebenen Überspannungen berücksichtigt. Durch die Nernst-Gleichung

$$E = E^0 + \frac{RT}{2F} \ln \left(\frac{p_{H_2} \cdot p_{O_2}^{0.5}}{p_{H_2O}} \right) \quad (1)$$

wird die Potentialdifferenz E der Elektroden bzw. die Ruhespannung in Abhängigkeit des Standardelektrodenpotentials E^0 , der Brennstoffzellentemperatur T sowie der Drücke p_{H_2} , p_{O_2} und p_{H_2O} beschrieben. Die Drücke werden vom Brennstoffzellensystem konstant geregelt und werden somit im Rahmen dieser Arbeit als konstant betrachtet. Als Konstanten fließen die allgemeine Gaskonstante R und die Faraday-Konstante F in die Gleichung ein. Die Klemmenspannung der Brennstoffzelle

$$u = E - u_{akt} - u_{ohm} - u_{konz} \quad (2)$$

ergibt sich aus der Ruhespannung abzüglich der Aktivierungsüberspannung u_{akt} , dem ohmschen Spannungsabfall u_{ohm} und der Konzentrationsüberspannung u_{konz} . Die Aktivierungsüberspannung lässt sich mithilfe der Tafel-Gleichung

$$u_{akt} = \frac{RT}{2\alpha F} \ln \left(\frac{i}{i_0} \right) \quad (3)$$

beschreiben, welche durch den Ladungstransferkoeffizienten α , den Laststrom i und die Konstante i_0 beschrieben wird. Bei den ohmschen Spannungsverlusten

$$u_{ohm} = R_{ohm} i \quad (4)$$

handelt es sich um Verluste am Innenwiderstand. Die Konzentrationsüberspannung

$$u_{konz} = -\frac{RT}{2F} \ln \left(1 - \frac{i}{i_l} \right) \quad (5)$$

hat erst bei hohen Lastströmen nahe des begrenzenden Stroms i_l einen signifikanten Einfluss auf die Klemmenspannung und bildet sinkende Konzentrationen der Arbeitsgase sowie zeitlich begrenzte Transportvorgänge innerhalb der Brennstoffzelle ab.

Das elektrische Ersatzschaltbild in Abbildung 5 zeigt die Erweiterung des stationären Modells um den Kondensator C_{DK} , welcher die Doppelschichtkapazität der Brennstoffzelle repräsentiert.

Das erweiterte Modell ist ein System erster Ordnung

$$u_{akt} + R_{akt} C_{DK} \dot{u}_{akt} = R_{akt} i \quad (6)$$

mit der Verstärkung R_{akt} und der Zeitkonstante $\tau = R_{akt} C_{DK}$ zur Abbildung der Dynamik.

Im nachfolgenden Kapitel werden die Parameteridentifikation und die Modellverifikation beschrieben.

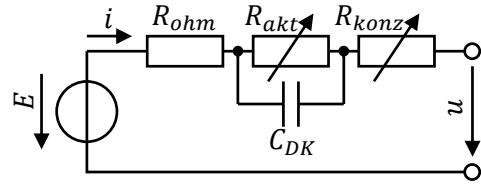


Abbildung 5: El. Ersatzschaltbild der Brennstoffzelle.

5 Systemidentifikation und Modellverifikation

Das Vorgehen zur Systemidentifikation und Modellverifikation ist in Abbildung 6 dargestellt. Ausgehend von A-priori-Kenntnissen des Systems erfolgt die theoretische Modellbildung (vgl. Kapitel 4) und die Planung von Messungen, welche zudem die Identifikationsaufgabe und Anforderungen berücksichtigt. Nach Durchführung der Messung werden durch einen Optimierungsalgorithmus optimale Parameter des theoretischen Modells zur Minimierung der Abweichung zwischen Modell und Messung ermittelt. Anschließend erfolgt die Modellverifikation, welche entweder eine iterative Verbesserung des theoretischen Modells und eine erneute Optimierung erfordert oder den Identifikationsprozess erfolgreich abschließt, sodass ein endgültiges Modell vorliegt.

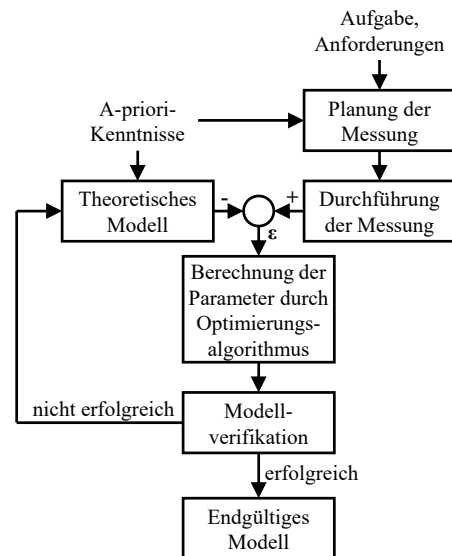


Abbildung 6: Ablauf der Systemidentifikation.

Es werden zunächst Messungen zur Struktur- und Parameteridentifikation im Frequenzbereich und anschließend zur Identifikation des nichtlinearen, stationären Verhaltens im Zeitbereich durchgeführt.

Als Messverfahren im Frequenzbereich wird die galvanostatische, elektrochemische Impedanzspektroskopie

(EIS) angewandt. Im Rahmen der EIS werden Frequency Sweeps von 0,01 Hz bis 100 kHz und einer messgerätebedingt maximalen Amplitude von ± 20 mA als Anregungssignale eingesetzt. Zusätzlich wird diesem Signal ein konstanter Gleichstromanteil zur Einstellung verschiedener Arbeitspunkte überlagert.

Aus initial parametrisierten, theoretischen Modellen (vgl. Kap. 4), welche um Arbeitspunkte (AP) linearisiert werden, und den gemessenen Frequenzgängen werden die Modellparameter nach der vorgestellten Identifikationsmethodik optimiert. Eine exemplarische Gegenüberstellung gemessener und simulierter Frequenzgänge ist in Abbildung 7 für ausgewählte Arbeitspunkte dargestellt.

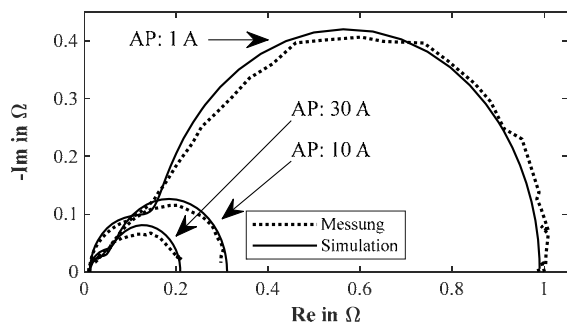


Abbildung 7: Exemplarische Ergebnisse der Identifikation im Frequenzbereich für verschiedene Arbeitspunkte.

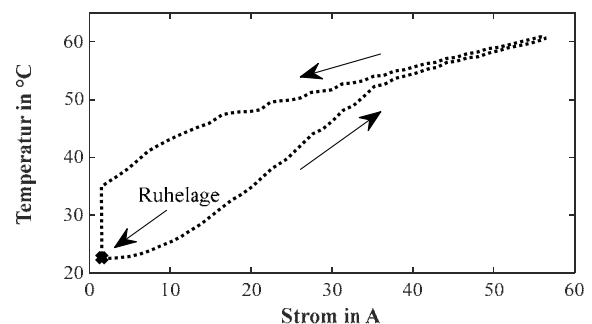
Es zeigt sich eine Abhängigkeit der gemessenen Impedanzen vom Arbeitspunkt. Je größer der Gleichstrom ist, desto kleiner werden die Impedanzen. Des Weiteren zeigt sich für alle betrachteten Arbeitspunkte eine gute Übereinstimmung zwischen Messung und Simulation, sodass sowohl die Modellstruktur als auch die identifizierten Parameter als hinreichend detailliert und verifiziert betrachtet werden können.

Durch Messungen im Zeitbereich wird das nichtlineare Verhalten der stationären Polarisationskennlinie bestimmt. Hierzu werden ausgehend von der Ruhelage linear stetig steigende Lastströme auf die Brennstoffzelle eingepreßt, welche bei Erreichen eines Maximalwerts linear bis zum Erreichen der Ruhelage wieder reduziert werden. Die Messergebnisse werden zur Parametrierung des stationären Modells nach Gleichung 2 genutzt.

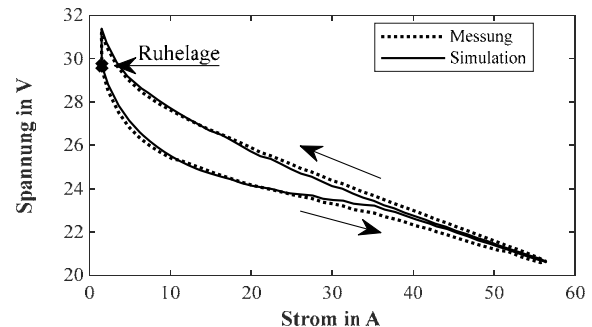
Abbildung 8 zeigt die Brennstoffzellentemperatur während der Messung (Abbildung 8a) sowie eine Gegenüberstellung von gemessener und simulierter Polarisationskennlinie (Abbildung 8b). Ausgehend von der Ruhelage zeigt sich, dass die Temperatur mit steigender Belastung und Dauer ansteigt und mit sinkender Belastung wieder abfällt. Der Temperaturabfall läuft langsamer als

der Anstieg ab, sodass die Brennstoffzelle bei Erreichen des Ruhestroms (u.a. Lüfter und Steuerung der Brennstoffzelle) weiter abkühlt und erst dadurch die Ruhelage wieder erreicht. Dieses Temperaturverhalten spiegelt sich in der Polarisationskennlinie wieder, welche für höhere Temperaturen auch höhere Spannungen annimmt. Die Spannung erreicht den Ausgangszustand, nachdem die Brennstoffzelle im Anschluss an die Belastung wieder ihre Ausgangstemperatur erreicht hat.

Die Gegenüberstellung der gemessenen und simulierten Polarisationskennlinie zeigt, dass durch die gewählte Modellstruktur und Modellierungstiefe die Realität mit einer hohen Genauigkeit abgebildet werden kann.



(a): Gemessene Stacktemperatur.

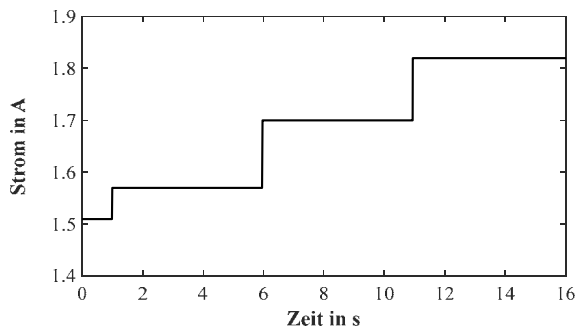


(b): Gegenüberstellung von gemessener und simulierter Polarisationskennlinie.

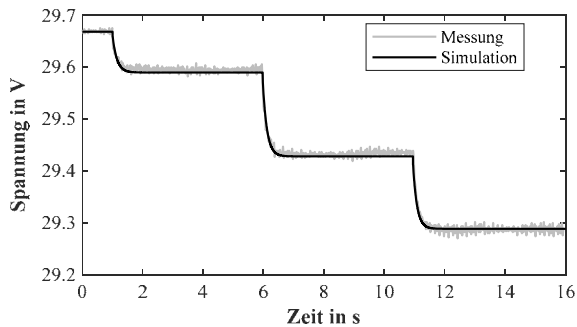
Abbildung 8: Exemplarische Ergebnisse der Identifikation im Zeitbereich.

Zur weitergehenden Verifikation werden zusätzliche Messungen im Zeitbereich sowie das nichtlineare Gesamtsimulationsmodell herangezogen.

In Abbildung 9 sind drei aufeinanderfolgende Sprunganregungen (Abbildung 9a) und -antworten (Abbildung 9b) als exemplarische Betriebsfälle dargestellt. Die simulierte Spannung entspricht der gemessenen Spannung ohne nennenswerte Abweichungen sowohl im dynamischen als auch im stationären Verhalten. Das identifizierte Systemmodell spiegelt somit das gemessene Systemverhalten ausreichend genau wider.



(a): Stromanregung zur Verifikation.



(a): Gegenüberstellung der gemessenen und simulierten Sprungantworten der Klemmenspannung.

Abbildung 9: Exemplarisches Ergebnis der Modellverifikation im Zeitbereich.

6 Resümee und Ausblick

In diesem Beitrag wurden die Modellbildung und Systemidentifikation einer PEM-Brennstoffzelle dargestellt, um eine Grundlage zur modellbasierten Entwicklung intelligenter Funktionen für Elektrofahrzeuge mit Range Extender zu bilden. Das Modell beschreibt sowohl das stationäre als auch das dynamische Verhalten der Brennstoffzelle in den untersuchten Betriebspunkten mit einer hinreichend hohen Genauigkeit.

Im folgenden Projektverlauf gilt es, die Brennstoffzelle weitergehend zu untersuchen und das Modell weiter zu optimieren, um z.B. auch das thermische Verhalten im Modell abzubilden. Des Weiteren kann mithilfe des Modells die Funktionsentwicklung durchgeführt werden, sodass neue Erkenntnisse und Beiträge im Themenfeld der energieoptimierten Fahrt erwartet werden.

Danksagung

Dieser Beitrag wurde im Rahmen des interdisziplinären Verbundprojekts *Zukünftige Fahrzeugtechnologien im Open Region Lab* durch das Niedersächsische MWK und die Volkswagenstiftung unter dem Förderkennzeichen

VWZN3236 gefördert. Die Verantwortung für den Inhalt liegt bei den Autoren.



Niedersächsisches Ministerium
für Wissenschaft und Kultur



VolkswagenStiftung

Referenzen

- [1] Scherler S, Liu-Henke X. *Model-based design of a multi-functional HiL test bench for investigations on a Range Extended Vehicle*, 4th International Symposium on Systems Engineering, Rom, Italien, 2018.
- [2] Eichlseder H, Klell M. *Wasserstoff in der Fahrzeugtechnik*. 3. Auflage, Springer Vieweg, 2012.
- [3] Karthik M, Gomathi K. *Design, modeling and simulation of a cascaded optimal neural network based fuel cell Powered Electric Vehicle*. 2nd International Conference on Electrical Energy Systems, Chennai, Indien, 2014.
- [4] Puranik S, Keyhani A, Khorrami, F. State-Space Modeling of Proton Exchange Membrane Fuel Cell. *IEEE Transactions on Energy Conversion*. 2010; 25(3): pp. 804-813.
- [5] Khorrami F, Puranik S, Keyhani A, Krishnamurthy P, She Y. *PEM fuel cell distributed generation system: Modeling and robust nonlinear control*. 48th IEEE Conference on Decision and Control, Shanghai, China, 2009.
- [6] Siemer J. *Lokale Entropieproduktionsraten in der Polymerelektrolyt-Membran-Brennstoffzelle* [Dissertation]. Helmut-Schmidt-Universität Hamburg, 2007.
- [7] Pasricha S, Keppler M, Shaw S, Nehrir M. Comparison and Identification of Static Electrical Terminal Fuel Cell Models. *IEEE Transactions on Energy Conversion*. 2007; 22(3): pp. 746-754.
- [8] Boscaino V, Capponi G, Miceli R, Rizzo Galluzzo G, Rizzo R. Comparison of models of fuel cells based on experimental data for the design of power electronic systems. *IET Renewable Power Generation*. 2015; 9(9): pp. 660-668.
- [9] Larminie J, Dicks A. *Fuel Cell Systems Explained*. 2nd Edition, Wiley, England, 2003.
- [10] Kurzweil P. *Brennstoffzellentechnik: Grundlagen, Komponenten, Systeme, Anwendungen*. 2. Auflage, Springer Vieweg, 2013.
- [11] Haubrock J. *Parametrierung elektrischer Äquivalenzschaltbilder von PEM Brennstoffzellen* [Dissertation]. Otto-von-Guericke-Universität Magdeburg, 2007.
- [12] Wagner N. *Electrochemical Impedance Spectroscopy*. In: Wang H, Yuan X, Li H, *PEM Fuel Cell Diagnostic Tools*. CRC Press, 2011.
- [13] Friede W, Raël S, Davat B. Mathematical model and characterization of the transient behavior of a PEM fuel cell. *IEEE Transactions on Power Electronics*. 2004; 19(5): pp. 1234-1241.

Simulation und Modellbildung einer integrierten Positioniereinrichtung für induktive Fahrzeug-Batterieladesysteme

Lyucheng Zhu¹, Oleg Schäfer^{1*}, Markus Henke¹, Jürgen Meins¹

¹Institut für Elektrische Maschinen, Antriebe und Bahnen, Technische Universität Braunschweig, Hans-Sommer-Str. 66, 38106 Braunschweig; *o.schaefer@tu-braunschweig.de

Abstract. Induktive Energieübertragungssysteme bewirken eine kontaktlose Energieübertragung über die magnetische Kopplung der Spulen. Diese hat einen entscheidenden Einfluss auf den Wirkungsgrad und hängt maßgeblich von den Geometrieigenschaften der Spulen sowie der relativen Position der Spulen zueinander ab. Induktive Energieübertragung zum Laden elektrischer Fahrzeuge ist durch einen großen Luftspalt und einer hohen Ungenauigkeit bei manueller Positionierung gekennzeichnet. Technische Hilfseinrichtungen zur automatischen und exakten Positionierung des Fahrzeugs und damit der Empfänger-spule führen zu höheren Wirkungsgraden bei der Energieübertragung und letztendlich zu höherer Kundenzufriedenheit. Am IMAB wurde ein Konzept für eine Ausrichtungsmethode entwickelt, das auf einer indirekten Magnetfeldmessung basiert. Über die induzierten Spannungen von auf Fahrzeugseite angebrachten Messspulen wird infolge des einwirkenden Hauptflusses eine relative Positionsabweichung ermittelt. Das Konzept wird über eine FEM mithilfe der Simulationsumgebung ANSYS auf seine generelle Eignung untersucht. Darin werden die notwendigen elektromagnetischen Betrachtungen für die Spulenpaarungen durchgeführt um daraus Variablen zur relativen Positionserfassung als Repräsentative für die korrespondierenden Spulenpaarungen definieren und verifizieren zu können.

Einleitung

Kontaktlose Energieübertragung weist aufgrund der zueinander elektrisch isolierten Komponenten zwischen Fahrzeug und Ladeeinheit gegenüber der konduktiven Energieübertragung Vorteile im Hinblick auf Sicherheit und Komfort, speziell bei rauen Umweltbedingungen, auf. Daher wird die kontaktlose Energieübertragung richtungsweisend als zukünftige Anwendung für Ladesys-

teme zur Verbesserung der Wettbewerbsfähigkeit elektrischer Fahrzeuge angesehen.

Verglichen mit anderen Anwendungen induktiver Energieübertragung, ist beim Laden elektrischer Fahrzeuge der von der Bodenfreiheitsklasse des Fahrzeugs abhängige Luftspalt größer. Zudem können starke horizontale Fehlausrichtungen infolge großer Abstände des Fahrzeugs gegenüber der Sendeeinheit entstehen. Diese senken den Systemwirkungsgrad und müssen folglich manuell oder mithilfe integrierter technischer Positioniereinrichtungen reduziert werden. Am IMAB wurde ein Konzept für eine Positioniereinrichtung entwickelt, welches das magnetische Feld der an der Ladestation befindlichen bodenseitigen Primärspule nutzt. Hierbei soll die Methode sowohl eine Kopplung bei Paarung gleicher Spulentopologien als auch ungleicher Topologien ermöglichen, sodass eine Interoperabilität beim Laden gewährleistet wird. Die zur Entwicklung des Konzepts notwendigen elektromagnetischen Untersuchungen zur Simulation und Vorausberechnung des Verfahrens werden mit dem 3D Modeler in ANSYS Electronics Desktop durchgeführt.

1 Simulation der magnetischen Felder

Prinzipiell besteht ein induktives Ladesystem für elektrische Fahrzeuge aus einer Leistungssende- (T_x) und einer Leistungsempfangseinheit (R_x), welche physisch getrennt und magnetisch über die Spulen gekoppelt sind. Die eingangsseitige Netzleistung wird nach einer DC-Wandlung in eine hochfrequente AC Leistung transformiert. Die Leistung wird im Luftspalt kontaktlos nach dem Faraday'schen Gesetz der elektromagnetischen Induktion mithilfe einer bodenseitigen Primär- und einer fahrzeugseitigen Sekundärspule übertragen [1]. Die Effizienz der Übertragung ist maßgeblich bestimmt über das

Kompensationsnetzwerk sowie über die Induktivitäten, welche von den Spuleneigenschaften abhängen. Der darin enthaltene magnetische Widerstand R_m ist unter anderem eine Funktion der Permeabilität und steigt infolge von z.B. Sättigung. Die Gegeninduktivität M verhält sich reziprok zum geometrischen Abstand der Spulen sowie reziprok zum magnetischen Widerstand [2].

$$M = k * \sqrt{L_p L_s} \tag{1}$$

$$= \frac{r_p^2 r_s^2}{\sqrt{r_p r_s} \sqrt{(r_p^2 + x^2)^3}} * \sqrt{\frac{N_p^2}{R_{m,p}} \frac{N_s^2}{R_{m,s}}}$$

r_p, r_s : Radius der Primär- und Sekundärspule
 x : Abstand zwischen den Spulenmitten

Die Eigenschaften der Spule sind vorgegeben aus dem Design des Herstellers und können in ANSYS entsprechend der Maße und Materialeigenschaften zu einem 3D Modell aufgebaut werden. Abbildung 1 klassifiziert die gängigen Spulentopologien. Hierzu zählen die Solenoid Spulen (flux-pipe pad) aus der Gruppe der doppel-seitigen Koppler und der zirkularen, rechteckigen und doppel-D Spulen, die zur Gruppe der einseitigen Koppler gehören [3].

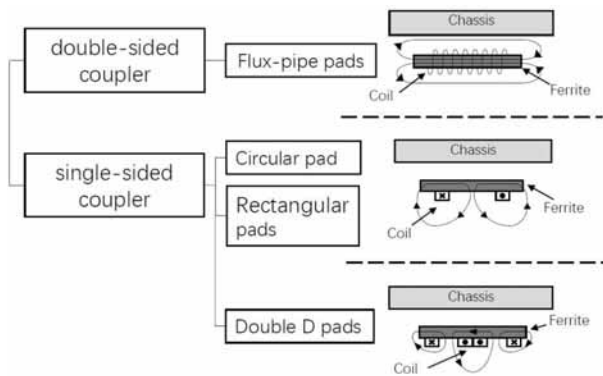


Abbildung 1: Klassifizierung der Spulentopologien

Die Spulen lassen sich mit unterschiedlichem Aufwand und Detaillierungsgrad zu 3D Modellen aufbauen. Sofern die Windungen der Wicklungen berücksichtigt werden sollen, steigt der Aufwand, die Simulationsdauer sowie die Genauigkeit der Ergebnisse. Die Modelle lassen sich direkt im 3D Modeler aber auch in einem CAD-Programm wie z.B. Solid Works mit anschließendem Import nach ANSYS realisieren. Die in Abbildung 2 dargestellte zirkuläre Spule wird mit einer Windung realisiert. Im zweiten Schritt lassen sich unterschiedliche Paarungen hinsichtlich des Übertragungsverhaltens untersuchen.

Anhand der Simulationsergebnisse können Rückschlüsse auf die notwendigen Positionen der Messspulen eines Übertragungssystems abgeleitet werden.



Abbildung 2: B_z -Feldverteilung in der zirkulären Spule

Das hier vorgestellte Konzept zur Spulenpositionierung beruht auf dem Faraday'schen Gesetz der Induktion. Hierbei ist lediglich die Z-Komponente (vertikale Komponente) des magnetischen Feldes für die Induktionswirkung relevant. In Abbildung 2 wird daher die magnetische Flussdichte B_z eingeblendet.

Bevor das Konzept entwickelt werden kann, müssen die Feldverteilungen der unterschiedlichen Spulentopologien, die prinzipiell als Primärspule in Betracht kommen, der in Abbildung 1 dargestellten Gruppen untersucht werden. Die Software ANSYS bietet hierbei die Möglichkeit sämtliche geometrischen Größen als Variablen zu definieren, sodass unter anderem die Parameter Luftspalt (Z-Achse) und die horizontalen Verschiebungen (X- und Y-Achse) während der Simulation anhand eines „Parametric Setup“ variiert werden. Hieraus ergeben sich die in Abbildung 3 exemplarisch dargestellten Feldverteilungen bei optimaler Ausrichtung sowie bei steigender Fehlausrichtung.

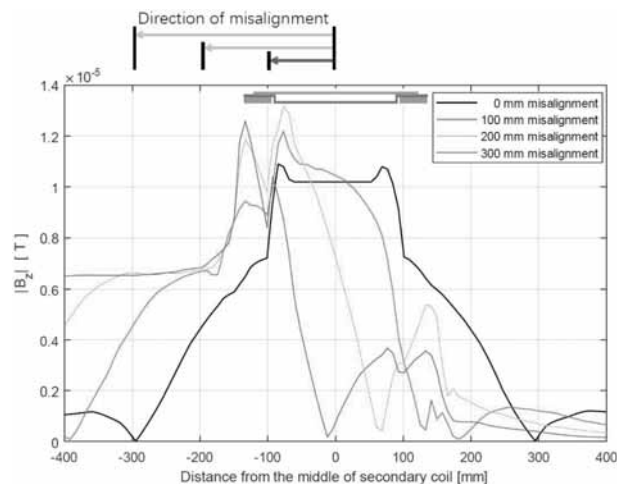


Abbildung 3: Flussdichteverteilung B_z bei unterschiedlichen Ausrichtungen auf horizontaler Ebene

Die Primärspule wird schrittweise um 100 mm von der Sekundärspule auf horizontaler Ebene entfernt und die magnetische Flussdichte B_z im Betrag als Funktion dessen aufgetragen. Es zeigt sich, dass neben der Verschiebung des Maximums der Flussdichte zusätzlich eine durch die Ferrite bedingte Verzerrung des Feldes auftritt.

Mit der Kenntnis der Feldverteilung können die Geometrieanforderungen sowie eine erste Aussage zur Anzahl und Position der Messspulen abgeleitet werden, die letztendlich der späteren Positionierung auch bei interoperabler Übertragung dienen werden.

2 Simulation der Messspulenanordnungen

In [4] wird eine Methode zur Positionierung der Spulen auf Basis einer Magnetfeldmessung über die induzierte Spannung vorgestellt. Diese Methode benötigt vorab definierte Look-up-Tabellen mit Informationen über die Spuleneigenschaften wie z.B. Geometrien. Das hier betrachtete Konzept soll dagegen eine relative Positionsabweichung ohne vorab definierter Spuleneigenschaften liefern. Die Ermittlung des Messbereichs und der Genauigkeit der Positionierung kann über eine FEM mithilfe einer Simulation erfolgen. Die notwendigen elektromagnetischen Untersuchungen zur Simulation und Vorausberechnung des Verfahrens werden mithilfe von ANSYS durchgeführt.

2.1 Prinzip des Positionierungskonzepts

Die Ausrichtung der Sekundärspule zur magnetisch optimalen Kopplung erfolgt durch Messung des von der Primärspule erzeugten Magnetfeldes mithilfe auf der Sekundärspule montierter Messspulen. Dieses Feld ist als Φ_m mit den Messspulen gekoppelt. Über die Beträge der induzierten Spannungen U_m in den Messspulen lassen sich Variablen definieren, die eine relative Positionsabweichung detektieren können.

$$U_m = N_m \frac{d\Phi_m}{dt} \quad (2)$$

Es bedarf hierbei zweier Variablen zur genauen Detektion einer Positionsabweichung. Abbildung 4 enthält die Variable A_p während der horizontalen Bewegung, die zur Detektion des relativen Positionsfehlers dient und sich aus den induzierten Spannungen von exemplarisch drei Messspulen ergibt. Sofern die Sekundärspule sich in optimaler Ausrichtung

zur Primärspule befindet, bei zirkularer Spulenanordnung entspricht dies den Mittelpunkten der Spulen, nähert sich der Betrag der Variablen Null an.

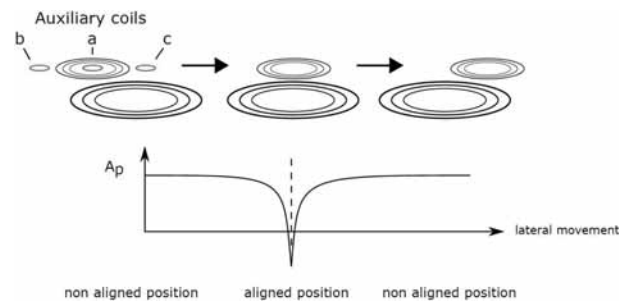


Abbildung 4: Erwarteter Verlauf von A_p während der lateralen Bewegung

Zur Detektion der Richtungsabweichung wird eine zweite Variable D_p eingeführt.

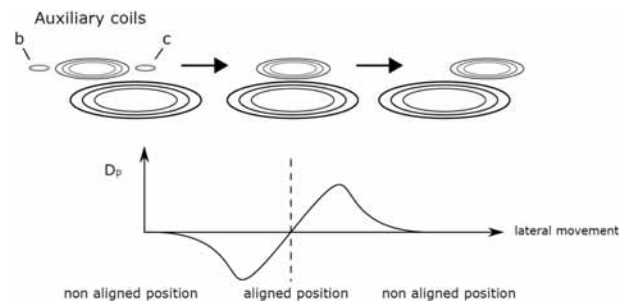


Abbildung 5: Erwarteter Verlauf von D_p während der lateralen Bewegung

Abbildung 5 verdeutlicht den Verlauf der Variablen D_p während der lateralen Bewegung auf horizontaler Ebene. Die Auswertung zweier Messspulen liefert einen Vorzeichenwechsel zur Erkennung der entsprechenden Abweichung.

2.2 Modellierung und Simulation

Mithilfe einer weiteren Simulation in ANSYS können Schlussfolgerungen auf die notwendige Anzahl sowie die Positionen der Messspulen gezogen werden. Hierbei bietet die Software die Möglichkeit diverse Punkte im Modell festzulegen, welche im Setup definiert werden können und im Simulationsergebnis aufgeführt werden. Diese Punkte stellen in diesem Fall die Orte für die platzierten Messspulen dar und sollen die Z-Komponenten des Hauptfeldes messen. Diese Methode wird an allen betrachteten Spulenanordnungen durchgeführt. Dies beinhaltet Paarungen gleicher Spulentopologien (z.B. zirkular-zirkular) sowie ungleicher Topologien (z.B. zirkular-

doppel D).

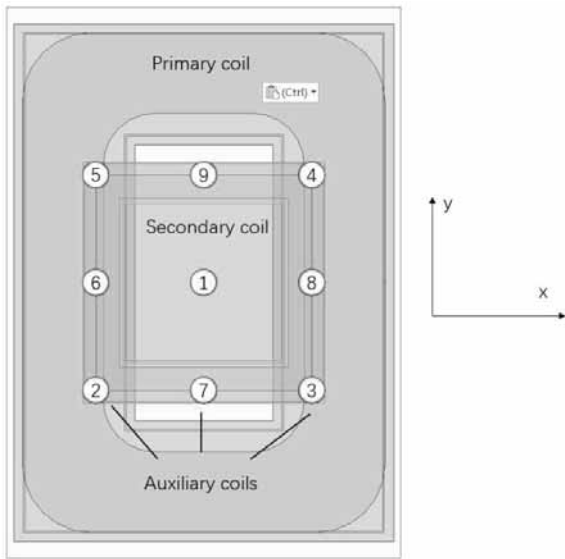


Abbildung 6: Mögliche Anzahl und Platzierungen der Messspulen

Mit dem „Fields Calculator“ gemäß Abbildung 7 lassen sich dann die induzierten Spannungen an den definierten Positionen im Bereich der Sekundärspule ermitteln, ohne dass die Messspulen modelliert werden müssen.

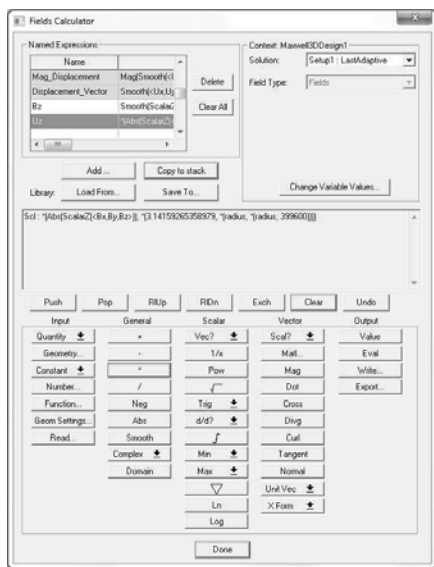


Abbildung 7: Berechnung der induzierten Spannungen über die Flussdichten im Bereich der Messspulenmitte

Die Ergebnisse Abbildung 8 veranschaulicht den simulierten Verlauf der zuvor definierten Variable A_p . Hierbei

wurde zusätzlich der Unterschied zwischen den möglichen Positionen der Messspulen berücksichtigt. Im linken Teilbild (a) wird die Variable zu A_a aus den an den Kanten befindlichen Spulen ermittelt und im linken Teilbild die Variable A_b aus den Eckspulen. Bis auf Simulationsungenauigkeiten sowie Feldverzerrungen entspricht die Variable A_b dem geforderten erwarteten Verlauf. Bei einer Annäherung der Sekundärspule liefert die definierte Variable eine Betragssenkung und einer Näherung gegen Null bei optimaler Ausrichtung. Es zeigt sich zudem, dass die Messung ab einem Bereich von unter 500 mm einsetzt.

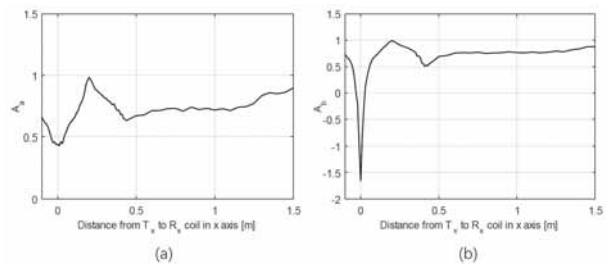


Abbildung 8: Simulierter Verlauf der Variable A während der lateralen Bewegung, (a) Messspulen an den Kanten und (b) an den Ecken der Sekundärspule

Abbildung 9 beinhaltet im linken Teilbild (a) die aus den an den Kanten der Sekundärspule befindenden Messspulen resultierende korrespondierende Variable zu D_p wohingegen die Abbildung (b) die Eckspulen repräsentiert.

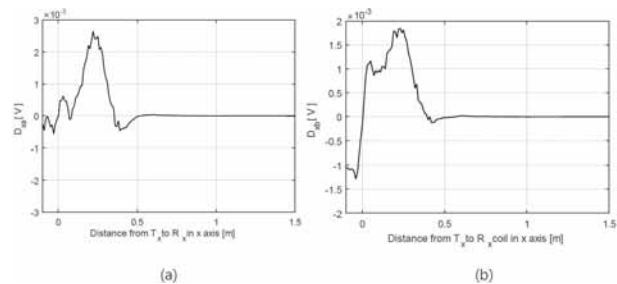


Abbildung 9: Simulierter Verlauf der Variable D während der lateralen Bewegung, (a) Messspulen an den Kanten und (b) an den Ecken der Sekundärspule

Mithilfe der Eckspulen lässt sich ein Verlauf feststellen, der dem erwarteten Verlauf entspricht. Eine laterale Verschiebung der Sekundärspule auf der positiven x-Achse mit Bewegungsrichtung zur Primärspule liefert positive Werte. Ein Vorzeichenwechsel findet ab dem Nullpunkt und damit dem Koordinatenursprung statt, welcher dem

Mittelpunkt der Primärspule entspricht. Der Messbereich des Konzepts wird auch in diesem Fall auf unter 500 mm Distanz zwischen Primär- und Sekundärspule begrenzt.

Somit kann mithilfe der Simulation die generelle Anwendbarkeit des Konzeptes festgestellt werden. Mithilfe der definierten Variablen zur relativen Positionsbestimmung lässt sich eine notwendige Anzahl und Position von Messspulen ableiten, wodurch sich eine in der Sekundärspule integrierte Positioniereinrichtung entwerfen lässt.

3 Zusammenfassung

In dieser Arbeit wird ein Konzept für eine integrierte Positioniereinrichtung für induktive Fahrzeug-Batterieladesysteme im Rahmen einer FEM Simulation mithilfe des Simulationstools „ANSYS Simplorer“ modelliert und simuliert. Dabei handelt es sich um ein Konzept auf Basis dem Gesetz Faraday'schen Induktion, welches über Messspulen eine relative Positionsabweichung der Primär- zur Sekundärspule liefert. Das Werkzeug bietet die Möglichkeit 3D-Modelle der zu betrachtenden Spulensysteme zu erzeugen und diese auf ihre Feldeigenschaften zu untersuchen. Es werden unter anderem Feldverzerrungen aufgrund von Einflüssen der Ferrite sichtbar, deren Auswirkung auf die Güte des entwickelten Positionierverfahrens berücksichtigt und untersucht werden kann. Mithilfe der „Optimetrics“ können bei Paarungen unterschiedlicher Spulentopologien die Primär- zur Sekundärspule verschoben werden, um diverse Parkzustände eines elektrischen Fahrzeugs simulieren zu können. Über die Auswertung der magnetischen Felder mitunter des enthaltenen Werkzeugs „Field Calculator“ können Berechnungen durchgeführt werden und damit Rückschlüsse auf die notwendige und hinreichende Anzahl sowie Positionen der Messspulen abgeleitet werden.

References

- [1] Dietrich TH, Löffler C, Meins J, Henke M, Wussow J, Engel B, Kurczveil T, Callegari J. Conceptual design and integration of a high power inductive charging system into a series vehicle. In: Hybrid and Electric Vehicles. 13th Symposium for Hybrid and Electric Vehicles, 2016Feb; Braunschweig, Germany: IST automotive nord e.V. page 122-134
- [2] Albach, M. *Induktivitäten in der Leistungselektronik*, Wiesbaden: Springer Vieweg, 2017, 422 p.
- [3] Li S, Mi CC, Wireless Power Transfer for Electric Vehicle Applications, *IEEE J. Emerg. Sek. Topics Power Electronics*, vol. 3, no.1 page 4-17, March 2015

- [4] Gao Y, Duan C, Oliveira AA, Ginart A, Farley KB, Tsz Ho Tse Z. 3-D Coil Positioning Based in Magnetic Sensing for Wireless EV Charging, *IEEE Transactions on Transportation Electrification Vol.3*, 2017, S.578-588, doi: 10.1109/TTE.2017.2696787

Untersuchung des kognitiven menschlichen Verhaltens bei der Personalisierung fahrzeugmechatronischer Systeme

Haoqi Tao^{1*}, Xiaobo Liu-Henke¹, Thomas Vietor²

¹Ostfalia Hochschule für angewandte Wissenschaften, Institut für Mechatronik, Salzdahlumer Straße 46/48, 38302 Wolfenbüttel, *hao.tao@ostfalia.de

²Technische Universität Braunschweig, Institut für Konstruktionstechnik, Langer Kamp 8, 38106 Braunschweig

Abstract. Der vorliegende Beitrag stellt die Untersuchung des menschlichen Verhaltens zur Berücksichtigung vom kognitiven Verhalten verschiedener Menschen im Entwicklungsprozess mechatronischer Fahrzeugsysteme vor. Die Untersuchung wird modellbasiert durchgeführt und mithilfe eines Anforderungsmanagements unterstützt. Der vorgestellte Ansatz wird durch Simulation des Wirkungskreises Fahrer¹-Fahrzeug-Fahrumgebung am Beispiel der Fahrzeugquerführung verifiziert.

Einleitung

Umfragen [1] zeigen, dass der Anteil falscher Warnungen von Spurhalteassistenten für weibliche Fahrer 8%, für männliche Fahrer 21% und für ältere Fahrer 13% (61 Jahre alt oder älter) beträgt, wohingegen er 62% für jüngere Fahrer (40 Jahre alt oder jünger) beträgt. Hieraus wird deutlich, dass die Wirksamkeit solcher Systeme für verschiedene Fahrer unterschiedlich ist. Das bedeutet, dass die Systemstrategie für verschiedene Arten von Fahrern unterschiedlich sein sollte, um das spezifische Verhalten zu berücksichtigen. Werden die Unterschiede menschlichen Verhaltens nicht bei der Systementwicklung betrachtet, wird die Akzeptanz des Fahrers nicht gewährleistet. Somit ist es notwendig, unterschiedliche menschliche Verhaltensweisen in der Systementwicklung zu berücksichtigen.

Im Teilprojekt „Personalisierung mechatronischer Systeme im Kraftfahrzeug“ des vom Niedersächsischen MWK geförderten Promotionskollegs KoMMA.G „Konfigurationen von Mensch, Maschine & Geschlecht – Interdisziplinäre Analysen zur Technikentwicklung“ wer-

den die Interaktionen zwischen Menschen und mechatronischen Fahrzeugsystemen, besonders Advanced Driver Assistance Systems (ADAS), untersucht. Darauf basierend soll der vorhandene Entwicklungsprozess mechatronischer Systeme um die Berücksichtigung unterschiedlichen menschlichen Verhaltens erweitert werden.

Abbildung 1 stellt das Konzept des erweiterten Entwicklungsprozesses mechatronischer Systeme dar. Unter Berücksichtigung eines strukturierten Anforderungsmanagements soll der mechatronische Entwicklungsprozess alle Entwicklungsschritte von der Modellbildung über die Analyse bis zur Realisierung beinhalten. Darüber hinaus illustriert Abbildung 1 mögliche Lücken im vorhandenen Entwicklungsprozess, welcher unterschiedliches, menschliches Verhalten nicht berücksichtigt [2].

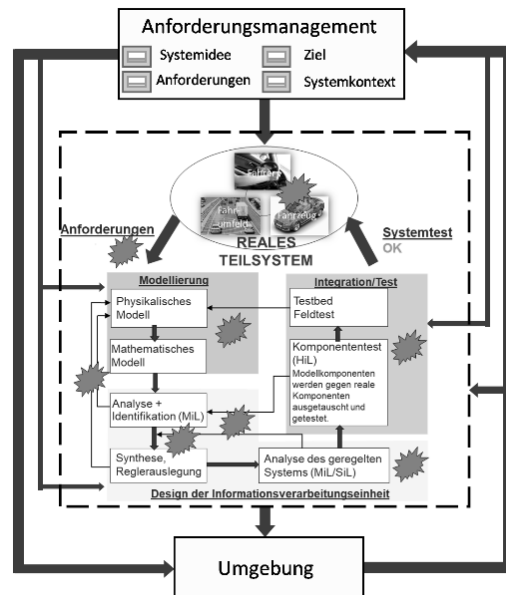


Abbildung 1: Konzept des Entwicklungsprozesses

¹ In diesem Beitrag wird zur besseren Lesbarkeit die männliche Form verwendet. Sie bezieht sich auf Personen beider Geschlechter.

Ein Schwerpunkt des Konzepts fokussiert die Untersuchung des kognitiven menschlichen Verhaltens und eine Analyse des möglichen Einflusses auf das Fahrverhalten. Im vorliegenden Beitrag werden die Modellbildung und Simulation des menschlichen Verhaltens unter Verwendung eines Anforderungsmanagements dargestellt.

1 Stand des Wissens

Bisher existieren viele verschiedene Fahrermodelle. Der erste Entwurf eines quasi-linearen Menschen-Modells, der sogenannte „universelle Regler“, wurde von Tustin entwickelt [3]. Die menschlichen Regeleigenschaften sind in Form einer Übertragungsfunktion beschrieben:

$$G_{Tu}(s) = \frac{K_{Tu}(T_{A,Tu}s + 1)}{(T_{1,Tu}s + 1)(T_{N,Tu}s + 1)} e^{-\tau_{Tu}s} \quad (1)$$

Dabei wird die menschliche Informationsaufnahme durch einen Verstärkungsfaktor K_{Tu} und ein Vorhaltglied mit der Zeitkonstante $T_{A,Tu}$ beschrieben. Die Regeleigenschaften des Menschen werden durch die Zeitkonstante $T_{1,Tu}$ repräsentiert. Die Zeitkonstante $T_{N,Tu}$ steht für die neuromuskuläre Verzögerung des menschlichen Handlungssystems. Zudem stellt Totzeit τ_{Tu} die menschliche Reaktionszeit dar.

Nach der Cross-Over-Theorie von McRuer [4] wird der aus Fahrer und Fahrzeug bestehende offene Regelkreis durch folgende Gleichung dargestellt:

$$G_M(s) \cdot G_{Fzg}(s) = \frac{\omega_c}{s} e^{-\tau_{Mc}s} \quad (2)$$

Die Schnittfrequenz ω_c wurde durch zahlreiche Experimente in einem Bereich von 0,08 bis 0,32 Hz bestimmt [3]. Die Zeitkonstante τ_{Mc} stellt die gesamte menschliche Verzögerung dar. Dieser Ansatz wurde zu zahlreichen unterschiedlichen Fahrermodellen weiterentwickelt. In [5] werden darüber hinaus Ansätze zur Parameteridentifikation vorgestellt.

Um das dynamische Querverhalten des Fahrers nachzubilden, wird ein Zwei-Ebenen-Modell entwickelt [6]. Auf einer Vorausschaustrategie basierend werden prädiktive Fahrermodelle zur Simulation der Spurfolge betrachtet [7], [8]. In [9] hat Apel die Fahrermodelle für die Quer- und Längsführung erstmals durch Einführung der Größen der Time-to-Collision (TTC) und Time-to-Line-Crossing (TLC) miteinander verbunden.

Die vorgestellten Fahrermodelle sind auf der Grundlage der Systemtheorie aufgebaut. Die Ergebnisse der Literaturrecherche zeigen, dass diese Modelle das mensch-

liche Verhalten teilweise beschreiben können. Diese Modelle werden jedoch basierend auf einem Durchschnittsfahrer aufgebaut und sind auf spezifische Fahrsituationen oder -aufgaben beschränkt.

In [10] und [11] wird ein Künstliches Neuronales Netzwerk (KNN) zur Nachbildung der menschlichen Eigenschaften bei der Fahrzeugführung eingesetzt. Die Untersuchungen zeigen, dass KNN-Fahrermodelle die nicht-lineare Charakteristik des Fahrerhaltens darstellen können. Das Hauptproblem besteht jedoch in der schwierigen Interpretierbarkeit, da KNN-Fahrermodelle als Black-Box-Modelle durch autonomes Lernen adaptiert werden.

Um die Personalisierung fahrzeugmechatronischer Systeme im vorhandenen Entwicklungsprozess umzusetzen, sollen neben technischen Beschreibungen auch die kognitiven Verhaltensweisen verschiedener Menschen im Fahrermodell repräsentiert werden.

2 Anforderungen

Die Modellierung des menschlichen Verhaltens ist eine komplexe Aufgabe [12]. Die Art und Komplexität der Modellierung ist stark vom individuellen Anwendungsfall abhängig und die charakteristischen Parameter für menschliches Verhalten hängen gleichfalls stark von der individuellen Fahraufgabe und -situation ab [5]. Die Modellierung erfordert daher eine detaillierte Analyse der Fahraufgabe und -situation (z. B. Geschwindigkeitskontrolle, Spurwechsel) sowie der Anwendungsfälle in der Entwicklung (z. B. Regelauslegung, Test). Als Ergebnisse dieser Analyse werden zahlreiche Anforderungen an das gesamte Fahrzeugsystem verfeinert, um die vom Fahrer geforderten Systemeigenschaften zu erfüllen. In diesem Zusammenhang werden die Beziehungen der Anforderungen durch das ein Diagramm dargestellt. Somit können die Anforderungen erfasst, verarbeitet und verwaltet werden, um die Berücksichtigung verschiedener Anforderungen im Entwicklungsprozess zu verbessern und eine Basis für die Nachverfolgung von Anforderungen zu schaffen [13]. Dies dient nicht nur der Unterstützung in der Modelbildungsphase, sondern auch für weitere Entwicklungsphasen (vgl. Abbildung 1).

In [14] wird ein Ansatz zur Anforderungsmodellierung vorgestellt, dessen Konzept acht Partialmodelle umfasst, welche die Beschreibung der komplexen Anforderungen und ihrer Beziehungen ermöglichen. In dieser Arbeit werden vier Partialmodelle dieser verwendet. Abbil-

Abbildung 2 stellt die aufgebauten vier Partialmodelle (Systemidee, Ziel, Anforderungen und Systemkontext) zur Anforderungsmodellierung für die Modellbildungsphase dar. Die Anforderungsmodellierung wird mithilfe der Modellierungssprache SysML in der Software *Enterprise Architect* durchgeführt.

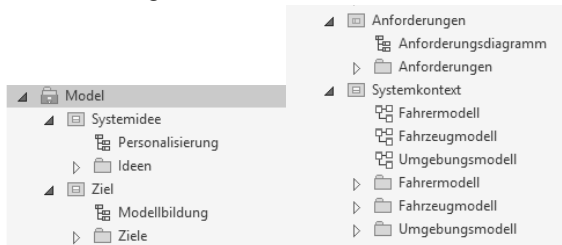


Abbildung 2: Partialmodelle für Anforderungsmodellierung

Abbildung 3 zeigt das Anforderungsdiagramm zur Nachbildung des menschlichen Verhaltens. Die Anforderungsobjekte werden mit den Stereotypen <<requirement>> gekennzeichnet, während die zu entwickelnden Modelle mit den Stereotypen <<block>> gekennzeichnet sind. Am linken Bildrand sind die hierarchischen Anforderungen an die Modellbildungsphase und rechts Entwicklungsziele und -ideen dargestellt. Beispielsweise ist die Idee „Personalisierung“ durch die Entwicklungsaufgabe „Modellbasierte Untersuchung der Interaktion“ erfüllt. Diese Untersuchung wird über die „Modellbasierte Untersuchung des menschlichen Verhaltens“ verfeinert. Um diese Entwicklungsaufgabe zu lösen, soll ein Fahrermodell entwickelt werden, welches die Anforderungen „Nachbildung der Vorausschaulänge“, „Nachbildung der Regeleigenschaften“ und „Nachbildung der Prädiktionsfähigkeit“ sowie „Nachbildung kognitiver Verhaltensweisen“ erfüllt.

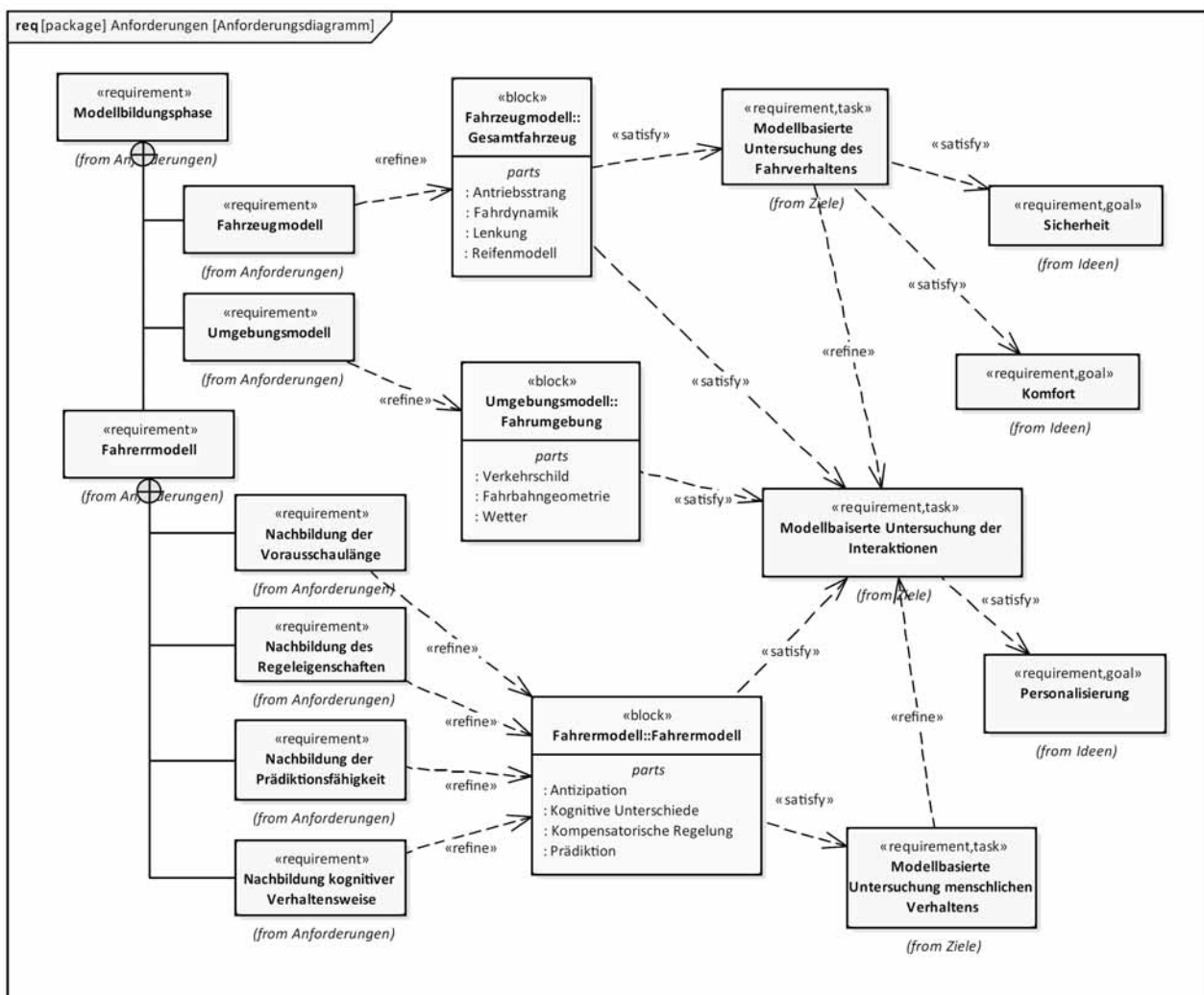


Abbildung 3: Anforderungsdiagramm

3 Modellbildung

In diesem Kapitel wird die Modellbildung des kognitiven menschlichen Verhaltens am Beispiel der Fahrzeugquerführung vorgestellt.

3.1 Fahrermodell

Gemäß der in Kapitel 2 hergeleiteten Anforderungen wird ein Fahrermodell entwickelt. Abbildung 4 zeigt die Struktur und die Hauptfunktionalitäten eines Fahrermodells, welches in der Fahrer-Fahrzeug-Fahrumsgebung eingebettet ist.

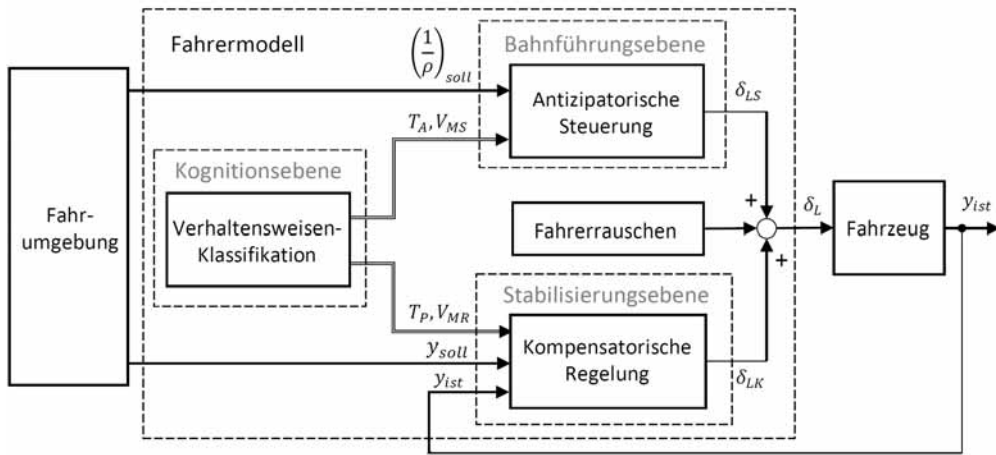


Abbildung 4: Struktur und Funktionalitäten des Fahrermodells

Um die Antizipation der Straßenkurve zu modellieren, soll das Einspurmodell zunächst um die Vorausschläge $l_V + l_P$ bzw. die Antizipationszeit $T_A = (l_V + l_P)/v$ des Menschen erweitert werden [15], s. Abbildung 5.

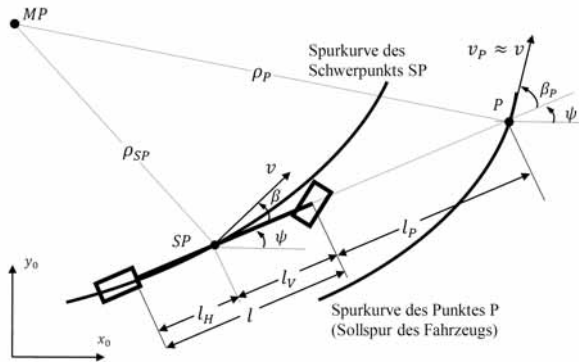


Abbildung 5: Erweitertes Einspurmodell

Die charakteristischen Werte des erweiterten Einspurmodells, die ungedämpfte Eigenkreisfrequenz $\omega_{n,id}$ und das zugehörige Dämpfungsmaß $\sigma_{n,id}$ werden anhand charakteristischer Bewegungsgleichungen hergeleitet:

Das Fahrermodell untergliedert sich in eine Führungs-, eine Stabilisierung- und eine Kognitionsebene. Die Führungsebene wird durch eine antizipatorische Steuerung, die auf der Fahrbahngeometrie basiert, beschrieben, um die vorausschauende Wahrnehmungsfähigkeit abzubilden. Die Stabilisierungsebene wird in Form einer kompensatorischen Regelung abgebildet, um die Reaktion auf Kursabweichungen zwischen Soll- und Ist-Spur zu modellieren. Auf kognitiver Ebene werden die unterschiedlichen Verhaltensweisen durch Algorithmen und die Variation verschiedener Fahrerparameter beschrieben.

$$\omega_{n,id} = \sqrt{\frac{c_{\alpha H} l}{J_z + m l_V (l_V + l_P)}} \quad (3)$$

$$\sigma_{n,id} = \frac{l + l_P}{2v} \cdot \frac{c_{\alpha H} l}{J_z + m l_V (l_V + l_P)} \quad (4)$$

Hieraus lässt sich das technische Verhalten des Fahrzeugs als Übertragungsfunktion

$$G_{\delta_L} = \frac{\delta_L(s)}{\kappa_{soll}(s)} = V_{Fzg} \frac{\frac{1}{\omega_n^2} s^2 + \frac{2\sigma}{\omega_n^2} s + 1}{\frac{2\sigma_{id}}{\omega_{n,id}^2} s^2 + \frac{1}{\omega_{n,id}^2} s + 1} \quad (5)$$

von Lenkwinkel zu Straßenkrümmung mit der Fahrzeugverstärkung

$$V_{Fzg} = i_L l (1 + (v/v_{ch})^2) \quad (6)$$

beschreiben. Nach [16] kann die antizipatorische Steuerung des Menschen durch die Übertragungsfunktion $G_{MS}(s)$ beschrieben werden (Gl. 7). Der Steuerungsanteil, der Lenkwinkel δ_{LS} , wird aufgrund der Sollkrümmung κ_{soll} über einen Verstärkungsfaktor V_{MS} und ein PT2-Verzögerungsglied sowie ein Vorhaltglied $e^{T_A s}$ eingestellt.

$$G_{MS} = \frac{\delta_{LS}(s)}{\kappa_{soll}(s)} = V_{MS} \frac{e^{T_A s}}{T_{2S}^2 s^2 + T_{1S} s + 1} \quad (7)$$

Das Vorhaltglied wird durch eine nach dem zweiten Glied abgebrochene Taylor-Reihe angenähert:

$$e^{T_A s} \approx 1 + T_A s + \frac{T_A^2}{2} s^2 \quad (8)$$

Aus dem Koeffizientenvergleich von G_{MS} und G_{δ_L} lassen sich die menschlichen Parameter durch Fahrzeugparameter darstellen. Es gilt wie in Tabelle 1 gezeigt.

Tabelle 1: Fahrerparameter für ideale Antizipation

Parameter	Werte
V_{MS}	$i_L l [1 + (v/v_{ch})^2]$
T_A	$2\sigma/\omega_n^2$
T_{1S}	$2\sigma_{id}/\omega_{n_{id}}^2$
T_{2S}	$1/\omega_{n_{id}}$

Der Mensch reagiert nicht direkt auf die augenblickliche Kursabweichung des Fahrzeugschwerpunktes Δy , sondern er sieht um eine bestimmte Prädiktionszeit T_p voraus und reagiert auf die geschätzte Querabweichung $\Delta y(t + T_p)$ des voraussichtlichen Punkts, vgl. [9], [15]. Somit wird hierbei das Prädiktionsglied

$$G_{Pr} = \frac{\Delta y(s) e^{T_p s}}{\Delta y(s)} \approx 1 + T_p s + \frac{T_p^2}{2} s^2 \quad (9)$$

zur Modellierung der menschlichen Wahrnehmung der Kursabweichung eingesetzt.

Zur kompensatorischen Regelung wird das McRuer-sche Präzisionsmodell [17] mit der Übertragungsfunktion G_{MR} des kompensatorischen Anteils des Lenkwinkels $\delta_L(s)$ zur Kursabweichung $\Delta y(s)$ verwendet.

$$G_{MR} = \frac{\delta_{LR}(s)}{\Delta y(s)} = V_{MR} \frac{1 + T_D s}{1 + T_I s} e^{-\tau s} \quad (10)$$

Dabei wird T_D zu Null gesetzt, da das Vorhaltgliedverhalten bereits durch das Prädiktionsglied berücksichtigt wird.

Die Vorausschaulänge bzw. Vorausschauzeit ist ein Maß für die Glattheit des Fahrverhaltens [18]. Sie steigt mit zunehmender Erfahrung des Fahrers. Die Intensität der Fahrerreaktion auf Veränderungen der Querabweichung kann mittels Menschverstärkung V_{MR} beschrieben werden. Somit kann der Unterschied verschiedener Verhaltensweisen auf Kognitionsebene durch Klassifikation der Verhaltensweisen nachgebildet werden. Die Klassifikation erfolgt mittels Variation der Parameter Voraus-

schauzeit T_p und Menschverstärkung V_{MR} . Gemäß Ergebnissen aus der Verkehrspsychologie werden vier verschiedene kognitive Verhaltensweisen definiert [18]: Erfahren und risikobereit, Erfahren und vorsichtig, Anfänger und risikobereit sowie Anfänger und vorsichtig. Tabelle 2 zeigt die entsprechenden Parameter der vier verschiedenen Verhaltensweisen.

Tabelle 2: Parameter für verschiedene Verhaltensweisen

Verhaltensweisen	Vorausschauzeit T_p	Verstärkung V_{MR}
Erfahren, risikobereit	0,7	0,2
Erfahren, vorsichtig	0,8	0,9
Anfänger, risikobereit	0,18	0,3
Anfänger, vorsichtig	0,2	1,1

3.2 Fahrzeugmodell

Als Fahrzeugmodell wird ein lineares Einspurmodell [15] verwendet. Unter Annahme kleiner Lenkwinkel beschreibt das Einspurmodell die dynamische Beziehung zwischen dem Lenkwinkel δ und die laterale Position y des Fahrzeugs bei konstanter Geschwindigkeit.

3.3 Umgebungsmodell

Das Umgebungsmodell wird in dieser Arbeit durch die Fahrbahngeometrie abgebildet. Zwecks Untersuchung des Querführungsverhaltens hat der Radius des Krümmungskreises ρ an der jeweiligen Stelle der Fahrbahn eine große Bedeutung. Im modernen Straßenbau kommen drei Grundelemente *Gerade*, *Kreis* und *Klothoide* zum Einsatz [19], die bei geeigneter Kombination eine Unstetigkeit des Querkraftverlaufes vermeiden. Die Krümmung wird mit dem Radius R des Kreises, der Bogenlänge S der Klothoide und dem Klothoidenparameter A ($\frac{R}{3} \leq A \leq R$) berechnet (Tabelle 3).

Tabelle 3: Krümmung der Fahrbahn-Grundelemente.

Elemente	Krümmung
Gerade	0
Kreis	$1/R$
Klothoide	S/A^2

Zur Berechnung einer Klothoide im raumfesten Koordinatensystem werden folgende Gleichungen genutzt:

$$X = \int_0^S \cos\left(\frac{S^2}{2A^2}\right) dS \quad (11)$$

$$Y = \int_0^S \sin\left(\frac{S^2}{2A^2}\right) dS \quad (12)$$

4 Simulation

Nach der Modellbildung soll die Simulation des gesamten Wirkungskreises Fahrer-Fahrzeug-Fahrumgebung mittels J-Turn-Manöver durchgeführt werden, um den Einfluss auf das Fahrverhalten zu analysieren.

In Abbildung 6 werden die Trajektorie, die Querabweichung und der Gierwinkel des Fahrzeugs sowie der vom Fahrer ausgeführte Lenkradwinkel bei vorsichtiger und risikobereiter Fahrt eines Anfängers miteinander verglichen. Es zeigt sich eine große Querabweichung zur Soll-Trajektorie und ebenfalls zur Querabweichung.

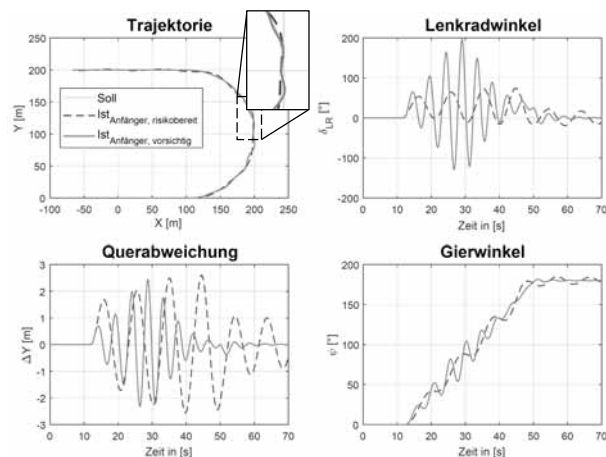


Abbildung 6: Gegenüberstellung von risikobereiter und vorsichtiger Fahrt

Der Verlauf des Lenkradwinkels und der des Gierwinkels weist nach, dass der vorsichtige Anfänger häufigere und stärkere Lenkbewegungen als der risikobereite Anfänger ausführt. Dieses Verhalten resultiert in einem schwingenden Verhalten des Gierwinkels mit höherer Frequenz und Amplitude als bei der risikobereiten Fahrt.

Das bedeutet, dass das menschliche Verhalten einen direkt Einfluss sowohl auf das stationäre als auch auf das dynamische Verhalten des Fahrzeugs hat.

Der Vergleich zwischen Anfänger und erfahrener Fahrer (Abbildung 7) zeigt analog zur vorherigen Analyse, dass das kognitive Verhalten verschiedener Menschen einen wesentlichen Einfluss auf das Fahrverhalten

des Fahrzeugs hinsichtlich statischen und dynamischen Verhaltens hat.

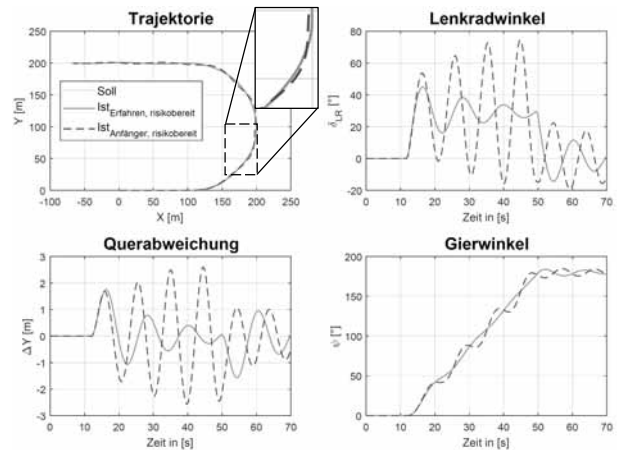


Abbildung 7: Gegenüberstellung von Anfänger und Erfahrenen

Aus den Simulationsergebnissen wird ersichtlich, dass unterschiedliche Fahrer einen unterschiedlichen Einfluss auf das Fahrverhalten ausüben. Ebenfalls hat ein Fahrer gleicher Erfahrung ebenfalls einen unterschiedlichen Einfluss auf das Fahrverhalten, wenn er mit unterschiedlichen Verhaltensweisen fährt. Diese Betrachtungen weisen nach, dass der kognitive Unterschied weiteren Entwicklungsphasen berücksichtigt werden muss.

5 Resümee und Ausblick

In diesem Beitrag wurden Modellbildung und Simulation des menschlichen Verhaltens unter Verwendung eines Ansatzes zur Anforderungsmodellierung dargestellt. Mittels Simulation wurden die Einflüsse verschiedener menschlicher Verhaltensweisen auf das Fahrverhalten analysiert.

Weitere Forschungsarbeiten befassen sich mit der Entwicklung eines Lenkungsprüfstands, um die Validierung unter Echtzeitbedingung mittels Hardware-in-the-Loop (HiL)-Simulation durchzuführen. Des Weiteren werden Verbesserungspotenziale zur Ergänzung des Fahrermodells fokussiert, um weitere individuelle Charakteristika hinsichtlich Psychologie und Kognitionstheorie zu repräsentieren.

Danksagung

Diese Arbeit wurde im Rahmen des Promotionskollegs KoMMA.G durch das Niedersächsische Ministerium für Wissenschaft und Kultur gefördert. Für die Förderung bedanken sich die Autoren herzlichst.



Niedersächsisches Ministerium
für Wissenschaft und Kultur



KONFIGURATIONEN VON MENSCH,
MASCHINE & GESCHLECHT
Interdisziplinäre Analysen zur Technikentwicklung

Literaturverzeichnis

- [1] A. Eichelberger und A. McCartt, „Toyota drivers' experiences with Dynamic Radar Cruise Control, Pre-Collision System, and Lane-Keeping Assist,“ *Journal of Safety Research*, 2015.
- [2] X. Liu-Henke, H. Tao und S. Jacobitz, „Konzeption des Entwicklungsprozesses zur Berücksichtigung von Genderaspekten in fahrzeugmechatronischen Systemen,“ in *Workshop der ASIM/GI-Fachgruppen*, Heilbronn, 2018.
- [3] H. Bubb, „Menschmodelle,“ in *Automobilergonomie, ATZ/MTZ-Fachbuch*, Wiesbaden, Springer Fachmedien, 2015.
- [4] D. T. McRuer, B. Graham, E. S. Krendel und W. Reisner, „Human Pilot Dynamics in Compensatory Systems,“ AFFDL-TR-65-15, wright patterson, 1965.
- [5] W. Wang, J. Xi und H. Chen, „Modeling and Recognizing Driver Behavior Based on Driving Data: A Survey,“ in *Mathematical Problems in Engineering*, Peking, 2014.
- [6] E. Donges, „Ein regelungstechnisches Zwei□ Ebenen□Modell des menschlichen Lenkverhaltens im Kraftfahrzeug,“ *Zeitschrift für Verkehrssicherheit*, pp. 98-112, 1978.
- [7] A. Reñski, „Identification of Driver Model Parameters,“ *INTERNATIONAL JOURNAL OF OCCUPATIONAL SAFETY AND ERGONOMICS*, pp. 79-92, 2001.
- [8] A. Reñski, „The Driver Model and Identification of Its Parameters,“ in *SAE*, Detroit, 1998.
- [9] A. Apel, Modellierung des Fahrverhaltens bei Längs- und Querführung von Pkw, Diss., Technische Universität Braunschweig, 1998.
- [10] T. Jürgensohn, Hybride Fahrermodelle, Diss., Technische Universität, 1997.
- [11] A. K. Prakash, „Artificial Neural Network Based Driver Modeling for Vehicle Systems,“ in *SAE International*, 2013.
- [12] G. Hiesgen, „Effiziente Entwicklung eines menschzentrierten Querführungsassistenzsystems mit einem Fahr Simulator,“ Diss., Universität Duisburg-Essen, 2011.
- [13] C. Ebert, Systematisches Requirements Engineering, Heidelberg: dpunkt, 2014.
- [14] C. Stechert, „Modellierung komplexer Anforderungen,“ Diss., TU Braunschweig, 2010.
- [15] M. Mitschke und H. Wallentowitz, Dynamik der Kraftfahrzeuge, Braunschweig: Vieweg Springer, 2014.
- [16] A. Horn, Fahrer-Fahrzeug-Kurvenfahrt auf trockener Straße, Diss., Technische Universität Braunschweig, 1985.
- [17] H.-J. Risse, Das Fahrerverhalten bei normaler Fahrzeugführung, Paderborn: VDI Fortschritt-Berichte, 1991.
- [18] R. Fischer, M. Ehmman und M. Irmscher, „Fahrermodellierung für Fahrdynamik und Verbrauchsberechnungen,“ VDI-Verlag, Berlin, 2008.
- [19] B. Klauenberg und C. Achttert, „Modellbasierter Entwurf einer antizipatorischen Steuerung zur Steuerung zur Querführung eines Fahrzeuges (SA),“ Ostfalia Hochschule, IMEC, Wolfenbüttel, 2011.

Modellbasierte Entwicklung einer Spurfolgeregelung mittels zeit-diskretem modellprädiktivem Regler (DMPC) für unteraktuierte Systeme

Jie Zhang^{1*}, Marian Göllner¹, Xiaobo Liu-Henke¹

¹Ostfalia Hochschule für angewandte Wissenschaften, Fakultät Maschinenbau, Institut für Mechatronik, Salzdahlumer Str. 46/48, 38302 Wolfenbüttel; * jie.zhang@ostfalia.de

Abstract. Im vorliegenden Beitrag wird ein zeitdiskreter modellprädiktiver Regelalgorithmus (DMPC) zur Entwicklung einer Spurfolgeregelung für ein unteraktuiertes und instabiles Mehrköpersystem ausgelegt. Dazu wird zunächst ein vereinfachtes lineares Zustandsraummodell des zu regelnden mechatronischen Systems hergeleitet und erweitert. Anschließend werden Regelalgorithmen ausgelegt, welche mittels diesem in einem Prädiktionshorizont das Spurfolgen optimieren. Um die Rechenkosten durch quadratische Programmierung bei der Einstellung der Abstimmungskosten des DMPC zu verringern, wird in dieser vorliegenden Arbeit eine Kaskadenregelstruktur verwendet.

Einleitung

Zur Optimierung des mechanischen Aufbaus, der Kosten und des Energieverbrauchs werden in Industrie und Verkehr unteraktuierte Mehrköpersysteme, die weniger Aktuatoren als Bewegungsfreiheitsgrade besitzen, verwendet. Jedoch sind unteraktuierte Mehrköpersysteme üblicherweise nicht-minimalphasig und daher bei Arbeitspunktwechseln sowie der Trajektorienfolge viel schwieriger als im voll aktuierten Fall zu regeln [1]. In diesem vorliegenden Beitrag soll deshalb ein zeitdiskreter modellprädiktiver Regelalgorithmus (DMPC) zur Entwicklung einer Spurfolgeregelung für solch ein unteraktuiertes und instabiles Mehrköpersystem angepasst und erweitert werden.

Gegenüber anderen Regelalgorithmen liegen die Vorteile eines DMPC insbesondere darin, dass während der Reglerauslegung die Beschränkung der Stellgröße berücksichtigt und die optimale Kontrollfolge mittels quadratischer Programmierung iterativ gelöst werden kann. Dadurch das der nächste zu erwartenden Sollwertpunkt der vorgegebenen Trajektorie mitberücksichtigt wird,

können exaktere Ergebnisse gewährleistet werden [2].

In diesem Beitrag wird die Entwicklung einer Spurfolgeregelung mittels DMPC für den Forschungsträger S(phere)-Mobile, der im Rahmen eines Forschungsvorhabens als ein hochdynamisches und unteraktuiertes System mit sphärischem Elektroantrieb mechatronisch entwickelt wird, durchgeführt. Aus diesem Anwendungsfall können Grundkenntnisse für die Reglerauslegung mittels DMPC für andere unteraktuierte Systeme gewonnen werden.

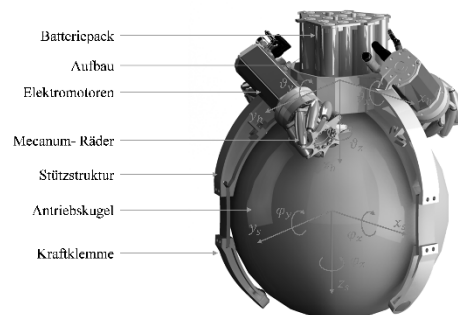


Abbildung 1. Konzept des S(phere)-Mobile nach [3].

Da solche unteraktuierten Systeme aufgrund ihrer Bewegungsfreiheitsgrade zahlreiche Zustandsvariable haben, kann lediglich durch Einstellung der Abstimmungsparameter des DMPC, das gesamte System nur schwer stabilisiert werden. Die Anzahl an rückgeführten Zustandsvariablen muss reduziert werden da sonst die Rechenkosten durch quadratische Programmierung beträchtlich zunehmen. Aus diesem Grund wird in dieser vorliegenden Arbeit eine Kaskadenregelstruktur verwendet, wobei der innere Regler zur Stabilisierung des Systems und der äußere Regler als Spurfolgeregler verwendet werden. Die Funktionalität wird durch Verifikation mittels MiL, SiL und HiL getestet.

1 Methode

Zur Entwicklung dieses komplexen Regelsystems wird der in Abbildung 2 dargestellte mechatronische Entwicklungskreislauf herangezogen. Am Beginn der theoretischen Untersuchung steht die Modellbildung. Das technische System wird nach Funktionsprinzipien im Hinblick auf die Anforderung in ein physikalisches bzw. mathematisches Ersatzmodell aus unterschiedlichen Fachdisziplinen wie Mechanik, Hydraulik und Elektronik abgebildet. Es repräsentiert das Systemverhalten. Die Regelstrategie, die vom einfachen Regler bis hin zu hierarchisch angeordneten Mehrgrößenregelstrukturen aufgebaut sein kann, wird anhand des Systemverhaltens festgelegt.

Die anschließende modellbasierte Komposition erfolgt gemäß des Rapid Control Prototyping (RCP) in einem durchgängig verifikationsorientierten Prozess aus Model-in-the-Loop (MiL), Software-in-the-Loop (SiL) und Hardware-in-the-Loop (HiL) [4].

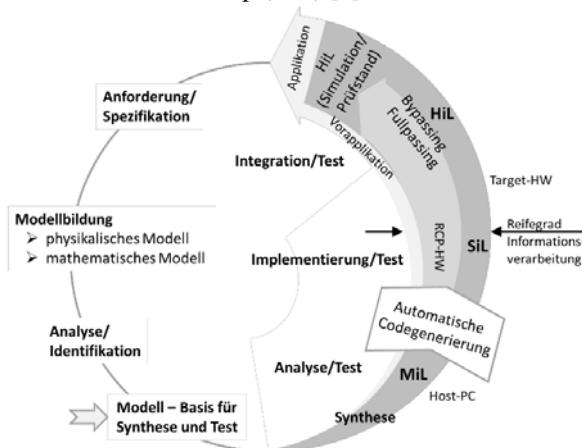


Abbildung 2. Mechatronischer Entwicklungskreislauf [4]

2 Modellbildung

Eine wesentliche Grundlage für die Auslegung und Entwicklung eines leistungsfähigen präzisierenden Regelungssystems ist ein mathematisches Modell der Regelstrecke.

2.1 Lineares Modell des S-Mobile

Zunächst wird ein physikalisches Modell nach angenommenen Vereinfachungen und Anforderungen aus dem realen Modell abgeleitet. Für die mathematische Modellierung des S-Mobile wird anschließend die analytische Methode bzw. Langrange'sche Gleichungen als

geeignete Methode zur Aufstellung der Bewegungsgleichungen des Mehrkörpersystems (MKS), deren gesamte Freiheitsgerade gleich 5 in einen verallgemeinerten Koordinatenvektor \underline{q} zusammengefasst werden, im Rahmen der Grundlagen der technischen Mechanik gewählt [5].

$$\underline{q} = [\theta_x \ \theta_y \ \theta_z \ \phi_x \ \phi_y]^T \quad (1)$$

Die nichtlinearen Bewegungsgleichungen des S-Mobile werden in die folgende Gleichung überführt:

$$\underline{y} = \underline{f}(\underline{u}, \underline{q}, \dot{\underline{q}}, \ddot{\underline{q}}) \quad (2)$$

Anhand der Definition der Taylor-Reihe werden die nichtlinearen Bewegungsgleichungen um den vorgegebenen Arbeitspunkt linearisiert und in eine Zustandsraumdarstellung überführt:

$$\begin{aligned} \dot{\underline{x}} &= \underline{A}_c \cdot \underline{x}_s + \underline{B}_c \cdot \underline{u} \\ \underline{y} &= \underline{C}_c \cdot \underline{x} \\ \underline{u} &= [\tau_1 \ \tau_2 \ \tau_3]^T \end{aligned} \quad (3)$$

$$\underline{x} = [\theta_x \ \dot{\theta}_x \ \theta_y \ \dot{\theta}_y \ \theta_z \ \dot{\theta}_z \ \phi_x \ \dot{\phi}_x \ \phi_y \ \dot{\phi}_y]^T$$

$$\underline{y} = [\theta_x \ \theta_y \ \theta_z \ \phi_x \ \phi_y]^T$$

Das Systemverhalten wird durch die Eigenwerte bzw. die Pollage des charakteristischen Polynoms aus der Dynamikmatrix $\underline{A}_{c,s}$ analysiert. Es ist auffällig, dass dieses System instabil gemäß den Eigenwerten ist, da nicht alle Eigenwerte auf der linken Seite der imaginären Achse liegen.

$$\lambda_{1,2,\dots,10} = \begin{bmatrix} 0 & 0 & 0 & 0 & 6,1287 & -6,1287 \\ & & & & 6,1315 & -6,1315 & 0 & 0 \end{bmatrix} \quad (4)$$

2.2 Erweitern des Zustandsraums

Anschließend muss eine geeignete mathematische Beschreibungsform des dynamischen Systems gefunden werden, da die gewünschte Realisierung der modellprädiktiven Regelung auf einer Echtzeithardware mit fester Schrittweite erfolgen soll. Um die Integration zu erleichtern, ist eine diskrete Beschreibungsform des Systems erstrebenswert. Die kontinuierliche Zustandsraumdarstellung in Gl. (3) wird dazu in eine diskrete Darstellung innerhalb eines Zeitschrittes überführt:

$$\begin{aligned} \Delta \underline{x}(k+1) &= \underline{A}_d \cdot \Delta \underline{x}(k) + \underline{B}_d \cdot \Delta \underline{u}(k) \\ \Delta \underline{y}(k+1) &= \underline{C}_d \cdot \underline{A}_d \cdot \Delta \underline{x}(k) + \underline{C}_d \cdot \underline{B}_d \cdot \Delta \underline{u}(k) \end{aligned} \quad (5)$$

Dargestellt in einem Zustandsraummodell nach Erweiterung um die ausgewählten Ausgangsgrößen kann das dynamische System wie folgt beschrieben werden:

$$\begin{aligned} \underline{x}(k+1) \\ \begin{bmatrix} \Delta \underline{x}(k+1) \\ \underline{y}(k+1) \end{bmatrix} &= \begin{bmatrix} \underline{A}_d & \underline{0}_d^T \\ \underline{C}_d \cdot \underline{A}_d & \underline{I}_{q \times q} \end{bmatrix} \begin{bmatrix} \underline{x}(k) \\ \underline{y}(k) \end{bmatrix} \\ &+ \begin{bmatrix} \underline{B}_d \\ \underline{C}_d \cdot \underline{B}_d \end{bmatrix} \cdot \Delta \underline{u}(k) \\ \underline{y}(k) &= \begin{bmatrix} \underline{A}_d & \underline{I}_{q \times q} \end{bmatrix} \begin{bmatrix} \Delta \underline{x}(k) \\ \underline{y}(k) \end{bmatrix} \end{aligned} \quad (6)$$

Ein Vorteil dieses erweiterten Zustandsraummodells ist, dass die Ausgangsgrößen des Systems direkt in einer Gleichung berechnet werden können. Somit können die Abweichungen von den gewünschten Werten direkt ebenfalls in dieser Gleichung berücksichtigt werden [6].

3 Modellbasierte Reglerauslegung

Auf Basis des erweiterten Zustandsraummodells können nun zukünftige Zustands- und Ausgangsgrößen des Systems mit gegebenem Prädiktions- und Kontrollhorizont bzw. N_p und N_c prädiziert werden.

Dargestellt in einer kompakten Matrixform ergibt sich:

$$\underline{Y} = \underline{F} \cdot \underline{x}(k_i) + \underline{\Phi} \cdot \Delta \underline{U} \quad (7)$$

Mit

$$\begin{aligned} \Delta \underline{U} &= [\Delta \underline{u}(k_i)^T \quad \Delta \underline{u}(k_i+1)^T \quad \dots \quad \Delta \underline{u}(k_i+2)^T \\ &\quad \Delta \underline{u}(k_i+N_c-1)^T]^T \\ \underline{Y} &= [\underline{y}(k_i+1)^T \quad \underline{y}(k_i+2)^T \quad \dots \quad \underline{y}(k_i+3)^T \\ &\quad \underline{y}(k_i+N_p)^T]^T \\ \underline{F} &= \begin{bmatrix} \underline{CA} \\ \underline{CA}^2 \\ \underline{CA}^3 \\ \vdots \\ \underline{CA}^{N_p} \end{bmatrix}; \\ \underline{\Phi} &= \begin{bmatrix} \underline{CB} & \mathbf{0} & \dots & \mathbf{0} \\ \underline{CAB} & \underline{CB} & \dots & \mathbf{0} \\ \underline{CA}^2 \underline{B} & \underline{CAB} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \underline{CA}^{N_p-1} \underline{B} & \underline{CA}^{N_p-2} \underline{B} & \dots & \underline{CA}^{N_p-N_c} \underline{B} \end{bmatrix} \end{aligned} \quad (8)$$

Das quadratische Gütemaß J wird auf Basis der vorgestellten Beschreibungsform des dynamischen Systems formuliert. Bei der Wahl eines quadratischen Ansatzes der Kostenfunktion kann sichergestellt werden, dass das

globale Minimum gefunden wird.

$$\underline{J} = (\underline{w} - \underline{Y})^T (\underline{w} - \underline{Y}) + \Delta \underline{U}^T \underline{R} \Delta \underline{U} \quad (9)$$

Mit \underline{w} als Sollwertvektor und \underline{R} zur Beeinflussung der Schnelligkeit des geschlossenen Regelkreises. Im ersten Teil des Gütemaßes wird die Abweichung von den Sollwerten, im zweiten Teil die Stellgrößenänderung gewichtet. Das Minimum durch partielles Ableitung nach der Stellgrößenänderung bestimmt:

$$\Delta \underline{U} = (\underline{\Phi}^T \underline{\Phi} + \underline{R})^{-1} \underline{\Phi}^T (\underline{w} - \underline{F} \underline{x}(k_i)) \quad (10)$$

Dieser Vorschrift entsprechend werden die Stellgrößenänderungen über den gesamten Kontrollhorizont N_c berechnet. Da die zugrundeliegende Modellbeschreibung nicht vollständig mit dem realen System übereinstimmt, wird in jedem Berechnungsschritt lediglich das erste Element des Vektors $\Delta \underline{u}$ als Stellgröße ausgegeben und die restlichen Elemente verworfen [7]:

$$\Delta \underline{u}(k_i) = [\mathbf{1} \ \mathbf{0} \ \dots \ \mathbf{0}]_{1 \times N_c} (\underline{\Phi}^T \underline{\Phi} + \underline{R})^{-1} \underline{\Phi}^T \cdot (\underline{w} - \underline{F} \underline{x}(k_i)) \quad (11)$$

Abbildung 3 stellt die Struktur des geschlossenen Regelkreises dar. Anhand dieser Struktur wird der aktuelle Stellgrößenvektor bestimmt.

$$\Delta \underline{u}(k_i) = \underline{K}_y \cdot \underline{w}(k_i) - \underline{K}_{MPC} \cdot \underline{x}(k_i) \quad (12)$$

Mit Vergleich von Gl. (1) und Gl. (12) werden die Zustandsrückführung \underline{K}_{MPC} und der Vorfilter \underline{K}_y bestimmt. Dieses Vorgehen wird im nächsten Berechnungsschritt wiederholt und als gleitender Horizont beschrieben. D.h. das Optimierungsproblem ist in einem beweglichen Zeithorizontfenster zu lösen.

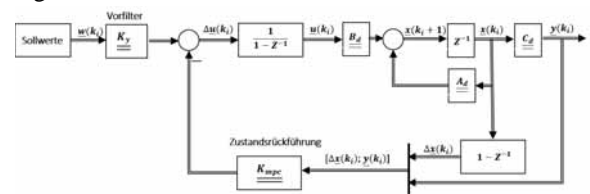


Abbildung 3. Geschlossener Regelkreis

Um die Position des S-Mobile zu regeln, werden der Zustands- und Ausgangsvektor des Zustandsmodells ermittelt und wie in folgender Gleichung dargestellt in den DMPC-Regelalgorithmus implementiert:

$$\underline{x} = [\Delta \theta_x \quad \Delta \dot{\theta}_x \quad \Delta \theta_y \quad \Delta \dot{\theta}_y \quad \Delta \theta_z \quad \Delta \dot{\theta}_z \quad \Delta \phi_x \quad \Delta \dot{\phi}_x \quad \Delta \phi_y \quad \Delta \dot{\phi}_y \quad \phi_x \quad \phi_y]^T \quad (13)$$

$$\underline{y} = [\phi_x \quad \phi_y]^T$$

Obwohl dieses neue Zustandsmodell vollständig

steuerbar ist, ist es lediglich durch Einstellung der Abstimmungsparameter der quadratischen Programmierung für die Bestimmung der Zustandsrückführung K_{MPC} das S-Mobile nur schwer zu stabilisierere.. Aus diesem Grund wird eine Kaskadenregelstruktur verwendet, wobei der innere Regelkreis als Aufbauwinkelregler zur Stabilisierung der Regelstrecke und der äußere Regelkreis als Spurfolgeregler entwickelt wird.

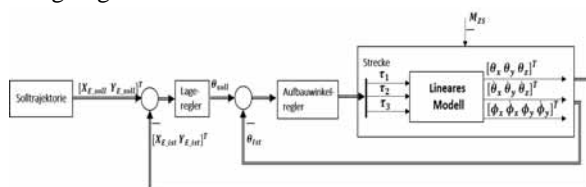


Abbildung 4. Kaskadenregelstruktur von S-Mobile

3.1 Aufbauwinkelreglung

Für die Reglerauslegung des inneren Regelkreises wird zunächst ein geeignetes zeitdiskretes Zustandsmodell mit dem folgenden Zustandsvektor als Grundlage aufgestellt.

$$\underline{x} = [\theta_x \ \dot{\theta}_x \ \theta_y \ \dot{\theta}_y \ \theta_z \ \dot{\theta}_z]^T \quad (14)$$

Nach der Diskretisierung mit gegebener Abtastzeit wird die innere Regelstrecke, deren Blockschaltbild in der folgenden Abbildung dargestellt wird, bestimmt.

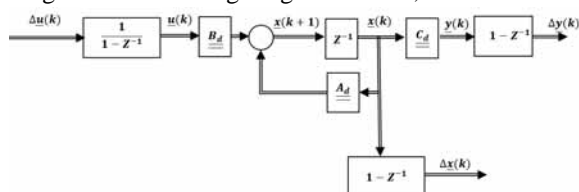


Abbildung 5. Blockschaltbild der inneren Regelstrecke

Der innere Regelkreis dient schwerpunktmäßig dazu, die Regelstrecke zu stabilisieren, während die stationäre Genauigkeit des inneren Regelkreises nicht im Fokus steht. Deshalb wird das zeitdiskrete Zustandsmodell ohne Erweiterung um die ausgewählten Ausgangsgrößen betrachtet.

Nach Wahl der entsprechenden Abstimmungsparameter wird die Zustandsrückführung des inneren Regelkreises bestimmt, wie folgend verdeutlicht:

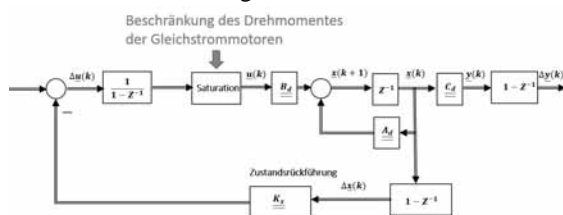


Abbildung 6. Blockschaltbild des inneren Regelkreises

3.2 Spurfolgeregelung

Nach der Synthese des inneren Regelkreises wird der komplette unterlagerte geschlossene Regelkreis in einem neuen Zustandsmodell zusammengefasst. Anschließend wird der Zustandsvektor unter Berücksichtigung der stationären Genauigkeit der Position um die Drehwinkel der Antriebskugel erweitert.

$$\begin{aligned} \underline{x}_{neu}(k+1) &= \underline{A}_{neu} \cdot \underline{x}_{neu}(k) + \underline{B}_{neu} \cdot \underline{\Delta u}_{neu}(k) \\ \underline{y}_{neu}(k) &= \underline{C}_{neu} \cdot \underline{x}_s(k) \\ \underline{\Delta u}_{neu} &= [\Delta\theta_x \ \Delta\theta_y \ \Delta\theta_z]^T \\ \underline{x}_{neu} &= [\Delta\theta_x \ \Delta\dot{\theta}_x \ \Delta\theta_y \ \Delta\dot{\theta}_y \ \Delta\theta_z \ \Delta\dot{\theta}_z \ \Delta\phi_x \ \Delta\phi_y \\ &\quad \Delta\phi_z \ \Delta\dot{\phi}_x \ \Delta\dot{\phi}_y \ \Delta\dot{\phi}_z \ \phi_x \ \phi_y \ \phi_z]^T \\ \underline{y}_{neu} &= [\phi_x \ \phi_y]^T \end{aligned} \quad (15)$$

Analog zur Bestimmung der Zustandsrückführung des inneren Regelkreises kann die Zustandsrückführung der Spurfolgeregelung mit den gewählten Abstimmungsparametern festgelegt werden.

Die Eigenwerte des gesamten geschlossenen überlagerten Regelkreises sind in der Abbildung 7 dargestellt. Aufgrund der Position dieser Eigenwerte in der S-Ebene ist das System stabil.

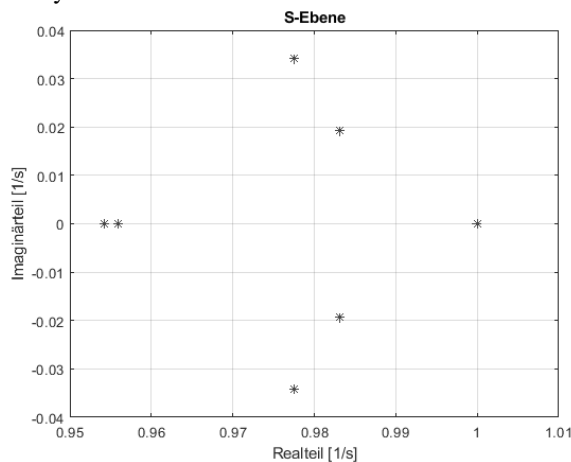


Abbildung 7. Eigenwerte des geschlossenen überlagerten Regelkreises

4 Funktionsabsicherung mittels MiL

Zur Funktionsabsicherung wird zunächst die Modell-in-the-Loop-Simulation herangezogen.

Ziel der Spurfolgeregelung ist primär, dass das S-Mobile stabil bleibt und gleichzeitig einer Solltrajektorie so schnell und exakt wie möglich folgt. Um dies zu Testen wurde eine polynomische Gleichung als Solltrajektorie

für die Bewegung in beide Richtungen auf den Regler gegeben. Abbildung 8 stellt das Simulationsergebnis dar.

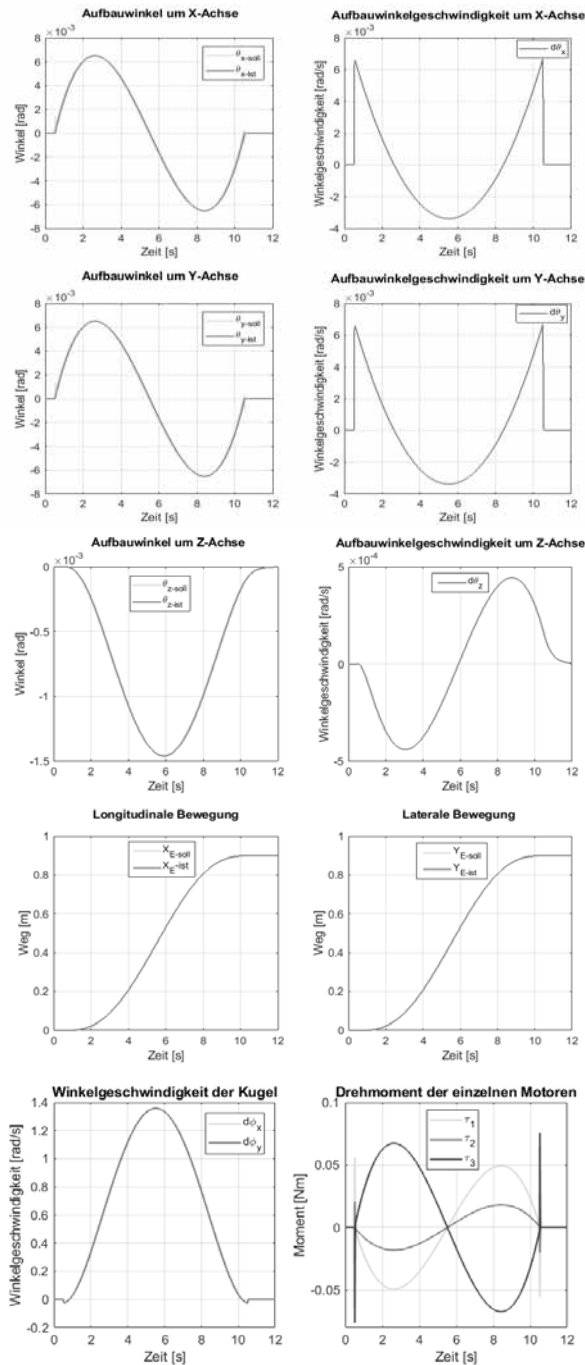


Abbildung 8. Simulationsergebnis

Im oberen Bereich des Ergebnisgraphen sind zunächst die aus dem Regelvorgang resultierenden Aufbauwinkel und Aufbauwinkelgeschwindigkeiten (rot) gegen die theoretisch optimalen Größen (grün) verglichen. Wie in Kapitel 3.1 erklärt werden die optimalen Größen als

Eingangsgrößen der unterlagerten Stabilisierungsregelung vom prädiktiven Spurfolgecontroller berechnet. Eine geringe Abweichung bedeutet in diesem Kontext also ein gutes Folgeverhalten.

Im unteren Bereich des Graphen sind die Soll- und Istwerte der longitudinalen und lateralen Bewegung aufgezeichnet. Die Sollwerte repräsentieren dabei den Gesamtsystemeingang, sind also die zu folgende Trajektorie. Hier ist ersichtlich, dass der Regler der Solltrajektorie mit geringer Verzögerung folgt. Zudem zeigt der darunter dargestellte zeitliche Verlauf des Drehmoments, dass das System während des Spurfolgevorgangs in den definierten Beschränkungen stabilisiert werden kann.

5 Resümee

In dieser Arbeit wurde ein zeitdiskreter modellprädiktiver Regler anhand eines linearen Zustandsraummodells als Näherung des dynamischen Verhaltens des S-Mobile entwickelt. Es wurde eine Kaskadenregelstruktur, in welcher der innere Regler zum Stabilisieren des unteraktuierten Systems und der äußere Regler zum Spurfolgen dient, entwickelt. Zuerst wurde dieses Modell um für die Regelung notwendige Ausgangsgrößen erweitert und für die spätere Ausführung auf Target Hardware in fester Schrittweite diskretisiert. Der diskrete Modellprädiktive Regelalgorithmus kann mit dessen Hilfe in einem beweglichen Ereignishorizont zukünftige Stellgrößen vorhersagen. Dies wurde mittels einer Modell-in-the-Loop Simulation anhand eines Referenzmodells des S-Mobile validiert.

Im weiteren Verlauf der Arbeit soll eine Erweiterung des Algorithmus unter Berücksichtigung der Nichtlinearität der Regelstrecke vollzogen werden. Dieser basiert auf der Methode der exakten Linearisierung und auf der Darstellung der Systemdynamik mittels Jacobi Matrizen, welche an verschiedenen Arbeitspunkten des Systems linearisiert werden um so einen Zusammenhang zwischen den Regelparametern und den nichtlinearen Zustandsgrößen der Strecken zu finden.

Weiterhin wird ein Forschungsträger aufgebaut an dem umfangreiche Hardware-in-the-Loop Test zur Verifizierung des Regelsystems durchgeführt werden sollen. Besonders Interessant sind Wechselwirkungen der Nichtlinearitäten mit dem linearen Regelsystem und der Gültigkeitsbereich der optimierten Parametersätze.

References

- [1] W. Schiehlen, P. Eberhard. *Technische Dynamik – Modelle für Regelung und Simulation*. Springer, 2006
- [2] A. Jezierski, J. Mozaryn, D. Suski. *A Comparison of LQR and MPC Control Algorithms of an Inverted Pendulum*. In: Mitkowski, W. et al: Trends in Advanced Intelligent Control, Optimization and Automation. Advances in Intelligent Systems and Computing, vol 577. Springer, Cham, 2017
- [3] X. Liu-Henke, M.Göllner, H. Tao. *Konzeption eines hochdynamischen Systems mit sphärischen Elektroantrieb*. Workshop der ASIM7GI-Fachgruppen, Reutlingen, Februar 20-21, 2014
- [4] R. Buchta. *Mechatronische Entwicklung eines Forschungselektrofahrzeugs zur Erprobung von Fahrdynamikregelungen und Fahrerassistenzsystemen*. Dissertation, Universität Magdeburg, 2016
- [5] H. Tao. „Entwurf des dreidimensionalen Streckenmodells des Sphere-Mobile und Auslegung eines optimalen Zustandsreglers“, Hochschule Ostfalia, Wolfenbüttel, 2016
- [6] L. Wang. *Model Predictive Control System Design and Implementation Using MATLAB*. Springer, 2009
- [7] W.H. Kwon, S. Han. *Receding Horizon Control-model predictive control for state models*. Springer, 2005.
- [8] X. Liu-Henke. *Fahrdynamikregelung*, Vorlesungsskript zur Vorlesung Fahrdynamikregelung, Hochschule Ostfalia, Wolfenbüttel, 2016

Simulation und Kompensation der Eigenspannungen in der additiven Fertigung

Tobias Mussehl, B.Eng., Martin Rambke, Prof. Dr.-Ing.

Ostfalia Hochschule für angewandte Wissenschaften
Fakultät Maschinenbau, Institut für Produktionstechnik
Salzdahlumer Straße 46/48, 38302 Wolfenbüttel

Abstract. Dieser Artikel befasst sich mit der Untersuchung der numerischen Simulation des Metall-Lasersinterns (SLM) mit mechanischen sowie thermomechanischen Berechnungsansätzen. Die Simulationsergebnisse ermöglichen eine Voraussage des Verzugs und darauf basierend eine Verzugskompensation. Mit Hilfe von Realversuchen wird die Ergebnisgenauigkeit bewertet.

This article deals with the investigation of the numerical simulation of Selective Laser Melting (SLM) with mechanical and thermomechanical calculation approaches. The simulation results allow a prediction of the distortion and based on this a distortion compensation. The accuracy of the results is evaluated by means of real tests.

Einleitung

An der Ostfalia Hochschule studieren zurzeit ca. 12.400 Studierende in 12 Fakultäten an vier Standorten (Salzgitter, Suderburg, Wolfenbüttel und Wolfsburg). Am Institut für Produktionstechnik (IPT) der Fakultät Maschinenbau sind momentan sieben Professorinnen und Professoren sowie 16 wissenschaftliche Mitarbeiter in der Lehre und Forschung tätig. Vor vier Jahren wurde das Fabrication Laboratory (www.FabLab38.de) initiiert, aus dem in 2017 - unter Beteiligung weiterer Fakultäten - die Gründung des Zentrums für additive Fertigung (ZaF) erfolgte. Auf mittlerweile 25 „3D Druckern“ werden studentische Projekte und Forschungsarbeiten in diversen Verfahren (FDM, SLA, SLS, Polyjet etc.) realisiert. Ebenfalls im Jahre 2017 konnte im Rahmen einer EFRE Infrastrukturmaßnahme eine Renishaw AM400 beantragt und beschafft werden, die eine Fertigung mit dem Metall-Lasersintern (SLM) ermöglicht [1].

1 Zielsetzung

Der am IPT verfolgte Forschungsansatz beinhaltet die virtuelle Entwicklung von Werkzeugen mit konturnahen Kühlkanälen für das so genannte Presshärten und die anschließende reale Fertigung mittels SLM (Rapid Tooling). Dabei treten folgende Fragestellungen auf:

- Welches Material (Werkzeugstahl) ist verfügbar und zielführend einsetzbar?
- Welche Kanalquerschnitte (Form und Größe) und welche zur Herstellung erforderlichen Stützstrukturen sollten ausgewählt werden?
- Welches Gesamtpotenzial gibt es hinsichtlich Werkzeuggewicht und Kühlleistung?

Für die Implementierung eines virtuellen Auslegungsprozesses soll hier zunächst die Verzugsoptimierung mit Simufact Additive untersucht und validiert werden. Hierbei wurden sowohl ein mechanischer als auch ein thermomechanischer Berechnungsansatz verwendet und Unterschiede in der Anwendung sowie die erreichbaren Genauigkeiten untersucht.

2 Kalibrierung der Simulationssoftware

Für den mechanischen Berechnungsansatz muss zunächst eine Kalibrierung der Simulationssoftware vorgenommen werden. Die dazu verwendeten Cantilever müssen unter den gleichen Fertigungsbedingungen hergestellt werden, wie die später zu fertigenden Bauteile, da die Fertigungsparameter (Material: AlSi10Mg, Schichtdicke: 30 µm, Laserleistung: 400W und Vorheiztemperatur der Bauplattform: 20°C sowie 170°C) die entstehenden Eigenspannungen und damit die Verzüge der Bauteile beeinflussen.

Nach der realen Fertigung der Cantilever wird eine Messung der Bauteilhöhe durchgeführt. Anschließend wird

ein Schnitt in einer Höhe von 2,9 mm oberhalb der Substratplattform durch die Verbindungsstege des Bauteils vorgenommen. Die im Bauteil vorhandenen Eigenspannungen werden durch diesen Schnitt freigesetzt und führen zum Verzug. Durch den Verzug entsteht eine Verschiebung des Messpunkts in z-Richtung und damit eine neue Bauteilhöhe die ebenfalls gemessen wird (Abb. 1).

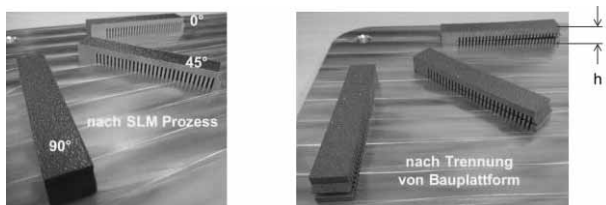


Abb. 1: Cantilever für die Kalibrierung [3]

Die ermittelte Höhenänderung wird in den Kalibrierungseinstellungen der Simulation hinterlegt und über einen internen Algorithmus ein entsprechender Parametersatz berechnet (Abb. 2). Für die Kalibrierung ist die Verwendung von zwei Cantilevern (0° und 90°) ausreichend [3].

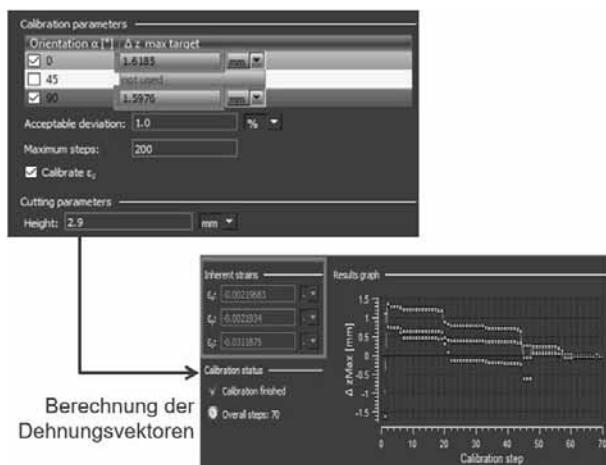


Abb. 2: Eingabe der Höhenänderung und Berechnung der Dehnungsvektoren

Im Gegensatz zum mechanischen Berechnungsansatz wird für die thermomechanische Simulation keine Kalibrierung auf Basis vordefinierter Proben durchgeführt. Die Einstellung des Temperaturfeldes erfolgt über eine thermische Simulation der Bauteile und der Bauplattform. Der thermomechanische Ansatz basiert auf Materialkennwerten und den thermischen Eigenschaften des eingesetzten Pulvermaterials und der Bauplattform, welche aus dem gleichen Material bestehen muss wie das eingesetzte Pulver. Zu den thermomechanischen Eigenschaften

zählen der Emissionsgrad und der Wärmeübergangskoeffizient. Der Emissionsgrad (emissivity) beschreibt den strahlungsbedingten Wärmeverlust des Materials während des Fertigungsprozesses. Der Wärmeübergangskoeffizient (heat transfer coefficient) ist ein materialabhängiger Parameter, welcher die Intensität des Wärmeübergangs zwischen aufgeschmolzenem Material und Pulver ausdrückt. Dieser Parameter ist sehr stark von der Partikelgrößenverteilung im Pulver und der Zusammensetzung des Pulvers abhängig. Da Streuungen in der Zusammensetzung und der Größenverteilung auftreten ist dieser Wert in der Realität nicht konstant, wurde in der Simulation allerdings als konstant angenommen. Die Belichtungsdauer (exposure time) und der Energieanteil der Belichtungsdauer (exposure energy fraction) werden mit einer thermischen Untersuchung ermittelt, wobei darauf zu achten ist, eine realitätsnahe Abbildung des Temperaturfeldes zu erreichen (Abb. 3). Aus diesem Grund wird in der thermischen Untersuchung ausschließlich das entstehende Temperaturfeld simuliert. Unter Berücksichtigung der Materialkennwerte des verwendeten Pulvers (AlSi10Mg) und der Bauplattform wird auf Grundlage des Simulationsergebnisses der thermischen Simulation die thermomechanische Simulation durchgeführt. [4]

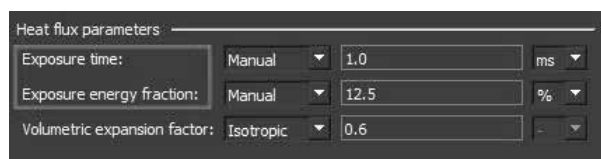


Abb. 3: Einstellung der thermischen Parameter

3 Konstruktion des Versuchsbauteils

Für die geplante Validierung der Verzugkompensation wurde ein Versuchsbauteil entwickelt. Die Konstruktionskriterien waren

- eine hohe Sensitivität hinsichtlich Verzug und Wärmeeinfluss,
- die Spannmöglichkeit für eine Nachbearbeitung und
- eine gute Messbarkeit vor und nach dem Entfernen der Stützstruktur.

Das Bauteil lässt sich in die drei Bestandteile Fuß, Mittelstück und Kopf unterteilen (Abb. 4). Charakteristisch für den Fuß ist die massivere Auslegung im Vergleich zum restlichen Bauteil. Er ist rotationssymmetrisch, um

die Nachbearbeitung auf einer Drehmaschine zu ermöglichen. An den Fuß grenzt das Mittelstück, welches dünnwandig und geschlitzt ausgelegt wurde, um in diesem Bereich gezielt eine Verformung des Bauteils zu erreichen und damit die auftretenden Eigenspannungen in Form von Verzügen sichtbar zu machen. Der mittlere Teil wurde ebenfalls rund ausgeführt, um auch hier eine leichte Entfernung der Stützstrukturen realisieren zu können. Eine Vielfalt an Prüfmerkmalen ermöglicht der als Oktagon ausgelegte Kopfteil [3].



Abb. 4: Versuchsbauteil für die Validierung der Simulationssoftware

4 Stützstrukturgenerierung für das Versuchsbauteil

Die Stützstruktur wird benötigt, um das Bauteil vor Verformungen während des Fertigungsprozesses zu schützen. Dabei wird die Stützstruktur zusammen mit dem Bauteil schichtweise aufgebaut. Die Stützstruktur dient darüber hinaus zur Wärmeabfuhr der eingebrachten thermischen Energie, um die Eigenspannungen innerhalb des Bauteils zu reduzieren und ungewollte Verformung nach der Herstellung (Verzüge) zu minimieren. Für die Erstellung der Stützstruktur wurde zum einen die Funktionalität der Simulationssoftware Simufact Additive genutzt, zum anderen wurden zwei Stützstrukturen mit Hilfe der Software Materialise Magics erzeugt (Abb. 5).

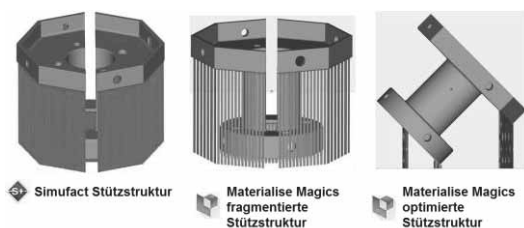


Abb. 5: Erzeugte Stützstrukturen [3]

Mit Hilfe von „Materialise Magics“ lassen sich Stützstrukturen erstellen und individuell anpassen, wobei eine Vielzahl von Strukturelementen zur Verfügung steht. Für das Versuchsbauteil wurde zum einen der Volumensupport verwendet und mit einer Fragmentierung versehen (Abb. 5 Mitte). Die Fragmentierung hat den Vorteil, dass weniger Material verwendet wird und die

Stützstruktur nach der Herstellung einfacher zu entfernen ist, als ein kompletter Volumensupport. Die Herstellungsdauer eines Bauteils im SLM-Verfahren beträgt oft mehrere Tage. Aus diesem Grund ist eine optimale Ausrichtung für das Bauteil besonders wichtig. Durch diese Optimierungen können mehrere Stunden Bearbeitungszeit eingespart werden, da weniger Material innerhalb einer Schicht aufgeschmolzen werden muss. Bei dem verwendeten Material AlSi10Mg ist es möglich, Bauteilbereiche bis zu einem Winkel von 45 Grad ohne Stützstruktur zu erzeugen. Als zweite Variante wurden daher einzelne Bauteilkanten mit einem Liniensupport an der Bauplattform fixiert. Im Vergleich zur fragmentierten Stützstruktur ist eine deutliche Reduzierung des Supportvolumens zu erkennen (Abb. 5 rechts), sodass das Bauteil schneller und kostengünstiger hergestellt werden kann.

5 Simulation und Analyse der Ergebnisse

Neben der Kalibrierung der Simulationssoftware sind weitere Einstellungen vor der Simulationsdurchführung vorzunehmen. Zuerst wird der Simulationsumfang festgelegt und die Maschine ausgewählt, auf der das Bauteil erzeugt werden soll, um eine Übereinstimmung der Bauplattformabmessungen zu gewährleisten. Die Bauteil deformation ergibt sich aus der Simulation des Fertigungsprozesses und dem simulierten Entfernen der Bauplattform sowie der Stützstrukturen. Abb. 6 verdeutlicht diesen Einrichtungsschritt der Simulation. Für den mechanischen Berechnungsansatz wird anschließend der zuvor generierte Kalibrierungssatz ausgewählt, um die Verzüge anhand der Verzugsvektoren zu berechnen. Für die thermomechanische Simulation ist lediglich die Eingabe der materialspezifischen Kennwerte vorzunehmen.

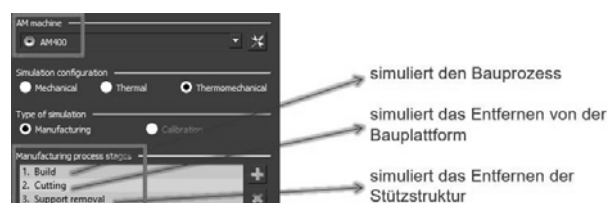


Abb. 6: Auswahl der Fertigungsanlage und des Simulationsumfangs

Zur Berechnung der einzelnen Bauteilschichten wird ein Netz aus Dreiecken über die Oberfläche des Bauteils gelegt. Die Elemente sollten nicht zu groß gewählt werden, da das Simulationsergebnis sonst ungenau werden kann. Zu kleine Elemente erhöhen hingegen die Berechnungsdauer. Anschließend wird ein weiteres Netz erstellt, das aus gleich großen Volumenelementen (Voxel) besteht (Abb. 7).

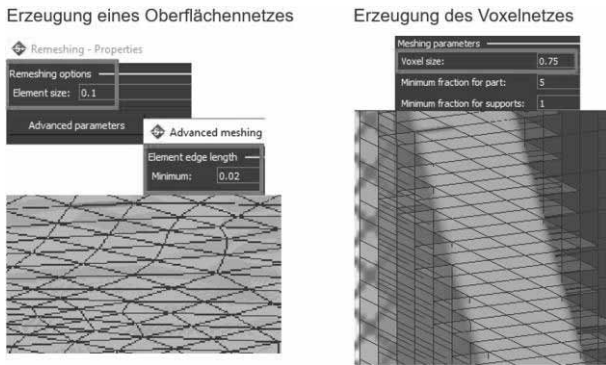


Abb. 7: Erzeugung eines Oberflächen- und eines Voxelnetzes für die Berechnung des Bauteilverzugs [3]

Die Volumenelemente füllen das gesamte Bauteil aus und sind für die Berechnung der Bauteilverformung notwendig. Die Voxelabmaße senkrecht zur Oberfläche sollten kleiner sein als die geringste Wandstärke des Bauteils, um die Verformung filigraner Bauteilelemente genau berechnen zu können. Während der Simulation berechnet die Software schichtweise die Verformung des Bauteils. Die Anzahl der Schichten ergibt sich dabei aus der Bauteilgröße, der Höhe der Stützstruktur unterhalb des Bauteils und der Größe der Volumenelemente.

Nachdem die Simulation abgeschlossen ist, wird das nach dem Fertigungsprozess zu erwartende Bauteil dargestellt. Anhand des simulierten Bauteils lassen sich die Verzüge erkennen. Abb. 8 zeigt die Sollgeometrie des Versuchsteils (blau), die Geometrie inkl. Verzug nach der Herstellung (rot) und das verzugskompensierte Bauteil (gelb).



Abb. 8: Simulations- und Kompensationsergebnis [3]

Die Verzugskompensation kann auf zwei verschiedene Arten erfolgen. Zum einen besteht die Möglichkeit einen Skalierungsfaktor einzustellen (z. B. -0,9), mit dem der negative Geometrie-Offset durchgeführt wird. Die zweite Methode besteht in der Optimierung des Skalierungsfaktors, bis der Verzug und damit die Abweichung zwischen simulierter Bauteilgeometrie und der Sollkontur einen definierten Grenzwert unterschreitet (Abb. 9).

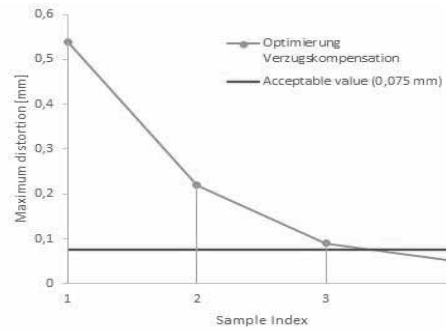


Abb. 9: Optimierung der Verzugskompensation [4]

6 Realversuche und Validierung

Die Fertigung aller Bauteile erfolgte aus AlSi10Mg-Pulver mit einer Schichtdicke von 30 µm und einer Laserleistung von 400 W. Variiert wurde die Vorheiztemperatur der Bauplattform (20°C ❄️ und 170°C 🏠). Das Bauteil auf Basis der Simufact Stützstruktur konnte nicht zerstörungsfrei von der Stützstruktur befreit werden (Abb. 10 unten links) und schied daher für die weitere Betrachtung aus.

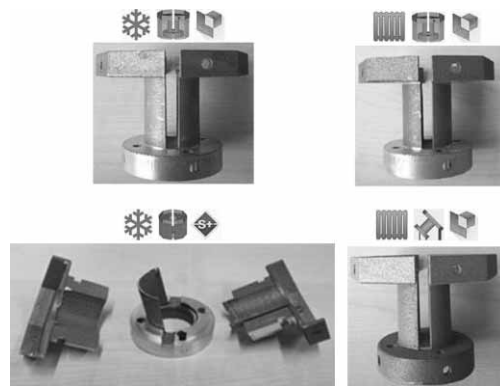


Abb. 10: Mit unterschiedlichen Stützstrukturen und Vorheiztemperaturen gefertigte Bauteile

Um einen Vergleichswert für die Validierung zu erhalten, wurden zunächst nicht verzugskompensierte Bauteile gedruckt und ausgewählte Geometriegrößen vor dem Entfernen der Stützstruktur gemessen (Abb. 11).

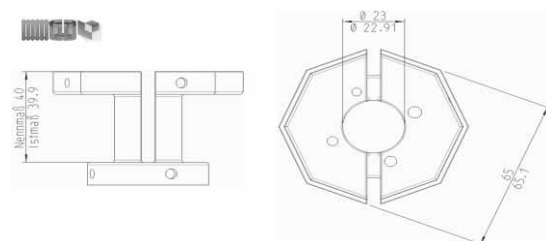


Abb. 11: Nenn- und Istmaße (nicht verzugskompensiert) vor dem Entfernen der Stützstruktur

Ziel war die Ermittlung der Streuung des Fertigungsverfahrens als Basis für die theoretisch erreichbaren Toleranzen (Abb. 12).

Nennmaß	Messwert	Messwert	Messwert	Toleranzbereiche
40	39,90	40,14	40,08	$\pm 0,15$
$\varnothing 23$	$\varnothing 22,91$	$\varnothing 23,03$	$\varnothing 22,92$	$\pm 0,10$
65	65,10	65,14	64,97	$\pm 0,15$

Abb. 12: Abgeleitete Toleranzbereiche aus der Streuung der Messwerte [3]

Im Anschluss erfolgte die Vermessung nach dem Entfernen der Stützstruktur und die Fertigung der verzugs-kompensierten Bauteile (mechanischer Berechnungsansatz). Als Grundlage für die Ermittlung der Maßhaltigkeit dienten die zuvor ermittelten Toleranzbereiche, um einen Abgleich zwischen Simulation und Versuch durchführen zu können (Abb. 13).

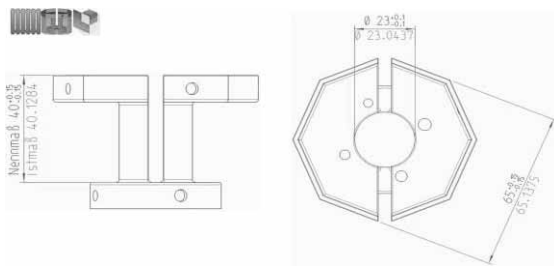


Abb. 13: Gegenüberstellung der Nennmaße inkl. der Toleranzbereiche und den Istmaßen (verzugs-kompensiertes Bauteil)

Abb. 14 zeigt beispielhaft eine Auswertung, in der die Maßabweichungen der verzugs-kompensierten Geometrien von der Sollkontur dargestellt sind. Die zuvor ermittelte Streuung des Fertigungsprozesses findet sich hier als Gutteilfenster wieder.

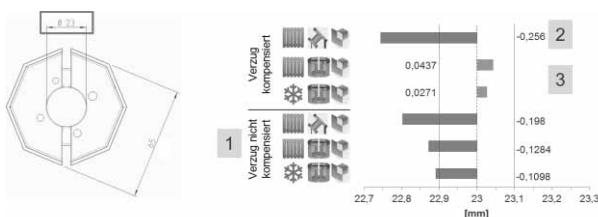


Abb. 14: Beispielhafte Auswertung der Messung (mechanischer Berechnungsansatz) [3]

Hier ist deutlich erkennbar, dass ohne Verzugskompensation die Toleranzen überschritten werden (Abb. 14, Bereich 1). Im Bereich 2 findet trotz Kompensation eine Überschreitung der Toleranzen statt. Offensichtlich ist

die Wärmeabfuhr bei Verwendung der so genannten optimierten Stützstruktur viel zu gering, sodass zu viele Eigenspannungen im Bauteil verbleiben, die in der mechanischen Berechnung nicht berücksichtigt werden. Bei Verwendung der fragmentierten Stützstruktur (Abb. 14, Bereich 3) wird mit und ohne Vorheizung eine deutliche Verbesserung der Maßhaltigkeit erzielt und das Gutteilfenster erreicht.

Für die Untersuchung des thermomechanischen Berechnungsansatzes wurde das um 45° verkippte Bauteil verwendet (Abb. 15), da hier die Vorhersagegenauigkeiten bei Verwendung des mechanischen Berechnungsansatzes am geringsten waren. Durch den thermomechanischen Ansatz wurde eine Verbesserung der Simulationsergebnisse erwartet, da hier die thermischen Einflüsse innerhalb des Fertigungsprozesses berücksichtigt werden. [4]

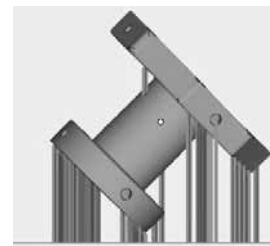


Abb. 15: 45° verkipptes Bauteil mit Stützstruktur

Abb. 16 zeigt beispielhaft eine Auswertung, in der die Maßabweichungen der verzugs-kompensierten Geometrie von der Sollkontur dargestellt sind. Verglichen werden hier Ergebnisse auf Basis des mechanischen Ansatzes mit den Ergebnissen auf Basis des thermomechanischen Ansatzes (mit fest eingestellter Offsetting, sowie mit automatischer Optimierung des Offsettingfaktors). Der thermomechanische Berechnungsansatz konnte die Erwartungen hinsichtlich der Bauteilgenauigkeit durch Verzugskompensation nicht ganz erfüllen:

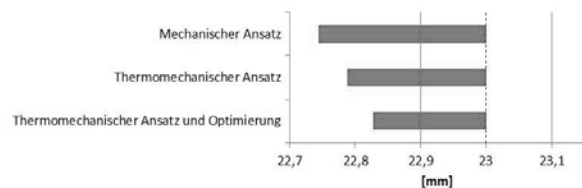


Abb. 16: Beispielhafte Validierung der Berechnungsansätze [4]

Die Maßhaltigkeit ließ sich im Vergleich zur Simulation mit mechanischem Berechnungsansatz zwar verbessern, jedoch liegen die Abweichungen weiterhin oberhalb von 0,1 mm. Die Vermutung, dass die Abweichungen des mechanischen Berechnungsansatzes allein auf die Nichtberücksichtigung der geringen Wärmeabfuhr zurückzuführen ist ließ sich nicht bestätigen. Vielmehr wurde

deutlich, dass der Bauteilverzug von weiteren Faktoren abhängig ist und diese in der Simulation berücksichtigt werden müssten. Zu diesen Faktoren zählen die Anordnung der Bauteile, sowie die Beeinflussung durch umliegende Objekte. In Abb. 17 ist die Auswertung der Maßabweichung unter Berücksichtigung der Bauteilposition auf der Bauplattform (ϕ) und dem Einfluss durch umliegende Objekte (ϕ_{obj}) dargestellt.

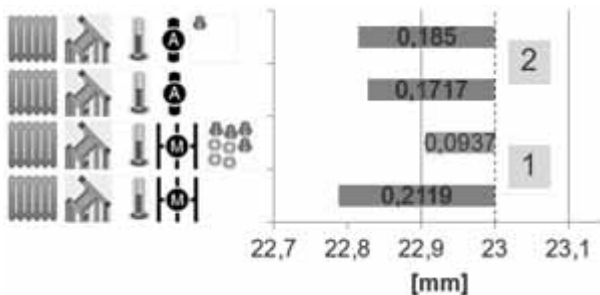


Abb. 17: Beispielhafte Auswertung von Maßabweichungen, ϕ - automatische Optimierung des Skalierungsfaktors, ϕ_{obj} - fest eingestellter Skalierungsfaktor

Es ist deutlich zu erkennen, dass die Berücksichtigung von umliegenden Objekten zu einer Verbesserung der Maßhaltigkeit führt im Vergleich zur Simulation des einzelnen Bauteils auf der Bauplattform (Abb. 17, Bereich 1). Im Bereich 2 findet trotz der Berücksichtigung der Bauteilposition keine Verbesserung der Maßhaltigkeit statt. Es ist davon auszugehen, dass die Position auf der Bauplattform keinen Einfluss hat, vielmehr ist der Einfluss durch weitere Objekte auf der Bauplattform für die Maßhaltigkeit entscheidend. Für eine Validierung der Simulationssoftware sind daher weitere Untersuchungen vorzunehmen, damit eine verlässliche Vorhersage der Verzüge realisiert werden kann.

7 Zusammenfassung und Ausblick

Der rein mechanische Berechnungsansatz in Simufact Additive ist vor allem für Bauteile mit einer massiv ausgelegten Stützstruktur geeignet. Die Wärmeabfuhr erfolgt bei diesen Bauteilen über die größeren Stützstrukturquerschnitte, wodurch der thermische Einfluss aufgrund von Festkörper-Pulver-Übergängen sehr gering ist. Außerdem ist die Berechnungszeit des mechanischen Ansatzes deutlich geringer als die des thermomechanischen. Geringe Mengen an Stützstrukturen besitzen hinsichtlich Kosten- und Zeitaufwand bei der additiven Fertigung Vorteile. Daher ist der thermomechanische Be-

rechnungsansatz vor allem anzuwenden, wenn die Fertigungskosten durch den Einsatz filigraner Stützstrukturen gesenkt werden sollen.

Beide Berechnungsmethoden erzeugen verzugskompensierte Bauteile, die nach wie vor Abweichungen von der Sollgeometrie aufweisen. Für den thermomechanischen Berechnungsansatz ist seit der neusten Softwareversion ebenfalls eine Kalibrierung möglich. Die Kalibrierung teilt sich hierzu in zwei Abschnitte, wobei zuerst in einer thermischen Kalibrierung der Energieanteil der Belichtungsdauer ermittelt wird und anschließend in einer thermomechanischen Kalibrierung der volumetrische Expansionsfaktor (volumetric expansion factor). Mit Hilfe der Kalibrierung des thermomechanischen Berechnungsansatzes ist eine weitere Verbesserung der Maßhaltigkeit zu erwarten, Untersuchungen hierzu sind im nächsten Untersuchungszeitraum geplant. Darüber hinaus ist mit dem Update auf Simufact Additive 4.0 die automatische Optimierung des Skalierungsfaktors erweitert worden, sodass die Berücksichtigung von umliegenden Objekten auch in diesem Fall möglich ist. Da die automatische Optimierung bereits zu einer Verbesserung der Maßhaltigkeit ohne Berücksichtigung der umliegenden Bauteile geführt hat, ist zu erwarten, dass eine weitere Verbesserung erzielt werden kann. Grundsätzlich diskutiert werden sollte auch, ob ein „globaler“ Skalierungsfaktor für die Verzugskompensation an einer kompletten Bauteilgeometrie optimal ist.

Entsprechend den hier dargestellten Ausführungen werden weitere Untersuchungen notwendig sein, um die Vorhersagegenauigkeit der Simulation und damit die Qualität der Verzugskompensation zu erhöhen.

References

- [1] www.ostfalia.de/cms/de/zaf/detail/news/9d18688e-43a4-11e8-a117-d96edd3be9f9 Abrufdatum: 22.08.2018.
- [2] Braess, H.-H.; Seiffert, U.; Vieweg Handbuch Kraftfahrzeugtechnik; Verlag: Vieweg (2012).
- [3] Mussehl, T.; Rambke, M.: Rapid-Tooling Ansätze mit dem Metall-Lasersintern - erste Ergebnisse, 19. Round Table, Simulation Manufacturing, Tagungsband, Marburg, Mai 16-17, (2018).
- [4] Mussehl, T.; Rambke, R.: Virtualisierung additive Fertigungsprozesse für das Rapid Tooling, 25. Interdisziplinäre Wissenschaftliche Konferenz Mittweida, Tagungsband, Mittweida, Oktober 24-25, (2018).

Expansion of Models for Heart Rate Variability beyond the Autonomic Nervous System

Jennifer Straub^{1*}, Martin Bachler²

¹Inst. for Analysis and Scientific Computing, Vienna University of Technology, Wiedner Hauptstraße 8-10, 1040 Vienna, Austria; *jennifer.straub@tuwien.ac.at

²AIT Austrian Institute of Technology GmbH, Center for Health & Bioresources, Biomedical Systems, Giefinggasse 4, 1020 Vienna, Austria;

Abstract. According to the WHO, diseases of the cardiovascular system are currently the main cause of death in high, middle- and low-income countries. Therefore, their understanding, prediction, and prevention with the help of non-invasive, cost effective, and quick methods is of great interest.

Analysis of the heart rate and its change over time can give valuable insight into the health status of a patient, and is easily derived from electrocardiogram data. Reduced heart rate variability (HRV) is associated with an increased probability of death after myocardial infarction, indicates inflammatory processes in the body, and it is symptomatic of mental disorders such as depression and burn-out. Modeling and simulation of HRV can provide new insight into the nonlinear interplay of cardiovascular regulation.

In this work three models for HRV are introduced and compared. They include the firing rate of the baroreceptors, activity of the sympathetic and parasympathetic nervous system, stroke volume, cardiac noradrenaline and acetylcholine concentration, and an arterial windkessel. An existing model for HRV based on respiration and baroreflex activity was implemented and analyzed. In order to improve the model performance, sympathetic activity as well as the pressure curve in the aortic arch and the duration of the systole were adapted, resulting in a second model. A third model, including three different types of baroreceptors and a dependence on the mean arterial pressure was implemented as well.

The models were realized in Simulink 8.5. Validation is performed based on two 5 minute electrocardiogram recordings from 30 subjects and the simulation results were compared to subject data based on standard HRV measures.

First results are promising, as they exhibit the nonlinear and complex modulation of HRV and provide basis for further extension for HRV models, paving the way for the future usage of model prediction in the field of cardiovascular diseases.

Semantisches Matching für die Konfiguration von Komponenten in Cyber-physischen Systemen

Daning Wang, Christoph Knieke, Sebastian Lawrenz, Andreas Rausch

Institut für Software and Systems Engineering, Technische Universität Clausthal, Arnold-Sommerfeld-Straße 1, 38678 Clausthal-Zellerfeld, Germany; daning.wang@tu-clausthal.de, christoph.knieke@tu-clausthal.de, sebastian.lawrenz@tu-clausthal.de, andreas.rausch@tu-clausthal.de

Zusammenfassung. Für die Weiterentwicklung eines Cyber-physischen Systems (CPS) in interdisziplinären Projekten treffen oftmals unterschiedliche Modellierungsansätze aufeinander. Insbesondere prozessorientierte Ansätze zur Beschreibung des Ablaufs des Gesamtsystems stehen den komponentenorientierten Ansätzen zur Realisierung des Systems gegenüber. Die Herausforderung besteht dabei in der gemeinsamen Entwicklung und Simulation auf Basis der unterschiedlichen Modellierungsansätze.

In diesem Papier wird ein Ansatz beschrieben, wie komponenten- und prozessorientierte Ansätze gemeinsam weiterentwickelt werden können, indem beide Modellierungsansätze auf ein gemeinsames graph-basiertes Domänenmodell abstrahiert werden. Mit Hilfe des graph-basierten Domänenmodells und graph-basierter Algorithmen kann eine optimale Architektur für die Integration neuer Komponenten (Software- und/oder Hardwarekomponenten) in ein bestehendes Modell ermittelt werden. Darüber hinaus hilft das Verfahren bei der Auswahl von kompatiblen Komponenten, indem diese semantisch charakterisiert werden. Dabei werden die Modelle der Komponenten mithilfe von Metadaten soweit beschrieben, dass ein Algorithmus nach eigenen Kombinationsregeln diese verstehen und klassifizieren kann.

Schlüsselworte: Semantisches Matching, Konfiguration von Komponenten, Cyber-physische Systeme

Einleitung

Cyber-physische Systeme (kurz CPS) sind charakterisiert durch eine Verknüpfung von realen (physischen) Objekten und Prozessen mit informationsverarbeitenden (virtuellen) Objekten und Prozessen über offene, teilweise globale und jederzeit miteinander verbundene Informationsnetze [1]. Komplexe CPS werden in der Regel interdisziplinär entwickelt und aufgebaut. Für die

Weiterentwicklung eines CPS in interdisziplinären Projekten treffen oftmals unterschiedliche Modellierungsansätze aufeinander. Insbesondere prozessorientierte Ansätze zur Ablaufbeschreibung des Gesamtsystems stehen den komponentenorientierten Ansätzen zur Realisierung des Systems gegenüber. Die Herausforderung besteht dabei in der gemeinsamen Entwicklung und Simulation auf Basis der unterschiedlichen Modellierungsansätze.

In diesem Papier wird ein Ansatz beschrieben, wie komponenten- und prozessorientierte Ansätze gemeinsam weiterentwickelt werden können, indem beide Modellierungsansätze auf ein gemeinsames graph-basiertes Domänenmodell abstrahiert werden [2]. Diese Abstrahierung beinhaltet die Systemstruktur und Eigenschaften der ursprünglichen Modelle. Anschließend werden graph-basierten Algorithmen auf die Abstrahierung angewendet, um die gemeinsame Weiterentwicklung in den unterschiedlichen Modellierungsansätzen zu ermöglichen.

Das Ziel der Weiterentwicklung besteht darin, neue Komponenten (Software- und/oder Hardwarekomponenten) in ein bestehendes Modell zu integrieren, um neue Funktionen zu realisieren. Außerdem soll bestimmt werden, an welcher Stelle im Gesamtmodell die neuen Komponenten integriert werden. Der vorgestellte Ansatz beschreibt die möglichen neuen Konfigurationen des Gesamtsystems. Eine Konfiguration bezeichnet hierbei eine bestimmte Vernetzung der neuen und vorhandenen Komponente(n) in einen Prozess. Darüber hinaus hilft das Verfahren bei der Auswahl von kompatiblen Komponenten, indem diese semantisch charakterisiert werden [3]. Dies bedeutet, dass die Modelle der Komponenten mithilfe von Metadaten soweit beschrieben werden, dass ein Algorithmus nach eigenen Kombinationsregeln diese verstehen und klassifizieren kann.

Mithilfe des vorgestellten Konzeptes werden Lösungsmöglichkeiten für das Problem der Weiterentwicklung komplexer CPS aufgezeigt. Dazu wird ein Ansatz für die Integration von neuen Komponenten in ein bestehendes CPS vorgestellt.

Das Papier ist folgendermaßen gegliedert: Im nachfolgenden Kapitel wird zunächst ein Anwendungsszenario eingeführt. Kapitel 2 erörtert den Stand der Technik. In Kapitel 3 werden die beiden betrachteten Modellierungsansätze sowie das graph-basierte Domänenmodell eingeführt. Außerdem wird die Abbildung der beiden Modellierungsansätze auf das graph-basierte Domänenmodell skizziert. Kapitel 4 beschreibt den Lösungsansatz. In Kapitel 5 werden eine Zusammenfassung des Papiers und ein Ausblick auf weitere Arbeiten gegeben.

1 Anwendungsszenario

Im Folgenden wird ein Szenario als Anwendungsbeispiel eingeführt. Das Szenario beschreibt ein einfaches CPS (siehe Abbildung 1), welches aus einem Ofen (links) und einem Umformer (rechts) besteht. Werkstücke werden in dem Ofen für eine bestimmte Zeit erhitzt und anschließend umgeformt. Der Cyberanteil berechnet hierbei die optimalen Regelparameter für die Halte-dauer des Werkstückes im Ofen. Allerdings wird der Prozess hierbei noch von einem Menschen überwacht, welcher defekte Werkstücke aussortiert und die Umformung manuell einstellt.

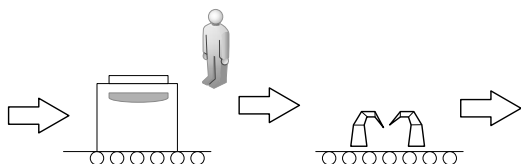


Abbildung 1: Übersicht über den Ist-Zustand des Fertigungsprozesses

Dieser Fertigungsprozess lässt sich wie in der Abbildung 1 bereits angedeutet als Prozesskette modellieren (*prozessorientierte Modellierung*). Er lässt sich aber auch in seine einzelnen Komponenten (Ofen, Mensch und Umformer) zerlegen auf deren Basis ein Komponentenmodell für die spätere Realisierung des Systems erstellt wird (*komponentenorientierte Modellierung*).

Auf Grundlage dieser Komponenten soll nun unter Zuhilfenahme von neuen Komponenten (RFID-Readern und RFID-Chips) ein neuer, verbesserter Prozess integriert werden. Das ursprüngliche CPS soll zu einem erweiterten CPS evolviert werden (siehe Abbildung 2).

Ziel der Erweiterung ist es, dass der Mensch durch die RFID-Technologie ersetzt wird. Bei der Auslegung des erweiterten Systems stellen sich für den Entwickler allerdings einige Fragen, wie viele neue Komponenten führe ich ein und an welcher Stelle integriere ich diese in das erweiterte Komponentenmodell? Eine Unterstützungshilfe hierfür wird in den folgenden Kapiteln genauer betrachtet.

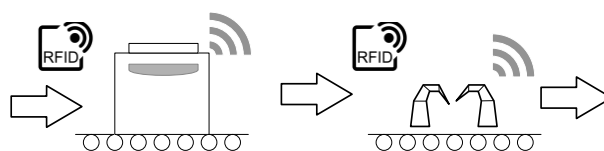


Abbildung 2: Übersicht über den Soll-Zustand des Fertigungsprozesses

2 Aktueller Stand der Technik

Um eine gemeinsame Simulation auf Basis der unterschiedlichen Modellierungsansätze zu realisieren, wird bei der Systementwicklung heute oft eine gleichzeitige Simulation zur Echtzeit-Synchronisation in interdisziplinären Simulationsumgebung verwendet. In [5] synchronisiert Bartelt die Komponenten von verschiedenen und unabhängigen Simulationsprogrammen, um die Daten auszutauschen. Aber die Änderungen der Anforderung sowie der Systemstruktur können nicht durch gleichzeitige Simulation synchronisiert werden.

Das Functional Mock-up Interface (FMI) ist ein von Daimler im Rahmen des Projektes MODELISAR entwickelter Standard zur Kopplung verschiedener Simulationsmodule [6]. Der Ansatz besteht darin, Simulationsmodule mithilfe einer einheitlichen Schnittstelle in anderen Simulationsmodulen einzubinden. Für die Kopplung verschiedener Module wird eine Master-Simulation definiert. In diesen Master werden dann die entsprechenden Module eingesetzt. Auch wie der Datenaustausch zwischen den Modulen abläuft, muss im Master modelliert werden.

Mosaik ist ein weiterer Ansatz mit dem Ziel, verschiedene Simulationsmodule zu koppeln [7]. Der Fokus liegt dabei auf dem Anwendungsgebiet intelligenter Stromnetze. Eine Programmierschnittstelle ermöglicht es, Adapter für verschiedene Simulationswerkzeuge zu entwickeln. Es wird angenommen, dass alle Simulationsmodule zeitbasiert sind und jeweils über eine feste Schrittweite verfügen. Eine Management Komponente

wählt anhand der Simulationszeit der Module die Reihenfolge aus, in der sie einen Schritt machen können.

Ptolemy ist ein weiterer Ansatz zum Überwinden der Heterogenität bei der Simulation komplexer Systeme [8]. Anders als bei den bereits beschriebenen Arbeiten wird hier allerdings nicht versucht, bestehende Simulationstools zu verbinden. Stattdessen wird eine Modellierungstechnik vorgeschlagen, welche unterschiedliche Semantiken umfasst. Die Modelle werden dabei aus einem Pool vordefinierter Elemente, den sogenannten Actors, zusammengesetzt. Um zu definieren, wie genau der Datenaustausch zwischen den Actors eines Modells abläuft, wählt der Modellierer einen sogenannten Director aus. Modelle können wiederum als Actor in anderen Modellen eingesetzt werden. Die so entstehende Hierarchie erlaubt die Kopplung verschiedener Semantiken.

Allen diesen vorherigen Lösungen ist gemein, dass sie zwar eine technische Integration verschiedener Modelle erlauben, allerdings kein geeignetes höherwertiges Konzept zur Strukturierung und Integration neuer Komponenten in bestehende Systeme ermöglichen. Stattdessen müssen neue Komponenten im Allgemeinen manuell mit einem hohen Aufwand integriert werden.

3 Grundlagen

3.1 Prozessorientierte Modellierung

Die Wertstromanalyse (engl. Value Stream Mapping, kurz: VSM) ist eine prozessorientierte standardisierte Modellierungssprache mit einer grafischen Darstellung, um die Prozessführung in Produktion und Dienstleistung zu beschreiben [9].

Die Materialflüsse sind visualisiert durch die Materialflusspfeile, sowie die Informationsflüsse durch die Informationsflusspfeile. Die Arbeitsprozesse vernetzen sich mit den Materialpfeilen. Damit kann das gesamte Cyber-physische System durch ein gemeinsames Netzwerk dargestellt werden. Der Ist-Zustand des Fertigungsprozesses aus dem zuvor beschriebenen Anwendungsszenario ist in Abbildung 3 mit Wertstromanalyse modelliert.

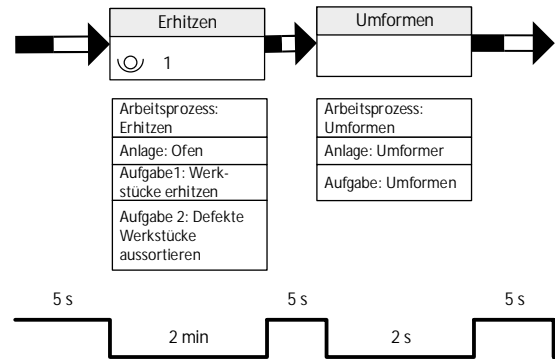


Abbildung 3: Ist-Zustand des Fertigungsprozesses mit Wertstromanalyse modelliert

Die Interpretationen der Symbole werden in der Abbildung 4 gezeigt. Die Interpretationen der anderen Symbole und die Kombinationsregeln sind unter [9] detailliert beschrieben.

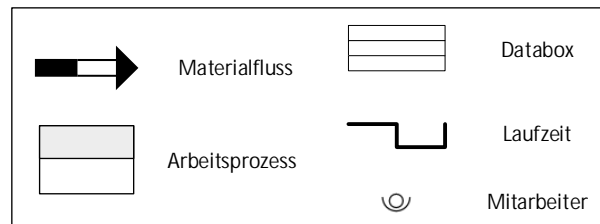


Abbildung 4: Notationselemente der Wertstromanalyse und deren Bedeutung

3.2 Komponentenorientierte Modellierung

Ein internes Blockdiagramm (IBD) ist eine auf der UML 2 basierte standardisierte komponentenorientierte Modellierungssprache. Das IBD besteht aus Systemkomponenten sowie Schnittstellen in Form von Ports und Verbindungen. Der Ist-Zustand des Fertigungsprozesses im gezeigten Anwendungsszenario ist in Abbildung 5 mit IBD beschrieben [7].

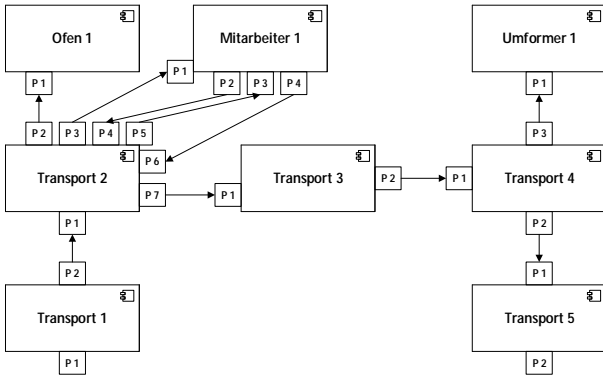


Abbildung 5: Der mit IBD modellierte Ist-Zustand

3.3 Graph-Struktur

Um die prozess- und komponentenorientierten Modelle universal repräsentieren zu können, wird eine gerichtete Graph-Struktur eingeführt:

$$\begin{aligned}
 G &:= (V, E, src, tgt) \\
 src &:= src|_{E \rightarrow V} \\
 tgt &:= tgt|_{E \rightarrow V}
 \end{aligned}
 \quad 1)$$

Die Elemente der Menge V seien die Knoten der Graph-Struktur G . Die Elemente der Menge E seien die gerichteten Kanten (Pfeile) in G . Die Kanten werden durch $E := V \times V$ beschrieben. Die Funktionen src und tgt bezeichnen die Verbindungsbeziehungen einer Kante mit seinem Quell-Knoten (engl. Source, kurz: src) und Ziel-Knoten (engl. Target, kurz: tgt). Jeder Knoten und jede Kante hat Attribute, um die semantischen Informationen für die Beschreibung zu speichern. Die konkreten Schritte werden im nachfolgenden Abschnitt weiter erläutert. Die Funktion $attributes$ bezeichnet die Datenstruktur der Attribute der Knoten und Kanten. Dabei bezeichnet P die Potenzmenge von Key-Value Paaren.

$$\begin{aligned}
 attributes &:= V \cup E \rightarrow P(\text{Key} \times \text{Value}) \\
 \text{Key} &:= \text{a set of values} \\
 \text{Value} &:= \text{a set of values}
 \end{aligned}
 \quad 2)$$

3.4 Transformation zwischen Modell und Graph-Struktur

Eine Transformationsfunktion bv repräsentiert die Transformation zwischen Wertstromanalyse Modellen (VSM) und Graphen.

$$bv(m_{VSM}) \leftrightarrow g_{VSM} \quad 3)$$

Dabei werden alle Arbeitsprozesse und alle Materialflüsse zu Knoten transformiert und alle Verbindungsbeziehungen zu Kanten. Beispielsweise wird die Verknüpfung von Arbeitsprozess und Materialfluss durch eine Verbindungsbeziehung realisiert.

Die Elemente Databoxes, Operator und Laufzeit mit den semantischen Informationen für die Beschreibung werden als Attribute in den dazugehörigen Knoten gespeichert. Dadurch wird die Transformation umkehrbar: Kanten und Knoten können wieder zu ihren ursprünglichen Flüssen und Elementen des VSM transformiert werden. Die folgende Abbildung 6 zeigt den mit Wertstromanalyse modellierten Ist-Zustand des Fertigungsprozess aus dem eingeführten Anwendungsszenario als Graph.



Abbildung 6: Der mit Wertstromanalyse modellierte Ist-Zustand als Graph

Der Prozess „Erhitzen“ wird beispielsweise zu einem Knoten $v2$ transformiert. Dabei werden die Elemente „Databox“ und „Laufzeit“ mit den darin enthaltenen semantischen Informationen zu Attributen dieses Knotens transformiert. Die Verbindungsbeziehung zwischen Prozess „Erhitzen“ und dem zweiten Materialfluss Element wird zur Kante $e2$ transformiert.

Der Soll-Zustand des Fertigungsprozesses aus Abbildung 2 kann ebenfalls durch einem Graphen beschrieben werden, siehe Abbildung 7. Der Knoten $v2'$ repräsentiert einen neuen Prozess im Vergleich zum Ist-Zustand. In dem Anwendungsszenario entspricht dieser neue Prozess dem „Erhitzen-Prozess mit RFID Technologie“.



Abbildung 7: Der mit Wertstromanalyse modellierte Soll-Zustand als Graph

Eine Transformationsfunktion bi repräsentiert die Transformation zwischen IBD Modellen und Graphen.

$$bi(m_{IBD}) \leftrightarrow g_{IBD} \quad (4)$$

Dabei sind alle Blöcke und Ports zu Knoten transformiert, und alle Verbindungsbeziehungen zu Kanten. Diese Transformation ist ebenfalls umkehrbar. Abbildung 8 zeigt den mit IBD modellierten Ist-Zustand des

Fertigungsprozesses aus dem Anwendungsszenario als Graph. Die Farben bezeichnen die Zuordnungen der Knoten im IBD Graph zu den Knoten v1 bis v5 im Wertstromanalyse Graph.

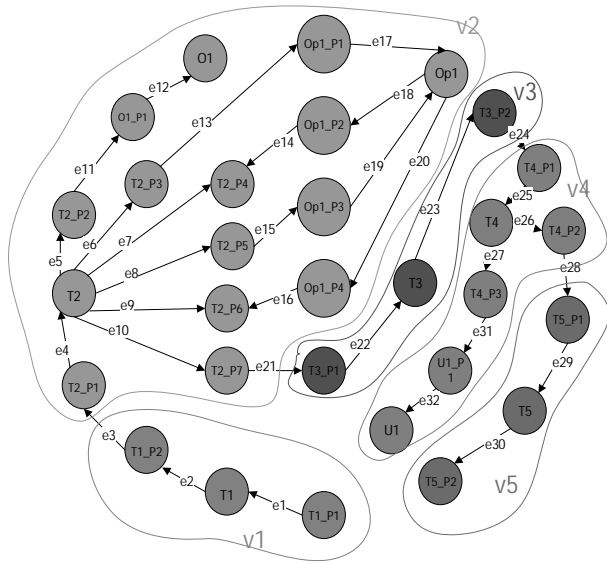


Abbildung 8: Der mit IBD modellierte Ist-Zustand als Graph mit Zuordnung zu dem VSM Graph

Die Komponente „Transport 1“ ist zu dem Knoten T1 transformiert worden. Die zugehörigen Schnittstellen P1 und P2 sind zu den Knoten T1_P1 und T1_P2 transformiert worden. Die Verbindungsbeziehungen zwischen den Komponenten „Transport 1“ und seinen Schnittstellen werden zu Kanten e1 und e2 transformiert.

Die Komponente „Transform 1“ mit den zwei Schnittstellen implementiert die Funktion im ersten Materialfluss Element. Aus diesem Grund sind die Knoten T1, T1_P1 und T1_P2 im IBD Graph den Knoten v1 des VSM Graphen zugeordnet.

4 Lösungsbeschreibung

Dieses Papier beschreibt ein Konzept, wie komponenten- und prozessorientierte Modelle gemeinsam weiterentwickelt werden können. Die Lösungsarchitektur wird in Abbildung 9 gezeigt.

Prozessorientiertes Modell des Ist- und Soll Zustands. Der Ist-Zustand eines CPS ist durch ein prozess- und komponentenorientiertes Modell beschrieben. Das prozessorientierte VSM Modell be-

schreibt die Funktionen und die Umsetzungsprozesse des CPS. Das weiterentwickelte CPS wird als Soll-Zustand des CPS bezeichnet und durch ein prozessorientiertes VSM Modell beschrieben.

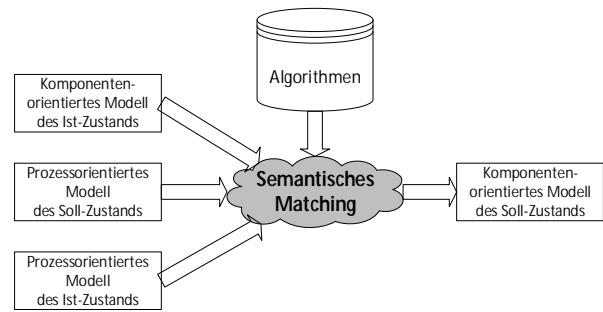


Abbildung 9: Lösungsarchitektur

Komponentenorientiertes Modell des Ist-Zustands. Das komponentenorientierte IBD Modell beschreibt die Systemstruktur und die Verbindungen zwischen den Komponenten des CPS. Die Zuordnungen der Komponenten im IBD Modell zu den Prozessen im VSM Modell müssen vom Entwickler des Modells definiert werden.

Algorithmen. Das bestehende CPS kann nach verschiedenen Kriterien optimiert werden. Ein potentielles Kriterium wäre hierbei der monetäre Ansatz, d.h., dass die von der Implementierung kostengünstigste Variante gewählt wird. Eine andere Möglichkeit wäre die Optimierung hinsichtlich der Zeit: Durch die Einführung neuer Komponenten kann der Gesamtprozess des neuen CPS Systems beschleunigt werden. Daneben gibt es noch viele weitere Kriterien, wie Qualität oder eine Kombination aus mehreren Kriterien. Die verschiedenen Optimierungsalgorithmen liegen in einer Datenbank vor und können vom Anwender beliebig ausgewählt werden.

In dem hier eingeführten Szenario wurde monetär optimiert. Dadurch wurde die kostenintensive Komponente *Mitarbeiter* durch günstige RFID Komponenten ersetzt, welche keine Kosten mehr pro Stunde verursachen, sondern lediglich einmalige Anschaffungskosten.

Alle Algorithmen und Regeln, z.B. für die Transformation zwischen Modell und Graph, die Graph-Suchverfahren und die Kombinationsregeln sind in einer Algorithmen Bibliothek gespeichert.

Semantisches Matching Plattform. Die drei vorhandenen Modelle werden über die sog. *Seman-*

tisches Matching Plattform zu drei entsprechenden Graphen transformiert. Dabei werden auch alle Beschreibungsinformationen und die Zuordnungsbeziehungen (Farben) zu den entsprechenden Attributen transformiert. Eine Menge von IBD Graphen wird basierend auf dem Unterschied zwischen Ist- und Soll-Zustand der zwei VSM Graphen durch die Graph-Suchverfahren und die Zuordnungsbeziehungen generiert. Diese Menge wird als Menge der Lösungskandidaten verstanden. Das semantische Matching in diesem Papier ist eine Methode, um alle Lösungskandidaten durch die Beschreibungsinformationen jeder Komponente zu filtern. Im Anwendungsszenario werden alle Lösungskandidaten mit den semantischen Informationen „Erhitzen-Prozess mit RFID Technologie“ betrachtet. Dabei werden alle IBD Modelle der Kandidaten, die solche Funktion nicht implementieren können, aus der Lösungsmenge entfernt. Die verbleibenden Modelle werden mit den Optimierungsalgorithmen weitergeprüft, bis die optimale Lösung gefunden wird.

5 Zusammenfassung und Ausblick

Es wurde ein Ansatz eingeführt, welcher die Weiterentwicklung von komplexen Cyber-physischen Systemen unterstützt. Die verwendeten heterogenen Modelle wurden zunächst auf eine gemeinsame graph-basierte Beschreibung abstrahiert. Anschließend wurde mithilfe der semantischen Matching Plattform eine optimale Integration der neuen Komponenten unter Berücksichtigung von Metadaten bestimmt. Diese Metadaten charakterisieren die einzelnen Prozesse oder Komponenten (abhängig von der vorherigen Modellierung) ausreichend, damit der Entscheidungsalgorithmus die bestmögliche Integration der neuen Komponenten/Prozesse in das bestehende CPS vornehmen kann.

In dem Anwendungsbeispiel empfahl der Algorithmus, RFID-Reader an den Entscheidungspunkten zu integrieren, in diesem Fall vor dem Hochofen und vor dem Umformer. Hierdurch konnte der Prozess verbessert werden, indem die Entscheidung über die Weiterverarbeitung der Produkte auf die RFID Technologie ausgelagert wurde und dadurch der Mensch nicht mehr zwangsweise Teil des Prozess sein muss.

Das zugrunde liegende Konzept wird aktuell kontinuierlich weiterentwickelt und im Forschungsprojekt

„Synus“¹ angewendet, um Industrie 4.0 Lösungen optimal in bereits bestehende Systeme zu integrieren. Hiermit soll insbesondere für KMUs eine Entscheidungshilfe für die Einführung neuer Technologien geboten werden.

Literatur

- [1] K. Bettenhausen, S. Kowalewski. *Cyber-Physical Systems: Chancen und Nutzen aus Sicht der Automation*. p 13, 2013.
- [2] H. Grönniger, J. O. Ringert, B. Rumpe. *System Model-Based Definition of Modeling Language Semantics*. In: Proc. of FMOODS/FORTE 2009, LNCS 5522. Lisbon, Portugal, 2009. pp 152-166.
- [3] Y. Laghouaouta, A. Anwar, M. Nassar. *A formal definition of model composition traceability*. International Journal of Computer Science Issues (IJCSI); 2015. vol. 12(6), pp 46-54 .
- [4] A. Schwartz. *Einführung in die Graphentheorie*. Mathematik für Informatik und Bioinformatik. Vorlesungsskript. Würzburg, 2013.
- [5] C. Bartelt, V. Böß, J. Brüning, B. Denkena, A. Rausch and J.P. Tatou. *A Software Architecture to Synchronize Interactivity of Concurrent Simulations in Systems Engineering*. 20th ISPE International Conference on Concurrent Engineering, At Melbourne, Volume: 20. 2013.
- [6] T. Blochwitz, M. Otter, J. Akesson, M. Arnold, C. Clauss, H. Elmqvist, M. Friedrich, A. Junghanns, J. Mauss, D. Neumerkel. *Functional mockup interface 2.0: The standard for tool independent exchange of simulation models*. 2012.
- [7] S. Schütte, S. Scherfke, M. Tröschel. *Mosaik: A framework for modular simulation of active components in Smart Grids*, in: Smart Grid Modeling and Simulation (SGMS), 2011 IEEE First International Workshop on. IEEE, 2011, pp. 55–60.
- [8] J. Eker, J.W. Janneck, E. Lee, J. Liu, X. Liu, J. Ludvig, S. Neuendorffer, S. Sachs, Y. Xiong, others. *Taming heterogeneity-the Ptolemy approach*. Proc. IEEE 91, 2003, pp 127–144.
- [9] M. Rother, J. Shook. *Learning to See: Value Stream Mapping to Create Value and Eliminate Muda*. Version 1.2, June 1999, Part II.
- [10] E. Huang, R. Ramamurthy, L.F. McGinnis, *System and simulation modeling using SysML*. WSC 07 Proc. of the 39th conference on Winter simulation, 2007, pp 796-803.

Danksagung. Diese Veröffentlichung entstand aus dem Forschungsprojekt "Synus" (Methoden und Werkzeuge zur synergetischen Konzeption und Bewertung

¹ www.tu-braunschweig.de/iwf/pul/forschung/projekte/synus

von Industrie 4.0-Lösungen), das vom Europäischen Fonds für regionale Entwicklung (EFRE | ZW 6-85012454) gefördert und vom Projektträger NBank verwaltet wird.

Leichtbauoptimierung von Crashstrukturen unter Einsatz einer automatisierten CAx-Prozesskette

Alexander Schülke¹, Andree Kafurke¹, Igor Sokrut¹, Prof. Dr.-Ing. Jürgen Hillmann¹, Prof. Dr.-Ing. Martin Müller¹

¹Institut für Fahrzeugbau, Hochschule für angewandte Wissenschaften, Kleiststraße 26 38440 Wolfsburg, mailto: al.schuelke@ostfalia.de

Abstract. Die vorliegende Arbeit befasst sich mit der Entwicklung und dem Aufbau einer vollautomatisierten CAx-Prozesskette zur Modellerstellung, Simulation und Optimierung sicherheitsrelevanter Bauteile im Crashlastfall. Exemplarisch wird aufgezeigt wie mithilfe des Konstruktionsprogramms CATIA V5, des Solvers Pamcrash zur Crashberechnung sowie der Software Isight zur Optimierung eine Verknüpfung erzeugt wird, die es erlaubt, CAD-basierte Formoptimierungen im explizit-dynamischen Lastfall durchzuführen. Es lassen sich mithilfe der Prozesskette deutliche Leichtbaupotentiale am Beispiel des seitlichen Türaufprallträgers sowie dem Front-/Heckquerträger im Lastfall Pfahlcrash aufzeigen.

Motivation

Um effektiv Leichtbau betreiben zu können, muss bereits früh in den Entwicklungsprozess eines Fahrzeugs eingegriffen werden. Ein Werkzeug, welches sich hier bereits vor Jahren etabliert hat, ist die Simulation mithilfe der Finite-Elemente-Methode (FEM). Die heutige Entwicklungsarbeit ist hierbei immer noch stark schnittstellenbehaftet. Dies verzögert den Entwicklungsprozess und schränkt Bauteiloptimierungen deutlich ein. [1] Mit einer direkten Schnittstelle zwischen Konstruktion und Simulation kann schneller auf Bauteiländerungen reagiert werden und mithilfe von Optimierungsstrategien effizient das Optimum eines Bauteils erreicht werden, welches ohne diesen Einsatz eventuell verborgen geblieben wäre.

Aufgrund des hohen Projektzeitdrucks bei der Entwicklung neuer Fahrzeuge werden Bauteile oftmals nur bis zur Erfüllung von Anforderungen entwickelt, was dafür sorgt, dass eine Optimierung ausbleibt. Insbesondere in der dynamischen Crashberechnung, sind die Möglichkeiten der Optimierung durch die Trennung der Simulation und Konstruktion stark eingeschränkt. Die klassische Topologieoptimierung ist für Crash-Strukturen derzeit

ebenfalls ungeeignet [2]. Dies führt zu einer vergleichsweise langsam iterierenden Performancesteigerung sicherheitsrelevanter Bauteile. Die Potentiale neuer Bauteile werden selten hinsichtlich aller Zielgrößen, wie beispielsweise Gewicht, Energieabsorption und Kosten, optimal ausgenutzt. Daher wird häufig auf Standardlösungen zurückgegriffen, die auch in vorherigen Projekten funktioniert haben und werden lediglich an die neuen Gegebenheiten angepasst. [3]

Um diesen Zustand zu verbessern, wird in der vorliegenden Arbeit die Verwendung von neuartigen automatisierten CAx-Prozessketten vorgeschlagen, die auf der parametrisch-assoziativen Kopplung der Disziplinen CAD, CAE und CAO basieren und deutlich über den heutigen Stand der Technik hinsichtlich der Arbeitsweise in diesen Disziplinen hinausgehen. Die vorgestellte CAx-Methodik wurde am Institut für Fahrzeugbau Wolfsburg der Ostfalia Hochschule für angewandte Wissenschaften entwickelt und bereits auf unterschiedliche Bauteile und Lastfälle angewendet [4, 5].

1 Anforderungen an die CAx-Prozesskette

Die vorgestellte CAx-Prozesskette für die Auslegung und Optimierung von Crashstrukturen basiert auf der CAD-Software CATIA V5, den Berechnungsprogrammen Pamcrash sowie Abaqus und der Optimierungssoftware Isight. Die CAx-Prozesskette ist auf unterschiedlichste Lastfälle und Konstruktionsdatensätze adaptierbar und kann vollautomatisiert multikriterielle Optimierungen durchführen.

1.1 Parametrisch-assoziative Konstruktion

Für eine spätere Optimierung ist der Aspekt der parametrisch-assoziativen Konstruktion entscheidend. Es han-

delt sich hierbei um eine Verknüpfung von Funktionselementen innerhalb einer Konstruktion. Ändert sich eine dieser Funktionen, wie z.B. ein parametrierter Radius, so werden alle verknüpften Funktionen in der Reihenfolge ihrer Abhängigkeit neu berechnet.

1.2 Automatisierte Vernetzung auf Basis der CAD-Flächen

Die Vernetzung des Bauteils erfolgt mithilfe der in CATIA V5 implementierten Produkte *FMS* und *FMD*. Dieser Vernetzer ermöglicht eine assoziative Verknüpfung des FE-Netzes mit der CAD-Fläche. Dies hat den Vorteil, dass das Netz bei einer Formänderung über ein Update entsprechend angepasst wird. Zudem werden die zugehörigen *Geometry Constraints* für die Erzeugung des aktualisierten Netzes beibehalten. Dies ermöglicht über den Prozess eine nahezu gleichbleibende Netzqualität.

Neben der Vernetzung können zudem die Schweißpunkte innerhalb des Bauteils in der Konstruktion definiert werden. Somit gehen bei einer starken Änderung der Form des Bauteils bei dem Übertrag in das Simulationsmodell keine Fügeverbindungen verloren. Ebenso lassen sich dadurch Schweißverbindungen in der Anzahl oder ihrer Ausprägung variieren.

Die Netze werden anschließend über das Advanced Meshing Tool (FMS) im Bulkformat ausgegeben, welches dem Eingabeformat des FE-Programms Nastran entspricht. Um das Format entsprechend für Pamcrash lesbar zu machen, wurde ein Skript in Python programmiert, welches zur Übersetzung des Formats eingesetzt werden kann. Bei der Nutzung des Abaqus Solvers kann das Berechnungsmodell direkt aus CATIA heraus generiert werden.

1.3 Kopplung zur Simulationssoftware Pamcrash

Ein entscheidender Aspekt für die Verkettung der Disziplinen ist die parametrisch-assoziative Kopplung zwischen dem Konstruktionsdatensatz und dem FEM-Modell.

Nach der Übersetzung des Formats in Pamcrash kann das ASCII File als *.inc abgespeichert werden. Ein Hinzufügen des zu optimierenden Teils erfolgt anschließend über die sogenannte Include-Funktion in Pamcrash. Im Falle des Anwendungsbeispiels Türaufprallträger wurde ein Basismodell eines Teilmodells einer Fahrzeugtür erzeugt und mit allen Randbedingungen versehen, die für die Berechnung benötigt werden. Alle relevanten Teile für den Seitencrash werden nun als Include-Dateien in Hauptmodell hinterlegt - so auch der Türaufprallträger. Dies ermöglicht es, das Bauteil als separiertes Teil innerhalb der

Prozessschleife auszutauschen ohne für jeden Optimierungsvorgang ein neues Berechnungsmodell zu erzeugen.

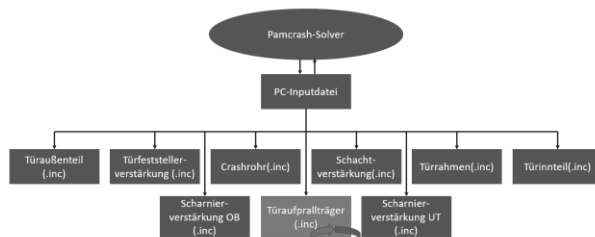


Abbildung 1: Dateistruktur des Analysemodells

1.4 Aufbau der multidisziplinären Optimierung

Die multidisziplinäre Optimierung (MDO) lässt sich mit zwei unterschiedlichen Ansätzen beschreiben. Zum einen sind hiermit unterschiedlich räumliche Positionen eines definierten Lastfalls innerhalb eines Optimierungszyklus gemeint. Dies können zum Beispiel verschiedene Pfahlcashpositionen sein, welche in definierten Abständen eine dynamische Belastung auf ein Bauteil aufbringen. Zum anderen bezieht sie sich auf tatsächlich unterschiedliche Lastfälle, welche innerhalb einer Optimierung abgeprüft werden. Bezogen auf die dynamische Crashauflegung kann dies ein Pfahlaufprall in Kombination mit einem Barrieren Crash sein.

Für den Einsatz der MDO können daher zwei nachgelagerte Prozessschleifen eingebracht werden, welche jeweils eine Zielgröße an den Algorithmus übergeben.

2 Aufbau der CAX-Prozesskette

Die CAX-Prozesskette besteht im Wesentlichen aus den vier Hauptkomponenten: Optimierungsalgorithmus, Pre-processing, Berechnung und Postprocessing.

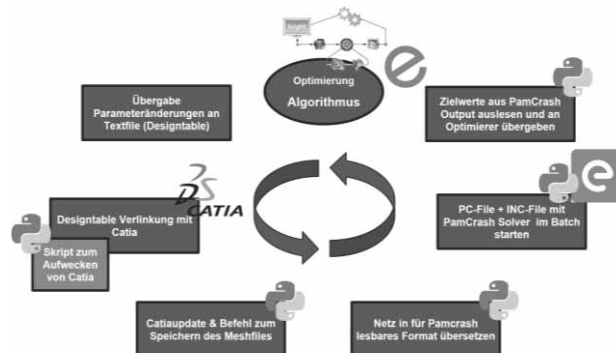


Abbildung 2: Workflow der CAX-Prozesskette

Den Rahmen der Prozessschleife bietet hierbei das Optimierungsprogramm Isight. Dieses bietet zum einen eine

große Auswahl an verschiedenen Optimierungsalgorithmen und Strategien. Zum anderen können hier Komponenten eingebracht werden, die programmierte Skripte und Makros enthalten um mit dem Betriebssystem und der darauf installierten Software kommunizieren können. Somit ist es möglich über den Optimierungsalgorithmus Parametervariationen an ein Textfile zu übergeben, welche wiederum als *Designable* mit den Konstruktionsdaten in CATIA V5 verknüpft sind.

Über ein VBA Makro kann anschließend ein Update angestoßen werden, welches dafür sorgt, dass sich alle Geometrien der neuen Parametrierung anpassen. Im Falle dieser Prozesskette wird das Makro über eine Reaktion auf ein neu erzeugtes File bewirkt. Daraufhin wird das FE-Netz aktualisiert und herausgeschrieben. Anschließend wird das Netzformat übersetzt und die PLINKS für das Bauteil neu hinterlegt.

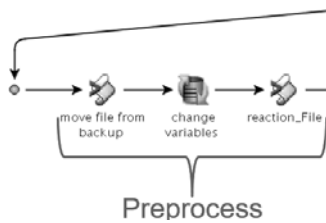


Abbildung 3: Preprocess Isight Prozesskette

Im Anschluss werden die Versagensbedingungen für Elemente und Fügeverbindungen in das neu erzeugte Bauteil eingebracht.

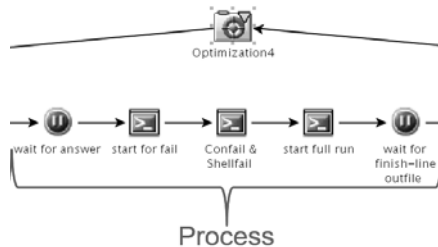


Abbildung 4: Process Isight Prozesskette

Nachdem das Analysemodell vollständig ist, wird die Berechnung gestartet. Nach Abschluss der Berechnung werden die Ergebnisse wiederum über ein Makro über den Visual Viewer von Pamcrash entnommen und an den Optimierungsalgorithmus zurück übergeben.

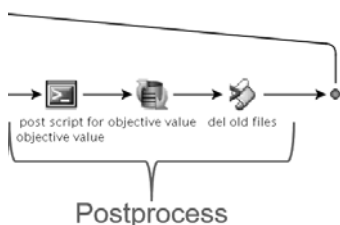


Abbildung 5: Postprocess Isight Prozesskette

3 Optimierung

3.1 Algorithmus

Bei der eigentlichen Optimierung wird zum jetzigen Zeitpunkt des Projekts der sogenannte Pointer-Algorithmus verwendet. Hierbei handelt es sich um eine Optimierungsstrategie, welche aus vier einzelnen Algorithmen besteht, die innerhalb des Prozesses und in Abhängigkeit der Zielfunktion variiert werden können. Bei den Algorithmen handelt es sich um den Downhill-Simplex, einem linearen Simplex-Algorithmus, einen genetischen Algorithmus sowie einer sequentiellen Programmierung. Der Pointer wird empfohlen, wenn keinerlei Informationen zu der jeweiligen Zielfunktion vorliegen.

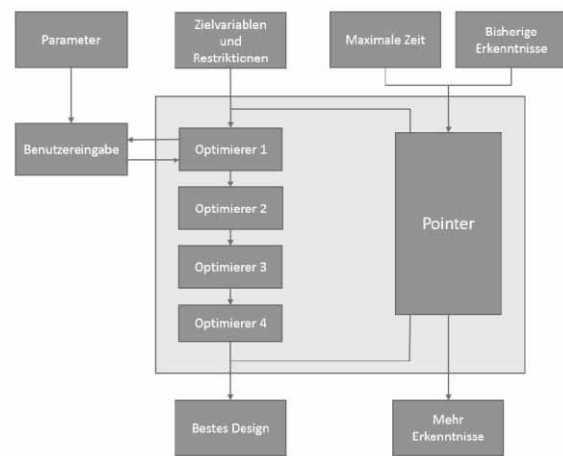


Abbildung 6: Pointer Algorithmus

3.2 Leichtbauansätze zur Optimierung

Hinsichtlich der Optimierung werden mehrere Leichtbaustrategien verfolgt. Unter anderem der Stoffleichtbau, der Formleichtbau sowie der Fertigungsleichtbau.

Beim Stoffleichtbau wird eine Materialsubstitution einzelner Bauteile während der Optimierung durchgeführt. Zudem können einzelne Materialcharakteristiken separiert variiert werden [6].

Bei der Optimierung in Hinblick auf den Fertigungsleichtbau wird das Tailored Blank Verfahren simuliert. Hierzu wird auf Konstruktionsebene das Bauteil in Längsrichtung in einzelne Abschnitte unterteilt. Den Abschnitten werden unterschiedliche Blechdicken zugeordnet, die anschließend dem Algorithmus als Parameter übergeben werden [7]. Bei der Unterteilung in die einzelnen Abschnitte sind zudem konstruktive Formeln mit den Fertigungsrestriktionen des Tailor Rolled Blank (TRB) Verfahren hinterlegt. Hierunter zählen zum einen die maximalen Blechdickenänderungen entlang einer Geometrie aber auch Parameter wie die definierten Übergangsbereiche zwischen den unterschiedlichen Unterteilungen

sowie die minimalen Plateaulängen. Somit ist das optimierte Endergebnis im Idealfall als fertiger Konstruktionsdatensatz weiterverwendbar.

Eine weitere Optimierungsmethode stellt die bereits erwähnte CAD-basierte Formoptimierung dar. Hierbei werden Parameter zur Beschreibung der äußeren Formgebung verwendet, die durch einen formellen Zusammenhang untereinander als Parameter definiert werden können. Ein Beispiel hierfür ist in Abbildung 7 dargestellt. Hierbei wird ein offenes Hutprofil hinsichtlich der Radien und den Zwischenflächen der Radien optimiert.

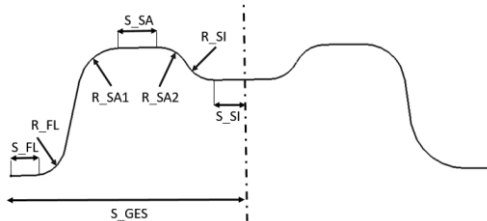


Abbildung 7: Parameterdefinition der Formoptimierung

Ein entscheidender Vorteil bei dem Einsatz dieser Strategie stellt die Optimierung auf Basis des CAD-Datensatzes dar. Im Gegensatz zu Verfahren wie der Topologieoptimierung sowie netzbasierter Formoptimierung (Morphing), wird für jeden Designvorschlag eine neue Vernetzung der Fläche durchgeführt. Dies bewirkt einerseits, dass es innerhalb der Vernetzung zu keinen Verzerrungen kommen kann, welche zu Berechnungsproblemen führen können. Andererseits sind die Daten als CAD-Datensatz weiterverwendbar ohne eine anschließende Adaption der Optimierungsergebnisse in eine erneute Konstruktion einfließen lassen zu müssen [8]. Deutlich hervorzuheben ist, dass durch die vorgestellte Methode auch Topologieoptimierungen möglich sind. Dies kann beispielsweise eine Rippenstruktur eines Kunststoffbauteils sein, die vorher parametrisiert und automatisiert aufgebaut wurde. [9]

Diese Methode erhöht zwar im Vorfeld den Aufwand für die Parametrierung der Konstruktion und erfordert einen ausreichend updatestabilen Aufbau, führt aber dazu, dass sich die Ergebnisse über den Prozess der Optimierung in einem realitätsnahen Rahmen (Fertigungsrestriktionen) bewegen und nur geringe nachgelagerte händische Arbeiten erforderlich sind. Außerdem lassen sich gut aufgebaute parametrisch-assoziative Konstruktionen in anderen Projekten leicht adaptieren. Die Adaption kann in vielen Fällen auch automatisiert erfolgen.

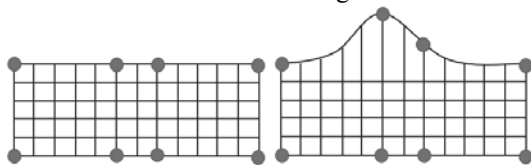


Abbildung 8: Netzverzerrung (netzbas. Formoptimierung)

4 Erkenntnisse

Die Ergebnisse der Optimierungen zeigen ein deutliches Gewichtseinsparpotential auf. Beim Beispiel Aufprallträgers konnte im seitlichen Pfahlcrash ein signifikantes Gewichtseinsparpotential von 40% bei gleichen Energieaufnahmeigenschaften aufgezeigt werden. Dies wurde durch die Optimierung auf Basis der implementierten Tailor Rolled Blank Methode erreicht. Es zeigt sich, dass die von Mises-Spannungen sich durch die lastfallspezifische Blechdickenverteilung deutlich homogener verteilen, was die bessere Ausnutzung des Materials belegt. Hierdurch wird außerdem erreicht, dass sich im Einleitungspunkt kein plastisches Gelenk ausbildet, was zu einer geringeren Intrusion in den Fahrzeuginnenraum führt. Durch den Einsatz weiterer Profilgeometrien, die anschließend weiter optimiert wurden, konnten Gewichtsreduktionen von über 50% erreicht werden.

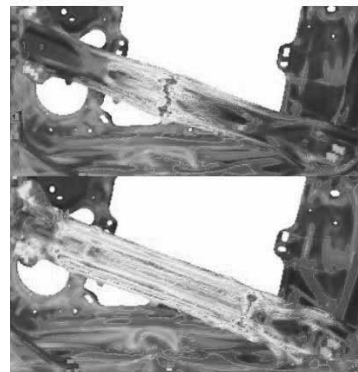


Abbildung 9: Ergebnis einer TRB-Optimierung

5 Zusammenfassung und Ausblick

Eine weitere Neuerung, welche in Zukunft in die Entwicklung der vorgestellten CAx-Prozessketten einfließen wird, ist neben der parametrisch-assoziativen Verknüpfung zwischen Konstruktion und Simulation auch die hochautomatisierte Modellbildung. Hierbei wird auf Basis eines vorhandenen Konstruktionsdatensatzes in Form eines CAD-Produktes ein Finite-Elemente-Modell abgeleitet, das bereits alle Rand-, Füge- und Kontaktbedingungen beinhaltet. Erste Ergebnisse zeigen bereits ein erhebliches Potential auf. Diese Art der Modellbildung setzt jedoch eine eindeutige Struktur in der CAD-Konstruktion voraus, um die notwendigen Randbedingungen wie z.B. die definierten Fügeareale eindeutig zu erkennen und in das FE-Modell übertragen zu können.

Zudem soll die vorgestellte Prozesskette in Zukunft auf weitere Modellvarianten und Lastfälle übertragen werden.

References

- [1] HERFELD, Ulrich: *Matrix-basierte Verknüpfung von Komponenten und Funktionen zur Integration von Konstruktion und numerischer Simulation*. München. Zugl.: München, Techn. Univ., Diss., 2007
- [2] SCHUMACHER, Axel; VIETOR, Thomas; FIEBIG, Sierk ; BLETZINGER, Kai-Uwe ; MAUTE, Kurt: *Advances in Structural and Multidisciplinary Optimization*. Cham: Springer International Publishing, 2018
- [3] DIETRICH, Tim: *Neue virtuelle Methoden zur Integration einer numerischen Fußgängerschutz-Optimierung in den Entwicklungsprozess einer Motorhaube*. München. Zugl.: München, Techn. Univ., Diss., 2013
- [4] SOKRUT, Igor; MÜLLER, Martin Prof. Dr.-Ing.: Maximale Leichtbaupotentiale durch CAD-integrierte FEM-Berechnung am Beispiel von Karosserieaußenflächen. In: *18. Kongress SIMVEC - Simulation und Erprobung in der Fahrzeugentwicklung 2016: Berechnung, Prüfstands- und Straßenversuch; Baden-Baden, 22. und 23. November 2016*. Düsseldorf: VDI Verlag GmbH, 2016 (VDI-Berichte, 2279), S. 391–403
- [5] SOKRUT, Igor; MÜLLER, Martin Prof. Dr.-Ing.: *Herausforderungen und Potentiale des Leichtbaus von dünnwandigen Karosserieaußenflächen (Faszination hybrider Leichtbau)*. Wolfsburg, 24.-25.05.2016
- [6] HENNING FRANK, MOELLER ELVIRA: *HANDBUCH LEICHTBAU: METHODEN, WERKSTOFFE, FERTIGUNG*. 1.AUFLAGE. CARL HANSER VERLAG. MÜNCHEN. 2011
- [7] FRIEDRICH, HORST E.: *LEICHTBAU IN DER FAHRZEUGTECHNIK*. 2.AUFLAGE. SPRINGER VIE-WEG.WIESBADEN.2017
- [8] HARZHEIM, LOTHAR: *STRUKTUROPTIMIERUNG: GRUNDLAGEN UND ANWENDUNG*. 2.AUFLAGE. EUROPA-LEHRMITTEL. HAAN-GRUITEN.2014
- [9] SOKRUT, IGOR; MÜLLER, MARTIN PROF. DR.-ING.: *WISSENSBASIERTE KONSTRUKTION MIT UNTERSTÜTZUNG DER CAD-INTEGRIERTEN SIMULATION (KUNSTSTOFFTRENDS IM AUTOMOBIL)*. WOLFSBURG, 04.-05.09.2013

Simulation Case Study: Using Simscape for Human Knee Joint Models

Ruth Leskovar^{1*}, Andreas Körner¹, Felix Breiteneker¹

¹Institute for Analysis and Scientific Computing, TU Wien, Wiedner Hauptstraße 8-10, 1040 Vienna, Austria;
*ruth.leskovar@tuwien.ac.at

Abstract. This contribution presents a knee model implemented in Simscape™ and analyses its usability regarding biomechanical aspects. The model simulates the flexion of a human knee. It contains the three bones of the human knee, which are linked together by two revolute joints and one spring damper element representing the patellar tendon. This illustrates a simplification of the human knee joint due to the restriction of degrees of freedom. Finally, this work discusses the advantages and disadvantages of using the multibody library in Simscape for biomechanical models.

Introduction

In the research field of biomechanics, mathematical models for anatomic joints play an important role analysing kinematics and kinetics in the human body. Mainly, two different modelling approaches are used, models based on partial differential equations and multibody systems. They differ in their mathematical description and therefore in application fields as well. Models described by partial differential equations (PDEs) depend on time and space, which gives the opportunity to analyse even small deformations, which take place in human bone and soft tissue, as ligaments and tendons, under repeated loads.

Multibody models are described by ordinary differential equations (ODEs), which leads to final solutions dependent on time only. These models do not require a precise description as PDE models, because detailed information about the geometries of the bodies does not influence the solution of the model strongly. Multibody models are often used for investigating gross motion and interaction of various connected bodies. Here, a description of movement change over time is needed and no deformations in tissues are investigated.

One possible application is the gait analysis. The study of motion sequences, in case of injuries compared to a

normal gait cycle, can give insights to rehabilitation and therapy techniques. Further on, biomechanical models, which simulate the human walk, can be extended to various walking scenarios, as running or climbing stairs. Results gained from these models can be used in the development of prostheses to improve functions of leg prostheses. For example, multibody models are developed for the investigation of interactions taking place in the human body of amputees wearing prostheses during various falling scenarios as it is done in [1].

1 Multibody Modelling

Physical modelling describes a system based on fundamental physical laws. This technique is often used for systems, where no mathematical equations describing the dynamical behaviour of the system are known. Multibody models are based on the physical modelling approach and describe relative motion between different bodies and the resulting dynamics. Multibody systems consist of bodies and joints, which link them together. Bodies are defined by their physical properties, as mass, centre of mass, density, inertial rotation, etc. There exist various types of joints, which differ in their amount and properties of degrees of freedom, resulting in rotational or translational movement respectively.

Figure 1 shows two rigid bodies, which are connected by a kinematic joint. The positions of the bodies, defined by their local coordinates, can be described in respect to the global coordinate system as it is explained more detailed in [2]. Since the physical model building process does not require mathematical equations, the development of multibody models is often simpler and faster than other methods. The description of a system by PDEs requires more detailed information about geometries defining the included bodies and the numerical solving procedure is more complex than using ODE solver, which are sufficient for multibody models. Furthermore, the physical modelling approach can be used

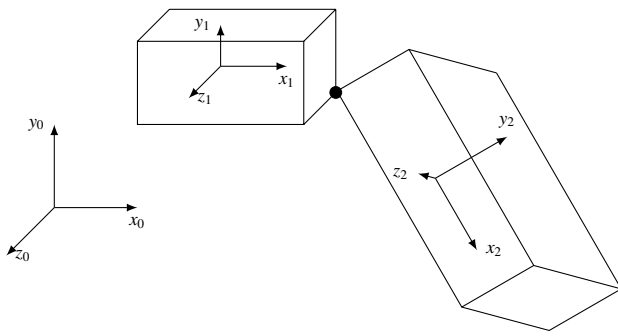


Figure 1: Illustration of a multibody system with local coordinates, indicated by x_1 and x_2 , and the global coordinate system, indicated by x_0 .

to detect mathematical equations of complex systems to describe their dynamical behaviour. The combination of components, whose dynamics are already known, leads to the desired system description. The analysis and further investigation of simulation results gives the possibility to describe the dynamics by mathematical equations.

In multibody models, the relative motion of a rigid body, which results by applying a force \mathbf{F} , is given by the set of a second order ordinary differential equations

$$M\ddot{\mathbf{x}} + J_x^T \lambda = \mathbf{F}, \quad (1)$$

with the mass matrix M of the system, the coordinates \mathbf{x} of the investigated body, the corresponding Jacobian matrix J and the Lagrange multipliers λ . The derivation can be conducted by using the Lagrange formalism and is explained in [3].

As already mentioned above, multibody models are used to analyse kinematics between bodies in biomechanics. For example, they are used to analyse the interactions in the human body, which take place during motion.

2 Conceptual Model for the Human Knee

The platform <https://simtk.org> offers a repository of biomechanical models. Researchers can share their work, including simulation models and corresponding files as geometries or data. The open access strategy facilitates the development of new models.

The following model simulates the flexion of a human knee and is based on the work of [4], [5] and

[6]. The investigated models are mainly implemented in AdamsTM, a multibody dynamics simulation software. These models describe the flexion of the tibia and the involved movement of the patella in respect to a fixed femur after applying a force at the tibia. Similar to the anatomy of the human knee, ligament forces are included, connecting the bones together and pre-determining the direction of motion. Additionally, contact forces influence the movement to prevent interpenetration of the bones.

The usage of the multibody library for Simscape, embedded in the Simulink[®] environment, requires the introduction of joints, otherwise no movement is possible. A possible simplification of the knee joint reduces the degrees of freedom in the knee to one rotational as it is stated in [7]. For the analysis of load distribution in the knee joint, this simplification is sufficient. This rotational degree of freedom describes the flexion of the tibia and is represented by one revolute joint. This revolute joint j_1 , connecting femur and tibia, is placed in the last third of the femoral condyle as it is depicted in Figure 2. A second revolute joint j_2 is introduced, linking the patella to the femur and describing the movement of the patella. This joint is situated at the centre of mass of the femur to ensure that the patella is sliding between the femoral condyles at the front side. In the anatomy of the human knee, the patella is situated in the patellar tendon, which connects the quadriceps to the tibia. In biomechanics, tendons are often implemented using spring damper forces. Therefore, one spring damper element is introduced, which connects tibia and patella. This component applies a linear force acting reciprocally between the connected bodies. This force f is pro-

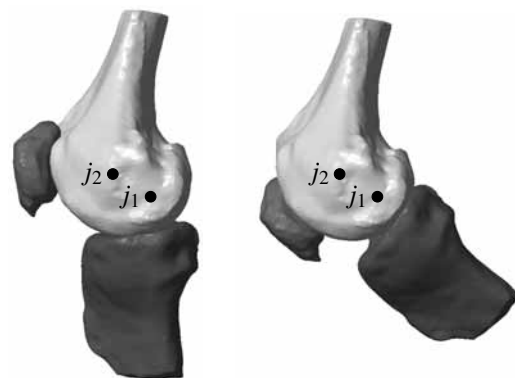


Figure 2: Visualisation of the joint centres in the knee model in the initial position left and in flexion right.

portional to the distance x between the connected bodies and the resulting velocity \dot{x} . It is calculated by

$$f = k \cdot (x - l) + D \cdot \dot{x}, \quad (2)$$

with the spring constant k and damper coefficient D . The natural length of the spring is given by l . This spring damper force does not imply any movement to the bodies, but ensures that the patella is sliding following the rotation of the tibia. The tibia starts to move only after an applied torque at the joint j_1 .

3 Simulation Model in Simscape for the Human Knee

Simscape is embedded in the Simulink environment and is intended for the development of physical systems. The library comprises elements for driveline, electrical, fluids and multibody models. Due to the embedment to Simulink, it is possible to use components of Simulink as well. For that purpose, special blocks convert the different signal types from 3D of Simscape to 1D of Simulink and vice versa.

The construction of a simulation model in Simscape is similar to Simulink and realised by drag and drop of components. The final model structure for the flexion of a human knee is shown in Figure 3. Each block represents one component of the model. The global parameters, as gravity and solver settings as well as the global reference frame, are defined in blocks too. Without these blocks, a simulation run of a Simscape model is not possible.

The knee model consists of three bones, femur, tibia and patella, which are implemented as rigid bodies. The corresponding blocks contain `stl`-files defining their geometry and shape as well as physical parameters, namely mass, centre of mass and inertial rotation. These files and parameters are given with the Adams models of [4], [5] and [6] and represent data of the right knee of a 77 year old man.

The stiffness of the spring and damping coefficient for the patellar tendon between tibia and patella are the same values as in the Adams model. The rigid transforms between the components assure the correct attachment points of the tendon to the bones and the right position of joint centres, respectively.

The revolute joints contain parameters for spring stiffness and damping coefficient of the joint and are summarised in Table 1. These parameters are not the same

as in the Adams models, because there no rotational joints are considered. These models deal with ligaments only, which contain parameters for translational movements. The conversion of these parameters for rotational movement requires knowledge about rotational stiffness and is not accessible in this case.

	Spring stiffness	Damping coefficient
j_1	$553.5 \frac{\text{N mm}}{\text{deg}}$	$1 \frac{\text{N mm s}}{\text{deg}}$
j_2	$33 \frac{\text{N mm}}{\text{deg}}$	$1 \frac{\text{N mm s}}{\text{deg}}$

Table 1: Parameters for the joints.

Therefore, the parameters for the joints are calculated by calibration and comparing the outputs from the Adams and Simscape model. Both model do not show the same behaviour due to the already discussed restrictions. One more subsequent difference from the model description is the input. In the Adams model, a force is acting at the tibia in posterior direction. The Simscape model requires a torque τ acting on the joint j_1 . The torque τ is calculated by using geometrical basics and is finally given by

$$\tau(t) = 28.8 \cdot \sin(2\pi \cdot 0.125(t - 1)) \cdot H(t - 1) \text{ Nm}, \quad (3)$$

with the step function H . The acting torque is implemented in Simscape by using Simulink blocks.

The output of the simulation model is the resulting angle between femur and tibia representing the flexion and extension of the human knee. Figure 4 shows the angle of the Adams model and the Simscape model. The comparison of the output leads to the missing joint parameters. The non-linearity of the Adams model, coming from the ligaments, results in a more steep waveform than the sine wave from the Simscape model.

4 Conclusion and Outlook

The presented knee model is a simplification of the human knee joint and it shows the possibilities and restrictions of using the multibody library in Simscape for biomechanical models. The usage of Simscape for building multibody models is a good option due to the embedment in the Simulink environment. This allows to use block elements from other libraries and all tools which are available in Simulink. Moreover, the postprocessing in MATLAB[®] offers many possibilities. This

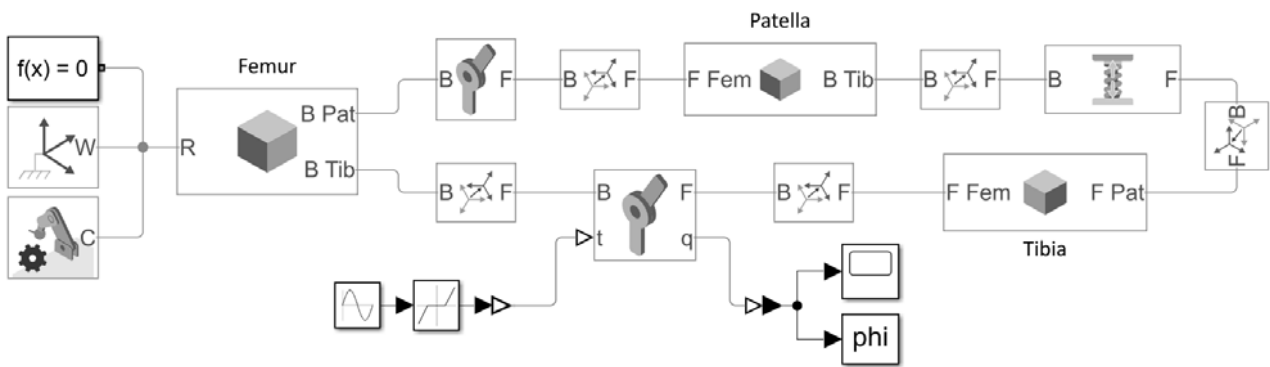


Figure 3: Block structure of the multibody model for the human knee implemented in Simscape.

gives flexibility in the development and extension of the model and later in the analysis. For example, the construction of more complex model structures and combination with system based modelling approaches allows to build feedback loops, which extend the areas of application for biomechanical models.

Since MathWorks® does not focus on physical modelling, the library of Simscape is restricted regarding some aspects. Anatomic joints are complex structures due to their composition of various tissues as bones, ligaments and tendons. In order to build a model which fulfils the biomechanical properties of the human knee joint, the consideration of a revolute joint is insufficient. The incorporation of crucial and collateral ligaments to the model increases the degrees of freedom and improves a realistic movement. As it is discussed in [8], ligaments show a non-linear behaviour regarding their stress-strain relationship. Therefore, they can not be modelled with linear spring damper elements but

using external functions. Nevertheless, the Simscape multibody library requires the use of joints to create the preconditions for the movements.

References

- [1] Welke B, Schwarze M, Hurschler C, Calliess T, Seehaus F. Multi-body simulation of various falling scenarios for determining resulting loads at the prosthesis interface of transfemoral amputees with osseointegrated fixation. *Journal of Orthopaedic Research*. 2013; 31(7):1123–1129.
- [2] Georg Rill TS. *Grundlagen und Methoden der Mehrkörpersimulation*, vol. 3. Springer Vieweg. 2017.
- [3] Quental C, Folgado J, Ambrósio J, Monteiro J. A multibody biomechanical model of the upper limb including the shoulder girdle. *Multibody System Dynamics*. 2012;28(1-2):83–108.
- [4] Guess TM, Thiagarajan G, Kia M, Mishra M. A subject specific multibody model of the knee with menisci. *Medical Engineering and Physics*. 2010;32(5):505–515.
- [5] Guess TM, Liu H, Bhashyam S, Thiagarajan G. A multibody knee model with discrete cartilage prediction of tibio-femoral contact mechanics. *Computer Methods in Biomechanics and Biomedical Engineering*. 2013; 16(3):256–270.
- [6] Bloemker KH, Guess TM, Maletsky L, Dodd K. Computational Knee Ligament Modeling Using Experimentally Determined Zero-Load Lengths. *The Open Biomedical Engineering Journal*. 2012;6:33–41.
- [7] Brinckmann P, Frobin W, Leivseth G, Drerup B. *Orthopädische Biomechanik*. MV Wissenschaft. 2012.
- [8] Blankevoort L, Huiskes R. Ligament-Bone Interaction in a Three-Dimensional Model of the Knee. *Journal of Biomechanical Engineering*. 1991;113(3):263.

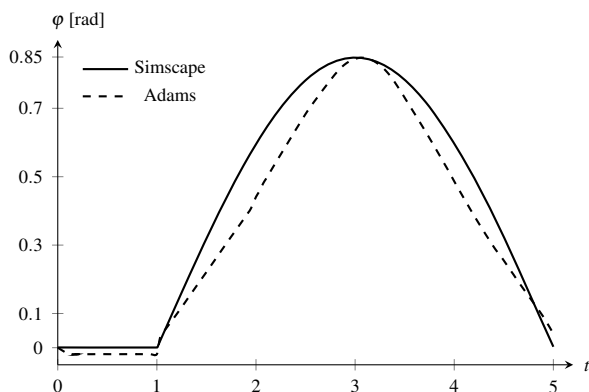


Figure 4: Angle ϕ between femur and tibia during flexion of the knee for the Adams and Simscape model.

A Simple Simulation Model to Test Detection Methods for Periodic Effects on Transit Traversal Times

Oliver Ullrich*, Daniel Lückerath

Fraunhofer Institute for Intelligent Analysis and Information Systems IAIS, Schloss Birlinghoven, 53757 Sankt Augustin, Germany *oliver.ullrich@iais.fraunhofer.de

Abstract. In bus transit networks without dedicated bus lanes vehicle traversal times are strongly dependent on the traffic density of the corresponding road segments. This paper takes the first steps towards demonstrating that the traffic density has regular patterns that yield a small number of periodic effects on the traversal time: the current traversal time is related to the traversal time of the directly preceding vehicles, and to the vehicles passing the same road segment on the preceding day at the same time of day, one week ago, and one year ago. Methods to detect the presence and impact of these periodic effects can be tested by using simulation: periodic effects can be introduced in a controlled manner, and detection methods using simulation output data can be evaluated against that known input. This paper proposes a simulation model to help with the test and evaluation of detection methods for periodic effects on bus transit traversal times. After a short introduction to aims and scope, it shares some background on bus transit networks and existing simulation models, and then goes on to describe a simple model that includes the controlled introduction of periodic effects. The paper closes with a short discussion of the necessary follow-up research.

Introduction

The delay of vehicle departures in bus transit networks is dependent on the time a vehicle needs to traverse road segments and the time necessary for passenger exchange at stops on the route. Passenger exchange times are dependent on the numbers of boarding and alighting passengers that in turn are dependent on the demand for transportation at a specific location and point in time, and on already existing delays of the current and pre-

ceding vehicle (vehicle bunching, see [4]).

In networks without dedicated bus lanes vehicle traversal times are under normal operating conditions strongly dependent on the traffic density on the corresponding road segment (see [3] and [6]).

More specifically, we propose that traversal times in networks with periodic schedules and small to medium basic intervals are strongly influenced by a small number of effects that occur in periodic patterns, and that can be detected by applying simple statistics on tracking data without necessitating further domain knowledge. Information on presence and impact of these periodic effects can be utilized to predict vehicle traversal times, and therefore delayed service in bus transit networks.

Preceding vehicles The traversal time of a vehicle on a specific road segment is related to the traversal time of the directly preceding vehicles on the same road segment. For example, a temporary congestion would influence not just one but a number of succeeding vehicles to approximately the same extent. That does not apply for the comparatively rare case of a congestion building up or dissipating spontaneously.

Time of day The traversal time is also related to the traversal time of vehicles passing the same road segment on the preceding days at the same time of day, for example during rush hours or during the night. That also does not apply in some cases, for example if the preceding day is a Friday or Sunday.

Day of week Traversal times are also related to vehicle behavior of a corresponding vehicle that traversed the same road segment during the same time and day in the preceding weeks. Thus, Friday night traffic is re-

lated to Friday night traffic, vehicle behavior on a Sunday can be compared to the preceeding Sunday traffic.

Time of year Finally, and to a lesser extent, comparing current traversal times to the behavior of corresponding vehicles a full year ago helps to model seasonal effects beyond traffic density, for example different base velocities due to weather, or specific traffic densities due to holidays or the start of a semester.

Assuming a minimal traversal time $t_e(r)$ for any segment r when it is completely void of any traffic, four factors $c_p(t, r)$, $c_d(t, r)$, $c_w(t, r)$, and $c_y(t, r) \in \mathbb{R} \geq 1$ can be defined that together describe the relation of a traversal time $t(t, r)$ at time t to that minimum time: the factor $c_p(t, r)$ the preceeding vehicle took longer than $t_e(r)$, a factor $c_d(t, r)$ describing the relation to the corresponding vehicle on the preceeding day, on the preceeding week $c_w(t, r)$, and on the corresponding day of the preceeding year $c_y(t, r)$.

If we define weights w_p , w_d , w_w , and $w_y \in \mathbb{R}$ with $w_p + w_d + w_w + w_y = 1$ this results in an estimated traversal time $t(t, r)$ at time t for a road segment r :

$$t(t, r) = (w_p \times c_p(t, r) + w_d \times c_d(t, r) + w_w \times c_w(t, r) + w_y \times c_y(t, r)) \times t_e(r) \quad (1)$$

We propose that, in networks in which traversal times are strongly dependent on traffic density, these four effects – with shifting weights – have a strong impact on traversal times.

These considerations leave us with three research questions:

1. Do periodic effects as described actually exist and play a significant role in bus traversal times?
2. If they exist, how to reliably detect the presence and amount of period effects?
3. If they can be detected, can they be used to reliably predict bus traversal times, and thereby departure delays in a bus network without applying further domain knowledge?

Applying a simulation model can help to find answers to research question 2: In a simulated bus transit network, periodic effects can be introduced in a controlled manner. Based on the simulated output data, different detection methods can then be evaluated regarding whether

they can detect and assess these introduced components correctly.

This paper describes the first steps towards the adaption of a simple bus transit simulation model to aid the development and test of methods that detect period effects on traversal times. After sharing some background on bus transit networks and related research on mesoscopic bus transit simulation (see Section 1) the paper continues with a description of the simple simulation model designed to test the detection methods (see Section 2). It closes with a short discussion on the next research steps (see Section 3).

1 Background

1.1 Bus Transit Networks

A bus transit system includes a street network and a set of bus stops where passenger exchanges take place. These bus stops are served by a set of transit vehicles executing service trips, i.e. pairings of starting times and sequences of bus stops, according to a timetable. Each individual vehicle executes several service trips, interspersed with deadhead trips, over the course of an operational day. That list of trips is called a rotation. Such a rotation usually begins with a deadhead trip from the vehicle depot to the first stop of its first service trip and, after a number of service trips, ends with a returning deadhead trip to the depot. The vehicle schedule defines the assignment of specific vehicles to rotations.

Some stops are marked as control points, i.e. locations in the network where control strategies may be employed, for example purposely delaying early vehicles until the scheduled departure time is reached. At other stops, vehicles depart as soon as the passenger exchange is complete. Directed paths through the network, connecting two successive stops, are called connections. They usually consist of several street segments, junctions, and signals that in turn can be shared by several connections. Signals control access to street segments, usually at junctions. Often, two or more signals constitute a signal group with a common scheduling strategy. Public transit vehicles generally follow predefined line routes, consisting of sequences of stops to be serviced.

1.2 Related Work

A number of simulation models covering bus transit can be found in literature (for example, see [1], [2], [5], [7],

[9], [10], and [11]).

One of the first models was proposed in 1979 by Andersson et al. in [1]. Those authors develop a mesoscopic event-based interactive simulation model for bus transit systems allowing users online testing of operational strategies, like short-turning trips or dead-heading vehicles. Andersson et al. model the bus system as a set of lines, i.e. collections of linked stops, where each stop possesses a separate holding bay for every line serving it. As a result, the model does not represent direct vehicle interactions. Instead, interactions between vehicles are modeled indirectly via the passenger exchange process. Because passengers can be served by multiple lines, delays or earliness of a vehicle of one line may affect the passenger exchange processes of vehicles of other lines, resulting in vehicle bunching effects. The traversal process of vehicles between successive stops is modeled mesoscopically using lognormal distributed random values dependent on the time of day.

A newer mesoscopic approach to bus transit simulation is proposed by Toledo et al. in [11]. The authors extend a mesoscopic simulation model for individual traffic based on queuing theory proposed by Burghout in [3] that represents the street network as a graph of interconnected queues and vehicles as individual entities traversing these queues based on speed/density functions. The nodes of the graph represent junctions and are modeled as collections of servers, one for each turning movement and each with different processing times based on, for example, the green time ratio of the corresponding signal.

Other recent simulation models including bus transit use microscopic agent-based modeling approaches and are based on generic multi-modal transit models (for example, see [2], [5], [9], and [10]). Lückérath et al. describe in [7] a mesoscopic bus transit simulation model designed to help assessing the impact of operational rules used to mitigate disturbances and disruptions of transit services. The model described below is a simplified version of that model.

2 A Simple Mesoscopic Simulation Model

2.1 Basic Model

The described model consists of representations of physical network components like road segments and stops, logical components like lines and rotations, and

vehicle dynamics like traversal and stop times.

The physical network is represented by a directed graph $G = (V, E)$, where stops and connections are modeled as nodes $v \in V$, and their neighborhood relations are represented by edges $e \in E$. A stop $s \in S$ is attributed with an identifier, time of day-specific passenger arrival rates, and a maximum capacity for concurrently stopping vehicles. In addition, some stops are marked as control points. As stops are assigned to exactly one station, each entity contains a reference to its station object. Connections $c = (s_i, s_j) \in C \subseteq S \times S$ are directed paths through the transit network between two successive stops s_i and s_j . Connections are attributed with a length, a planned traversal time $t_p(c)$ according to the timetable, and might also contain a list of atomic street segments and a set of signals.

A line $l \in L$ is modeled as an ordered list $l_i = (s(i_1), c(i_1), s(i_2), c(i_2), \dots, c(i_{n-1}), s(i_n))$ of stops $s(i_j) \in S$ that are to be serviced successively by a vehicle of a specific type, interspersed with the relevant connections $c(i_j) \in C$ between each two successive stops. This avoids elaborate path finding over the course of a simulation run. Planned service trips are tuples of a line to be served and a planned departure time at the first stop of that line. The set of all service trips defines the services available to prospective passengers during an operational day, i.e. the timetable. Rotations are ordered sequences of all the trips to be executed by one – not yet specified – vehicle during a single operational day. A vehicle schedule assigns one specific vehicle to each rotation.

Vehicles are represented by entities traversing the model graph over the course of a simulation run according to timetable, vehicle schedule, and operational strategies, encapsulating a significant part of the simulation logic. Vehicles are classified according to their type and attributes: the vehicle type defines vehicle length, capacity, maximum velocity, minimum passenger exchange time and exchange rate.

The traversal time for connections and the passenger exchange times at stops are the stochastic elements of the model. For traversal times, a lognormal distribution d is assumed for the traversal times of a connection $c = (s_i, s_j)$ (see [1]). As the planned application does not necessitate an exact replication of real-world values, the parameters of this distribution, i.e. expectancy value μ_c^d and standard deviation σ_c^d , can be approximated from the planned traversal times $t_p(c)$.

The passenger exchange times are modeled follow-

ing the method described in [4]. The method is suitable for high frequency transit systems, where it can be assumed that passengers arrive randomly during the inter-arrival time of two successive vehicles, instead of arriving in bulk shortly before the planned departure time. Furthermore, the method facilitates the modeling of bus bunching, i.e. the effect that two vehicles form an undesired platoon because the vehicle in front takes on more passengers than planned and subsequently suffers longer passenger exchange times, while the rear vehicle takes on fewer passengers as planned and thus catches up to the vehicle in front.

If the number $N(b, s)$ of passengers entering a vehicle b at a stop s , and the average time I_b a passenger takes to enter vehicle b are known, the passenger exchange time $T(b, s)$ can be determined as follows:

$$T(b, s) = T_b^{min} + I_b \times N(b, s) \quad (2)$$

Here, T_b^{min} describes a vehicle specific minimum time, e.g. for opening and closing the vehicle's doors. If the passenger arrival rate a_s at stop s is known, $N(b, s)$ can be modeled dependent on the basic interval $T(L(b))$ of line $L(b)$ currently served by vehicle b . With $N(b, s) = T(L(b)) \times a_s$ the passenger exchange time can then be approximated as shown in Equation 3.

$$T(b, s) = T_b^{min} + I_b \times T(L(b)) \times a_s \quad (3)$$

If instead of the basic interval between vehicles of the same line, simulated headways between successive vehicles servicing the same stop are used, the model becomes dynamic and thus suitable for a simulation model. If $t_{dep}(b-1, s)$ describes the time the predecessor of a vehicle b has serviced the stop, the passenger exchange time $T_{sim}(b, s)$ can be determined as in shown in Equation 4.

$$T_{sim}(b, s) = \begin{cases} T_b^{min} & b \text{ is first vehicle at } s \\ T + (t_{sim} - t_{dep}(b-1, s)) \times a_s \times I_b & \text{else} \end{cases} \quad (4)$$

The model is being executed using a event-based simulation framework described in [12].

2.2 Introducing Periodic Effects

In a real-world bus transit network, the proposed periodic effects would impact vehicle traversal times directly and passenger exchange times indirectly – as different numbers of passengers accumulate during differ-

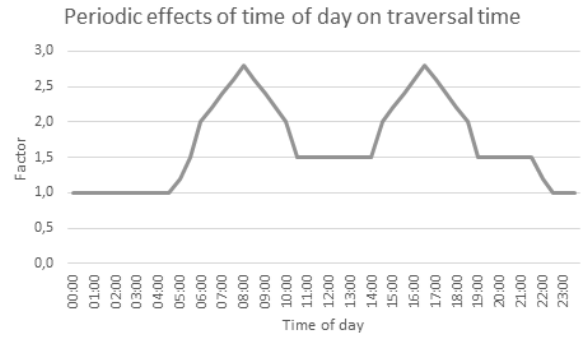


Figure 1: Periodic effects factor $c_d(t, r)$ on the traversal time of a bus on a road segment r at a given time t on a work day

ent time periods between two departures. The modeling of the passenger exchange times is already covered by the basic simulation model, it can be adopted without further change.

In the basic model, traversal times are modeled as drawn from a lognormal distribution with expectancy value μ_c^d and standard deviation σ_c^d , both constant for each connection c . To introduce periodic effects on the traversal time, and based on Equation 1, the expectancy value for connection c and time t can be computed as:

$$\mu_c^d(t) = (w_p \times c_p(t, r) + w_d \times c_d(t, r) + w_w \times c_w(t, r) + w_y \times c_y(t, r)) \times t_e(r) \quad (5)$$

That equation assumes that one road segment r can always be represented by one connection c .

The time series for $c_p(t, r)$, $c_d(t, r)$, $c_w(t, r)$, and $c_y(t, r)$ can simply be read from a table. Figure 1 depicts the development of the daily component $c_d(t, r)$ on a typical work day, without significant traffic during the night, two rush hour periods, and some midday and evening traffic. The weights w_p , w_d , w_w , and w_y can be set according to the desired focus of each experiment. The standard deviation σ_c^d can be retained unchanged.

3 Further Work

This paper described a simulation model aimed at testing detection methods for the impact of periodic effects on vehicle traversal times in bus transit networks.

To examine the research questions posed above, significant follow-up work is still necessary. As a next step

following the implementation of the described model as an event-based simulation application (see [12]) a method to detect the periodic effects has to be implemented, calibrated, and tested. At the time of this writing the authors envision a simple method based on reinforced learning (see [8]).

Following onto that, the simulated network has to be replaced first by historical tracking data of real-world bus routes, then – following further calibration and validation – by real-time bus tracking data. That would yield a real-time delay prediction method based on scarce tracking data and independent of further domain knowledge, that runs very resource conservingly.

References

- [1] Andersson, P., Hermansson, A., Tengvald, E., Scalia-Tomba, G.: Analysis and simulation of an urban bus route. In: *Transportation Research Part A*, Vol. 13, No. 6, 1979, pp. 439-466.
- [2] Behrisch, M., Erdmann, J., Krajzewicz, D.: Adding intermodality to the microscopic simulation package SUMO. In: *Proceedings of the 11th Middle Eastern Simulation Multiconference*, Alexandria, Egypt, 2010, pp. 59-66.
- [3] Burghout, W.: *Hybrid microscopic-mesoscopic traffic simulation*. Dissertation, Department of Infrastructure, Royal Institute of Technology, Sweden. University of Stockholm, 2004.
- [4] Chapman, R., Michel, J.: Modelling the Tendency of Buses to Form Pairs. In: *Transportation Science*, Vol. 12, No. 2, 1978, pp. 165-175.
- [5] Kendziorra, A., Weber, M.: Extensions for logistics and public transport in SUMO. In: *Proceedings of 3rd SUMO User Conference – Intermodal Simulation for Intermodal Transportation*, Berlin, Germany, 2015, pp. 83-90.
- [6] Kieu, L.-M., Bhaskar, A., Chung, E.: Public Transport Travel-Time Variability Definitions and Monitoring. In: *Journal of Transportation Engineering*, Vol. 141, No. 1, 2015.
- [7] Lückcrath, D., Rische, N., Speckenmeyer, E., Ullrich, O.: A Mesoscopic Bus Transit Simulation Model Based on Scarce Data. In: *Simulation Notes Europe (SNE)*, Vol. 28, No. 1, 2018, pp. 1-10.
- [8] Sugiyama, M.: *Statistical Reinforcement Learning*. CRC Press, Boca Raton, 2015.
- [9] Suzumura, T., Kanezashi, H.: Multi-modal traffic simulation platform on parallel and distributed systems. In: *Proceedings of the 2014 Winter Simulation Conference*, Savannah, Georgia, 2015, pp. 769-780.
- [10] Suzumura, T., McArdle, G., Kanezashi, H.: A high performance multi-modal traffic simulation platform and its case study with the Dublin city. In: *Proceedings of the 2015 Winter Simulation Conference*, Huntington Beach, California, 2015, pp. 767-778.
- [11] Toledo, T., Cats, O., Burghout, W., Koutsopoulos, H.: Mesoscopic simulation for transit operations. In: *Transportation Research Part C*, Vol. 18, No. 6, 2010, pp. 896-908.
- [12] Ullrich, O., Lückcrath, D. An Introduction to Discrete-Event Modeling and Simulation. In: *Simulation Notes Europe (SNE)*. Vol. 27, No. 1, 2017, pp. 9-16.