# Eine Methodik zur Kopplung räumlich verteilter realer und virtueller Prototypen

Von der Fakultät für Ingenieurwissenschaften,
Abteilung Maschinenbau und Verfahrenstechnik
der Universität Duisburg-Essen
zur Erlangung des akademischen Grades
eines

Doktors der Ingenieurwissenschaften
Dr.-Ing.

genehmigte Dissertation

von
Peter Gabriel Baumann
aus
Calw

Gutachter:
Prof. Dr.-Ing. Dr. h.c. Dieter Schramm
Prof. Dr.-Ing. Lars Mikelsons

Tag der mündlichen Prüfung: 16. Juli 2024

# Kurzfassung

Die zunehmende Vernetzung domänenübergreifender mechatronischer Systeme führt zu einer verstärkten Interaktion der einzelnen Systemkomponenten, die es im gesamten Entwicklungsprozess zu berücksichtigen gilt. In der Entwurfsphase wird dies durch die Co-Simulation von Simulationsmodellen der einzelnen Komponenten erreicht. In der Verifikations- und Validierungsphase dagegen, in der erste Komponenten bereits physisch vorliegen und die Interaktion des Gesamtsystems durch Kopplungen realer und virtueller Prototypen (RVP) dargestellt ist, werden diese Simulationsmodelle selten wiederverwendet. Ein Grund dafür ist, dass bei der Datenübertragung zwischen den Prototypen, die räumlich verteilt vorliegen können, zwangsläufig Netzwerkeffekte wie Latenzzeiten auftreten, die das Systemverhalten von RVPen in unbekannter und meist nachteiliger Weise beeinflussen.

In dieser Arbeit wird eine Kopplungsmethodik vorgestellt, welche den negativen Einfluss dieser Netzwerkeffekte auf das Systemverhalten von RVPen kompensiert. Die Kopplungsmethodik ist allgemein anwendbar und setzt kein Systemwissen über die gekoppelten Prototypen voraus. Daher ist sie besonders für den Einsatz in der Industrie geeignet, da dort zum Schutze des geistigen Eigentums Prototypen häufig nur als Black-Box vorliegen. Zunächst wird eine neue Klasse generischer Kopplungsalgorithmen entwickelt, die sich aus einem neuartigen Verfahren zur Extrapolation im Fehlerraum ergeben. Es wird gezeigt, dass diese Kopplungsalgorithmen in vorwärts gerichtete neuronale Netze mit identischem Eingangs-Ausgangs-Verhalten überführbar sind. Dadurch werden die Kopplungsalgorithmen zum einen um die Eigenschaft der Lernfähigkeit erweitert und zum anderen wird im Vergleich mit ausgewählten Algorithmen aus der Literatur eine verbesserte Latenzzeitkompensation nichtlinearer Koppelsignale erreicht. Weiterhin wird eine allgemeine Analyse des Kopplungsprozess von RVPen im Frequenzbereich durchgeführt, die sowohl zur Bestimmung des Einflusses von Netzwerkeffekten und Kopplungsalgorithmen auf das Systemverhalten von RVPen als auch für eine optimale Auslegung von Kopplungsalgorithmen verwendet wird. Außerdem hilft die Frequenzbereichsanalyse dabei Gültigkeitsbereiche für Kopplungsalgorithmen abzuleiten, welche zur Parametrierung ebenfalls in dieser Arbeit entwickelter Methoden zur Überwachung von Kopplungsalgorithmen dienen. Das Potenzial der entwickelten Kopplungsmethodik zur Kompensation des negativen Einflusses der Netzwerkeffekte wird unter realen Bedingungen anhand eines in der Industrie produktiv genutzten RVPs eines Hybridfahrzeugs demonstriert. Dabei zeigt sich, dass besonders die im Frequenzbereich optimal ausgelegten und anschließend trainierten Kopplungsalgorithmen den durch die Netzwerkeffekte entstehenden Kopplungsfehler verringern und somit das Systemverhalten des RVPs verbessern.

# Abstract

The ever-increasing connection of cross-domain mechatronic systems leads to increased interaction between the individual system components, which must be considered throughout the entire development process. In the design phase, this is achieved by co-simulation simulation models of the components. However, in the verification and validation phase, in which the first components are already physically available, and the interaction of the overall system is therefore represented by couplings of mixed real-virtual prototypes (RVP), these simulation models are rarely reused. One reason for this is that during data transmission between the prototypes, which can be spatially distributed, network effects such as latencies inevitably occur which influence the system behavior of RVPs in an unknown and usually unfavorable manner.

This thesis introduces a coupling methodology to compensate for the influence of the network effects on the system behavior of RVPs. The methodology is generally applicable and does not require any system knowledge about the coupled prototypes, which is why it is particularly suitable for use in industry, where prototypes are often only available as black boxes to protect intellectual property. First, a new class of generic coupling algorithms is developed, which results from a newly proposed method for extrapolation in error space. It is shown that these coupling algorithms can be converted into feedforward neural networks with identical input-output behavior. Thus, the ability to learn online is added and, in comparison with selected algorithms from the literature, an improved latency compensation of strongly nonlinear coupling signals is achieved. Furthermore, a general analysis of the coupling process of RVPs is carried out in the frequency domain, which is used to determine the influence of network effects and coupling algorithms on the system behavior of RVPs as well as for an optimal design of coupling algorithms. In addition, the frequency domain analysis is used to derive validity areas for coupling algorithms, which are used to parameterize monitoring methods for coupling algorithms developed in this thesis. The potential of the coupling methodology to compensate for the negative influence of the network effects is demonstrated under real conditions using an RVP of a hybrid vehicle that is productively used in industry. It is shown that coupling algorithms optimized in the frequency domain and subsequently trained reduce the coupling error caused by network effects, thereby improving the system behavior of the RVP.

# Inhaltsverzeichnis

**Kurzfassung** ... **iii**

**Abstract** ... **v**

**Inhaltsverzeichnis** ... **vii**

**Abbildungsverzeichnis** ... **xi**

**Tabellenverzeichnis** ... **xv**

**Abkürzungsverzeichnis** ... **xvii**

**Symbolverzeichnis** ... **xix**

**1 Einleitung** ... **1**

1.1 Motivation ... 1

1.1.1 Der Entwicklungsprozess komplexer mechatronischer Systeme ... 2

1.1.2 Räumlich verteilte Real-Virtuelle Prototypen ... 5

1.1.3 Problematik der Kopplung Real-Virtueller Prototypen ... 8

1.2 Einordnung der Arbeit in den wissenschaftlichen Kontext ... 9

1.2.1 Einsatzfelder Real-Virtueller Prototypen ... 9

1.2.2 Kompensation von Kopplungsfehlern ... 12

1.3 Wissenschaftlicher Beitrag und Aufbau der Arbeit ... 15

**2 Grundlagen gekoppelter Prototypen** ... **17**

2.1 Klassische Kopplungen virtueller Prototypen ... 17

2.1.1 Model-In-The-Loop ... 17

2.1.2 Software-In-The-Loop ... 24

2.1.3 Hardware-In-The-Loop ... 26

2.2 Aspekte räumlich verteilter Real-Virtueller Prototypen ... 29

2.2.1 Technische Randbedingungen ... 29

2.2.2 Echtzeitanforderungen an die virtuellen Prototypen ... 31

2.2.3 Echtzeit-Co-Simulation ... 32

2.3 Beispielsysteme für die Anwendung der entwickelten Methodik ... 35

2.3.1 Zweimassenschwinger ... 36

2.3.2 Nichtlinearer Zweimassenschwinger ... 37

2.4 Vergleichsmetriken ... 38

**3 Algorithmen zur Kopplung Real-Virtueller Prototypen ... 39**

3.1 Problemstellung und Anforderungen ... 39

3.2 Extrapolation im Fehlerraum ... 42

3.2.1 Verfahren zur Konstruktion eines Kopplungsalgorithmus ... 42

3.2.2 Ableitung konkreter Kopplungsalgorithmen ... 44

3.3 Diskussion der Kopplungsalgorithmen ... 46

3.3.1 ARMA und ARIMA Modelle ... 47

3.3.2 Klassifikation der Kopplungsalgorithmen als ARIMA Modelle ... 48

3.3.3 Überprüfung der Anforderungen ... 54

3.4 Anwendung auf lineare Real-Virtuelle Prototypen ... 56

3.5 Vorwärts gerichtetes, künstliches neuronales Netz als nichtlinearer Kopplungsalgorithmus ... 59

3.5.1 Generierung der Architektur der FFNN ... 61

3.5.2 Online Lernen der FFNN ... 64

3.6 Anwendung auf nichtlineare Real-Virtuelle Prototypen ... 65

**4 Analyse räumlich verteilter Real-Virtueller Prototypen ... 71**

4.1 Systemanalyse linearer Real-Virtueller Prototypen im Frequenzbereich ... 72

4.1.1 Laplace-Transformation des Kopplungsprozesses von RVPen ... 74

4.1.2 Untersuchung der Kopplungsalgorithmen im Frequenzbereich ... 78

4.1.3 Stabilität linearer Real-Virtueller Prototypen ... 83

4.2 Optimale Auslegung der Kompensation im Frequenzbereich ... 89

4.2.1 Definition des Optimierungsproblems ... 90

4.2.2 Anwendung der Optimierung auf lineare Real-Virtuelle Prototypen ... 93

4.3 Diskussion der Analysemethodik ... 100

**5 Aufbau und Überwachung Real-Virtueller Prototypen ... 103**

5.1 Vorbereitung der Ausführung von Real-Virtuellen Prototypen ... 104

5.1.1 Wahl der Makroschrittweite ... 105

5.1.2 Zeitsynchronisation ... 106

5.1.3 Start der Ausführung ... 107

5.2 Online Bewertung der Extrapolationsgüte ... 109

5.2.1 Verfahren nach Stettinger ... 111

5.2.2 Erweiterung der Fehlertoleranzfunktion ... 114

5.3 Umgang mit Diskontinuitäten in Koppelsignalen ... 119

5.3.1 Verfahren zur online Erkennung von Diskontinuitäten ... 119

5.3.2 Umschalten zwischen Kopplungsalgorithmen ... 125

**6 Anwendung der Kopplungsmethodik ..... 129**

6.1 Implementierung der Kopplungsmethodik ..... 130

6.1.1 Bereitstellung der Kopplungsmethodik mithilfe des FMI-Standards ..... 131

6.1.2 Client-Server-Architektur für das online Lernen ..... 133

6.2 Anwendung auf produktiv genutzten Real-Virtuellen Prototyp ..... 135

6.2.1 Analyse des Real-Virtuellen Prototyps ..... 137

6.2.2 Vorangehende Untersuchungen mithilfe einer Co-Simulation ..... 141

6.2.3 Ergebnisse der Kopplungsmethodik am Real-Virtuellen Prototyp ..... 144

6.2.4 Diskussion der Ergebnisse ..... 152

**7 Zusammenfassung und Ausblick ..... 155**

**Anhang ..... 159**

**Literaturverzeichnis ..... 161**

**Publikationen des Autors ..... 177**

# Abbildungsverzeichnis


**Abb. 1.1** V-Modell der Entwicklungsmethodik für mechatronische und cyber-physische Systeme in Anlehnung an VDI/VDE 2206 (2021) ..... 3
**Abb. 1.2** Kopplung eines realen und mehrere virtueller Prototypen über das Internet (Baumann et al., 2019a) ..... 6
**Abb. 1.3** Netzwerkeffekte in einem RVP nach (Stettinger et al., 2017) ..... 8
**Abb. 2.1** Piktografische Darstellung der MIL-Simulation eines batterieelektrischen Fahrzeugs mit den Domänen Fahrdynamik, Hochvoltbatterie, Antriebsstrang sowie Regel- und Fahrfunktionen ..... 18
**Abb. 2.2** Kopplung zweier Modelle mittels Co-Simulation (Busch, 2012) ..... 20
**Abb. 2.3** Schemata für Ausführungsreihenfolge einer Co-Simulation mit zwei Modellen. Links: Jacobi-Schema. Rechts: Gauss-Seidel-Schema. ..... 22
**Abb. 2.4** Piktografische Darstellung einer SIL-Simulation eines batterieelektrischen Fahrzeugs ..... 25
**Abb. 2.5** Blockdiagramm einer HIL-Simulation ..... 26
**Abb. 2.6** Datenaustausch zweier Prototypen über ein Netzwerk nach Stettinger et al. (2017) ..... 32
**Abb. 2.7** Linearer, gedämpfter Zweimassenschwinger ..... 36
**Abb. 2.8** Nichtlinearer, gedämpfter Zweimassenschwinger mit einseitigem Anschlag. ..... 37
**Abb. 3.1** Linearer Zweimassenschwinger mit ZOH als Kopplungsalgorithmus ..... 57
**Abb. 3.2** Vergleich verschiedener Kopplungsalgorithmen mit der Referenzlösung bei Anwendung auf den linearen Zweimassenschwinger. Links: Übersicht. Rechts: Ausschnitt ..... 58
**Abb. 3.3** Architektur der in dieser Arbeit verwendeten FFNN ..... 63
**Abb. 3.4** Ablauf des online Lernens des FFNN-Kopplungsalgorithmus ..... 64
**Abb. 3.5** Vergleich der geschätzten Geschwindigkeit der ersten Masse mit verschiedenen Kopplungsalgorithmen ..... 66
**Abb. 3.6** Vergleich der Position der ersten Masse am Ausgang des Prototyps unter Verwendung verschiedener Kopplungsalgorithmen. Links: Übersicht, Rechts: Ausschnitt ..... 67
**Abb. 3.7** Fokus auf Zeitpunkt des Anschlags am Eingang des zweiten Prototyps. Oben: Position der ersten Masse. Unten: Geschwindigkeit der ersten Masse. ..... 68
**Abb. 4.1** Blockdiagramme der Übertragungsfunktionen für zeitdiskrete (oben) und zeitkontinuierliche (unten) Frequenzanalyse linearer RVPen ..... 73
**Abb. 4.2** Elemente eines RVPs mit Netzwerkeffekten und Kopplungsalgorithmus ..... 74
**Abb. 4.3** Bode-Diagramm des Kopplungsprozesses einer nicht-iterativen Co-Simulation bzgl. des Verhältnisses aus Frequenz und Makroschrittweite ..... 79
**Abb. 4.4** Bode-Diagramm des Kopplungsprozesses einer nicht-iterativen Co-Simulation bzgl. des Verhältnisses aus Frequenz und Makroschrittweite (Ausschnitt) ..... 80

**Abb. 4.5** Bode-Diagramm des Kopplungsprozesses von RVPen mit $k = 3$ bzgl. des Verhältnisses aus Frequenz und Makroschrittweite ..... 81
**Abb. 4.6** Bode-Diagramm des Kopplungsprozesses von RVPen mit $k = 3$ bzgl. des Verhältnisses aus Frequenz und Makroschrittweite (Ausschnitt) ..... 82
**Abb. 4.7** Bode-Diagramm der offenen Kette unter Verwendung verschiedener Kopplungsalgorithmen mit $k = 3$ ..... 85
**Abb. 4.8** Nyquist-Diagramme der Übertragungsfunktion der offenen Kette $GRVP(s)$ für die verschiedenen Kopplungsalgorithmen mit $\Delta T = 0{,}02$ und $k = 3$. Der kritische Punkt ist mittels eines roten Kreuzes, der Punkt $\omega = 0$ mittels eines schwarzen Kreises markiert. ..... 88
**Abb. 4.9** Bode-Diagramm der optimierten Kopplungsalgorithmen bzgl. des Verhältnisses aus Frequenz und Makroschrittweite ..... 94
**Abb. 4.10** Bode-Diagramm der optimierten Kopplungsalgorithmen bzgl. des Verhältnisses aus Frequenz und Makroschrittweite (Ausschnitt) ..... 95
**Abb. 4.11** Bode-Diagramme der offenen Kette unter Verwendung der optimierten Kopplungsalgorithmen mit $k = 3$ ..... 96
**Abb. 4.12** Nyquist-Diagramme der Übertragungsfunktion der offenen Kette $GRVP(s)$ für die optimierten Kopplungsalgorithmen mit $\Delta T = 0{,}02s$ und $k = 3$. Der kritische Punkt ist mittels eines roten Kreuzes, der Punkt $\omega = 0$ mittels eines schwarzen Kreises markiert. ..... 97
**Abb. 5.1** Verfahren zur Bewertung der Extrapolationsgüte nach Stettinger et al. (2017) ..... 112
**Abb. 5.2** Vergleich der Extrapolationsfehler für verschiedene Kopplungsalgorithmen mit der Fehlertoleranz nach Stettinger et al. (2017) (mitte und unten) anhand eines Testsignals (oben) ..... 114
**Abb. 5.3** Vergleich der parametrierten Fehlertoleranzfunktion $etol$ mit dem Extrapolationsfehler vom EROS4 (unten) für ein Testsignal mit der Grenzfrequenz vom EROS4 (oben) ..... 117
**Abb. 5.4** Bewertung der Extrapolation vom ZOH (oben) und FOH (unten) für zwei überlagerte Sinusschwingungen mit der entwickelten Toleranzfunktion ..... 118
**Abb. 5.5** Demonstration der Anwendung des halben Hann-Fensters auf ein verschobenes Testsignal mit Diskontinuität im Vergleich zur Anwendung des Hann-Fensters ..... 123
**Abb. 5.6** Resultierendes Frequenzspektrum für den Signalabschnitt aus Abb. 5.5 und dem Signalabschnitt aus dem vorherigen Makrozeitschritt ..... 124
**Abb. 5.7** Verhalten verschiedener Kopplungsalgorithmen beim Auftreten einer Diskontinuität im Signalverlauf mit $k = 3$ Makroschritten Verzögerung ohne Verwendung des Verfahrens ..... 126
**Abb. 5.8** Verhalten verschiedener Kopplungsalgorithmen beim Auftreten einer Diskontinuität im Signalverlauf mit $k = 3$ Makroschritten Verzögerung mit Diskontinuitäten-Erkennung und Umschalten des Algorithmus ..... 127
**Abb. 6.1** Softwaremodule der entwickelten Methodik zur Kopplung räumlich verteilter realer und virtueller Prototypen sowie deren Wirkzusammenhänge ..... 130
**Abb. 6.2** Ablauf der implementierten Server-Client Kommunikation für das online Lernen eines FFNNs ..... 134

**Abb. 6.3** Schematischer Aufbau des verwendeten RVPs mit Fokus auf den Datenaustausch nach Lachenmaier et al. (2020) ..... 136
**Abb. 6.4** Spektrum des Drehmomentsignals über $\omega\Delta T$ in Prozent der Nyquistfrequenz mit $\Delta T = 0{,}01s$, Fokus auf kleine Amplituden ..... 140
**Abb. 6.5** Vergleich der Ergebnisse der Co-Simulation mit und ohne simulierte Latenzzeiten. Oben: Fahrzeuggeschwindigkeit. Unten: Drehmoment an der Antriebswelle ..... 143
**Abb. 6.6** Bode-Diagramm des optimierten und des gelernten FFNN im Vergleich mit ZOH und EROS3 für $k = 6$ bzgl. des Verhältnisses aus Frequenz und Makroschrittweite ..... 146
**Abb. 6.7** Vergleich der Anwendung vom ZOH mit verschiedenen anderen Kopplungsalgorithmen für das latenzzeitkompensierte Drehmomentsignal des RVPs ..... 148
**Abb. 6.8** Balkendiagramm des summierten kombinierten Kopplungsfehlers nach Sprague und Geers ($\cdot\ 102$) für das Drehmomentsignal des RVPs ..... 149
**Abb. 6.9** Beispiele für die Erkennung von Diskontinuitäten anhand EROS3 für das Drehmomentsignal des RVPs. Oben: Keine Diskontinuitäten erkannt. Unten: Diskontinuitäten erkannt und Kopplungsalgorithmus umgeschaltet. ... 151

# Tabellenverzeichnis


**Tab. 2.1** Systemparameter des Zweimassenschwingers ..... 36
**Tab. 3.1** Übersicht über die Eigenschaften verschiedener linearer Kopplungsalgorithmen ..... 54
**Tab. 3.2** Mittlerer absoluter Fehler bezüglich der Referenzlösung des linearen Zweimassenschwingers für verschiedene Kopplungsalgorithmen ..... 58
**Tab. 3.3** Summierter Kopplungsfehler mit Sprague und Geers (· 102) als Distanzmaß für verschiedene Kopplungsalgorithmen ..... 59
**Tab. 3.4** Summierter mittlerer absoluter Kopplungsfehler bezüglich der Referenzlösung des nichtlinearen Zweimassenschwingers für verschiedene Kopplungsalgorithmen ..... 69
**Tab. 4.1** Grenzfrequenzen der Amplituden- und Phasenbedingung unterschiedlicher Kopplungsalgorithmen in Prozent der Nyquistfrequenz für verschiedene Latenzzeiten $k\Delta T$. Die resultierende Grenzfrequenz $\omega\Delta T = \min(\omega a\Delta T, \omega p\Delta T)$ ist jeweils markiert ..... 83
**Tab. 4.2** Stabilitätsgrenzen von $GRVP(s)$ für die betrachteten Kopplungsalgorithmen ..... 89
**Tab. 4.3** Mittlerer absoluter Fehler bezüglich der Referenzlösung des linearen Zweimassenschwingers für die optimierten Kopplungsalgorithmen ..... 98
**Tab. 4.4** Summierter Kopplungsfehler mit Sprague und Geers (· 102) als Distanzmaß für die optimierten Kopplungsalgorithmen ..... 99
**Tab. 5.1.** Parameter der entwickelten Toleranzfehlerfunktion für verschiedene Kopplungsalgorithmen basierend auf deren Grenzfrequenzen aus Tab. 4.1 für $\Delta T = 0{,}02s$ und $k = 3$ ..... 116
**Tab. 6.1** Summierter Kopplungsfehler mit Sprague und Geers (· 102) als Distanzmaß für verschiedene Kopplungsalgorithmen ..... 149
**Tab. 6.2** Systemverhalten repräsentierende Werte für das unveränderte Drehmomentsignal vom Steuerungssystem des Motorprüfstands, gemessen am Eingang der Kopplungsalgorithmen ..... 150

# Abkürzungsverzeichnis

| | |
|---|---|
| ARMA | Autoregressive-Moving Average |
| ARIMA | Autoregressive Integrated Moving Average |
| Co-Simulation | Coupled-Simulation |
| CPD | Change-Point-Detection |
| DCP | Distributed-Co-Simulation-Protocol |
| DFT | Discrete Fourier Transform |
| EROS | Error Space Extrapolation |
| FFNN | Feedforward Neural Network |
| FFT | Fast-Fourier-Transform |
| FMI | Functional Mock-Up Interface |
| FMU | Functional Mock-Up Unit |
| FOH | First-Order-Hold |
| HIL | Hardware in the Loop |
| LSTM | Long Short-Term Memory |
| MAE | Mean Absolute Error |
| MBSE | Model-Based Systems Engineering |
| MIL | Model in the Loop |
| RNN | Recurrent Neural Network |
| RVP | Real-Virtueller Prototyp |
| SIL | Software in the Loop |
| SMB | Sliding-Mode Beobachter |
| TSN | Time-Sensitive-Networking |
| UDP | User Datagram Protocol |
| UUT | Unit-Under-Test |
| ZOH | Zero-Order-Hold |

# Symbolverzeichnis

| | |
|---|---|
| $a, A$ | Gewichtungsparameter von Kopplungsalgorithmen |
| $A_w$ | Korrekturfaktor der DFT |
| $\beta$ | Sensitivitätsparameter der Diskontinuitäten-Erkennung |
| $C_{S\&G}$ | Kombinierter Fehler nach Sprague und Geers |
| $c, p, d, q$ | Parameter eines ARIMA Modells |
| $\mathcal{D}$ | Definitionsmenge |
| $\Delta$ | Differenzen-Operator |
| $\Delta T$ | Makroschrittweite in $s$ |
| $\delta T$ | Mikroschrittweite in $s$ |
| $\delta(\cdot)$ | Dirac-Funktion |
| $e$ | Kopplungsfehler |
| $\boldsymbol{\Theta}$ | Offsetvektor eines Neurons eines FFNNs |
| $\zeta, \delta$ | Parameter des Verfahrens zur Bewertung der Extrapolationsgüte |
| $f(\cdot)$ | Kopplungsalgorithmus |
| $f_p(\cdot)$ | Kurvenschar |
| $G(\cdot)$ | Übertragungsfunktion |
| $J(\cdot)$ | Kostenfunktion |
| $j$ | Imaginäre Einheit |
| $k$ | Anzahl an Makroschritten Verzögerung |
| $\lambda$ | Designparameter des SMBs |
| $M_{S\&G}$ | Amplitudenfehler nach Sprague und Geers |
| $MAE$ | Mittlerer absoluter Fehler |
| $m$ | Anzahl von Signalwerten |
| $n$ | Beliebige natürliche Zahl |
| $P_{S\&G}$ | Phasenfehler nach Sprague und Geers |
| $\sigma(\cdot)$ | Aktivierungsfunktion eines FFNNs |

| | |
|---|---|
| $r$ | Relativer Grad eines Systems |
| $s$ | Komplexe Bildvariable |
| $\tau$ | Verzögerungszeit in $s$ |
| $t$ | Zeit in $s$ |
| $u$ | Eingangssignal im Zeitbereich |
| $U$ | Eingangssignal im Frequenzbereich |
| $y$ | Ausgangssignal im Zeitbereich |
| $Y$ | Ausgangssignal im Frequenzbereich |
| $\hat{y}$ | Geschätztes Ausgangssignal im Zeitbereich |
| $\hat{Y}$ | Geschätztes Ausgangssignal im Frequenzbereich |
| $\omega$ | Kreisfrequenz in $rad/s$ |
| $\boldsymbol{W}$ | Gewichtungsmatrix eines Neurons eines FFNNs |
| $\boldsymbol{x}$ | Vektor zwischen den Schichten eines FFNNs |

# 1 Einleitung

## 1.1 Motivation

Die Effizienz der Grundfunktionalität vieler etablierter mechatronischer Produkte ist bis nah an die Grenzen der technischen Machbarkeit optimiert. Waschmaschinen waschen mit geringem Strom- und Wasserbedarf und Akkuschrauber sind leicht, zuverlässig und robust. Um weiterhin den wirtschaftlichen Erfolg dieser Produkte zu gewährleisten, entwickeln Unternehmen neuartige Zusatzfunktionen, die ihrer Produkte bei gleicher Grundfunktionalität anwenderfreundlicher gestalten. Diese Innovationen finden meist auf Systemebene durch holistische Betrachtung des Produkts statt, indem Informationen aus verschiedenen Domänen des Produkts kombiniert oder mehrere Produkte vernetzt werden. Beispielsweise können moderne Kühlschränke basierend auf den Lebensmitteln, die sich darin befinden, automatisch eine Einkaufsliste erstellen und diese an das Smartphone des Benutzers schicken. Weiterhin können Wäschetrockner automatisch starten, sobald das Energiesystem des Hauses meldet, dass für das ausgewählte Wasch- und Trockenprogramm aktuell ausreichend grüner Solarstrom von der Fotovoltaikanlage auf dem Dach gewonnen wird.

Das Potential, mit domänenübergreifenden Funktionalitäten neue Geschäftsfelder zu erschließen, ist im Automobilbereich bei der Entwicklung von Kraftfahrzeugen besonders groß. Neuartige Fahrassistenzfunktionen und das hochautomatisierte Fahren sind hierfür Beispiele, da für deren Realisierung ein zuverlässiges und direktes und im Gegensatz zu den oben genannten Beispielen kontinuierliches Zusammenspiel mehrerer Fahrzeugdomänen erforderlich ist, wie unter anderem Fahrdynamik, Lenkung und Bremse. Ebenso ist für die Umsetzung von Hybridantriebskonzepten oder die effektive Verwendung von Diagnosefunktionen eine Interaktion verschiedener Fahrzeugdomänen nötig. Außerdem erleichtern aktuelle Veränderungen in der Automobiltechnik, wie die Elektrifizierung des Antriebsstrangs oder die zunehmende Vernetzung von Fahrzeugen, neuen Fahrzeugherstellern den Einstieg in den Markt. Deren Alleinstellungsmerkmal sind vor allem die angebotenen Services und neuartige Fahrgastraumkonzepte und weniger die funktionellen Bestandteile, z.B. des Fahrwerks, welche sie daher als Gesamtpaket einkaufen möchten. Dadurch steigt die Nachfrage nach vorintegrierten Hardwarekomponenten, in denen mehrere Fahrzeugdomänen interagieren. Beispiele hierfür sind E-Achsen, in denen Elektromotoren, Leistungselektronik und Getriebe kombiniert sind, oder modulare Plattformen des gesamten Fahrgestells.

Für die Entwicklung solcher domänenübergreifenden Funktionalitäten muss das Produkt Kraftfahrzeug bereits ab einer frühen Phase des Entwicklungsprozesses holistisch auf Systemebene betrachtet werden. Das dafür notwendige Einführen neuer domänenübergreifender Entwicklungsmethoden stellt die etablierten Unternehmen der Branche vor große Herausforderungen, da die auf domänenspezifische Komponentenentwicklung ausgerichtete hierarchische Unternehmensstruktur an die domänenübergreifende Produktentwicklung angepasst werden muss.

Auch bewährte Modelle zur Zusammenarbeit zwischen Unternehmen müssen für die Entwicklung domänenübergreifender Funktionalitäten überdacht werden (Bernhart, 2018). Größere funktionaler Abhängigkeiten der Domänen auf Systemebene erschweren die präzise Definition von Anforderungen, auf deren Basis zuliefernde Firmen Hardwarekomponenten entwickeln können. Daher müssen neue Entwicklungsmethoden etabliert werden, die es in allen Phasen des Entwicklungsprozesses ermöglichen, Wechselwirkungen zwischen den von unterschiedlichen Unternehmen entwickelten Domänen zu identifizieren und daraus Anforderungen an die einzelnen Domänen abzuleiten. Gleichzeitig muss das geistige Eigentum der Unternehmen bestmöglich geschützt werden.

### 1.1.1 Der Entwicklungsprozess komplexer mechatronischer Systeme

Das in Abb. 1.1 dargestellte V-Modell der Entwicklungsmethodik für mechatronische und cyber-physische Systeme aus der Richtlinie VDI/VDE 2206 (2021) formalisiert den Entwicklungsprozess mechatronischer Produkte. Demnach werden zuerst Anforderungen an das zu entwickelnde Produkt definiert und auf deren Basis, dargestellt in der linken Seite des V-Modells, ein technischer Entwurf des Gesamtsystems unter Berücksichtigung der notwendigen Interaktion zwischen den technischen Domänen erstellt. Daran schließt der domänenspezifische Entwurf mit der Entwicklung der einzelnen Komponenten des Systems an. Auf der rechten Seite des V-Modells ist die Systemintegration dargestellt, in der das Produkt realisiert wird. Dabei wird verifiziert und validiert, ob das Produkt die zuvor gestellten Anforderungen erfüllt. In der Regel ist dies beim ersten Durchlauf des V-Modells nicht der Fall und der Systementwurf muss entsprechend angepasst werden. Nach jeder Änderung des Entwurfs und daraus folgender Anpassung einer oder mehrerer Komponenten beginnt die Systemintegration von neuem, weshalb es sich beim Entwicklungsprozess mechatronischer Produkte um einen iterativen Prozess handelt.

In den letzten Jahrzehnten wurde die Simulation des technischen Produktverhaltens mittels mathematischer Modelle als wichtige Methode in den Entwicklungsprozess integriert. Virtuelle Prototypen bestehend aus Simulationsmodellen sind in der Regel kostengünstiger realisierbar als reale Prototypen des gesamten Produkts. Außerdem können sie mit weniger Auf-

wand erstellt und an Änderungen des Systementwurfs angepasst werden, sobald eine Verletzung einer Anforderung festgestellt wird. Dies trägt zu einer Verkürzung der Entwicklungszyklen bei, wodurch jede Iteration des Entwicklungsprozesses, die auf Basis von (teilweise) virtuellen Prototypen durchgeführt werden kann, Zeit und Kosten spart.

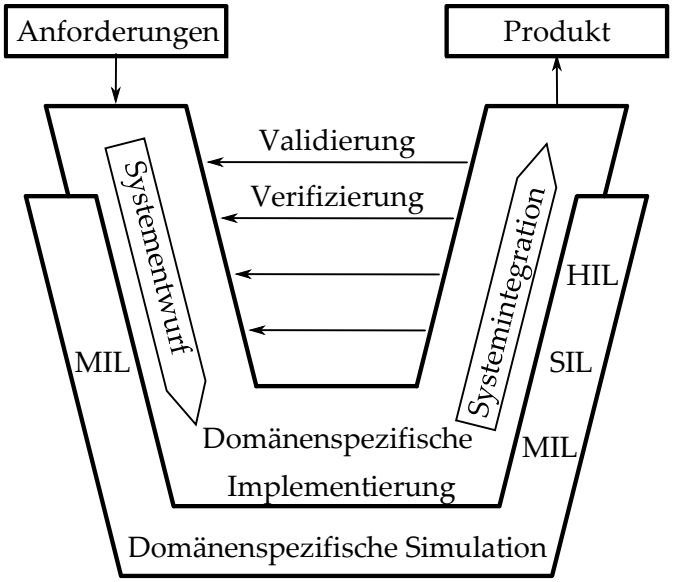


**Abb. 1.1** V-Modell der Entwicklungsmethodik für mechatronische und cyber-physische Systeme in Anlehnung an VDI/VDE 2206 (2021)

Für die Entwicklung domänenübergreifender Funktionalitäten ist die Simulation virtueller Prototypen nicht nur eine Methodik, um Zeit und Kosten zu sparen, sondern eine Notwendigkeit zur Beherrschung der Komplexität des zu entwickelnden Produkts (Hirz et al., 2013). Eine detaillierte Beschreibung des Gesamtsystems mithilfe mathematischer Simulationsmodellen ist aufgrund der hohen Komplexität bereits zu Beginn des Entwicklungsprozesses für die Definition der Anforderungen an die einzelnen Domänen nötig. Unter Anwendung der Methode des Model-Based Systems Engineering (MBSE) (Wymore, 2018) werden durch Simulationen des Gesamtsystems aus den Anforderungen an domänenüberreifende Funktionen plausible Anforderungen an jede einzelne Domänen abgeleitet. Außerdem kann die Erfüllung dieser Anforderungen während der Systemintegration häufig nur virtuell auf Systemebene überprüft werden. Beispielsweise müssen zur Absicherung hochautomatisierter Fahrfunktionen eine große Anzahl Testkilometer gefahren (Wachenfeld und Winner, 2015) und weiterhin gezielt bestimmte kritische Szenarien erzeugt und das Funktionsverhalten reproduzierbar getestet werden (Pütz et al., 2017). Beides ist nur mithilfe virtuell erzeugter Testszenarien durchführbar.

Aus diesen Gründen werden sehr früh im Entwicklungsprozess von Kraftfahrzeugkomponenten Model-In-The-Loop (MIL) (s. Abschnitt 2.1.1) Simulationsumgebungen aufgebaut, indem funktionale Simulationsmodelle aller relevanter Domänen gekoppelt werden. Diese bil-

den das Gesamtsystemverhalten detailliert nach und ermöglichen es, aus Anforderungen domänenübergreifender Funktionalitäten domänenspezifische Anforderungen abzuleiten. In der Phase der Systemintegration, nachdem die einzelnen Komponenten durch domänenspezifische Simulationen entsprechend der Anforderungen angepasst wurden, kann mithilfe der MIL-Simulationen die Erfüllung grundlegender Anforderungen verifiziert werden. Durch die Anbindung von Software (Software-In-The-Loop (SIL), s. Abschnitt 2.1.2) oder Hardware (Hardware-In-The-Loop (HIL), s. Kap.2.1.3) an die Simulation werden einzelne Domänen detaillierter abgebildet und die Erfüllung spezifischerer Anforderungen überprüfbar. Je mehr Funktionalität durch den Einsatz von Simulationen validiert und verifiziert werden kann, desto größer ist die Zeit- und Kostenersparnis in der Produktentwicklung, da sich die Anzahl an aufwendigen Tests am fertigen Produkt reduziert.

Aufgrund des iterativen Ablaufs des Entwicklungsprozesses, bei dem nach Nichterfüllung einer Anforderung eine Anpassung des Systementwurfs notwendig ist, muss gegebenenfalls mehrfach während der Produktentwicklung zwischen MIL-, SIL- und HIL-Simulationen gewechselt werden. Ein nahtloser Übergang von MIL-, SIL- und HIL-Simulationen ist daher ein wichtiger Zeit- und Kostenfaktor und Voraussetzung für einen effizienten Produktentwicklungsprozess. Nahtloser Übergang bedeutet, dass MIL-Simulationsumgebungen auch in der Systemintegration bei SIL- und HIL-Simulationen wiederverwendbar sind. Idealerweise wird beispielsweise beim Übergang von MIL zu SIL ein funktionales Simulationsmodell gegen ein entsprechendes virtuelles Steuergerät ausgetauscht, ohne weitere Änderungen an der Simulationsumgebung oder den anderen Simulationsmodellen vornehmen zu müssen. Dies ermöglicht es außerdem, die meist sehr umfangreiche, zu einer Simulation gehörende, Werkzeugkette wie beispielsweise einen Testszenarienkatalog oder eine Umgebung zur Durchführung kontinuierlicher Tests in SIL- und HIL-Simulationen weiter zu verwenden.

Der nahtlose Übergang von MIL zu SIL wurde in den letzten Jahren bereits in einigen Arbeiten demonstriert. Wissel et al. (2018) haben gezeigt, wie aus funktionalen Modellen einer Motorsteuerung automatisch virtuelle Steuergeräte generiert und in eine MIL-Simulation eingebunden werden können. Weiterhin wurde zusätzlich zu zwei virtuellen Steuergeräten auch deren Kommunikation virtualisiert und in eine MIL-Simulationsumgebung integriert (Baumann et al., 2019c). Forschungsbedarf verbleibt bei der numerisch korrekten Ausführung der durch die Kopplung von kontinuierlichen Modellen und ereignisbasierter Software entstehenden hybriden Co-Simulation (s. Abschnitt 2.1.2).

Die Durchgängigkeit von MIL- bzw. SIL-Simulationen zu klassischen HIL-Simulationen (s. Abschnitt 2.1.3) ist für Integrationstests auf Systemebene dagegen nicht gegeben. Ursache hierfür ist, dass HIL-Simulationen üblicherweise zum Testen einzelner Komponenten des Gesamtsystems eingesetzt werden und dafür optimiert sind. Bei HIL-Simulationen ist meist die angebundene Hardware die Unit-Under-Test (UUT), beispielsweise ein reales Steuergerät

oder ein realer Motor, und die Simulationsmodelle bedienen lediglich die Schnittstellen der Hardware, haben einen niedrigen Detailierungsgrad und ihr Funktionsumfang ist speziell zum Testen der angebundenen Hardware ausgelegt. Außerdem müssen die Simulationsmodelle bei einer HIL-Simulation auf einem Prüfstandsrechner mit Echtzeitbetriebssystem lauffähig sein, weshalb sie harte Echtzeitanforderungen erfüllen müssen. Dies ermöglicht eine zuverlässige und verzögerungsarme Kommunikation zwischen Simulation und angebundener Hardware, erfordert aber gleichzeitig eine Generierung von C-Code aus den Simulationsmodellen. Diese Anforderungen an die Simulation schränken je nach verwendetem Modellierungswerkzeug den Funktionsumfang ein, welcher bei der Modellierung zur Verfügung steht, und erfordern daher eine Erstellung der Simulationsmodelle speziell für deren Anwendung bei der HIL-Simulation (s. Abschnitt 2.1.3).

Für Integrationstests von domänenübergreifenden Funktionalitäten auf Systemebene bei denen einzelne Komponenten des Gesamtsystems bereits als fertige Hardwarekomponente vorliegen, sind klassische HIL-Simulationen somit aufgrund der fehlenden Durchgängigkeit nicht geeignet. Stattdessen müssen dafür, ausgehend von bestehenden detaillierten MIL bzw. SIL-Simulationsumgebungen, einzelne oder mehrere Simulationsmodelle durch einzelne oder mehrere Hardwarekomponenten flexibel ersetzt werden können, ohne weitere Anpassungen an der Simulationsumgebung vornehmen zu müssen.

### 1.1.2 Räumlich verteilte Real-Virtuelle Prototypen

In dieser Arbeit wird eine domänenübergreifende MIL- oder SIL-Simulation, bei der einzelne Simulationsmodelle durch Hardware ersetzt wurden, Real-Virtueller Prototyp (RVP) genannt (mixed real-virtual prototype (Baumann et al., 2019b)). Reale Prototypen sind reale Hardwarekomponenten des zu entwickelnden Produkts, während virtuelle Prototypen die virtuelle Abstraktion der anderen Komponenten als beispielsweise funktionales Simulationsmodell oder simulierter Software-Code bezeichnen. Virtuelle Prototypen werden unverändert aus MIL- oder SIL-Simulation übernommen und auf einem Standard-PC ausgeführt. Die Bezeichnung RVP wurde für dieses Konzept bereits von Eilers und Müller-Schloer (2005) verwendet. Eine genauere mathematische Definition der Kopplung von RVPen erfolgt in Abschnitt 2.2.

Zwei Sachverhalte grenzen einen RVP von der klassischen HIL-Simulation ab: Zum einen ist nicht zwingend die Hardware die UUT, sondern alle gekoppelten Prototypen sind gleichberechtigt, wodurch die Interaktion der verschiedenen Domänen auf Systemebene im Fokus steht. Je nach Anwendungsfall, der mit einem RVP untersucht wird, können mehrere (reale und virtuelle) Prototypen gleichzeitig die UUT sein. Zum anderen ist zur Gewährleistung der Durchgängigkeit im Entwicklungsprozess der Ursprung eines RVPs immer eine Kopplung rein virtueller Prototypen, von denen einzelne durch reale Prototypen ersetzt werden, ohne Veränderungen an den anderen Prototypen vornehmen zu müssen.

Abb. 1.2 zeigt einen RVP, der im Rahmen des öffentlich geförderten Projekts ACOSAR entwickelt wurde (Baumann et al., 2019b). Er demonstriert den Einsatz des Distributed Co-Simulation Protocols (DCP), eines standardisierten Kommunikationsprotokolls (Krammer et al., 2018) für den systematischen Aufbau von RVPen (Krammer, Schiffer und Benedikt, 2021). Obwohl es sich nicht um einen industriell genutzten Aufbau handelt, können an ihm zum einen der Nutzen von RVPen erläutert als auch die Problematik der Kopplung von realen und virtuellen Prototypen analysiert werden, für die in dieser Arbeit Lösungen entwickelt werden.

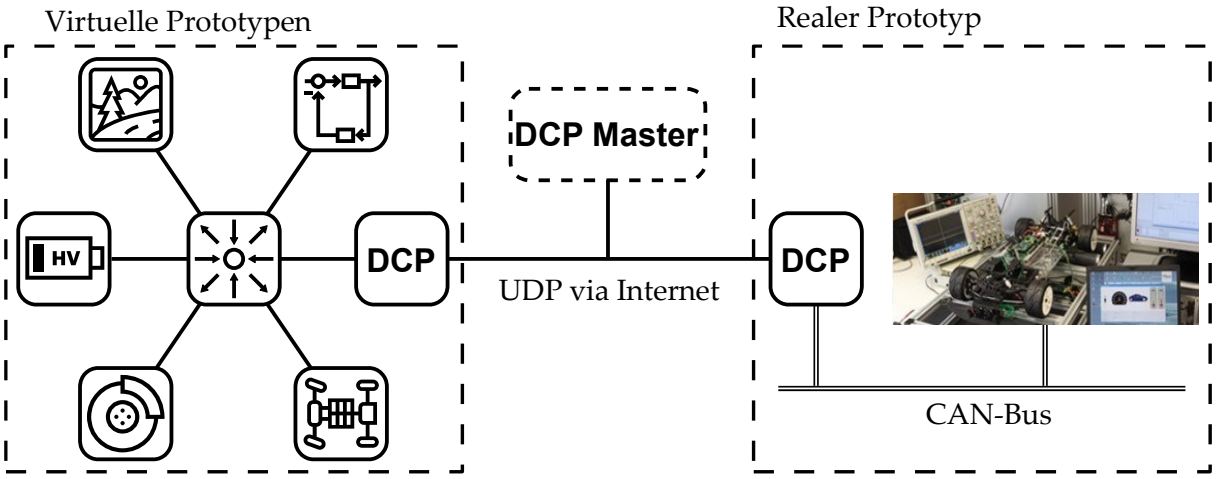


**Abb. 1.2** Kopplung eines realen und mehrere virtueller Prototypen über das Internet (Baumann et al., 2019a)

Auf der linken Seite der Abbildung sind piktografisch fünf Domänen eines batterieelektrischen Fahrzeugs dargestellt. Dabei handelt es sich um die Hochvoltbatterie, den Antriebsstrang, Bremsen und Fahrassistenzfunktionen sowie die Umwelt mit anderen Verkehrsteilnehmern. Sie liegen jeweils als virtueller Prototyp in Form eines detaillierten Simulationsmodells vor, deren jeweilige Modellierungswerkzeuge über eine Co-Simulations-Middleware verbunden sind (s. Abschnitt 2.1.1.). Als Ausgangspunkt des RVPs dient eine MIL-Simulationsumgebung, bei der als sechste Domäne die Fahrdynamik des Fahrzeugs ebenfalls als Simulationsmodell abgebildet war. Der RVP entsteht durch Ersetzen der (longitudinale) Fahrdynamik durch den realen Prototyp eines skalierten Fahrzeugs, welches sich auf einem Rollenprüfstand befindet. Dabei muss keine Anpassungen an den fünf verbliebenen MIL-Simulationsmodellen vorgenommen werden, deren Berechnung weiterhin auf einem Standard-Windows-PC stattfindet. Zur Realisierung des Signalaustauschs zwischen den realen und virtuellen Prototypen bietet sich ein Netzwerkprotokoll wie UDP an (s. Abschnitt 2.2.1), wodurch eine räumliche Verteilung der Prototypen innerhalb eines Netzwerks möglich ist. Die virtuellen Prototypen des hier gezeigten RVPs werden bei der Robert Bosch GmbH in Renningen, Deutschland ausgeführt, während sich der angebundene Prüfstand beim Kompetenzzentrum Virtual Vehicle in Graz, Österreich befindet.

RVPen wie dieser erfüllen die Anforderung der Durchgängigkeit des modellbasierten Entwicklungsprozesses. Reale Prototypen einzelner Komponenten können in bestehende MIL bzw. SIL-Simulationsumgebungen integriert werden, ohne die Modelle anpassen zu müssen. Die UUT bei RVPen ist nicht eine einzelne Komponente oder Regelfunktion. Stattdessen kann aufgrund des hohen Detailierungsgrads aller Domänen die korrekte Interaktion zwischen den Domänen auf Systemebene betrachtet und validiert werden. An dem RVP aus Abb. 1.2 wird beispielsweise ein Szenario untersucht bei dem das reale Fahrzeug, das sich auf dem Rollenprüfstand befindet, in einer virtuellen Umgebung mit virtuellem Verkehr interagiert. Mithilfe eines simulierten adaptiven Tempomats folgt das reale Fahrzeug dem virtuellen Verkehr. Das reale Fahrzeug gibt dabei das Beschleunigungsverhalten vor, während das Modell der Hochvoltbatterie den Ladezustand sowie die Batterietemperatur simuliert. Während einer aktiven Verzögerung des Fahrzeugs interagiert das Antriebsstrangmodell außerdem mit dem Modell der Bremsen zur Berechnung der rekuperierten Energie.

Die Kommunikation der Prototypen zur Laufzeit über ein Rechnernetzwerk erhöht die Flexibilität des RVPs, da dadurch eine räumliche Verteilung der Prototypen ermöglicht wird. Zum einen erlaubt dies die Anbindung mehrerer realer Prototypen, die aufgrund ihrer Größe und der komplexen zugehörigen Prüfeinrichtung an ihren Standort gebunden sind. Auf diese Weise können mehrere Domänen des Gesamtsystems als realer Prototyp abgebildet und der Reifegrad der Testumgebung erhöht werden. Zum anderen erleichtert und stärkt die Verwendung räumlich verteilter RVPen die unternehmensübergreifende Zusammenarbeit. Diese ist bei der Entwicklung von Produkten mit domänenübergreifenden Funktionalitäten ein wichtiger Aspekt, da aufgrund der Komplexität häufig mehrere Unternehmen gemeinsam an der Entwicklung einzelner Komponenten des Produkts arbeiten, wie es in der Automobilindustrie oder der Luft- und Raumfahrtechnik üblich ist. Durch Verwendung von RVPen kann eine Systemintegration durchgeführt werden, ohne dass zusammenarbeitende Unternehmen (reale oder virtuelle) Prototypen ihrer Produktkomponenten frühzeitig austauschen müssen. Das geistige Eigentum der Unternehmen wird geschützt, indem die realen und virtuellen Prototypen jeweils bei den Unternehmen verbleiben und lediglich Informationen über deren funktionale Schnittstellen geteilt werden müssen. Dadurch sind außerdem notwendige Anpassungen an den Prototypen schneller und effizienter möglich. Weiterhin erlaubt eine unternehmensübergreifende Prototypenkopplung die Aufteilung von Rechnerressourcen zwischen den Unternehmen, wodurch beispielsweise für die rechenintensive Validierung hochautomatisierter Fahrfunktionen, Simulationsserver mehrerer Unternehmen parallel genutzt werden können. Das öffentlich geförderte Projekt UPSIM beschäftigt sich mit diesen und weiteren Möglichkeiten zur Stärkung der unternehmensübergreifenden Zusammenarbeit durch den kollaborativen Einsatz simulationsbasierter Entwicklungsmethoden (ITEA3, 2020).

### 1.1.3 Problematik der Kopplung Real-Virtueller Prototypen

Obwohl RVPen, wie in Abschnitt 1.1.2 gezeigt, zu einem durchgängigen und effizienten modellbasierten Entwicklungsprozess führen, ist deren Verwendung in der Industrie noch nicht gleichermaßen etabliert wie die MIL- bzw. SIL-Simulationen, aus denen sie hervorgehen. Eine Ursache dafür ist, dass beim Signalaustausch zur Laufzeit von gekoppelten Prototypen, bei denen mindestens ein Prototyp real und damit ein Echtzeitsystem ist, unweigerlich Fehler auftreten. Bei dem in Abb. 1.2 dargestellten RVP wird dies durch Betrachtung der Schnittstelle zwischen Antriebsstrang und Fahrdynamik deutlich. Während diese beiden Komponenten im finalen Produkt Fahrzeug physisch miteinander verbunden sind und unmittelbar miteinander wechselwirken, werden bei diesem RVP die Koppelgrößen (z.B. leistungsbestimmende Größen wie Drehzahl und Moment an der Antriebswelle) zwischen den Prototypen über das Internet ausgetauscht. Die Verzögerungszeit $\tau$ dieser Kommunikation, auch Latenzzeit genannt, verfälscht das Gesamtsystemverhalten. Aufgrund von bilateralen Abhängigkeiten der gekoppelten Systeme kann eine zu hohe Umlaufzeit zur Instabilität führen. Die Umlaufzeit zwischen zwei gekoppelten Prototypen bezeichnet in dieser Arbeit die Zeit, die vergeht, bis ein Prototyp auf ein von ihm gesendetes Signal ein Feedback von dem anderen Prototyp erhält. Weiterhin können, wie in Abb. 1.3 dargestellt, Datenpakete bei der Kommunikation verloren gehen. Außerdem unterliegen die virtuellen Prototypen Echtzeitbedingungen, da sie mit realen Prototypen gekoppelt werden (vgl. Abschnitt 2.2.2). Ein dritter Grund, der den Einsatz von RVPen einschränkt, ist die Realisierung und Konfiguration der technischen Verbindung der Prototypen. Diese ist aufgrund der räumlichen Verteilung der Prototypen und der Tatsache, dass gegebenenfalls verschiedene Organisationseinheiten für die Prototypen zuständig sind, kompliziert.

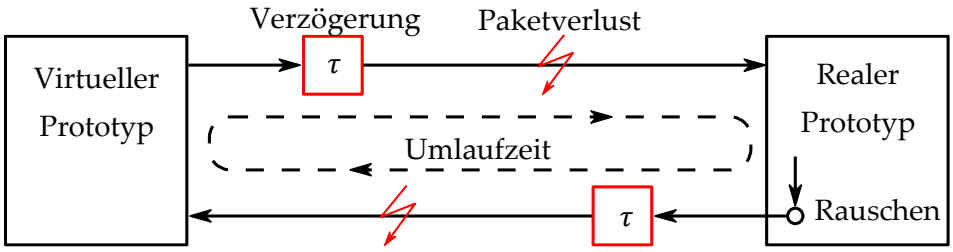


**Abb. 1.3** Netzwerkeffekte in einem RVP nach (Stettinger et al., 2017)

Durch Verwendung des Distributed-Co-Simulation-Protocols (DCP) (Krammer et al., 2018) lässt sich der Konfigurations- und Realisierungsaufwand von RVPs deutlich verringern. Anstatt für jeden RVP eine eigene, proprietäre Lösung für die Kommunikation und den Datenaustausch entwickeln zu müssen, kann die Datenübertrag nun mittels des DCP-Standards implementiert werden. Neben der Standardisierung der verschiedenen Phasen des Ausführens eines RVPs ist auch die Struktur der versendeten Nachrichten vereinheitlicht. Zusätzlich ist

ein Prozess vorgeschlagen, der den unternehmensübergreifenden Aufbau von RVPs erleichtert, wodurch dieser zeit- und kosteneffizient möglich ist. Krammer, Ferner und Watzenig (2019) erweitern das DCP um eine Zeitsynchronisation.

Auf die Herausforderung der Echtzeitbedingungen, welche an die virtuellen Prototypen gestellt werden, wird in Abschnitt 2.2.2 nochmals eingegangen, aber aufgrund immer schnellerer Simulationshardware sowie die parallele Architektur gekoppelter virtueller Prototypen verhindert dies nicht die Verwendung von RVPen.

Somit ist der aus dem Signalaustausch resultierende Kopplungsfehler (vgl. Abschnitt 2.2.3) als größte Herausforderung für den Einsatz von RVPen anzusehen. Schreiber et al. (2018) analysieren die Netzwerkeffekte, die den Signalaustausch beeinträchtigen und kommen zu dem Schluss, dass die Latenzzeit der Signalübertragung die größte Auswirkung auf das gekoppelte System hat und durch den Einsatz eines Kopplungsalgorithmus kompensiert werden muss, um einen aussagekräftigen Betrieb des RVPs zu gewährleisten.

## 1.2 Einordnung der Arbeit in den wissenschaftlichen Kontext

In dieser Arbeit wird ein Kopplungsalgorithmus zur Kompensation von Netzwerkeinflüssen in RVPs entwickelt. Zur Einordnung des Anwendungsgebietes des neuartigen Kopplungsalgorithmus werden im ersten Teil dieses Kapitel die verschiedenen Einsatzfelder räumlich verteilter RVPen vorgestellt. Besonderes Augenmerk liegt dabei auf verteilten Prototypen, die im Entwicklungsprozess von Kraftfahrzeugkomponenten Verwendung finden, da der in dieser Arbeit entwickelte Kopplungsalgorithmus an einem RVP aus diesem Umfeld untersucht wird. Im weiteren Verlauf werden andere Anwendungsgebiete vorgestellt, bei denen Systeme miteinander unter ähnlichen Randbedingungen gekoppelt sind, weshalb Latenzzeiten dort auch auftreten und die entwickelte Methodik Anwendung finden kann.

Im zweiten Teil dieses Unterkapitels wird der Stand der Technik bezüglich der Kompensation des Kopplungsfehlers in kommunizierenden Systemen vorgestellt. Dabei finden sowohl Algorithmen Beachtung, die speziell für die Anwendung auf räumlich verteilte RVPen entwickelt wurden als auch Methoden aus verwandten Bereichen.

### 1.2.1 Einsatzfelder Real-Virtueller Prototypen

In der Literatur finden sich einige Anwendungsfälle aus unterschiedlichen Branchen bei denen RVPen oder ähnliche Konzepte die modellbasierte Entwicklung und Validierung komplexer mechatronischer Produkte unterstützen.

In der Automobilbranche etabliert Düser (2010) den Begriff X-In-The-Loop (XIL) als modellbasiertes Entwicklungskonzept auf Systemebene unter dem die klassischen modellbasierten

Entwicklungskonzepte MIL, SIL und HIL zusammengefasst sind (Tibba et al., 2016). Düser (2010) implementiert eine flexible XIL-Simulationsumgebung, die sowohl Schnittstellen zu Simulationsmodellen als auch zu realen Komponenten wie Prüfständen anbietet und für eine Verwendung in allen Phasen des Produktentwicklungsprozesses ausgelegt ist. Bei XIL ist die UUT üblicherweise ein Modell, eine Software oder eine reale Hardware. Da der Begriff XIL außerdem Anwendungsfälle umfasst, bei denen die UUT nicht eine einzelne Domäne, sondern das gesamte Produkt auf Systemebene ist (Albers et al., 2013), sind auch RVPen unter der Bezeichnung XIL zusammenzufassen. Das Konzept Test-Rig-In-The-Loop, welches von Augsburg et al. (2011) eingeführt wurde, ist ebenfalls als eine Variante von XIL anzusehen. Dabei werden mehrere Prüfstände verschiedener Domänen miteinander verbunden und als Prüfstandsnetzwerk betrieben. Eine räumliche Verteilung dieser Testumgebungen durch eine Kopplung von Prüfständen über das Internet ist eine logische Erweiterung eines lokalen Prüfstandnetzwerks und wurde bereits vor mehr als zehn Jahren untersucht (Fathy et al., 2006, Ersal et al., 2009). Weitere Arbeiten neueren Datums zeigen Anwendungsfälle mit höherer Komplexität. Wenxu, Song und Zhang (2017) verbinden Komponenten über eine sehr große Distanz, indem sie einen Fahrsimulator in Deutschland betreiben, dessen Fahrdynamik Modell auf einem PC in China berechnet wird und Zhang et al. (2018) betreiben ein Netzwerk aus drei Prüfständen zur Untersuchung eines hybriden Antriebskonzepts. Verschiedene Architekturen von Prüfstandsnetzwerken für die Entwicklung von elektrischen Fahrzeugen sind außerdem in Ivanov et al. (2019) und Alfonso et al. (2022) dargestellt. Es ist geplant ein europaweites Netzwerk aus fünf Prüfständen aufzubauen. Seit Einführung des oben erwähnte DCP-Standards wird dieser häufig für die Konfiguration der Datenübertragung von RVPen genutzt. Neben anderen verwenden es Rautenberg et al. (2023) zur Kopplung eines Prüfstands einer elektrischen Maschine und einem Verbrennungsmotorenprüfstands.

Neben Prüfständen gibt es noch andere Echtzeitsysteme, die mit Simulationsmodellen verbunden werden und Echtzeitbedingungen an diese stellen. Beispielsweise kann der Mensch mithilfe eines Fahrsimulators (Maas et al., 2014) oder über augmented reality (Albers et al., 2019) mit Simulationsmodellen interagieren. Diese gekoppelten Systeme werden ebenfalls durch Netzwerkeffekte beeinflusst und es entstehen Kopplungsfehler.

Die in Abschnitt 1.1.1 beschriebene notwendige Durchgängigkeit von rein virtuellen Entwicklungskonzepten wie MIL und SIL bis hin zu RVPen ist Gegenstand aktueller Forschung. Klein et al. (2017) zeigen diese Durchgängigkeit, indem sie ausgehend von einer domänenübergreifenden MIL-Simulation das physikalische Modell eines Verbrennungsmotors durch einen Motorprüfstand ersetzen. In weiteren Arbeiten ergänzen die Autoren ihre Testumgebung mit einer elektrischen Domäne, sodass sie ein Mild-Hybrid Fahrzeug repräsentiert (Klein et al., 2018a, Klein et al., 2018b). In einer Variante binden sie anstatt des Motorprüfstands einen Bordnetzprüfstand an die Testumgebung an (Xia et al., 2017). Für die Realisierung des RVPs müssen die Simulationsmodelle bei Klein et al. (2017) jedoch auf einem Echtzeit-Prüfstandsrechner

zur Ausführung gebracht werden, wodurch eine Kompilation der Modelle notwendig ist. Zehetner et al. (2014) dagegen verbinden direkt die offline genutzten MIL-Simulationsmodelle mit einem Motorprüfstand ohne die ausführende Hardware für die Simulationsumgebung ändern zu müssen. Die verwendete Methode zur Kopplung von realen mit virtuellen Prototypen nennen sie Echtzeit-Co-Simulation (vgl. Abschnitt 2.2) und betonen die Notwendigkeit der Latenzzeitkompensation in RVPen (Stettinger et al., 2013).

Wie oben erwähnt können RVPen auch aus SIL-Simulationen durch die Anbindung von realen Prototypen entstehen. Himmler (2014) zeigt diese Durchgängigkeit von einer SIL-Simulation zu einem RVP, indem er reale Steuergeräte mit der Simulation verbindet. Im Unterschied zur Ankopplung eines Prüfstands einer physikalischen Komponente an die Simulation wird dadurch keine physikalische Verbindung zwischen den Komponenten der realen und virtuellen Prototypen aufgebrochen. Stattdessen erfolgt die Kopplung über Signale, die auch im Betrieb des Fahrzeugs über ein Bussystem, bei dem ebenfalls Latenzen auftreten, ausgetauscht werden. Daher ist in diesem Fall der Umgang mit Netzwerkeffekten in der Steuergerätesoftware selbst zu implementieren, wodurch kein zusätzlicher Kopplungsalgorithmus zur Kompensation der Latenzzeiten notwendig ist. Wie in Abschnitt 2.1.2 beschrieben, ist die Herausforderung bei SIL-Simulationen die Modellierung des Bussystems, um die bei der Signalübertragung zwischen Steuergeräten auftretenden Netzwerkeffekte bereits während der Entwicklung der Software berücksichtigen zu können.

In anderen Branchen ist eine Durchgängigkeit von virtuellen Entwicklungskonzepten zu Kopplungen mit Echtzeitsystemen ebenfalls ein wichtiger Zeit- und Kostenfaktor in der Produktentwicklung. Sadjina et al. (2019) präsentieren eine Pilotstudie aus dem maritimen Umfeld, bei dem ein Arduino UNO die Positionsregelung eines simulierten Containerschiffs in Echtzeit übernimmt. Auch intelligente Stromnetze sind Systeme mit einer hohen Komplexität, bei denen räumlich verteilte RVPs beispielsweise zum Validieren von realen Relais eingesetzt werden können, sofern die Herausforderung durch die auftretenden Latenzzeiten gelöst ist (Steinbrink et al., 2017).

Nicht nur während der Entwicklung, sondern auch im Betrieb technischer Produkte gibt es Anwendungen in denen reale Prototypen mit virtuellen Prototypen kommunizieren und unter Echtzeitbedingungen Signale austauschen. Eine zunehmende Anzahl an Produkten wird in naher Zukunft im Betrieb mit dem Internet (vgl. Internet Of Things (IOT)) verbunden sein und in Echtzeit von einem so genannten digitalen Zwilling überwacht. Ein digitaler Zwilling ist hierbei die virtuelle Repräsentation des Produkts, welche einen identischen funktionalen Umfang besitzt (Boschert und Rosen, 2016). Neben der Überwachung kann der digitale Zwilling auch rechenaufwendige Optimierungen der Betriebsabläufe des Produkts übernehmen. Allerdings sind diese Funktionalitäten typischerweise nicht zeitkritisch, wodurch Latenzzeiten unkritisch sind und für diese Anwendung nicht kompensiert werden müssen. Kritischer

sind Latenzzeiten im Betrieb dagegen im Bereich der Networked Control Systems, bei denen Regelstrecke und Regelungsalgorithmus über ein Netzwerk kommunizieren (Baillieul und Antsaklis, 2007). Ein Beispiel hierfür ist die Teleoperation (Lawrence, 1993), bei der Menschen die Rolle des Regelungsalgorithmus übernehmen und Handlungen über Distanz ausführen. Dadurch ist eine Fernsteuerung von Fahrzeugen oder eine medizinische Operation über Distanz möglich. Teleoperationen sind für den Menschen intuitiver durchzuführen, wenn die Latenzzeiten von einem Algorithmus kompensiert werden (Lu et al., 2019).

Sind ausschließlich virtuelle Prototypen über eine größere Distanz beispielsweise über das Internet miteinander verbunden, spielen Latenzzeiten dagegen keine Rolle. Virtuelle Prototypen wie in Hines und Borriello (1997) müssen keine Echtzeitbedingungen einhalten und können daher ihre Berechnungen unterbrechen und abwarten bis sie Signale der anderen virtuellen Prototypen empfangen. Fujimoto (2015) verbindet Simulationsmodelle über eine Cloud, ein Ansatz der steigende Aufmerksamkeit erhält, da sich auf diese Weise Rechenressourcen effektiv nutzen lassen. Für die Kopplung rein virtueller Prototypen kann sogar bereits auf standardisierte Implementierungen wie FMU-proxy zurückgegriffen werden (Hatledal et al., 2019).

### 1.2.2 Kompensation von Kopplungsfehlern

In Abschnitt 1.1.3 ist gezeigt, dass der unweigerlich auftretende Kopplungsfehler das größte Hindernis für die Verwendung von RVP ist, da dieser das Systemverhalten verfälscht und im schlimmsten Fall zur Instabilität führen kann. Nach Schreiber et al. (2018) ist der Kopplungsfehler die Folge mehrerer Netzwerkeffekte, wie Latenzzeiten und der daraus resultierenden Umlaufzeit, Nachrichtenverluste bei der Signalübertragung sowie Rauscheinflüsse in Messeinrichtungen. In der Literatur findet sich eine Vielzahl von Kopplungsalgorithmen, die den Einfluss einzelner oder mehrere dieser Effekte kompensieren und so den Kopplungsfehler minimieren.

Hier werden solche Kopplungsalgorithmen vorgestellt, die speziell für den Einsatz in RVPen entwickelt wurden. Diese kompensieren die Netzwerkeffekte direkt auf Basis der ausgetauschten Signale, sodass die gekoppelten Prototypen von den Netzwerkeffekten unabhängig sind und nicht angepasst werden müssen. Dadurch ist die durchgängige Verwendung derselben Prototypen in allen Phasen des Entwicklungsprozesses möglich (Stettinger et al., 2017).

Die Auslegung einiger veröffentlichter Kopplungsalgorithmen zur Kompensation von Netzwerkeffekten in RVPen greift auf exakte mathematische Modelle aller gekoppelten Prototypen zurück. Dazu zählt die Arbeit von Gan, Todd und Apsley (2014), in der die auftretenden Latenzzeiten mit einem Smith-Prädiktor kompensiert werden. Bei einem Smith-Prädiktor wird einer der Prototypen des RVPs parallel zur Kopplung mit einem anderen Prototyp lokal mit einem mathematischen Modell desselben verbunden. Dadurch kann das Verhalten des geschlossenen Systems ohne Latenzzeiten vorhergesagt und somit der zukünftige Verlauf der

Koppelgrößen geschätzt werden. Cale et al. (2018) dagegen implementieren für jeden der gekoppelten Prototypen ihres RVPs einen Luneberger Beobachter, um den Einfluss variabler Latenzzeiten zu mildern. Die Autoren weisen darauf hin, dass die Güte der verwendeten Modelle einen großen Einfluss auf die Performanz der Kompensation hat. Weiterhin findet die modellprädiktive Regelung als Kopplungsalgorithmus zur Kompensation von Latenzzeiten Anwendung (Bemporad, 1998, Holiš, Bobál und Vojtěšek, 2017).

Das von Stettinger et al. präsentierte modellbasierte Kopplungselement zur Kompensation von Latenzzeiten und Rauscheinflüssen in kontinuierlichen Koppelsignalen von RVPen benötigt keine mathematischen Modelle der gekoppelten Prototypen. Die Kompensation der Netzwerkeffekte erfolgt durch Extrapolation der Koppelsignale mittels linearer Systeme zweiter Ordnung. Diese bilden jeweils das lokale Übertragungsverhalten eines Eingangs-Ausgangspaares eines gekoppelten Prototyps ab. Dazu werden die Parameter der linearen Systeme zur Laufzeit auf Basis eines erweiterten Kalmanfilters identifiziert. Die Aktivierung der Extrapolation erfolgt nach Abschluss der Identifikationsphase (Stettinger et al., 2014b). Stettinger et al. (2015) zeigen die Anwendung auf ein geregeltes Airball-System, während in Stettinger et al. (2014a) die Methode auf einen geregelten Motorprüfstand angewandt wird. Um das modellbasierte Kopplungselement verwenden zu können, muss die zu kompensierende Latenzzeit bekannt und zeitinvariant sein. Tranninger et al. (2016) zeigen wie durch Einsatz eines Puffers zeitvariante Latenzen in zeitinvariante umgewandelt werden können, wodurch die Methode auch bei variablen Umlaufzeiten nutzbar ist. Stettinger et al. (2016) zeigen anhand eines geregelten Magnetschwebemoduls, dass das modellbasierte Kopplungselement auch einen Paketverlust bei der Kommunikation der Prototypen ausgleichen kann, indem fehlende Nachrichten durch Extrapolation mittels der identifizierten Systeme approximiert werden. Weiterhin führt diese Arbeit einen Stabilitätsbeweis unter Berücksichtigung des modellbasierten Kopplungselements durch. Allerdings weisen die Autoren darauf hin, dass die Stabilitätsaussage aufgrund einer vorgenommenen Linearisierung der nichtlinearen Systemgleichungen des Magnetschwebemoduls nur lokal gültig ist. Die vorgestellten Arbeiten zum modellbasierten Kopplungselement führen viele verschiedene Parameter ein, die für die Identifikationsphase als auch für die Extrapolation und deren Aktivierung eingestellt werden müssen. Tranninger et al. (2018) geben einen Überblick über die verschiedenen Parameter und implementieren einen Algorithmus, um diese entsprechend des gewünschten Kompensationsverhaltens einzustellen.

Ersal et al. (2014) verfolgen für die Kompensation der Netzwerkeffekte den Ansatz des iterativen Lernens unter Verwendung einer Ansatzfunktion, die der Struktur eines PD-Reglers ähnlich ist. Durch mehrmaliges Ausführen des RVPs wird mithilfe dieser Funktion der Verlauf jedes Eingangssignals in einem iterativen Vorgehen so gewählt, dass der Kopplungsfehler (vgl. Abschnitt 2.2) minimiert wird. Das Einstellen der benötigten Identifikationsparameter wird in Ge et al. (2014) diskutiert.

Von Stettinger et al. (2017) wird ein Kopplungsalgorithmus vorgeschlagen, der ebenfalls die lokale Dynamik jedes Koppelsignals eines RVPs durch eine Funktion abbildet. Die Funktionsstruktur entspricht in diesem Fall der eines Finite Impulse Response (FIR) Filters 4.Ordnung, dessen Filterkoeffizienten mittels eines Recursive-Least-Squares-Algorithmus mit Vergessensfaktor identifiziert werden. Die Autoren weisen darauf hin, dass der vorgeschlagene Algorithmus ausschließlich für kontinuierliche Koppelsignale entworfen ist und nicht für Signale, die beispielsweise als Folge von Ereignissen wie Schaltvorgängen in einem der gekoppelten Prototypen Diskontinuitäten aufweisen.

Ein weiterer Kopplungsalgorithmus zur Kompensation von Latenzzeiten in RVPen, der keine mathematischen Modelle der gekoppelten Prototypen benötigt, wird erstmals in Tandon et al. (2013) erwähnt. Dabei wird für jeden der gekoppelten Prototypen ein Sliding-Mode Beobachter (SMB) entworfen, mit dessen Hilfe der zukünftige Verlauf der Koppelgrößen extrapoliert wird. Die Reglerverstärkung ist der einzige Parameter dieses Kopplungsalgorithmus und beeinflusst die Stabilität des Verfahrens abhängig von der zu kompensierenden Latenzzeit. Ge et al. (2017) stellen in einer Stabilitätsanalyse des Beobachters im Frequenzbereich fest, dass dieser Kopplungsalgorithmus vor allem zur Kopplung von Prototypen geeignet ist, die ein natürliches Tiefpassverhalten ausweisen. Das Verfahren wird von Zheng et al. (2018) für die Anwendung auf RVPen mit variablen Latenzzeiten erweitert.

Liu et al. (2020) vergleichen die Methode von Tandon et al. (2013) mit einem Kopplungsalgorithmus auf Basis eines PD-Reglers, der den zukünftigen Signalverlauf über die erste Ableitung vorhersagt. Sie stellen fest, dass der PD-Regler vor allem zu Beginn der Laufzeit des RVPs einen kleineren Kopplungsfehler verursacht als der Algorithmus von Tandon et al. (2013).

Weiterhin gibt es Kopplungsalgorithmen, die zwar keine mathematischen Modelle der gekoppelten Prototypen benötigen, aber auf eine bestimmte Anwendung ausgelegt sind. Guillo-Sansano et al. (2014) stellen beispielsweise ein Verfahren vor, um den Verlauf überlagerter harmonischer Schwingungen zu extrapolieren. Die Autoren nutzen die Tatsache, dass die beiden Koppelgrößen sinusförmig sind und keinen Trend oder Versatz aufweisen. Das Verfahren kann zur Kopplung von Prototypen verwendet werden, welche die elektrischen Größen Spannung und Stromstärke austauschen.

Auch in der Teleoperation werden Charakteristiken der in jedem Anwendungsfall identischen Koppelgrößen Kraft und Geschwindigkeit (bzw. Moment und Winkelgeschwindigkeit) für die Kompensation der Latenzzeiten genutzt. Yokokohji, Imaida und Yoshikawa (1999) transformieren die Koppelgrößen in trendfreie sinusförmige Signale und nutzen für die Latenzzeitkompensation aus, dass diese Signale in jedem Fall kontinuierlich sind. In neueren Arbeiten werden Beobachter für die Latenzzeitkompensation in Teleoperationen eingesetzt, welche auf

physikalische Gesetzmäßigkeiten der definierten Koppelgrößen Kraft und Geschwindigkeit zurückgreifen (Natori et al., 2009, Shimmyo et al., 2019).

Bei Networked Control Systems ist die Anforderung nicht wie in den bisher vorgestellten Bereichen den Einfluss der Netzwerkeffekten durch den Einbau zusätzlicher Methoden zu kompensieren, sondern stattdessen die Regelung so zu entwerfen, dass sie auch unter Netzwerkeffekten stabil ist. Naghshtabrizi und Hespanha (2005) legen ihre Reglung unter Verwendung eines Modells der Netzwerkeffekte aus, um deren asymptotische Stabilität zu gewährleisten. In Liu et al. (2015) ist eine Bedingung für die exponentielle Stabilität von Networked Control Systems unter variablen Verzögerungszeiten hergeleitet.

## 1.3 Wissenschaftlicher Beitrag und Aufbau der Arbeit

Für eine zeit- und kosteneffiziente Entwicklung domänenübergreifender Funktionen in komplexen mechatronischen Produkten müssen Simulationsmodelle durchgängig in allen Phasen des Entwicklungsprozesses verwendbar sein. Dazu ist es notwendig, Simulationsmodelle direkt, ohne Anpassungen an ihnen vornehmen zu müssen, mit bereits vorliegenden Hardwarekomponenten des Produkts zu einem Real-Virtuellen Prototyp (RVP) zusammenschließen zu können. Der aufgrund von Netzwerkeffekten auftretende Kopplungsfehler verfälscht das Systemverhalten von RVPen und verhindert dadurch ihren flächendeckenden industriellen Einsatz.

Der wissenschaftliche Beitrag dieser Arbeit ist daher die Entwicklung einer Kopplungsmethodik, die den vermehrten Einsatz von RVPen in der Produktentwicklung ermöglicht, indem sie den Kopplungsfehler mittels einer Kompensation der dominanten Netzwerkeffekte während des Datenaustauschs von RVPen verringert. Durch Einsatz dieser Kopplungsmethodik soll das Systemverhalten von RVPen idealerweise so sein, als wären keine Netzwerkeffekte vorhanden. Im Gegensatz zu allen anderen bekannten Verfahren soll die entwickelte Kopplungsmethodik auch dann verwendbar sein, wenn die Koppelsignale Diskontinuitäten aufweisen. Weiterhin soll eine Analyse des entwickelten Kopplungsalgorithmus erfolgen, damit dessen Einfluss auf das Systemverhalten eines RVPs im Voraus abschätzbar ist.

Um eine Anwendung der Kopplungsmethodik mit geringem Aufwand zu gewährleisten, soll sie im Gegensatz zu den im Bereich der Networked Control Systems häufig angewandten Methoden unabhängig der gekoppelten Prototypen sein und darf nicht auf deren Anpassung beruhen. Zur Laufzeit muss die Kompensation der Netzwerkeffekte daher ausschließlich auf Basis der Koppelsignale und messbarer Informationen zu den Netzwerkeffekten stattfinden. Weiterhin soll die Kopplungsmethodik in RVPen aus allen Branchen anwendbar sein und nicht auf eine bestimmte Anwendung oder eine spezielle Klasse von Koppelsignalen (z.B.

trendfreie harmonische Schwingung) beschränkt sein. Außerdem soll die Methodik kein mathematisches Modell der gekoppelten Prototypen benötigen, da besonders bei unternehmensübergreifenden RVPen nicht gewährleistet werden kann, dass White-Box Modelle aller Prototypen vorliegen.

Die in Abschnitt 1.2.2 vorgestellten Kopplungsalgorithmen von Ersal et al. (2014), Stettinger et al. (2014b) und Stettinger et al. (2017) erfüllen zwar ebenfalls diese Anforderungen bezüglich der generellen Anwendbarkeit, verwenden zur Parametrierung der Kompensation allerdings eine Systemidentifikation zur Laufzeit des RVPs. Dies ist ein Nachteil, da es zum einen erfordert, dass der RVP auch ohne eine Kompensation der Netzwerkeffekte stabil ist und zum anderen eine mehrmalige Ausführung des RVPs notwendig ist, bis der Kopplungsalgorithmus eingesetzt werden kann. Die in dieser Arbeit entwickelte Kopplungsmethodik soll dies umgehen, indem sie parametriert werden kann, ohne den RVP dafür ausführen zu müssen. Außerdem soll eine Anpassung des Kopplungsalgorithmus zur Laufzeit auf sich gegebenenfalls änderndes Systemverhalten des RVPs möglich sein. Um den Nutzen der entwickelten Kopplungsmethodik sicherzustellen, soll sie auf eine Weise implementiert werden, die eine breite Anwendung ermöglicht und an einem in der Industrie verwendetem RVP demonstriert werden.

Im nächsten Kapitel werden die notwenigen Grundlagen für die Entwicklung der Kopplungsmethodik geschaffen, indem RVPen definiert, von anderen gekoppelten Prototypen abgegrenzt und allgemein inklusive ihrer Netzwerkeffekte analysiert werden. Außerdem werden dort Beispielsysteme und Vergleichsmetriken vorgestellt, die an mehreren Stellen dieser Arbeit zu Analysezwecken verwendet werden. In Kap. 3 wird ein Verfahren zur Extrapolation im Fehlerraum entwickelt, aus dem sich mehrere Kopplungsalgorithmen ableiten lassen, die das Kernstück der Kopplungsmethodik bilden. Diese werden mit bestehenden Algorithmen aus der Literatur verglichen und an Beispielsystemen erprobt. Weiterhin werden vorwärts gerichtete, künstliche neuronale Netze verwendet, um eine Anwendung der entwickelten Algorithmen auf nichtlineare Systeme sowie deren Lernfähigkeit zu ermöglichen. Eine Analyse im Frequenzbereich inklusive Stabilitätsuntersuchungen anhand eines Beispielsystems erfolgt in Kap. 4 für alle untersuchten Kopplungsalgorithmen. Basierend auf dieser Analyse wird eine Methode entwickelt, um Kopplungsalgorithmen zur Kompensation gegebener Netzwerkeffekte optimal auszulegen. Anschließend werden in Kap. 5 Methoden aus der Literatur erweitert bzw. neu entwickelt, um zum einen Kopplungsalgorithmen online während der Laufzeit eines RVPs zu bewerten und zum anderen diese auch für Koppelsignale, die Diskontinuitäten aufweisen, anwendbar zu machen. Weiterhin ist in diesem Kapitel aufgezeigt, welche Aspekte bei der Vorbereitung der Ausführung eines RVPs beachtet werden müssen. Die Implementierung und Demonstration der gesamten Kopplungsmethodik an einem produktiv genutzten RVP wird in Kap. 6 diskutiert, bevor die Arbeit in Kap. 7 zusammengefasst wird.

# 2 Grundlagen gekoppelter Prototypen

## 2.1 Klassische Kopplungen virtueller Prototypen

In Kapitel 1.1.1 wurde bereits der Entwicklungsprozess komplexer mechatronischer Systeme erläutert und die Wichtigkeit des Einsatzes modellbasierter Entwicklungsmethoden herausgearbeitet. Außerdem wurde festgestellt, dass es unter anderem im Automobilbereich durch die zunehmende funktionale Interaktion verschiedener Domänen notwendig ist, bereits in der Entwurfsphase und im weiteren Verlauf des Entwicklungsprozesses das gesamte System bei der Entwicklung einzelner Komponenten des Produkts zu betrachten. Dies ist durch eine Kopplung von Prototypen möglich, deren Reifegrad dem aktuellen Entwicklungsstand der entsprechenden Produktkomponente entspricht. Sie können als rein funktionales Simulationsmodell, als ausführbarer Seriencode einer zu entwickelnden Software oder bereits als fertige Hardwarekomponente des finalen Produkts vorliegen, wodurch sich die Eigenschaften der Kopplung entscheidend ändert.

In den folgenden Abschnitten werden die drei klassischen Konzepte von Prototypenkopplungen Model-In-The-Loop, Software-In-The-Loop und Hardware-In-The-Loop detailliert beschrieben, sowie die numerischen Herausforderungen der jeweiligen Kopplungen herausgestellt. Es wird weiterhin darauf eingegangen, weshalb die in Abschnitt 1.1.1 erforderte durchgängige Verwendung virtueller Prototypen im Entwicklungsprozess bei klassischen Hardware-In-The-Loop Simulationen nicht gegeben ist. Diese Lücke kann, wie bereits dargestellt, durch RVPen gefüllt werden, deren Kopplung in Abschnitt 2.2 definiert wird. Das Verständnis der klassischen Prototypenkopplungen und der zu bewältigenden numerischen Herausforderungen ist eine notwendige Voraussetzung für die Entwicklung des Kopplungsalgorithmus für RVPen in Kap. 3.

### 2.1.1 Model-In-The-Loop

Der Begriff der Model-In-The-Loop (MIL) Simulation kam in den 2000er Jahren in der Automobilindustrie auf. Gühmann (2005) definiert MIL als Konzept zur vollständigen Virtualisierung eines komplexen mechatronischen Systems. Sowohl physikalische Komponenten (z.B. das Fahrzeug) als auch die UUT (z. B. ein Regelungsalgorithmus) liegen als virtueller Prototyp in Form eines Simulationsmodells vor. Diese Definition ist weiterhin gültig. Der Begriff MIL wird auch außerhalb der Automobiltechnik für Cyber-Physische Systeme verwendet, bei denen alle Komponenten eines Gesamtsystems als Computersimulationsmodell abstrahiert sind

(Kim und Kumar, 2012). Diese virtuellen Prototypen stammen im Allgemeinen aus verschiedenen ingenieurstechnischen Domänen, weshalb deren Simulationsmodelle in unterschiedlichen, domänenspezifischen Modellierungswerkzeugen entwickelt werden (Lee, 2015).

MIL-Simulationen ermöglichen Funktionstests des Gesamtsystems bereits in der Entwurfsphase des Entwicklungsprozesses (Turlea, 2019). In der Automobilindustrie ist die domänenübergreifende MIL-Simulation mittlerweile eine zentrale Entwicklungsmethode des Systems-Engineerings auf Fahrzeugebene. Durch den Zusammenschluss detaillierter Simulationsmodelle verschiedener Fahrzeugkomponenten entsteht eine virtuelle Abbildung des gesamten Fahrzeugs, anhand derer früh im Entwicklungsprozess beispielsweise die korrekte Funktionsweise domänenübergreifender Fahrfunktionen wie eines Spurwechselassistenten überprüft werden können (Boumans, Gallet und Schulmeister, 2017). Abb. 2.1 zeigt piktografisch am Beispiel eines batterieelektrischen Fahrzeugs welche Domänenmodelle für die Betrachtung des Gesamtsystems verbunden werden.

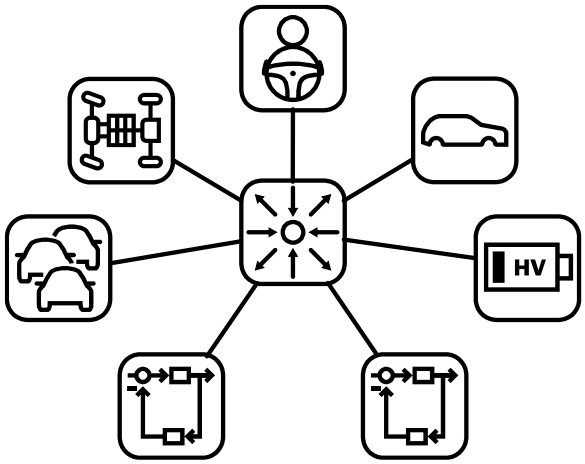


**Abb. 2.1** Piktografische Darstellung der MIL-Simulation eines batterieelektrischen Fahrzeugs mit den Domänen Fahrdynamik, Hochvoltbatterie, Antriebsstrang sowie Regel- und Fahrfunktionen

Für den Aufbau von MIL-Simulationsumgebungen, mit denen Untersuchungen des Gesamtsystems durchgeführt werden können, müssen daher mehrere Simulationsmodelle miteinander gekoppelt werden. Nach Geimer, Krüger und Linsel (2006) kann dies durch eine Kopplung auf Modellebene oder eine Kopplung auf Programmebene erfolgen.

Für die Vereinigung verschiedener Simulationsmodelle auf Modellebene, wodurch deren Lösung mittels eines einzigen Lösers ermöglicht wird, nennt Dronka (2004) fünf Möglichkeiten, von denen in der Praxis keine flächendeckend eingesetzt werden kann. Entweder müssen dafür alle Modelle im selben Modellierungswerkzeug vorliegen oder ein Modell hat eine so geringe Komplexität, dass dessen Verhalten in einem Modellierungswerkzeug einer anderen Domäne ersatzweise dargestellt werden kann. Weiterhin könnten alle Modelle in einer software-

neutralen Modellierungssprache wie Modelica (Fritzson, 2010) umgeschrieben oder softwareneutrale Modellgleichungen verwendet werden. Diese Möglichkeiten werden in der Praxis nur äußerst selten angewandt, da die Anpassung der Simulationsmodelle der einzelnen Domänen zusätzliche Kosten verursacht. Die fünfte Möglichkeit der Kopplung auf Modellebene, der Export der Modellgleichungen in eine Form, die den Import in ein anderes Modellierungswerkzeug erlaubt, ist heutzutage die üblichste Form der Kopplung auf Modellebene. Besonders die Entwicklung des Standards Functional Mock-up Interface (FMI) for Model-Exchange (Blochwitz et al., 2012) hat dazu beigetragen, da das standardisierte Austauschformat den Simulationsingenieuren und Simulationsingenieurinnen das Problem der Portierbarkeit der Modellgleichungen abnimmt. Alle dargelegten Möglichkeiten der Kopplung von Simulationsmodellen auf Modellebene setzen voraus, dass ein Löser dafür geeignet ist das vereinigte Modell effizient zu lösen. Dies ist allerdings im Allgemeinen aufgrund von Eigenschaften der gekoppelten Simulationsmodelle wie unterschiedlicher Zeitkonstanten oder Steifigkeiten häufig nicht der Fall (Petzold, 1983).

Bei Kopplung auf Programmebene behält jedes Simulationsmodell seinen eigenen Löser und mehrere Modellierungswerkzeuge tauschen zu bestimmten Zeitpunkten Signalwerte aus (Geimer, Krüger und Linsel, 2006). Dies wird mittels Methoden der Co-Simulation (Coupled-Simulation) realisiert und ist aufgrund der hohen Flexibilität und Modularität heutzutage die präferierte Variante, um mehrere Simulationsmodelle verschiedenster ingenieurstechnischer Domänen zu einer MIL-Simulationsumgebung eines Gesamtsystems zu verbinden (Gomes et al., 2018). Dabei ist abgesehen von der Definition von Schnittstellen keine Anpassung der Simulationsmodelle notwendig. Das FMI for Co-Simulation (Blochwitz et al., 2012) ist ein standardisiertes Austauschformat für Simulationsmodelle mit eigenem Löser. Kürzlich wurde die Version 3.0 des FMI-Standards veröffentlicht (Hansen et al., 2022).

**Co-Simulation**

Der Begriff Co-Simulation fasst Methoden, Theorien und Techniken zusammen, um Simulationsmodelle mit jeweils eigenem Löser bilateral miteinander zu koppeln. Gomes et al. (2018) unterscheidet zwischen zwei grundsätzlich verschiedenen Paradigmen der Co-Simulation: Der ereignisbasierten und der zeitkontinuierlichen. Eine ereignisbasierte Co-Simulation entsteht durch die Kopplung ausschließlich ereignisbasierter Simulationsmodelle wie beispielsweise Zustandsautomaten (van Tendeloo und Vangheluwe, 2017), die über Ereignisse mit Zeitstempel miteinander kommunizieren. Gomes et al. (2018) zeigen, dass sich ereignisbasierte Co-Simulationen genau wie ereignisbasierte Simulationsmodelle verhalten können und daher keine zusätzlichen Methoden zur Kopplung notwendig sind. Die Herausforderung der ereignisbasierten Co-Simulation besteht demnach darin, die Kausalität auftretender Ereig-

nisse festzustellen und umgehend allen an der Co-Simulation teilnehmenden Simulationsmodellen mitzuteilen und ihnen so die Möglichkeit zu geben auf das Ereignis sowie auf daraufhin ausgelöste Folgeereignisse zu reagieren.

Die MIL-Simulation eines zeitkontinuierlichen mechanischen Systems erfordert dagegen Methoden der zeitkontinuierlichen Co-Simulation, bei der idealerweise kontinuierliche Signale zwischen den gekoppelten Simulationsmodellen ausgetauscht werden. Im Folgenden wird gezeigt, weshalb der Austausch kontinuierlicher Signale zwischen den Modellen in der Praxis nicht möglich ist und deswegen bei jeder kontinuierlichen Co-Simulation ein numerischer Kopplungsfehler entsteht.

Laut einer Umfrage (Schweiger et al., 2019) liegen den Simulationsmodellen $i$ einer kontinuierlichen Co-Simulation in den meisten Fällen ein Satz differential-algebraischer Gleichungen

$$\begin{aligned} \dot{\boldsymbol{x}}_i &= \boldsymbol{f}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, t) \\ \boldsymbol{y}_i &= \boldsymbol{g}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \end{aligned} \tag{2-1}$$

mit der Simulationszeit $t$, dem Zustandsvektor $\boldsymbol{x}_i$, den nichtlinearen Vektorfunktionen $\boldsymbol{f}_i$ und $\boldsymbol{g}_i$, sowie dem Eingangsvektor $\boldsymbol{u}_i$ und dem Ausgangsvektor $\boldsymbol{y}_i$ zugrunde. Eine Co-Simulation entsteht durch Kopplung der verschiedenen Modelle über ihre Ein- und Ausgänge gemäß der Kopplungsbedingung

$$\boldsymbol{u}(t) = \boldsymbol{L} \cdot \boldsymbol{y}(t) \;\text{ mit }\; \boldsymbol{u}(t) = \begin{pmatrix} \boldsymbol{u}_1(t) \\ \boldsymbol{u}_2(t) \\ \vdots \\ \boldsymbol{u}_i(t) \end{pmatrix}, \quad \boldsymbol{y}(t) = \begin{pmatrix} \boldsymbol{y}_1(t) \\ \boldsymbol{y}_2(t) \\ \vdots \\ \boldsymbol{y}_i(t) \end{pmatrix}. \tag{2-2}$$

Die Kopplungsmatrix $\boldsymbol{L}$ hat pro Zeile üblicherweise einen Eintrag und ordnet jedem Eingang den Ausgang eines anderen Simulationsmodells zu. In Abb. 2.2 ist die Kopplung am Beispiel zweier Modelle dargestellt.

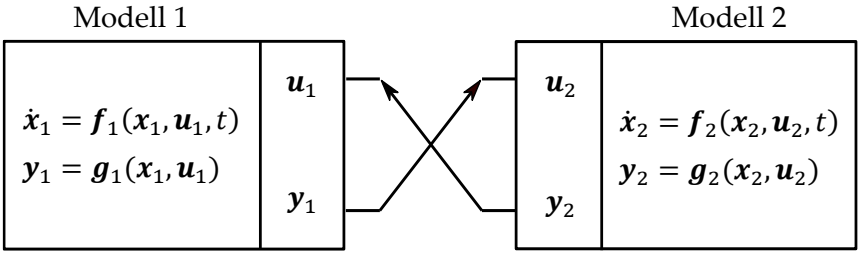


**Abb. 2.2** Kopplung zweier Modelle mittels Co-Simulation (Busch, 2012)

Zur Lösung einer Co-Simulation wird jedes Simulationsmodell mit einem für dieses Modell passenden Löser und einer konstanten oder variablen Schrittweite integriert. Die Zeitschritt-

weite, zu denen die einzelnen Modellgleichungssysteme ausgewertet werden, heißt Mikroschrittweite und wird mit $\delta T_i$ bezeichnet. Die Mikroschrittweiten der Modelle sind voneinander unabhängig und können sich, da die Modelle im Allgemeinen Systeme verschiedener Domänen abbilden, um mehrere Größenordnungen unterscheiden. Daher wird für den Signalaustausch zwischen den Modellen die Makroschrittweite $\Delta T$ als ein zusätzliches Zeitraster definiert. Somit ergeben sich die Zeitpunkte

$$t_n = \sum_j \Delta T_j \text{ mit } j = 0, 1, 2, \dots n \tag{2-3}$$

als Folge konstanter oder veränderlicher Makrozeitschritte $\Delta T_n$, an denen die Simulationsmodelle ihre Simulationszeit synchronisieren und die abgetasteten Ausgangssignale

$$\boldsymbol{y}_n = \boldsymbol{y}(t_n) \text{ mit } n = 0, 1, 2, \dots N \tag{2-4}$$

bereitstellen. Die Kopplungsbedingung (2-2) ist daher nur zu den Zeitpunkten $t_n$ und nicht wie gefordert zu jedem beliebigen Zeitpunkt $t$ exakt erfüllt, weshalb ein Algorithmus zur Rekonstruktion $h(\cdot)$ eingesetzt wird, die den zeitkontinuierlichen Verlauf der Eingänge aller Simulationsmodelle

$$\hat{\boldsymbol{u}}(t) = \hat{\boldsymbol{y}}(t) = h(\boldsymbol{y}, t) \text{ für } t_n \leq t < t_{n+1} \tag{2-5}$$

rekonstruiert (Busch, 2012).

Die Ausführungsreihenfolge der Simulationsmodelle während des Ablaufs einer Co-Simulation beeinflusst welche Signalwerte von $\boldsymbol{y}$ der Rekonstruktion zur Verfügung stehen. Die beiden in Abb. 2.3 gezeigten Schemata Jacobi (parallel) und Gauss-Seidel (sequentiell) sind nach Gomes et al. (2018) die in der Praxis am häufigsten verwendeten. Petridis (2014) untersucht weitere Verfahren im Detail.

Es wird ersichtlich, dass unabhängig von der Ausführungsreihenfolge der Eingangsvektor mindestens eines Modells im Bereich $t_n \leq t < t_{n+1}$ ausschließlich auf Basis vergangener Ausgangswerte $\boldsymbol{y}_{i,j}$ mit $j \leq n$ der anderen Modelle rekonstruiert werden muss, um Eingangswerte für die Mikroschritte in diesem Zeitintervall zu erhalten. Als Algorithmen zur Rekonstruktion werden in diesem Fall Extrapolationsverfahren wie die konstante Extrapolation Zero-Order-Hold (ZOH) oder die lineare Extrapolation First-Order-Hold (FOH) verwendet. In Kap. 3 werden diese und weitere Verfahren diskutiert und deren Anwendbarkeit auf die Kopplung räumlich verteilter Prototypen untersucht.

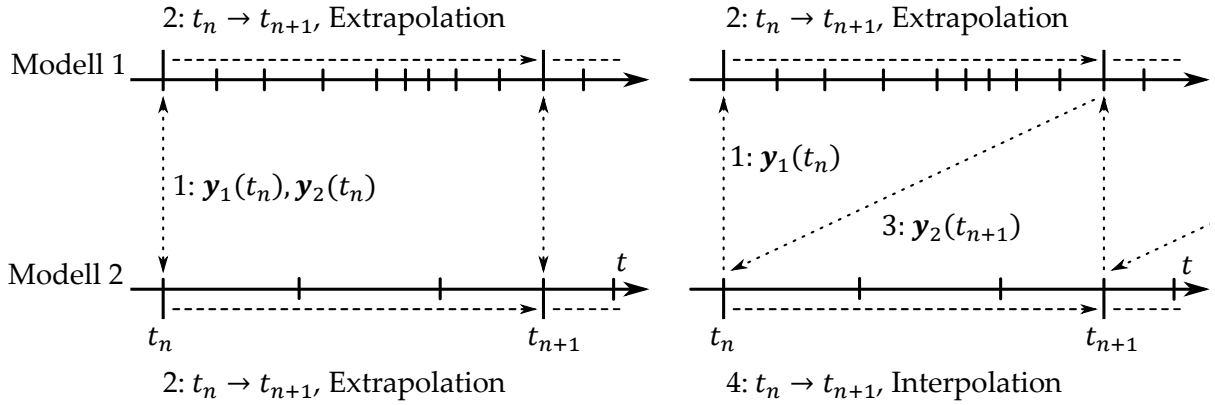


**Abb. 2.3** Schemata für Ausführungsreihenfolge einer Co-Simulation mit zwei Modellen. Links: Jacobi-Schema. Rechts: Gauss-Seidel-Schema.

Aus dem Unterschied zwischen dem tatsächlichen Signalverlauf $\boldsymbol{y}(t)$ und dem rekonstruierten, kontinuierlichen Verlauf $\hat{\boldsymbol{y}}(t)$ kann zu jedem Makrozeitschritt der Kopplungsfehler

$$\boldsymbol{e}(t) = d\big(\boldsymbol{y}(t), \hat{\boldsymbol{y}}(t)\big) \text{ mit } t_n \leq t < t_{n+1} \tag{2-6}$$

mithilfe eines passenden Distanzmaßes $d$ quantifiziert werden. Es ist zu beachten, dass der Kopplungsfehler für den Bereich $t_n \leq t < t_{n+1}$ nur nachträglich zum Zeitpunkt $t_{n+1}$ berechnet werden kann, da dann der tatsächliche Signalverlauf für dieses Zeitintervall bekannt ist. Der Kopplungsfehler kann das Ergebnis sowie die Stabilitätseigenschaften einer Co-Simulation maßgeblich beeinflussen (Busch, 2012). Da er direkt von dem verwendeten Algorithmus zur Rekonstruktion abhängt, ist es unter Umständen möglich, dass eine stabile Co-Simulation eines stabilen Systems nur unter Verwendung eines geeigneten Rekonstruktionsalgorithmus möglich ist.

Stabilität und deren Robustheit sind wichtige Gütekriterien für Systeme. Bei gekoppelten Systemen wie der Co-Simulation, wird zwischen Nullstabilität und numerischer Stabilität unterschieden. Nullstabilität bezeichnet die Stabilität des gekoppelten Systems für eine infinitesimal kleine Makroschrittweite und ist eine notwendige Voraussetzung für eine stabile Co-Simulation (Kübler und Schiehlen, 2000). Numerische Stabilität liegt dagegen vor, wenn das System für eine endliche Makroschrittweite konvergiert. Diese hängt von der Konfiguration der Co-Simulation ab. Bislang gibt es in der Literatur kein allgemeingültiges Vorgehen, um eine Co-Simulation so zu konfigurieren, dass numerische Stabilität sichergestellt ist (Benedikt, Watzenig und Hofer, 2013). Stattdessen werden Methoden präsentiert mit denen Stabilitätsgebiete für Co-Simulationen in Abhängigkeit der Makroschrittweite bestimmbar sind. Diese Untersuchungen sind aufgrund der in der Regel verschiedenen Mikroschrittweiten der gekoppelten Modelle aufwendig und nur für spezielle Konfigurationen der Co-Simulation möglich (Li, 2017, Glumac und Kovacic, 2019). Schweiger et al. (2019) stellen in ihrer Umfrage fest, dass

der größte Forschungsbedarf im Bereich der Co-Simulation aktuell darin besteht, die Auswirkungen der Rekonstruktion des kontinuierlichen Signalverlaufs auf das Systemverhalten und somit auf Ergebnisse der Co-Simulation abzuschätzen zu können, wodurch die Verlässlichkeit von Co-Simulationen erhöht wird. In Kap. 4 dieser Arbeit werden Stabilitätsanalysen gekoppelter Systeme anhand des Nyquist-Kriteriums durchgeführt.

Bei der Wahl der Makroschrittweite einer zeitkontinuierlichen Co-Simulation wird ein Kompromiss aus Rechengeschwindigkeit und Genauigkeit der Co-Simulation eingegangen (Benedikt, Stippel und Watzenig, 2010). Einerseits führt eine Verringerung der Makroschrittweite zu einem kleineren Kopplungsfehler, da das Zeitintervall $[t_n, t_{n+1})$ kleiner wird, in dem die Eingangssignale rekonstruiert werden müssen. Andererseits verlangsamt dies die Co-Simulation, da der Kommunikationsaufwand aufgrund des kleineren Austauschintervalls steigt und gleichzeitig die Löser der einzelnen Simulationsmodelle unter Umständen gezwungen werden kleinere Mikroschritte durchzuführen, um an jedem Makrozeitpunkt Signalwerte bereitstellen zu können. Benedikt et al. (2013) definieren eine empirische Regel, nach der die Makroschrittweite gewählt werden kann. Ideen die Makroschrittweite beispielsweise durch das Festlegen von Fehlertoleranzen wie in Schierz, Arnold und Clauß (2012) zur Laufzeit der Co-Simulation zu variieren, haben sich in industriellen Anwendungen bislang nicht durchgesetzt. Ein Grund dafür ist, dass einige in der Industrie genutzten Simulationswerkzeuge keine variable Mikroschrittweite erlauben, welches eine Voraussetzung für den effektiven Einsatz einer variablen Makroschrittweite ist (Meyer, Kraft und Schweizer, 2021).

Eine iterative Ausführung der Co-Simulation könnte analog zur impliziten Lösung von Differentialgleichungssystemen den Kopplungsfehler einer Co-Simulation deutlich verringern. Allerdings müsste dazu jedes Simulationsmodell in der Lage sein, die Simulationszeit sowie den internen Zustand zurückzusetzen. Dies ist in den klassischen Modellierungswerkzeugen aufgrund des enormen Konfigurationsaufwands bei komplexen Systemen nicht vorgesehen, weshalb in der Praxis fast ausschließlich die nicht-iterative Co-Simulation Anwendung findet (Benedikt, Watzenig und Hofer, 2013).

Es ist Stand der Technik für den Aufbau einer MIL-Simulation bei der mehr als drei Simulationsmodelle mittels Co-Simulation verbunden werden, eine Co-Simulations-Middleware zu verwenden. Die Middleware ist eine eigenständige Software, die Schnittstellen zu den üblichen Modellierungswerkzeugen bereitstellt und somit die zeitliche Synchronisation sowie den Datenaustausch zwischen den Simulationsmodellen ermöglicht (vgl. Abb. 2.1). Sie stellt weiterhin die aufgezeigten Einstellungsmöglichkeiten für die zeitkontinuierliche Co-Simulation bereit. Günther (2017) vergleicht den Funktionsumfang einiger Co-Simulations-Middlewares.

### 2.1.2 Software-In-The-Loop

Die Methode der Software-In-The-Loop (SIL) Simulation erlaubt die virtuelle Validierung von Steuerungs- und Regelungssoftware. Da eine Validierung nur am endgültigen Produkt durchgeführt werden kann, reicht eine MIL-Simulation, bei der die UUT als funktionales Simulationsmodell abstrahiert ist, dazu nicht aus. Stattdessen muss der Seriencode der Software, der später auf der Zielhardware verwendet wird, sowie die Kommunikation zwischen den verschiedenen Softwarekomponenten, die im Produkt meist über ein Bussystem stattfindet, validiert werden. Damit dies virtuell und dadurch kostengünstig und wiederholbar möglich ist, wird die zu testende Software auf einem PC emuliert, wobei, wie im nächsten Abschnitt dargestellt, verschiedene Ausbaustufen möglich sind. SIL ergibt sich durch eine Kopplung der emulierten Komponenten mit virtuellen Prototypen der anderen Komponenten des Gesamtsystems.

Erstmalig wurde der Begriff SIL von Kwon et al. (1998) zur Beschreibung ihrer Kopplung von Regelungssoftware und eines Simulationsmodells einer physikalischen Komponente über ein Netzwerk verwendet. Heutzutage werden SIL-Simulationen im Automobilbereich zur virtuellen Validierung von Steuergeräten (electronic-control-unit (ECU)) eingesetzt. Dazu werden virtuelle ECUs erstellt, deren Softwareumfang vom gewünschten Testziel abhängt und daher variieren kann (Baumann et al., 2019d). Zur rein funktionalen Validierung reicht es aus, wenn die virtuelle ECU lediglich die Applikationssoftware zusammen mit einer Laufzeitumgebung enthält. Für realistischere Tests, beispielsweise der Ablaufplanung auf Task-Ebene oder der Kommunikation der virtuellen ECU via eines Bussystems, muss zusätzlich die Basissoftware der ECU zusammen mit deren Abhängigkeiten emuliert werden. Außerdem ist es möglich, auch die Hardware der ECU zu simulieren, wodurch detaillierte Analysen des Zeitverhaltens der ECU durchführbar sind. Die Arbeiten von Junghanns, Mauss und Seibt (2014) sowie Mikelsons und Samlaus (2017) erlauben es bestehende MIL-Simulationen (vgl. Abb. 2.1) durch die Kopplung von virtuellen ECUs nahtlos in SIL-Simulationen zu überführen, damit große Teile der mit hohem Aufwand entwickelten MIL-Simulationen wiederverwendet werden können. Abb. 2.4 zeigt eine SIL-Simulationsumgebung für die Validierung domänenübergreifender Fahrfunktionen, die aufgrund ihrer Komplexität über mehrere ECUs verteilt sein können. Um die Interaktion der ECUs zu simulieren, müssen mehrere virtuelle ECUs sowie deren Kommunikation untereinander mittels eines virtuellen Busses modelliert werden (Baumann et al., 2019c).

Für die Kopplung ereignisbasierter Software wie virtueller ECUs oder Kommunikationsbusse und zeitkontinuierliche, signalbasierte Simulationsmodelle sind Methoden und Theorien der hybriden Co-Simulation notwendig, welche in Middlewares wie Silver (Junghanns et al., 2012) oder FERAL (Kuhn et al., 2013) implementiert sind, die eine Kopplung von Systemen beider

Paradigmen ermöglichen. Schweiger et al. (2019) stellen in ihrer Umfrage fest, dass die Herausforderungen der hybriden Co-Simulation, welche im nächsten Abschnitt aufgezeigt werden, vielen Anwendern nicht ausreichend bewusst sind.

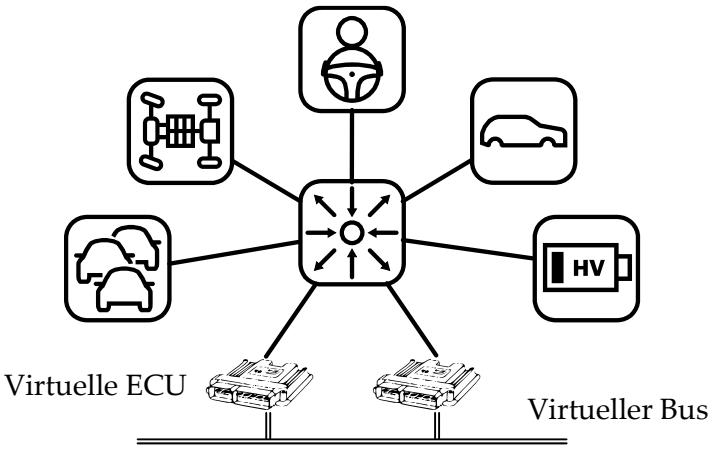


**Abb. 2.4** Piktografische Darstellung einer SIL-Simulation eines batterieelektrischen Fahrzeugs

**Hybride Co-Simulation**

Eine hybride Co-Simulation entsteht durch Kopplung ereignisbasierter und zeitkontinuierlicher Simulationsmodelle. Für den Datenaustausch zwischen den Modellen dieser beiden Paradigmen gibt es zwei Möglichkeiten. Die erste besteht darin, die ereignisbasierten Modelle als zeitkontinuierliches Modell zu behandeln und genau wie bei einer zeitkontinuierlichen Co-Simulation den Signalaustausch an bestimmten Zeitpunkten durchzuführen (vgl. Abschnitt 2.1.1). Dafür muss das ereignisbasierte Modell erweitert werden, damit es relevante Ereignisse in den Eingangssignalen identifizieren kann (Lawrence et al., 2016). Alternativ kann der Datenaustausch zwischen den Modellen wie bei der ereignisdiskreten Co-Simulation durch Übertragung von Ereignissen stattfinden. Dafür müssen allerdings Ereignisse in den Ausgängen der zeitkontinuierlichen Modelle identifiziert werden (Vangheluwe, 2000). Beides ist herausfordernd, da ereignisbasierte Simulationsmodelle exakt zu den Zeitpunkten ausgewertet werden müssen, an denen ein Ereignis in einem der gekoppelten Modelle auftritt. Diese können aber erst im darauffolgenden Makrozeitpunkt übermittelt werden, wodurch sie zeitverzögert an den Eingängen der ereignisbasierten Simulationsmodellen verfügbar sind (Gomes et al., 2018).

Im Unterschied zur zeitkontinuierlichen Co-Simulation kann die Verzögerung bei der Übertragung von Ereignissen nicht durch eine Verwendung von Extrapolationsverfahren kompensiert werden, da Ereignisse sich nicht in den Ausgangssignalen der Simulationsmodelle ankündigen. Eine sequentielle Ausführung der Simulationsmodelle kann, ähnlich wie bei Lawrence et al. (2016), dabei helfen die Ereignisse zeitlich korrekt von den zeitkontinuierlichen zu

den ereignisbasierten Modellen zu übertragen. Für eine verzögerungsfreie, bilaterale Übertragung ist es allerdings notwendig, dass die zeitkontinuierlichen Simulationsmodelle ihre Integrationsschritte zurücknehmen können (vgl. iterative Co-Simulation in Abschnitt 2.1.1) (Gomes et al., 2018).

Bei SIL-Simulationen im Automobilbereich wird die hybride Co-Simulation üblicherweise realisiert, indem die ereignisbasierte Seriensoftware durch Export als FMU in ein zeitkontinuierliches Simulationsmodell überführt wird. Dadurch kann sie nahtlos in bestehende MIL-Simulationsumgebungen eingebunden werden (Wissel et al., 2018, Baumann et al., 2019d). Ein zeitlich korrekter Austausch der Ereignisse zwischen zeitkontinuierlichen und ereignisbasierten Simulationsmodellen ist bei den publizierten SIL-Simulationsumgebungen jedoch aufgrund des oben beschriebenen Problems nicht gegeben und Gegenstand aktueller Forschung.

### 2.1.3 Hardware-In-The-Loop

Eine Hardware-In-The-Loop (HIL) Simulation ergibt sich aus der bilateralen Kopplung von realen Prototypen mit Echtzeit-Simulationsmodellen. Der reale Prototyp ist dabei eine reale mechanische Komponente (beispielsweise Fahrsimulator oder Prüfstand) oder ein reales Steuergerät. Die Echtzeit-Simulationsmodelle simulieren je nach Anwendung das restliche physikalische System oder Steuerungs- und Regelungsalgorithmen. Handelt es sich bei der Hardware um eine mechanische Komponente, bilden, wie in Abb. 2.5 gezeigt, Sensoren und Aktuatoren die Schnittstelle zwischen den virtuellen und realen Komponenten. Bei einem Steuergerät erfolgt der Datenaustausch über analoge und digitale I/O-Schnittstellen.

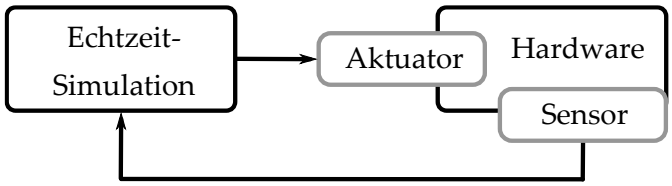


**Abb. 2.5** Blockdiagramm einer HIL-Simulation

Bereits in den 1960er Jahren wurden HIL-Simulationen von der NASA in der Luft- und Raumfahrtechnik produktiv für die Validierung von Hardwarekomponenten genutzt (Gross, 1967). In den darauffolgenden Jahrzehnten nahmen weitere Branchen das Konzept HIL als festen Bestandteil in den Entwicklungsprozess komplexer mechatronischer Systeme auf. Isermann, Schaffnit und Sinsel (1999) machten HIL in der Automobilentwicklung bekannt, indem sie einen Motorprüfstand mit Simulationsmodellen weiterer physikalischer Fahrzeugkomponenten koppelten. Auch in der Robotik (Tejado et al., 2016), Elektrotechnik (Lauss et al., 2015) und im maritimen Umfeld (Aarseth et al., 2014) gibt es Beispiele für den produktiven Einsatz von HIL-Simulationen. Mihalič, Truntič und Hren (2022) geben einen aktuellen Überblick über die Anwendungen und Herausforderungen von HIL-Simulationen.

Sarhadi und Yousefpour (2015) haben festgestellt, dass die Wichtigkeit von HIL-Simulation in der Produktentwicklung komplexer mechatronischer Systeme weiter zunimmt und sie vor allem in späteren Phasen des Entwicklungsprozesses Anwendung findet. Üblicherweise werden HIL-Simulationen nach rein virtuellen Tests (MIL und SIL) und vor Tests an realen Prototypen des Gesamtprodukts durchgeführt. Wünsch (2008) nutzt eine HIL-Simulation um eine Steuerungshardware vorzeitig an einer virtuellen Werkzeugmaschine in Betrieb zu nehmen, ohne das Risiko die echte Werkzeugmaschine durch eine Fehlfunktion der Steuerung zu beschädigen. Weiterhin kann mittels einer HIL-Simulation die Funktionalität von Hardwarekomponenten unter realistischen Bedingungen getestet werden, die in Tests mit Prototypen des Gesamtprodukts nur schwer nachzustellen wären. Reinold et al. (2019) testen einen realen Verbrennungsmotor und eine reale elektrische Maschine in realistischen und reproduzierbaren virtuellen Verkehrsszenarien. Weiterhin kann mit HIL auch die korrekte Funktionalität von Sensoren oder Aktoren untersucht werden (Isermann, Schaffnit und Sinsel, 1999) oder die Interaktion von Nutzer und Hardware unter Verwendung von Fahrsimulatoren (Maas et al., 2014). Ist das Echtzeit-Simulationsmodell die UUT (beispielsweise ein funktionales Simulationsmodell eines neuartigen Regelungsalgorithmus) dient die angebundene Hardwarekomponente dazu realistische Umstände wie Rauschen oder Verzögerungen darzustellen (Lee et al., 2011). Diese Vorgehensweise für die Steuerungs- und Regelungsentwicklung wird Raid Control Prototyping genannt (Abel und Bollig, 2006).

**Echtzeit-Simulation**

Damit Simulationsmodelle mit Hardwarekomponenten bilateral gekoppelt werden können, muss die Simulation in Echtzeit ausführbar sein. Im Gegensatz zur klassischen Simulation, bei der die Rechenzeit nicht begrenzt ist und nur die Kosten der Simulation beeinflusst, bildet nun die Echtzeit die obere Schranke der Simulationszeit. Die Folge aus Kommunikationszeitpunkten $t_n$ zwischen Hardwarekomponenten und Simulationsmodell kann nicht frei bestimmt werden, da den Simulationsmodellen für die Integration der Simulationszeit von $t_n$ bis $t_{n+1}$ maximal $t_{calc} = t_{n+1} - t_n - t_{conv}$ Rechenzeit in Echtzeit zur Verfügung steht. Die Zeitspanne $t_{conv}$ ist dabei die benötigte Zeit, um die Eingangssignale der Simulation einzulesen, die von der Hardware außerhalb des PCs generiert werden, sowie die Ausgangssignale der Simulation für die Nutzung durch die Hardware aufzubereiten (Cellier und Kofman, 2006). Ist das Zeitintervall $[t_n, t_{n+1})$ zu kurz, nimmt $t_{conv}$ einen zu großen Anteil davon ein und die verbleibende Rechenzeit ist für eine Integration der Simulationsmodelle in diesem Zeitintervall nicht mehr möglich. Es wird zwischen harter und weicher Echtzeitanforderung (hard-real-time, soft-real-time) unterschieden. Während bei ersterer die Rechenzeit in jedem einzelnen Simulationsschritt innerhalb der vorgegebenen Schranke bleiben muss, werden bei letzterer gelegentliche Verletzungen der Rechenzeitbeschränkung toleriert.

Um der Anforderung der harten Echtzeitfähigkeit zu genügen, reicht es nicht die durchschnittliche Rechenzeit pro Zeitintervall der Simulationsmodelle zu betrachten, stattdessen muss die maximale Rechenzeit, die für das Zeitintervall $[t_n, t_{n+1})$ benötigt wird, in jedem Fall kleiner als $t_{calc}$ sein. Dazu müssen die Simulationsmodelle rechenzeitoptimiert werden. Die Stellschrauben dafür sind eine Vereinfachung des Funktionsumfangs des Modells, der Einsatz expliziter Methoden geringer Ordnung mit fester Schrittweite als Integrationsverfahren oder eine Erhöhung der Integrationsschrittweite (Cellier und Kofman, 2006). Weiterhin dürfen im Simulationsmodell oder im Integrationsverfahren keine iterativen Methoden verwendet werden, da ansonsten die maximale Laufzeit eines Integrationsschritts unter Umständen sehr groß und unberechenbar wird, wodurch Echtzeitverletzungen auftreten können. Außerdem müssen die Simulationsmodelle bei einer Echtzeit-Co-Simulation auf Computersystemen ausgeführt werden, welche Schnittstellen zu den für Hardwarekomponenten üblichen Kommunikationsmedien haben. Im Automobilbereich sind dies die Bussysteme CAN, Flexray oder LIN sowie analoge und digitale I/O-Schnittstellen oder eine direkte Ethernet Verbindung mit UDP oder TCP als Kommunikationsprotokoll (Zimmermann und Schmidgall, 2006). Echtzeitsysteme wie beispielsweise Labcar von ETAS (ETAS, 2020), Speedgoat von Mathworks (Speedgoat, 2020), oder Scalexio von dSpace (dSpace, 2020) bieten all diese Schnittstellen und leistungsfähige Prozessoren an, um Modelle unter harten Echtzeitbedingungen simulieren zu können. Allerdings ist zur Ausführung der Simulationsmodelle auf einem Echtzeitsystem immer eine Kompilation des Modells als ausführbare Datei nötig, wodurch je nach verwendetem Modellierungswerkzeug weitere Modellanpassungen notwendig sind. Beispielsweise wird in Matlab/Simulink von Mathworks nicht bei jeder Funktion die Codegenerierung unterstützt.

Obwohl bei einer HIL-Simulation genau wie bei einer MIL-Simulation zeitkontinuierliche Systeme miteinander verbunden werden, spielen Kopplungsfehler bzw. der Umgang mit diesen in der Literatur keine Rolle. Es wird versucht die Kommunikationsschrittweite gleich der festen Schrittweite der kompilierten Simulationsmodelle zu setzen, sodass in jedem Integrationsschritt neue Signalwerte der Hardwarekomponente vorliegen und dadurch eine Rekonstruktion nicht notwendig ist. Dies wird durch eine hohe Übertragungsgeschwindigkeit zwischen Hardwarekomponente und Echtzeitcomputersystem infolge einer schnellen Signalverarbeitung sowie kurzer, direkter physischer Verbindungen erreicht (Vekić et al., 2012). Jeon et al. (2010) erreichen bei ihrer HIL-Simulation beispielsweise eine Übertragungszeit von nur $50\mu s$. Aufgrund der dafür erforderlichen Nähe des Echtzeitsystems und der Hardwarekomponente sind HIL-Simulationen örtlich gebunden, wodurch Flexibilität verloren geht und unternehmensübergreifende Zusammenarbeit nur durch den Austausch von Modellen oder Hardwarekomponenten möglich ist.

Durch die notwendige Anpassung der Simulationsmodelle für eine HIL-Simulation, können MIL- oder SIL-Simulationsumgebungen nicht unverändert wiederverwendet werden, wodurch die Durchgängigkeit in der modellbasierten Produktentwicklung verloren geht (vgl.

Abschnitt 1.1.1). Zwar integriert beispielsweise Himmler (2014) mehrere domänenübergreifende Modelle auf einem HIL-Echtzeitcomputersystem, allerdings muss dafür jedes einzelne Modell den obigen Anforderungen entsprechen und als kompilierter C-Code vorliegen, wodurch der Anwender Flexibilität verliert.

## 2.2 Aspekte räumlich verteilter Real-Virtueller Prototypen

Genau wie HIL-Simulationen ergeben sich auch RVPen aus der Kopplung realer und virtueller Prototypen. Allerdings handelt es sich bei den virtuellen Prototypen eines RVPs nicht um Echtzeit-Simulationsmodelle, sondern um Simulationsmodelle einer MIL- oder SIL-Simulation, die nicht auf einem Echtzeitsystem ausgeführt und auch nicht für die Verwendung unter Echtzeitbedingungen modelliert und kompiliert werden. Stattdessen werden virtuelle Prototypen eines RVPs meist auf einem Standard-PC (üblicherweise mit Windows-Betriebssystem) ausgeführt. Dies ermöglicht die durchgängige Verwendung von unveränderten Simulationsmodellen im gesamten Entwicklungsprozess. In Kap. 1 wurde bereits ein Überblick zu Real-Virtuellen Prototypen (RVP), deren Anwendungsbereiche, die Problematik der auftretenden Netzwerkeffekte, sowie vorhandene Kopplungsalgorithmen zur Kompensation der Netzwerkeffekte gegeben. In diesem Abschnitt wird genauer auf einige Aspekte von RVPen eingegangen, die für die Entwicklung des Kopplungsalgorithmus in Kap. 3 notwendig sind. Dazu gehören die technischen Randbedingungen der Kommunikation, die Anforderungen an die gekoppelten virtuellen Prototypen in einem RVP und die mathematische Beschreibung des Datenaustauschs einer Echtzeit-Co-Simulation, aus der sich die Randbedingungen des Kopplungsalgorithmus ergeben.

### 2.2.1 Technische Randbedingungen

Im Gegensatz zu MIL- und SIL-Simulationen, bei denen die Prototypenkopplung üblicherweise lokal auf einem einzelnen PC erfolgt, sind die gekoppelten Prototypen in RVPs räumlich verteilt. Diese verteilte Ausführung der Prototypen erfordert einen Datenaustausch zur Laufzeit des RVPs zwischen den PCs, welche die virtuellen Prototypen berechnen, und den Echtzeitsystemen der Hardwarekomponenten. Dabei müssen zwei Dinge betrachtet werden. Zum einen die verwendeten Kommunikationsprotokolle und die dazu gehörigen physischen Verbindungen der Prototypen und zum anderen die Konfiguration des Datenaustauschs entsprechend den Signalabhängigkeiten zwischen den Prototypen.

Aufgrund der angebundenen realen Prototypen unterliegt auch die Kommunikation zwischen den Prototypen der Echtzeitanforderung. Schreiber et al. (2018) beschreiben Indikatoren mit denen sich die Qualität einer Echtzeit-Kommunikation messen lässt. Diese sind die Zeitverzögerung zwischen dem Senden und dem Empfangen einer Nachricht, der dabei auftretende

Jitter, die prozentuale Anzahl an verloren Nachrichten, sowie die Schwankungen dieser Größen zur Laufzeit des RVPs. Die Feldbusse aus dem Automobilbereich CAN, FlexRay, LIN und EtherCAT (Zimmermann und Schmidgall, 2006) sind zwar für Echtzeit-Kommunikationen geeignet, wie ein Benchmark bezüglich der beschriebenen Indikatoren ergibt (Schreiber et al., 2018), können aber nur eine geringe Distanz überbrücken, weshalb sie in RVPs nur in Spezialfällen Anwendung finden. In allen bekannten RVPen, deren Prototypen über eine größere Distanz verteilt sind, wird stattdessen das Ethernet basierte UDP-Protokoll verwendet. Schreiber et al. (2018) begründen dies mit der hohen Flexibilität, da nahezu jeder Computer über eine Netzwerkkarte verfügt und somit das UDP-Protokoll unterstützt und mit anderen Computern, sogenannten Knoten, über ein Ethernet-Netzwerk verbunden ist. Es gibt lokale beispielsweise unternehmensinterne Netzwerke und das Internet als globales öffentliches Netzwerk, welches für einen weltweiten Datenaustausch zwischen Computern via UDP genutzt werden kann. Außerdem ist die Latenzzeit der Nachrichten bei UDP im Vergleich zum anderen etablierten Internetprotokoll TCP wesentlich geringer, da die Nachrichten verbindungslos gesendet werden und keine Ende-zu-Ende Verbindung zwischen Sender und Empfänger aufgebaut wird.

Der Datenaustausch zwischen gekoppelten Prototypen per UDP hat auch Nachteile. Da das UDP-Protokoll nicht verbindungsorientiert ist, können Nachrichten vereinzelt verloren gehen, ohne, dass der Empfänger dies bemerkt. Weiterhin ist die Qualität der Verbindung davon abhängig, wie viele weitere Knoten über dasselbe Ethernet-Netzwerk kommunizieren. Findet neben der Kommunikation zwischen den Prototypen eines RVPs kein weiterer Datenaustausch über das Netzwerk statt, ist die Qualität gut, d.h. die Latenzzeit liegt im niedrigen Bereich weniger Millisekunden und der Jitter ist gering. Findet der Datenaustausch dagegen über das Internet statt und teilt sich dadurch die Bandbreite mit dem Datenfluss zwischen zahlreichen anderen Knoten, nimmt die Qualität der Verbindung mit zunehmender Distanz ab (Kaune et al., 2009). Bei länderübergreifender Verbindung innerhalb Europa ist mit einer durchschnittlichen Umlaufzeit von knapp $50ms$ und einem Jitter von $\pm 25ms$ zu rechnen (Schreiber et al., 2018, Baumann et al., 2019b).

Die Verwendung des Standards Time-Sensitive-Networking (TSN) könnte die Qualität des Datenaustauschs bei RVPs über Ethernet-Netzwerke zukünftig verbessern. Der Standard beruht auf einer exakten Zeitsynchronisation aller Knoten des Netzwerks und teilt den Datenaustausch in Zyklen ein, in denen bestimmte Zeitfenster für Nachrichten mit hoher Priorität reserviert sind. Diese können dann in niedriger Latenzzeit und geringem Jitter versandt werden, da in ihrem Zeitfenster der Datenverkehr im Netzwerk sehr gering ist. Jiang et al. (2019) evaluieren bereits die Verwendbarkeit von TSN für HIL-Simulationen. Der neue Mobilfunkstandard 5G könnte ebenfalls zukünftig für den Datenaustausch bei RVPs eingesetzt werden, sobald entsprechende Mobilfunkmasten flächendeckend verfügbar sind. Agiwal, Roy und

Saxena (2016) fassen den Stand der Technik des 5G Standard zusammen, nach dem Umlaufzeiten von $1ms$ einer bilateralen Kommunikation erreicht werden können.

### 2.2.2 Echtzeitanforderungen an die virtuellen Prototypen

Damit der nahtlose Übergang von MIL- bzw. SIL-Simulationen zu RVPs gewährleistet werden kann, müssen die virtuellen Prototypen eines RVPs auf der gleichen Simulationshardware berechnet werden, die auch bei der vorangegangenen rein virtuellen Prototypenkopplung verwendet wurde. Dabei handelt es sich üblicherweise um Windows-Computer, da die meisten in der Industrie genutzten Modellierungswerkzeuge der Systemsimulation für Windows entwickelt sind. Bei RVPen unterliegen aufgrund der angebundenen realen Prototypen allerdings auch die Berechnungen der virtuellen Prototypen Echtzeitanforderungen. Dies bedeutet nach den beispielsweise von Schreiber et al. (2018) genannten Charakteristika eines Echtzeitsystems, dass dieses in der Lage sein muss, Aktionen prompt und innerhalb einer bestimmten, vorhersehbaren Frist mit auszuführen. Windows kann das strikte Einhalten von Fristen bei Berechnungen, welches zum Erreichen harter Echtzeitbedingungen notwendig ist, nicht garantieren, da unter anderem Hintergrundprozesse laufenden Anwendungen die Ressourcen des Prozessors für einige Millisekunden entziehen können. Weiche Echtzeitanforderungen können mit einem entsprechend konfigurierten Windows-Computer aber durchaus erreicht werden, solange überprüft wird, wie oft Fristverletzungen auftreten und welches die Konsequenzen einer solchen Fristverletzung sind (Terhell, 2014).

Übertragen auf die Berechnung virtueller Prototypen bedeutet das Einhalten von Echtzeitbedingungen, dass die Simulationszeit mit der Echtzeit synchronisiert sein muss. Die numerische Integration einer bestimmten Zeitspanne der Simulationsmodelle darf demnach nicht länger dauern als die entsprechende Zeitspanne in Echtzeit. Stimmen die vergangene Simulationszeit und Echtzeit zu Zeitpunkten, an denen Daten ausgetauscht werden, nicht überein, liegt eine Echtzeitverletzung vor. Je schneller die zeitliche Integration der Simulationsmodelle durchgeführt werden kann, desto mehr Pufferzeit steht vor jedem Datenaustausch zur Verfügung, wodurch die Wahrscheinlichkeit für Echtzeitverletzungen sinkt. Eine Vergrößerung des Zeitrasters, in dem der Datenaustausch mit den gekoppelten Prototypen stattfindet, verringert ebenfalls die Wahrscheinlichkeit für Echtzeitverletzungen, da auf diese Weise die absolut verfügbare Pufferzeit vor jeder Frist erhöht werden kann. Das Auftreten von Echtzeitverletzungen kann allerdings nie vollständig verhindert werden, da der Aufwand der Berechnungen und damit die benötigte Rechenzeit der Simulationsmodelle in einzelnen Zeitintervallen steigen kann. Gründe hierfür sind eine Verringerung der Schrittweite bei expliziten Lösern mit Schrittweitensteuerung oder eine steigende Anzahl an notwendigen Iterationen bei der numerischen Lösung der im allgemeinen nichtlinearen Gleichung eines impliziten Lösungsverfahrens. Außerdem erhöhen auftretende Ereignisse, da diese für eine genaue Lösung des Modells exakt zeitlich lokalisiert werden müssen, genauso wie algebraische Schleifen die Rechenzeit.

Sollte es zu einer Echtzeitverletzung gekommen sein, müssen die noch fehlenden Berechnungen des letzten Integrationsintervalls im nächsten Intervall nachgeholt werden, wodurch ohne ausreichende Pufferzeiten die Simulationszeit und die Echtzeit auseinanderdriften können.

Trotz der Echtzeitanforderung ist es möglich mehrere virtuelle Prototypen eines RVPs unter weichen Echtzeitbedingungen auf einem einzelnen Windows-PC gemäß der in Abschnitt 2.1.1 vorgestellten Architektur auszuführen, da die aktuell üblichen Mehrkern-Prozessoren eine parallele Ausführung mehrerer Anwendungen begünstigt.

### 2.2.3 Echtzeit-Co-Simulation

Die Echtzeit-Co-Simulation ist ein Spezialfall der Co-Simulation und fasst Methoden zur Kopplung von Prototypen unter bestehender Echtzeitanforderung zusammen, bei denen die realen und virtuellen Prototypen räumlich verteilt sind und über ein Netzwerk kommunizieren (Stettinger et al., 2013). Somit ist der Datenaustausch der Prototypen eines RVPs als Echtzeit-Co-Simulation formulierbar. Dabei macht es keinen Unterschied, ob die Prototypen über kurze Distanz in einem lokalen Netzwerk oder über große Distanz via Internet verbunden sind.

Der in Abb. 2.6 dargestellte Datenaustausch zwischen Prototypen unter Echtzeitanforderung über ein Netzwerk erfolgt genau wie der Datenaustausch zwischen den Simulationsmodellen einer Co-Simulation an bestimmten Zeitpunkten, die über die Folge

$$t_n = n\Delta T \tag{2-7}$$

mit der Makroschrittweite $\Delta T$ definiert sind. Im Unterschied zur Co-Simulation ist die Makroschrittweite bei der Echtzeit-Co-Simulation konstant, weshalb die Kommunikationszeitpunkte äquidistant sind. Die Vorteile einer adaptiven Makroschrittweite, nämlich die Verringerung der Rechenzeit der Co-Simulation durch dynamische Anpassung des Zeitrasters an das aktuelle Systemverhalten, wirken sich bei der Echtzeit-Co-Simulation nicht aus, da die Ablaufgeschwindigkeit der Simulation durch die Echtzeitanforderung fixiert ist.

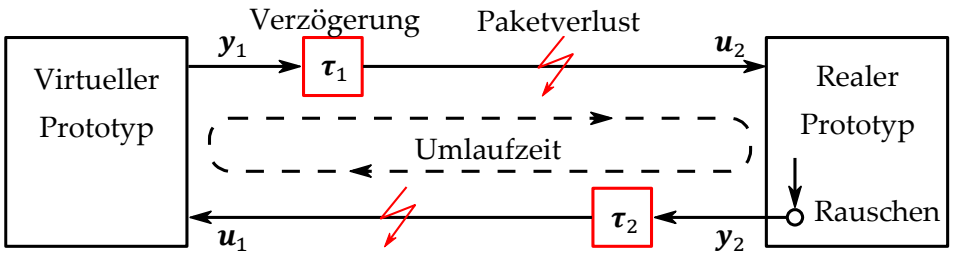


**Abb. 2.6** Datenaustausch zweier Prototypen über ein Netzwerk nach Stettinger et al. (2017).

Die Ursache des Kopplungsfehlers einer Co-Simulation begründet sich wie in Abschnitt 2.1.1 gezeigt darin, dass die Kopplungsbedingung (2-2) statt wie gefordert für alle Zeiten nur zu den Zeitpunkten $t_n$ erfüllt ist und das Koppelsignal zwischen den Makrozeitpunkten rekonstruiert werden muss. Bei einer Echtzeit-Co-Simulation ist die Kopplungsbedingung aufgrund der in Abb. 2.6 dargestellten Netzwerkeffekte (vgl. Abschnitt 2.2.1) dagegen zu keinem Zeitpunkt exakt erfüllbar, wodurch der Kopplungsfehler einer Echtzeit-Co-Simulation im Allgemeinen größer ist als bei einer Co-Simulation. Ersichtlich wird dies bei Betrachtung eines einzelnen Eingangssignals $u_i$, welches mit dem Ausgang $y_j$ eines anderen Prototyps über ein Netzwerk verbunden ist. Die zeitvariante Latenzzeit $\tau_j(t)$ verfälscht das Ausgangssignal, wodurch sich für das Eingangssignal statt der idealen Gleichheit

$$u_i(t) = y_j(t) \tag{2-8}$$

der veränderte Verlauf

$$\hat{u}_i(t) = \hat{y}_j(t) = y_j\big(t - \tau_i(t)\big) \tag{2-9}$$

ergibt, wenn immer der neuste verfügbare Signalwert unverändert als Eingang verwendet wird. Die Latenzzeit in Netzwerken hängt wie von Yang et al. (2005) beschrieben von vielen verschiedenen Faktoren ab und kann aufgrund des übermittelten Datentyps, sowie einer möglichen Priorisierung von Nachrichten für jedes Eingangssignal verschieden sein.

Nach Diskretisierung mit der Makroschrittweite $\Delta T$, die das Zeitraster des Datenaustauschs definiert, gilt für den Eingang

$$\hat{u}_{i,n} = \hat{y}_{j,n} = y_{j,n-k_{i,n}} \text{ mit } t_n \leq t < t_{n+1} \tag{2-10}$$

und

$$k_{i,n} = \min\{m \in \mathbb{N} \mid m \geq \tfrac{\tau_{i,n}}{\Delta T}\} \text{ mit } \tau_{i,n} > 0. \tag{2-11}$$

Laut Gl. (2-11) gibt es einen direkten Zusammenhang zwischen der Makroschrittweite $\Delta T$ und der Anzahl der Makroschritte Verzögerung $k$. In der Literatur ist die Makroschrittweite meist so gewählt, dass im Mittel $k \leq 10$ ist, weshalb auch die Kopplungsalgorithmen auf dieses Verhältnis zwischen Makroschrittweite und Verzögerung ausgelegt sind (Stettinger et al., 2013, Schreiber et al., 2018, Tranninger et al., 2016, Stettinger et al., 2017). In Abschnitt 5.1.1 wird dieses Verhältnis genauer analysiert.

Der Verlust von Daten muss auch berücksichtigt werden, da er bei einer Nachrichtenübermittlung über ein Netzwerk nicht ausgeschlossen werden kann. Dieser wird wie in Tranninger et al. (2016) mithilfe einer Funktion $\delta_i$ abgebildet. Zum Zeitpunkt $t_n$ gilt

$$\delta_{i,n} = \begin{cases} 1, & \text{falls Daten empfangen wurden und} \\ 0, & \text{falls keine Daten empfangen wurden.} \end{cases} \tag{2-12}$$

Zusammen mit Gl. (2-10) ergibt sich damit für das Eingangssignal

$$\hat{u}_{i,n} = \hat{y}_{j,n} = \delta_{i,n}\left(y_{j,n-k_i}\right) + \left(1 - \delta_{i,n}\right)\hat{y}_{j,n-1} \text{ mit } t_n \leq t < t_{n+1}, \tag{2-13}$$

wenn nach einer verloren gegangenen Nachricht der zuletzt empfangene Signalwert wieder verwendet wird (Tranninger et al., 2016). Somit wirkt sich ein Datenverlust zum Zeitpunkt $t_n$ als eine punktuelle Erhöhung der Latenzzeit um einen Makroschritt aus.

Neben der Netzwerkeffekte wirken sich auch mögliche Echtzeitverletzungen der virtuellen Prototypen auf den Datenaustausch zwischen den Prototypen aus (vgl. Abschnitt 2.2.2). Solche Echtzeitverletzungen beeinflussen die Echtzeit-Co-Simulation auf dieselbe Weise wie ein Datenverlust bei der Nachrichtenübermittlung. Ein Ausgangssignal $y_{i,n}$, welches zum Zeitpunkt $t_n$ aufgrund einer Echtzeitverletzung nicht versendet werden kann, ist vom Empfänger nicht von einer im Netzwerk verloren gegangenen Nachricht zu unterscheiden. Echtzeitverletzungen bewirken demnach ebenfalls eine punktuelle Erhöhung der Latenzzeit.

Rauscheffekte, die bei der Messung physikalischer Größen in realen Prototypen auftreten, wirken sich auch auf die Ausgangssignale aus. In der Regel wird das Rauschen direkt von einem Filter gedämpft, das optimal auf die verwendete Messtechnik und den Frequenzbereich der zu messenden Größe abgestimmt ist, wodurch der Einfluss des Rauschens auf die Signalwerte kompensiert wird. Allerdings wird dabei aufgrund der Filtereigenschaften das Signal verzögert. Je nachdem wie groß diese Verzögerung der Filterung im Vergleich zu den Latenzzeiten in der Datenübertragung ist, kann diese zu der Latenzzeit der Datenübertragung addiert und von einem Kopplungsalgorithmus kompensiert werden.

Somit sind Latenzzeiten die Ursache für den Kopplungsfehler der Echtzeit-Co-Simulation, welcher genau wie bei einer Co-Simulation nach Gl. (2-6) definiert ist. Daher wird bei einer Echtzeit-Co-Simulation ein Kopplungsalgorithmus $f(\cdot)$ benötigt, der die Latenzzeiten kompensiert. Dieser schätzt den kontinuierlichen Verlauf des Ausgangssignals eines Prototyps am Eingang des anderen gekoppelten Prototyps auf Basis der zur Verfügung stehenden vergangenen Signalwerten von $y_j$, sodass

$$\hat{u}_i(t) = \hat{y}_j(t) = f\left(y_j, k_i, t\right) \text{ für } t_n \leq t < t_{n+1} \tag{2-14}$$

gilt und der Kopplungsfehler minimiert wird. Im Vergleich zur Co-Simulation ist die Rekonstruktion der idealen Eingangssignale im Zeitintervall $[t_n, t_{n+1})$ einer Echtzeit-Co-Simulation zusätzlich erschwert, da nur vergangene Signalwerte verfügbar sind. Aus demselben Grund ist der Kopplungsfehler unter Verwendung der naiven Zero-Oder-Hold Extrapolation wie in Gl. (2-13) bei der Echtzeit-Co-Simulation im Allgemeinen größer, wodurch die Notwendigkeit eines Kopplungsalgorithmus steigt.

Nicht alle Prototypen eines RVPs sind über ein Netzwerk gekoppelt. Geht der RVP, wie in Abschnitt 1.1.2 gezeigt, aus einer domänenübergreifenden MIL-Simulation hervor, werden meist die Simulationsmodelle mehrerer virtueller Prototypen auf einem einzigen Computer berechnet. Diese Kopplungen unterliegen in der Echtzeit-Co-Simulation zwar auch der Anforderung der Echtzeitfähigkeit, wodurch meist eine parallele Ausführungsreihenfolge der Simulationsmodelle notwendig ist, aber unterscheiden sich ansonsten nicht von den Prototypenkopplungen einer Co-Simulation. Der Grund dafür ist, dass der Datenaustausch zwischen den Modellierungswerkzeugen auf einem Computer sehr schnell und zuverlässig ist, da ein lokaler Server oder ein gemeinsam genutzter Speicherbereich verwendet werden kann. Während eines Echtzeit-Makrozeitschritts kann daher jedes Simulationsmodell die zeitliche Integration des entsprechenden Zeitintervalls in Simulationszeit vornehmen und anschließend die Co-Simulations-Middleware den Datenaustausch koordinieren.

Eine Echtzeit-Co-Simulation ist aufgrund des verwendeten Netzwerks und der weichen Echtzeitbedingungen der virtuellen Prototypen im Allgemeinen nicht deterministisch. Daher sollten alle Netzwerkeffekte, sowie Rauscheinflüsse und Echtzeitverletzungen der virtuellen Prototypen zur Laufzeit überwacht werden, damit deren Einfluss auf das Systemverhalten abgeschätzt werden kann.

## 2.3 Beispielsysteme für die Anwendung der entwickelten Methodik

Zur Bewertung und zum Vergleich von Kopplungsalgorithmen für RVPen werden in dieser Arbeit an mehreren Stellen Beispielsysteme mit unterschiedlichen Eigenschaften verwendet, welche im Folgenden eingeführt werden. Dabei handelt es sich jeweils um simulierte RVPen, die durch die lokale Kopplung zweier virtueller Prototypen auf einem Computer entstehen und dessen Netzwerkeffekte in Form von Verzögerungszeiten beim Datenaustausch zwischen den Prototypen künstlich erzeugt werden. Dies hat den Vorteil, dass zu jedem Beispielsystem durch ideale, verzögerungsfreie Kopplung der beiden virtuellen Prototypen eine Referenzlösung berechnet werden kann, welche die Kopplungsalgorithmen bei perfekter Kompensation der Netzwerkeffekte reproduzieren sollen.

### 2.3.1 Zweimassenschwinger

Ein Zweimassenschwinger ist ein mechanisches System, dessen Schwingungsverhalten bei Vernachlässigung von Reibeffekten mit einem linearen mathematischen Modell beschreibbar ist. Abb. 2 zeigt einen gedämpften Zweimassenschwinger, bei dem die beiden Massen sowohl mit festen Rändern als auch untereinander über Feder-Dämpfer-Elemente verbunden sind.

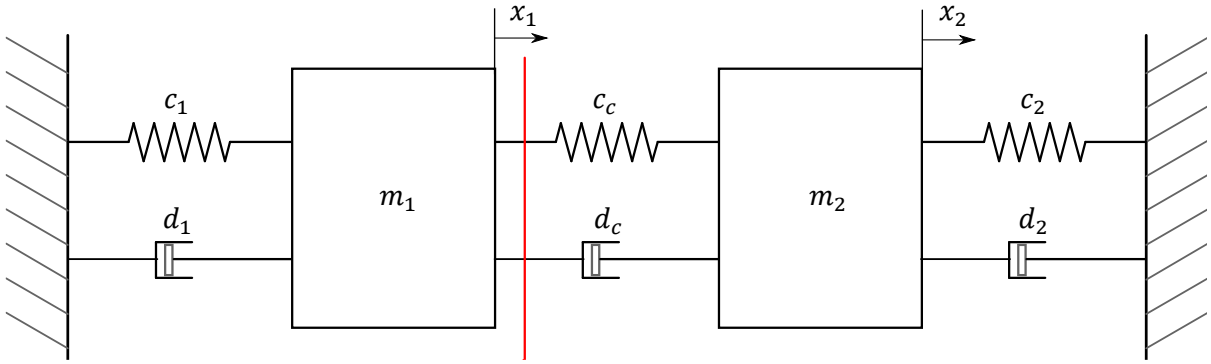


**Abb. 2.7** Linearer, gedämpfter Zweimassenschwinger

In dieser Arbeit sind die Parameter des Zweimassenschwingers nach Tab. 2.1 so gewählt, dass beide Massen mit verschiedenen Eigenfrequenzen schwingen, wenn sie nicht verbunden sind.

**Tab. 2.1** Systemparameter des Zweimassenschwingers

| Parameter | Wert |
|---|---|
| $m_1; m_2$ | 1; 0,1 |
| $c_1; c_c; c_2$ | 1; 2; 10 |
| $d_1; d_c; d_2$ | 0,01; 0,001; 0,001 |

In der Literatur ist der Zweimassenschwinger ein häufig verwendetes Beispielssystem zur Untersuchung gekoppelter Systeme (Busch, 2016, Li, 2017, Tandon et al., 2013). Er wird dazu an der roten Linie aus Abb. 2 in zwei Systeme unterteilt, die untereinander zu den Makrozeitpunkten Signale austauschen. Bei der hier verwendeten Positions-Kraft-Kopplung übermittelt das linke System die Position $x_1$ und Geschwindigkeit $\dot{x}_1$ der linken Masse an das rechte System. Dieses kann daraus die Kraft $f_c$ berechnen, die aufgrund des mittleren Feder-Dämpfer-Elements auf beide Massen wirkt und diese an das linke System zurückgeben. Die Systemgleichungen der beiden linearen Systeme haben die Form

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \\ \boldsymbol{y} &= \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}\boldsymbol{u}, \end{aligned} \tag{2-15}$$

wobei sich der Zustand $\boldsymbol{x}$ aus Position und Geschwindigkeit der jeweiligen Masse zusammensetzt und die Ausgangsgrößen eines Systems $\boldsymbol{y}$ der Eingangsgröße $\boldsymbol{u}$ des anderen Systems entspricht. Die Matrizen des ersten, linken Systems ergeben sich zu

$$\boldsymbol{A}_1 = \begin{bmatrix} 0 & 1 \\ -\frac{c_1}{m_1} & -\frac{d_1}{m_1} \end{bmatrix}, \qquad \boldsymbol{B}_1 = \begin{bmatrix} 0 \\ \frac{1}{m_1} \end{bmatrix},$$
$$\boldsymbol{C}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \boldsymbol{D}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{2-16}$$

und die des zweiten, rechten Systems zu

$$\boldsymbol{A}_2 = \begin{bmatrix} 0 & 1 \\ -\frac{c_2 + c_c}{m_2} & -\frac{d_2 + d_c}{m_2} \end{bmatrix}, \qquad \boldsymbol{B}_2 = \begin{bmatrix} 0 & 0 \\ \frac{c_c}{m_2} & \frac{d_c}{m_2} \end{bmatrix},$$
$$\boldsymbol{C}_2 = [c_c \quad d_c], \qquad \boldsymbol{D}_2 = [-c_c \quad -d_c]. \tag{2-17}$$

Die Netzwerkeffekte eines RVPs werden simuliert, indem jedes Koppelsignal um $k$ Zeitschritte verzögert wird, wobei sich $k$ aus Gl. (2-11) ergibt. Pro Koppelsignal befindet sich ein Kopplungsalgorithmus jeweils an den Eingängen der entsprechenden Prototypen und kompensiert nach Gl. (2-14) zum einen die Verzögerung $k$ und rekonstruiert zum anderen den kontinuierlichen Signalverlauf zwischen den Makrozeitpunkten.

### 2.3.2 Nichtlinearer Zweimassenschwinger

Als nichtlineares Beispielsystem dient ein Zweimassenschwinger, der wie in Abb. 2.8 gezeigt um einen einseitigen mechanischen Anschlag ergänzt wird. Die Masse $m_1$ kann sich dadurch aus der Ruhelage nur um $x_{1,stopp}$ in negative $x_1$-Richtung bewegen und stößt dort hart an. Der Anschlag ist als teilelastischer Stoß mit der Stoßzahl $r_{stopp}$ modelliert, wodurch für die Geschwindigkeit zum Zeitpunkt des Stoßes

$$\dot{x}_1' = -r_{stopp} \cdot \dot{x}_1 \tag{2-18}$$

gilt (Gall, 2001). Das Vorgehen bei der Aufteilung des Systems in zwei gekoppelte Prototypen, die Modellierung der Netzwerkeffekte und die Wahl der Systemparameter entspricht der des linearen Zweimassenschwingers.

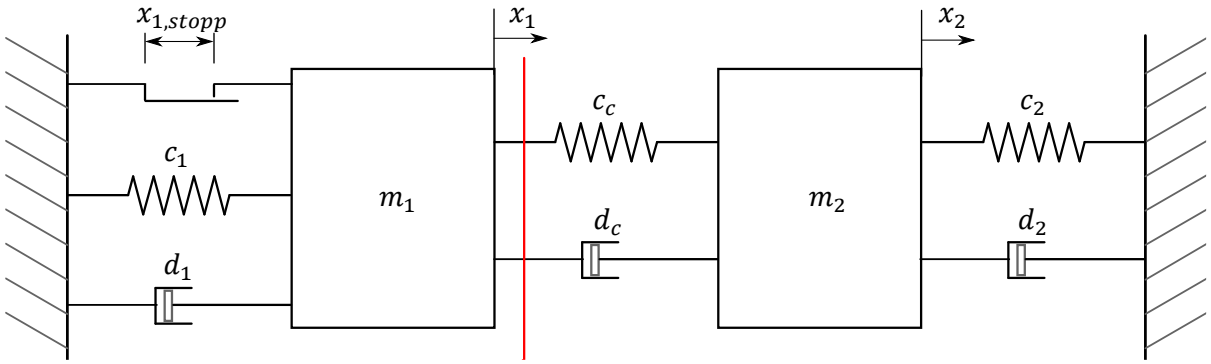


**Abb. 2.8** Nichtlinearer, gedämpfter Zweimassenschwinger mit einseitigem Anschlag.

Die Nichtlinearität aufgrund des mechanischen Anschlags führt zu einem Strukturbruch im dynamischen Verhalten des Zweimassenschwingers. Der kontinuierliche Verlauf der Koppelsignale im linearen Bereich $x_1 > x_{1,stopp}$ wird an der Stelle $x_1 = x_{1,stopp}$ durch Diskontinuitäten in allen Koppelsignalen unterbrochen.

## 2.4 Vergleichsmetriken

In dieser Arbeit werden zwei Metriken zur Bewertung und Vergleich von Kopplungsalgorithmen verwendet. Der akkumulierte Kopplungsfehler wird über die Metrik von Sprague und Geers (2004) quantifiziert. Diese besteht aus drei Werten und unterscheidet zwischen Amplitudenfehler $M_{S\&G}$, Phasenfehler $P_{S\&G}$ und der Kombination $C_{S\&G}$ aus beidem, wodurch die Verwendung dieser Metrik neben der reinen Fehlerquantifizierung auch Rückschlüsse auf die Veränderung des Systemverhaltens des RVPs durch den Kopplungsalgorithmus zulässt. Sei $\boldsymbol{Y}$ der Vektor aus $N$ Signalwerten an Makrozeitpunkten eines Koppelsignals am Ausgang eines Prototyps und $\hat{\boldsymbol{Y}}$ der Vektor der zeitlich entsprechenden Vorhersage von $\boldsymbol{Y}$ am Eingang eines anderen Prototyps, gilt nach Sprague und Geers (2004)

$$M_{S\&G} = \sqrt{\frac{\sum_{i=1}^{N} y_i^{\,2}}{\sum_{i=1}^{N} \hat{y}_i^2}} - 1,$$

$$P_{S\&G} = \frac{1}{\pi}\cos^{-1}\left(\frac{\sum_{i=1}^{N} y_i \hat{y}_i}{\sqrt{\sum_{i=1}^{N} y_i^2 \sum_{i=1}^{N} \hat{y}_i^2}}\right) \text{ und} \tag{2-19}$$

$$C_{S\&G} = \sqrt{M_{S\&G}^2 + P_{S\&G}^2}.$$

Liegt außerdem eine Referenzlösung als Vektor $\boldsymbol{Y}_{ref}$ mit $N$ Einträgen vor, die bei einer verzögerungsfreien Kopplung der Prototypen des RVPs zu den Makrozeitpunkten aufgezeichnet wurde, wird die Abweichung der Koppelsignale der Referenzlösung und des entsprechenden Signalvektors $\boldsymbol{Y}$ des RVPs über den mittleren absoluten Fehler

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |y_i - y_{ref,i}| \tag{2-20}$$

ermittelt.

# 3 Algorithmen zur Kopplung Real-Virtueller Prototypen

Die Verwendung von RVPen beschleunigt den Entwicklungsprozess komplexer mechatronischer Systeme aufgrund der Möglichkeit einer durchgängigen Nutzung von Simulationsmodellen (s. Kap. 1). Allerdings erschwert der vor allem auf Latenzzeiten zurückzuführende Kopplungsfehler (s. Abschnitt 2.2) den Einsatz von RVPen, da keine generell verwendbare Kopplungsmethodik zur Kompensation der Latenzzeiten existiert.

In diesem Kapitel werden daher neuartige Kopplungsalgorithmen zur Kompensation von Latenzzeiten in RVPen entwickelt, die das Kernstück der Kopplungsmethodik bilden. Dazu werden zunächst die Problemstellung und die daraus resultierenden Anforderungen an den Kopplungsalgorithmus definiert. Anschließend wird in Abschnitt 3.2 ein Verfahren zur Extrapolation im Fehlerraum vorgestellt und daraus zwei neuartige lineare Kopplungsalgorithmen abgeleitet. In Abschnitt 3.3 folgt eine Klassifikation der aus Abschnitt 1.2.2 bekannten und der neuartigen Kopplungsalgorithmen als Autoregressive Integrated Moving Average (ARIMA) Modell. Dies schafft eine Vergleichsbasis, wodurch die Eigenschaften der aus unterschiedlichen Ingenieursdisziplinen stammenden Algorithmen bewertbar werden. Abschnitt 3.4 zeigt die Anwendung der verglichenen Kopplungsalgorithmen auf lineare RVP. Mittels neuronaler Netze werden die neuartigen Algorithmen anschließend um Nichtlinearitäten erweitert. Zusammen mit einer Strategie zum online Lernen ermöglicht dies die Vorhersage nichtlinearer Signalverläufe, welche abschließend in Abschnitt 3.6 demonstriert wird.

## 3.1 Problemstellung und Anforderungen

Zur Laufzeit eines RVPs findet der Datenaustausch zwischen den Prototypen unter Echtzeitbedingungen über ein Netzwerk statt, wobei die ausgetauschten Signale einer solchen Echtzeit-Co-Simulation verzögert werden (vgl. Abschnitt 2.2). Diese Latenzzeiten werden mittels eines Kopplungsalgorithmus kompensiert, indem der Verlauf der Koppelsignale vorhergesagt wird. Damit der Kopplungsalgorithmus flexibel einsetzbar ist, basiert er ausschließlich auf bisherigen Werten des Signals zu Makrozeitpunkten. Ableitungen oder Signalwerte aus vorangegangenen Simulationen werden nicht berücksichtigt, da diese im Allgemeinen nicht verfügbar sind. Weiterhin wird der Kopplungsalgorithmus auf jedes Eingangssignal einzeln angewandt. Die dadurch gewonnene Flexibilität hat allerdings zur Folge, dass Zusammenhänge

zwischen verschiedenen Koppelsignalen, wie beispielsweise bei bilateralen Energiekopplungen, nicht in die Vorhersage einfließen und dadurch Erhaltungssätze verletzt werden können. Somit handelt es sich bei den in dieser Arbeit untersuchten Kopplungsalgorithmen um ein Problem der univariaten Zeitreihenvorhersage (Univariate Time Series Forecasting).

In Abschnitt 2.2.3 wurde bereits die Beziehung zwischen einem beliebigen Eingangs-Ausgangspaar in einem RVP unter Beachtung von Netzwerkeffekten und einem allgemeinen Kopplungsalgorithmus $f(\cdot)$ hergeleitet (vgl. Gl. (2-14)). Zur leichteren Lesbarkeit werden in diesem Kapitel die Indizes $i$ und $j$ der Eingänge und Ausgänge vernachlässigt. Für den zeitkontinuierlichen Verlauf eines beliebigen Eingangssignals gilt demnach

$$\hat{u}(t) = \hat{y}(t) = f\left(\boldsymbol{y}_{k,m}, k, t\right) \text{ mit } \boldsymbol{y}_{k,m} = \begin{bmatrix} y_{n-k} \\ y_{n-k-1} \\ \vdots \\ y_{n-k-m+1} \end{bmatrix} \text{ für } t_n \leq t < t_{n+1}. \tag{3-1}$$

Die Zeitfolge $(t_n, t_{n+1}, \dots)$ ergibt sich durch Abtastung mit der konstanten Makroschrittweite $\Delta T$. Die Latenzzeit berechnet sich als Vielfaches der Makroschrittweite zu $k\Delta T$ mit $k \in \mathbb{N}$ (vgl. Abschnitt 2.2.3). Der Kopplungsalgorithmus $f(\cdot)$ schätzt aus den $m \in \mathbb{N}$ vergangenen Signalwerten $\boldsymbol{y}_{k,m}$ des verzögerten Ausgangssignals $y$ einen aktuellen und zeitkontinuierlichen Verlauf $\hat{y}(t)$, der entsprechend der Kopplungsbedingung aus Abschnitt 2.2.3 dem geschätzten Eingangsverlauf $\hat{u}(t)$ entspricht. Dieses Kapitel widmet sich der Bestimmung des Kopplungsalgorithmus $f(\cdot)$ zur Schätzung von $\hat{y}(t)$.

Es ist sinnvoll den Kopplungsalgorithmus aus Gl. (3-1) in eine Kompensation und eine Rekonstruktion aufzuteilen, da nicht jeder gekoppelte Prototyp einen kontinuierlichen Signalverlauf am Eingang benötigt. Liegt ein Prototyp beispielsweise als ausführbarer Softwarecode (vgl. Abschnitt 2.1.2) oder als Hardwarekomponente vor, werden dessen Berechnungen bzw. die Ansteuerung der Hardware in einem festen Zeitraster durchgeführt, welches meist der Makroschrittweite entspricht. Für diese Prototypen ist es ausreichend durch eine Kompensation $g(\cdot)$ der Latenzzeit nur den aktuellen Signalwert

$$\hat{y}_n = g(\boldsymbol{y}_{k,m}, k) \text{ mit } \boldsymbol{y}_{k,m} = \begin{bmatrix} y_{n-k} \\ y_{n-k-1} \\ \vdots \\ y_{n-k-m+1} \end{bmatrix} \tag{3-2}$$

zum Zeitpunkt $t_n$ zu schätzen, da bis $t_{n+1}$ keine weiteren Werte benötigt werden. Trotzdem sollten diese diskreten Werte durch Abtastung einer kontinuierlichen Funktion generiert werden, da Signalwertsprünge in bestimmten Eingangssignalen physikalischer Systeme nicht realistisch sind und zu unerwünschten Systemverhalten führen können.

Der zweite Teil des Kopplungsalgorithmus, die Rekonstruktion $h(\cdot)$ des kontinuierlichen Signalverlaufs

$$\hat{y}(t) = h(\hat{\boldsymbol{y}}_l, t) \text{ mit } \hat{\boldsymbol{y}}_l = \begin{bmatrix} \hat{y}_n \\ \hat{y}_{n-1} \\ \vdots \\ y_{n-l+1} \end{bmatrix} \text{ für } t_n \leq t < t_{n+1} \tag{3-3}$$

ist nur an den Eingangssignalen von Prototypen notwendig, die zum Beispiel aufgrund eines physikalischen Simulationsmodells Eingangswerte für Mikroschritte benötigen. Die Berechnung erfolgt auf Basis der zuvor geschätzten Signalwerte $\hat{\boldsymbol{y}}_l$ und entspricht exakt dem Rekonstruktionsproblem einer Co-Simulation (vgl. Gl. (2-4)). Der Kopplungsalgorithmus $f(\cdot)$ aus Gl. (3-1) entsteht aus einer Addition der Kompensation $g(\cdot)$ und der Rekonstruktion $h(\cdot)$.

An den neuartigen Kopplungsalgorithmus zur Kompensation von Latenzzeiten werden in dieser Arbeit folgende Anforderungen gestellt:

- **Minimierung des Kopplungsfehlers.** Erfüllen von Gl. (3-1), sodass der Kopplungsfehler

  $$e(t) = d\big(y(t), \hat{y}(t)\big) \text{ mit } t_n \leq t < t_{n+1} \tag{3-4}$$

  unter Verwendung eines geeigneten Distanzmaßes $d$ minimiert wird. Da $y(t)$ nur an den Makrozeitpunkten zur Verfügung steht, kann der Kopplungsfehler nur dort exakt bestimmt werden und es gilt

  $$e_n = d(y_n, \hat{y}_n). \tag{3-5}$$

- **Anwendbarkeit auf nichtlineare, diskontinuierliche Koppelsignale.** Die Dynamiken gekoppelter Prototypen sind in der Regel nichtlinear und zeitvariant. Dies wirkt sich auf verschiedene Weise auf die Koppelsignale aus. Es ist zu unterscheiden, ob sich die Koppelsignale trotz Nichtlinearität stetig fortsetzen oder Diskontinuitäten enthalten. Koppelsignale von RVPen werden als diskrete Zeitreihen übertragen, für welche die Eigenschaft der Stetigkeit nie erfüllt ist und sie daher streng genommen zu jedem Makrozeitpunkt eine Diskontinuität oder Unstetigkeitsstelle aufweisen. Allerdings entstehen die Koppelsignale durch Abtastung einer kontinuierlichen Größe eines realen oder virtuellen Prototyps, für die Stetigkeit durch die Abwesenheit von Unstetigkeitsstellen bzw. „Sprüngen“ im Signalverlauf definiert ist. In dieser Arbeit wird daher definiert, dass in einem Koppelsignal zum Zeitpunkt $t_n$ eine Diskontinuität bzw. Unstetigkeitsstelle vorliegt, wenn die kontinuierliche Größe des realen oder virtuellen Prototyps, aus dem das Koppelsignal durch Abtastung hervorgeht, im Zeitraum $t_{n-1} < t \leq t_n$ eine Unstetigkeitsstelle besitzt.

  Aufgrund des kurzen Vorhersagehorizonts können kontinuierliche, nichtlineare Koppelsignale mit linearen Funktionen approximiert werden, wodurch eine Kompensation von Latenzzeiten mit linearen Kopplungsalgorithmen möglich ist. Stellen, an denen Diskontinuitäten in den Koppelsignalen auftreten, weisen dagegen auf eine plötzliche Änderung des dynamischen Verhaltens des sendenden Prototyps hin. Eine korrekte Vorhersage des

Signalverlaufs an den Unstetigkeitsstellen ist dagegen nur möglich, wenn der Kopplungsalgorithmus selbst nichtlinear ist (s. Abschnitt 3.5).

- **Lernfähigkeit.** Der Algorithmus soll lernfähig sein, um den Kopplungsfehler zur Laufzeit des RVPs durch Anpassung an das entsprechende Koppelsignal weiter zu verringern. Weiterhin ist eine Vorhersage der Unstetigkeitsstellen im Signalverlauf nur mit einem lernfähigen Algorithmus möglich, da diese Stellen immer signalspezifisch sind. Allerdings soll der entwickelte Kopplungsalgorithmus nicht auf eine Lernphase angewiesen sein, sondern bereits mit einer generischen Startparametrierung sinnvolle Vorhersagen liefern und die Latenzzeiten kompensieren.

- **Wenige Designparameter.** Die Anzahl an Parametern, welche vom Anwender eingestellt werden müssen, soll gering sein. Für alle Parameter sollen Richtlinien für deren Einstellung gegeben werden, damit kein Expertenwissen für die Verwendung des Kopplungsalgorithmus notwendig ist.

- **Geringer Berechnungsaufwand.** Der Berechnungsaufwand des Kopplungsalgorithmus soll möglichst gering sein, da RVPen der Echtzeitanforderung unterliegen.

- **Bekannter Einfluss auf Systemverhalten.** Die Auswirkungen des Kopplungsalgorithmus auf das Systemverhalten des RVPs sollen analysierbar sein (s. Kap. 4).

- **Einsetzbarkeit bei variablen Latenzzeiten.** Aufgrund der Kommunikation über ein nichtdeterministisches Netzwerk sind die zu kompensierenden Latenzzeiten im Allgemeinen variabel (s. Abschnitt 2.2.3). Der Vorhersagehorizont muss daher zur Laufzeit veränderbar sein.

## 3.2 Extrapolation im Fehlerraum

Die Minimierung der durch Latenzzeiten hervorgerufenen Kopplungsfehler ist die wichtigste Anforderung an einen Kopplungsalgorithmus für RVPen. Daher wird im Folgenden ein Verfahren zur Konstruktion generischer Kopplungsalgorithmen vorgestellt, welches explizit auf dieser Minimierung aufbaut. Das Verfahren wurde in dem im Rahmen dieser Arbeit entstandenen Beitrags (Baumann et al., 2024) veröffentlicht.

### 3.2.1 Verfahren zur Konstruktion eines Kopplungsalgorithmus

Das Verfahren wählt aus einer Kurvenschar aller zur Verfügung stehenden Kurven eines Kopplungsalgorithmus $f_p(\boldsymbol{y}_{k,m}, k, t)$ die zwischen den nächsten Makrozeitschritten verwendete Kurve $f_{\hat{p}}(\boldsymbol{y}_{k,m}, k, t)$ aus. Die Form dieser Kurve ist durch den Parametervektor $\hat{\boldsymbol{p}} \in \mathcal{D}_p$ definiert, der über seine Definitionsmenge beliebig bestimmt ist. Damit ergibt sich mit Gl. (3-1)

der vorhergesagte zukünftige Signalverlauf bis zum Zeitpunkt $t_{n+1}$, an dem ein neuer Signalwert für die weitere Vorhersage zur Verfügung steht, zu

$$\hat{y}(t) = f_{\hat{p}}\left(\boldsymbol{y}_{k,m}, k, t\right) \text{ mit } \boldsymbol{y}_{k,m} = \begin{bmatrix} y_{n-k} \\ y_{n-k-1} \\ \vdots \\ y_{n-k-m+1} \end{bmatrix} \text{ für } t_n \leq t < t_{n+1}. \tag{3-6}$$

Ziel des Verfahrens ist es $f_{\hat{p}}(t)$ so zu wählen, dass der geschätzte Kopplungsfehler am Ende des gültigen Zeitintervalls (vgl. Gl. (3-5))

$$\hat{e}_{n+1} = d(y_{n+1}, \hat{y}_{n+1}) \tag{3-7}$$

gemäß eines Distanzmaßes $d$ minimiert wird.

Für eine gegebene Kurvenschar $f_p$ mit $p \in \mathcal{D}_p$ und ein gegebenes Distanzmaß $d$ schätzt das Verfahren dazu in jedem Makroschritt denjenigen Wert $\hat{p}$ des Parameters $p$, welcher den Kopplungsfehler $\hat{e}_{n+1}$ betragsmäßig minimiert,

$$\hat{p}: \min_{p \in \mathcal{D}_p} \hat{e}_{n+1}(p) = \min_{p \in \mathcal{D}_p} d\left(y_{n+1}, f_p\left(\boldsymbol{y}_{k,m}, k, t_{n+1}\right)\right). \tag{3-8}$$

Kern des Verfahrens ist die Schätzung des Parameters $\hat{p}$ durch Analyse und Extrapolation des vergangenen Signalverlaufs nicht im Signalraum, sondern im Fehlerraum. Das Verfahren gliedert sich in eine Abfolge von vier Schritten, die für jedes auf diese Weise vorhergesagte Signal bei jedem Makrozeitschritt einmal durchlaufen werden.

1. **Ermittle (hypothetische) Kopplungsfehler vergangener Makroschritte.** Anhand der gegebenen, bisherigen Zeitreihe der Koppelgröße $\boldsymbol{y}_{k,m}$ werden an jedem der letzten $q \geq 1$ Zeitpunkte eine Menge von möglichen Extrapolationsfehlern $e_i(p)$ mit $n - k - q < i \leq n - k$ ermittelt. Dies kann entweder analytisch gemäß Gl. (3-7) erfolgen oder durch Approximation, indem $e_i(p)$ nur für $s_i$ Stützstellen $p_{i,j}$ exakt ausgewertet und der Verlauf von $e_i(p)$ zwischen den Stützstellen durch Kurvenanpassung an Modellfunktionen $g_i$ bestimmt wird.

2. **Schätzung zukünftiger Fehlerverläufe.** Mit den berechneten $q$ Fehlerkurven $e_i(p)$ und einem geeigneten Extrapolationsverfahren $E$ wird durch Anwendung von $E$ auf $e_i(p)$ die Fehlerkurve $\hat{e}_{n+1}(p)$ für den nächsten Zeitschritt geschätzt. Analog zum vorherigen Schritt kann die Fehlerkurve $\hat{e}_{n+1}(p)$ analytisch berechnet oder approximiert werden, indem $\hat{e}_{n+1}(p)$ nur für $s_{n+1}$ Stützstellen $p_{n+1,j}$ exakt ausgewertet wird und der Verlauf von $\hat{e}_{n+1}(p)$ zwischen den Stützstellen $e_{n+1}(p_{n,j})$ durch Kurvenanpassung an die Modellfunktion $g_{n+1}$ ermittelt wird.

3. **Finde den kleinsten zukünftigen Fehler.** Mit der Schätzung der nächsten Fehlerkurve $\hat{e}_{n+1}(p)$ wird $\hat{p}$ durch Approximation der Gl. (3-8) bestimmt.

4. **Approximation des zukünftigen Signalverlaufs.** Unter Verwendung von $\hat{p}$ wird der zukünftige Signalverlauf gemäß Gl. (3-6) approximiert.

Das Verfahren lässt sich variieren in

- der Wahl der Kurvenschar $f_p(t)$,
- der Wahl des Distanzmaßes $d$,
- der Anzahl $q$ der betrachteten vergangenen Zeitpunkte,
- der Wahl des Verfahrens $E$ für die Abbildung $e_i \rightarrow \hat{e}_{n+1}$,
- der Wahl des Verfahrens zur Bestimmung desjenigen Wertes $\hat{p}$ von $p$, welcher $\hat{e}_{n+1}(p)$ betragsmäßig minimiert.

Falls die Fehlerkurven $e_i(p)$ oder $\hat{e}_{n+1}(p)$ nicht analytisch hergeleitet werden können, müssen zusätzlich

- die Anzahl $s$ der Stützstellen $p_{i,j}$ bzw. $p_{n+1,j}$,
- sowie die Modellfunktionen $g_i$ bzw. $g_{n+1}$ für die Kurvenanpassung an die Stützstellen

festgelegt werden.

### 3.2.2 Ableitung konkreter Kopplungsalgorithmen

Mithilfe des entwickelten allgemeinen Verfahrens zur Konstruktion von Kopplungsalgorithmen durch Extrapolation im Fehlerraum werden in dieser Arbeit zwei konkrete Algorithmen hergeleitet, welche als Error Space (EROS) Extrapolation, eine Extrapolation im Fehlerraum bezeichnet werden.

Für beide neuartigen Algorithmen bilden dabei Polynome ersten Grades die Kurvenschar

$$f_p(\boldsymbol{y}_{k,m}, k, t) = p_0 + p_1 \left(k + \frac{t-t_n}{\Delta T}\right) \text{ für } t_n \leq t < t_{n+1}. \tag{3-9}$$

Ein linearer Ansatz ist laut den Anforderungen aus Abschnitt 3.1 für einen generischen Kopplungsalgorithmus ausreichend, da aufgrund des kurzen Vorhersagehorizonts kontinuierliche Abschnitte der Koppelsignale mit linearen Funktion approximierbar sind. In Abschnitt 3.5 werden die hier entwickelten linearen Kopplungsalgorithmen mittels neuronaler Netze als

nichtlineare Funktion verallgemeinert, sodass auch Vorhersagen an Unstetigkeitsstellen durch eine Anpassung an die jeweiligen Koppelsignale möglich ist.

Für den latenzzeitkompensierten und rekonstruierten Signalverlauf ergibt sich mit der linearen Kurvenschar

$$\hat{y}(t) = y_{n-k} + \hat{p}_1 \left(k + \frac{t-t_n}{\Delta T}\right) \text{ für } t_n \leq t < t_{n+1}, \tag{3-10}$$

wenn das Signal kontinuierlich fortgesetzt wird, indem der letzte bekannte Signalwert $\hat{p}_{n,0} = y_{n-k}$ als Stützstelle verwendet wird. Zur Bestimmung von $\hat{p}_1$ mittels Extrapolation im Fehlerraum (s. Gl. (3-8)) erfolgt die Schätzung des zukünftigen Fehlerverlaufs mit der Taylorreihe

$$\hat{e}_{n+1}(p) = e_{n-k}(p) + (k+1)\Delta T \frac{d}{dt} e_{n-k}(p) + \frac{1}{2}\left((k+1)\Delta T\right)^2 \left(\frac{d}{dt}\right)^2 e_{n-k}(p) + \cdots \tag{3-11}$$

unter Verwendung des Rückwärtsdifferenzenquotienten zur Approximation der Ableitungen. Die Anzahl $q$ der benötigten vergangenen Fehlerverläufe $e_i(p)$ hängt von der Ordnung der Taylorreihe ab.

**EROS3**

Beim EROS3 genannten Kopplungsalgorithmus wird die Taylorreihe zur Schätzung des zukünftigen Fehlerverlaufs nach der ersten Ordnung abgebrochen, wodurch mit Gl. (3-11)

$$\hat{e}_{n+1}(p) = e_{n-k}(p) + (k+1)\big(e_{n-k}(p) - e_{n-k-1}(p)\big) \tag{3-12}$$

gilt. Wird der absolute Fehler als Distanzmaß $d$ gewählt, ergeben sich die benötigten Fehlerverläufe analytisch zu

$$e_i(p) = \hat{y}_i(p) - y_i = y_{i-k-1} + p_1(k+1) - y_i, \tag{3-13}$$

wodurch hier keine Kurvenanpassung notwendig ist. Mit den $q = 2$ vergangenen Fehlerverläufen

$$\begin{aligned} e_{n-k}(p) &= y_{n-2k-1} + p_1(k+1) - y_{n-k} \quad \text{und} \\ e_{n-k-1}(p) &= y_{n-2k-2} + p_1(k+1) - y_{n-k-1} \end{aligned} \tag{3-14}$$

ist die betragsmäßige Minimierung des Kopplungsfehlers in Gl. (3-8) durch Berechnung von $\hat{e}_{n+1}(p) \stackrel{!}{=} 0$ analytisch lösbar und es ergibt sich

$$\hat{p}_1 = \frac{k+2}{k+1} y_{n-k} - y_{n-k-1} - \frac{k+2}{k+1} y_{n-2k-1} + y_{n-2k-2}. \tag{3-15}$$

Das Einsetzen von $\hat{p}_{n,1}$ in Gl. (3-10) führt zum Kopplungsalgorithmus EROS3

$$\hat{y}(t) = y_{n-k} + \left(\frac{k+2}{k+1}y_{n-k} - y_{n-k-1} - \frac{k+2}{k+1}y_{n-2k-1} + y_{n-2k-2}\right)\left(k + \frac{t-t_n}{\Delta T}\right) \quad \text{für } t_n \le t < t_{n+1}, \tag{3-16}$$

dessen Bezeichnung sich durch die Anzahl der verwendeten vergangenen Signalwerte bei der Kompensation eines einzelnen Makroschritts Verzögerung motiviert.

**EROS4**

Zum Vergleich wird mittels des Verfahrens ein weiterer Kopplungsalgorithmus hergeleitet, welcher für die Kompensation eines einzelnen Makroschritts Verzögerung vier vergangene Signalwerte zur Schätzung des Signalverlaus einbezieht und daher EROS4 genannt wird. Dieser entsteht durch Hinzunahme eines weiteren Summanden der Taylorreihe des geschätzten, zukünftigen Fehlerverlaufs

$$\begin{aligned}\hat{e}_{n+1}(p) = e_{n-k}(p) + (k+1)\big(e_{n-k}(p) - e_{n-k-1}(p)\big) + \\ + \tfrac{1}{2}(k+1)^2\big(e_{n-k}(p) - 2e_{n-k-1}(p) + e_{n-k-2}(p)\big).\end{aligned} \tag{3-17}$$

Analog zum Vorgehen beim EROS3 ergibt sich damit für

$$\begin{aligned}\hat{p}_1 = \tfrac{1}{k+1}(c_1 y_{n-k} - c_2 y_{n-k-1} + c_3 y_{n-k-2} - c_1 y_{n-2k-1} + c_2 y_{n-2k-2} - \\ - c_3 y_{n-2k-3}) \\ \text{mit } c_1 = \tfrac{1}{2}k^2 + 2k + \tfrac{5}{2},\ c_2 = k^2 + 3k + 2 \text{ und } c_3 = \tfrac{1}{2}k^2 + k + \tfrac{1}{2}\end{aligned} \tag{3-18}$$

und somit für den Kopplungsalgorithmus EROS4 der geschätzte Signalverlauf

$$\begin{aligned}\hat{y}(t) = y_{n-k} + \Big(\tfrac{1}{k+1}(c_1 y_{n-k} - c_2 y_{n-k-1} + c_3 y_{n-k-2} - c_1 y_{n-2k-1} + c_2 y_{n-2k-2} - \\ - c_3 y_{n-2k-3})\Big)\left(k + \tfrac{t-t_n}{\Delta T}\right) \text{ für } t_n \le t < t_{n+1}.\end{aligned} \tag{3-19}$$

## 3.3 Diskussion der Kopplungsalgorithmen

In diesem Abschnitt werden die neuartigen Kopplungsalgorithmen EROS3 und EROS4 mit den aus der Literatur bekannten Algorithmen verglichen, welche bereits in Abschnitt 1.2.2 vorgestellt wurden. Außerdem wird die Erfüllung der in Abschnitt 3.1 definierten Anforderungen überprüft. Obwohl die Algorithmen aus verschiedenen Bereichen wie der Regelungstechnik, Systemtheorie und Simulationstechnik stammen und unabhängig voneinander entwickelt wurden, ist es möglich eine einheitliche Vergleichsbasis zu schaffen, indem die Algorithmen in Autoregressive Integrated Moving Average (ARIMA) Modelle überführt werden.

Verglichen werden neben den EROS-Algorithmen der Schätzer auf Basis eines PD-Reglers von Liu et al. (2020), die auf Theorien des Sliding-Mode Beobachters basierende Vorhersagemethode, welche erstmals in Tandon et al. (2013) Erwähnung findet und in dieser Arbeit mit SMB abgekürzt wird, das modellbasierte Kopplungselement von Stettinger et al. (2014b) und der in Stettinger et al. (2017) publizierte rekursive FIR-Filter zur Kompensation von Latenzzeiten. Sie wurden ausgewählt, da sie neben dem Koppelsignal keine weiteren Informationen der angebundenen Prototypen benötigen und somit am allgemeinsten verwendbar sind (vgl. Abschnitt 1.2.2).

### 3.3.1 ARMA und ARIMA Modelle

Autoregressive Moving Average (ARMA) Modelle sind eine Klasse linearer, zeitdiskreter mathematischer Modelle, welche seit ihrer Entdeckung durch Box und Jenkins (1970) intensiv für die Modellierung und Analyse von Zeitreihen verwendet werden. Eines ihrer wichtigsten Anwendungsfelder ist die Vorhersage univariater Zeitreihen mit kurzem Vorhersagehorizont, also exakt die Aufgabe, welche zur Kompensation von Latenzzeiten bei RVPen notwendig ist. Angepasst auf die in dieser Arbeit verwendete Notation ergibt sich mittels eines allgemeinen ARMA($p$,$q$) Modells für den einen Makroschritt in die Zukunft geschätzten aktuellen Signalwert

$$\hat{y}_n = c + \sum_{l=1}^{p} a_l y_{n-l} + \sum_{j=1}^{q} b_j e_{n-j}. \tag{3-20}$$

Er berechnet sich aus dem autoregressiven Teil, welcher aus der Multiplikation von $p$ vergangenen Ausgangswerten $y_{n-l}$ mit jeweils einer Konstanten $a_l$ hervorgeht, dem mit $b_j$ gewichteten gleitenden Durchschnitt über $q$ vergangene absolute Fehler der Vorhersage

$$e_{n-j} = \hat{y}_{n-j} - y_{n-j}, \tag{3-21}$$

sowie einer Konstanten $c$. Durch rekursives Einsetzen in Gl. (3-20) und der Startbedingung $e_{n-q} = 0$ wird ersichtlich, dass bei ARMA Modellen ausschließlich vergangene Werte von $y$ in die Vorhersage von $\hat{y}_n$ einfließen.

ARMA($p$,$q$) Modelle bilden stationäre Zeitreihen ab. Zur Vorhersage nicht stationärer Signale ist daher eine Erweiterung zu einem $d$-fach integrierten ARMA Modell notwendig, wobei $d$ den Differenzierungsgrad der Zeitreihe bezeichnet. Dazu werden im ARMA Modell alle Werte der Zeitreihe $y_n$ durch $d$-malige erste Differenzen $\Delta^d y_n$ ersetzt, wodurch ein ARIMA($p$,$d$,$q$) mit der Form

$$\Delta^d \hat{y}_n = c + \sum_{l=1}^{p} a_l \Delta^d y_{n-l} + \sum_{j=1}^{q} b_j e_{n-j} \tag{3-22}$$

entsteht. Dabei bezeichnet Δ den Differenzen-Operator, der angewendet auf ein Signal, den Unterschied zwischen dem aktuellen Signalwert und dem vorangegangenen darstellt. Es gilt $\Delta y_n = y_n - y_{n-1}$ und bei $d$-maliger Anwendung des Operators somit

$$\Delta^d y_n = \sum_{j=0}^{d} \binom{d}{j} (-1)^j y_{n-j}. \tag{3-23}$$

### 3.3.2 Klassifikation der Kopplungsalgorithmen als ARIMA Modelle

Im Folgenden werden verschiedene Kopplungsalgorithmen als ARIMA($p$,$d$,$q$) Modell klassifiziert, wodurch eine gemeinsame Basis geschaffen und die Eigenschaften der Algorithmen verglichen werden können. Zuerst wird auf die Kopplungsalgorithmen eingegangen, die üblicherweise in der Co-Simulation zur Rekonstruktion der Signale eingesetzt werden (vgl. Abschnitt 2.1.1). Es wird überprüft, ob diese Algorithmen auch für die Vorhersage mehrerer Makroschritte verwendbar sind, wie es zur Kompensation von Latenzzeiten notwendig ist. Anschließend werden die neuartigen, in Abschnitt 3.2 hergeleiteten EROS-Algorithmen und die für den Vergleich ausgewählten Kopplungsalgorithmen aus der Literatur (s. Abschnitt 3.1) ebenfalls als ARIMA($p$,$d$,$q$) Modell klassifiziert und deren Eigenschaften mit den aus der Co-Simulation bekannten Algorithmen zur Rekonstruktion verglichen.

**Zero-Order-Hold (ZOH)**

ZOH ist der trivialste und gleichzeitig der Standardalgorithmus zur Rekonstruktion eines Signals bei der Co-Simulation. Es gilt

$$\hat{y}(t) = y_n \text{ für } t_n \leq t < t_{n+1}, \tag{3-24}$$

wodurch das Signal innerhalb eines Makroschritts konstant gehalten wird. Der Algorithmus ist auch für die Vorhersage ganzer Zeitschritte verwendbar. Für die Vorhersage eines einzelnen Zeitschritts gilt

$$\hat{y}_n = y_{n-1}. \tag{3-25}$$

Nach Gl. (3-22) entspricht dies der Definition eines ARIMA(0,1,0) Modells mit $c = 0$. Es modelliert Zeitreihen, die nicht stationär sind und sich ausgehend vom aktuellen Wert zufällig fortsetzen. Übertragen auf Signale bei RVPs eignet sich ZOH somit, wenn das Signal keinen Trend besitzt oder stark verrauscht ist, da Signalwerte durch ZOH nicht verändert werden. Mit ZOH sind auch Vorhersagen mehrere Zeitschritte $k$ inklusive der Rekonstruktion eines kontinuierlichen Signalverlaufs zwischen den Makroschritten möglich. Für den geschätzten Eingangswert gilt in diesem Fall

$$\hat{y}(t) = y_{n-k} \text{ für } t_n \leq t < t_{n+1}. \tag{3-26}$$

Da dafür keine Berechnungen notwendig sind, sondern lediglich der neuste verfügbare Signalwert so lange verwendet wird, bis ein neuerer vorliegt ist ZOH der Standardalgorithmus bei RVPen.

**First-Order-Hold (FOH)**

Die lineare Extrapolation

$$\hat{y}(t) = y_n + (y_n - y_{n-1})\frac{(t-t_n)}{\Delta T} \text{ für } t_n \leq t < t_{n+1} \tag{3-27}$$

findet ebenfalls Anwendung zur Rekonstruktion von Signalen bei Co-Simulationen und kann mit

$$\hat{y}_n = 2y_{n-1} - y_{n-2} \tag{3-28}$$

genau wie ZOH für die Vorhersage eines einzelnen Zeitschritts eingesetzt werden. In diesem Fall ergibt sich ein ARIMA(0,2,0) Modell mit $c = 0$. Dies bedeutet, dass die Zeitreihe linear und mit der Annahme fortgesetzt wird, dass deren zweite Ableitung gleich Null ist. FOH bietet Vorteile gegenüber ZOH, sobald der zu schätzende Signalverlauf einen Trend erhält, da die erste Ableitung des Signals mittels des Differenzenquotienten approximiert wird und in die Vorhersage einfließt. Genau wie beim ZOH kann auch mit FOH der Signalverlauf bei einer Latenzzeit von $k$ Makroschritten vorhergesagt und rekonstruiert werden, wobei sich der geschätzte Verlauf zu

$$\hat{y}(t) = y_{n-k} + (y_{n-k} - y_{n-k-1})\left(k + \frac{t-t_n}{\Delta T}\right) \text{ für } t_n \leq t < t_{n+1} \tag{3-29}$$

ergibt.

**Weitere Algorithmen zur Rekonstruktion**

Neben den häufig verwendeten Algorithmen zur Rekonstruktion ZOH und FOH wurden in den letzten Jahren einige fortschrittliche Verfahren entwickelt, um den Kopplungsfehler der Rekonstruktion in Co-Simulationen zu verringern. Diese sind allerdings nicht auf das Problem der Kompensation von Latenzzeiten bei RVPen übertragbar, da die Randbedingungen bei einer Co-Simulation und der Kompensation von Latenzzeiten nicht dieselben sind.

Das Nearly Energy Preserving Coupling Element (NEPCE) von Benedikt (2012) verwendet eine Regelung, um einen Energieverlust bzw. eine Energiezunahme beim Signalaustausch zwischen den gekoppelten Systemen über die Zeit auszugleichen. Dabei nutzt es die Tatsache, dass diese Energiedifferenz eines vergangenen Zeitschritts mit hoher Genauigkeit bestimmt

werden kann, wenn Signalwerte zu Mikroschritten vorliegen. Bei RVPs kann das NEPCE somit nicht verwendet werden, da dem Kopplungsalgorithmus keine Informationen zu Mikroschritten zur Verfügung stehen.

Drenth (2016) präsentiert ein Anti-Aliasing Filter, das ähnlich zum NEPCE verhindern soll, dass durch die Rekonstruktion der Koppelsignale die Energiebilanz im gekoppelten System verändert wird. Dieses Filter kann nur auf entgegen gerichtete physikalische Koppelsignale angewendet werden, welche zusammen einen Energiefluss beschreiben (power bonds). Dies schränkt das Einsatzfeld des Anti-Aliasing Filter ein, wodurch es den Anforderungen für einen Kopplungsalgorithmus aus Abschnitt 3.1 nicht genügt und somit nicht für die Anwendung bei RVPs in Betracht gezogen wird.

Sadjina und Pedersen (2016) betrachten ebenfalls solche Energie-Bindungen und sorgen für Energieerhaltung in der Kopplung, indem sie die Makroschrittweite dynamisch verändern. Wie bereits in Abschnitt 2.2 erläutert ist die Makroschrittweite bei RVPs durch die Bandbreite des Netzwerks und die Echtzeitanforderung definiert und kann nicht beliebig angepasst werden, wodurch auch dieser Ansatz nicht auf einen Kopplungsalgorithmus von RVPs übertragbar ist.

**PD-Regler**

Nun folgt eine Betrachtung der vier Kopplungsalgorithmen für RVP, die bereits in Abschnitt 1.2.2 vorgestellt und für den Vergleich ausgewählt wurden. Liu et al. (2020) schlagen eine Schätzung des Signalverlaufs

$$x_{com}(t) = x_d(t) + K_\tau \dot{x}_d(t) \tag{3-30}$$

vor, indem ähnlich der Struktur eines PD-Reglers der um $\tau$ verzögerte Signalwert $x_d(t) = x(t-\tau)$ und dessen mit $K_\tau = \tau$ skalierte zeitliche Änderung addiert werden. Liu et al. (2020) analysiert den PD-Regler nur für kontinuierliche Signale. Der Einfluss des nachrichtenbasierten Datenaustauschs über ein Netzwerk ist durch Diskretisierung mit der Makroschrittweite $\Delta T$ beschreibbar (vgl. Abschnitt 2.2.3), und es ergibt sich für den vorhergesagten Signalwert zum Zeitpunkt $t_n$

$$\hat{y}_n = y_{n-k} + k(y_{n-k} - y_{n-k-1}) = (k+1)\, y_{n-k} - k \cdot y_{n-k-1.} \tag{3-31}$$

Dabei wurde $\dot{x}_d$(t) mit dem Differenzenquotienten approximiert und $x_{com}$ und $x_d$ in die in dieser Arbeit verwendete Notation überführt. Für die Latenzzeit gilt, wie in Abschnitt 2.2.3 definiert,

$$k = \min\{m \in \mathbb{Z} \mid m \geq \frac{\tau}{\Delta T}\}. \tag{3-32}$$

Ein Vergleich der Vorhersage eines einzelnen Zeitschritts durch Einsetzen von $k = 1$ in Gl. (3-31) mit Gl. (3-28) zeigt, dass die Kompensation der Latenzzeiten mittels PD-Regler ebenfalls einem ARIMA(0,2,0) Modell mit $c = 0$ und somit der linearen Extrapolation (FOH) exakt entspricht.

**Sliding-Mode Beobachter (SMB)**

Tandon et al. (2013) definieren ihren Algorithmus zur Kompensation von Latenzzeiten über die Differentialgleichung

$$\dot{\hat{y}}(t) = \dot{y}(t-\tau) + \lambda[y(t-\tau) - \hat{y}(t-\tau)], \tag{3-33}$$

wobei $\dot{\hat{y}}(t)$ die aktuelle zeitliche Änderung der Vorhersage ist und der Designparameter $\lambda$ im Bereich $0 < \lambda < \bar{\lambda} = \frac{\pi}{2\tau}$ vom Anwender festgelegt werden muss. Äquivalent zum Vorgehen beim PD-Regler, ergibt sich nach Diskretisierung und Überführung der Notation für eine Vorhersage eines Zeitschritts

$$\hat{y}_n = 2y_{n-1} - y_{n-2} + b_1 e_{n-1} \tag{3-34}$$

mit $b_1 = 1 - \lambda\Delta T$ und $e_{n-1} = \hat{y}_{n-1} - y_{n-1}$. Dabei handelt es sich um ein ARIMA(0,2,1) Modell mit $c = 0$, der sogenannten linearen exponentiellen Glättung. Anstatt wie beim FOH das Signal unter der Annahme die zweite Ableitung sei gleich Null fortzusetzen, wird hier die zweite Ableitung mittels des Kopplungsfehlers im vergangenen Zeitschritt approximiert. Der Parameter $\lambda$ muss bei diesem Algorithmus vom Anwender innerhalb der definierten Grenzen gewählt werden und skaliert den Einfluss der Glättung. Der Signalverlauf bei einer Vorhersage von $k$ Schritten und Rekonstruktion des kontinuierlichen Verlaufs ergibt sich zu

$$\hat{y}(t) = \hat{y}_n + (y_{n-k} - y_{n-k-1} - \lambda T e_{n-k})\frac{(t-t_n)}{\Delta T} \text{ für } t_n \le t < t_{n+1} \tag{3-35}$$

mit

$$\hat{y}_n = 2y_{n-k} - y_{n-k-1} + b_1 e_{n-k} - \sum_{j=1}^{k-1} b_{j+1}\, e_{n-k-j} \tag{3-36}$$

und $b_{i+1} = b_1 = \lambda\Delta T,\ i \in [1, k-1]$.

**Modellbasiertes Kopplungselement**

Das modellbasierte Kopplungselement von Stettinger et al. (2014b) schätzt den zukünftigen Verlauf der Koppelsignale mittels zeitdiskreter, linearer Modelle der Form

$$\widehat{S}\text{:}\quad \boldsymbol{x}_{n+1} = \begin{bmatrix} 0 & -\tilde{a}_1 \\ 1 & -\tilde{a}_2 \end{bmatrix}\boldsymbol{x}_n + \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \end{bmatrix} u_n \tag{3-37}$$

$$y_n = [0 \quad 1]\boldsymbol{x}_n$$

mit den zu identifizierenden Parametern $\tilde{a}_1$, $\tilde{a}_2$, $\tilde{b}_1$ und $\tilde{b}_2$. Jedes Modell approximiert lokal das Übertragungsverhalten eines Eingangs-Ausgangspaares eines der gekoppelten Prototypen. Nach einer Umformung und der Verwendung der Notation dieser Arbeit ergibt sich für die Vorhersage eines Zeitschritts

$$\hat{y}_n = a_1 y_{n-1} + a_2 y_{n-2} + b_1 e_{n-1} + b_2 e_{n-2}, \tag{3-38}$$

welches einem ARIMA(2,0,2) Modell mit den Koeffizienten $a_1 = (\tilde{b}_2 - \tilde{a}_2)$, $a_2 = (\tilde{b}_1 - \tilde{a}_1)$, $b_1 = -\tilde{a}_2$, $b_2 = -\tilde{a}_1$ und $c = 0$ entspricht. Vorhersagen von mehreren Zeitschritten ergeben sich durch rekursives Einsetzen in die Modellgleichungen. Mit den diskreten, identifizierten Modellen sind nur Vorhersagen zu Makrozeitpunkten möglich, weshalb das modellbasierte Kopplungselement nicht direkt zur Rekonstruktion des kontinuierlichen Signalverlaufs nutzbar ist. Es können dazu entweder die in der Co-Simulation üblichen Algorithmen zur Rekonstruktion verwendet werden oder es wird mithilfe des modellbasierten Kopplungselements neben $\hat{u}_n$ auch $\hat{u}_{n+1}$ vorhergesagt und anschließend zwischen diesen Werten beispielsweise linear interpoliert.

**FIR-Filter**

Stettinger et al. (2017) modellieren für die Kompensation von Latenzzeiten bei RVPs den Verlauf der Koppelsignale mit einem Filter mit endlicher Impulsantwort (FIR-Filter), dessen Funktion mittels einer Differenzengleichung vierter Ordnung

$$y_n = \boldsymbol{\Phi}_n^T \boldsymbol{p}_n \;\text{ mit }\; \boldsymbol{\Phi}_n = \begin{bmatrix} -y_{n-1} \\ -y_{n-2} \\ -y_{n-3} \\ -y_{n-4} \end{bmatrix} \;\text{ und }\; \mathbf{p}_n = \begin{bmatrix} \tilde{a}_1 \\ \tilde{a}_2 \\ \tilde{a}_3 \\ \tilde{a}_4 \end{bmatrix} \tag{3-39}$$

ausgedrückt wird. Genau wie die bereits diskutierten Methoden kann auch das FIR-Filter in ein ARIMA Modell überführt werden. Mit angepasster Notation gilt demnach

$$\hat{y}_n = \sum_{l=1}^{4} a_l y_{n-l}, \tag{3-40}$$

welches einem ARIMA(4,0,0) Modell mit den Gewichtungsfaktoren $a_i = -\tilde{a}_i$ für $i \in [1,4]$ und $c = 0$ entspricht. Somit handelt es sich um einen autoregressiven Algorithmus, der direkt den Signalwert und nicht dessen Differenz vorhersagt, wie die Algorithmen mit $d > 0$. Die vier Parameter des FIR-Filters sind unbekannt und müssen daher ähnlich wie beim modellbasierten Kopplungselement in einer Lernphase identifiziert werden, bevor das Filter zur Kompensation der Latenzzeiten eingesetzt werden kann. Stettinger et al. (2017) merken an, dass die

Parameter $a_i$ auch zeitvariant sein können, um die Zeitpunkte von Diskontinuitäten im Signalverlauf zu lernen und somit vorhersagen zu können. Dieses Konzept wird von den Autoren allerdings nicht weiterverfolgt.

Die Vorhersage mehrerer Zeitschritte mit dem FIR-Filter erfolgt rekursiv mit Gl. (3-40). Weiterhin ist eine Rekonstruktion des Signalverlaufs zwischen den Makrozeitschritten mit dem FIR-Filter nicht direkt möglich. Es muss genau wie beim modellbasierten Kopplungselement auf eine in der Co-Simulation übliche Algorithmen zur Rekonstruktion oder eine Interpolation zwischen zwei geschätzten Signalwerten zurückgegriffen werden.

**EROS3**

In Gl. (3-16) ist der Signalverlauf vom EROS3 bei einer Vorhersage von $k$ Schritten und Rekonstruktion des kontinuierlichen Verlaufs zwischen den einzelnen Vorhersagen bereits gegeben. Die Vorhersage vom EROS3 für einen Zeitschritt ergibt sich damit mit $k = 0$ und $t = t_{n+1}$ zu

$$\hat{y}_n = 3y_{n-1} - 3y_{n-2} + y_{n-3}. \tag{3-41}$$

Nach Gl. (3-22) handelt es sich somit um ein ARIMA(0,3,0) Modell mit $c = 0$.

**EROS4**

Analog zu EROS3 gilt durch Einsetzen von $k = 0$ und $t = t_{n+1}$ in Gl. (3-19) für die Vorhersage eines Zeitschritts mit EROS4

$$\hat{y}_n = 3y_{n-1} - 3y_{n-2} + y_{n-3} + \frac{1}{2}(y_{n-1} - 3y_{n-2} + 3y_{n-3} - y_{n-4}). \tag{3-42}$$

welches als ARIMA(1,3,0) Modell mit $a_1 = \frac{1}{2}$ und $c = 0$ klassifizierbar ist.

### 3.3.3 Überprüfung der Anforderungen

Nachdem alle Kopplungsalgorithmen in ARIMA Modelle überführt wurden, werden nun ihre Eigenschaften verglichen und die in Abschnitt 3.1 gestellten Anforderungen überprüft. Tab. 3.1 gibt eine entsprechende Übersicht.

**Tab. 3.1** Übersicht über die Eigenschaften verschiedener linearer Kopplungsalgorithmen

| Algorithmus | ARIMA Modell | Anzahl verwendeter Signalwerte $m$ bei $k$ Schritten Verzögerung | Anzahl Parameter | Ohne Lernen einsetzbar? | Direkte Abhängigkeit von $k$? |
|---|---|---|---|---|---|
| ZOH | ARIMA(0,1,0) | 1 | 0 | Ja | Ja |
| FOH | ARIMA(0,2,0) | 2 | 0 | Ja | Ja |
| PD-Regler | | | | | |
| SMB | ARIMA(0,2,1) | $n$ | 1 | Ja | Ja |
| Modellbasierte Kopplung | ARIMA(2,0,2) | $n$ | 4 | Nein | Nein |
| FIR-Filter | ARIMA(4,0,0) | 4 | 4 | Nein | Nein |
| **EROS3** | **ARIMA(0,3,0)** | $\boldsymbol{k+3}$ | **0** | **Ja** | **Ja** |
| **EROS4** | **ARIMA(1,3,0)** | $\boldsymbol{k+4}$ | **(1)** | **Ja** | **Ja** |

Bezüglich der gestellten Anforderung lassen sich folgende Aussagen zu den untersuchten Kopplungsalgorithmen treffen:

- Die Minimierung des Kopplungsfehlers bei RVPen ist das Ziel aller diskutierten Kopplungsalgorithmen. Unterschiede hierbei werden bei der Anwendung der Kopplungsalgorithmen an Testfällen in Abschnitt 3.4 und 3.6, sowie einem industriell genutzten RVP in Abschnitt 6.2 sichtbar. Bis auf das modellbasierte Kopplungselement und den SMB können alle Algorithmen sowohl für die Kompensation als auch die Rekonstruktion eingesetzt werden.

- Da alle betrachteten Kopplungsalgorithmen linear sind, ist keiner in der Lage Diskontinuitäten in Koppelsignalen vorherzusagen. Die neuartigen EROS-Algorithmen werden in Abschnitt 3.5 um eine entsprechende Funktionalität erweitert. Statt Diskontinuitäten korrekt vorherzusagen, führen lineare Kopplungsalgorithmen an diesen Stellen stattdessen zu großen Kopplungsfehlern, da sie eine plötzliche Änderung in der Dynamik der Prototypen nicht abbilden können. Ein wichtiges Unterscheidungsmerkmal linearer Kopplungsalgorithmen ist daher die Zeitspanne nach einer Diskontinuität, in welcher die Vorhersage durch Signalwerte von vor der Diskontinuität signifikant beeinflusst wird. Dies hängt direkt von der Anzahl vergangener Signalwerte ab, die für eine Vorhersage verwendet werden. Bei ARIMA($p$,$d$,$q$) Modellen mit $q = 0$ werden für die Vorhersage eines Zeitschritts die Summe aus $p + d$ vergangene Signalwerte verwendet. ZOH, FOH, das FIR-

Filter sowie die EROS-Algorithmen nutzen somit eine bekannte Anzahl vergangener Signalwerte zur Berechnung der Vorhersage. Beim EROS-Algorithmus nimmt diese Anzahl zwar linear mit dem Vorhersagehorizont zu, liegt bei kleinen $k$ aber im Bereich der anderen Algorithmen. Bei ARIMA Modellen mit $q > 0$, wie dem SMB und dem modellbasierten Koppelelement, wird die aktuelle Vorhersage aus vergangenen Vorhersagen berechnet, wodurch alle $n$ vergangenen Signalwerte in die Vorhersage einfließen. Dadurch beeinflusst eine vergangene Diskontinuität die mit diesen Algorithmen berechneten Vorhersagen über einen langen Zeitraum signifikant und führt zu Schwingungen in der Vorhersage (s. Abschnitt 3.6). Ein weiterer Nachteil der Kopplungsalgorithmen mit $q > 0$ ist die lange Einschwingzeit der jeweiligen Algorithmen bei deren Initialisierung. Diese liegt in den Beispielanwendungen aus der Literatur bei beiden Algorithmen bei $> 30k$ und ist somit selbst mit $k = 1$ deutlich höher als die Zahl der verwendeten Signalwerte der anderen Methoden (Stettinger et al., 2014b, Liu et al., 2020).

- Nur das FIR-Filter und das modellbasierte Kopplungselement sind lernfähig und passen die Gewichte ihres jeweiligen ARIMA Modells zur Laufzeit an die Dynamiken der Koppelsignale an, um den Kopplungsfehler weiter zu minimieren. Allerdings ist der Lernvorgang bei beiden Algorithmen notwendig, bevor sinnvolle Vorhersagen getroffen werden können. Dies erschwert die Anwendung dieser beiden Algorithmen. Während des Lernvorgangs muss ein anderer Kopplungsalgorithmus aktiv sein, der den RVP stabilisiert. Weiterhin wird bei beiden Kopplungsalgorithmen der Recursive-Least-Squares Algorithmus zum Lernen verwendet, welcher vom Anwender parametriert werden muss.

- Die Anzahl der freien Parameter der Kopplungsalgorithmen ergibt sich aus der Summe $p + q$ des entsprechenden ARIMA Modells. Durch den $d$-Anteil eines ARIMA Modells werden keine Parameter hinzugefügt, da die Gewichte der vergangenen Signalwerte durch Bildung des $d$-ten Differenzenquotienten fixiert sind, wodurch ZOH, FOH und EROS3 keine Designparameter haben. Der Parameter, welcher den autoregressiven Teil vom EROS4 gewichtet ist durch das Identifikationsverfahren ebenfalls fixiert und muss nicht vom Anwender eingestellt werden. Der Parameter des SMBs muss vom Anwender gewählt werden, wofür aufgrund der Beschreibung von Zheng et al. (2018) kein Expertenwissen notwendig ist. Die Parameter beim FIR-Filter und dem modellbasierten Kopplungselement werden zur Laufzeit durch den Lernalgorithmus gewählt, dessen Parametrierung wiederum Expertenwissen erfordert.

- Der Berechnungsaufwand für die Vorhersagen ist bei allen Algorithmen gering. Am aufwändigsten ist die Berechnung beim FIR-Filter und dem modellbasierten Kopplungselement, da der geschätzte Signalverlauf nicht direkt von der Latenzzeit $k$ abhängt und daher die Vorhersage mehrerer Zeitschritte rekursiv berechnet werden muss.

- Alle Algorithmen sind für die Anwendung bei variablen Verzögerungszeiten geeignet. Entweder ist die Vorhersage direkt von der Latenzzeit $k$ abhängig oder die Vorhersage für einen Zeitschritt wird rekursiv $k$-mal durchgeführt.

Zusammengefasst erfüllen die entwickelten linearen Kopplungsalgorithmen EROS3 und EROS4 die bisher überprüften Anforderungen aus Abschnitt 3.1 genauso gut wie die aus der Co-Simulation abgeleiteten Algorithmen ZOH und FOH. Da für die Verwendung EROS-Algorithmen keine Parametrierung durch einen vorangehenden Lernvorgang notwendig ist, können sie leichter als die anderen untersuchten Algorithmen eingesetzt werden.

## 3.4 Anwendung auf lineare Real-Virtuelle Prototypen

Anhand des linearen Zweimassenschwingers aus Abschnitt 2.3.1 wird untersucht, welche der Kopplungsalgorithmen den Kopplungsfehler bei RVPen mit ausschließlich kontinuierlichen Signalen am besten minimieren. Neben den Methoden ZOH, FOH, EROS3 und EROS4, welche die definierten Anforderungen eines Kopplungsalgorithmus am besten erfüllen, wird auch der SMB in den Vergleich einbezogen, da er ebenfalls ohne vorangegangenes Lernen einsetzbar ist.

Die Makroschrittweite des simulierten RVPs wird auf $\Delta T = 0{,}02s$ und die Anzahl an Makroschritten Verzögerung auf $k = 3$ gesetzt, woraus eine Umlaufzeit von $t_{rtt} = 0{,}12s$ resultiert, welche im Bereich der maximal zu erwarteten Umlaufzeit bei einer Verteilung des RVPs innerhalb Europas entspricht (Schreiber et al., 2018). Zur Vermeidung von Unstetigkeiten in den Koppelsignalen zu Beginn der Simulation erfolgt die Anregung des Zweimassenschwingers aus der Ruhe, indem bei $t = 1s$ für $0{,}5s$ ein kontinuierlicher Kraftverlauf auf beide Massen aufgeprägt wird.

Abb. 3.1 zeigt für jedes der drei Koppelsignale den Verlauf der verzögerungsfreien Referenzsimulation und des Zweimassenschwingers inklusive Verzögerungen und mit ZOH als Kopplungsalgorithmus. Die erste Masse übergibt ihre Position $x_1$ und Geschwindigkeit $\dot{x}_1$, während die zweite Masse die Koppelkraft $f_k$ berechnet (s. Abschnitt 2.3.1).

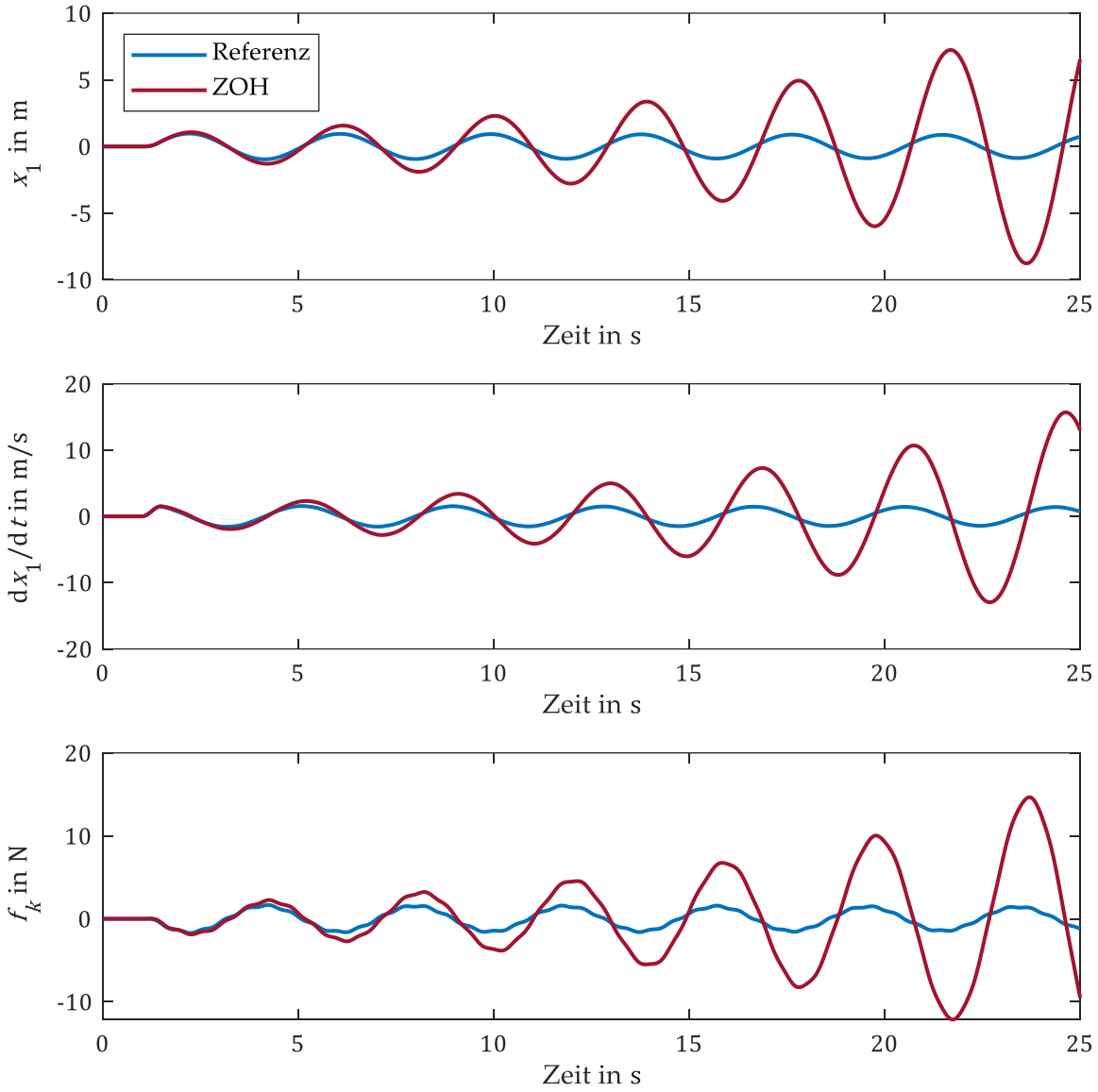


**Abb. 3.1** Linearer Zweimassenschwinger mit ZOH als Kopplungsalgorithmus

Der Standardalgorithmus ZOH kompensiert die Latenzzeit nicht, wodurch der Zweimassenschwinger instabil wird. Die vier anderen Algorithmen sind allesamt in der Lage den Zweimassenschwinger trotz Verzögerungen zu stabilisieren. Für den freien Designparameter des SMBs wird in dieser Arbeit $\lambda = 0{,}85\bar{\lambda}$ gewählt. Dieser liegt knapp unterhalb des Werts welchen Zheng et al. (2018) für eine bestmögliche Kompensation des Kopplungsfehlers empfehlen, da sich aufgrund der in dieser Arbeit beachteten Diskretisierung infolge des Datenaustauschs der Stabilitätsbereich des SMBs reduziert (s. Anhang).

Abb. 3.2 zeigt auf der linken Seite den Kraftverlauf, in dem die niederfrequente Schwingung der ersten Masse und die Schwingung der zweiten Masse mit höherer Frequenz überlagert sind. Im Ausschnitt auf der rechten Seite der Abbildung ist zu erkennen, dass der Signalverlauf sowohl beim EROS3 als auch beim EROS4 genauer der Referenzlösung entspricht als beim FOH oder dem SMB.

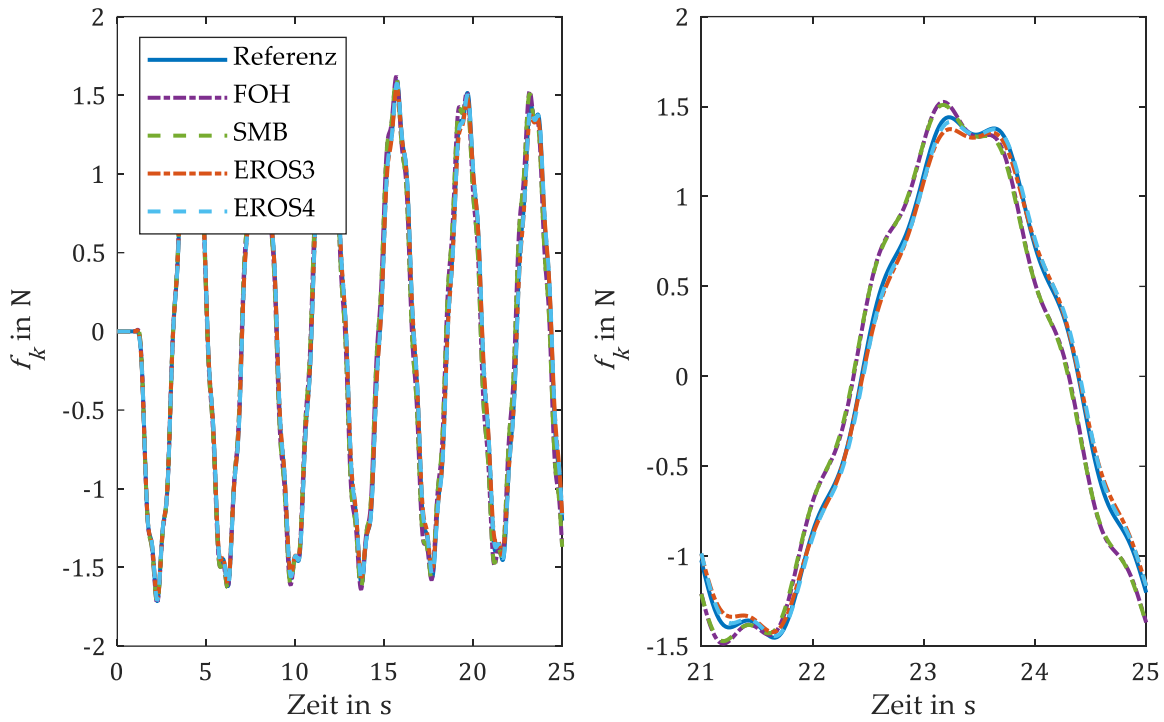


**Abb. 3.2** Vergleich verschiedener Kopplungsalgorithmen mit der Referenzlösung bei Anwendung auf den linearen Zweimassenschwinger. Links: Übersicht. Rechts: Ausschnitt.

Dieser visuelle Eindruck lässt sich durch einen Vergleich der Signalverläufe mittels der Metriken aus Abschnitt 2.4 bestätigen. Der in Tab. 3.2 dargestellte mittlere absolute Fehler ist bei Verwendung der EROS-Algorithmen bei jedem Koppelsignal bezüglich der Referenzlösung mindestens 75% kleiner als bei den anderen beiden stabilisierenden Kopplungsalgorithmen.

**Tab. 3.2** Mittlerer absoluter Fehler bezüglich der Referenzlösung des linearen Zweimassenschwingers für verschiedene Kopplungsalgorithmen

| | MAE - Summe | MAE - $f_k$ | MAE - $x_1$ | MAE - $\dot{x}_1$ |
|---|---|---|---|---|
| ZOH | 6,83 | 2,60 | 1,56 | 2,67 |
| FOH | 0,20 | 0,081 | 0,047 | 0,075 |
| SMB | 0,21 | 0,084 | 0,048 | 0,077 |
| **EROS3** | **0,050** | **0,024** | **0,0096** | **0,016** |
| **EROS4** | **0,038** | **0,017** | **0,0083** | **0,013** |

Die Quantifizierung des summierten Kopplungsfehlers aller Koppelsignale über die Metrik von Sprague und Geers (2004) (s. Abschnitt 2.4) zeigt weiterhin, dass die neuartigen Algorithmen EROS3 und EROS4 den Kopplungsfehler am besten minimieren. Aus Tab. 3.3 ist außerdem ersichtlich, dass der Kopplungsfehler beim ZOH und EROS3 vor allem auf eine Phasenverschiebung zurückzuführen ist, während FOH und SMB die Amplitude der Koppelsignale bei der Vorhersage überschätzen. EROS4 zeigt dagegen als einziger Algorithmus eine Balance zwischen Amplituden und Phasenfehler und somit den geringsten kombinierten Kopplungsfehler. In Kap. 4 wird im Detail auf die Wirkung der verschiedenen Kopplungsalgorithmen im Frequenzbereich eingegangen.

**Tab. 3.3** Summierter Kopplungsfehler mit Sprague und Geers ($\cdot 10^2$) als Distanzmaß für verschiedene Kopplungsalgorithmen

| | $C_{S\&G}$ | $M_{S\&G}$ | $P_{S\&G}$ |
|---|---|---|---|
| ZOH | 10,51 | 1,76 | 10,3 |
| FOH | 2,66 | 2,31 | 0,95 |
| SMB | 2,67 | 2,34 | 0,97 |
| **EROS3** | $\mathbf{0,81}$ | $\mathbf{0,20}$ | $\mathbf{0,73}$ |
| **EROS4** | $\mathbf{0,56}$ | $\mathbf{0,36}$ | $\mathbf{0,38}$ |

## 3.5 Vorwärts gerichtetes, künstliches neuronales Netz als nichtlinearer Kopplungsalgorithmus

Im Folgenden werden vorwärts gerichtete neuronale Netze (feedforward neural network, FFNN) eingesetzt, welche die linearen Kopplungsalgorithmen erweitern (Baumann et al., 2019a). Wie in diesem Abschnitt detailliert erläutert wird, bestehen FFNNs aus einer Kombination vieler Teilfunktionen, so genannter künstliche Neuronen, die jeweils nichtlinear sein können. Durch die Kombination dieser vielen nichtlinearen Funktionen können sie nichtlineares Verhalten abbilden und wie in Abschnitt 3.5 gezeigt wird auch an Unstetigkeitsstellen im Signalverlauf Latenzzeiten kompensieren. FFNN werden in vielen verschiedenen Bereichen zur Vorhersage von Zeitreihen verwendet und zählen zu den parametrischen Vorhersagemethoden (Cheng et al., 2015). Dies bedeutet, dass sie zukünftige Signalverläufe mithilfe einer nichtlinearen Funktion approximieren, deren Parameter in einem Training identifiziert werden müssen.

Ein vorwärts gerichtetes künstliches neuronales Netz (FFNN) setzt sich aus mehreren Schichten zusammen, die wiederum aus mehreren künstlichen Neuronen bestehen. Die Schichten aller in dieser Arbeit verwendeten FFNNs sind vollständig vernetzt. Die $h$-te Schicht berechnet mit Eingangsvektor $\boldsymbol{x}^{h-1}$ den Ausgangsvektor

$$\boldsymbol{x}^h = \sigma(\mathbf{W}^h \boldsymbol{x}^{h-1} + \boldsymbol{\Theta}^h) \tag{3-43}$$

mit einer im Allgemeinen nichtlinearen Aktivierungsfunktion $\sigma(\cdot)$ pro Neuron. Die Dimension der Gewichtungsmatrix $\boldsymbol{W}^h$ und des Offsetvektors $\boldsymbol{\Theta}^h$ hängen dabei von der Dimension des Eingangsvektors und der Anzahl der Neuronen ab, welche gleich der Dimension des Ausgangsvektors der $h$-ten Schicht ist. Der Eingangsvektors des neuronalen Netzes ist $\boldsymbol{x}^0$ und der Ausgang entsprechend $\boldsymbol{x}^h$.

Neben den Gewichtungsmatrizen und Offsetvektoren wird das Verhalten eines FFNN durch die Aktivierungsfunktion in den Neuronen beeinflusst. Damit sich ein FFNN nichtlinear verhalten kann, muss mindestens in einem Neuron statt der linearen Aktivierung $\sigma_{lin}(x) = x$ eine nichtlineare Aktivierungsfunktion gewählt werden. In dieser Arbeit werden Leaky rectified linear units (LReLU) mit

$$\sigma_{lrelu}(x) = \begin{cases} x \text{ für } x \geq 0 \\ \alpha x \text{ für } x < 0 \end{cases} \tag{3-44}$$

und $\alpha = 0{,}01$ verwendet (Maas, Hannun und Ng, 2018), welche eine Variante der häufig genutzten Aktivierungsfunktion Rectified linear units (ReLU) sind (Nannapaneni und Kulkarni, 2018). Bei LReLU ist der Gradient nicht aktiver Neuronen ($x < 0$) im Gegensatz zu ReLU von Null verschieden, weshalb deren Parameter während des Lernvorgangs auch im inaktiven Zustand veränderbar sind. Dies ist bei kleinen Netzen wichtig, da dadurch eine dauerhafte Inaktivität einer signifikanten Anzahl an Neuronen vermieden wird.

Für die Vorhersage univariater Zeitreihen mit FFNN besteht der Eingangsvektor $\boldsymbol{x}^0$ üblicherweise aus den letzten $p$ vergangenen Werten eines Signals $\boldsymbol{y}$**.** Der Ausgang des Netzes ist skalar und bildet die Vorhersage $\hat{y}$ eines zukünftigen Zeitschritts. Bereits Dorffner (1996) stellte fest, dass sich ein nach Gl. (3-43) gebildetes und zur Vorhersage univariater Zeitreihen verwendetes FFNN zu

$$\hat{y}_n = c + \sum_{i=1}^{p} a_i y_{n-i} \tag{3-45}$$

vereinfachen lässt, wenn in jedem Neuron jeder Schicht die lineare Aktivierungsfunktion $\sigma_{lin}(x) = x$ verwendet wird. Weiterhin kann ein beliebig großes FFNN, welches sich ausschließlich aus Neuronen mit linearer Aktivierungsfunktion zusammensetzt, in ein FFNN bestehend aus einem einzigen Neuron mit linearer Aktivierungsfunktion überführt werden, ohne dessen Eingangs-Ausgangs-Verhalten zu ändern. Die Gewichte $a_i$ sowie die Konstante $c$ entsprechen dabei genau der Gewichte und dem Offset des einzelnen Neurons. Gl. (3-45) entspricht außerdem exakt der Definition eines ARMA($p$,0) Modells (vgl. Gl. (3-20)), woraus folgt, dass diese als FFNN mit einem einzelnen Neuron darstellbar sind (Dorffner, 1996). Weiterhin gilt dies auch für ARIMA($p$,$d$,0) Modelle, da sie ein Spezialfall der ARMA($p + d$,$q$) Modelle sind (Tsay und Tiao, 1984). Dabei werden die durch Bildung des $d$-ten Differenzenquotienten fixierten Parameter des ARIMA Modells als freie Parameter des autoregressiven Teils

interpretiert und das ARMA Modell verliert die Eigenschaft der Stationärität. Nach Tab. 3.1 in Abschnitt 3.3.3 können somit die Kopplungsalgorithmen ZOH, FOH, das FIR-Filter sowie die neuartigen Algorithmen EROS3 und EROS4 in ein FFNN mit einem Neuron und linearer Aktivierungsfunktion überführt werden.

Durch die Beschreibung der Kopplungsalgorithmen als FFNN, ist deren Anpassung auf die jeweiligen Koppelsignale der RVPen möglich, indem auf die sehr effizienten und auf die Struktur von FFNNs angepassten Trainingsverfahren zurückgegriffen wird (Kingma und Ba, 2014).

Es sei angemerkt, dass ARIMA Modelle mit gleitendem Mittelwert Anteil ($q > 0$) nicht durch ein FFNN darstellbar sind, da von diesen Modellen alle vergangenen Signalwerte berücksichtigt werden und somit der Eingangsvektor in jedem neuen Zeitschritt größer werden würde. Dies ist stattdessen mit rekurrenten neuronalen Netzen (RNN), beispielsweise mit Long short-term memory (LSTM) Architekturen, möglich, welche ihre vergangenen Vorhersagen für zukünftige Vorhersagen nutzen (Schmidhuber, 2015). Somit ist die hier vorgestellte Verallgemeinerung nicht für das modellbasierte Kopplungselement und den SMB von Tandon et al. (2013) gültig.

### 3.5.1 Generierung der Architektur der FFNN

Ausgehend von einem FFNN zur Zeitreihenvorhersage mit einem einzelnen Neuron, dessen Eingangs-Ausgangs-Verhalten dem eines Kopplungsalgorithmus der Form ARIMA($p$,$d$,0) entspricht, soll die Anzahl an Neuronen des FFNN erhöht werden, ohne dessen Eingangs-Ausgangs-Verhalten zu verändern.

Diese Erweiterung des FFNN erhöht die Anzahl der Parameter, welches bei gleichzeitiger Hinzunahme von Neuronen mit LRELU Aktivierungsfunktion dazu führt, dass das FFNN komplexe nichtlineare Zusammenhänge abbilden kann. Weiterhin ist dadurch, dass das Eingangs-Ausgangs-Verhalten des initialen FFNN dem eines linearen Kopplungsalgorithmus entspricht, die Vorhersage kontinuierlicher Signalverläufe mithilfe des FFNN bereits ohne zusätzliches Training sehr gut (vgl. Abschnitt 3.4). Zum einen verringert dies die benötigte Trainingszeit sowie die Menge an Trainingsdaten, die zur weiteren Anpassung an die Koppelsignale benötigt werden, wodurch dies online mit beschränkter Trainingszeit möglich ist. Zum anderen ist das FFNN dadurch nicht auf eine Lernphase angewiesen, sondern ist bereits mit einer generischen Startparametrierung ausgestattet und von Beginn an in der Lage sinnvolle Vorhersagen für den zukünftigen Signalverlauf treffen. Diese in Abschnitt 3.1 gestellte Anforderung an die Lernfähigkeit eines Kopplungsalgorithmus erfüllt bislang kein bekannter Algorithmus.

Techniken, um aus einem bestehenden Netzwerk $f(\cdot)$ ein größeres und mächtigeres Netzwerk $g(\cdot)$ mit den Parametern $\boldsymbol{W}'$,$\boldsymbol{\Theta}'$ zu erstellen, sodass

$$f(\boldsymbol{x}, \boldsymbol{W}, \boldsymbol{\Theta}) = g(x, \boldsymbol{W}', \boldsymbol{\Theta}'), \forall \boldsymbol{x} \tag{3-46}$$

gilt, werden Network Morphism genannt (Wei et al., 2016).

In dieser Arbeit wird $g(\cdot)$ durch Anwendung der NET2NET Methoden von Chen, Goodfellow und Shlens (2015) erstellt. Dies erfolgt in drei Schritten:

1. **Einfügen weiterer Schichten mittels NET2DEEPERNET.** Dazu wird eine Schicht durch zwei Schichten ersetzt, sodass

   $$\boldsymbol{x}^h = \sigma\left(\mathbf{W}^h \boldsymbol{x}^{h-1} + \boldsymbol{\Theta}^h\right) \stackrel{!}{=} \sigma\left(\boldsymbol{A}^h \sigma\left(\mathbf{W}^h \boldsymbol{x}^{h-1} + \boldsymbol{\Theta}^h\right) + \boldsymbol{b}^h\right) \tag{3-47}$$

   gilt. Diese Gleichung ist nach Initialisierung der Gewichtungsmatrix $\boldsymbol{A}^h$ als Identitätsmatrix $\boldsymbol{I}$ und mit dem Offsetvektors $\boldsymbol{b}^h = 0$ erfüllt. Beide Parameter können bei späterem Training des Netzes frei angepasst werden. Weiterhin muss die Aktivierungsfunktion der ersetzten Schicht idempotent sein, sodass

   $$\sigma(\boldsymbol{I}\sigma(\boldsymbol{x})) = \sigma(\boldsymbol{x}), \forall \boldsymbol{x} \tag{3-48}$$

   gilt. Diese Eigenschaft ist für die lineare Aktivierungsfunktion mit $\sigma_{lin}(x) = x$ erfüllt (Chen, Goodfellow und Shlens, 2015).

2. **Einfügen weiterer Neuronen in den Schichten mittels NET2WIDERNET**, durch Replikation einzelner Neuronen innerhalb derselben Schicht (nicht der Ausgangsschicht). Aktivierungsfunktion, Offset und Gewichte am Eingang und am Ausgang der hinzugefügten Neuronen entsprechen dabei denen des replizierten Neurons. Damit weiterhin alle Neuronen des Netzwerks dieselben Eingangswerte erhalten und das Eingangs-Ausgangs-Verhalten des Netzwerks unverändert bleibt, müssen anschließend die Parameter der anschließenden Schicht angepasst werden. Alle Parameter, welche zuvor den Ausgang des replizierten Neurons gewichteten, gewichten nun, aufgrund der vollständigen Vernetzung der Schichten, zusätzlich die Ausgänge der hinzugefügten Neuronen. Zur Sicherstellung eines unveränderten Eingangs-Ausgangs-Verhaltens des Netzwerks müssen sie deshalb durch $1 + N$ geteilt werden, wobei $N$ die Anzahl der hinzugefügten Neuronen ist. Abschließend müssen alle Parameter der hinzugefügten Neuronen mit einem kleinem Rauschen addiert werden, um die entstandene Symmetrie im Netzwerk aufzubrechen, da sie ansonsten beim Lernen genau dieselben Änderungen erfahren würden und sich damit kein Vorteil durch das hinzufügen weiterer Neuronen ergäbe (Chen, Goodfellow und Shlens, 2015).

3. **Einfügen der LRELU Aktivierungsfunktion.** In dieser Arbeit werden LRELUs in ein bestehendes Netzwerk eingefügt, indem ausgehend von

$$\sigma_{lin}(x) = \frac{1}{1+\alpha}\left(\sigma_{lrelu}(x) + \sigma_{lrelu}(-x)\right) \tag{3-49}$$

einzelne Neuronen mit linearer Aktivierungsfunktion durch zwei Neuronen mit LRELU Aktivierungsfunktion ersetzt werden. Anschließend müssen zur Erfüllung von Gl. (3-49) alle Eingangs- und Ausgangsgewichte sowie der Offset eines der beiden Neuronen mit $-1$ multipliziert werden. Zusätzlich ist eine Multiplikation der Ausgangsgewichte beider Neuronen mit $\frac{1}{1+\alpha}$ notwendig.

Für den Einsatz eines FFNN als Kopplungsalgorithmus bei RVPen ist zur Reduktion des Berechnungsaufwands der Vorhersagen, sowie der Trainingszeit, die Anzahl der Neuronen so gering wie möglich zu halten. Damit die Netzwerke trotzdem in der Lage sind komplexe, nichtlineare Dynamiken abzubilden, ist die Architektur der FFNN in dieser Arbeit wie in Abb. 3.3 dargestellt gewählt. Die erste Schicht enthält vier Neuronen, welche jeweils vollständig mit dem Eingangsvektor $\boldsymbol{x}^0$ vernetzt sind und die lineare Aktivierungsfunktion enthalten. Daran schließt eine weitere Schicht mit zwei Neuronen und LRELU Aktivierungsfunktion an, bevor als Ausgang einer weiteren Schicht mit einem Neuron und linearer Aktivierungsfunktion die skalare Vorhersage der Zeitreihe berechnet wird. Die Dimension von $\boldsymbol{x}^0$ entspricht der Anzahl der berücksichtigten vergangenen Werte $m$ der vorhergesagten Zeitreihe und hängt somit vom linearen Kopplungsalgorithmus ab, welchen das FFNN erweitert. Ausgehend vom FFNN mit einem Neuron muss daher zunächst zweimal die NET2DEEPERNET Methode ausgeführt werden. Mithilfe von NET2WIDERNET wird das Neuron der ersten Schicht anschließend vervierfacht und abschließend das einzelne Neuron der zweiten Schicht durch zwei Neuronen mit LRELU Aktivierungsfunktion ersetzt.

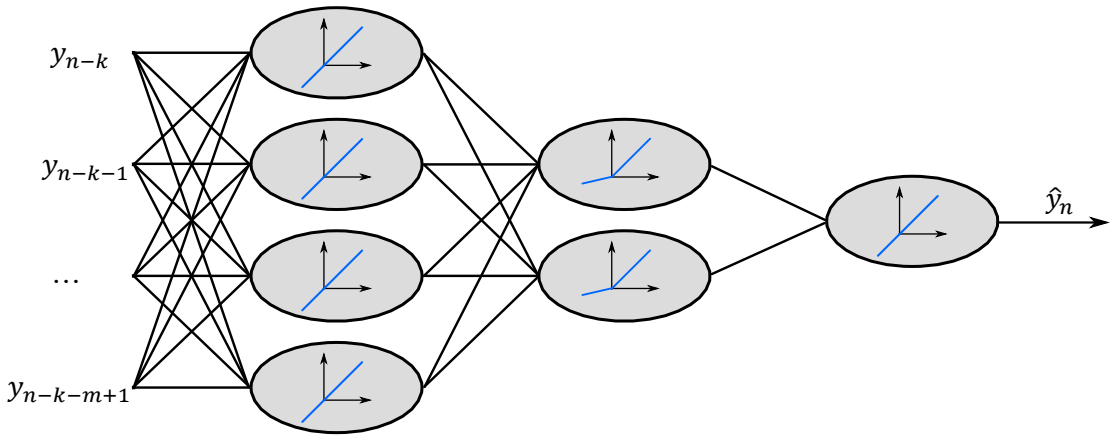


**Abb. 3.3** Architektur der in dieser Arbeit verwendeten FFNN

Die Wahl der optimalen Architektur von FFNNs ist seit vielen Jahren Gegenstand der Forschung (Protopapadakis, Voulodimos und Doulamis, 2017). Die in dieser Arbeit verwendete

Architektur ist nur eine mögliche. Angesichts der Vielzahl der Parameter, die die Architektur bestimmen, ist es wahrscheinlich, dass FFNNs mit anderen Schichtzusammensetzungen, zum Beispiel mit mehr Neuronen oder anderen Aktivierungsfunktionen, besser für den Einsatz als Kopplungsalgorithmus von RVPen geeignet sind als das hier gewählte FFNN.

### 3.5.2 Online Lernen der FFNN

Während der Laufzeit von RVPen sollen die FFNN, welche als ARIMA($p$,$d$,0) initialisiert und deren Architektur dem Vorgehen des vorherigen Abschnitts folgend erweitert wurde, online auf den Signalverlauf der Koppelsignale angepasst werden. Dies ermöglicht es dem FFNN-Kopplungsalgorithmus unstetige Signalverläufe vorherzusagen und dadurch den Kopplungsfehler auch bei nichtlinearen Signalverläufen weiter zu minimieren.

Überwachte Lernverfahren trainieren FFNN mithilfe von Trainingsdaten. Die Verfahren basieren meist auf der Berechnung des Gradienten der FFNN, welcher trotz der hohen Komplexität des Eingangs-Ausgangs-Verhaltens aufgrund der Struktur der FFNN effizient berechenbar ist. In dieser Arbeit wird für das Training der FFNN der gradientenbasierte Optimierer Adam, welcher die Berechnungsressourcen besonders effizient nutzt (Kingma und Ba, 2014). Adam adaptiert in jeder Iteration die Lernrate jedes Parameters des FFNN unter Berücksichtig der ersten beiden Momente (Erwartungswert und Varianz) der Gradienten.

In Abb. 3.4 ist der Ablauf der online Anpassung dargestellt. Da RVPen und damit auch der Kopplungsalgorithmus der Echtzeitanforderung unterliegen, findet die online Anpassung der Parameter des FFNN unabhängig von der Vorhersage des Signalverlaufs statt. Dazu werden pro Koppelsignal während der Initialisierung des RVPs zwei identische FFNN gemäß Abschnitt 3.5.1 generiert. Eines bildet den eigentlichen Kopplungsalgorithmus und berechnet die Vorhersage des Koppelsignals. Dies erfolgt periodisch in jedem Makroschritt und ist in der

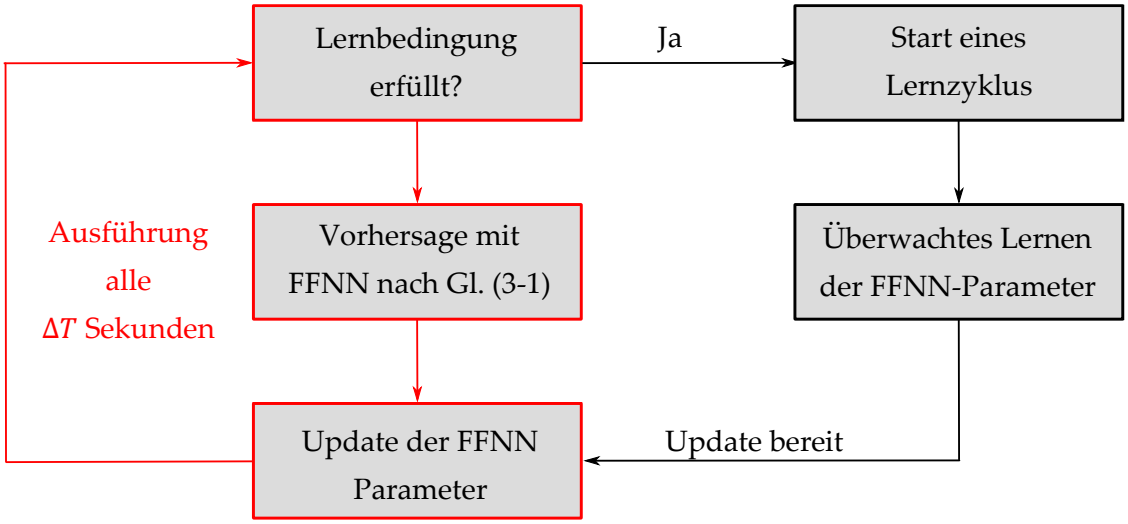


**Abb. 3.4** Ablauf des online Lernens des FFNN-Kopplungsalgorithmus

Abb. 3.4 durch die roten Blöcke markiert. Ist die Lernbedingung erfüllt, startet ein neuer Lernzyklus, bei dem das zweite FFNN trainiert wird. Als Trainingsdaten dienen dazu die Signalwerte des Koppelsignals, die seit dem Start des vorherigen Lernzyklus entstanden sind. Die Lernbedingung ist in dieser Arbeit nach Ablauf einer festen Zeitspanne erfüllt, die vom Anwender festgelegt werden muss. In zukünftigen Arbeiten könnte die online Berechnung des Kopplungsfehlers (s. Abschnitt 5.2) zur Definition der Lernbedingung genutzt werden. Das Training wird kurz vor Beginn des nächsten Lernzyklus abgebrochen und das FFNN, welches die Vorhersagen berechnet, mit den angepassten Parameter des zweiten FFNN aktualisiert. Bei der industriellen Anwendung des Kopplungsalgorithmus in Abschnitt 6.1 wird genauer auf die Implementierung des online Lernens eingegangen.

Das in Abschnitt 3.5 entwickelte Lernverfahren erlaubt es dagegen zur Laufzeit Kopplungsalgorithmen, die auf beliebigen ARIMA($p$,$d$,0) Modellen beruhen und bereits parametriert sind, auf die Koppelsignale anzupassen und bildet somit eine Erweiterung für die EROS-Algorithmen.

## 3.6 Anwendung auf nichtlineare Real-Virtuelle Prototypen

Anhand des nichtlinearen Zweimassenschwingers mit mechanischem Anschlag aus Abschnitt 2.3.2 wird die Lernfähigkeit des neuartigen FFNN-Kopplungsalgorithmus demonstriert. Weiterhin wird gezeigt, dass mit diesem Algorithmus Unstetigkeiten in Signalverläufen von RVPen vorhersagbar sind.

Wie bei den Untersuchungen am linearen Zweimassenschwinger in Abschnitt 3.4 wird die Makroschrittweite des simulierten RVPs auf $\Delta T = 0{,}02\text{s}$ und die Anzahl an Makroschritten Verzögerung in beide Austauschrichtungen auf $k = 3$ gesetzt. Die Anregung des Zweimassenschwingers erfolgt wieder aus der Ruhe durch Aufprägung einer Kraft auf beide Massen. Der mechanische Anschlag erlaubt eine maximale Auslenkung um $x_{1,stopp} = -1m$ in $x_1$-Richtung und für die Stoßzahl gilt $r_{stopp} = 1$, wodurch der Anschlag elastisch modelliert ist. Die Generierung des untersuchten FFNN-Kopplungsalgorithmus erfolgt mittels des in Abschnitt 3.5.1 dargelegten Verfahrens, wobei dessen Startparametrierung der des linearen EROS3 Kopplungsalgorithmus entspricht.

Abb. 3.5 zeigt den von verschiedenen Kopplungsalgorithmen geschätzten Verlauf der Geschwindigkeit der ersten Masse bei einer Ausführung des RVPs. Neben dem lernfähigen FFNN-Kopplungsalgorithmus werden die linearen Algorithmen FOH und der SMB mit der Referenzlösung verglichen. ZOH ist hier nicht gezeigt, da dieser Algorithmus wie bereits beim linearen Zweimassenschwinger das System nicht stabilisieren kann (vgl. Abb. 3.1). Die Diskontinuitäten infolge der Stöße am mechanischen Anschlag sind im Signalverlauf deutlich zu erkennen. Im kontinuierlichen Signalverlauf vor dem ersten Stoß zeigt das FFNN, obwohl es

noch nicht trainiert wurde, eine sehr gute Übereinstimmung mit der Referenz, da es entsprechend des Eingangs-Ausgangs-Verhaltens des EROS3 Algorithmus initialisiert wurde. An der Stelle des ersten Stoßes überschätzen dagegen alle Kopplungsalgorithmen die Höhe des „Sprungs“ im Signalverlauf deutlich. Beim zweiten Anschlag ist die Übereinstimmung des FFNN mit der Referenz dagegen bereits wesentlich besser (gelbes Signal in Abb. 3.5 schwingt bei zweitem Anschlag bei $t{\sim}7s$ kaum über), da das FFNN zwischen den Anschlägen online angepasst wurde und nun den nichtlinearen Signalverlauf nachbilden kann (s. Abschnitt 3.5.2). Die Dauer eines Lernzyklus ist hier auf zwei Sekunden begrenzt, wodurch der erste Zyklus zwischen den ersten beiden Anschlägen beginnt und auch wieder endet.

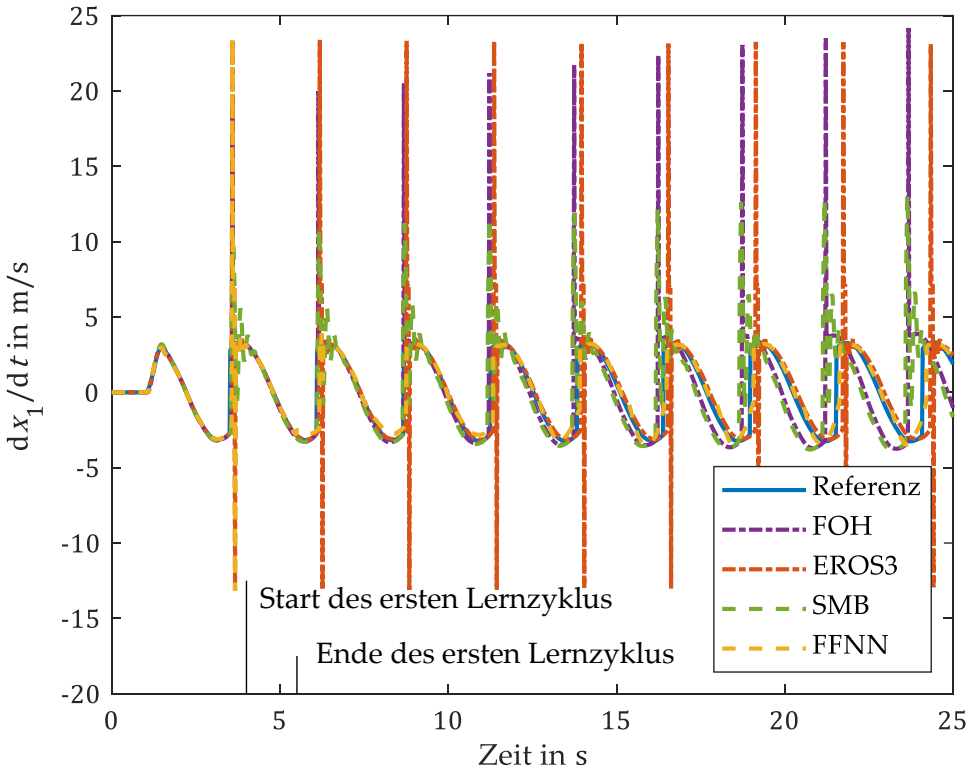


**Abb. 3.5** Vergleich der geschätzten Geschwindigkeit der ersten Masse mit verschiedenen Kopplungsalgorithmen

Der große Kopplungsfehler der linearen Kopplungsalgorithmen an den Unstetigkeitsstellen wirkt sich signifikant auf das Systemverhalten aus. In Abb. 3.6 ist der Positionsverlauf der ersten Masse unter Verwendung der verschiedenen Kopplungsalgorithmen aufgezeigt, welcher am Ausgang des entsprechenden Prototyps gemessen wurde. In der Vergrößerung auf der rechten Seite ist zu sehen, dass die Position bei einer Latenzzeitkompensation mit den beiden linearen Algorithmen bereits deutlich von der Referenz abweicht, während sie bei Verwendung des FFNN-Kopplungsalgorithmus mit der Referenzlösung besser übereinstimmt. Ein Vergleich der Hochpunkte auf der linken Seite der Abbildung zeigt weiterhin ein langsames Aufschwingen, wenn die linearen Kopplungsalgorithmen verwendet werden. Dies deutet darauf hin, dass diese den nichtlinearen Zweimassenschwinger unter den gegebenen Netzwerkeffekten nicht stabilisieren.

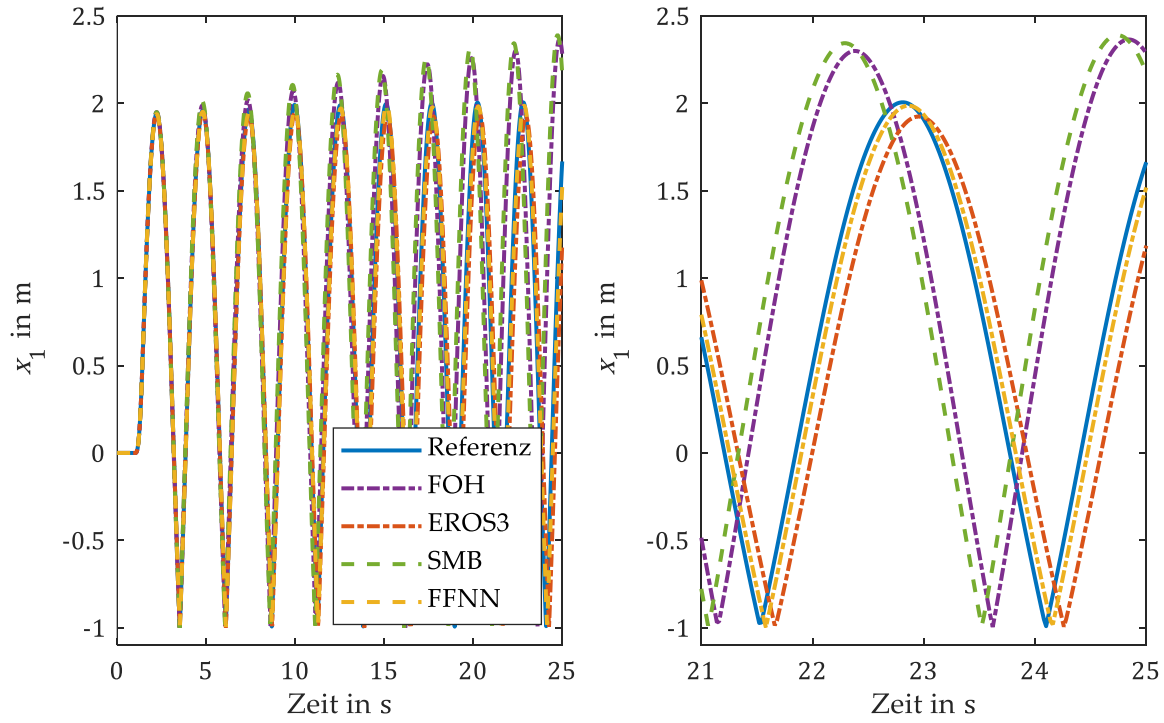


**Abb. 3.6** Vergleich der Position der ersten Masse am Ausgang des Prototyps unter Verwendung verschiedener Kopplungsalgorithmen. Links: Übersicht, Rechts: Ausschnitt

Zur genaueren Untersuchung der Vorhersage der verglichenen Kopplungsalgorithmen an den Unstetigkeitsstellen legen die Signalausschnitte, welche in Abb. 3.7 gezeigt sind, den Fokus auf den Zeitpunkt eines Anschlags. Da der Kontakt mit dem Anschlag aufgrund der geschlossenen Kopplung der Prototypen bei Verwendung verschiedener Kopplungsalgorithmen nicht mehr gleichzeitig stattfindet, sind die Verläufe zur Verbesserung der Vergleichbarkeit entlang der x-Achse verschoben. Die Auswirkungen des Stoßes sind somit bei allen verglichenen Simulationen im ersten Makroschritt nach dessen Auftreten im Verlauf der Ausgangssignale des ersten Prototyps erkennbar und erreichen den Eingang des Kopplungsalgorithmus $k = 3$ Makroschritte später.

Im oberen Teil der Abb. 3.7 ist die Position der ersten Masse dargestellt, welche infolge des Anschlags keine Werte kleiner $x_{1,stopp} = -1m$ annimmt. Sobald $x_{1,stopp}$ erreicht ist, knickt der Signalverlauf aufgrund des Vorzeichenwechsels in dessen Steigung ab. Der Zeitpunkt dieser Diskontinuität folgt somit einem festen Muster und ist aus dem vergangenen Signalverlauf vorhersagbar. Aufgrund seiner online Lernfähigkeit und seines nichtlinearen Eingangs-Ausgangs-Verhaltens ist der FFNN-Kopplungsalgorithmus in der Lage dieses Muster zu lernen und entsprechend zu reagieren. Bereits vier Makroschritte vor der Diskontinuität passt es die Steigung des vorhergesagten Signalverlaufs an, sodass keine Position kleiner $x_{1,stopp}$ vorhergesagt wird. Am fünften Makroschritt, sobald zwei Signalwerte nach der Diskontinuität am Eingang des FFNN vorliegen, kann das FFNN den stetigen Signalverlauf vorhersagen und die Vorhersage und Referenz stimmen genau wie vor der Diskontinuität wieder sehr gut überein.

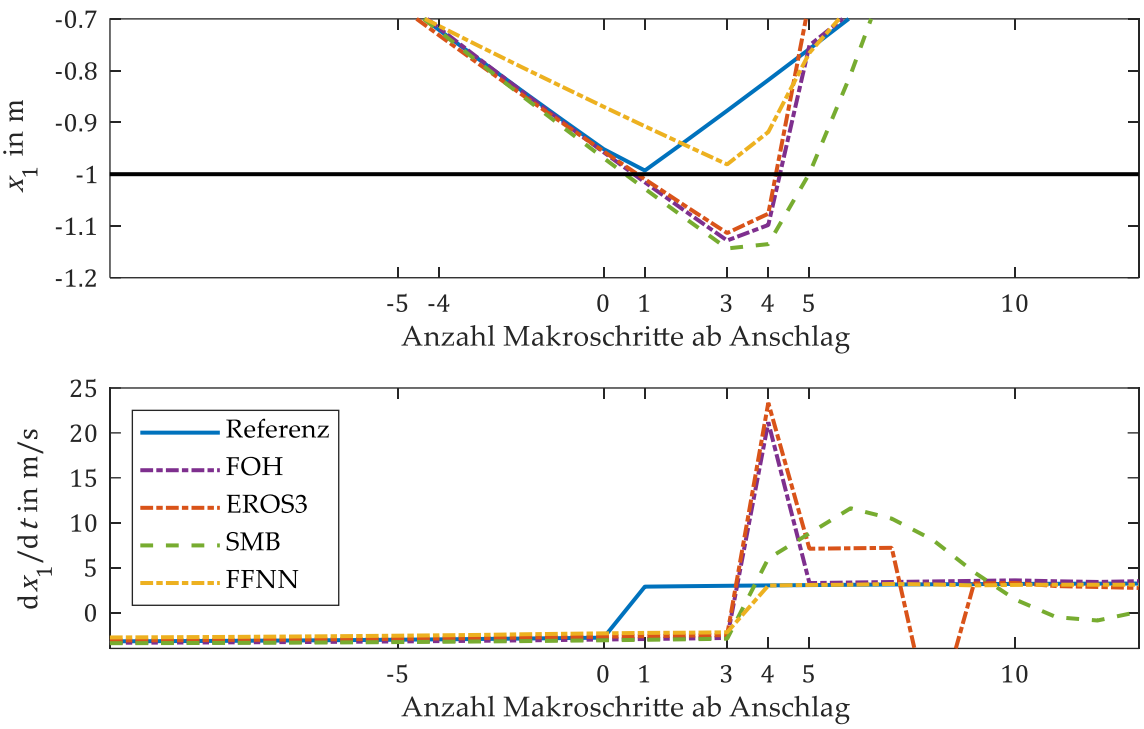


**Abb. 3.7** Fokus auf Zeitpunkt des Anschlags am Eingang des zweiten Prototyps. Oben: Position der ersten Masse. Unten: Geschwindigkeit der ersten Masse.

Im unteren Teil der Abb. 3.7 ist der Verlauf der Geschwindigkeit in der Umgebung der Diskontinuität dargestellt. Im Gegensatz zum Verlauf der Position folgen die Zeitpunkte der Diskontinuität in der Geschwindigkeit keinem lernbaren Muster, weshalb der FFNN-Kopplungsalgorithmus diese nicht vorhersagen kann (s. Abweichung zwischen dem Verlauf der Referenz und des FFNN zwischen dem 0. und 3. Makroschritt). Allerdings hat es gelernt den Signalverlauf nach einer Diskontinuität mit hoher Genauigkeit vorherzusagen, sobald diese nach Ablauf der Verzögerung im vierten Makroschritt nach dem Anschlag am Eingang des Kopplungsalgorithmus anliegt.

Der Verlauf der Geschwindigkeit zeigt weiterhin den Nachteil linearer Kopplungsalgorithmen, welche den gleitenden Durchschnitt über vergangene Kopplungsfehler berücksichtigen (ARIMA($p$,$d$,q), $q > 0$) gegenüber Kopplungsalgorithmen, deren Vorhersage ausschließlich auf der Gewichtung vergangener Signalwerte beruht. Während beim FOH die Vorhersage auf nur zwei vergangenen Signalwerten basiert und somit nach einer Diskontinuität nur in einem Makroschritt stark von der Referenz abweicht, führt die Berücksichtigung vergangener Kopplungsfehler beim SMB über einen langen Zeitraum nach einer Diskontinuität zu größeren Kopplungsfehlern.

Die verbesserte Vorhersage des FFNN-Kopplungsalgorithmus aufgrund dessen Lernfähigkeit und des nichtlinearen Eingangs-Ausgangs-Verhaltens führt im Vergleich mit den linearen Al-

gorithmen zu einer quantifizierbar besseren Übereinstimmung der Verläufe der Koppelsignale bezüglich der Referenzlösung. Tab. 3.4 zeigt den summierten mittleren absoluten Fehler (vgl. 2.4) bezüglich der Referenzlösung für die verglichenen Kopplungsalgorithmen. Im Verlauf der Kopplungskraft ist die Abweichung zur Referenzlösung um 77%, bei der Position 85% und bei der Geschwindigkeit um 64% verringert.

**Tab. 3.4** Summierter mittlerer absoluter Kopplungsfehler bezüglich der Referenzlösung des nichtlinearen Zweimassenschwingers für verschiedene Kopplungsalgorithmen

| | MAE - Summe | MAE - $f_k$ | MAE - $x_1$ | MAE - $\dot{x}_1$ |
|---|---|---|---|---|
| ZOH | 16,57 | 5,81 | 3,55 | 7,21 |
| FOH | 1,94 | 0,65 | 0,38 | 0,92 |
| EROS3 | 1,14 | 0,32 | 0,13 | 0,69 |
| SMB | 2,58 | 0,85 | 0,47 | 1,26 |
| **FFNN** | $\mathbf{0,53}$ | $\mathbf{0,15}$ | $\mathbf{0,056}$ | $\mathbf{0,33}$ |

# 4 Analyse räumlich verteilter Real-Virtueller Prototypen

Netzwerkeffekte wie Latenzzeiten führen zu Kopplungsfehlern und beeinflussen das Systemverhalten von RVPen, weshalb Kopplungsalgorithmen zur Minimierung dieser Fehler eingesetzt werden. Da diese Algorithmen wiederum Einfluss auf das Systemverhalten von RVPen nehmen, wird im Folgenden das Systemverhalten von RVPen inklusive Netzwerkeffekten und Kopplungsalgorithmen untersucht und daraus Gültigkeitsbereiche für die generischen Kopplungsalgorithmen aus Abschnitt 3.3 abgeleitet. Weiterhin wird eine Methodik vorgeschlagen, um basierend auf den Eigenschaften der Kopplung und dem Systemverhalten von RVPen einen optimalen Kopplungsalgorithmus zur Minimierung des Kopplungsfehlers zu berechnen.

Bei einer Co-Simulation erfolgen Analysen des Systemverhaltens über Aussagen zur Stabilität des gekoppelten Systems und deren Robustheit, indem zunächst die Nullstabilität festgestellt wird und anschließend die Gebiete der numerischen Stabilität in Abhängigkeit verschiedener Konfigurationsparameter berechnet werden. Ein allgemeingültiges Vorgehen zur Wahl der Konfiguration der Co-Simulation zur Sicherstellung der Stabilität existiert bislang nicht (vgl. Abschnitt 2.1.1). Belastbare Aussagen zur numerischen Stabilität von RVPen zu treffen ist zusätzlich erschwert. Das Systemverhalten realer Prototypen sowie der dazu gehörenden Sensorik und Aktuatorik ist nicht exakt mathematisch beschreibbar und die auf Wahrscheinlichkeiten beruhenden Netzwerkeffekte erhöhen die Anzahl der Einflussfaktoren auf die numerische Stabilität und führen unter Umständen zu nicht deterministischem Verhalten.

Um trotzdem Aussagen über den Einfluss der Netzwerkeffekte und der Kopplungsalgorithmen auf das Systemverhalten von RVPen zu erhalten, wird in dieser Arbeit ein ähnlicher Ansatz verfolgt, wie Benedikt, Watzenig und Hofer (2013) für die Analyse der nicht-iterativen Co-Simulation anwenden. Da der Signalaustausch die Ursache des Kopplungsfehlers ist, analysieren sie den Kopplungsprozess der Co-Simulation, indem sie die Co-Simulation im Frequenzbereich modellieren, anstatt das gesamte gekoppelte System zu betrachten.

In Abschnitt 4.1 wird zunächst der Kopplungsprozess für RVPen inklusive Abtastung, Netzwerkeffekte und Kopplungsalgorithmus im Frequenzbereich modelliert und damit das Systemverhalten der neuartigen EROS-Kopplungsalgorithmen mit dem der etablierten Algorithmen verglichen. Anschließend wird das Modell des Kopplungsprozesses verwendet, um über das Nyquist Kriterium Aussagen über die Stabilität von RVPen unter Verwendung verschiedener Kopplungsalgorithmen zu treffen. Dieses Modell des Kopplungsprozesses bildet die

Basis um in Abschnitt 4.2 ein Optimierungsproblem zur Auslegung von optimalen Kopplungsalgorithmen aufzustellen und dessen Anwendung zu diskutieren. Dazu wird zunächst eine Kostenfunktion definiert, welche die Eigenschaften der Kopplung und Eigenschaften der gekoppelten Systeme mit einbezieht, aber gleichzeitig kein detailliertes Systemwissen der gekoppelten Systeme voraussetzt. Anschließend wird ein optimierter Kopplungsalgorithmus für den Zweimassenschwinger aus Abschnitt 2.3.1 berechnet, indem das Optimierungsproblem gelöst wird. Es folgt ein Vergleich der optimierten mit den generischen Kopplungsalgorithmen. In Abschnitt 4.3 werden die Einsatzmöglichkeiten und Einschränkungen der entwickelten Analysemethodik sowie der darauf aufbauenden optimalen Auslegung von Kopplungsalgorithmen diskutiert.

## 4.1 Systemanalyse linearer Real-Virtueller Prototypen im Frequenzbereich

In RVPen sind zeitkontinuierliche Prototypen über zeitdiskrete Signale gekoppelt, weshalb die Systemanalyse im Frequenzbereich sowohl mit diskreten als auch kontinuierlichen Methoden möglich ist. Bei beiden Varianten, welche in Abb. 4.1 dargestellt sind, wird jeweils eine Übertragungsfunktion für jeden Prototypen, sowie eine Übertragungsfunktion für den Kopplungsprozess zwischen zwei Prototypen bestimmt. Die Impulsübertragungsfunktionen der diskreten Analyse mittels der z-Transformation werden zwischen diskreten Signalen aufgestellt. Daher muss für deren Berechnung die Rekonstruktion der kontinuierlichen Eingangssignale aus den zeitdiskreten Koppelsignalen und die Abtastung der Ausgangssignale mit einbezogen werden. Die Impulsübertragungsfunktion des Signalaustauschs besteht in diesem Fall aus Latenzzeiten und deren Kompensation. Bei der kontinuierlichen Systemanalyse im Frequenzbereich unter Verwendung der Laplace-Transformation wird die Übertragungsfunktion der gekoppelten Prototypen direkt bezüglich deren zeitkontinuierlichen Eingangs- und Ausgangssignalen aufgestellt. Die Übertragungsfunktion des Signalaustauschs wird in diesem Fall durch zeitkontinuierliche Modellierung der Abtastung, der Latenzzeiten, sowie Kompensation und Rekonstruktion generiert, wie in Abb. 4.1 zu sehen ist.

In dieser Arbeit wird die zeitkontinuierliche Frequenzanalyse mittels Laplace-Transformation gewählt, da auf diese Weise die Netzwerkeffekte Abtastung und Latenzzeiten zusammen mit dem aus Kompensation und Rekonstruktion bestehendem Kopplungsalgorithmus (s. Abschnitt 3.1) in einer Übertragungsfunktion zusammengefasst werden können (Baumann et al., 2019a). Neben Untersuchungen des gesamten RVPs als geschlossenes System, erlaubt dies Analysen des gesamten Kopplungsprozesses unabhängig der gekoppelten Prototypen durchzuführen.

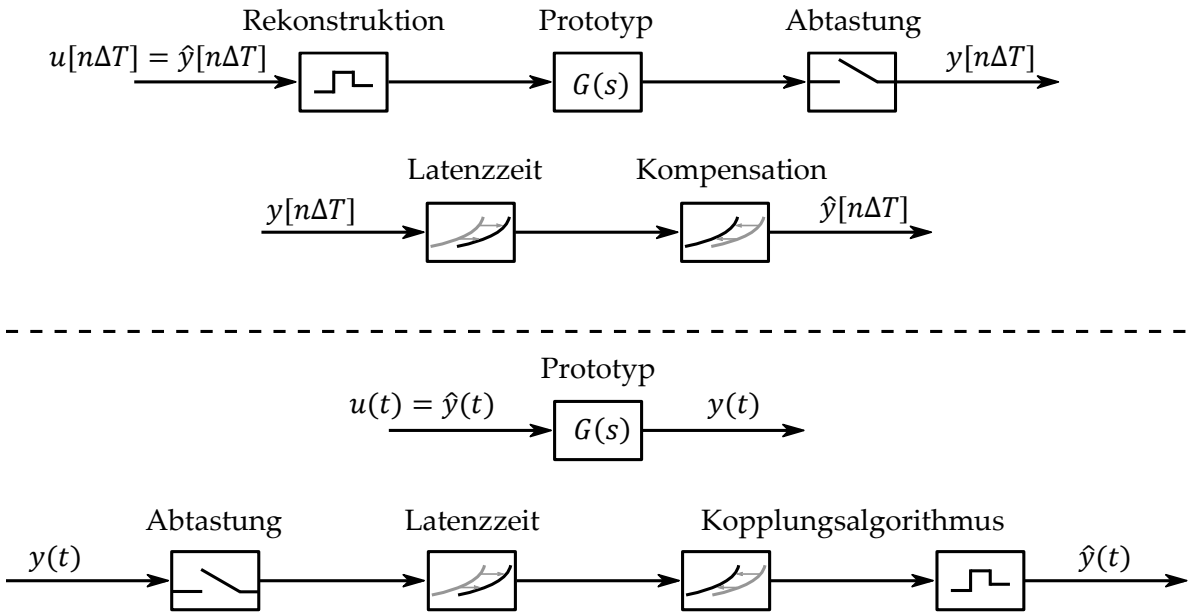


**Abb. 4.1** Blockdiagramme der Übertragungsfunktionen für zeitdiskrete (oben) und zeitkontinuierliche (unten) Frequenzanalyse linearer RVPen

Für die zeitkontinuierliche Analyse linearer RVPen im Frequenzbereich müssen drei Annahmen getroffen werden (Baumann et al., 2019a).

1. Alle gekoppelten Prototypen, sowie deren Eingangs- und Ausgangssignale werden als ideal zeitkontinuierlich angenommen. Für virtuelle Prototypen bedeutet dies, dass Fehler vernachlässigt werden, die bei der numerischen Lösung der Systemgleichungen auftreten. Für reale Prototypen dagegen bedeutet dies, dass alle Ausgangssignale kontinuierlich messbar und alle Eingangssignale ideal aktuierbar sind.

2. Die Makroschrittweite $\Delta T$ ist konstant und klein genug, um Aliasing zu verhindern (keine Einschränkung bei RVPen, s. Abschnitt 2.2.3). Dies kann über das Nyquist-Shannon-Abtasttheorem mit der Bedingung $\bar{\omega}\Delta T \ll \pi$ überprüft werden, wobei $\bar{\omega}$ die maximale Bandbreite des Koppelsignals in $rad/s$ beschreibt (Alan V. Oppenheim, Alan S. Willsky und S. Hamid Nawab, 1996).

3. Die Latenzzeit ist nach Abschnitt 2.2.3 ein $k$-faches der Makroschrittweite und ist für beide Übertragungsrichtungen konstant.

Mit der ersten Annahme gilt für die Übertragungsfunktion des Kopplungsprozesses zweier gekoppelter Prototypen

$$G_p(s) = \frac{U(s)}{Y(s)} = \frac{\hat{Y}(s)}{Y(s)}. \tag{4-1}$$

Wie in Abb. 4.2 dargestellt, lässt sich diese Übertragungsfunktion in drei Übertragungsfunktionen aufteilen, die jeweils ein Element des Kopplungsprozesses abbilden. Es ergibt sich

$$G_p(s) = G_r(s)G_l(s)G_a(s), \tag{4-2}$$

wobei $G_r(s)$ den Kopplungsalgorithmus, $G_l(s)$ die aufgrund der zweiten Annahme als konstant angenommenen Latenzzeit und $G_a(s)$ die für die Signalübertragung notwendige Abtastung des kontinuierlichen Koppelsignals abbildet.

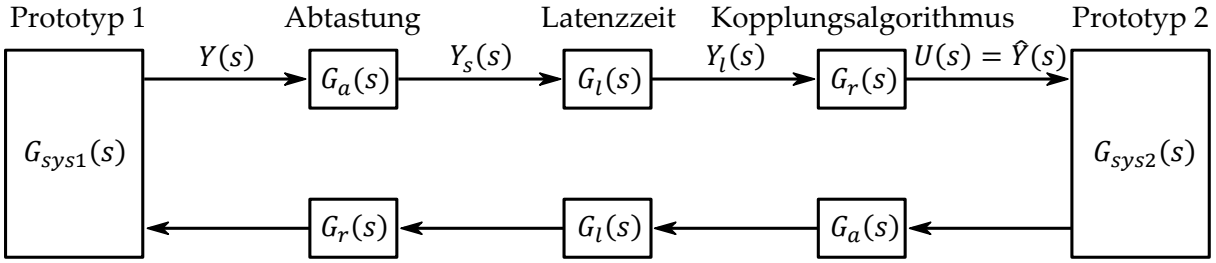


**Abb. 4.2** Elemente eines RVPs mit Netzwerkeffekten und Kopplungsalgorithmus

Weiterhin zeigt Abb. 4.2, dass, sobald $G_p(s)$ bekannt ist, durch Hinzunahme der Übertragungsfunktionen zweier gekoppelter Prototypen $G_{sys1}(s)$ und $G_{sys2}(s)$ der gesamte RVP im Frequenzbereich abgebildet werden kann. Da dies der Form eines geschlossenen Regelsystems entspricht, kann in Abschnitt 4.1.3 das Nyquist-Kriterium zur Untersuchung der Stabilität angewandt werden.

### 4.1.1 Laplace-Transformation des Kopplungsprozesses von RVPen

Im Folgenden wird die Übertragungsfunktion des Kopplungsprozesses $G_p(s)$ für RVPen durch Laplace-Transformation der drei Elemente Abtastung, Latenzzeit und Kopplungsalgorithmus bestimmt.

#### Transformation der Abtastung

Die Auswirkungen einer Abtastung auf kontinuierliche Signale im Frequenzbereich sind zum Beispiel im Detail von Alan V. Oppenheim, Alan S. Willsky und S. Hamid Nawab (1996) hergeleitet. Ein zeitkontinuierliches Koppelsignal $y(t)$ wird durch Multiplikation mit einem Dirac-Kamm, einer periodischen Folge von Dirac-Stößen $\delta(t)$ mit Periode $\Delta T$, abgetastet. Es ergibt sich das abgetastete Signal

$$y_a(t) = \sum_{n=-\infty}^{\infty} y_n \delta(t - n\Delta T). \tag{4-3}$$

Mithilfe des Faltungssatzes der Fourier-Transformation

$$Y_a(j\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} Y(j\omega)P(j(\omega - \Theta)d\Theta. \tag{4-4}$$

und des Fourier-Paares des Dirac-Kamms

$$P(j\omega) = \frac{2\pi}{\Delta T} \sum_{n=-\infty}^{\infty} \delta(\omega - n\omega_a) \tag{4-5}$$

berechnet sich das Spektrum des abgetasteten Signals zu

$$Y_a(j\omega) = \frac{1}{\Delta T} \sum_{n=-\infty}^{\infty} Y(\mathrm{j}\omega - \mathrm{j}n\omega_a). \tag{4-6}$$

Es ergibt sich ein periodisches Spektrum, welches sich aus dem Spektrum des ursprünglichen Signals $Y(i\omega)$ durch eine Skalierung mit $1/\Delta T$, sowie durch Wiederholung mit der Kreisfrequenz $\omega_a = 2\pi/\Delta T$ bildet. Unter der zweiten Annahme ist das Nyquist-Shannon-Abtasttheorem erfüllt und die sich wiederholenden Spektren überlappen sich nicht. Im Frequenzbereich $|\omega| < \pi/\Delta T$, in dem die Bandbreite des abgetasteten Signals liegt, skaliert somit die Abtastung lediglich das Spektrum des Koppelsignals (Benedikt, Watzenig und Hofer, 2013) und es gilt

$$Y_a(j\omega) = \frac{1}{\Delta T} Y(j\omega). \tag{4-7}$$

Nach Einführen der Laplace Variable $s = j\omega$ ergibt sich für die Übertragungsfunktion der Abtastung innerhalb der Bandbreite des abgetasteten Signals

$$G_a(s) = \frac{Y_a(s)}{Y(s)} = \frac{1}{\Delta T}. \tag{4-8}$$

Ausgehend von Gl. (4-3) lässt sich die Laplace-Transformation des abgetasteten Signals auch in Abhängigkeit der abgetasteten Signalwerte $y_n = y(n\Delta T)$ ausdrücken, indem das Laplace-Paar

$$\delta(t - n\Delta T) \;\circ\!\!-\!\!\bullet\; e^{-sn\Delta T} \tag{4-9}$$

angewandt wird. Für ein abgetastetes Koppelsignal mit $y(t) = 0$ für $t < 0$ folgt

$$Y_a(s) = \sum_{n=0}^{\infty} y_n e^{-sn\Delta T}. \tag{4-10}$$

**Transformation der Latenzzeit**

Unter Hinzunahme einer konstanten Latenzzeit $k\Delta T$ (Annahme 3) in Gl. (4-3) gilt für das abgetastete Signal

$$y_l(t) = \sum_{n=0}^{\infty} y_{n-k} \delta(t - n\Delta T), \tag{4-11}$$

womit sich

$$Y_l(s) = e^{-sk\Delta T} \underbrace{\sum_{n=0}^{\infty} y_n e^{-sn\Delta T}}_{Y_a(s)} \tag{4-12}$$

unter Anwendung des Verschiebungssatzes der Laplace-Transformation ergibt. Mit Gl. (4-10) gilt für die Übertragungsfunktion der Latenzzeit

$$G_l(s) = \frac{Y_l(s)}{Y_a(s)} = e^{-sk\Delta T}, \tag{4-13}$$

wonach sich eine konstante Latenzzeit im Frequenzbereich als eine reine Phasenverschiebung auswirkt.

**Transformation des Kopplungsalgorithmus**

Die Laplace-Transformation erfolgt allgemein für lineare Kopplungsalgorithmen, deren Vorhersage eines Zeitschritts einem ARIMA($p$,$d$,$q$) Modell mit $q = 0$ entspricht und dessen Rekonstruktion des kontinuierlichen Signalverlaufs über ein Polynom nullten oder ersten Grades dargestellt ist. Somit erweitert dies die bereits in Baumann et al. (2019a) veröffentlichte Transformation und ist neben den aus der Co-Simulation bekannten Kopplungsalgorithmen ZOH und FOH auch für die neuartigen Algorithmen EROS3 und EROS4 (s. Abschnitt 3.2) gültig. Die im Folgenden hergeleitete allgemeine Transformation ist ebenso für den nichtlinearen FFNN-Kopplungsalgorithmus (s. Abschnitt 3.5) anwendbar, da dieser aus stückweise linearen Funktionsabschnitten aufgebaut ist. Die Laplace-Transformation des von Tandon et al. (2013) entwickelten SMB Kopplungsalgorithmus, welcher auf einem Sliding-Mode Beobachter basiert und vergangene Vorhersagen in zukünftige Vorhersagen mit einbezieht ($q > 1$), wird im Anhang durchgeführt.

Die Kopplungsalgorithmen, welche in Abschnitt 3.3.2 als ARIMA($p$,$d$,0) Modell klassifiziert wurden, haben die allgemeine Form

$$\hat{y}(t) = \underbrace{\sum_{l=0}^{m-1} a_l y_{n-k-l}}_{\hat{y}_1(t)} + \underbrace{\sum_{j=0}^{m-1} A_j y_{n-k-j} \frac{t - t_n}{\Delta T}}_{\hat{y}_2(t)} \text{ für } t_n \le t < t_{n+1}. \tag{4-14}$$

Für die Anzahl der verwendeten vergangenen Signalwerte gilt dabei $m = p + d$. Durch Anwendung der Laplace-Transformation folgt mit $y(t) = 0$ für $t < 0$

$$\mathcal{L}\{\hat{y}(t)\} = \hat{Y}(s) = \int_0^{\infty} \hat{y}(t) e^{-st} dt, \tag{4-15}$$

welches aufgrund der abschnittsweisen Definition von $\hat{y}(t)$ und $t_n = n\Delta T$ mithilfe der Summe

$$\hat{Y}(s) = \sum_{n=0}^{\infty} \int_{n\Delta T}^{(n+1)\Delta T} \hat{y}(t) e^{-st} dt \tag{4-16}$$

beschreibbar ist. Die Linearitätseigenschaft der Laplace-Transformation erlaubt die Lösung von Gl. (4-16) durch unabhängige Transformation der beiden Summanden $\hat{y}_1(t)$ und $\hat{y}_2(t)$ aus Gl. (4-14). Diese ergibt sich durch Anwendung des Verschiebungssatzes auf die verzögerten Signalwerte von $y(t)$. Beim ersten Summanden führt dies nach erfolgter Integration zu

$$\begin{aligned} \mathcal{L}\{\hat{y}_1(t)\} &= \sum_{l=0}^{m-1} a_l e^{-(k+l)s\Delta T} \sum_{n=0}^{\infty} y_n \left[ -\frac{1}{s} e^{-st} \right]_{n\Delta T}^{(n+1)\Delta T} \\ &= \sum_{l=0}^{m-1} a_l e^{-ls\Delta T} \frac{1 - e^{-s\Delta T}}{s} \underbrace{e^{-ks\Delta T}}_{G_l(s)} \underbrace{\sum_{n=0}^{\infty} y_n \, e^{-sn\Delta T}}_{Y_a(s)} \end{aligned} \tag{4-17}$$

und beim zweiten Summanden entsprechend zu

$$\begin{aligned} \mathcal{L}\{\hat{y}_2(t)\} &= \sum_{j=0}^{m-1} A_j e^{-(k+j)s\Delta T} \sum_{n=0}^{\infty} y_n \left[ -\frac{1}{s^2 \Delta T} e^{-st} - \frac{1}{s\Delta T} t e^{-st} + \frac{1}{s} n e^{-st} \right]_{n\Delta T}^{(n+1)\Delta T} \\ &= \sum_{j=0}^{m-1} A_j e^{-js\Delta T} \frac{1 - (1 + s\Delta T) e^{-s\Delta T}}{s^2 \Delta T} \underbrace{e^{-ks\Delta T}}_{G_l(s)} \underbrace{\sum_{n=0}^{\infty} y_n \, e^{-sn\Delta T}}_{Y_a(s)}. \end{aligned} \tag{4-18}$$

Die Addition von Gl.(4-17) und Gl.(4-18) ergibt mit $Y_l(s) = G_l(s) Y_a(s)$ (s. Abb. 4.2)

$$\hat{Y}(s) = \left( \sum_{l=0}^{m-1} a_l e^{-ls\Delta T} \frac{1 - e^{-s\Delta T}}{s} + \sum_{j=0}^{m-1} A_j e^{-js\Delta T} \frac{1 - (1 + s\Delta T) e^{-s\Delta T}}{s^2 \Delta T} \right) Y_l(s), \tag{4-19}$$

woraus die allgemeine Übertragungsfunktion

$$G_r(s) = \frac{\hat{Y}(s)}{Y_l(s)} = \sum_{l=0}^{m-1} a_l e^{-ls\Delta T} \frac{1 - e^{-s\Delta T}}{s} + \sum_{j=0}^{m-1} A_j e^{-js\Delta T} \frac{1 - (1 + s\Delta T) e^{-s\Delta T}}{s^2 \Delta T} \tag{4-20}$$

eines linearen Kopplungsalgorithmus folgt, dessen Vorhersage mittels Gl. (4-14) beschreibbar ist.

Obwohl die Laplace-Transformation nur für lineare Systeme definiert ist, ist das Übertragungsverhalten des nichtlinearen FFNN-Kopplungsalgorithmus aus Abschnitt 3.5 ebenfalls über Gl. (4-20) darstellbar. Durch Verwendung der abschnittsweise linearen Aktivierungsfunktion LReLU ist auch das FFNN abschnittsweise linear und in der Lage zwischen $2^{n_{LReLU}}$ verschiedenen linearen Funktionen umzuschalten, wobei $n_{LReLU}$ für die Anzahl der Neuronen mit LReLU Aktivierungsfunktion im FFNN steht. Für jede dieser linearen Funktionen ergibt sich ein separates Übertragungsverhalten, da die Parameter $a_l$ und $A_j$ verschieden sind.

**Transformation des Kopplungsprozesses**

Die Übertragungsfunktion des gesamten Kopplungsprozesses zweier Prototypen eines RVPs inklusive Abtastung, konstanter Latenzzeit und Kopplungsalgorithmus ergibt sich somit nach Gl. (4-2) zu

$$\begin{aligned} G_p(s) &= G_r(s)G_l(s)G_a(s) \\ &= \left( \sum_{l=0}^{m-1} a_l e^{-ls\Delta T} \frac{1-e^{-s\Delta T}}{s\Delta T} + \sum_{j=0}^{m-1} A_j e^{-js\Delta T} \frac{1-(1+s\Delta T)e^{-s\Delta T}}{s^2\Delta T^2} \right) e^{-ks\Delta T} \end{aligned} \tag{4-21}$$

Bei $G_p(s)$ handelt es sich infolge der enthaltenen Totzeiten um eine transzendente Funktion mit einer einzigen Polstelle bei $s = 0$, deren Vielfachheit abhängig von $a_l$ und $A_j$ eins oder zwei ist. Die Grenzwerte der Übertragungsfunktion ergeben sich zu

$$\begin{aligned} &\lim_{s\to 0-} G_p(s) = \lim_{s\to 0+} G_p(s) = \sum_{l=0}^{m-1} a_l + \frac{1}{2}\sum_{j=0}^{m-1} A_j \text{ und} \\ &\lim_{s\to -\infty} G_p(s) = \lim_{s\to +\infty} G_p(s) = 0. \end{aligned} \tag{4-22}$$

Aufgrund der Nullstelle bei $s = 0$ erfolgt die Berechnung der links- und rechtsseitigen Grenzwerte an dieser Stelle unter Verwendung der Regel von de L'Hospital. Bei allen in dieser Arbeit betrachteten Kopplungsalgorithmen sind $a_l$ und $A_j$ so gewählt, dass $G_p(0) = 1$ gilt. Dies stellt die korrekte Schätzung konstanter Signalverläufe sicher.

### 4.1.2 Untersuchung der Kopplungsalgorithmen im Frequenzbereich

Die Übertragungsfunktion des Kopplungsprozesses erlaubt eine Untersuchung des Signalaustauschs von RVPen in Abhängigkeit des verwendeten Kopplungsalgorithmus und unabhängig der gekoppelten Prototypen.

Zuerst wird der Kopplungsprozess der nicht-iterativen Co-Simulation ohne zusätzliche Latenzzeiten ($k = 0$) unter Verwendung der linearen Kopplungsalgorithmen untersucht, welche bereits in Abschnitt 3.4 miteinander verglichen wurden. Die Übertragungsfunktion des Kopplungsprozesses ergibt sich aus Gl. (4-21), wobei $a_l$ und $A_j$ gemäß Gl. (4-14) und der Definitionen der jeweiligen Kopplungsalgorithmen aus Kap. 3 gewählt werden. Es ergibt sich die Übertragungsfunktion des Kopplungsprozesses der nicht-iterativen Co-Simulation für ZOH ausgehend von Gl. (3-26) zu

$$G_{p,ZOH}(s) = \frac{(1-e^{-s\Delta T})}{s\Delta T}, \tag{4-23}$$

für FOH ausgehend von Gl. (3-29) zu

$$G_{p,FOH}(s) = \frac{(1 + s\Delta T)\left(1 - e^{-s\Delta T}\right)^2}{s^2 \Delta T^2} \tag{4-24}$$

für EROS3 ausgehend von Gl. (3-16) zu

$$G_{p,EROS3} = \frac{(1 - e^{-s\Delta T})}{s\Delta T} + \left(2 - 3e^{-s\Delta T} + e^{-2s\Delta T}\right)\frac{1 - (1 + s\Delta T)e^{-s\Delta T}}{s^2 \Delta T^2} \tag{4-25}$$

und für EROS4 ausgehend von Gl. (3-19) zu

$$G_{p,EROS4} = \frac{(1 - e^{-s\Delta T})}{s\Delta T} + \left(\frac{5}{2} - \frac{9}{2}e^{-s\Delta T} + \frac{5}{2}e^{-2s\Delta T} - \frac{1}{2}e^{-3s\Delta T}\right)\frac{1 - (1 + s\Delta T)e^{-s\Delta T}}{s^2 \Delta T^2}. \tag{4-26}$$

Der SMB wird hier nicht extra aufgeführt. Dessen Übertragungsverhalten entspricht für $k = 0$ exakt der vom FOH, da die Kopplungsbedingung an den Makrozeitpunkten erfüllt ist und daher $\hat{y}_n = y_n \; \forall \, n$ gilt (vgl. Gl. (3-35)).

Abb. 4.3 zeigt das Bode-Diagramm des Kopplungsprozesses einer nicht-iterativen Co-Simulation für unterschiedliche Kopplungsalgorithmen über dem Verhältnis aus Frequenz und Makroschrittweite, welches auf die Nyquistfrequenz normiert ist. Es ist der Frequenzbereich $\omega\Delta T < \pi$ bis zur Nyquistfrequenz dargestellt, welches dem Gültigkeitsbereich der Übertragungsfunktion entspricht (s. Gl. (4-7)). Frequenzen größer der Nyquistfrequenz treten im Koppelsignal nach der zweiten Annahme nicht auf (s. Seite 73). Eine perfekte Kompensation der

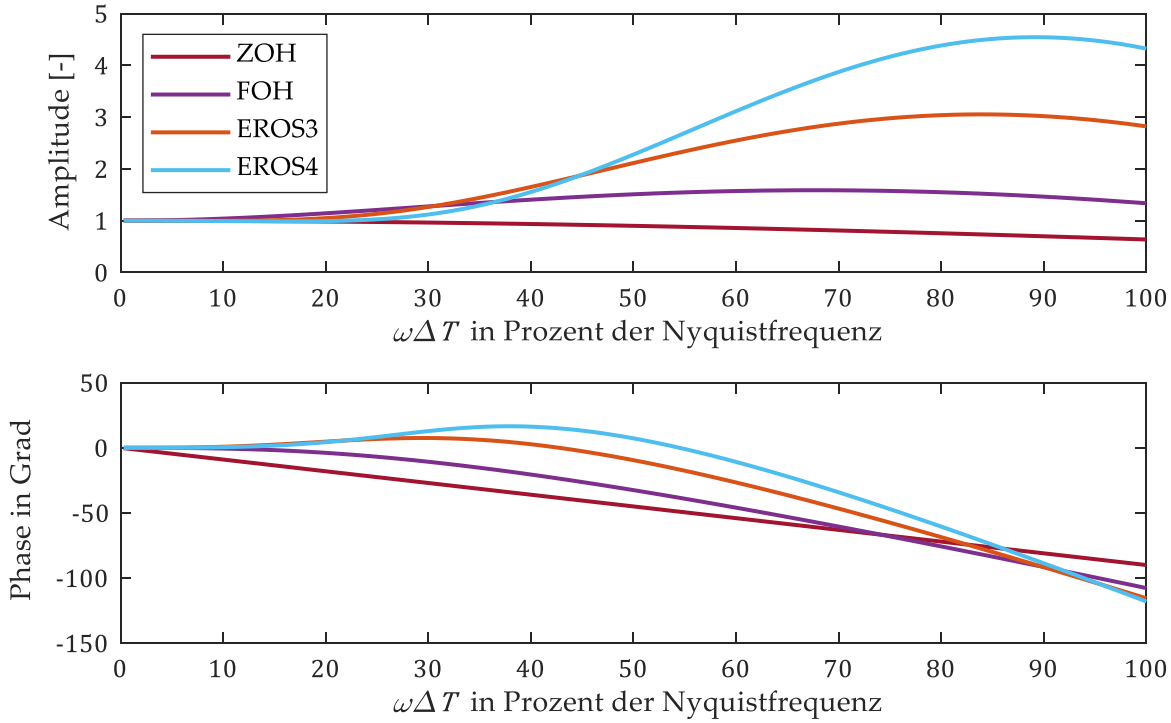


**Abb. 4.3** Bode-Diagramm des Kopplungsprozesses einer nicht-iterativen Co-Simulation bzgl. des Verhältnisses aus Frequenz und Makroschrittweite

Netzwerkeffekte ist für Frequenzen gegeben, bei denen das Übertragungsverhalten des Kopplungsprozesses neutral mit einer Amplitudenverstärkung von eins und einer Phasenverschiebung von null ist. Wie bereits von Benedikt, Watzenig und Hofer (2013) bei ihrer Analyse vom ZOH und FOH in der nicht-iterativen Co-Simulation beschreiben, weichen sowohl die Amplitude als auch die Phase für alle Kopplungsalgorithmen im größten Teil des betrachteten Frequenzbereichs deutlich von diesem idealen Verhalten ab. Daher führen die Autoren heuristisch Grenzen für die maximale Abweichung vom idealen Übertragungsverhaltens des Kopplungsprozesses ein, mit denen der resultierende Kopplungsfehler als tolerierbar angenommen wird. Für die Amplitudenverstärkung wählen sie $|G_P(j\omega_a)| < 1{,}03$ und für die Phasenverschiebung $\left|\angle G_p(j\omega_p)\right| < 3^\circ$, wodurch sich für jeden Kopplungsalgorithmus der gültige Frequenzbereich $\overline{\omega\Delta T} = \min\{\overline{\omega_a\Delta T}, \overline{\omega_p\Delta T}\}$ ergibt.

Im Ausschnitt des Bode-Diagramms in Abb. 4.4 sind diese Grenzen jeweils durch gestrichelte Linien dargestellt. Es ist zu erkennen, dass beim ZOH die Bedingung der Phasenverschiebung bereits für deutlich kleinere Frequenzen verletzt wird, als es bei der Bedingung der Amplitudenverstärkung der Fall ist. Beim FOH ist dies genau umgekehrt. Die EROS-Algorithmen zeigen im niedrigen Frequenzbereich eine bessere Balance zwischen Amplituden- und Phasenfehler, wodurch deren normierte Grenzfrequenz $\overline{\omega\Delta T}$ größer ist als bei den etablierten Algorithmen ZOH und FOH. Die exakten Grenzwerte der gültigen Frequenzbereiche sind weiter unten in Tab. 4.1 gelistet. Verglichen mit den etablierten Kopplungsalgorithmen erlaubt der Einsatz vom EROS3 oder EROS4 in der Co-Simulation mit dem nach Benedikt, Watzenig und

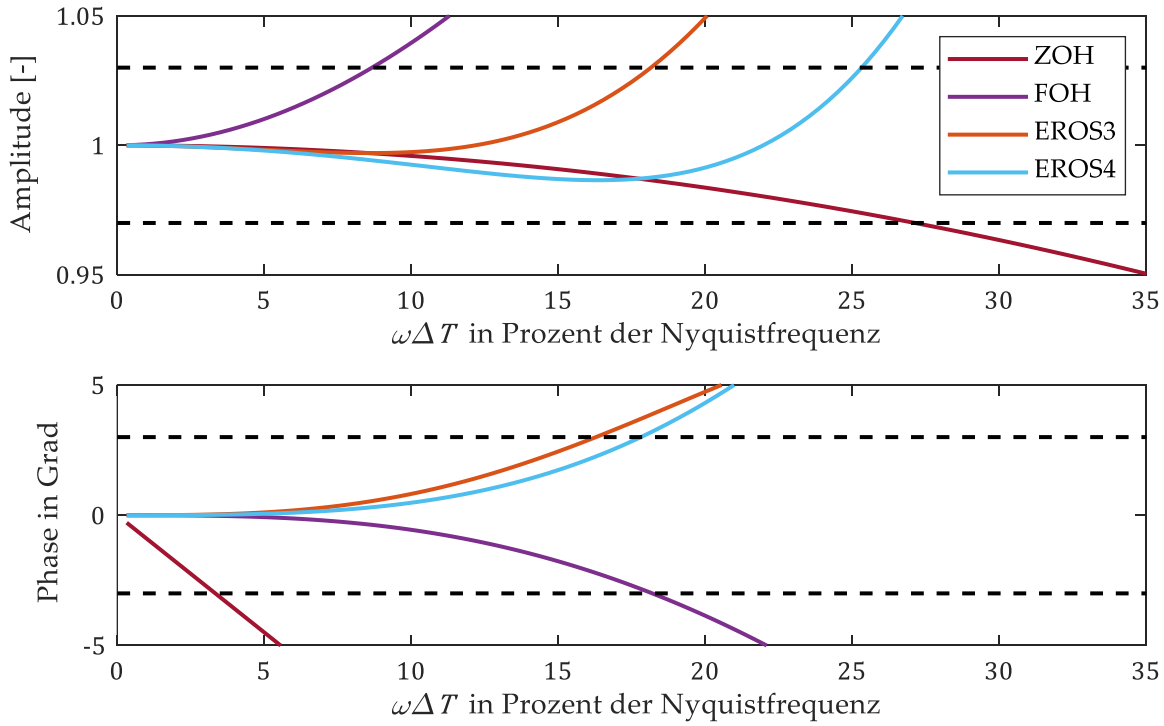


**Abb. 4.4** Bode-Diagramm des Kopplungsprozesses einer nicht-iterativen Co-Simulation bzgl. des Verhältnisses aus Frequenz und Makroschrittweite (Ausschnitt)

Hofer (2013) tolerierbarem Amplituden- bzw. Phasenfehler somit größere Makroschrittweiten und damit einen geringeren Rechenaufwand bzw. eine höhere Genauigkeit der Co-Simulation bei gleicher Makroschrittweite und somit gleichem Rechenaufwand.

Abb. 4.5 und Abb. 4.6 zeigen die entsprechenden Bode-Diagramme für den Kopplungsprozesses von RVPen inklusive einer Latenzzeit, die $k = 3$ Makroschritten entspricht. Die Übertragungsfunktionen des Kopplungsprozesses werden für die verschiedenen Kopplungsalgorithmen analog zum gerade gezeigten Vorgehen berechnet. Da sich das Verhalten des SMBs für $k > 0$ vom FOH unterscheidet (s. Herleitung im Anhang), ist dessen Übertragungsfunktion nun ebenfalls dargestellt.

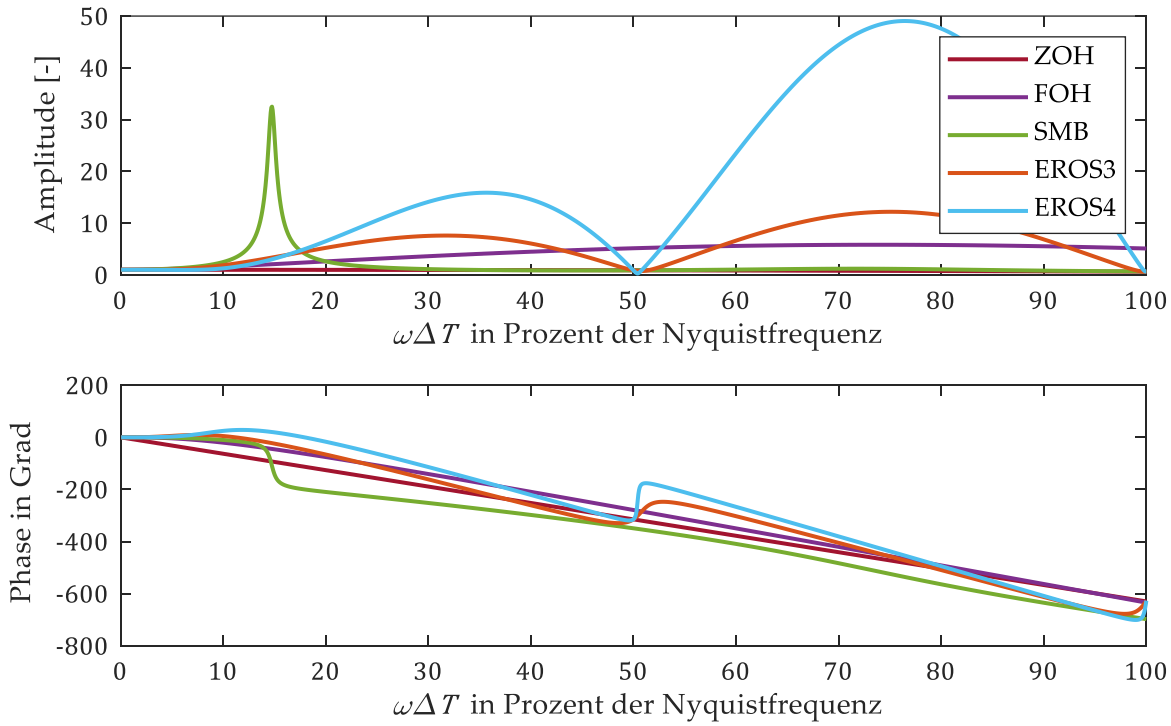


**Abb. 4.5** Bode-Diagramm des Kopplungsprozesses von RVPen mit $k = 3$ bzgl. des Verhältnisses aus Frequenz und Makroschrittweite

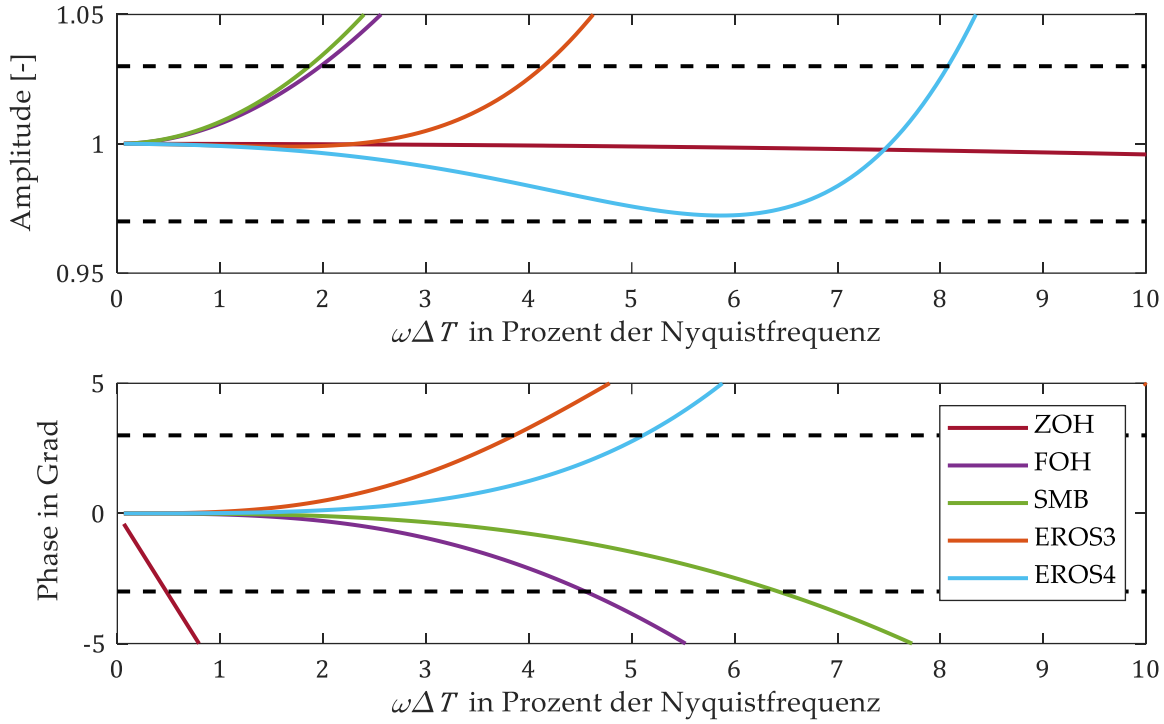


**Abb. 4.6** Bode-Diagramm des Kopplungsprozesses von RVPen mit $k = 3$ bzgl. des Verhältnisses aus Frequenz und Makroschrittweite (Ausschnitt)

Bis auf ZOH sind alle Algorithmen in der Lage den durch die Latenzzeiten hervorgerufenen zusätzlichen Phasenfehler (s. Gl. (4-13)) für niedrige Frequenzen zu kompensieren, bezahlen dies jedoch mit einer hohen Amplitudenverstärkung für größere Frequenzen. Auffällig ist die lokale Verstärkung der Amplitude des SMBs im Bereich um $\omega \approx \lambda$ (entspricht 14,2% der Nyquistfrequenz bei $\Delta T = 0{,}02$ und $k = 3$). Beim kontinuierlichen SMB ist an dieser Stelle ein Pol, welcher für $0 < \lambda < \bar{\lambda} = \pi/2\tau$ stabil ist (Zheng et al., 2018). Aufgrund der in dieser Arbeit berücksichtigten Diskretisierung des Datenaustauschs (s. Abschnitt 3.3.2) ist an dieser Stelle in der Übertragungsfunktion stattdessen ein komplex konjugiertes Polpaar, dessen Stabilitätsbereich kleiner als im kontinuierlichen Fall ist. Für $k = 3$ gilt für den Stabilitätsbereich $0 < \lambda < 0{,}89\bar{\lambda}$ (s. Anhang), weshalb in dieser Arbeit mit $\lambda = 0{,}85\bar{\lambda}$, wie von Zheng et al. (2018) empfohlenen, ein Wert knapp unter der Stabilitätsgrenze der Polstelle gewählt wird. Die anderen Kopplungsalgorithmen, welche die Phase im niedrigen Frequenzbereich kompensieren verstärken dagegen beinahe alle Frequenzen oberhalb ihrer in Tab. 4.1 gelisteten Grenzfrequenzen.

**Tab. 4.1** Grenzfrequenzen der Amplituden- und Phasenbedingung unterschiedlicher Kopplungsalgorithmen in Prozent der Nyquistfrequenz für verschiedene Latenzzeiten $k\Delta T$. Die resultierende Grenzfrequenz $\overline{\omega\Delta T} = \min(\omega_a\Delta T, \overline{\omega_p\Delta T})$ ist jeweils markiert.

| | $k=0$ | | $k=1$ | | $k=3$ | | $k=6$ | |
|---|---|---|---|---|---|---|---|---|
| | $\overline{\omega_a\Delta T}$ | $\overline{\omega_p\Delta T}$ | $\overline{\omega_a\Delta T}$ | $\overline{\omega_p\Delta T}$ | $\overline{\omega_a\Delta T}$ | $\overline{\omega_p\Delta T}$ | $\overline{\omega_a\Delta T}$ | $\overline{\omega_p\Delta T}$ |
| ZOH | 27,13 | **3,33** | 27,13 | **1,11** | 27,13 | **0,48** | 27,13 | **0,25** |
| FOH | **8,66** | 18,20 | **4,01** | 9,10 | **1,97** | 4,55 | **1,11** | 2,57 |
| SMB | **8,66** | 18,20 | **4,25** | 13,16 | **1,87** | 6,42 | **0,99** | 3,37 |
| EROS3 | 18,15 | **16,28** | 8,75 | **7,61** | 4,14 | **3,86** | 2,29 | **2,26** |
| EROS4 | 25,32 | **17,81** | 14,70 | **9,28** | 8,06 | **5,11** | 4,90 | **3,12** |

Bei allen Algorithmen verringert sich die Grenzfrequenz und damit die zulässige Bandbreite des Koppelsignals mit steigender Latenzzeit. Die Grenzfrequenz der entwickelten EROS-Algorithmen ist dabei aufgrund der besseren Balance zwischen Amplituden- und Phasenfehler höher als bei den Vergleichsalgorithmen. EROS4 ist im niedrigen Frequenzbereich etwas besser als EROS3, verstärkt dabei allerdings höhere Frequenzen deutlich mehr als die anderen Algorithmen. Ein Vergleich der Grenzfrequenzen mit den in Abschnitt 3.4 ermittelten Kopplungsfehlern bei Anwendung der Kopplungsalgorithmen zeigt, dass der Kopplungsfehler bei Algorithmen mit größerer Grenzfrequenz geringer ist. Allerdings verstärken FOH und die beiden EROS-Algorithmen Frequenzen oberhalb der Grenzfrequenz und Nyquistfrequenz, welches bei Anregung dieser Frequenzen zu Aliasing Effekten führen kann (vgl. Abschnitt 4.1.1). Die Auswirkungen auf das Systemverhalten werden in Abschnitt 4.1.3 deutlich.

Wie oben gezeigt beeinflussen Kopplungsalgorithmen bei einer Co-Simulation lediglich das Verhältnis aus Genauigkeit und Rechenzeit. Durch Verringerung der Makroschrittweite kann bei Co-Simulationen mit jedem geeigneten Kopplungsalgorithmus der Kopplungsfehler gegen Null geführt werden, da die Kopplungsbedingung an jedem Makroschritt erfüllt ist (s. Abschnitt 2.1.1). Bei RVPen dagegen ist die Makroschrittweite durch die Echtzeitanforderung an die virtuellen Prototypen nicht beliebig verkleinerbar und die Kopplungsbedingung ist aufgrund der Latenzzeiten des Datenaustauschs niemals erfüllt. Da die Makroschrittweite und $k$ antiproportional zusammenhängen, lässt sich somit die These aufstellen, dass die EROS-Algorithmen, welche höhere Grenzfrequenzen aufweisen als die Vergleichsalgorithmen, RVPen auch für größere Latenzzeiten (relativ zur Bandbreite der Koppelsignale) stabilisieren. Dies wird im nächsten Abschnitt durch Betrachtung eines Gesamtsystems überprüft.

### 4.1.3 Stabilität linearer Real-Virtueller Prototypen

Wie im vorangegangenen Abschnitt bei der Analyse des Kopplungsprozesses gezeigt, können unabhängig der gekoppelten Prototypen Aussagen darüber getroffen werden, in welchem

Frequenzbereich mit den jeweiligen Kopplungsalgorithmen geringe Kopplungsfehler zu erwarten sind. In diesem Abschnitt wird die Analyse im Frequenzbereich auf ein Gesamtsystem eines RVPs ausgeweitet und eine Methodik mittels des Nyquist-Kriteriums vorgeschlagen, um die Stabilitätsgrenzen der betrachteten Kopplungsalgorithmen abhängig der Latenzzeiten zu ermitteln. Als Beispielsystem dient der lineare Zweimassenschwinger aus Abschnitt 2.3.1.

Nach Abb. 4.2 ist ein RVP als geschlossenes Regelsystem interpretierbar, dessen Übertragungsfunktion der offenen Kette durch Multiplikation der Übertragungsfunktionen der gekoppelten Prototypen entsteht. Die Auswirkungen der Netzwerkeffekte und des verwendeten Kopplungsalgorithmus werden berücksichtigt, indem jedes Koppelsignal mit der Übertragungsfunktion des Kopplungsprozesses multipliziert wird. Die Gesamtübertragungsfunktion der offenen Kette des Zweimassenschwingers ergibt sich damit zu

$$G_{RVP}(s) = \underbrace{[g_{11}(s) \quad g_{12}(s)]}_{\boldsymbol{G}_{sys2}(s)} \begin{bmatrix} G_p(s) & 0 \\ 0 & G_p(s) \end{bmatrix} \underbrace{\begin{bmatrix} g_{11}(s) \\ g_{21}(s) \end{bmatrix}}_{\boldsymbol{G}_{sys1}(s)} G_p(s). \tag{4-27}$$

Die Übertragungsfunktionen der Prototypen des Zweimassenschwingers $\boldsymbol{G}_{sys1}(s)$ und $\boldsymbol{G}_{sys2}(s)$ werden dazu aus den in Abschnitt 2.3 aufgestellten Systemgleichungen über den Zusammenhang

$$\boldsymbol{G}(s) = \boldsymbol{C}(s\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{B} + \boldsymbol{D} \tag{4-28}$$

bestimmt (Lunze, 2013). Es wird angenommen, dass die Netzwerkeffekte in beide Übertragungsrichtungen gleich sind und für jedes Koppelsignal identische Kopplungsalgorithmen verwendet werden, wodurch sich die einzelnen $G_p(s)$ nicht unterscheiden. $G_{RVP}(s)$ beschreibt in diesem Fall das Übertragungsverhalten des gekoppelten RVPs bezüglich der Koppelsignale am Ausgang des zweiten Systems. Das Übertragungsverhalten des RVPs entsteht durch Schließung des Regelsystems über eine positive Rückführung, welche durch Multiplikation von $G_{RVP}(s)$ mit minus eins in die übliche negative Rückführung überführbar ist. Durch den Schnitt am Ausgang des zweiten Systems des Zweimassenschwingers ist $G_{RVP}(s)$ ein SISO-System, wodurch die folgende Stabilitätsanalyse erleichtert wird.

Abb. 4.7 zeigt das Bode-Diagramm der offenen Kette des $G_{RVP}(s)$ für den Zweimassenschwinger mit den in dieser Arbeit untersuchten linearen Kopplungsalgorithmen. Die gewählten Systemparameter sind in Abschnitt 2.3 gegeben und die Netzwerkeffekte sind zunächst wie in Kap. 3 über $\Delta T = 0{,}02\text{s}$ und $k = 3$ definiert. Der Verlauf der Referenz beschreibt das Übertragungsverhalten bei idealer Kopplung der Prototypen ($G_p(s) = 1$).

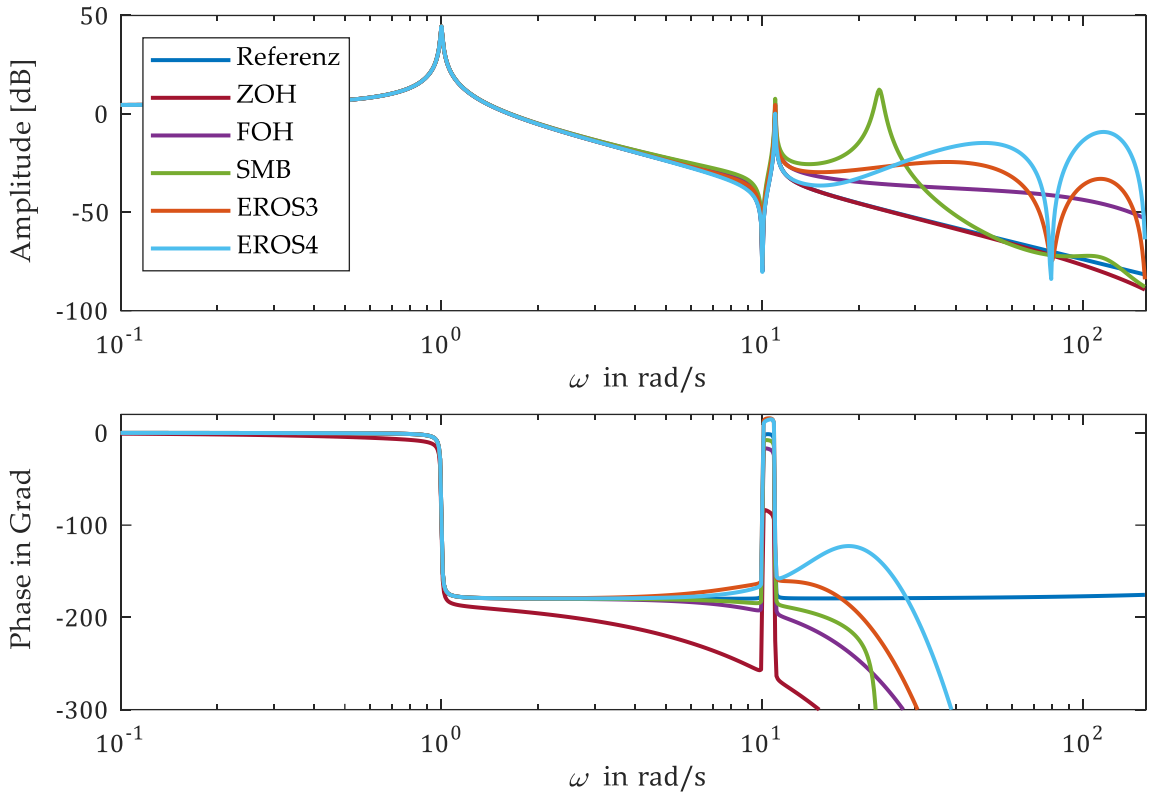


**Abb. 4.7** Bode-Diagramm der offenen Kette unter Verwendung verschiedener Kopplungsalgorithmen mit $k = 3$

Wie auf Basis der vorangegangenen Analyse der linearen Kopplungsalgorithmen zu erwarten, weist das Amplitudendiagramm für alle Kopplungsalgorithmen bis auf ZOH für hohe Frequenzen eine größere Amplitudenverstärkung als die ideale Kopplung auf. Da diese Verstärkung der Amplitude erst für Frequenzen hinter beiden Eigenfrequenzen des Zweimassenschwingers bei $\omega_1 = 1\,rad/s$ und $\omega_2 = 11\,rad/s$ das Maximum erreicht, dämpft die Dynamik des Zweimassenschwingers dieses. Beim FOH, EROS3 und EROS4 wird daher der kritische Wert von 0dB für Frequenzen oberhalb der zweiten Eigenfrequenz nicht überschritten, wohingegen beim SMB aufgrund des bereits diskutierten Polpaares eine maximale Verstärkung der Amplitude von 12,3db hinter der zweiten Eigenfrequenz vorliegt und das Systemverhalten des Zweimassenschwingers bei Anregung dieser Frequenz qualitativ verändert. Im Phasendiagramm strebt die Phasenverschiebung aufgrund der Netzwerkeffekte für alle Kopplungsalgorithmen Richtung $-\infty$. Während beim ZOH diese Phasenverschiebung bereits im Bereich der ersten Eigenfrequenz um mehr als 10° von der Referenz abweicht, kompensieren die anderen Kopplungsalgorithmen die Phasenverschiebung bis in den Bereich der zweiten Eigenfrequenz.

### Stabilitätsanalyse unter Verwendung des Nyquist-Kriteriums

Das Nyquist-Kriterium erlaubt es Aussagen über die Eingangs-Ausgangs-Stabilität eines geschlossenen Regelsystems auf Basis der Übertragungsfunktion der offenen Kette zu treffen.

Da es auch für Systeme mit Totzeiten (z.B. Latenzzeiten) gültig ist, eignet es sich für die Stabilitätsanalyse von RVPen.

Die Definition des Nyquist-Kriteriums findet sich unter anderem in Lunze (2013):

> *Eine offene Kette $\Sigma_0$ mit der Übertragungsfunktion $G_0(s)$ führt genau dann auf einen eingangs-ausgangs-stabilen Regelkreis $\bar{\Sigma}$, wenn die Ortskurve $G_0(j\omega)$ für $\omega = -\infty \ldots + \infty$ den Punkt $-1 + j0$ der komplexen Ebene $-n^+$-mal im Uhrzeigersinn umschließt. Dabei bezeichnet $n^+$ die Anzahl der Pole von $G_0(s)$ mit positivem Realteil.*

Die Ortskurve ist dabei die Abbildung der Nyquist-Kurve, welche im Ursprung beginnt und in einem Halbkreis mit unendlichem Radius die rechte Halbebene im Uhrzeigersinn umschließt. Für die Anwendung des Nyquist-Kriteriums müssen folgende Bedingungen erfüllt sein:

- Die offene Kette darf nicht sprungfähig sein, weshalb

$$\lim_{s \to \pm\infty} G_0(s) = 0 \tag{4-29}$$

  gelten muss (Lunze, 2013). Dies ist für die offene Kette des Zweimassenschwingers $G_{RVP}(s)$ nach Gl. (4-22) und Gl. (4-27) für alle betrachteten Kopplungsalgorithmen erfüllt.

- Für die statische Verstärkung der offenen Kette muss $G(0) > 0$ gelten, wenn das Regelsystem negative Rückführung besitzt (Lunze, 2013). Dies ist für $G_{RVP}(s)$ ebenfalls nach den Gl. (4-22) und Gl. (4-27) für alle betrachteten Kopplungsalgorithmen gültig.

- Ein System mit Totzeiten ist unendlich dimensional und kann daher unendlich viele Polstellen besitzen. Zur Anwendung des Nyquist-Kriteriums ist es notwendig, dass die Anzahl der instabilen Pole der offenen Kette endlich ist. Dies ist nach Jugo (2001) gegeben, wenn das System nicht sprungfähig ist und ist demnach für $G_{RVP}(s)$ nach der ersten Bedingung erfüllt.

Die Übertragungsfunktion der offenen Kette $G_{RVP}(s)$ besitzt unabhängig des betrachteten Kopplungsalgorithmus keine instabilen Polstellen. Die Polstellen bei $s = 0$, welche aufgrund des Modells des Kopplungsprozesses in $G_{RVP}(s)$ vorhanden sind (s. Gl. (4-22)), zählen nicht als instabiler Pol, da sie nicht von der Nyquist-Kurve umschlossen werden. Die Nyquist-Kurve wird dazu in einem Halbkreis mit infinitesimal kleinem Radius um diese Polstellen herumgeführt, ohne die Aussage des Stabilitätskriteriums zu beeinflussen (Lunze, 2013). In diesem Fall der stabilen offenen Kette ist das geschlossene Regelsystem nach dem Nyquist-Kriterium eingangs-ausgangs-stabil, wenn der kritische Punkt $(-1 + j0)$ von der Ortskurve nicht umschlossen wird. Sobald ein Eingangs-Ausgangs-Paar stabil ist, ist bei einer stabilen

offenen Kette außerdem innere Stabilität zwischen den Teilsystemen gegeben und damit Stabilität aller Eingangs-Ausgangs-Paare.

In Abb. 4.8 sind die Nyquist-Diagramme der offenen Kette $G_{RVP}(s)$ für die verschiedenen Kopplungsalgorithmen dargestellt. Die Referenz beschreibt den Fall idealer Kopplung, während bei den anderen Diagrammen Netzwerkeffekte mit $\Delta T = 0{,}02$ und $k = 3$ vorliegen. Die Ortskurve verläuft in Pfeilrichtung und bildet die Nyquist-Kurve ab, welche bei $\omega = 0$ (durch schwarzen Kreis gekennzeichnet) beginnt und dort nach Umschlingen der rechten Halbebene auch wieder endet. Der zu positiven Frequenzen gehörende Teil der Ortskurve ist als durchgezogene und der zu negativen Frequenzen gehörende Teil als gestrichelte Linie dargestellt. Das rote Kreuz markiert jeweils den kritischen Punkt. Die kreisförmigen Abschnitte der Ortskurve in der Abbildung der Referenz sind auf die beiden Eigenfrequenzen des Zweimassenschwingers zurückzuführen und umschließen den kritischen Punkt nicht, woraus die Eingangs-Ausgangs-Stabilität des Zweimassenschwingers im Fall idealer Kopplung folgt.

Mit ZOH als Kopplungsalgorithmus wird der kritische Punkt mehrfach umschlungen, woraus nach dem Nyquist-Kriterium Instabilität folgt. Wie in Tab. 4.2 aufgezeigt, ist der RVP mit ZOH bei der gegebenen Makroschrittweite selbst ohne Latenzzeiten ($k = 0$) nicht stabilisierbar. Bei Verwendung der anderen Kopplungsalgorithmen ist der RVP für die in Abb. 4.8 gegebenen Netzwerkeffekte stabil, da der kritische Punkt nicht umschlungen wird. Dies deckt sich mit den Beobachtungen aus Abschnitt 3.4. FOH hat allerdings eine sehr geringe Amplitudenreserve im Bereich der zweiten Eigenfrequenz und wird für $k > 3$ genau in diesem Bereich instabil (s. Tab. 4.2). Beim SMB ist im Bereich $\omega \approx \lambda$ der Ortskurve aufgrund des dort liegenden komplex konjugierten Polpaares des Kopplungsalgorithmus (s. Abb. 4.5) ein weiterer Kreisausschnitt zu erkennen, welcher für größere $k$ näher an die zweiten Eigenfrequenz rückt und das System destabilisiert. Die EROS-Algorithmen haben ein größeres Stabilitätsgebiet. Während EROS3 auch im Bereich der zweiten Eigenfrequenz instabil wird, ist die Ursache der Instabilität vom EROS4 die starke Verstärkung hoher Frequenzen. Diese werden von den Dynamiken des Zweimassenschwingers nicht mehr ausreichend gedämpft und führen beim Betrieb des RVPs zu Aliasing. Im Nyquist-Diagramm ist dies durch die für Totzeiten charakteristische Spirale um den Ursprung erkennbar. Ein Umschlingen des kritischen Punktes im Bereich der Eigenfrequenzen des Zweimassenschwingers tritt beim EROS4 erst für $k > 9$ auf. Für den betrachteten RVPen bildet demnach EROS3 den bzgl. Stabilität besten Kompromiss zwischen geringer Amplitudenverstärkung und Kompensation der Phasenverschiebung im Bereich der Bandbreite der gekoppelten Prototypen sowie Verstärkung hoher Frequenzen.

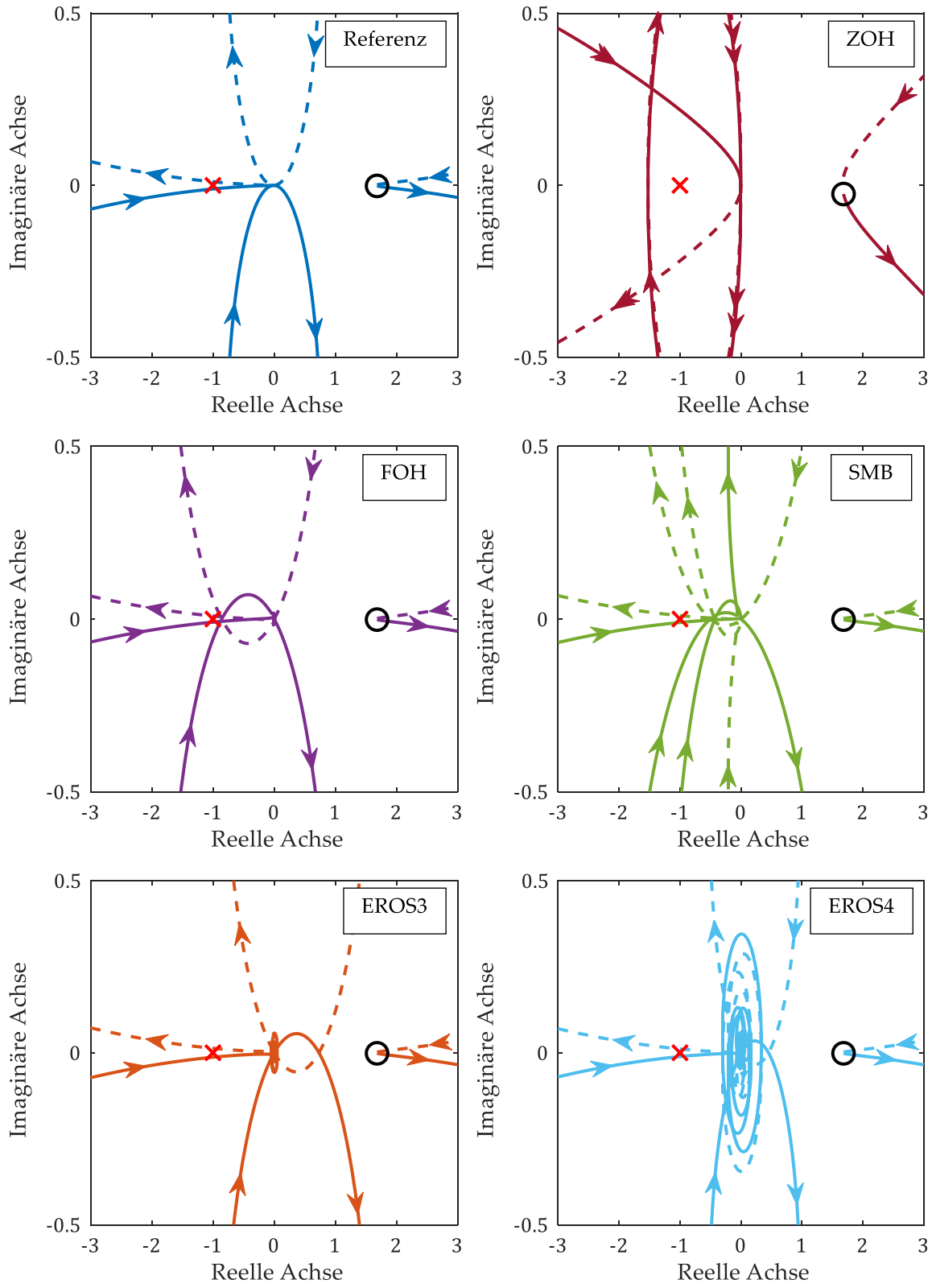


**Abb. 4.8** Nyquist-Diagramme der Übertragungsfunktion der offenen Kette $G_{RVP}(s)$ für die verschiedenen Kopplungsalgorithmen mit $\Delta T = 0{,}02$ und $k = 3$. Der kritische Punkt ist mittels eines roten Kreuzes, der Punkt $\omega = 0$ mittels eines schwarzen Kreises markiert.

**Tab. 4.2** Stabilitätsgrenzen von $G_{RVP}(s)$ für die betrachteten Kopplungsalgorithmen

| | Maximales $k$ für welches $G_{RVP}(s)$ bei $\Delta T = 0{,}02$ stabil ist | Frequenzbereich, der bei größerem $k$ zu Instabilität führt |
|---|---|---|
| ZOH | − | $\sim\omega_1$ |
| FOH | 3 | $\sim\omega_2$ |
| SMB | 3 | $\sim\omega_2$ |
| **EROS3** | **5** | $\boldsymbol{\sim\omega_2}$ |
| **EROS4** | **4** | $\boldsymbol{> \omega_2}$ |

Wie auf Basis der Bode-Diagramme aus Abschnitt 4.1.2 bereits erwartet, erlauben die neuartigen EROS-Algorithmen einen stabilen Betrieb des RVPs für größere Latenzzeiten als die Vergleichsalgorithmen. Dies ermöglicht die Verteilung der Prototypen des RVPs über eine größere Distanz und somit flexiblere Einsatzmöglichkeiten.

Wie oben gezeigt, existiert bei linearen RVPen ein Zusammenhang zwischen der von Benedikt, Watzenig und Hofer (2013) eingeführten und heuristisch ermittelten Grenzfrequenzen und dem Kopplungsfehler. Allerdings ist die Betrachtung der Grenzfrequenz im Unterschied zur hier verwendeten Methode, kein Maß für die Stabilität eines RVPs. Weiterhin ist an diesem Beispiel zu sehen, dass die Grenzfrequenz eher konservativ gewählt ist. Wie in Abschnitt 3.4 gezeigt, bilden EROS3 und ERSO4 den Verlauf der Referenz unter den gegebenen Netzwerkeffekten sehr gut ab, obwohl die Bandbreite außerhalb der Grenzfrequenz liegt. Das maximale Verhältnis aus Makroschrittweite und Bandbreite ergibt sich zu $\overline{\omega\Delta T} = 7{,}00$, wenn als Bandbreite der Frequenzbereich bis zur zweiten Eigenfrequenz des Zweimassenschwingers festgelegt wird und liegt damit nach Tab. 4.1 für $k = 3$ über den berechneten Grenzfrequenzen.

## 4.2 Optimale Auslegung der Kompensation im Frequenzbereich

Bei der Analyse der Kopplungsalgorithmen im vorhergehenden Kapitel ist für lineare Systeme gezeigt, dass die Kompensation des Phasenfehlers bei gleichzeitig möglichst geringer Amplitudenverstärkung innerhalb der Bandbreite der RVPen den Kopplungsfehler verringert. Allerdings kann der Einsatz von Kopplungsalgorithmen zur Kompensation der Latenzzeiten aufgrund einer Verstärkung hoher Frequenzen zur Instabilität der RVPen führen. Die am häufigsten verwendeten und in Abschnitt 3.3 diskutierten Kopplungsalgorithmen bilden auf verschiedene Weise einen Kompromiss dieser sich gegenseitig bedingenden Effekte, wodurch es keinen Kopplungsalgorithmus gibt, der für alle RVPen am besten geeignet ist. Allerdings kann, wie in Abschnitt 3.5 gezeigt, ein FFNN als Kopplungsalgorithmus eingesetzt werden, der sich zu Beginn wie ein bekannter Kopplungsalgorithmus verhält und sich online an das Verhalten eines bestimmten RVPen anpasst.

Das Ziel dieses Kapitels ist es bereits vor der Ausführung eines RVPs, einen linearen Kopplungsalgorithmus zu berechnen, der die Phasenverschiebung aufgrund der Latenzzeiten in den Koppelsignalen eines RVPs optimal kompensiert, dabei die Amplitudenverstärkung innerhalb der Bandbreite des Koppelsignals neutral lässt und die Amplitude hoher Frequenzen maximal so viel verstärkt, dass das gekoppelte System nicht instabil wird. Um diesen optimalen Kopplungsalgorithmus zu finden, wird basierend auf dem Verfahren zur Analyse der RVPen im Frequenzbereich aus dem vorherigen Kapitel ein Optimierungsproblem entworfen, dessen Lösung eben jenem optimalen Kopplungsalgorithmus für den speziellen RVPen entspricht.

### 4.2.1 Definition des Optimierungsproblems

Das Optimierungsproblem wird für lineare Kopplungsalgorithmen der Form eines ARIMA Modells mit $q = 0$ aufgestellt, da dies der generischen Form der meistverwendeten Kopplungsalgorithmen entspricht (vgl. Abschnitt 3.3.2). Damit ist die Berechnung eines optimalen Kopplungsalgorithmus nur für lineare RVPen möglich. Falls die Koppelsignale des RVPs starke Nichtlinearitäten aufweisen, kann ein FFNN mit dem Verhalten des optimalen linearen Kopplungsalgorithmus initialisiert werden und sich online auf diese Nichtlinearitäten anpassen (vgl. Abschnitt 3.5).

Das Optimierungsproblem dessen Lösung den optimalen Kopplungsalgorithmus ergibt, basiert direkt auf dem Verfahren zur Analyse der RVPen im Frequenzbereich aus dem vorherigen Kapitel.

In Gl. (4-14) in Kap. 4 ist die allgemeine Form eines als ARIMA($p$,$d$,0) klassifizierten Kopplungsalgorithmus gegeben und Gl. (4-21) beschreibt den gesamten Kopplungsprozesses eines RVPs im Frequenzbereich. Die zu optimierenden Parameter sind die Faktoren nullter und erster Ordnung der Funktion des Algorithmus, welche in den Vektoren $\boldsymbol{a}$ und $\boldsymbol{A}$ definiert sind. Die Makroschrittweite $\Delta T$ und die Anzahl an Makroschritten Verzögerung $k$ hängen vom Aufbau des RVPs ab (vgl. Abschnitt 2.2) und sind daher bei der Auslegung eines Kopplungsalgorithmus für Koppelsignale eines bestimmten RVPs bereits bekannt bzw. können gemessen werden. Die Anzahl der vergangenen Signalwerte $m$, die der Kopplungsalgorithmus verwendet, sind für diese Optimierung ebenfalls als gegeben angenommen. Am Ende dieses Kapitels wird diskutiert, wie dieser Parameter zu wählen ist.

Die gewählte Kostenfunktion des Optimierungsproblems

$$J(\boldsymbol{a}, \boldsymbol{A}) = \alpha J_a + \beta J_p + \gamma J_r \tag{4-30}$$

besteht aus drei Summanden, deren Gewichtungsparameter $\alpha$, $\beta$ und $\gamma$ weiter unten diskutiert werden (Baumann et al., 2019a). Die ersten beiden Summanden bestrafen eine nicht erfolgte

Kompensation der Phasenverschiebung oder eine hinzugefügte Verstärkung der Amplitude des Koppelsignals innerhalb dessen Bandbreite. Dabei definiert der erste Summand

$$J_a = \int_{\omega_u}^{\omega_g} \frac{1 - \left|G_p(j\omega)\right|}{\omega_g - \omega_u} \mathrm{d}\omega \tag{4-31}$$

die Kosten, die entstehen, wenn die Amplitudenverstärkung des Kopplungsprozesses innerhalb der Bandreite des Koppelsignals des RVPs $\omega_b \in [\omega_u, \omega_g]$ nicht neutral d.h. ungleich eins ist. Entsprechend enthält der zweite Summand

$$J_p = \int_{\omega_u}^{\omega_g} \frac{\angle G_p(j\omega)}{\omega_g - \omega_u} \mathrm{d}\omega \tag{4-32}$$

die Kosten, die durch eine nicht neutrale Phasenverschiebung, d.h. ungleich Null, innerhalb der Bandbreite entstehen. Über den dritten Summanden

$$J_r = \int_0^{\omega_u} \max\left(|G_p(j\omega)|, 1\right) \mathrm{d}\omega - \omega_u + \int_{\omega_g}^{\frac{2\pi}{\Delta T}} \max\left(\left|G_p(j\omega)\right| - \left(\frac{\omega}{\omega_g}\right)^{\frac{1}{2^r}}, 0\right) \mathrm{d}\omega \tag{4-33}$$

fließen die Kosten einer Amplitudenverstärkung des Kopplungsprozesses außerhalb der Bandbreite ein, die nicht durch die natürliche Dämpfung des RVPs ausgeglichen wird. Während für $\omega < \omega_u$ jede Amplitudenverstärkung größer eins die Kosten erhöht, wird für $\omega > \omega_g$ die natürliche Dämpfung des RVPs über dessen relativen Grad $r$ einbezogen. Der Kopplungsalgorithmus darf somit hohe Frequenzen in einem Maß verstärken, dass maximal der Dämpfung des RVPs entspricht, sodass hohe Frequenzen im Gesamtsystem nicht angeregt werden.

Das Optimierungsproblem ergibt sich somit zu

$$\min_{\boldsymbol{a},\boldsymbol{A}} J(\boldsymbol{a}, \boldsymbol{A}), \text{sodass } G_p(0) = 1. \tag{4-34}$$

Die Nebenbedingung stellt sicher, dass $\lim_{\omega \to 0} G_p(s = i\omega) = 1$ gilt und somit konstante Signalverläufe ($\omega = 0$) korrekt vorhergesagt werden (vgl. Abschnitt 4.1).

Für die Gewichtungsparameter gilt

$$\gamma \gg \alpha, \beta, \tag{4-35}$$

damit die Amplitudenverstärkung hoher Frequenzen des Kopplungsalgorithmus innerhalb der gegebenen Grenzen bleibt und nicht die Stabilität des RVPs gefährdet. Weiterhin wird basierend auf der Argumentation mittels heuristischer Grenzen von Benedikt, Watzenig und Hofer (2013) $\alpha = 100\beta$ gesetzt, damit innerhalb der Bandbreite des Koppelsignals eine Phasendifferenz von 1° genauso große Kosten erzeugt wie ein Amplitudenfehler von 1%.

Der größte Vorteil dieser Definition der Kostenfunktion ist, dass nur sehr wenig Informationen über das zu koppelnde System benötigt werden, wodurch sie allgemein anwendbar ist. Es sind keine Übertragungsfunktionen der gekoppelten Prototypen notwendig. Stattdessen ist es ausreichend die Bandbreite der Koppelsignale und den relativen Grad des Systems zu kennen, welche für reale Prototypen beispielsweise über eine Fourier-Analyse abschätzbar sind (Bloomfield, 2004). Selbst wenn die Bandbreite oder der relative Grad eines RVPs nicht genau bestimmt werden können, kann mithilfe dieses Verfahrens auch mit einer konservativen Schätzung der Parameter ein Kopplungsalgorithmus gefunden werden, der verglichen mit den generischen Kopplungsalgorithmen den Kopplungsfehler verringert. In Abschnitt 6.2 wird dies anhand eines produktiv genutzten RVPs demonstriert.

**Einfluss der Anzahl vergangener Signalwerte**

Ein weiterer für die Optimierung zu wählender Parameter ist die Anzahl der vergangenen Signalwerte $m$, welche der optimierte Kopplungsalgorithmus verwendet. Der Parameter beeinflusst wie in Gl. (4-21) den Kopplungsprozess $G_p(s)$ und fließt darüber in die Kostenfunktion des Optimierungsproblems in Gl. (4-30) ein.

Eine genaue Betrachtung der Übertragungsfunktion der Kopplungsalgorithmen

$$G_r(s) = \sum_{l=0}^{m-1} a_l e^{-lsΔT} \frac{1 - e^{-sΔT}}{s} + \sum_{j=0}^{m-1} A_j e^{-jsΔT} \frac{1 - (1 + sΔT)e^{-sΔT}}{s^2 ΔT} \tag{4-36}$$

aus Gl. (4-20) zeigt, dass der Parameter $m$ festglegt aus wie vielen mit $a_l$ bzw. $A_j$ gewichteten Summen die beiden Summanden jeweils bestehen. Eine Anwendung der Eulerschen Formel auf den Faktor $e^{-nsΔT}$, welcher in beiden Summen vorkommt, offenbart weiterhin, dass mit zunehmendem $m$ die Übertragungsfunktion aus mehr überlagerten sinusförmigen Wellen besteht, deren Frequenz proportional zu $m$ zunimmt, da mit $s = jω$

$$e^{-jnΔTω} = \cos(\mathrm{n}ΔTω) + j \sin(nΔTω) \text{ für } n = 0{,}1 \ldots m - 1 \tag{4-37}$$

gilt.

Somit nimmt mit einem größeren Wert für $m$ die Anpassungsfähigkeit des optimierten Kopplungsalgorithmus zu. Durch die größere Anzahl gewichteter überlagerter sinusförmiger Wellen mit höherer Frequenz aus denen die Übertragungsfunktion entsteht, ist eine Kompensation der Latenzzeiten durch Einsatz des optimierten Kopplungsalgorithmus für größere und genauer definierte Frequenzbereiche möglich, ohne dort gleichzeitig die Amplitude zu verstärken. Allerdings führt dies gleichzeitig aufgrund des Wasserbetteffekts (Janschek, 2010) zu einer größeren Amplitudenverstärkung in anderen Frequenzbereichen, welche aufgrund des dritten Summanden der Kostenfunktion nicht unbegrenzt zugelassen werden. Daher verbessert sich das Ergebnis der Optimierung zunächst mit wachsendem $m$, erreicht aber ab einem

bestimmten Wert ein Plateau (s. Abschnitt 4.2.2). Gleichzeitig ist bei gleicher Kompensationsfähigkeit immer der Kopplungsalgorithmus vorzuziehen, der weniger vergangen Signalwerte berücksichtigt, da er unter anderem weniger Rechenzeit benötigt und nach Diskontinuitäten wieder schneller anwendbar ist (s. Abschnitt 5.3). Die Empfehlung zur Wahl der Anzahl vergangen Signalwerte $m$ in dieser Arbeit ist daher die Optimierung mehrfach mit unterschiedlichen Werten für $m$ durchzuführen und dann den kleinsten Wert zu wählen ab dem sich das Optimierungsergebnis nicht mehr signifikant ändert.

### 4.2.2 Anwendung der Optimierung auf lineare Real-Virtuelle Prototypen

Das Optimierungsproblem wird auf den linearen Zweimassenschwinger aus Abschnitt 2.3.1 angewandt, der mit denselben Parametern ($\Delta T = 0{,}02s,\ k = 3$) wie in Abschnitt 3.4 als RVP simuliert wird, um eine Vergleichbarkeit zwischen den optimierten Kopplungsalgorithmen und den untersuchten Algorithmen aus Abschnitt 3.4 herzustellen.

**Parametrierung des Optimierungsproblems**

Wie im vorangegangenen Kapitel erläutert, ist für die vollständige Definition des Optimierungsproblems lediglich eine Abschätzung der Bandbreite des RVPs sowie dessen relativen Grades notwendig. Beides wird für den Zweimassenschwinger aus dessen in Abb. 4.7 in Abschnitt 4.1.3 dargestellten Übertragungsfunktionen abgelesen. Die Bandbreite des Systems ergibt sich zu

$$\omega_b \in [\omega_u, \omega_g] = [0{,}5\frac{rad}{s}, 12\frac{rad}{s}], \tag{4-38}$$

da dessen Amplitudenverstärkung außerhalb dieses Bereichs kleiner eins ist, da die Eigenfrequenzen des untersuchten Systems bei $\omega_1 = 1\,rad/s$ und $\omega_2 = 11\,rad/s$ liegen. Der relative Grad ergibt sich zu $r = 1$, da der Grad des Nennerpolynoms um eins größer als der des Zählerpolynoms ist.

Für die Gewichtungsparameter gelten gemäß der Argumentation aus Abschnitt 4.2.1 die Werte

$$\alpha = 1,\ \beta = 0{,}01 \text{ und } \gamma = 1000. \tag{4-39}$$

Durch numerisches Lösen des so parametrierten Optimierungsproblems werden mit zufälligen Startwerten die Vektoren $\boldsymbol{a}$ und $\boldsymbol{A}$ beispielsweise für drei Kopplungsalgorithmen berechnet, die sich in der Anzahl der verwendeten vergangenen Signalwerten $m = \{3{,}6{,}7\}$ unterscheiden. Die Werte $m = 6$ und $m = 7$ werden gewählt, da sie bei der verwendeten Verzögerung von $k = 3$ denen vom EROS3 bzw. EROS4 entsprechen, und somit eine Vergleichbarkeit herstellt. Der Kopplungsalgorithmus mit $m = 3$ ist wesentlich weniger flexibel, weshalb eine schlechtere Kompensation des Kopplungsfehlers zu erwarten ist (s. Abschnitt 4.2.1).

### Analyse der optimierten Kopplungsalgorithmen im Frequenzbereich

In Abb. 4.9 sind die Bode-Diagramme der drei optimierten Kopplungsalgorithmen sowie die vom EROS3 und EROS4 über das auf die Nyquistfrequenz normierte Verhältnis aus Frequenz und Makroschrittweite dargestellt. Abb. 4.10 zeigt einen vergrößerten Ausschnitt. Das Verhältnis aus Frequenz und Makroschrittweite, für welche die Kopplungsalgorithmen optimiert sind, ergeben sich aus der Bandbreite des RVPs (s. Gl. (4-38)) und der verwendeten Makroschrittweite von $\Delta T = 0{,}02$ zu $[0{,}3\%, 7{,}64\%]$ bezogen auf die Nyquistfrequenz. Es ist zu sehen, dass die Amplitudenverstärkung und die Phasenverschiebung für die beiden optimierten Kopplungsalgorithmen mit $m = 6$ und $m = 7$ in diesem Bereich nahezu neutral bleiben. Dabei ist der Kopplungsalgorithmus mit $m = 7$ nur geringfügig besser als der mit $m = 6$. Die generischen Kopplungsalgorithmen EROS3 und EROS4, die von allen untersuchten generischen Algorithmen den Kopplungsfehler am besten kompensieren (s. Abschnitt 3.4), weisen innerhalb dieser Bandbreite bereits eine größere Amplitudenverstärkung bzw. Phasenverschiebung auf (s. Tab. 4.1). Der optimierte Kopplungsalgorithmus mit $m = 3$ hat innerhalb der Bandbreite ebenfalls schon eine große Amplitudenverstärkung, da er aufgrund der geringen Anzahl von verwendeten Signalwerten nicht die notwendige Flexibilität dafür besitzt (s. Abschnitt 4.2.1).

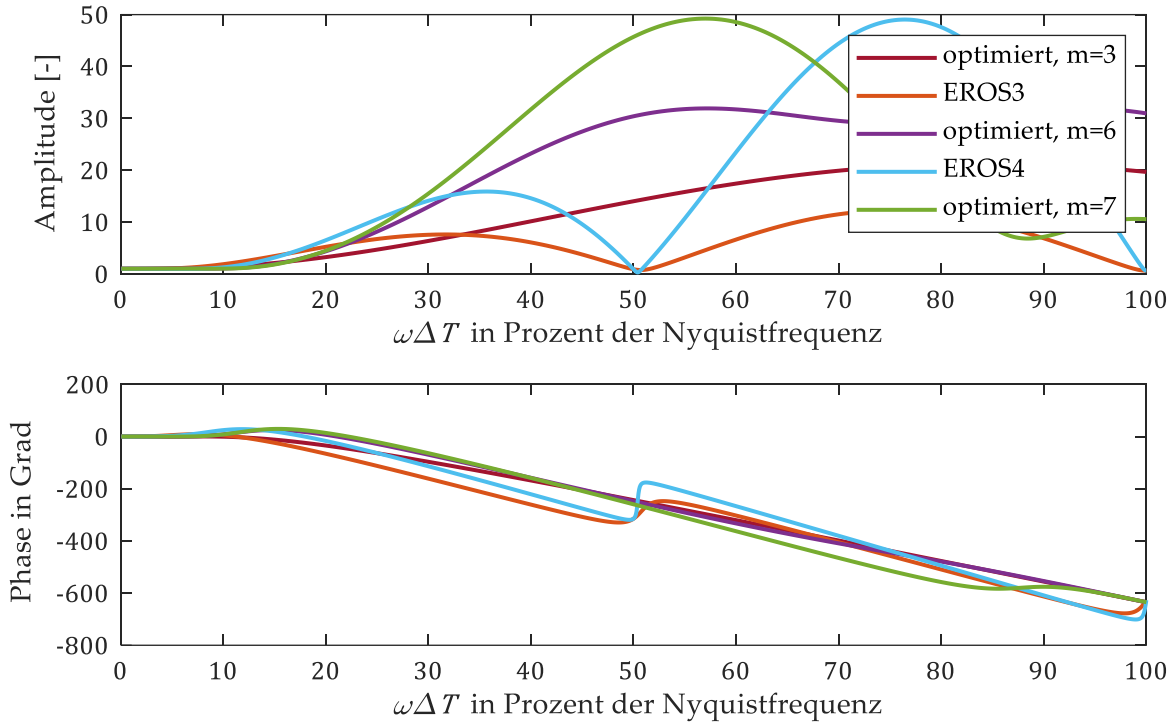


**Abb. 4.9** Bode-Diagramm der optimierten Kopplungsalgorithmen bzgl. des Verhältnisses aus Frequenz und Makroschrittweite

Die geforderte geringe Empfindlichkeit der beiden optimierten Kopplungsalgorithmen mit $m = 6$ und $m = 7$ im Bereich der Bandbreite des RVPs bedingt gemäß des Wasserbetteffekts

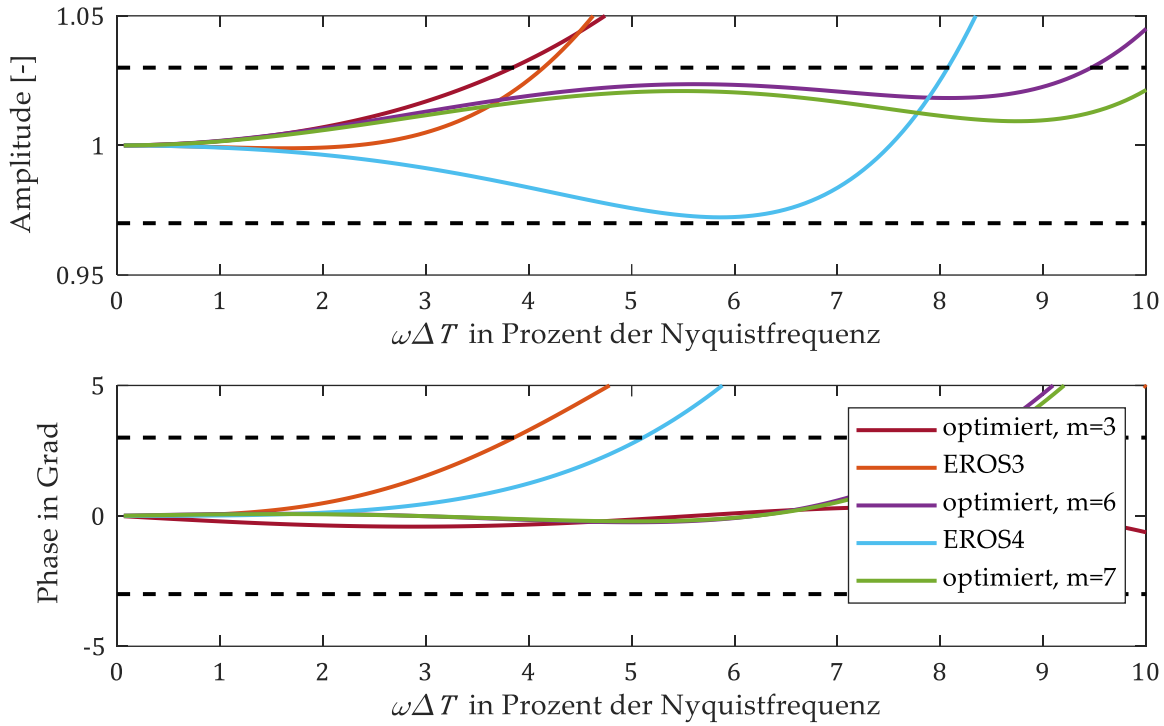


**Abb. 4.10** Bode-Diagramm der optimierten Kopplungsalgorithmen bzgl. des Verhältnisses aus Frequenz und Makroschrittweite (Ausschnitt)

(Janschek, 2010) eine hohe Amplitudenverstärkung in einem anderen Bereich. Wie in Abb. 4.9 gezeigt, haben beide Kopplungsalgorithmen eine hohe Amplitudenverstärkung für Frequenzen oberhalb der Bandbreite, die im Vergleich zu EROS3 und EROS4 noch etwas höher ist, und deren Maximum für ein kleineres Verhältnis aus Frequenz und Makroschrittweite auftritt.

Um den Einfluss der untersuchten Kopplungsalgorithmen auf das Gesamtsystem zu zeigen, ist in Abb. 4.11 das Bode-Diagramm der offenen Kette $G_{RVP}(s)$ für den Zweimassenschwinger inklusive der Algorithmen gezeigt. Nur die optimierten Algorithmen mit $m = 6$ und $m = 7$ halten die Amplitudenverstärkung und Phasenverschiebung im Bereich der zweiten Eigenfrequenz, die im oberen Bereich der Bandbreite liegt, neutral. Die Abweichung von der Referenz ist minimal. Für Frequenzen, die über der zweiten Eigenfrequenz liegen, reicht, wie in der Kostenfunktion des Optimierungsproblems gefordert, die natürliche Dämpfung des Systems aus, um die hohe Amplitudenverstärkung der Kopplungsalgorithmen so zu kompensieren, dass sie unterhalb der kritischen Verstärkung von eins ($0dB$) liegen.

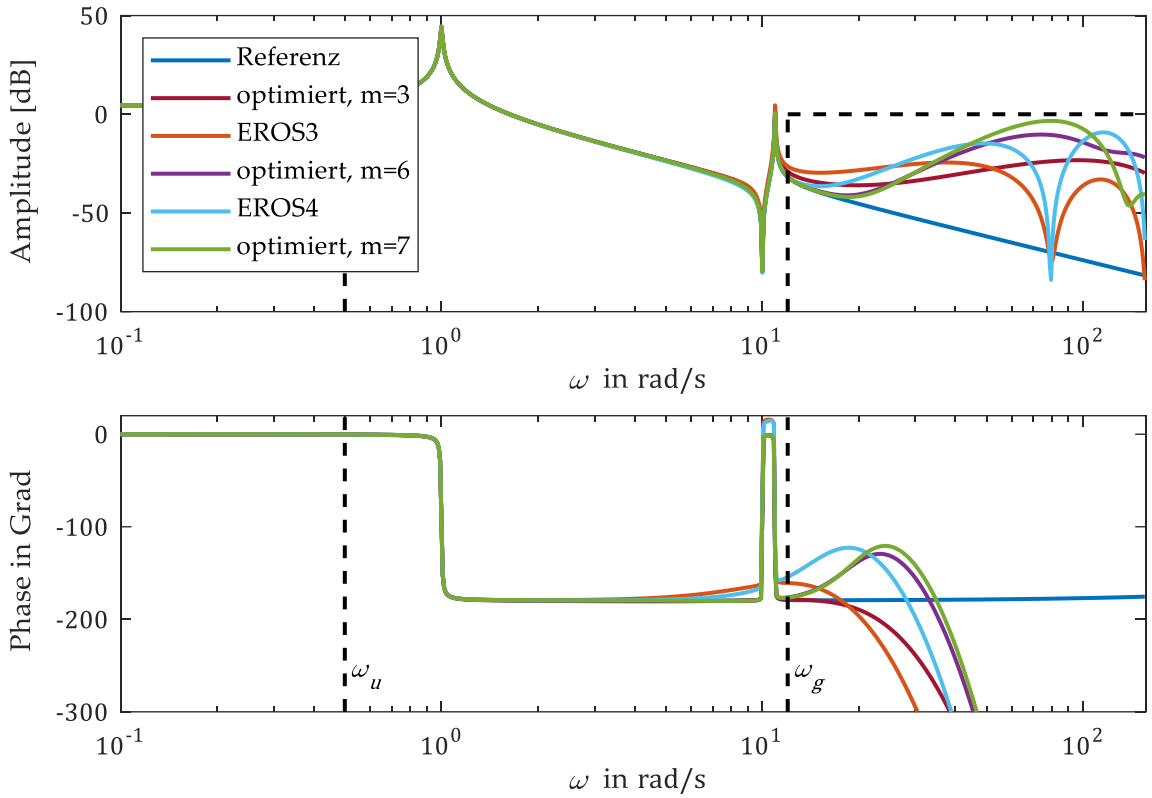


**Abb. 4.11** Bode-Diagramme der offenen Kette unter Verwendung der optimierten Kopplungsalgorithmen mit $k = 3$

Durch Lösung des Optimierungsproblems wird ein Optimum aus den beiden sich bedingenden Effekte aus gewünschtem neutralem Verhalten der optimierten Kopplungsalgorithmen innerhalb der Bandbreite und erlaubter Amplitudenverstärkung außerhalb der Bandbreite gefunden. Dabei hängt es von der definierten Funktion der Kopplungsalgorithmen ab, welche sich in der Anzahl der verwendeten vergangenen Signalwerte unterscheidet, wie nah die optimale Lösung an die Stabilitätsgrenze des Systems kommen kann, um die Amplitudenverstärkung und Phasenverschiebung innerhalb der Bandbreite optimal zu neutralisieren.

Die analog zu Abschnitt 4.1.3 erstellten Nyquist-Diagramme in Abb. 4.12 zeigen, dass die Ortskurve für alle hier untersuchten Kopplungsalgorithmen den kritischen Punkt $(-1, 0)$ nicht umschlingt. Da die Kopplungsalgorithmen alle der allgemeinen Form ARIMA($p$,$d$,0) entsprechen und somit die Übertragungsfunktion der offenen Kette $G_{RVP}(s)$ keine instabilen Pole besitzt, ist das System der Argumentation aus Abschnitt 4.1.3 folgend stabil.

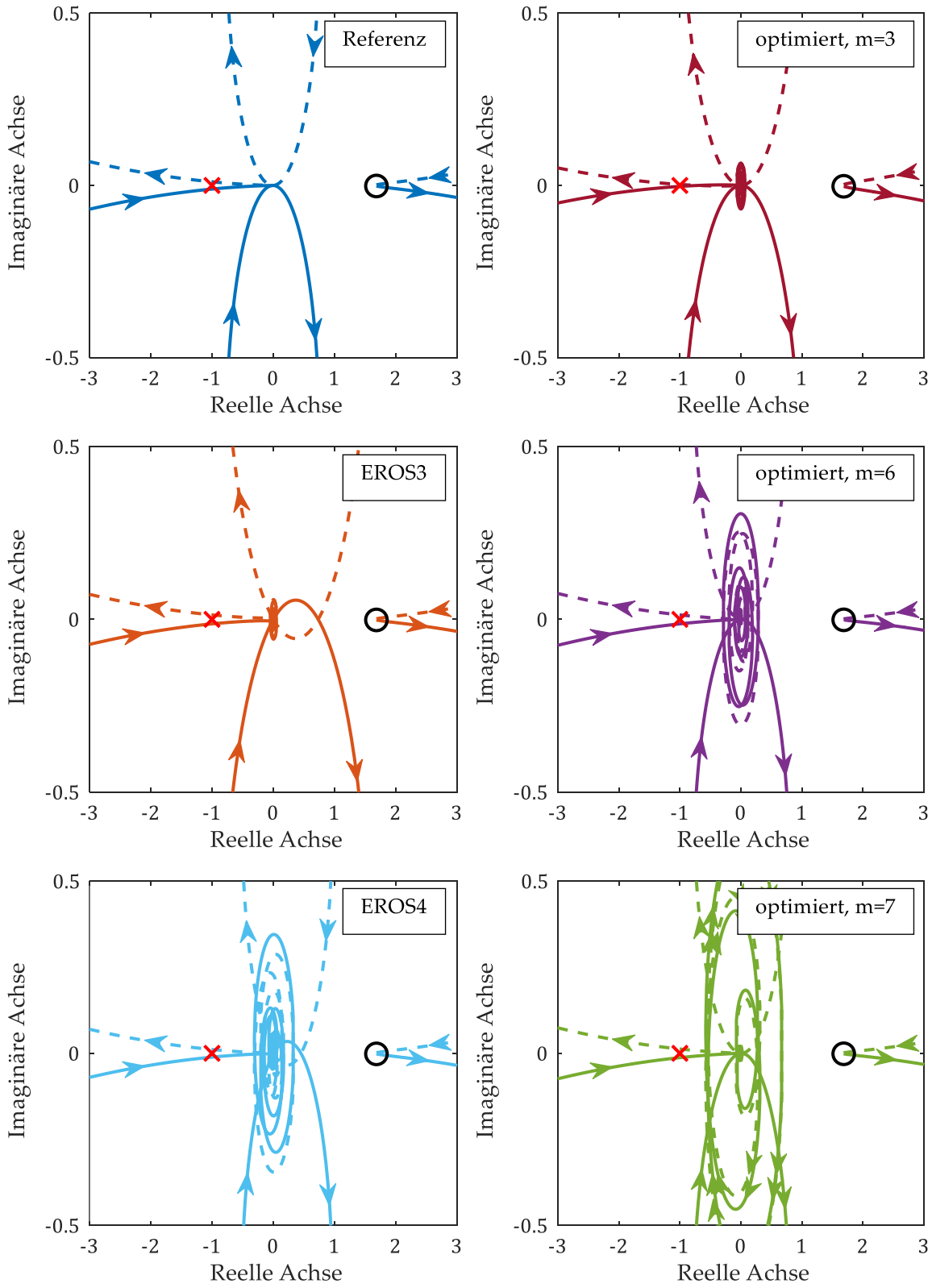


**Abb. 4.12** Nyquist-Diagramme der Übertragungsfunktion der offenen Kette $G_{RVP}(s)$ für die optimierten Kopplungsalgorithmen mit $\Delta T = 0{,}02s$ und $k = 3$. Der kritische Punkt ist mittels eines roten Kreuzes, der Punkt $\omega = 0$ mittels eines schwarzen Kreises markiert.

Die bereits beobachtete gute Kompensation der latenzzeitbedingten Phasenverschiebung im Bereich der zweiten Eigenfrequenz zeigt sich in den Nyquist-Diagrammen an den kleineren Ellipsen, die den dargestellten Ausschnitt nach unten (bzw. für negative Frequenzen nach oben) verlassen. Bei den optimierten Kopplungsalgorithmen liegen diese genau wie bei der Referenz vollständig in der unteren Halbebene und nicht wie bei den generischen Kopplungsalgorithmen aufgrund der nicht kompensierten Phasenverschiebung teilweise in der oberen Halbebene.

Die Strudel der Ortskurve um den Ursprung, die vor allem beim EROS4 und den beiden optimierten Kopplungsalgorithmen mit $m = 6$ und $m = 7$ eine Amplitude nahe eins aufweisen entstehen durch die Amplitudenverstärkung der Kopplungsalgorithmen für größere Frequenzen, die gerade so hoch ist, dass sie vom System gedämpft werden können und nicht den kritischen Punkt umschlingen. Ein Umschlingen des kritischen Punktes würde zur Instabilität führen.

**Vergleich der optimierten Kopplungsalgorithmen in der Simulation des RVPs**

Die in der vorangegangenen Analyse der Kopplungsalgorithmen guten Kompensationseigenschaften bestätigen sich in der Simulation des RVPs, die analog zu Abschnitt 3.4 mit allen in diesem Kapitel untersuchten Kopplungsalgorithmen durchgeführt wird. Eine Quantifizierung des Kopplungsfehlers und ein Vergleich des Verlaufs der Koppelsignale gegenüber der Referenzlösung mittels der Metriken aus Abschnitt 2.4 sind in Tab. 4.3 und Tab. 4.4 dargestellt.

In Tab. 4.3 ist der mittlere absolute Fehler der Koppelsignale bezüglich der Referenzlösung gelistet, die durch Simulation des idealen, verzögerungsfreien Systems entsteht. Der Fehler ist bei Verwendung der optimierten Kopplungsalgorithmen jeweils geringer als beim entsprechenden EROS-Algorithmus mit der gleichen Anzahl der verwendeten vergangenen Signalwerte. Bei dem optimierten Algorithmus mit $m = 6$ ist der Fehler um 12% geringer als beim EROS3 und beim optimierten Algorithmus mit $m = 7$ ist der absolute Fehler gegenüber der EROS4 3% geringer.

**Tab. 4.3** Mittlerer absoluter Fehler bezüglich der Referenzlösung des linearen Zweimassenschwingers für die optimierten Kopplungsalgorithmen

| | MAE - Summe | MAE - $f_k$ | MAE - $x_1$ | MAE - $\dot{x}_1$ |
|---|---|---|---|---|
| optimiert, m=3 | 0,10 | 0,040 | 0,023 | 0,039 |
| EROS3 | 0,050 | 0,024 | 0,0096 | 0,016 |
| optimiert, m=6 | 0,044 | 0,017 | 0,010 | 0,017 |
| EROS4 | 0,038 | 0,017 | 0,0083 | 0,013 |
| optimiert, m=7 | 0,037 | 0,015 | 0,084 | 0,014 |

Die Quantifizierung des summierten Kopplungsfehlers aller Koppelsignale über die Metrik von Sprague und Geers in Tab. 4.4 zeigt deutlich, dass die optimierten Kopplungsalgorithmen den Signalverlauf aller Koppelsignale für die Dauer der Verzögerung von $k = 3$ Makroschritten sehr gut prädizieren. Der kombinierte Fehler nach Sprague und Geers ist beim optimierten Kopplungsalgorithmus mit $m = 6$ um 43% geringer als EROS3 und der optimierte Kopplungsalgorithmus mit $m = 7$ weist einen um 29% geringeren Fehler als EROS4 auf. Sogar der optimierte Algorithmus mit $m = 3$ prädiziert den Signalverlauf nach dieser Metrik besser als EROS3, obwohl dieser weniger vergangene Signalwerte als EROS3 berücksichtigt.

**Tab. 4.4** Summierter Kopplungsfehler mit Sprague und Geers ($\cdot 10^2$) als Distanzmaß für die optimierten Kopplungsalgorithmen

| | $C_{S\&G}$ | $M_{S\&G}$ | $P_{S\&G}$ |
|---|---|---|---|
| optimiert, m=3 | 0,71 | 0,44 | 0,52 |
| EROS3 | 0,81 | 0,20 | 0,73 |
| optimiert, m=6 | 0,46 | 0,41 | 0,17 |
| EROS4 | 0,56 | 0,36 | 0,38 |
| optimiert, m=7 | 0,40 | 0,35 | 0,16 |

Wie bereits in den Bode-Diagrammen Abb. 4.10 und Abb. 4.11 beobachtet, kompensieren die optimierten Kopplungsalgorithmen die Latenzzeit innerhalb der Bandbreite des Systems besser als die generischen Kopplungsalgorithmen, wodurch der Phasenverlauf der offenen Kette den Bereich der Eigenfrequenzen besser dem des idealen gekoppelten Systems entspricht. Dieser Vorteil der optimierten Kopplungsalgorithmen bestätigt sich auch in der Simulation, deren Ergebnisse in Tab. 4.3 Tab. 4.4 gelistet sind. Der mithilfe der Metrik von Sprague and Geers direkt quantifizierbare Phasenfehler ist bei den optimierten Kopplungsalgorithmen deutlich geringer als bei den generischen Kopplungsalgorithmen und maßgeblich für den oben beobachteten geringeren kombinierten Fehler verantwortlich. Dieser Vorteil der optimierten Kopplungsalgorithmen gegenüber den generischen EROS-Algorithmen zeigt sich in Tab. 4.4 im Wert des absoluten Fehlers für das Signal $f_k$. Dieses Koppelsignal enthält hohe Anteile der hochfrequenteren Schwingung bedingt durch die zweite Eigenfrequenz (vgl. Abb. 3.2 in Abschnitt 3.4) und ist Hauptursache für die geringere Abweichung des summierten absoluten Fehlers bzgl. der Referenzlösung unter Verwendung der optimierten gegenüber der generischen Kopplungsalgorithmen.

**Zusammenfassung**

Dieses Beispiel zeigt, dass die Beschreibung des Kopplungsprozesses für RVPen im Frequenzbereich aus Gl. (4-21) zur optimalen Auslegung eines als ARIMA($p$,$d$,0) klassifizierten Kopplungsalgorithmus nutzbar ist. Die optimierten Kopplungsalgorithmen minimieren die Kostenfunktion aus Gl. (4-21), wodurch sie auf das zu koppelnde System angepasst sind und Latenz-

zeiten besser kompensieren als die generischen Kopplungsalgorithmen. Dabei ist nur sehr wenig Systemwissen nötig, welches allein durch eine Fourier-Transformation der Koppelsignale gewonnen werden kann, wodurch dieses Verfahren auch für reale RVPen anwendbar ist (s. Abschnitt 6.2).

## 4.3 Diskussion der Analysemethodik

Die Tatsache, dass das dynamische Systemverhalten vieler virtueller Prototypen und praktisch aller realen Prototypen nichtlinear ist, schränkt die Stabilitätsanalyse aus Abschnitt 4.1 für reale RVPen ein. Aussagen über die Stabilität von RVPen unter dem Einfluss von Latenzzeiten und Kopplungsalgorithmen können für nichtlineare Prototypen mithilfe von linearen Modellen der Prototypen getroffen werden, welche beispielsweise durch eine Systemidentifikation mit gegebenenfalls anschließender Linearisierung erstellt werden. Die Ungenauigkeit dieser mathematischen Modelle erhöht dabei die Unsicherheit der Stabilitätsanalyse.

Die Methode zur optimalen Auslegung der Kompensation aus Abschnitt 4.2 ist dagegen auch bei RVPen, deren Prototypen nichtlinear sind uneingeschränkt möglich, da nur die Bandbreite des Koppelsignale und der relative Grad (bzw. die Steigung des Amplitudengangs für hohe Frequenzen) bekannt sein muss. Beides kann beispielsweise mittels einer Fourier Analyse der Koppelsignale geschätzt werden. Liegen im Koppelsignal der nichtlinearen Prototypen keine Diskontinuitäten (für Definition s. Abschnitt 3.1) vor, sind die identifizierten linearen Kopplungsalgorithmen genau wie bei linearen Prototypen sofort anwendbar. Führen die Nichtlinearitäten der gekoppelten Prototypen dagegen zu vereinzelten Diskontinuitäten im Koppelsignal, wird eine zusätzliche Methodik benötigt, um diese vor der Anwendung des Kopplungsalgorithmus zu erkennen, korrekt zu reagieren und dadurch den Kopplungsfehler gering zu halten (s. Abschnitt 5.3).

Sowohl in der Stabilitätsanalyse als auch bei der optimalen Auslegung der Kopplungsalgorithmen ist eine konstante Verzögerung von $k$ Makroschritten für den Datenaustausch zwischen den Prototypen angenommen. Zur Abbildung variabler Verzögerungszeiten kann die Analysemethode erweitert werden, indem diese basierend auf Messungen mittels einer Wahrscheinlichkeitsverteilung beschrieben wird. Als Verteilungsfunktion bietet sich beispielsweise eine zeitverschobene Erlang-Verteilung an, da diese die Eigenschaften von variablen Latenzen in Netzwerken gut abbildet (Suksomboon et al., 2017) und außerdem in den Frequenzbereich transformierbar ist (Sztrik, 2016). Eine über die zeitverschobene Erlang Verteilung beschriebene variable Verzögerungszeit könnte somit direkt als $G_l(s)$ (s. Gl. (4-13)) in die Analysemethode aus Abschnitt 4.1 aufgenommen werden. Allerding reicht die Betrachtung konstanter Verzögerungen für alle in dieser Arbeit betrachteten Beispiele von RVPen aus, da dort die Variabilität der Latenzzeiten sehr gering ist und dadurch die Verzögerungszeiten durch eine geschickte Wahl der Makroschrittweite vereinheitlicht werden kann (s. Abschnitt 5.1).

Eine weitere Einschränkung der Analyse in diesem Kapitel ist, dass für die Beschreibung des Kopplungsprozesses im Frequenzbereich nur zwei gekoppelte Systeme mit jeweils nur einem Koppelsignal pro Richtung betrachtet werden. Für eine exakte Analyse der Stabilitätseigenschaften von RVPen mit mehr als zwei gekoppelten Prototypen und mehreren Koppelsignalen ist das nicht ausreichend. Allerdings reicht es aus, um die dynamisch kritischste Kopplung zu analysieren, welche in der Regel aus zwei entgegen gerichteten physikalischen Koppelsignalen bestehen, die zusammen einen Energiefluss beschreiben (power bonds). Eine optimale Auslegung der Kopplungsalgorithmen mit der Methodik aus Abschnitt 4.2 ist für größere Netzwerke von Prototypen jedoch uneingeschränkt möglich, da für jedes Koppelsignal einzeln die Bandbreite geschätzt und ein passender Kopplungsalgorithmus ausgelegt wird.

# 5 Aufbau und Überwachung Real-Virtueller Prototypen

Aufgrund der angebundenen realen Prototypen und der zur Datenübertragung meistens verwendeten unzuverlässigen Netzwerkkommunikation unterliegen RVPen der Echtzeit und sind im Allgemeinen nicht deterministisch (vgl. Abschnitt 2.2.2). Außerdem ist das Systemverhalten der gekoppelten Prototypen nicht im Detail bekannt, weshalb potenziell jedes Koppelsignal Diskontinuitäten enthalten kann. Daher ist für den effizienten Betrieb realer RVPen eine Latenzzeitkompensation von stetigen Signalen bzw. von Signalen mit vorhersehbaren Diskontinuitäten, wie sie in Kap. 3 und Kap. 4 entwickelt und detailliert analysiert wurde, allein nicht ausreichend. In diesem Kapitel werden Methoden entwickelt, um diese zusätzlichen Herausforderungen bei der Verwendung von Kopplungsalgorithmen in RVPen zu beherrschen.

Bereits beim Aufbau und der Initialisierung eines RVPs, welche unter Echtzeitbedingung stattfindet, sind Herausforderungen zu bewältigen. Zum einen muss der Datenaustausch der gekoppelten Prototypen parametriert werden, indem das Verhältnis aus Makroschrittweite $\Delta T$ und der Anzahl der Makroschritte Verzögerung $k$ passend zur Latenzzeit $\tau$ und dessen Variabilität korrekt gewählt wird (vgl. Abschnitt 2.2.3). Zum anderen muss vor der Ausführung eines RVPs festgelegt werden in welcher Reihenfolge die Prototypen zu starten sind, welchen Zeitpunkt jeder Prototyp als Startpunkt seiner Ausführung definiert und wie diese Zeitpunkte und damit die Zeitraster der verschiedenen gekoppelten Prototypen zueinanderstehen.

Neben der Vorbereitung der Ausführung von RVPen wird in diesem Kapitel die online Überwachung von RVPen während des Betriebs diskutiert. Eine Überwachung ist zum einen notwendig, um einen sicheren Betrieb der realen Prototypen wie beispielsweise Motor- oder Batterieprüfständen zu gewährleisten und zum anderen um Fehler, welche die Aussagekraft der Ergebnisse beeinträchtigen, direkt festzustellen und gegebenenfalls die Ausführung des RVPs abzubrechen und neu zu starten. Auf diese Weise kann besonders bei langen Simulationen Zeit und somit Geld gespart werden. Die meisten Fehlerquellen wie beispielsweise Paketverluste über einen langen Zeitraum in der Kommunikation oder Eingangssignale, die den sicheren Betrieb von realen Prototypen gefährden, können mit Standardmaßnahmen wie einer Überwachung der Netzwerkverbindung oder einer individuellen Statusüberwachung für jeden angebundenen realen Prototypen realisiert werden. Eine weitere mögliche Fehlerquelle, die das Ergebnis von RVPen verfälschen kann, sind Kopplungsalgorithmen, die außerhalb ihres Gültigkeitsbereichs verwendet werden und auf diese Weise zu einem Kopplungsfehler

führen. Zur online Erkennung dieser Fehler wird daher in diesem Kapitel die Idee von Stettinger et al. (2017) zur Bestimmung des Kopplungsfehlers über die Extrapolationsgüte der Kopplungsalgorithmen diskutiert und erweitert. Außerdem werden die Parameter der definierten Toleranzfunktion zur Bewertung der Extrapolationsgüte für jeden Kopplungsalgorithmus identifiziert.

Weiterhin wird in diesem Kapitel eine Methodik vorgeschlagen, um die Stetigkeit der Koppelsignale während der Ausführung zu überprüfen. Im Gegensatz zur online Bestimmung der Güte, die jeweils nur für vergangene Extrapolationsschritte möglich ist, wird dabei jedes Koppelsignal vor der Anwendung des Kopplungsalgorithmus auf Diskontinuitäten überprüft, um Fehler bei der Verwendung von Kopplungsalgorithmen mit großer Amplitudenverstärkung hoher Frequenzen wie EROS3 oder EROS4 (s. Kap. 3) zu verhindern.

Diese online Erkennung von Diskontinuitäten in Koppelsignalen erlaubt es, während der Ausführung eines RVPs den Kopplungsalgorithmus umzuschalten, bevor er auf eine Diskontinuität im Signal angewandt wird. Dadurch lassen sich Kopplungsalgorithmen, die aufgrund einer großen Amplitudenverstärkung hoher Frequenzen nicht für die Extrapolation von Signalen mit Diskontinuitäten geeignet sind, trotzdem auf diese Signale anwenden, indem an der Diskontinuität kurzfristig auf einen Kopplungsalgorithmus umgeschaltet wird, der hohe Frequenzanteile im Signal infolge der Diskontinuität nicht verstärkt. Bei gleichzeitiger Überwachung der Extrapolationsgüte ist somit ein effizienter Einsatz der Kopplungsalgorithmen bei gleichzeitiger Gewährleistung eines sicheren Betriebs des RVPs möglich. In der Co-Simulation (vgl. Abschnitt 2.1.1) gibt es ebenfalls Methodiken wie beispielsweise CHOPtrey von Ben Khaled-El Feki et al. (2017), die online aktuelle Signalverhalten erkennen und die Extrapolationsmethode entsprechend kurzfristig anpassen.

In Abschnitt 5.1 wird zunächst die Vorbereitung der Ausführung von RVPen diskutiert bevor in Abschnitt 5.2 die Methodik von Stettinger et al. (2017) zur Bestimmung der Extrapolationsgüte aufgegriffen, erweitert und untersucht wird. Abschnitt 5.3 schlägt eine Methodik zur online Erkennung von Diskontinuitäten vor und diskutiert Möglichkeiten zum sinnvollen Umschalten von Kopplungsalgorithmen während des Betriebs.

## 5.1 Vorbereitung der Ausführung von Real-Virtuellen Prototypen

Im Folgenden werden verschiedene Aspekte diskutiert, welche die Vorbereitungsphase von RVPen betreffen und für die erfolgreiche Verwendung der in dieser Arbeit entwickelte Kopplungsmethodik relevant sind. Diese sind zum einen die Parametrierung der Datenübertragung durch die Wahl der Makroschrittweite, zum anderen die Zeitsynchronisation der verschiedenen Prototypen und außerdem die Definition der verschiedenen Zeitraster der gekoppelten Prototypen zueinander, welche sich aus dem Startmechanismus des RVPs ergeben.

### 5.1.1 Wahl der Makroschrittweite

Bevor ein RVP ausgeführt werden kann, muss die Makroschrittweite $\Delta T$ festgelegt werden. Sie definiert in welchem Zeitraster die Koppelsignale der gekoppelten Prototypen ausgetauscht werden. Wie bereits in Gl. (2-11) in Abschnitt 2.2.3 gezeigt, ist die Makroschrittweite bei RVPen konstant und es gibt es einen direkten Zusammenhang zwischen der Makroschrittweite $\Delta T$ und der Anzahl der Makroschritte Verzögerung $k$ abhängig von der variablen Latenzzeit $\tau(t)$. Dieser ergibt sich zum Zeitpunkt $t_n$ für den Datenaustausch zweier gekoppelter Prototypen zu

$$k_n = \min\{m \in \mathbb{N} \mid m \geq \tfrac{\tau_n}{\Delta T}\} \text{ mit } \tau_n > 0. \tag{5-1}$$

Durch das antiproportionale Verhältnis zwischen $\Delta T$ und $k$ führt, im Unterschied zur Co-Simulation, eine gegen null gehende Makroschrittweite nicht automatisch zu einem gegen null gehenden Kopplungsfehler. Allerdings verringert eine gegen null gehende Makroschrittweite den Abtastungsfehlers (s. Abschnitt 4.1), wodurch sich die tatsächliche Verzögerung $k\Delta T$ der Koppelsignale an die physikalisch bedingte Latenzzeit $\tau(t)$ annähert. Daher ist die Makroschrittweite in erster Linie möglichst klein zu wählen. Allerdings macht es keinen Sinn die Makroschrittweite der Datenübertragung kleiner zu wählen als die kleinste Mikroschrittweite der gekoppelten Prototypen, da die übertragenden Daten ansonsten teilweise redundant sind. Die Mikroschrittweite $\delta T$ realer Prototypen und echtzeitfähiger virtueller Prototypen ist in der Regel konstant und abhängig deren Systemverhaltens so gewählt, dass keine Aliasing-Effekte auftreten.

Somit gilt als erster Richtwert für die Wahl der Makroschrittweite von RVPen

$$\Delta T = \min(\delta T). \tag{5-2}$$

In typischen RVPen aus der Literatur ergibt sich nach dieser Regel ein $k < 10$, sodass die dort verwendeten Kopplungsalgorithmen auf diesen Wertebereich von $k$ spezialisiert sind (Stettinger et al., 2013, Schreiber et al., 2018, Tranninger et al., 2016, Stettinger et al., 2017).

Je nach Aufbau des RVPs und der verwendeten Kopplungsmethodik kann es in manchen Fällen sinnvoll sein von dieser Regel abzuweichen und eine größere Makroschrittweite zu wählen. Dabei sollte die größere Makroschrittweite immer ein Vielfaches von $\min(\delta T)$ sein, um Fehler durch ungünstige Verschiebung der verschiedenen Zeitraster zueinander zu vermeiden. Eine größere Makroschrittweite ist zum einen zu wählen, wenn aus der nach Gl. (5-6) gewählten Makroschrittweite ein $k \geq 10$ resultiert. Hier lässt sich durch beispielsweise eine Verdopplung der Makroschrittweite die Menge an übertragenen Daten halbieren sowie die Rechenzeit der Kopplungsmethodik pro Makroschritt verringern, wenn ein iterativer Kopplungsalgorithmus wie der SMB oder das modellbasierte Kopplungselement (s. Abschnitt 3.3)

verwendet wird. In diesem Fall sollte aber zunächst durch eine Analyse des Kopplungsprozesses (s. Kap. 4) überprüft werden, ob der RVP mit einer Verzögerung von mehr als zehn Mikroschritten eines Prototyps unter Einsatz eines Kopplungsalgorithmus überhaupt noch stabilisierbar ist, oder ob für einen erfolgreichen Betrieb des RVPs technische Lösungen für eine geringere Latenzzeit der Datenübertragung gefunden werden müssen.

Ein weiterer Grund die Makroschrittweite höher als den in Gl. (5-2) empfohlenen Wert zu wählen ist, dass sich dadurch eine gegebenenfalls auftretende Variabilität der Latenzzeit $\tau(t)$ nicht auf die tatsächliche Verzögerung $k\Delta T$ auswirkt. Liegt zum Beispiel der Mittelwert der Latenzzeit zweier gekoppelter Prototypen bei $\bar{\tau} = 0{,}03s$ und der Jitter im Bereich $\pm\, 0{,}005s$ würde bei einer Makroschrittweite von $\Delta T = 0{,}01s$ die Anzahl der Makroschritte Verzögerung zwischen $k = 3$ und $k = 4$ wechseln. Unter Verwendung einer Zeitsynchronisation (s. Abschnitt 5.1.2) und mit einer online Berechnung der Latenzzeit (s. Abschnitt 6.1) wäre dies bei der Ausführung des RVPs beherrschbar, würde aber eine vorangehende Analyse des Kopplungsprozesses und die optimale Auslegung des Kopplungsalgorithmus einschränken (s. Abschnitt 4.3). Daher ist es in einem solchen Fall sinnvoll die Makroschrittweite zu erhöhen, um die Anzahl der Makroschritte Verzögerung konstant zu halten. Im gegebenen Beispiel ergibt sich durch eine Erhöhung der Makroschrittweite auf $\Delta T = 0{,}02s$ konstant $k = 2$, wodurch eine Analyse des Kopplungsprozesses nach Kap. 4 uneingeschränkt möglich ist.

### 5.1.2 Zeitsynchronisation

Während der Ausführung von RVPen, führt jeder der gekoppelten Prototypen zeitlich gesteuerte Abläufe wie beispielsweise die Messung von Signalen im Raster der Makroschrittweite aus. Damit diese Abläufe zeitlich möglichst korrekt ablaufen, greift jeder Prototyp auf die Echtzeituhr des jeweiligen Simulations-PCs bzw. des Steuerungssystems des Prüfstands zu, welche sich typischerweise über das Network Time Protocol (NTP) mit einem Zeitserver synchronisieren (Mills et al., 2010). Diese kontinuierliche Zeitsynchronisation mit einem Zeitserver reduziert den Gangfehler der Echtzeituhren auf Werte, die viele Größenordnungen unter den Zeitrastern von RVPen sind. Diese liegen üblicherweise im Bereich von Millisekunden, wodurch eine zeitliche Drift keine Fehlerquelle für RVPen ist.

Allerdings ist für manche Arten der Ausführung von RVPen (s. Abschnitt 5.1.3) und auch aus praktischen Gründen wie beispielsweise dem Vergleich von Ergebnissen, die während der Ausführung eines RVPs von verschiedenen Prototypen des RVPs gespeichert werden, eine absolute Genauigkeit der Echtzeituhren aller Prototypen relativ zueinander notwendig. Der Fehler dieser Zeitsynchronisation sollte Größenordnungen kleiner als die Makroschrittweite sein, um die Ergebnisse des RVPs nicht nennenswert zu beeinflussen. Diese geforderte absolute Genauigkeit wird nicht erreicht, wenn sich jeder Prototyp eigenständig mittels NTP mit

einem Zeitserver synchronisiert, da je nach Ort der räumlich verteilten Prototypen verschiedene Zeitserver als Basis gewählt werden und die Synchronisation mittels NTP hierarchisch stattfindet und nicht jede Echtzeituhr direkt mit dem Zeitserver spricht, wodurch sich Fehler aufsummieren.

Zur Erreichung einer hohen absoluten Genauigkeit der Zeitsynchronisation ist daher eine direkte Synchronisation der PCs der gekoppelten Prototypen notwendig. Rautenberg et al. (2023) stellen dies ebenfalls fest und fordern daher, dass eine entsprechende Zeitsynchronisation in die Spezifikation des Distributed-Co-Simulation-Protocols (DCP) (s. Abschnitt 1.1.3) aufgenommen wird. Krammer, Ferner und Watzenig (2019) stellen ein entsprechendes Konzept vor, welches noch nicht in die Spezifikation aufgenommen wurde. Sie schlagen vor, den Standard IEEE 1588 Precision Time Protocol (PTP) zu verwenden, mit dem eine direkte Zeitsynchronisation im unteren Mikrosekunden Bereich möglich ist (Eidson und Lee, 2002).

### 5.1.3 Start der Ausführung

Die Beteiligung mehrerer räumlich verteilter realer und virtueller Prototypen erschwert den Start der Ausführung von RVPen. Eine Herausforderung besteht darin, dass alle Prototypen zur selben Zeit bereit für die Ausführung sein müssen. Bei virtuellen Prototypen umfasst diese Vorbereitung zum Beispiel das Starten der Simulationswerkzeuge und das Lösen der Anfangsbedingung der Simulationsmodelle, während bei realen Prototypen der entsprechende Prüfstand beispielsweise in einer zeitaufwändigen Aufwärmphase in die vordefinierten Betriebsbedingungen gebracht werden muss. Die Spezifikation des DCP (s. Abschnitt 1.1.3) definiert in einem Zustandsautomaten mehrere Zustände, die alle gekoppelten Prototypen vor dem Start der Ausführung eines RVPs gemeinsam durchlaufen müssen, um ein gleichzeitiges Erreichen der jeweiligen Startbedingungen sicherzustellen (The Modelica Association, 2019).

Für die Anwendung der in dieser Arbeit entwickelten Kopplungsmethodik spielt das Erreichen der Startbedingung eine untergeordnete Rolle. Relevanter ist die genaue Definition des Startzeitpunkts $t = 0$ jedes einzelnen der gekoppelten Prototypen, da sich daraus definiert, ob und wie die verschiedenen Zeitraster der Prototypen relativ zueinander verschoben sind. Dies ergibt sich direkt aus dem Mechanismus wie ein RVP gestartet wird. Im Folgenden werden zwei verschiedene Möglichkeiten zur Bestimmung des Startzeitpunkts analysiert:

**Variante 1: Gleichzeitiger Start aller Prototypen**

Der gleichzeitige Start aller Prototypen ist der Normalfall, welcher in der DCP-Spezifikation beschrieben ist. Die Ausführung des RVPs entspricht dabei exakt der Theorie aus Abschnitt 2.2.3. Der Anwender des RVPs definiert einen bestimmten absoluten und in der Zukunft liegenden Startzeitpunkt $t_{start}$ und teilt diesen allen Prototypen mit. Ist der Startzeitpunkt erreicht, starten alle Prototypen gleichzeitig ihre Ausführung und beginnen in dem

durch die Makroschrittweite definierten Zeitraster ihre Ausgangswerte zu versenden. Ihre Eingangswerte erhalten sie von den anderen Prototypen im gleichen Zeitraster aber entsprechend der jeweiligen Anzahl an Makroschritten Verzögerung $k$ Zeitschritte später.

Da bei Verwendung dieses Startmechanismus die Zeitraster aller Prototypen gleichzeitig ablaufen und nicht zueinander verschoben sind, ist er flexibel für RVPen mit beliebig vielen Prototypen anwendbar, welche untereinander stark vernetzt sind. Weiterhin ist die in dieser Arbeit entwickelte Kopplungsmethodik in dieser Variante direkt einsetzbar, indem sie die entsprechende Latenzzeit in jedem Koppelsignal kompensiert.

Allerdings erfordert dieser Startmechanismus eine direkte Zeitsynchronisation zwischen den Prototypen, deren Genauigkeit um mindestens eine Größenordnung unterhalb der Makroschrittweite liegen sollte, da ansonsten der gleichzeitige Start der Prototypen nicht mit ausreichender Genauigkeit möglich ist (s. Abschnitt 5.1.2). Außerdem stellt die Zeitsynchronisation sicher, dass die für die Anwendung der Kopplungsmethodik notwendige Berechnung der Latenzzeit zwischen den Prototypen und pro Übertragungsrichtung korrekt durchgeführt wird. Diese kann bei Verwendung dieses Startmechanismus aus der Differenz der mitgesendeten Sendezeit eines sendenden Prototyps $A$ und der Empfangszeit eines empfangenden Prototyps $B$ zu

$$k_{A\to B} = \min\{m \in \mathbb{N} \mid m \geq \frac{\tau_{A\to B}}{\Delta T}\} \text{ mit } \tau_{A\to B} = t_{B,\,\text{empfangen}} - t_{A,\,\text{gesendet}} \tag{5-3}$$

bestimmt werden und ist somit nur korrekt, wenn beide Prototypen zur Messung der Zeiten dieselbe absolute Zeitbasis verwenden (s. Gl. (2-11) in Abschnitt 2.2.3).

**Variante 2: Prototypen starten kurz nacheinander**

Eine weitere Möglichkeit den Start der Ausführungen eines RVPs zu realisieren, besteht darin, dass der Anwender einen Prototyp $A$ startet und den Startbefehl zeitgleich von Prototyp $A$ an die anderen Prototypen sendet. Auf diese Weise ist die Zeitbasis eines der nachfolgenden Prototypen $B$ um die Latenzzeit verschoben, welche der Startbefehl für die Übermittlung benötigt hat. Wenn die Latenzzeit zwischen den Prototypen $\tau_{A\to B}$ konstant ist wird auch die nachfolgende Datenübertragung in der Richtung $A \to B$ während der Ausführung genau um die Verschiebung der Zeitbasis verzögert sein. Dadurch wirkt es so als gäbe es keine Verzögerung in der Übertragungsrichtung $A \to B$, da wenn Prototyp B zu seinem Zeitpunkt $t_n$ eine Berechnung startet bereits die Ergebnisse des Prototyps A von dessen Zeitpunkt $t_n$ vorliegen. Für die Koppelsignale, die in diese Richtung ausgetauscht werden, ist daher keine Kopplungsmethodik zur Kompensation von Latenzzeiten nötig. In der Übertragungsrichtung $B \to A$ dagegen wirkt in diesem Fall die Summe der tatsächlichen Verzögerungen in beide Richtungen und muss somit an den Eingängen des Prototyps $A$ durch Einsatz der Kopplungsmethodik kompensiert werden.

Neben der Tatsache, dass nicht für jedes Koppelsignal eine Instanz der Kopplungsmethodik verwendet werden muss, ist ein weiterer Vorteil dieser Variante, dass keine direkte Zeitsynchronisation der Prototypen untereinander notwendig ist. An den Eingängen des Prototyps A kann die zu kompensierende Umlaufzeit

$$k_{A\to B\to A} = \min\{m \in \mathbb{N} \mid m \geq \frac{\tau_{RTT}}{\Delta T}\} \text{ mit } \tau_{RTT} = t_{A,\text{ empfangen}} - t_{A,\text{ gesendet}} \tag{5-4}$$

ohne eine Abhängigkeit von Zeitmessungen des Prototyps $B$ bestimmt werden.

Diese Variante der Ausführung ist allerdings nicht für alle RVPen verwendbar. Sind mehr als zwei Prototypen beteiligt funktioniert sie nur, wenn alle nachfolgenden Prototypen nur mit dem startenden Prototyp gekoppelt sind und untereinander keine weiteren für das Systemverhalten relevante Signale austauschen, da die Verschiebung der Zeitraster der nachfolgenden Prototypen zueinander unbekannt ist. Weiterhin muss wie oben erwähnt die Latenzzeit zwischen den Prototypen konstant sein.

**Erweiterung durch optimale Ausrichtung der Zeitraster**

Die vorangegangene Variante kann erweitert werden, damit sie auch für RVPen mit variabler Latenzzeit in der Datenübertragung zwischen den Prototypen verwendbar ist, indem ein im Rahmen dieser Arbeit patentiertes Verfahren zur Synchronisation von realen und virtuellen Prototypen angewandt wird (Mikelsons et al., 2023). Der Kern des Verfahrens besteht darin die Umlaufzeit zwischen den Prototypen und dessen Variabilität vor Beginn der Ausführung zu messen und daraus zu berechnen, wie die Zeitraster der Prototypen gegenseitig verschoben sein müssen, damit die Latenzzeit für jede Richtung nach der Diskretisierung mit der Makroschrittweite konstant ist.

Statt die nachfolgenden Prototypen sofort zu starten, sobald diese den Startbefehl des zuerst gestarteten Prototyps empfangen, beginnen sie ihre Ausführung nun nach Erhalt des Startbefehls in dem durch das Verfahren vorher definierten Zeitraster. Auch mit der Erweiterung funktioniert diese Variante ohne eine direkte Zeitsynchronisation zwischen den Prototypen, da aufgrund der optimalen Ausrichtung der Zeitraster keine absoluten Zeitpunkte ausgetauscht werden müssen (Mikelsons et al., 2023).

## 5.2 Online Bewertung der Extrapolationsgüte

Eine mögliche Fehlerquelle, die das Systemverhalten von RVPen signifikant beeinflusst und somit deren Ergebnisse verfälschen kann, sind Kopplungsalgorithmen, die auf Signale angewandt werden für deren Eigenschaften sie nicht geeignet sind. Daher muss die Verwendung von Kopplungsalgorithmen überwacht werden. In diesem Kapitel wird aufbauend auf der Arbeit von Stettinger et al. (2017) ein Verfahren vorgeschlagen, um mittels der Extrapolations-

güte zu überwachen, dass der Frequenzbereich eines Koppelsignals innerhalb des Gültigkeitsbereichs des angewandten Kopplungsalgorithmus liegt. Wird ein Kopplungsalgorithmus außerhalb seines Gültigkeitsbereichs verwendet, kann der dabei entstehende Kopplungsfehler, wie in Abschnitt 4.1.3 gezeigt, zur Instabilität von RVPen führen. Für den sicheren und effizienten Einsatz von Kopplungsalgorithmen ist es notwendig, diese Überwachung der Güte online zur Laufzeit des RVPs durchzuführen. Dadurch kann zur Zeitersparnis das aktuelle Experiment sofort abgebrochen und neugestartet werden, sobald eine Verletzung der Güte vorliegt. Eine nachträgliche Überprüfung der Güte bietet keine Vorteile, da diese aufgrund des nicht deterministischen Verhaltens von RVPen nur bedingt eine Aussage für zukünftige Ausführungen des RVPs hätte und daher nach jedem Durchlauf wiederholt werden müsste.

Die Herleitung der Methodik zur online Bewertung der Güte eines Kopplungsalgorithmus beginnt mit der Betrachtung des verbleibenden Kopplungsfehlers während der Ausführung eines RVPs. Abgeleitet aus Abschnitt 2.1.1 ist der Kopplungsfehler

$$\boldsymbol{e}(t) = d\big(\boldsymbol{y}(t), \hat{\boldsymbol{y}}(t)\big) \text{ mit } t_n \leq t < t_{n+1} \tag{5-5}$$

der Unterschied zwischen den tatsächlichen Signalverläufen $\boldsymbol{y}(t)$ und den mittels eines Kopplungsalgorithmus geschätzten Verläufen $\hat{\boldsymbol{y}}(t)$, der mit einem geeigneten Distanzmaß $d$ kontinuierlich bestimmt wird. Da in dieser Arbeit für jedes Koppelsignal ein unabhängiger Kopplungsalgorithmus zur Kompensation der Latenzzeiten verwendet wird, ist auch der verbleibende Kopplungsfehler für jedes Koppelsignal einzeln berechenbar. Weiterhin ist eine exakte Bestimmung des Kopplungsfehlers nur an den Makrozeitpunkten möglich, da nur zu diesen Zeitpunkten der tatsächliche Signalwert von den gekoppelten Prototypen kommuniziert wird. Der zu bewertende Kopplungsfehler vereinfacht sich dadurch auf den Fehler

$$e_{ext,n} = |y_n - \hat{y}_n|, \tag{5-6}$$

der nach der Extrapolation eines Koppelsignals mittels eines Kopplungsalgorithmus verbleibt. Dabei ist zu beachten, dass dieser Fehler jeweils nur $k$ Zeitschritte rückwirkend berechnet werden kann, da der korrekte Signalwert $y_n$ aufgrund der Latenzzeit erst zum Zeitpunkt $t_{n+k}$ vorliegt. Der Fehler $e_{ext,n}$ bezieht sich somit auf die Extrapolation von $\hat{y}_n$ zum Zeitpunkt $t_n$, kann aber erst zum Zeitpunkt $t_{n+k}$ berechnet werden, wenn der korrekte Signalwert $y_n$ vorliegt (s. Gl. (2-14) in Abschnitt 2.2.3).

Das zu entwickelnde Verfahren zur online Bewertung des in Gl. (5-6) definierten Extrapolationsfehlers und der daraus resultierenden Extrapolationsgüte der Kopplungsalgorithmen soll folgende Anforderungen erfüllen:

1. **Geringer Berechnungsaufwand.** Wie oben erläutert soll die Extrapolationsgüte online während der Ausführung eines RVPs bestimmt werden. Um dies ohne eine Gefährdung der Echtzeitfähigkeit des Systems zu erreichen, muss der notwendige Berechnungsaufwand zur Bewertung des Extrapolationsfehlers möglichst gering sein. Außerdem darf sich der Berechnungsaufwand nicht stark ändern, wodurch beispielsweise Verfahren mit einer unbekannten Anzahl an Iterationen ausgeschlossen werden.

2. **Allgemeine Anwendbarkeit.** Das Verfahren soll allgemein auf alle Koppelsignale von RVPen anwendbar sein und daher unabhängig von speziellen Eigenschaften der Signale sein.

3. **Festlegung der Güte anhand Grenzfrequenzen der Kopplungsalgorithmen.** In Abschnitt 4.1.2 sind für die in dieser Arbeit betrachteten Kopplungsalgorithmen Grenzfrequenzen in Abhängigkeit des Verhältnisses aus Bandbreite eines Koppelsignals und Makroschrittweite $\omega\Delta T$ definiert (s. Tab. 4.1). Die Bewertung der Extrapolationsgüte soll sich an diesen Grenzfrequenzen orientieren, um die Güte einer Extrapolation nur dann als ausreichend zu bewerten, wenn der gesamte Frequenzbereich des Signals unterhalb der Grenzfrequenz des Kopplungsalgorithmus liegt. Daraus folgt auch, dass das Verfahren für konstante Verläufe von Koppelsignalen keine Fehler tolerieren darf, da die Extrapolation eines konstanten Signalverlaufs trivial ist.

### 5.2.1 Verfahren nach Stettinger

Die Idee der online Bestimmung der Extrapolationsgüte für RVPen ist nicht neu. Stettinger et al. (2017) bewertet für jeden Makrozeitschritt $n$ den Extrapolationsfehler $e_{ext,n}$, indem er ihn mit einem Toleranzwert $e_{tol,n}$ vergleicht. Dabei gilt die Extrapolation als ausreichend, wenn die Ungleichung

$$e_{ext,n} \leq e_{tol,n} \tag{5-7}$$

erfüllt ist. Der Toleranzwert $e_{tol,n}$ ist nach Stettinger et al. (2017) variabel und ergibt sich aus der Differenz des idealen Ergebnisses der Extrapolation und eines früheren Werts zu

$$\begin{aligned} e_{tol,n} &= |y_n - y(t_n - t_{tol})| \text{ mit} \\ t_{tol} &= \delta \cdot k \cdot \Delta\mathrm{T} \text{ und } \delta > 0. \end{aligned} \tag{5-8}$$

Das ideale Ergebnis der Extrapolation liegt erst $k$ Makrozeitschritte nach der Extrapolation vor, weshalb die Güte nur rückwirkend bestimmt werden kann. Für den Anteil des Extrapolationshorizonts schlägt Stettinger et al. (2017) $\delta = 0{,}1$ vor. Da die Koppelsignale in RVPen diskret sind, liegen sie im Allgemeinen zum Zeitpunkt $t_n - t_{tol}$ nicht vor, weshalb $y(t_n - t_{tol})$ mittels Interpolation bestimmt werden muss.

Durch Anwendung einer linearen Interpolation zwischen zwei benachbarten Makrozeitschritten kann die Berechnung von $e_{tol,n}$ umgeformt werden und es wird ersichtlich, dass es sich bei der Toleranzfunktion

$$e_{tol,n} = |y_n - y(t_n - t_{tol})| = \left|\frac{y_n - y(t_n - t_{tol})}{t_{tol}}\right| \cdot t_{tol} = \overset{lin.\ Int.}{=} \left|\frac{y_n - y_{n-1}}{\Delta T}\right| \cdot t_{tol} = |f_n'| \cdot t_{tol} \tag{5-9}$$

um eine Gewichtung der ersten Ableitung nach der Zeit des Signals handelt. Das Verfahren von Stettinger et al. (2017) ist in Abb. 5.1 skizziert. Im dargestellten Fall wäre der Fehler $e_{ext,n}$ größer als die Toleranz $e_{tol,n}$ und die zum Zeitpunkt $t_{n-k}$ durchgeführte Extrapolation nach Gl. (5-7) somit nicht ausreichend.

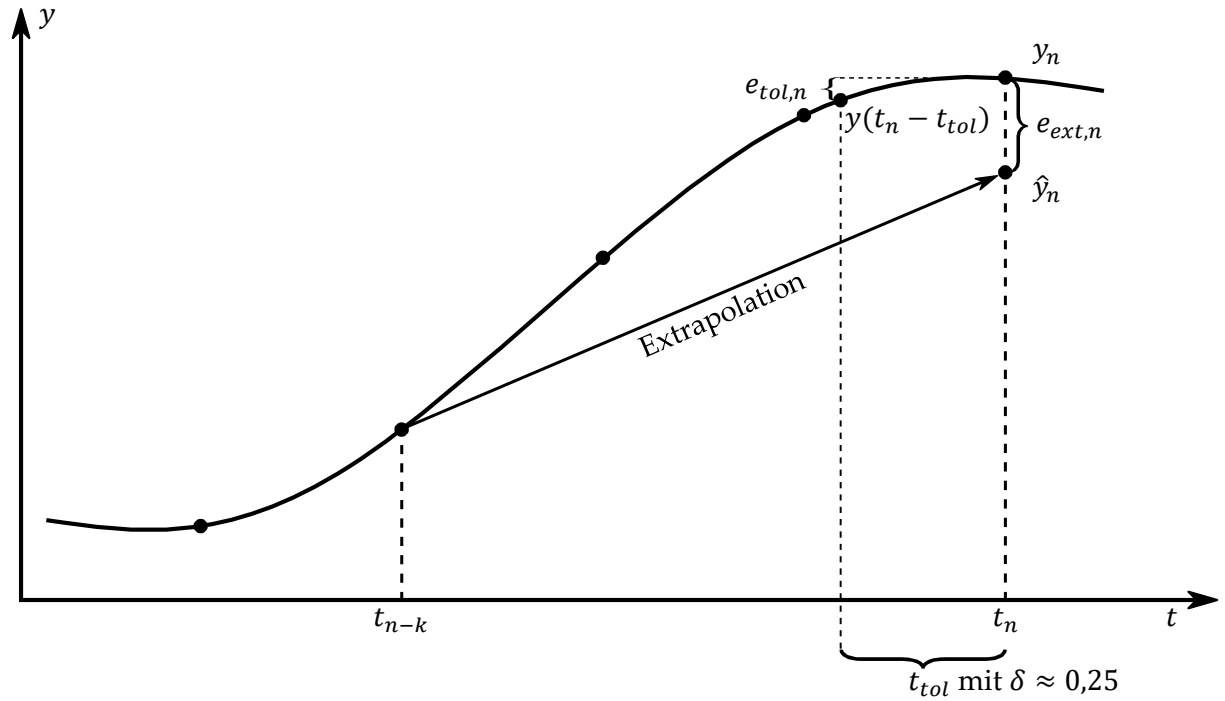


**Abb. 5.1** Verfahren zur Bewertung der Extrapolationsgüte nach Stettinger et al. (2017).

Das Verfahren erfüllt die erste und zweite Anforderung, die oben für die Bewertung der Extrapolationsgüte definiert sind. Da die Toleranzfunktion lediglich die erste Ableitung verwendet, welche aus dem ersten Differenzenquotienten berechnet wird, ist der Berechnungsaufwand sehr gering und allgemein anwendbar. Die dritte Anforderung ist jedoch nicht erfüllt: Die Toleranzfunktion ist für jeden Kopplungsalgorithmus identisch und über den Parameter $\delta = 0{,}1$ definiert. Die Grenzfrequenzen der Kopplungsalgorithmen fließen nicht in die Definition der Toleranzfunktion ein.

Weiterhin ist das Verfahren nach Stettinger et al. (2017) nicht geeignet, um die Extrapolationsgüte der in dieser Arbeit untersuchten Kopplungsalgorithmen zu bestimmen, deren entspre-

chendes ARIMA Modell einen Differenzierungsgrad größer als eins aufweist ($d > 1$), wie beispielsweise FOH, EROS3 und EROS4 (vgl. Abschnitt 3.3.1). Dies liegt daran, dass die Fehlertoleranz im Bereich von Extremstellen im Koppelsignal unabhängig ihrer Parametrierung auf null sinkt, da sie ausschließlich von der ersten Ableitung abhängt (s. Abb. 5.1). Auf der anderen Seite entstehen bei der Extrapolation mittels Kopplungsalgorithmen mit $d > 1$ vor allem an Extremstellen Fehler, wodurch dessen Extrapolation nie als ausreichend deklariert werden würde, obwohl die Frequenz des Signals innerhalb des Gültigkeitsbereichs des jeweiligen Kopplungsalgorithmus liegt.

Um dies zu demonstrieren ist in Abb. 5.2 die Fehlertoleranz nach Stettinger et al. (2017) für ein sinusförmiges Testsignal dargestellt und mit den Extrapolationsfehlern für verschiedene Kopplungsalgorithmen verglichen. Die Frequenz des Testsignals, welches im oberen Teil der Abb. 5.2 dargestellt ist, entspricht dabei genau der Grenzfrequenz vom EROS3. Mit dem gewählten $\Delta T = 0{,}02s$ und $k = 3$ ergibt sich diese nach Tab. 4.1 zu $\omega = 0{,}9650\, rad/s$. Wie im mittleren und unteren Teil der Abb. 5.2 zu beobachten, strebt die Fehlertoleranz $e_{tol}$ gemäß der Erwartung an Extremstellen gegen null, wodurch die Extrapolation keiner der Kopplungsalgorithmen mit Differenzierungsgrad größer eins als ausreichend deklariert wird, obwohl die Frequenz des Testsignals für EROS4 und EROS4 innerhalb des Gültigkeitsbereichs liegt.

Das Verfahren nach Stettinger et al. (2017) erfüllt somit die dritte Anforderung nicht, die am Anfang des Kapitels für ein Verfahren zur Bestimmung der Extrapolationsgüte für RVPen definiert wurde. Daher muss das Verfahren adaptiert werden muss. Die Grundidee des Verfahrens, online den tolerierbaren Extrapolationsfehler mit einer variablen Toleranzfunktion auszudrücken und mit dem aktuellen Extrapolationsfehler zu vergleichen, soll dabei beibehalten werden und lediglich die Toleranzfunktion erweitert werden.

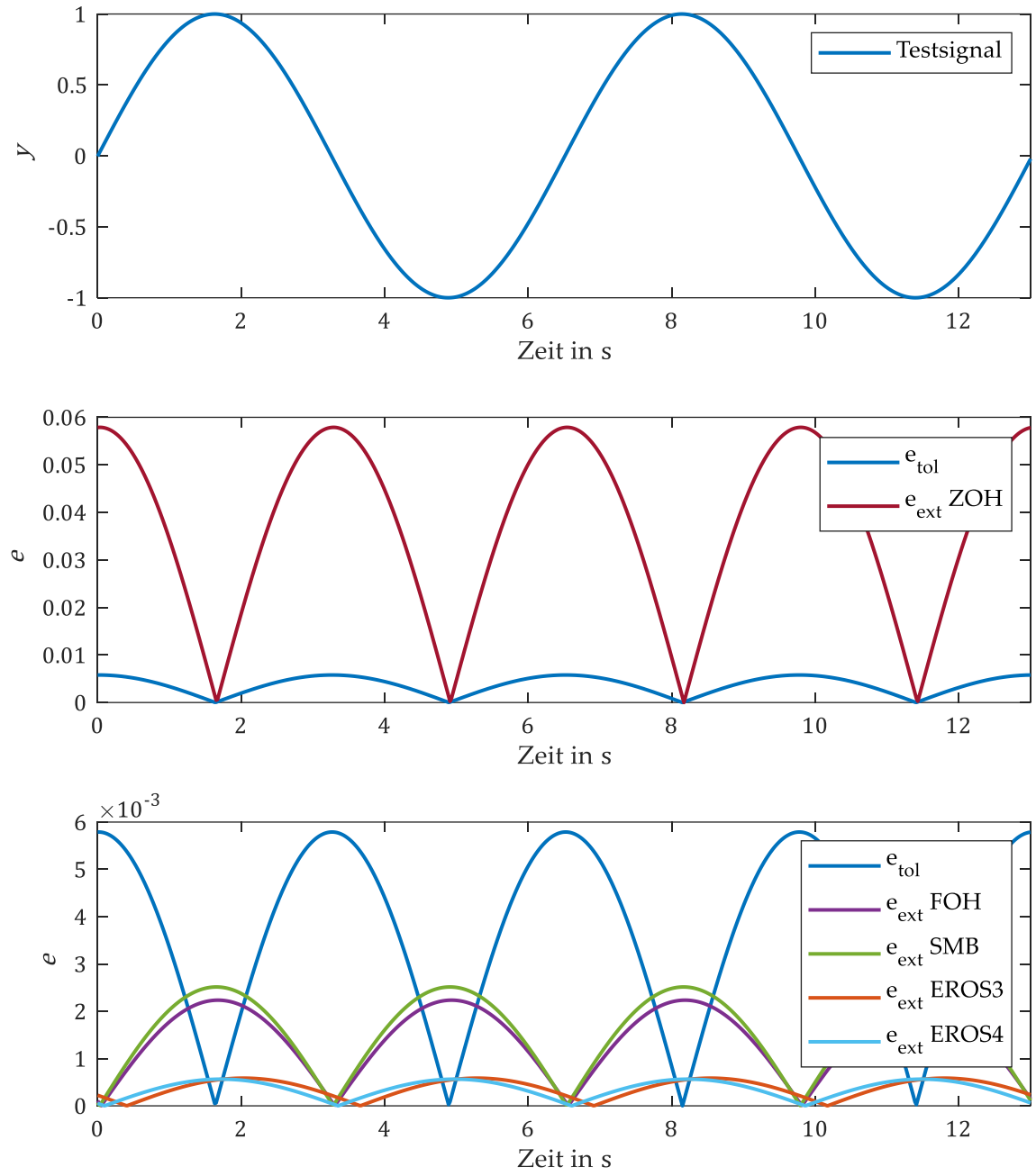


**Abb. 5.2** Vergleich der Extrapolationsfehler für verschiedene Kopplungsalgorithmen mit der Fehlertoleranz nach Stettinger et al. (2017) (mitte und unten) anhand eines Testsignals (oben)

### 5.2.2 Erweiterung der Fehlertoleranzfunktion

Wie im mittleren Teil der Abb. 5.2 zu sehen, passt der qualitative Verlauf der Fehlertoleranz von Stettinger et al. (2017) zu dem des Extrapolationsfehlers vom ZOH. Allerdings passt er nicht zu denen der Kopplungsalgorithmen mit Differenzierungsgrad größer eins, wie es im

unteren im unteren Teil der Abb. 5.2 dargestellt ist. Dies ist damit zu erklären, dass die Extrapolationsfunktion vom ZOH eine Funktion nullter Ordnung ist und somit vor allem Fehler erster Ordnung aufweist. Bei den anderen Kopplungsalgorithmen liegt eine Funktion erster Ordnung zu Grunde, weshalb sich deren Extrapolationsfehler größtenteils aus Fehlern zweiter Ordnung zusammensetzt. Wie in Abb. 5.2 am Beispiel EROS4 gezeigt, kann ein Kopplungsalgorithmus Extrapolationsfehler zweiter Ordnung aufweisen, auch wenn er auf ein Signal angewandt wird, das innerhalb seines Gültigkeitsbereichs liegt. Eine Möglichkeit um dies abzubilden besteht darin, dass auch die Toleranzfunktion Fehler zweiter Ordnung beinhaltet, weshalb die Toleranzfunktion von Stettinger et al. (2017) um einen entsprechenden Term mit der zweiten Ableitung des Signals erweitert wird. Die neue Fehlertoleranzfunktion ergibt sich zu

$$\begin{aligned} e_{tol,n} &= (|f_n'| + \zeta|f_n''|) \cdot t_{tol} = \\ &= \left(\left|\frac{y_n - y_{n-1}}{\Delta T}\right| + \zeta \left|\frac{y_n - 2y_{n-1} + y_{n-2}}{\Delta T^2}\right|\right) \cdot t_{tol} \\ &\text{mit } t_{tol} = \delta \cdot k \cdot \Delta\mathrm{T} \text{ und } \delta > 0,\ \zeta > 0 \end{aligned} \tag{5-10}$$

und enthält zwei Parameter. Zum einen den Parameter $\delta$, welcher weiterhin die Länge der Toleranzzeit $t_{tol}$ definiert und zum anderen einen neuen Parameter $\zeta$, der das Verhältnis aus erster und zweiter Ableitung innerhalb der Toleranzunktion festlegt. Im Folgenden werden diese Parameter, wie in der dritten Anforderung am Anfang des Kapitels beschrieben, für jeden Kopplungsalgorithmus so identifiziert, dass nur die Extrapolation von Signalen nach Gl. (5-7) als ausreichend bewertet wird, deren Bandbreite innerhalb des Gültigkeitsbereichs des jeweiligen Kopplungsalgorithmus liegen.

Die beiden Parameter der Toleranzfunktion werden anhand des Extrapolationsfehlers identifiziert, welcher bei Verwendung des Kopplungsalgorithmus an der Grenze seines Gültigkeitsbereichs auftritt. Dazu wird der Kopplungsalgorithmus auf ein Testsignal angewandt, dessen Frequenz exakt der Grenzfrequenz $\bar{\omega}$ des Kopplungsalgorithmus entspricht. Mithilfe des Verlaufs des dabei gemessenen Extrapolationsfehlers ergibt sich der Parameter $\zeta$ aus dem Fehler an den Extremstellen des Testsignals $e_{es}$ und dem Fehler an den Wendepunkten des Testsignals $e_{wp}$ zu

$$\zeta = \frac{e_{es}}{e_{wp}}. \tag{5-11}$$

Die Idee dabei ist, dass Fehler, die an einer Extremstelle entstehen vom Term der zweiten Ableitung der Toleranzfunktion approximiert werden müssen, da die erste Ableitung dort null ist. Fehler an Wendepunkten werden entsprechend über den Term der ersten Ableitung approximiert. Somit beschreibt $\zeta$ das Verhältnis aus Fehlern erster und zweiter Ordnung, die bei der Anwendung des Kopplungsalgorithmus auf ein Signal mit seiner Grenzfrequenz auftreten. Der Parameter $\delta$ skaliert die Höhe des tolerierten Extrapolationsfehlers und ergibt sich

durch Umformung von Gl. (5-10) mithilfe des gemessenen Extrapolationsfehlers des Testsignals zu

$$\delta = \frac{e_{tol,n}}{(|f_n'| + \zeta|f_n''|) \cdot \Delta T \cdot k}. \tag{5-12}$$

Es bietet sich an Gl. (5-12) an einer Extremstelle auszuwerten, da dort für das sinusförmige Testsignal $|f_n'| = 0$ und $|f_n''| = \bar{\omega}^2$ gilt und der Fehler an dieser Stelle $e_{tol} = e_{es}$ bereits aus der Bestimmung von $\zeta$ bekannt ist. Der Skalierungsparameter ergibt sich somit zu

$$\delta = \frac{e_{es}}{\zeta \cdot \bar{\omega}^2 \cdot \Delta T \cdot k} \tag{5-13}$$

und hängt somit direkt von der Grenzfrequenz des verwendeten Kopplungsalgorithmus ab. Außerdem ergibt sich eine Abhängigkeit von der gewählten Makroschrittweite $\Delta T$ und der aktuellen Anzahl an Makroschritten Verzögerung $k$, wodurch der Parameter $\delta$ hier, genau wie bei dem Verfahren nach Stettinger et al. (2017), variabel ist.

Tab. 5.1 zeigt die Parameter $\zeta$ und $\delta$ beispielhaft für $\Delta T = 0{,}02s$ und $k = 3$ für die untersuchten generischen Kopplungsalgorithmen. Wie bereits in Abb. 5.2 beobachtet, wird der Fehler vom ZOH größtenteils über die erste Ableitung approximiert (kleines $\zeta$). Bei den anderen Algorithmen ist der Anteil der zweiten Ableitung in der Toleranzfunktion größer.

**Tab. 5.1.** Parameter der entwickelten Toleranzfehlerfunktion für verschiedene Kopplungsalgorithmen basierend auf deren Grenzfrequenzen aus Tab. 4.1 für $\Delta T = 0{,}02s$ und $k = 3$

| | $\zeta$ | $\delta$ |
|---|---|---|
| ZOH | 0,0048 | 0,99 |
| FOH | 44,58 | 0,0009 |
| SMB | 85,81 | 0,0005 |
| EROS3 | 2,46 | 0,0039 |
| EROS4 | 8,40 | 0,0012 |

Abb. 5.3 zeigt beispielhaft für EROS4 den Vergleich aus parametrierter Fehlertoleranzfunktion und Extrapolationsfehler für das Testsignal mit der Grenzfrequenz vom EROS4. Wie für ein Signal am Rande des Gültigkeitsbereichs vom EROS4 zu erwarten, sind Toleranz und Fehler in manchen Bereichen nahezu identisch, wodurch das entwickelte Verfahren die Extrapolationsgüte in diesem Fall nur knapp als ausreichend bewerten würde.

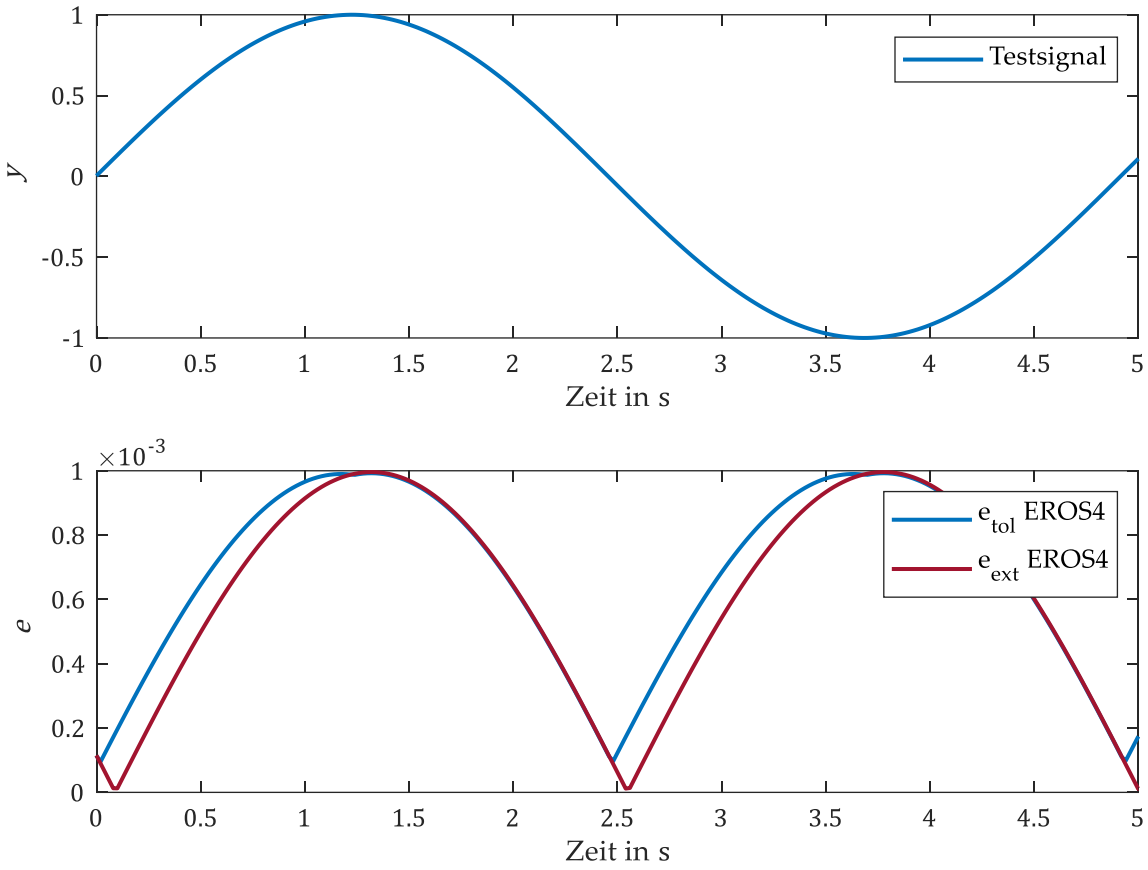


**Abb. 5.3** Vergleich der parametrierten Fehlertoleranzfunktion $e_{tol}$ mit dem Extrapolationsfehler vom EROS4 (unten) für ein Testsignal mit der Grenzfrequenz vom EROS4 (oben)

Zur Demonstration des Verfahrens zur Bewertung der Extrapolationsgüte wird es an einem Signal getestet, welches aus zwei sich überlagernden Sinuswellen unterschiedlicher Frequenz, Amplitude und Offset besteht. Deren Frequenzen liegen mit $\omega = 0{,}5\,rad/s$ und $\omega = 0{,}75\,rad/s$ für die über die Parameter $\Delta T = 0{,}02s$ und $k = 3$ beispielhaft festgelegte Latenzzeit genau zwischen der Grenzfrequenz vom FOH und EROS3 (vgl. Tab. 4.1). Abb. 5.4 zeigt das Ergebnis der online Bewertung der Extrapolationsgüte für ZOH (oben) und FOH (unten). Obwohl beide Kopplungsalgorithmen außerhalb ihres Gültigkeitsbereichs verwendet werden, ist deren Extrapolation nicht durchgehend schlecht bewertet. Dies ist korrekt, da deren Extrapolationsfehler nicht überall zu groß ist. FOH extrapoliert das Signal in Bereichen zwischen Extremstellen auch für Frequenzen außerhalb des Gültigkeitsbereichs ausreichend gut, generiert aber an den Extremstellen selbst zu große Fehler, die vom Verfahren als nicht ausreichend bewertet sind. Beim ZOH ist es andersherum. In Bereich von Extremstellen ist die konstante Extrapolation gut, da sich der Signalwert kaum ändert und zwischen den Extremstellen wird der Fehler als zu groß bewertet. Die Extrapolation vom EROS3 und EROS4 ist für das dargestellte Signal durchgängig als gut bewertet. Dies entspricht ebenfalls dem erwarteten Ergebnis,

da die Frequenzen des Signals innerhalb des Gültigkeitsbereichs der beiden Algorithmen liegen.

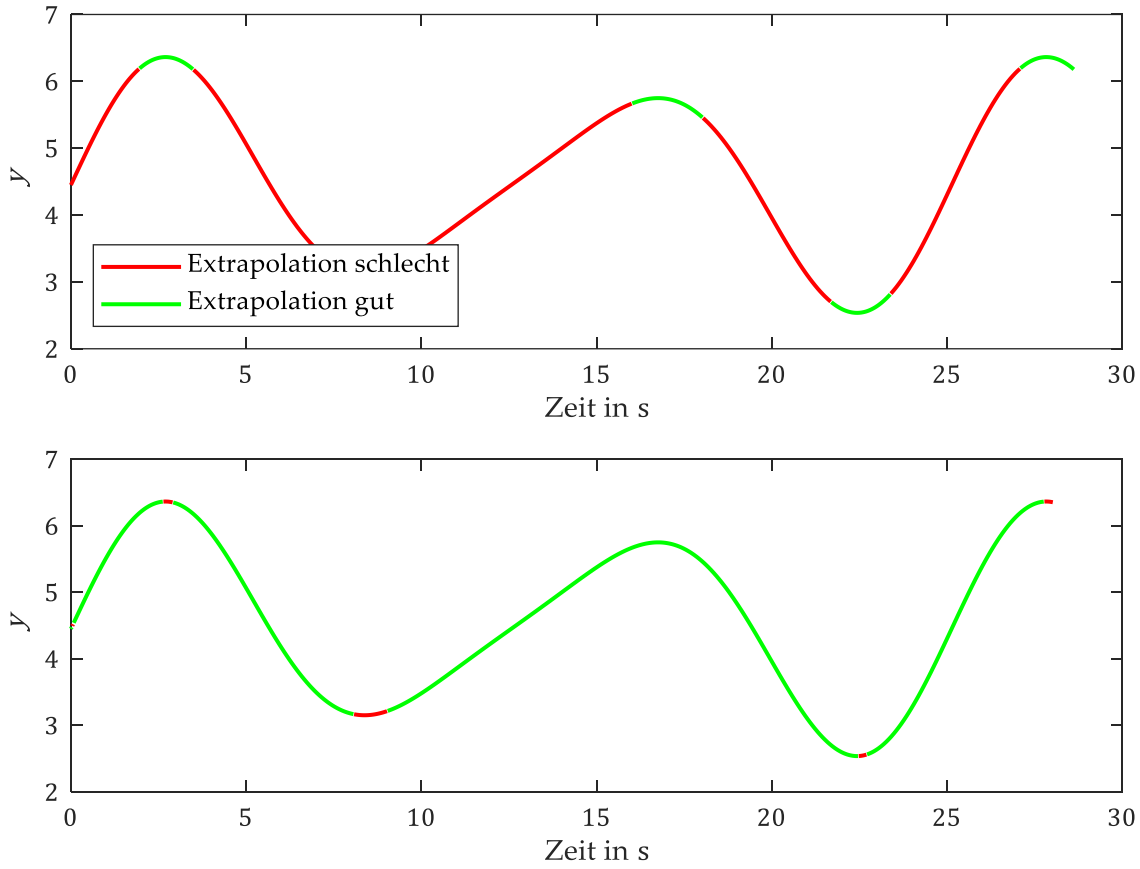


**Abb. 5.4** Bewertung der Extrapolation vom ZOH (oben) und FOH (unten) für zwei überlagerte Sinusschwingungen mit der entwickelten Toleranzfunktion

Aus diesen Beobachtungen lässt sich ableiten, dass ein Kopplungsalgorithmus nur für die Anwendung auf ein Koppelsignal geeignet ist, wenn der Extrapolationsfehler nie die Toleranz überschreitet.

Zusammengefasst erfüllt das aufgezeigte Verfahren zur online Bestimmung der Extrapolationsgüte die drei Anforderungen, die am Anfang des Kapitels definiert sind. Der Berechnungsaufwand ist sehr gering, da lediglich die erste und zweite Ableitung aus den jeweiligen Rückwärtsdifferenzenquotienten bestimmt werden müssen. Außerdem ist es allgemein anwendbar und lässt sich für jeden Kopplungsalgorithmus einzeln über dessen Grenzfrequenzen parametrieren.

Neben des hier gezeigten Ansatzes der Verwendung einer Toleranzfunktion im Zeitbereich zur online Bestimmung der Extrapolationsgüte wurde im Rahmen dieser Arbeit außerdem versucht, die in den Koppelsignalen enthaltenen Frequenzen über die Instantaneous Fre-

quency (IF) (Huang et al., 2009) zu schätzen und direkt mit den Grenzfrequenzen der Kopplungsalgorithmen zu vergleichen. Allerdings konnte die IF mit keiner der untersuchten Methoden online genau genug geschätzt werden, um sie sinnvoll mit den Grenzfrequenzen der Kopplungsalgorithmen vergleichen zu können. Sowohl bei der Empirical Mode Decomposition (Holzinger und Benedikt, 2014) als auch die Direct Quadrature Method (Huang et al., 2009), die beide aufgrund eines relativ geringen Berechnungsaufwands für online Anwendungen geeignet sind, muss für die präzise Bestimmung der IF ein langer, stetiger Zeithorizont mit mehreren Extremstellen betrachtet werden. Diese Voraussetzungen sind für ein Koppelsignal eines RVPs, in dem vereinzelte Diskontinuitäten auftreten können, nicht allgemein erfüllt, weshalb der Ansatz mittels IF in dieser Arbeit nicht für die Bestimmung der Extrapolationsgüte verwendet wurde.

## 5.3 Umgang mit Diskontinuitäten in Koppelsignalen

Alle in dieser Arbeit untersuchten generischen Kopplungsalgorithmen, deren entsprechendes ARIMA Modell einen Differenzierungsgrad größer als eins ($d > 1$) aufweist (vgl. Abschnitt 3.3.2), neigen dazu hohe Frequenzen in Koppelsignalen zu verstärken (s. Abschnitt 4.1.2). Dadurch entstehen große Extrapolationsfehler, wenn sie zu Zeitpunkten verwendet werden, an denen eine Diskontinuität im Signalverlauf eines Koppelsignals vorkommt. Einzig der Standardalgorithmus ZOH überhöht Sprungstellen im Koppelsignal nicht, führt dagegen in Abschnitten, in denen das Koppelsignal stetig ist, zu größeren Extrapolationsfehlern als andere Algorithmen wie FOH, EROS3 und EROS4 (s. Abschnitt 3.4). Somit ist keiner der untersuchten generischen Kopplungsalgorithmen für Koppelsignale geeignet, deren meist stetiger Verlauf von vereinzelten Diskontinuitäten unterbrochen ist. Eine Ausnahme bilden Koppelsignale mit Diskontinuitäten, deren Auftreten von den Koppelsignalen abhängt und daher durch einen lernenden Algorithmus vorhersagbar sind (vgl. Abschnitt 3.5). Da Koppelsignale mit Diskontinuitäten, die keinem Muster folgen, in RVPen häufig vorkommen (z.B. Lastmoment eines Motors), wird in diesem Kapitel, wie in Abschnitt 3.1 gefordert, eine Methodik vorgeschlagen, um die Latenzzeiten in stetigen Abschnitten von Koppelsignalen zu kompensieren ohne vereinzelte Diskontinuitäten fälschlicherweise zu verstärken.

Das vorgeschlagene Verfahren besteht aus zwei Teilen: Einer Erkennung von Diskontinuitäten im Koppelsignal (s. Abschnitt 5.3.1) und eine Vorgehensweise zum Umschalten des verwendeten Kopplungsalgorithmus (s. Abschnitt 5.3.2), um auf eine erkannte Diskontinuität zu reagieren. Es wurde im Rahmen dieser Arbeit im Artikel (Baumann et al., 2024) veröffentlicht.

### 5.3.1 Verfahren zur online Erkennung von Diskontinuitäten

In Abschnitt 3.1 wurde bereits definiert, dass in einem Koppelsignal zum Zeitpunkt $t_n$ eine Diskontinuität bzw. Unstetigkeitsstelle vorliegt, wenn die kontinuierliche Größe des realen

oder virtuellen Prototyps, aus dem das Koppelsignal durch Abtastung hervorgeht, im Zeitraum $t_{n-1} < t \leq t_n$ eine Unstetigkeitsstelle besitzt.

Ziel ist es daher ein Verfahren zu entwickeln, welches anhand der abgetasteten Signalwerte eines Koppelsignals erkennt, ob in der ursprünglichen kontinuierlichen Ausgangsgröße eine Diskontinuität vorliegt. Analytisch ist dieses Problem nicht zu lösen, da die Methoden zur Erkennung von Unstetigkeitsstellen wie das Epsilon-Delta Kriterium oder die Berechnung einseitiger Grenzwerte von diskreten Zeitreihen nicht anwendbar sind. Stattdessen muss ein Maß gefunden werden, um in jedem Makroschritt abzuschätzen, ob die Anwendung eines Kopplungsalgorithmus mit Differenzierungsgrad größer eins zu einem signifikanten Fehler führen würde, falls eine Unstetigkeitsstelle in der kontinuierlichen Größe vorliegt.

Da die Erkennung erfolgen muss, bevor ein Kopplungsalgorithmus auf dieses Signal angewandt wird, muss erkannt werden, ob eine Diskontinuität zwischen dem letzten und dem aktuellen Signalwert, also am äußersten Rand der Zeitreihe, vorliegt. Das Verfahren zur Bestimmung der Extrapolationsgüte aus Abschnitt 5.2 ist nicht zur Erkennung von Diskontinuitäten am Rand der Zeitreihe geeignet, da die Güte in diesem Verfahren erst $k$ Makrozeitschritte rückwirkend bestimmt wird (vgl. Abb. 5.1 in Abschnitt 5.2.1) und dieser kurze Zeitraum von $k$ Zeitschritten bereits ausreicht, um nach einer Diskontinuität große Extrapolationsfehler in das System einzubringen. Das Verfahren zur Erkennung von Diskontinuitäten wird online während der Laufzeit von RVPen eingesetzt, weshalb es echtzeitfähig und die Berechnungsdauer somit möglichst gering und beschränkt sein muss.

In der Statistik sind Methoden und Verfahren zur Erkennung von Änderungen in Zeitreihen unter dem Oberbegriff Change-Point-Detection (CPD) (Truong, Oudre und Vayatis, 2020) zu finden und die hier vorliegende Herausforderung der online Erkennung am Rand der Zeitreihe wird als online Change-Point-Detection (Aminikhanghahi und Cook, 2017) bezeichnet. Ziel der verschiedenen CPD-Verfahren ist es jeweils eine abrupte Änderung in statistischen Größen wie dem Mittelwert oder der Varianz zu finden. Im Gegensatz zu den hier betrachteten Koppelsignalen von RVPen liegen den analysierten Zeitreihen bei der CPD in der Regel keine stetigen Ausgangsgrößen zu Grunde, weshalb sie für den hier benötigten Anwendungsfall nicht optimal geeignet sind. Im Bereich der Computerwissenschaften und insbesondere bei Internet-of-Things (IoT) Anwendungen, welche aufgrund des Datenaustauschs über ein Netzwerk ähnlich zu RVPen sind, wird die Erkennung von Änderungen in zeitlichen Daten Anomalie-Erkennung genannt (Gupta et al., 2013, Cook, Mısırlı und Fan, 2019). Anwendungsfälle zur online Erkennung von Anomalien sind in diesen Bereichen seltener und werden meist mithilfe lernender Methoden wie beispielsweise Recurrent Neural Networks (RNNs) adressiert (Saurav et al., 2018).

**Kern des entwickelten Verfahrens**

Wie oben erwähnt müssen Diskontinuitäten in Koppelsignalen von RVPen nur erkannt werden, wenn sie bei der Anwendung des ausgewählten Kopplungsalgorithmus auf dieses Signal zum Zeitpunkt der Diskontinuität einen großen Extrapolationsfehler verursachen würden. Wie anhand der Frequenzbereichsanalyse aus Abschnitt 4.1 bekannt, ist das der Fall, wenn durch die Diskontinuität plötzlich hohe Frequenzanteile im Spektrum des Koppelsignals auftreten. Das hier vorgeschlagene Verfahren basiert auf der Idee abrupte Zunahmen der Amplitude hoher Frequenzen im Koppelsignal zu erkennen, da dies die direkte Ursache für große Extrapolationsfehler von Kopplungsalgorithmen ist, deren entsprechendes ARIMA Modell einen Differenzierungsgrad größer als eins aufweist. Um eine Veränderung der Amplitude hoher Frequenzen im Signal zu erkennen, wird dessen Frequenzspektrum online für jeden neuen Signalwert bestimmt und mit dem Frequenzspektrum aus dem vorherigen Makroschritt verglichen. Aufgrund der limitierten Berechnungszeit ist wichtig, dass für die Erkennung der Zunahme hoher Frequenzanteile ein niedrig aufgelöstes Frequenzspektrum ausreichend ist.

**Online Berechnung des Frequenzspektrums**

Die Berechnung des Frequenzspektrums erfolgt in jedem Makroschritt durch Anwendung der Diskreten Fourier Transformation (DFT) auf die letzten $N$ Werte eines Koppelsignals $y$, wodurch die Zeitreihe in die Frequenzanteile

$$Y_q = DFT(y) = \sum_{n=1}^{N} y_n \cdot e^{-\frac{j2\pi}{N}qn} \tag{5-14}$$

transformiert wird. Da die Signale von RVPen im Zeitbereich reell sind, sind sie im Frequenzbereich hermitesch. Dies bedeutet, dass das Ergebnis der DFT eines reellwertigen Signals $rDFT(\cdot)$ nur die Frequenzen mit positivem Frequenzanteil beinhaltet. Der Vektor der Frequenzanteile $Y_q$ reduziert sich somit auf die ersten $\lfloor N/2 \rfloor + 1$ Einträge der DFT, weshalb deren Amplitude entsprechend skaliert werden muss. Es ergibt sich

$$Y = \frac{2}{N} |rDFT(y)|. \tag{5-15}$$

Die Länge des berücksichtigten Signalabschnitts $N$ beeinflusst über den Zusammenhang

$$\omega_{res} = \frac{2\pi}{N} \tag{5-16}$$

die Auflösung der Frequenzanteile in der DFT. Ein größeres $N$ erhöht die Auflösung aber auch die Berechnungsdauer. Außerdem verringert ein größeres $N$ aufgrund der Skalierung der rDFT mit $2/N$ den Einfluss des letzten Signalwerts, welcher auf eine Diskontinuität getestet

werden soll, auf die einzelnen Frequenzanteile. Weiterhin erfordert der effiziente Einsatz der meisten Fast-Fourier-Transform (FFT) Algorithmen, die zur Berechnung der DFT verwendet werden, eine Signallänge entsprechend einer Zweierpotenz. Daher wird für das hier vorgeschlagene Verfahren zur Diskontinuitäten-Erkennung in Koppelsignalen von RVPen $N = 8$ festgelegt. Die Auflösung der DFT ergibt sich somit zu $\omega_{res} = \pi/4$, wodurch sie 25% der Nyquistfrequenz des Signals entspricht.

Eine wichtige Annahme, die bei der Anwendung der DFT auf einen Signalabschnitt getroffen wird, ist die periodische Fortsetzung des Signalabschnitts. Ist dies nicht der Fall werden aufgrund des Leakage-Effekts fälschlicherweise hohe Frequenzanteile zum Frequenzspektrum hinzugefügt. Um diesen Effekt zu verringern, wird der untersuchte Signalabschnitt vor der Transformation mit einer Fensterfunktion multipliziert. Üblicherweise werden dazu Funktionen verwendet, welche die jeweiligen Ränder des Signalabschnitts gegen Null gehen lassen, um so eine Periodizität des Signalabschnitts zu erreichen. Für das hier entwickelte Verfahren zur Erkennung von Diskontinuitäten ist die Verwendung eines solchen Fensters nicht zielführend, da dies eine zu erkennende Diskontinuität am Ende eines Signalabschnitts stark dämpfen und so eine Erkennung unmöglich machen würde (vgl. angewandtes Hann-Fenster in Abb. 5.5). Um diesem Effekt entgegenzuwirken, wird der betrachtete Signalabschnitt zunächst so verschoben, dass für den letzten Wert des Abschnitts $y_n = 0$ gilt und anschließend mit einem einseitigen bzw. halben Hann-Fenster der Form

$$h(n) = 0{,}5\left(1 - \cos\left(\frac{\pi n}{M}\right)\right), 1 \leq n \leq M \tag{5-17}$$

multipliziert. Die Fensterbreite $M$ entspricht dabei der Länge des Signalabschnitts $N$ und $n$ dem Index der Werte des Signalabschnitts, wobei ein größeres $n$ für einen neueren Signalwert steht. Das halbe Hann-Fenster besteht aus der linken Hälfte des Hann-Fensters (Oppenheim, 1999).

Die Anwendung des halben Hann-Fensters auf einen verschobenen Signalabschnitt, welcher am aktuellen Makrozeitpunkt $t_n$ eine Diskontinuität enthält, ist in Abb. 5.5 gezeigt. Der resultierende Signalabschnitt (gelb) ist wie gefordert periodisch fortsetzbar, ohne eine zusätzliche Diskontinuität zwischen den Perioden einzufügen, und die Sprungstelle im letzten Signalwert ist wie im ursprünglichen Signalabschnitt (blau) gut erkennbar. Zum Vergleich zeigt Abb. 5.5 außerdem die Anwendung des normalen Hann-Fensters auf den nicht verschobenen Signalabschnitt. Das Hann-Fenster glättet die Diskontinuität am Rand des Signalabschnitts, wodurch diese im Frequenzspektrum nicht mehr erkennbar wäre. Die Veränderung des Signalabschnitts durch den Einsatz einer einseitigen Fensterfunktion wirkt sich nicht negativ auf das Verfahren aus, da, wie oben erläutert, lediglich die relative Veränderung des Frequenzspektrums des aktuellen Signalabschnitts zum letzten Signalabschnitt berücksichtigt und nicht das genaue Frequenzspektrum benötigt wird.

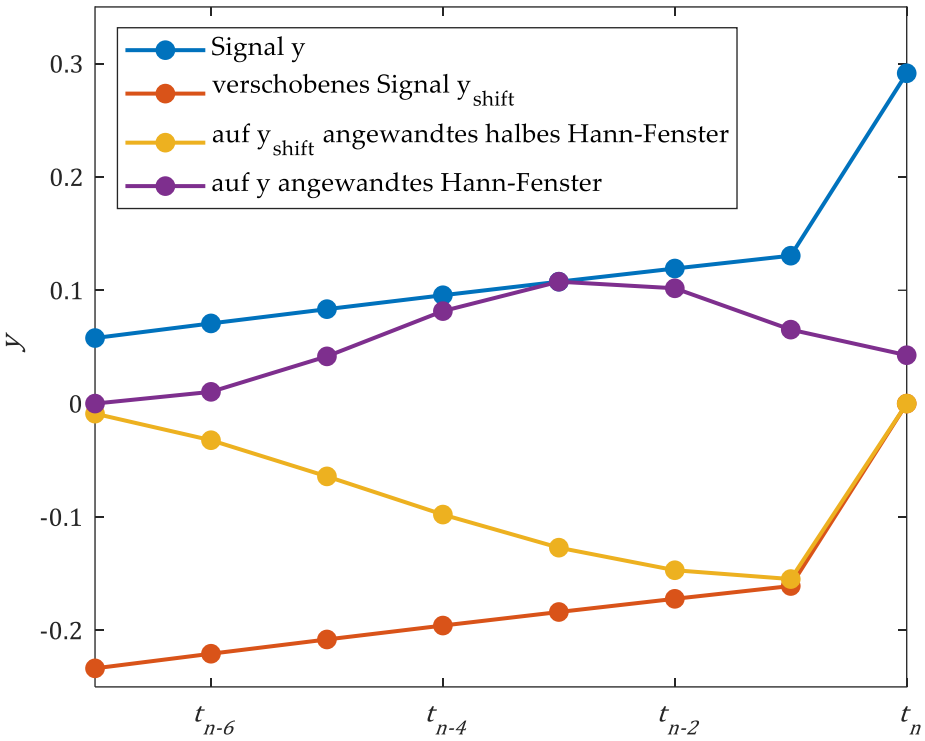


**Abb. 5.5** Demonstration der Anwendung des halben Hann-Fensters auf ein verschobenes Testsignal mit Diskontinuität im Vergleich zur Anwendung des Hann-Fensters

Die Anwendung einer Fensterfunktion auf ein Signal verändert dessen Amplitude und beeinflusst somit auch dessen Frequenzspektrum. Dieser Effekt wird kompensiert, indem das aus der DFT resultierende Frequenzspektrum mit einem Korrekturfaktor

$$A_w = \frac{M}{\sum_{n=1}^{M} h(n)} \tag{5-18}$$

multipliziert wird (Brandt, 2023). Für das halbe Hann-Fenster aus Gl. (5-17) ergibt sich der Korrekturfaktor zu $A_w = 1{,}7778$.

**Kriterium für Diskontinuität**

Das in dieser Arbeit entwickelte Verfahren erkennt eine Diskontinuität im aktuellen Signalwert eines Koppelsignals $y_n$, wenn die Bedingung

$$\sum_{k=2}^{5} Y_{q,n} > \beta \sum_{k=2}^{5} Y_{q,n-1} \tag{5-19}$$

erfüllt ist. Das Frequenzspektrum $Y_{q,n}$ des Signals im aktuellen Makrozeitpunkt $t_n$ berechnet sich dabei über die mit dem Faktor $A_w$(s. Gl. (5-18)) korrigierte, reellwertigen DFT aus Gl. (5-15) unter Verwendung des verschobenen und durch Einsatz des halben Hann-Fensters (s. Gl. (5-15)) veränderten Signalabschnitt. Das Frequenzspektrum $Y_{q,n-1}$ bezeichnet das entsprechende Spektrum, welches im vorangegangenen Makroschritt $t_{n-1}$ berechnet wurde. Die

Summationsindizes ergeben sich aus der Tatsache, dass für $N = 8$ und den daraus resultierenden $\lfloor N/2 \rfloor + 1 = 5$ Einträgen von $Y_q$, hohe Frequenzanteile über den Grenzfrequenzen der Kopplungsalgorithmen (s. Tab. 4.1 in Abschnitt 4.1.2) über der Auflösung des Frequenzspektrums von 25% der Nyquistfrequenz liegen. Der Parameter $\beta$ definiert die Sensitivität des Verfahrens und ist in dieser Arbeit basierend auf einer experimentellen Analyse auf $\beta = 5$ festgelegt.

Abb. 5.6 zeigt für das Beispielsignal aus Abb. 5.5 das Frequenzspektrum des Signalabschnitts zum aktuellen Makroschritt, welches eine Diskontinuität enthält, und das Frequenzspektrum desselben Signals einen Makrozeitschritt früher, welches noch keine Diskontinuität enthält. Wie erwartet erhöht eine Diskontinuität die Amplitude aller Frequenzanteile im Spektrum. Die Anteile oberhalb von 25% der Nyquistfrequenz sind in diesem Fall im Spektrum mit der Diskontinuität um Faktor 9,3 größer, wodurch das Kriterium aus Gl. (5-19) erfüllt ist und die Diskontinuität somit richtigerweise erkannt wird.

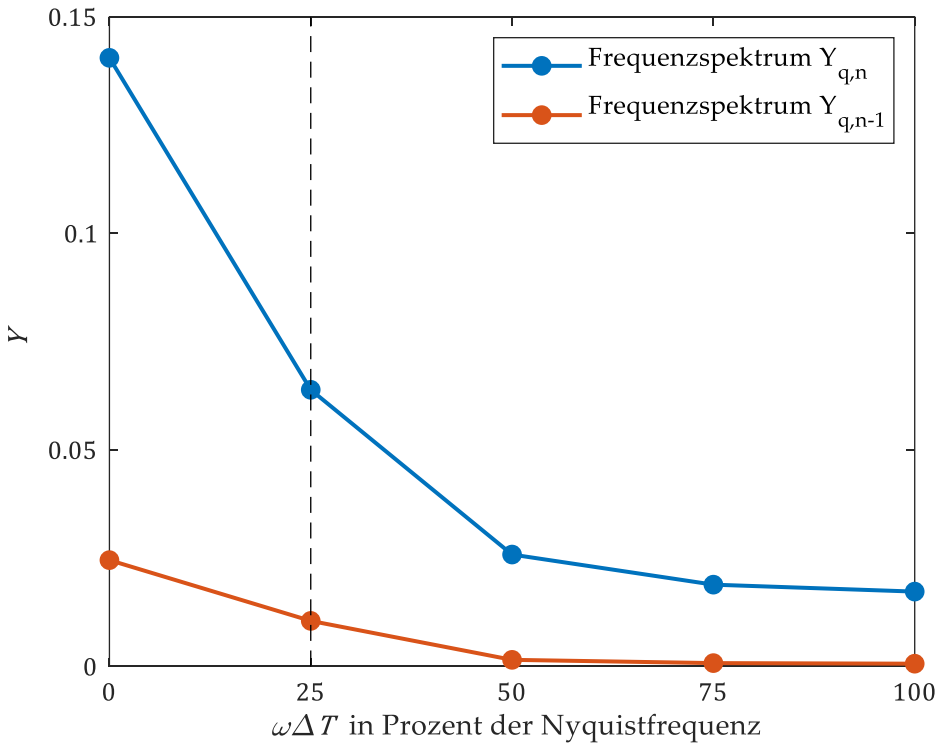


**Abb. 5.6** Resultierendes Frequenzspektrum für den Signalabschnitt aus Abb. 5.5 und dem Signalabschnitt aus dem vorherigen Makrozeitschritt

Mithilfe des hier vorgestellten Verfahrens zur Erkennung von Diskontinuitäten durch den Vergleich von Frequenzspektren verschiedener Signalabschnitte, werden Diskontinuitäten in Koppelsignalen erkannt, welche bei der Anwendung eines Kopplungsalgorithmus, der hohe Frequenzen verstärkt, zu großen Extrapolationsfehlern führen würde. Es ist wie gefordert online einsetzbar, da es Diskontinuitäten am Rand des Signalabschnitts erkennt und aufgrund

des mit $N = 8$ kurzen betrachteten Signalabschnitts einen geringen Berechnungsaufwand aufweist.

### 5.3.2 Umschalten zwischen Kopplungsalgorithmen

Im vorherigen Kapitel ist gezeigt, dass Diskontinuitäten in Koppelsignalen von RVPen online erkennbar sind, bevor ein Kopplungsalgorithmus auf sie angewandt wird. In diesem Abschnitt wird vorgeschlagen, wie im Fall einer erkannten Diskontinuität durch das Umschalten zwischen verschiedenen Kopplungsalgorithmen große Extrapolationsfehler zu vermeiden sind. Die CHOPtrey Methodik von Ben Khaled-El Feki et al. (2017), welche für die Co-Simulation von MIL-Simulationsumgebungen entwickelt wurde, verfolgt eine ähnlichen Ansatz zum Umschalten zwischen Algorithmen zur Verringerung des Rekonstruktionsfehlers.

Dazu ist in Abb. 5.7 zunächst anhand eines Beispiels dargestellt, wie sich verschiedene Kopplungsalgorithmen bei einer Diskontinuität im Koppelsignal ohne ein Umschalten verhalten. Die Abbildung zeigt einen Signalverlauf (Referenz), der eine abgetastete Ausgangsgröße eines realen oder virtuellen Prototyps repräsentiert. Bei der Datenübertragung zu einem anderen Prototyp entsteht eine Latenzzeit von $k = 3$ Makrozeitschritten, deren Einfluss auf das Signal mit verschiedenen dargestellten Kopplungsalgorithmen am Eingang des empfangenden Prototyps kompensiert werden soll. Bei $t = 1s$ liegt eine Diskontinuität vor, die den stetigen Verlauf der Ausgangsgröße unterbricht. Während der nächsten $k = 3$ Zeitschritte extrapolieren die Kopplungsalgorithmen den Signalverlauf entsprechend seiner Dynamik vor der Diskontinuität weiter, bis nach Ablauf der Verzögerung der erste Signalwert nach der Diskontinuität am Eingang der Kopplungsalgorithmen anliegt. Wie basierend auf der Frequenzbereichsanalyse (s. Abb. 4.5 in Abschnitt 4.1.2) zu erwarten, führt der Einsatz der gezeigten Kopplungsalgorithmen an der Sprungstelle zu großen Fehlern, da das Signal dort hohe Frequenzanteile enthält. FOH überhöht den „Sprung“ an der Diskontinuität entsprechend Gl. (3-29) für einen Zeitschritt um Faktor vier. EROS3 führt gemäß Gl. (3-16) für fünf Zeitschritte zu einem Extrapolationsfehler, indem der eigentlichen Signalwert meist stark überschätzt, aber auch einmal unterschätzt wird. Der SMB überhöht die Diskontinuität im Zeitschritt nach der Diskontinuität weniger als die anderen Algorithmen, regt aber eine Schwingung im Bereich seiner Polstelle an (s. Gl. (3-35), die für mehr als 20 Zeitschritte im Signalverlauf erkennbar ist. Der nicht gezeigte EROS4-Algorithmus, beschrieben in Gl. (3-19), führt für sechs Zeitschritte zu durchschnittlich noch größeren Fehlern als EROS3 und der ebenfalls nicht gezeigte Standardalgorithmus ZOH aus Gl. (3-26) kompensiert die Latenzzeit in den stetigen Abschnitten des Signals nicht, überhöht aber auch nicht die Diskontinuität. Zusammengefasst weicht die Vorhersage von Kopplungsalgorithmen nach der Diskontinuität so lange deutlich von der Referenzlösung ab, wie sie gleichzeitig Signalwerte zu Zeitpunkten vor und nach der Diskontinuität für die Vorhersage verwenden (s. Tab. 3.1 in Abschnitt 3.3.3).

Zur Vermeidung der Extrapolationsfehler wird daher, sobald über das Verfahren zur Erkennung der Diskontinuitäten aus Abschnitt 5.3.1 eine solche erkannt wurde, der aktuell verwendete Kopplungsalgorithmus so umgeschaltet, dass nie gleichzeitig Signalwerte, die vor und nach der Diskontinuität liegen in die Vorhersage einer der linearen Kopplungsalgorithmen einfließt. Eine Möglichkeit dies zu erreichen, besteht darin immer dann auf die Extrapolation mit dem Standardalgorithmus ZOH umzuschalten, wenn innerhalb der letzten $m-1$ Zeitschritte eine Diskontinuität aufgetreten ist. Dabei ist $m$ die Anzahl vergangener Signalwerte, die der genutzte Kopplungsalgorithmus benötigt (vgl. Abschnitt 4.2.1). Dieses Verfahren funktioniert bei zustandsfreien Algorithmen wie FOH, EROS3 und EROS4, aber nicht bei Algorithmen wie dem SMB, welches einen internen Zustand beinhaltet und dessen Prädiktion daher theoretisch unendlich lange durch Signalwerte beeinflusst wird, die vor der Diskontinuität liegen. Den Zustand des SMBs nach jeder Diskontinuität zurückzusetzen ist ebenfalls keine Option, da der SMB eine sehr lange und vor allem nicht genau abschätzbare Einschwingzeit hat. Somit ist das entwickelte Verfahren zur Erkennung von Diskontinuitäten nicht in Kombination mit dem SMB und anderen zustandsbehafteten Kopplungsalgorithmen verwendbar, wodurch diese Algorithmen nicht auf Signale anwendbar sind, in deren Verlauf vereinzelt Diskontinuitäten auftreten können.

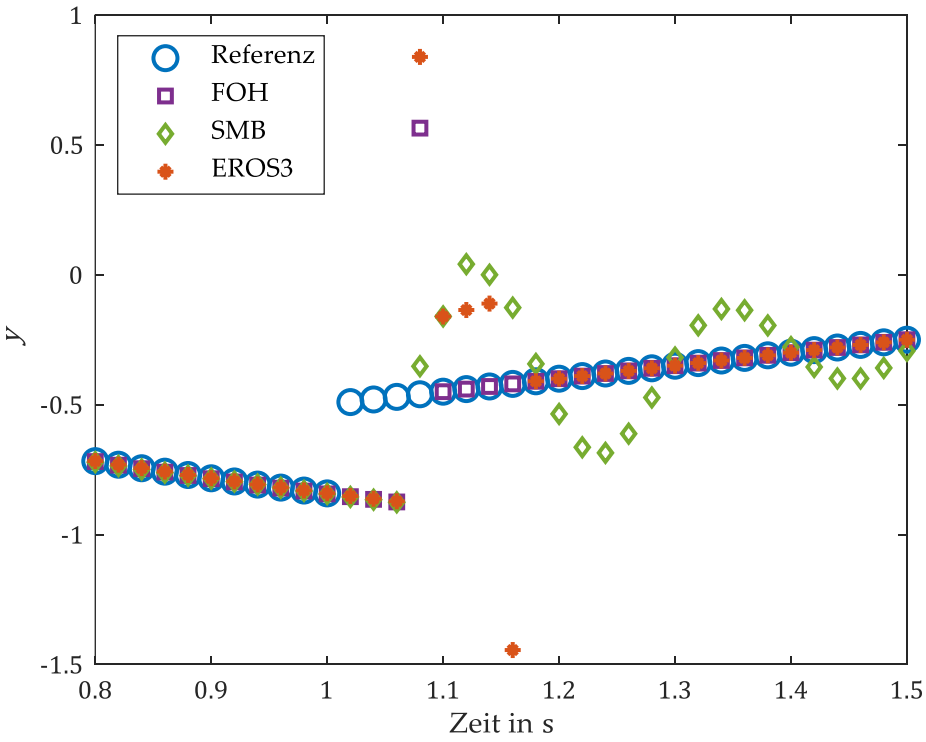


**Abb. 5.7** Verhalten verschiedener Kopplungsalgorithmen beim Auftreten einer Diskontinuität im Signalverlauf mit $k = 3$ Makroschritten Verzögerung ohne Verwendung des Verfahrens

Liegt der Frequenzbereich des Koppelsignals für die über $\Delta T$ und $k$ festgelegte Latenzzeit innerhalb des Gültigkeitsbereichs vom FOH (s. Tab. 4.1), kann der Umschaltvorgang bei der Verwendung von Algorithmen mit $m > 2$ noch weiter verbessert werden. Anstatt nach einer Diskontinuität für $m - 1$ Zeitschritte auf den Standardalgorithmus ZOH umzuschalten, wird in diesem Fall nur ein Zeitschritt ZOH verwendet und anschließend zunächst auf FOH umgeschaltet, bis $m - 1$ Zeitschritte nach der Diskontinuität mit dem ursprünglichen verwendeten Kopplungsalgorithmus fortgefahren wird. Dies ist möglich, da die Anzahl der verwendeten vergangenen Signalwerte beim FOH $m = 2$ ist und nicht von $k$ abhängt.

Diese Ausprägung des Umschaltvorgangs ist in Abb. 5.8 demonstriert. Das Referenzsignal entspricht exakt dem aus Abb. 5.7 nur wird nun in jedem Zeitschritt vor Anwendung des jeweiligen Kopplungsalgorithmus das Verfahren zur online Erkennung von Diskontinuitäten aus Abschnitt 5.3.1 ausgeführt und, falls dieses anschlägt, der Kopplungsalgorithmus umgeschaltet. Es ist zu beobachten, dass die Diskontinuität korrekt erkannt wird und im darauffolgenden Zeitschritt ZOH anstatt des ausgewählten Kopplungsalgorithmus für die Extrapolation des Signalverlaufs verwendet wird. Ist FOH als Kopplungsalgorithmus ausgewählt wird ab dem zweiten Zeitschritt nach der Diskontinuität wieder auf FOH zurückgeschalten. Wird EROS3 als Kopplungsalgorithmus eingesetzt, wird wie oben beschrieben zunächst ein Zeitschritt ZOH und anschließend weitere vier Schritte FOH verwendet, bevor auf EROS3 zurückgeschaltet wird. Der Übergang vom FOH zu EROS3 ist in der Abb. 5.8 nicht zu erkennen, da

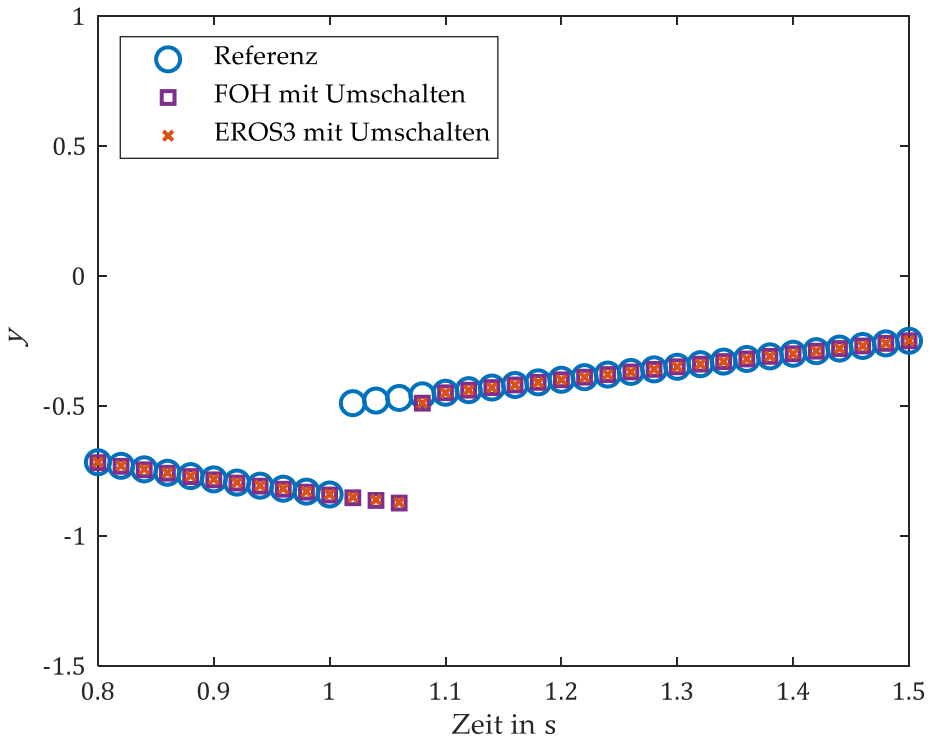


**Abb. 5.8** Verhalten verschiedener Kopplungsalgorithmen beim Auftreten einer Diskontinuität im Signalverlauf mit $k = 3$ Makroschritten Verzögerung mit Diskontinuitäten-Erkennung und Umschalten des Algorithmus

FOH und EROS3 für das dargestellte Signal sehr ähnliche Ergebnisse liefern. Insgesamt verringert sich durch Einsatz der Methode zur Erkennung von Diskontinuitäten der Extrapolationsfehler beim FOH und EROS3 in den Zeitschritten, nachdem die Diskontinuität am Eingang der Algorithmen vorliegt, deutlich.

Der Extrapolationsfehler während der $k$ Zeitschritte Latenzzeit durch die Datenübertragung zwischen den gekoppelten Prototypen nach einer Diskontinuität bleibt natürlich auch bei Verwendung der vorgestellten Methodik bestehen, da die Information über das Auftreten der Diskontinuität erst nach $k$ Zeitschritten am Eingang der Kopplungsmethodik vorliegt. Extrapolationsfehler in diesen Signalabschnitten lassen sich nur durch den Einsatz lernender Methoden verringern, indem das Auftreten der Diskontinuität selbst vorhergesehen wird (s. Abschnitt 3.5).

Zusammengefasst ist gezeigt, dass mithilfe der Erkennung von Diskontinuitäten, Kopplungsalgorithmen, die aufgrund einer großen Amplitudenverstärkung hoher Frequenzen nicht für die Extrapolation von Signalen mit Diskontinuitäten geeignet sind, trotzdem auf diese Signale anwendbar sind. Damit kann der der Kopplungsalgorithmus passend zu der Bandbreite des Signals in dessen stetigen Signalabschnitten gewählt werden, ohne große Kopplungsfehler nach Diskontinuitäten in Kauf nehmen zu müssen. Der Nutzen der Methodik an einem produktiv genutzten RVP wird in Abschnitt 6.2 gezeigt.

Bei der Kombination des Verfahrens zur Erkennung von Diskontinuitäten mit dem FFNN Kopplungsalgorithmus und dem dazugehörigen online Lernen Verfahren aus Abschnitt 3.5.2 sind zwei Fälle zu unterscheiden. Wenn der FFNN Algorithmus durch vorangegangenes online Lernen in der Lage ist die im Koppelsignal auftretenden Diskontinuitäten vorherzusagen und die Latenzzeiten entsprechend zu kompensieren (vgl. Abschnitt 3.5), ist eine gleichzeitige Verwendung der Diskontinuitätenerkennung nicht notwendig, da diese die Diskontinuität erst nach Ablauf der Latenzzeit erkennen kann und der Kopplungsalgorithmus zu diesem Zeitpunkt bereits auf die Diskontinuität reagiert hätte. In diesem Fall sollte auch das online Lernen ohne Veränderung fortgesetzt werden, um die Vorhersage des Signals inklusive der Diskontinuitäten zu verbessern. Wenn dagegen die Diskontinuitäten nicht durch den FFNN Kopplungsalgorithmus vorhersagbar sind, ist eine gleichzeitige Verwendung der Erkennung von Diskontinuitäten sinnvoll und der Umschaltevorgang nach einer erkannten Diskontinuität sollte genauso wie bei EROS3 erfolgen. Zusätzlich sollte in diesem Fall das online Lernen nach einer erkannten Diskontinuität reinitialisiert werden. Für den neuen Lernzyklus sollten dabei nur Trainingsdaten verwendet werden, die ausschließlich nach der Diskontinuität gewonnen wurden, um zu vermeiden, dass sich das FFNN auf ein Verhalten des Koppelsignals anpasst, welches aufgrund der Diskontinuität nicht mehr gültig ist.

# 6 Anwendung der Kopplungsmethodik

Die in den vorangegangenen Kapiteln dieser Arbeit entwickelte Methodik zur Kopplung räumlich verteilter realer und virtueller Prototypen wird in diesem Kapitel an einem produktiv genutzten RVP erprobt. Dabei wird gezeigt, dass die entwickelten Algorithmen in Kombination mit den Analyse- und Überwachungsmethoden den Einfluss der Netzwerkeffekte wie Latenzzeiten auf die Ergebnisse des RVPs kompensieren und damit dessen Nutzen erhöhen. Der untersuchte RVP stellt ein Hybridfahrzeug dar und besteht aus einem Motorprüfstand, einer realen ECU und einem Systemsimulationsmodell, in dem weitere notwendige Fahrzeugkomponenten modelliert sind. Lachenmaier et al. (2020) beschreiben den genauen Aufbau des RVPs sowie dessen Verwendung in der Automobilindustrie zur Entwicklung von Hybridstrategien. Ein Teil der in diesem Kapitel gezeigten Ergebnisse wurden bereits in dem im Rahmen dieser Arbeit entstandenen Beitrag (Baumann et al., 2024) veröffentlicht.

Damit bewertbar ist, ob die entwickelte Kopplungsmethodik den Einfluss der Latenzzeiten auf das Systemverhalten des realen RVPs tatsächlich kompensiert, muss dieser Einfluss der Latenzzeiten bekannt und quantifizierbar sein. Dazu wird basierend auf detaillierten und validierten Simulationsmodellen aller realen Prototypen des RVPs und den virtuellen Prototypen ein MIL Co-Simulationsmodell des RVPs erstellt. Dieses unterliegt im Gegensatz zum RVP nicht den Echtzeitanforderungen und ist daher sowohl mit einem idealen Datenaustausch ohne Latenzzeiten als auch mit künstlichen Latenzzeiten zwischen den Modellen simulierbar. Mithilfe dieses Co-Simulationsmodells wird identifiziert, wie sich die Latenzzeiten auf die Ergebnisse des RVPs auswirken, um anschließend bewerten zu können, ob die in dieser Arbeit entwickelte Methodik eben diese Effekte infolge der Latenzzeit auf das Verhalten des RVPs kompensiert.

Zunächst wird in Abschnitt 6.1 eine auf dem FMI-Standard basierende Toolkette vorgestellt, welche die einzelnen Aspekte der Methodik wie die Kopplungsalgorithmen aus Kap. 3 und die Überwachungsmethoden aus Kap. 5 implementiert, damit sie in typischen Aufbauten von RVPen mit möglichst geringem Aufwand anwendbar sind. Im daran anschließenden Abschnitt 6.2 wird der verwendete RVP und das dazugehörige Co-Simulationsmodell zunächst beschrieben und mithilfe der Methoden aus Kap. 4 analysiert. Anschließend wird mithilfe des Co-Simulationsmodells der Einfluss der Netzwerkeffekte auf das Systemverhalten des RVPs bestimmt, indem Kriterien identifiziert werden, mit denen eine Änderung des Systemverhaltens aus den Koppelsignalen ablesbar ist. Nach diesen Vorarbeiten werden die erzielten Er-

gebnisse bei der Ausführung des RVPs mit und ohne Einsatz der Kopplungsmethodik verglichen und es wird bewertet, ob der negative Einfluss der Netzwerkeffekte auf das Systemverhalten durch die Kopplungsmethodik verringert wird. Das Kapitel schließt mit einer Diskussion der gewonnenen Erkenntnisse für die einzelnen Aspekte der Kopplungsmethodik.

## 6.1 Implementierung der Kopplungsmethodik

Die Methodik zur Kopplung räumlich verteilter realer und virtueller Prototypen aus Kap. 3 und Kap. 5 wird für die Implementierung in mehrere Softwaremodule aufgeteilt, welche teilweise auch unabhängig voneinander nutzbar sind. Alle Softwaremodule sind mithilfe der Programmiersprache Python implementiert, da Python viele Bibliotheken mit Methoden des maschinellen Lernens bereitstellt, welche für die Implementierung des FFNNs aus Abschnitt 3.5 hilfreich sind.

In Abb. 6.1 sind die einzelnen Softwaremodule und ihre Wirkzusammenhänge bei der Anwendung der Kopplungsmethodik zur Kompensation der Latenzzeit eines Koppelsignals $y$ zum Zeitpunkt $t_n$ dargestellt. Alle Module, die in jedem Makro- und Mikroschritt ausgeführt werden müssen, sind, wie in Abschnitt 6.1.1 näher erläutert wird, als Functional Mock-Up Unit (FMU) bereitgestellt. Außerhalb der FMU befindet sich nur der Server für das online Lernen des FFNN-Kopplungsalgorithmus. Die Funktionsweise des Servers sowie dessen Kommunikation mit dem entsprechenden Client in der FMU wird in Abschnitt 6.1.2 aufgezeigt.

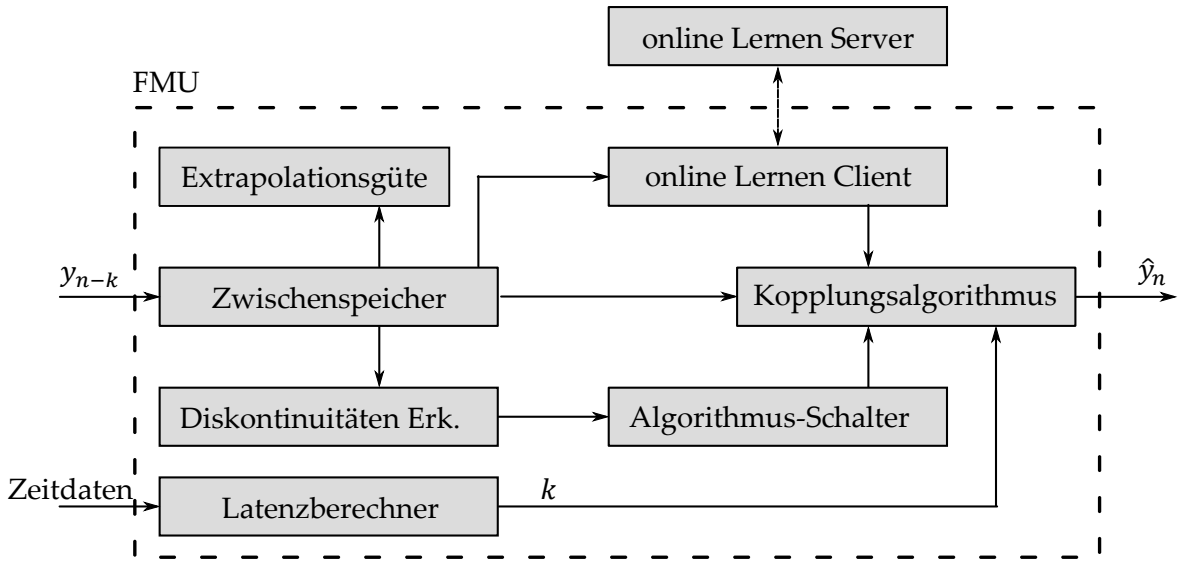


**Abb. 6.1** Softwaremodule der entwickelten Methodik zur Kopplung räumlich verteilter realer und virtueller Prototypen sowie deren Wirkzusammenhänge

Innerhalb der FMU enthält das Kopplungsalgorithmus-Modul das Kernstück der Methodik. Es implementiert die diskutierten Kopplungsalgorithmen aus Abschnitt 3.3, das FFNN aus Abschnitt 3.5 und Algorithmen, deren Gewichtungsparameter sich aus der Optimierung aus Abschnitt 4.2 ergeben. Zu jeder Zeit ist einer dieser Algorithmen aktiv und kompensiert die

aktuelle Latenzzeit durch Extrapolation des Signalverlauf um $k$ Makrozeitschritte, wodurch der Signalwert $\hat{y}_n$ geschätzt wird. Die dafür notwendigen vergangenen Signalwerte werden vom Zwischenspeicher-Modul bereitgestellt. Weiterhin ist in der FMU das Verfahren zur Erkennung von Diskontinuitäten (für Definition s. Abschnitt 3.1) im Koppelsignal aus Abschnitt 5.3 im entsprechenden Modul implementiert. Wie in Abschnitt 5.3.2 vorgeschlagen ist es mit einem Modul verbunden, welches nach einer erkannten Diskontinuität den Kopplungsalgorithmus kurzzeitig umschaltet, um große Kopplungsfehler im Signalabschnitt kurz nach einer Diskontinuität zu verhindern. Das Verfahren zur Bewertung der Extrapolationsgüte ist ebenfalls in einem eigenen Modul implementiert, sodass die Ausführung eines RVPs bei Verletzung der Güte abgebrochen werden kann. Außerdem befindet sich der Latenzberechner innerhalb der FMU, welcher basierend auf den Zeitdaten der Prototypen die aktuelle Anzahl an Makrozeitschritten Verzögerung $k$ zur Laufzeit bestimmt, wodurch eine genaue Kompensation der Latenzzeit sowie eine Reaktion auf variable Latenzzeiten möglich ist (s. Abschnitt 5.1). Die Ein- und Ausgänge aller Softwaremodule werden zur Laufzeit mithilfe eines Logging-Moduls gespeichert, welches in Abb. 6.1 aus Gründen der Übersichtlichkeit nicht dargestellt ist.

In jedem Zeitschritt führt die FMU eine do-step-Methode aus, welche nacheinander Methoden der einzelnen Module aufruft. Ist der aktuelle Zeitschritt ein Makrozeitschritt erfolgt zuerst die Bestimmung der aktuellen Latenzzeit. Diese ist entweder als konstant parametriert oder wird, falls die gekoppelten Prototypen zeitsynchronisiert sind, aus den zur Verfügung stehenden Zeitdaten berechnet (s. Abschnitt 5.1.2). Anschließend wird die Güte der vergangenen Extrapolation bewertet und gespeichert. Je nach Implementierung kann bei einer Verletzung der Güte entweder eine Warnung ausgegeben oder die Ausführung des RVPs automatisch abgebrochen werden. Danach wird der aktuelle Signalverlauf auf eine Diskontinuität untersucht und der Kopplungsalgorithmus entsprechend gewählt. Am Ende der do-step-Methode erfolgt die eigentliche Kompensation der Latenzzeit im Koppelsignal durch den ausgewählten Kopplungsalgorithmus mittels der Berechnung des Signalwerts $\hat{y}_n$. Ist der aktuelle Zeitschritt ein Mikroschritt wird lediglich der Kopplungsalgorithmus zur Bestimmung des Signalwerts für diesen Mikroschritt ausgeführt. Das online Lernen zur Anpassung des Kopplungsalgorithmus ist zu rechenaufwändig, um es in jeder Zeitschrift zu wiederholen, weshalb es parallel zur do-step-Methode in einem separaten Prozess stattfindet. Dies ist in Abschnitt 6.1.2 näher beschrieben.

### 6.1.1 Bereitstellung der Kopplungsmethodik mithilfe des FMI-Standards

Die einzelnen Prototypen in RVPen liegen typischerweise in verschiedenen Tools vor, die zur Laufzeit miteinander gekoppelt werden und auf diese Weise den Datenaustausch zwischen den Prototypen ermöglichen. Diese Tools sind zum einen meist auf bestimmte Domänen spe-

zialisierte Simulationswerkzeuge, die für die Implementierung und Ausführung der virtuellen Prototypen verwendet werden, und zum anderen Steuerungssysteme für Prüfstände, welche die Kommunikation mit den realen Prototypen ermöglichen. Damit die in dieser Arbeit entwickelte Kopplungsmethodik mit geringem Integrationsaufwand für möglichst viele RVPen nutzbar ist, muss sie in den verschiedenen Tools verwendbar sein. Dies wird durch die Verwendung einer standardisierten Schnittstelle erreicht.

Der weitverbreitetste Standard, der von den meisten kommerziellen Simulationswerkzeugen und Steuerungssystemen für Prüfstände unterstützt wird, ist das Functional Mock-Up interface (FMI) (vgl. Abschnitt 2.1.1). Obwohl das FMI primär für die Kopplung von Simulationsmodellen gedacht ist, eignet es sich hervorragend für die toolunabhängige Bereitstellung der Kopplungsmethodik in dieser Arbeit. Dazu wird wie oben beschrieben das Zusammenspiel der einzelnen Softwaremodule als FMU gekapselt, welche in jedem Zeitschritt die do-step Methode ausführt. Die FMU enthält somit die gesamte Kopplungsmethodik und ist in jedem Tool, welches das FMI unterstützt einsetzbar. Die Eingänge der FMU sind, wie oben in Abb. 6.1 gezeigt, der neuste zum aktuellen Zeitpunkt verfügbare Signalwert eines Koppelsignals $y_{n-k}$ und falls vorhanden die Zeitdaten, aus denen die aktuelle Latenzzeit zwischen den gekoppelten Prototypen bestimmbar ist. Der geschätzte zukünftige Signalwert $\hat{y}_n$ bildet den Ausgang der FMU. Da vom importierenden Tool kein numerischer Löser für die Berechnung der do-step-Methode der Kopplungsmethodik nötig ist, wird die Variante FMI for co-simulation für die FMU gewählt. Das importierende Tool stellt der FMU somit lediglich die Eingänge zur Verfügung und fragt für den jeweiligen Makro- oder Mikroschritt den Ausgang der FMU ab. Mittels der ebenfalls ausgewählten direct-feedthrough Option wird sichergestellt, dass bei der Verwendung der FMU im importierenden Tool keine zusätzliche Verzögerung um einen Makrozeitschritt entsteht.

Die verbleibende Herausforderung ist die Implementierung der FMU, da eine FMU in der Programmiersprache C implementiert sein muss, die Softwaremodule der Kopplungsmethodik allerdings in Python vorliegen. Zur Lösung dieses Implementierungsproblems gibt es verschiedene Möglichkeiten. Thierry S. Nouidui and Michael Wetter (2018) stellen die SimulatorToFMU Software zur Verfügung, bei der die in C implementierten Methoden der FMU unter Verwendung der C-API für Python mit den entsprechenden in Python implementierten Methoden kommunizieren. Einen ähnlichen Ansatz verfolgt das open source Paket Modelica-ExternalLibrary. Bei beiden Ansätzen wird der Python-Prozess, welcher von der FMU gerufen wird, in einem Sub-Task des Tools ausgeführt, das die FMU importiert. Dabei wird zur Laufzeit die Python-Ausführungsumgebung des importierenden Tools verwendet, wodurch eine korrekte Ausführung des Python-Codes nur gewährleistet ist, wenn diese Ausführungsumgebung alle verwendeten Python-Bibliotheken in der geforderten Version enthält. Diese Abhängigkeit vom importierenden Tool widerspricht der allgemeinen Verwendbarkeit der Kopplungsmethodik, welche durch die Verwendung des FMI-Standards erreicht werden soll.

Daher wird für die Kapselung der Kopplungsmethodik als FMU in dieser Arbeit ein anderer Ansatz implementiert. Die in C implementierte FMU eröffnet zu Beginn der Ausführung des RVPs eine Socketverbindung mit dem Python-Code der Kopplungsmethodik, welcher in einem getrennten Prozess berechnet wird. Während der Ausführung des RVPs findet der Datenaustausch zwischen der FMU und dem Kopplungsmethodik-Prozess über diese Socketverbindung statt. Die Konvertierung zwischen den C-Datentypen und Python-Datentypen der über die Socketverbindung ausgetauschten Werte erfolgt mit Hilfe des Python Pakets ctypes. Die Vorteile dieser Lösung sind zum einen, dass der Python-Code der Kopplungsmethodik in seinem eigenen Prozess ausgeführt wird und somit nicht den Limitierungen der Python-Ausführungsumgebung des Tools unterliegt, welches die FMU importiert. Zum anderen handelt es sich hierbei um eine sehr leichtgewichtige Lösung. Für das Bauen der FMU wird lediglich ein C-Compiler benötigt. Die Ausführungsgeschwindigkeit dieser Toolkette ist durch die Trennung der Prozesse und die direkte Kommunikation zwischen ihnen via der Socket-Kommunikation höher als bei den anderen beiden Ansätzen.

Die Rechenzeit, die zur einmaligen Ausführung der FMU inklusive der Socket-Kommunikation und der Berechnung der darin gekapselten Kopplungsmethodik benötigt wird, liegt mit der in dieser Arbeit verwendeten Standardhardware bei unter $2ms$. Bei Anwendung der FMU auf die Eingangssignale realer Prototypen wirkt diese Rechenzeit wie eine zusätzliche Latenzzeit. Dadurch ist die Kopplungsmethodik in dieser Implementierung nur sinnvoll einsetzbar, wenn die aufgrund sonstiger technischer Randbedingungen bedingte Latenzzeit deutlich höher als die Rechenzeit der FMU ist. Jedoch ist eine Verwendung der FMU für Eingangssignale virtueller Prototypen auch bei geringerer Latenzzeit sinnvoll, wenn sowohl die Ausführung der FMU als auch die Integration eines Makrozeitschritts des Simulationsmodells des virtuellen Prototyps innerhalb der Dauer des Makroschritts in Echtzeit möglich ist (s. Abschnitt 6.2).

Es ist davon auszugehen, dass eine Implementierung der gesamten Kopplungsmethodik in einer kompilierten Sprache und eine direkte Einbettung des kompilierten Codes in die FMU die Rechenzeit deutlich verringern würde. Dies wäre zum Beispiel in der Programmiersprache C mit dem FMU SDK von Qtronic (2018) oder in Sprache Julia basierend auf der Software FMI.jl von Thummerer, Kircher und Mikelsons (2021) möglich.

### 6.1.2 Client-Server-Architektur für das online Lernen

Der Kern der in Abschnitt 3.5.2 vorgestellten Methode zum online Lernen des FFNN-Kopplungsalgorithmus besteht darin das FFNN während der Ausführung eines RVPs auf den aktuellen Signalverlauf anzupassen, um die Kompensation der Latenzzeiten zu verbessern. Sobald eine definierte Lernbedingung erfüllt ist, werden dazu die Parameter des FFNNs durch überwachtes Lernen mit vergangenen Signalwerten als Trainingsdaten aktualisiert. Da die für

den Lernvorgang benötige Berechnungsdauer in der Regel deutlich größer als die Zeitschrittweite ist, mit der die Kopplungsmethodik zyklisch ausgeführt wird, ist der Lernvorgang selbst nicht innerhalb des als FMU gekapselten Python-Codes implementiert (s. Abb. 6.1). Stattdessen ist das online Lernen mittels Client-Server Architektur realisiert. Dabei bietet ein Server, der unabhängig von der restlichen Kopplungsmethodik in einem separaten Prozess ausgeführt wird, einen Dienst zum Training der FFNNs an. Der online Lernen Client, der als eines der Softwaremodule vorliegt, die als FMU gekapselt sind, nutzt diesen Dienst, indem er über eine Socketverbindung Anfragen an den Server stellt.

Die Aufgabenverteilung und Kommunikation zwischen Client und Server für das online Lernen ist in Abb. 6.2 gezeigt. In der Initialisierungsphase der FMU stellt der Client dem Server die Konfiguration der Architektur des FFNNs zur Verfügung (s. Abschnitt 3.5.1). Während der Ausführung des RVPs befindet sich der Client in einem von drei Zuständen und überprüft in jedem Zeitschritt, wenn er von der do-step Methode der FMU angefragt wird, ob die Bedingung für einen Zustandswechsel erfüllt ist.

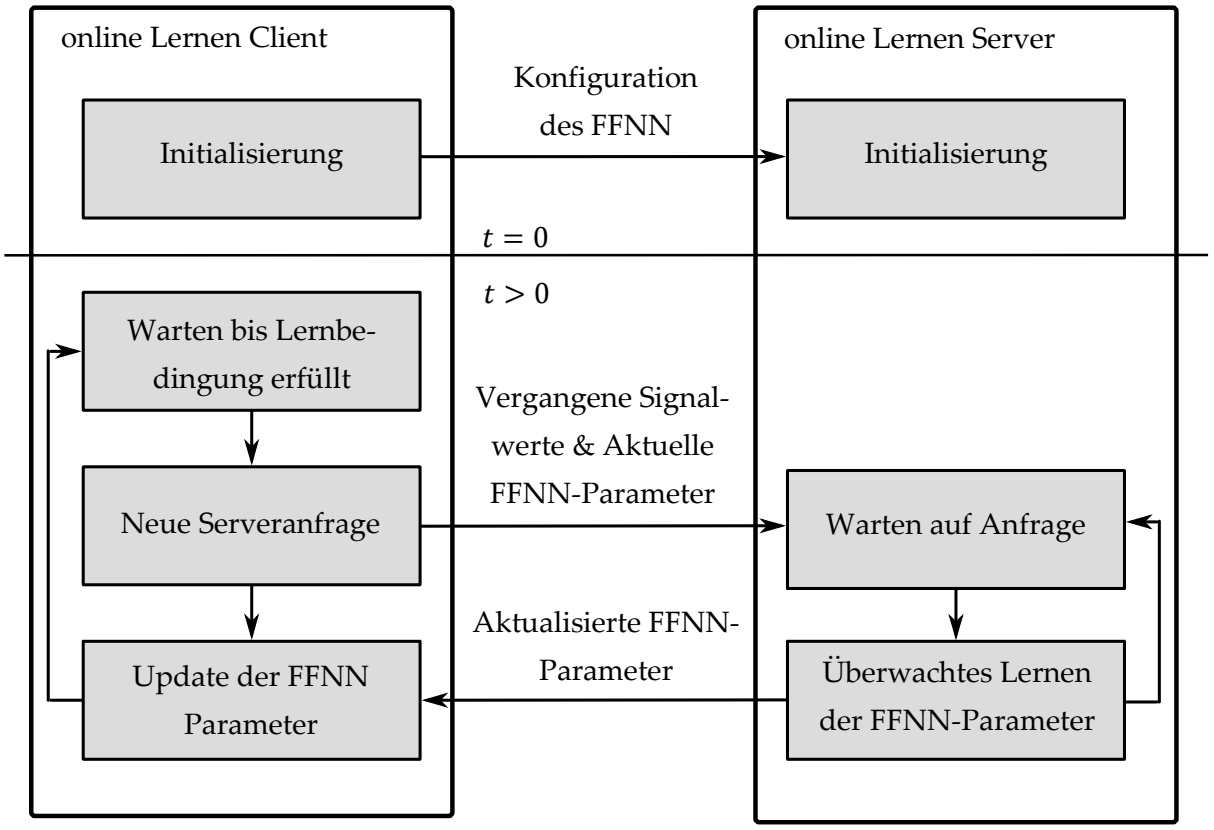


**Abb. 6.2** Ablauf der implementierten Server-Client Kommunikation für das online Lernen eines FFNNs

Im oberen Zustand, dargestellt in Abb. 6.2, überprüft der Client in jedem Zeitschritt, ob die Lernbedingung erfüllt ist. Ist dies der Fall stellt der Client eine Anfrage zur online Anpassung des FFNNs an den Server und versendet dazu die Historie des Koppelsignals, auf die das FFNN angewandt wird, sowie die aktuellen Parameter des FFNNs. In den darauffolgenden

Zeitschritten befindet sich der Client im mittleren Zustand und überprüft, ob der Server den aktuellen Lernvorgang bereits abgeschlossen und aktualisierte Parameter für das FFNN kommuniziert hat. Falls ja, wechselt der Client in den unteren Zustand und setzt die neuen Parameter des für die Vorhersage des Signalverlauf verwendeten FFNN-Kopplungsalgorithmus, bevor er wieder in den oberen Zustand wechselt und ein neuer Lernzyklus beginnt. Der Server dagegen wird nicht durch die Zeitschritte der FMU getriggert, da er sich außerhalb dieser befindet. Stattdessen wartet er auf Anfragen von einem Client zum überwachten Lernen eines FFNNs und arbeitet diese nacheinander ab.

Diese Implementierung als Client-Server Architektur erlaubt eine Verwendung des online Lernen Algorithmus ohne die Berechnungsdauer der do-step Methode der FMU zu erhöhen. Die Berechnungsdauer aller in Abb. 6.2 dargestellten Aufgaben, die der Client pro Lernzyklus erfüllen muss, sind im Vergleich zu der Berechnungsdauer der anderen Teile der Kopplungsmethodik um Größenordnungen geringer und die rechenintensiven Berechnungen des Servers beinträchtigen die Berechnungsdauer der FMU nicht, da sie in einem separaten Prozess parallel durchgeführt werden.

## 6.2 Anwendung auf produktiv genutzten Real-Virtuellen Prototyp

Der Aufbau und der Verwendungszweck des produktiv genutzten RVPs auf den im Folgenden die in dieser Arbeit entwickelte Methodik zur Kompensation der Latenzzeiten angewandt wird, ist detailliert in Lachenmaier et al. (2020) beschrieben. Zusammengefasst handelt es sich bei dem Aufbau um ein real-virtuelles Fahrzeug mit einer P2.5 Hybridtopologie als Antriebskonzept. Bei diesem Antriebskonzept ist im Fahrzeug zusätzlich zu einem Verbrennungsmotor ein elektrischer Motorgenerator seitlich am Doppelkupplungsgetriebe verbaut, der sowohl rein elektrisches Fahren als auch regeneratives Bremsen ermöglicht. Der Vorteil dieser Hybridtopologie ist, dass der Verbrennungsmotor und der elektrische Motorgenerator unterschiedliche Drehmomentpfade nutzen und dadurch näher an ihrem jeweiligen optimalen Wirkungsgrad betrieben werden können. Zusätzlich ist es mithilfe des Doppelkupplungsgetriebes möglich in Fahrphasen in denen einer der beiden Antriebe allein verwendet wir, den jeweils anderen Motor zur Vermeidung von Schleppverlusten abzukoppeln.

Der RVP, der das Hybridfahrzeug darstellt, besteht aus zwei realen und einem virtuellen Prototyp. Bei den beiden realen Prototypen handelt es sich zum einen um einen Verbrennungsmotor inklusive Abgasnachbehandlung, der sich auf einem Motorprüfstand befindet, und zum anderen um ein reales Motorsteuergerät, welches mit dem Motor über einen Controller-Area-Network-Bus (CAN-Bus) verbunden ist. Der virtuelle Prototyp umfasst ein Simulationsmodell, in dem die Fahrzeugdynamik, der elektrische Motorgenerator, die Batterie, der Antriebsstrang inklusive Doppelkupplungsgetriebe sowie weitere Fahrzeugkomponenten wie beispielsweise elektrische Verbraucher des Fahrzeugs modelliert sind. Außerdem beinhaltet

das Simulationsmodell einen virtuellen Fahrer, der das real-virtuelle Fahrzeug entsprechend eines gegebenen Geschwindigkeitsverlaufs regelt.

Abb. 6.3 zeigt schematisch den Aufbau des untersuchten RVPs inklusive der signifikanten Koppelsignale zwischen den einzelnen Prototypen. Das Simulationsmodell wird unter weichen Echtzeitbedingungen auf einem Simulations-PC mit Windows-Betriebssystem ausgeführt, der sich räumlich vom Motorprüfstand getrennt in einem anderen Raum befindet und über ein Netzwerk mit dem Steuerungssystem des Motorprüfstands verbunden ist. Dieses Echtzeitsystem steuert das Dynamometer, welches die Gegenlast für den Verbrennungsmotors darstellt, und ermöglicht den Datenaustausch mit diesem. Weiterhin stellt es die CAN-Bus Schnittstelle des Verbrennungsmotors zum Steuergerät bereit.

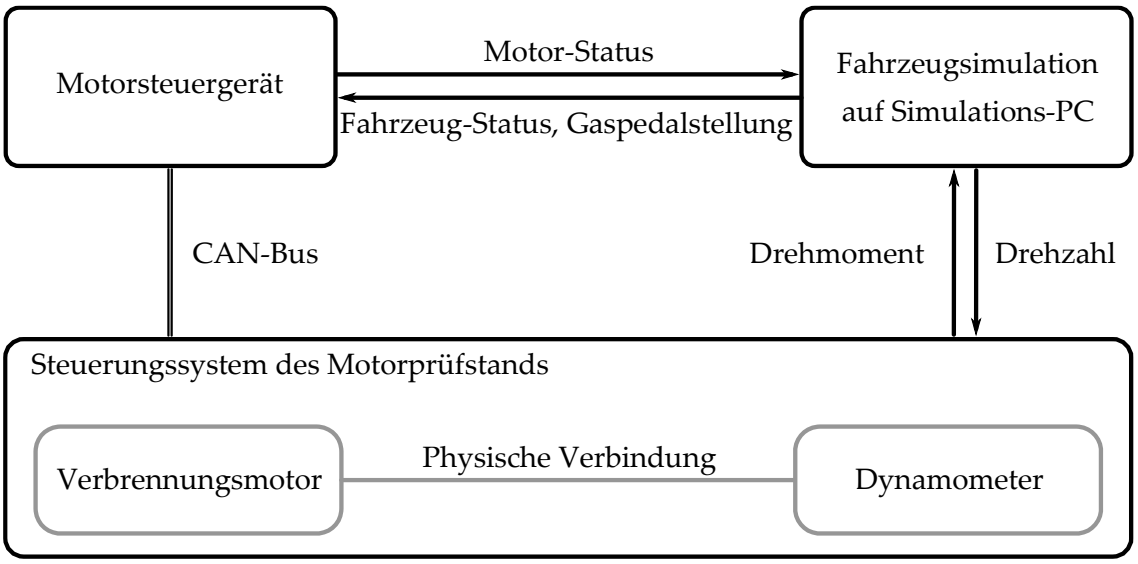


**Abb. 6.3** Schematischer Aufbau des verwendeten RVPs mit Fokus auf den Datenaustausch nach Lachenmaier et al. (2020)

Der RVP ist bei der Robert Bosch GmbH zur Entwicklung von Hybridstrategien unter Berücksichtigung des realen Kraftstoffverbrauchs sowie realer $CO_2$- und Schadstoffemissionen im Einsatz. Die Ergebnisse des RVPs wurden mit denen aus realen Testfahrten verglichen, wobei das qualitative Verhalten des RVPs als sehr gut bewertet wurde (Lachenmaier et al., 2020).

Für die untersuchte P2.5 Hybridtopologie belaufen sich die durchschnittlichen potenziellen Kraftstoffeinsparungen verglichen mit einem rein konventionellen Antriebskonzept auf etwa 19% (Melaika, Mamikoglu und Dahlander, 2019). Damit liegen die Optimierungsmöglichkeiten durch Anpassungen an der Hybridbetriebsstrategie in Bezug auf den Kraftstoffverbrauch und die Energieerzeugung des Verbrennungsmotors im einstelligen Prozentbereich (Benajes et al., 2019). Um basierend auf dem untersuchten RVP optimale Betriebsstrategien definieren zu können, ist es daher entscheidend, dass die übertragenen Antriebsenergien des Verbrennungsmotors zwischen den gekoppelten Prototypen nicht aufgrund von Netzwerkeffekten

verfälscht werden. Das Ziel der Anwendung der entwickelten Kopplungsmethodik auf diesen RVP besteht daher darin, die Latenzzeiten zu kompensieren und somit Energieerhaltungsverletzungen in der Kopplung zwischen den gekoppelten Prototypen zu minimieren, um auf diese Weise die Glaubwürdigkeit der mithilfe des RVPen entwickelten Hybridstrategien zu verbessern.

### 6.2.1 Analyse des Real-Virtuellen Prototyps

Bevor die entwickelte Kopplungsmethodik auf den RVP angewandt werden kann, muss dieser zunächst analysiert werden. Zum einen müssen die technischen Randbedingungen des Datenaustauschs zwischen den gekoppelten Prototypen (s. Abschnitt 2.2.1) bestimmt werden und zum anderen ist eine Identifizierung der Parameter notwendig, welche zur optimalen Auslegung eines Kopplungsalgorithmus gemäß Abschnitt 4.2 benötigt werden. Daher wird im Folgenden zunächst bewertet, bei welchen Koppelsignalen der gekoppelten Prototypen eine Kompensation der Latenzzeiten notwendig und sinnvoll ist. Anschließend werden basierend auf den gemessenen Latenzzeiten und weiteren technischen Randbedingungen benötigte Parameter wie die Makroschrittweite $\Delta T$ und die daraus resultierende Anzahl an Makroschritten Verzögerung $k$ sowie die Bandbreite der Koppelsignale bestimmt.

**Analyse der Schnittstellen zwischen den gekoppelten Prototypen**

Wie in Abb. 6.3 dargestellt, ist jeder der drei Prototypen des RVPs mit den jeweils anderen beiden vernetzt und tauscht Signale mit ihnen aus. Die drei sich ergebenden Schnittstellen sind bei einem realen Fahrzeug auf verschiedene Weise realisiert, wodurch sich der Einfluss der Latenzzeiten auf das Systemverhalten des RVPs für jede Schnittstelle stark unterscheidet.

Die Schnittstelle zwischen dem Fahrzeugsimulationsmodell und dem Verbrennungsmotor, über die in die eine Richtung die Drehzahl des Motors und in die andere Richtung das Drehmoment an der Antriebswelle ausgetauscht wird, ist bei einem realen Fahrzeug durch die Antriebswelle realisiert. Abgesehen von den Torsionen in der Welle ist diese physische Verbindung im realen Fahrzeug, das durch den RVP abgebildet wird, frei von Latenzzeiten. Solche Schnittstellen, bei denen entgegen gerichtete physikalische Koppelsignale zusammen einen Energiefluss beschreiben (power bonds), der beim entsprechenden realen System über eine physische Verbindung stattfindet, sind sogar in Co-Simulationen ohne Echtzeitanforderung eine Herausforderung. Selbst kleine Kopplungsfehler, die bei einer Co-Simulation aufgrund der Abtastung der ausgetauschten Signale entstehen, führen zu einer Zu- oder Entnahme von Energie innerhalb der Schnittstelle, wodurch sich die Gesamtenergiemenge innerhalb des Systems verändert und somit das Systemverhalten beeinflusst (Sadjina et al., 2017). Beim untersuchten RVP ist diese Schnittstelle besonders kritisch, da über sie die gesamte, durch den Verbrennungsmotor bereitgestellte Antriebsenergie des Fahrzeugs übertragen wird und Fehler in

dieser die Hybridstrategie direkt beeinflusst, welche der hauptsächliche Untersuchungsgegenstand des RVPs ist. Damit ist es essentiell die Kopplungsfehler in der Schnittstelle zwischen Simulationsmodell und Verbrennungsmotor bei diesem RVP durch einen Einsatz der entwickelten Kopplungsmethodik zu verringern.

Bei den Koppelsignalen der anderen beiden Schnittstellen zwischen den gekoppelten Prototypen ist keine Kompensation durch den Einsatz der Kopplungsmethodik notwendig. Zwischen den beiden realen Prototypen Verbrennungsmotor und Steuergerät findet der Datenaustausch beim untersuchten RVP genau wie beim abgebildeten realen Fahrzeug per CAN-Bus statt. Somit entsprechen die beim RVP auftretenden Verzögerungen und Quantisierungsfehler denjenigen im realen Fahrzeug, weshalb hier keine Kompensation der Latenzzeiten angebracht ist. Zwischen Steuergerät und Simulationsmodell werden nur Statussignale des Fahrzeugs bzw. des Verbrennungsmotors übertragen, welche nicht direkt voneinander abhängen, weshalb an dieser Schnittstelle aufgrund von Latenzzeiten keine Rückkopplungseffekte auf das gesamte Systemverhalten zu erwarten ist. Außerdem wird über diese Schnittstelle die aktuelle Gaspedalstellung, welche der virtuelle Fahrer zur Geschwindigkeitsregelung vorgibt, vom Simulationsmodell an das Steuergerät übertragen. Allerdings ist die Zeitkonstante dieser Regelung, wie in Abschnitt 6.2.2 beobachtet wird, so groß, dass die Latenzzeiten der Übertragung diese nicht nennenswert beeinflussen und somit auch hier keine Kompensation der Latenzzeiten notwendig ist.

Somit fokussiert sich die Anwendung der Methodik auf die Koppelsignale zwischen Verbrennungsmotor und Simulationsmodell. Dadurch sind bei den mit dem RVP gefahrenen virtuellen Fahrzyklen nur die Phasen interessant, bei denen das Fahrzeug nicht rein elektrisch fährt, da ansonsten der Verbrennungsmotor abgekoppelt ist und keinen Einfluss auf das Fahrzeugverhalten hat.

**Latenzzeiten und Makroschrittweite**

Ein wichtiger Parameter für die Verwendung der Kopplungsmethodik ist die Makroschrittweite des Datenaustauschs $\Delta T$, der größer oder gleich der Mikroschrittweite der einzelnen Prototypen und in Abhängigkeit der erwarteten Latenzzeit $k$ zwischen den gekoppelten Prototypen gewählt werden sollte (s. Abschnitt 5.1.1). Wie von Lachenmaier et al. (2020) beschrieben, liegt die kleinste mögliche Schrittweite von Fahrzeugsimulationsmodell und Verbrennungsmotor jeweils bei $0{,}001s$ und die des Steuergeräts bei $0{,}01s$.

Zur Bestimmung der Latenzzeit an der kritischen Schnittstelle zwischen Verbrennungsmotor und Simulationsmodell, an der die Kopplungsmethodik eingesetzt werden soll, wird zunächst die Umlaufzeit gemessen. Diese liegt für den Datenaustausch vom Simulations-PC mit Windows-Betriebssystem, welcher das Simulationsmodell ausführt, hin zur Steuerung des Motorprüfstands und zurück zum Simulations-PC zwischen 0,007s und 0,008s. Zusätzlich zur

Umlaufzeit wirkt die aufgrund von Rauscheinflüssen notwendige Filterung des am Verbrennungsmotor gemessenen Drehmoments an der Antriebswelle wie eine zusätzliche Latenzzeit (s. Abschnitt 2.2.1). Das Tiefpassfilter wird vom Steuerungssystem des Prüfstands bereitgestellt und ist optimal auf die verwendete Messtechnik und den Frequenzbereich des zu messenden Drehmoments ausgelegt, weshalb es für die Untersuchungen in dieser Arbeit nicht verändert wurde. Die durch das Filter bedingte Phasenverschiebung wirkt wie eine zusätzliche Latenzzeit von 0,05s auf das gemessene Drehmoment.

Damit liegt die sich durch die technischen Randbedingungen ergebende Latenzzeit in Summe im Bereich von $\tau(t) = 0{,}0575s \pm 0{,}0005s$ für die Datenübertragung vom Simulationsmodell zum Verbrennungsmotor und zurück. Nach den Überlegungen aus Abschnitt 5.1.1 wird die Makroschrittweite auf $\Delta T = 0{,}01s$ festgelegt, wodurch sich für die Anzahl an Makroschritten Verzögerung $k = 6$ ergibt. Die zu kompensierende Umlaufverzögerung liegt somit bei $k\Delta T = 0{,}06s$, und ist aufgrund der geringen Variabilität der Umlaufzeit konstant (s. Abschnitt 5.1.1).

**Start der Ausführung**

Wie in Abschnitt 5.1.3 beschrieben, ergibt sich aus der Startreihenfolge und der resultierenden Ausrichtung der Zeitraster der Prototypen zueinander in welcher Übertragungsrichtung welcher Anteil der gesamten Latenzzeit wirkt und an welchen Stellen somit eine Kompensation der Latenzzeiten sinnvoll ist. Da keine exakte Zeitsynchronisation des Steuergeräts, des Motorprüfstands und des Simulation-PCs implementiert ist, scheidet ein gleichzeitiger Start aller Prototypen aus. Stattdessen wird der RVP gestartet, indem das Simulationsmodell gestartet wird. Um eine Vergleichbarkeit der Ergebnisse zwischen verschiedenen virtuellen Testfahrten sicherzustellen, befindet sich der Verbrennungsmotor zum Zeitpunkt des Starts des Simulationsmodell bereits im Leerlauf und hat eine vordefinierte Betriebstemperatur erreicht. Das Steuergerät ist ebenfalls bereits aktiv, da sich auf ihm die Leerlaufregelung des Motors befindet. Somit definiert sich der Startzeitpunkt $t = 0$ für den Verbrennungsmotor durch das Eintreffen des Startbefehls zusammen mit den ersten Daten der Koppelsignale, welche vom Simulationsmodell berechnet werden. Damit wirkt für diese Schnittstelle die gesamte Verzögerungszeit von $k\Delta T = 0{,}06s$ auf die Übertragungsrichtung von Verbrennungsmotor zum Simulationsmodell (s. Abschnitt 5.1.3). Das Drehmoment an der Antriebswelle ist das einzige Koppelsignal in dieser Übertragungsrichtung und somit das einzige Signal des RVPs an dem die entwickelte Kopplungsmethodik zur Kompensation der Latenzzeiten eingesetzt werden muss (s. Abb. 6.3).

Die Kompensation der Latenzzeiten im Signal des Drehmoments erfolgt am Eingang des Simulationsmodells und kann dort aufgrund der in Abschnitt 6.1 aufgezeigten Kapselung der implementierten Methodik als FMU und der Unterstützung des FMI-Standards des verwendeten Simulationswerkzeugs direkt und in vollem Umfang verwendet werden. Trotz der in jedem Zeitschritt vorangehenden Berechnung der Kopplungsmethodik ist die Integration des

Simulationsmodells zuverlässig in Echtzeit möglich, wodurch die benötigte Rechenzeit der Kopplungsmethodik nicht wie eine zusätzliche Latenzzeit wirkt.

**Frequenzspektrum des Drehmoments an der Antriebswelle**

Damit für das betrachtete Drehmomentsignal an der Antriebswelle in Abschnitt 6.2.3 ein Kopplungsalgorithmus nach der Methodik aus Abschnitt 4.2 optimal ausgelegt werden kann, muss das Frequenzspektrum des Signals bekannt sein. Dieses ist in Abb. 6.4 dargestellt und ist mittels Fourier Analyse aus einer Messung des Signals berechnet, bei der der Verbrennungsmotor auf dem Prüfstand lange in verschiedenen Lastbereichen betrieben wurde. Statt wie üblich über der Frequenz ist dieses Spektrum stattdessen über dessen Anteil an der Nyquistfrequenz für die festgelegte Makroschrittweite von $\Delta T = 0{,}01s$ aufgetragen. Die größten Amplituden gehören zum Nutzsignal und liegen im Bereich von unter 3% der Nyquistfrequenz. Trotz des oben beschriebenen Filters verbleiben aufgrund der Messtechnik Störfrequenzen im Signal, welche ungefähr im Bereich zwischen 10-15% der Nyquistfrequenz liegen. Nach Tab. 4.1 in Abschnitt 4.1.2 sind somit alle untersuchten generischen Kopplungsalgorithmen ungeeignet die hier notwendigen $k = 6$ Makroschritte Latenzzeit zu kompensieren, da die Störfrequenzen und je nach Algorithmus auch Teile des Nutzsignals außerhalb des Gültigkeitsbereichs der Algorithmen liegen.

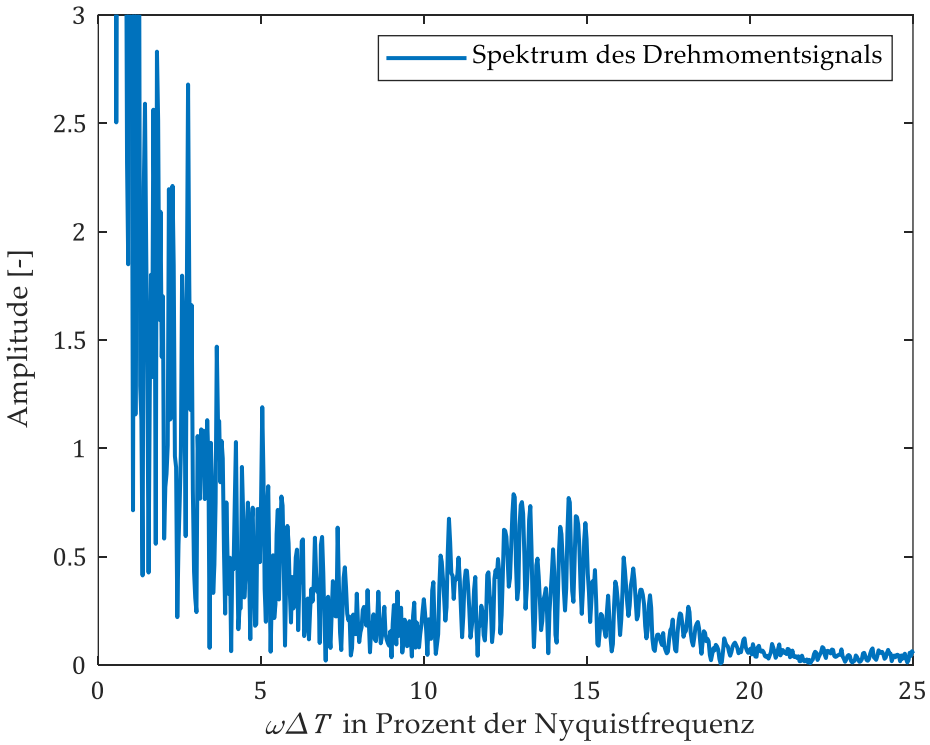


**Abb. 6.4** Spektrum des Drehmomentsignals über $\omega\Delta T$ in Prozent der Nyquistfrequenz mit $\Delta T = 0{,}01s$, Fokus auf kleine Amplituden

In Abschnitt 6.2.3 wird analysiert, ob diese Algorithmen in Kombination mit der Diskontinuitäten-Erkennung trotzdem auf das Drehmomentsignal anwendbar sind. Weiterhin wird untersucht, ob trotz der hochfrequenten Störungen im Signal ein optimal ausgelegter Kopplungsalgorithmus, der sich online an den Signalverlauf anpasst, den Kopplungsfehler verringern und das Systemverhalten des RVPs verbessern kann.

### 6.2.2 Vorangehende Untersuchungen mithilfe einer Co-Simulation

Eine Herausforderung bei der Anwendung der Kopplungsmethodik bei RVPen ist es festzustellen, ob eine Veränderung der Signalverläufe durch den Einsatz der Algorithmen auch tatsächlich eine Verbesserung darstellt. Es ist nicht ohne weiteres möglich zu zeigen, dass das Systemverhalten aufgrund der Methodik um genau jene Effekte korrigiert wird, welche durch die Latenzzeiten verursacht werden, da die Latenzzeiten technisch bedingt sind und der RVP nicht ohne sie betrieben werden kann. Eine Bestimmung des Kopplungsfehlers, beispielsweise mithilfe der Metrik von Sprague und Geers (2004) (s. Abschnitt 2.4), erlaubt zwar den Einfluss der Kopplungsmethodik auf einzelne Signale zu quantifizieren, hilft aber nur bedingt dabei den Einfluss der Methodik auf das Gesamtsystemverhalten des RVPs zu bewerten.

Um trotzdem eine Referenz für das Systemverhalten ohne Latenzzeiten des hier untersuchten RVPs eines Hybridfahrzeugs zu erhalten und festzustellen welchen Effekt Latenzzeiten auf das Systemverhalten haben, wird der RVP als rein virtuelle Co-Simulation aufgebaut. Dazu wird ein Simulationsmodell erstellt, welches das Verhalten des realen Verbrennungsmotors inklusive des Steuergeräts abbildet und mit demselben Fahrzeugsimulationsmodell gekoppelt ist, welches als virtueller Prototyp des RVPs verwendet wird. Die Koppelsignale zwischen den Modellen in der erstellten Co-Simulation entsprechen denen des in Abb. 6.3 dargestellten Aufbaus des RVPs.

Mit dem Co-Simulationsmodell, das den RVP darstellt, wird dieselbe Fahrsequenz zweimal simuliert. Bei der einen Simulation wird die Referenzlösung mit möglichst geringen Kopplungsfehlern simuliert, indem die Makroschrittweite des Signalaustauschs der Modelle auf $0{,}001s$ festgelegt wird. Dies entspricht der Mikroschrittweite der Modelle und ist daher der kleinste Wert, der ohne eine Änderung der Modelle möglich ist. Bei der anderen Simulation des Co-Simulationsmodells werden die realen technischen Randbedingungen des RVPs exakt abgebildet. Dazu wird die Makroschrittweite genau wie für den RVP in Abschnitt 6.2.1 auf $\Delta T = 0{,}01s$ gesetzt und das Drehmomentsignal künstlich um $k\Delta T = 0{,}06s$ verzögert. Die entwickelte Kopplungsmethodik wird nicht eingesetzt, um zu festzustellen, wie sich das Systemverhalten unter den gegebenen technischen Randbedingungen im Vergleich zur Simulation mit möglichst geringem Kopplungsfehler verändert.

Ein Vergleich der Ergebnisse der beiden Simulationen ist in Abb. 6.5 dargestellt. Es ist ein 12*s* Ausschnitt einer längeren Fahrsequenz gezeigt, in der das virtuelle Hybridfahrzeug unter Hinzunahme des Verbrennungsmotors zweimal kurz beschleunigt. Anhand dieses Vergleichs lassen sich folgende Aussagen darüber treffen, wie die Latenzzeiten das Verhalten des simulierten Systems beeinflussen:

- Das untersuchte System ist auch mit simulierten Latenzzeiten stabil. Keines der dargestellten und auch keines der anderen Koppelsignale weist darauf hin, dass das System mit den simulierten Verzögerungen instabil ist oder sich nahe an der Stabilitätsgrenze befindet.

- Die Latenzzeiten haben nur einen geringen Effekt auf die Fahrzeuggeschwindigkeit. Der Geschwindigkeitsverlauf in der Simulation mit Latenzzeiten ist im Vergleich zu der Simulation ohne Latenzzeiten lediglich um die Dauer der Verzögerung verschoben und weist abgesehen davon nur minimale Unterschiede verglichen mit der Simulation ohne Latenzzeiten auf. Wie bereits in Abschnitt 6.2.1 angenommen ist die Zeitkonstante der Geschwindigkeitsregelung des virtuellen Fahrers groß genug, dass die simulierten Latenzzeiten die Regelung nicht nennenswert beeinflussen.

- Der Vergleich des Drehmomentverlaufs offenbart, dass dieses Signal durch die Latenzzeiten beeinflusst wird. In der Simulation mit Latenzzeiten ist in der ersten Beschleunigungsphase das maximal benötigte Drehmoment an der Antriebswelle um ungefähr 6% größer und der Hochpunkt wird 0,3*s* später erreicht. Dies ist die größte Veränderung im Systemverhalten aufgrund der Latenzzeiten und damit zu erklären, dass an der Schnittstelle zwischen Verbrennungsmotor und Antriebsstrang aufgrund der Latenzzeiten Energie verloren geht, wodurch zum Erreichen derselben Geschwindigkeit ein größeres Maximalmoment benötigt wird. Die übertragene Energiemenge lässt aus dem Drehmomenten- und Drehzahlverlauf für beide Simulationen berechnen und ist mit simulierten Latenzzeiten um 2,5% höher als ohne.

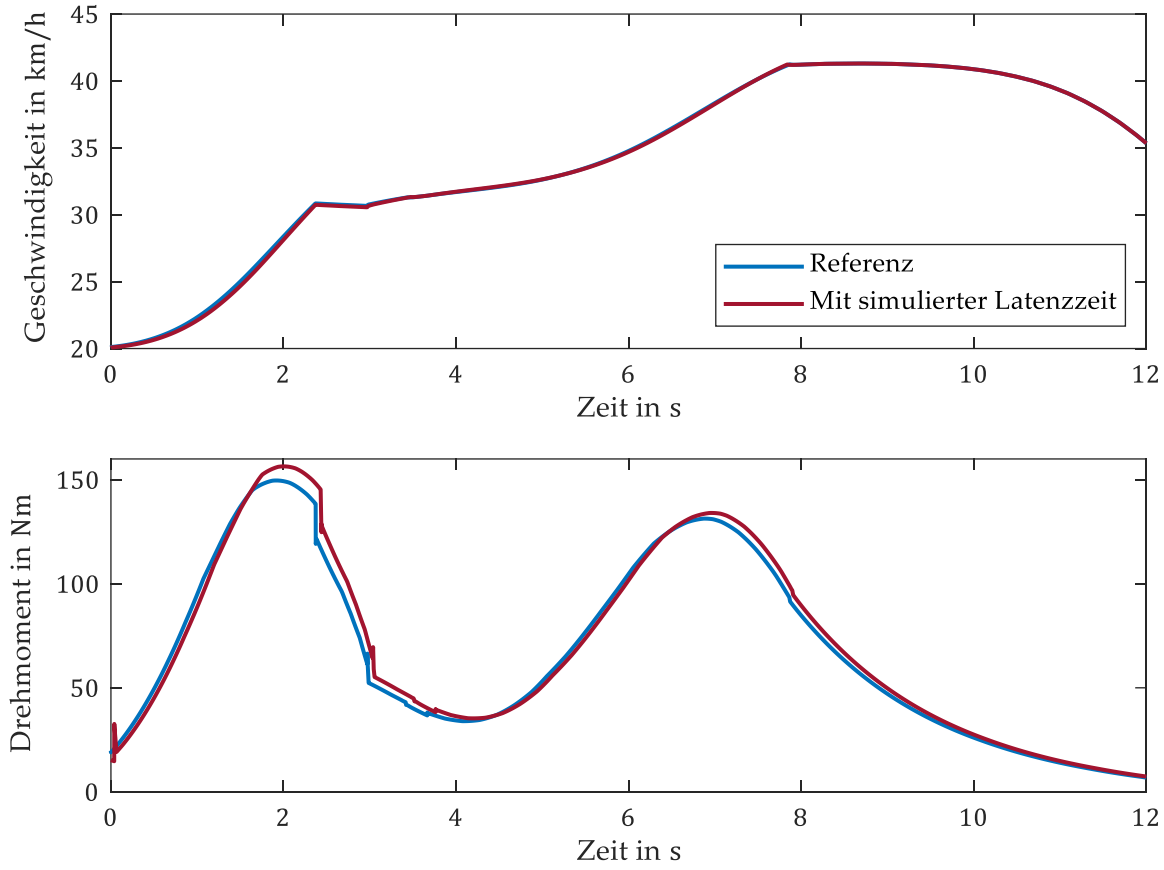


**Abb. 6.5** Vergleich der Ergebnisse der Co-Simulation mit und ohne simulierte Latenzzeiten. Oben: Fahrzeuggeschwindigkeit. Unten: Drehmoment an der Antriebswelle.

Übertragen auf den untersuchten RVP auf den in Abschnitt 6.2.3 die Kopplungsmethodik angewandt wird, bedeuten diese Simulationsergebnisse, dass die folgenden Kriterien ein Maß für eine erfolgreiche Kompensation der Latenzzeiten sind:

- Eine Verringerung des Kopplungsfehlers im Drehmomentsignal.
- Eine Verringerung ($-6\%$) und ein früheres Erreichen ($-0{,}3s$) des Maximalmoments in Beschleunigungsphasen.
- Eine Verringerung der zwischen Verbrennungsmotor und simulierten Antriebsstrang übertragenen Energiemenge ($-2{,}5\%$).

Aus den Simulationsergebnissen lässt sich ableiten, dass die Latenzzeiten, die bei der Kopplung der Prototypen des RVPs auftreten, einen Einfluss auf die übertragene Energiemenge des Verbrennungsmotors haben. Die Höhe des Einflusses liegt dabei in derselben Größenordnung wie die gesamten Optimierungsmöglichkeiten in Bezug auf den Energieverbrauch, die für die

untersuchte Hybridtopologie realistisch sind (s. Abschnitt 6.2). Um aussagekräftige Ergebnisse für das Einsparpotenzial verschiedener Hybridstrategien am untersuchten RVP erzielen zu können, ist daher eine Kompensation der Latenzzeiten notwendig.

Weiterhin zeigen die Simulationsergebnisse, dass das Drehmomentsignal Diskontinuitäten im Verlauf aufweist (s. $t \approx 8s$ in Abb. 6.5) und daher am RVP die generischen Kopplungsalgorithmen nur mit vorangehender Erkennung von Diskontinuitäten verwendet werden können. Außerdem sind die in Abb. 6.4 identifizierten höheren Frequenzen im Drehmomentsignal des RVPs tatsächlich ungewollte Störfrequenzen, da diese in der hier durchgeführten Simulation nicht auftreten.

### 6.2.3 Ergebnisse der Kopplungsmethodik am Real-Virtuellen Prototyp

In diesem Kapitel wird die entwickelte Kopplungsmethodik auf den in Abschnitt 6.2.3 vorgestellten und analysierten RVP angewandt. Die Kompensation der Latenzzeit von $k\Delta T = 0{,}06s$ erfolgt mithilfe der in Abschnitt 6.1 gezeigten Implementierung der Methodik als FMU, welche am Eingang des Fahrzeugsimulationsmodells auf das am Motorprüfstand gemessene Drehmomentsignal eingesetzt wird. Zunächst wird ein FFNN-Kopplungsalgorithmus mit dem Optimierungsverfahren aus Abschnitt 4.2 optimal ausgelegt. Anschließend werden die Ergebnisse der Latenzzeitkompensation für verschiedene Konfigurationen der Kopplungsmethodik diskutiert.

**Auslegung und online Lernen des FFNN-Kopplungsalgorithmus**

Für die optimale Auslegung eines Kopplungsalgorithmus nach Abschnitt 4.2 werden als einzige Systemparameter die Bandbreite des Signals, auf welches der Kopplungsalgorithmus angewandt wird, sowie der relative Grad des gekoppelten Systems benötigt. Die Methode zur Auslegung ist so definiert, dass zur Bestimmung beider Parameter keine detaillierte Übertragungsfunktion des RVPs notwendig ist und diese Parameter stattdessen aus Messungen des Koppelsignals abschätzbar sind.

Aufgrund der Störfrequenzen im Signal werden in der Kostenfunktion des Optimierungsproblems verschiedene Grenzfrequenzen für die Summanden der Amplitude und der Phase verwendet. Für die Kompensation der Latenzzeiten ist eine neutrale Phasenverschiebung nur innerhalb der Bandbreite des Nutzsignals wichtig, während eine neutrale Amplitudenverstärkung zusätzlich auch für die Störfrequenzen gelten muss, um deren Verstärkung zu vermeiden. Basierend auf dem in Abb. 6.4 dargestellten Spektrum des Drehmomentsignals, ergeben sich somit die Grenzfrequenzen

$$\omega_{b,p} \in \left[\omega_{u,p}, \omega_{g,p}\right] = \left[0{,}1\frac{rad}{s}, 10\frac{rad}{s}\right] \tag{6-1}$$

zur Bestrafung des Phasenfehlers. Dabei berechnet sich die Kreisfrequenz $\omega$ aus dem in Abb. 6.4 verwendeten Anteil an der Nyquistfrequenz in Prozent für die gegebene Makroschrittweite von $\Delta T = 0{,}01s$ durch Multiplikation mit $\pi$. Für die Grenzfrequenzen zur Bestrafung einer nicht neutralen Amplitude gilt

$$\omega_{b,a} \in \left[\omega_{u,a}, \omega_{g,a}\right] = \left[0{,}1\frac{rad}{s}, 55\frac{rad}{s}\right], \tag{6-2}$$

damit sowohl der Frequenzbereich des Nutzsignals als auch der, in dem die Störfrequenzen liegen, eingeschlossen ist. Der relative Grad des Übertragungsverhaltens der offenen Kette aus Motorprüfstand und Fahrzeugsimulationsmodell wird konservativ zu $r = 2$ geschätzt, da keines der beiden Systeme einen direkten Durchgriff zwischen den Ein- und Ausgangssignalen Drehmoment und Drehzahl aufweist. Ein weiterer Parameter bei der optimalen Auslegung ist die Anzahl der vergangenen Signalwerte, die der Kopplungsalgorithmus verwenden soll. Dem Vorgehen aus Abschnitt 4.2.1 folgend ergibt sich diese zu $m = 9$, da sich das Optimierungsergebnis mit größeren Werten nicht mehr signifikant verbessert. Mithilfe der durch die Lösung des Optimierungsproblems berechneten Gewichtungsparameter wird über die Methode des Network Morphism aus Abschnitt 3.5.1 ein linearer FFNN-Kopplungsalgorithmus generiert, der das optimierte Verhalten ausweist und über das online Lernen während der Ausführung des RVPs angepasst werden kann.

Das Übertragungsverhalten des optimierten FFNN-Kopplungsalgorithmus ist in Abb. 6.6 für die gegebene Anzahl der Makroschritte Verzögerung von $k = 6$ gezeigt und mit der Übertragungsfunktion vom ZOH und EROS3 verglichen. Wie in der Kostenfunktion vorgegeben ist die Amplitudenverstärkung des optimierten FFNN-Algorithmus im Bereich der Störfrequenzen wesentlich geringer als beim EROS3. Die durch die Anzahl der verwendeten vergangenen Signalwerte gegebene Flexibilität des Kopplungsalgorithmus reicht aus, um einen Tiefpunkt des Amplitudenverlaufs in der Nähe der maximalen Verstärkung der Störfrequenzen zu ermöglichen. Im Bereich des Nutzsignals dagegen ist die Amplitudenverstärkung beim optimierten FFNN-Algorithmus höher und die Kompensation des durch die Latenzzeit hervorgerufenen Phasenfehlers schlechter als beim EROS3.

Die andere in Abb. 6.6 dargestellte Übertragungsfunktion ist die desselben FFNN-Kopplungsalgorithmus nach dessen Anpassung durch online Lernen während der Ausführung des RVPs. Dazu wurde mit der Methodik aus Abschnitt 3.5.2 während der Ausführung des RVPs der im nächsten Abschnitt gezeigten Fahrsequenz zwei Lernzyklen mit jeweils 100 Epochen durchgeführt. Der angepasste Kopplungsalgorithmus weist verglichen mit dem optimierten Kopplungsalgorithmus eine noch geringe Amplitudenverstärkung im gesamten Bereich des Nutzsignals als auch der Störfrequenzen auf, wodurch der Phasenfehler etwas schlechter aber immer noch deutlich besser als beim Standardalgorithmus ZOH kompensiert wird.

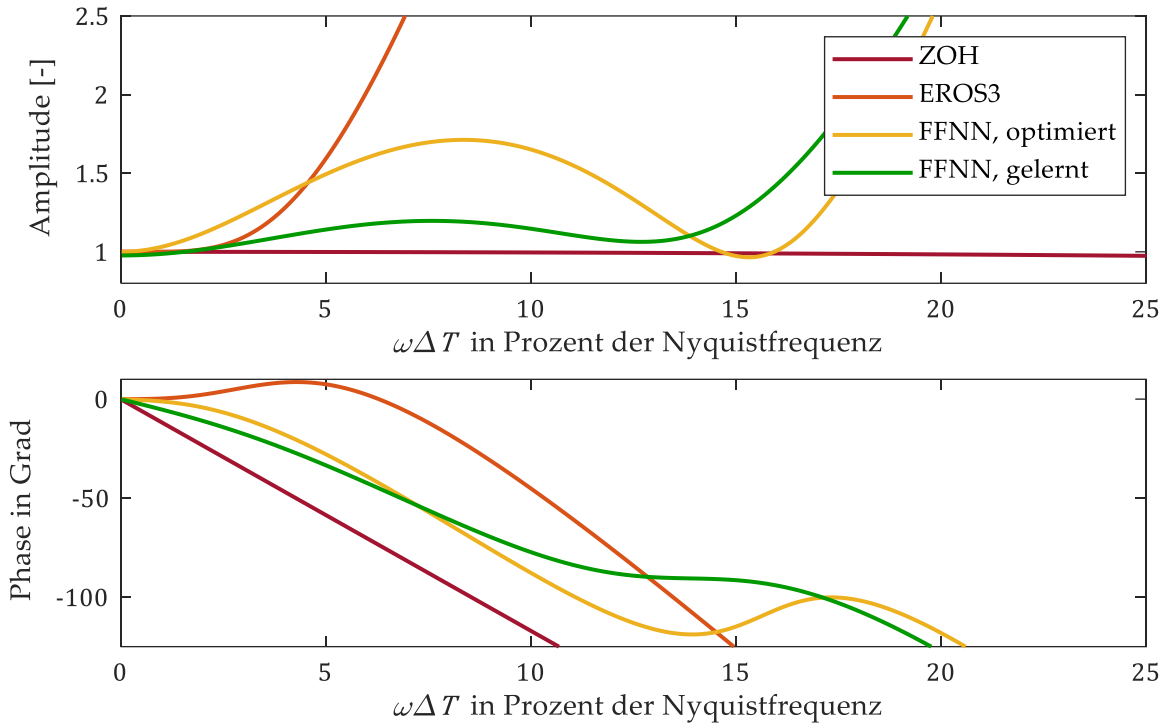


**Abb. 6.6** Bode-Diagramm des optimierten und des gelernten FFNN im Vergleich mit ZOH und EROS3 für $k = 6$ bzgl. des Verhältnisses aus Frequenz und Makroschrittweite

Basierend auf der Analyse ihrer Übertragungsfunktionen bilden sowohl der optimierte als auch der durch online Lernen angepasste FFNN-Kopplungsalgorithmus einen Kompromiss aus Kompensation der Latenzzeiten des Nutzsignals und möglichst geringer Verstärkung der Störfrequenzen.

### Ergebnisse für verschiedene Kopplungsalgorithmen

Der Vergleich verschiedener Konfigurationen der entwickelten Methodik erfolgt anhand derselben $12s$ dauernden Fahrsequenz, wie sie auch in der Simulation (s. Abschnitt 6.2.2) verwendet wurde. Das Hybridfahrzeug führt dabei eine Beschleunigung mithilfe des Verbrennungsmotors auf dem Prüfstand durch, welche von einem Schaltvorgang bei $t = 3.2s$ unterbrochen ist. Abb. 6.7 zeigt das Drehmomentsignal für verschiedene Kopplungsalgorithmen jeweils am Ausgang der FMU, die die Kopplungsmethodik implementiert, wodurch die Einflüsse der unterschiedlichen Algorithmen auf das Signal zu sehen sind. Alle Methoden werden mit dem Standardalgorithmus ZOH verglichen, bei dem das Koppelsignal nicht verändert wird. Die verglichenen Methoden sind zum einen die generischen Algorithmen FOH, EROS3 und EROS4 jeweils inklusive der in dieser Arbeit entwickelten Erkennung von Diskontinuitäten aus Abschnitt 5.3.1 und des dazugehörenden Umschaltens des Algorithmus auf ZOH, sobald die Erkennung anschlägt (s. Abschnitt 5.3.2). Die Hinzunahme der Erkennung von Diskontinuitäten ist notwendig, da sich die Bandbreite des Drehmomentsignals deutlich außerhalb des

in Tab. 4.1 in Abschnitt 4.1.2 definierten Gültigkeitsbereichs der generischen Kopplungsalgorithmen befindet. Obwohl das Drehmomentsignal vom Steuerungssystem des Prüfstands gefiltert wird, wirken die Störfrequenzen im abgetasteten Signal wie Diskontinuitäten. Diese sollen erkannt werden, um eine starke Überhöhung ihrer Amplitude durch den Kopplungsalgorithmus zu verhindern. Die anderen beiden Kopplungsalgorithmen, die in Abb. 6.7 verglichen werden, sind das optimierte und das durch online Lernen angepasste FFNN, deren Übertragungsfunktion in Abb. 6.6 gezeigt ist. Für diese wird die Erkennung von Diskontinuitäten nicht hinzugenommen, da sie wie im Bode-Diagramm zu sehen ist, aufgrund der Optimierung und des Lernens die hohen Störfrequenzen nicht signifikant verstärken sollten.

Der SMB-Kopplungsalgorithmus (s. Abschnitt 3.3.2) ist nicht mit in den Vergleich aufgenommen, da er ebenfalls die Störfrequenzen verstärkt aber im Unterschied zu FOH, EROS3 und EROS4 nicht zusammen mit der Erkennung von Diskontinuitäten verwendet werden kann (s. Abschnitt 5.3.2). Die online Bewertung der Extrapolationsqualität aus Abschnitt 5.2 erfolgt während der Ausführung des RVPs, wird aber nicht wie in Abschnitt 5.2 vorgeschlagen, verwendet, um die Ausführung des RVPs vorzeitig abzubrechen. Dies ist bei diesem RVP nicht notwendig, da Maßnahmen wie das Beschränken der Eingangssignale der realen Prototypen getroffen werden, um eine Beschädigung der Prototypen durch z.B. aufgrund großer Kopplungsfehler stark schwingender Eingangssignale zu verhindern.

Rein visuell zeigen sich die größten Unterschiede im Vergleich zum Standardalgorithmus ZOH bei Verwendung der FFNNs in der ersten Beschleunigungsphase im Bereich um $t = 2s$. Dort entspricht der Verlauf des Drehmoments eher dem der Simulation (vgl. Abb. 6.5), ist jedoch von einer deutlich sichtbaren hochfrequenten Schwingung überlagert, die beim optimierten FFNN eine größere Amplitude als beim online angepassten FFNN aufweist. Beim FOH, EROS3 und EROS4 jeweils mit Umschalten nach einer Diskontinuität ähnelt der gesamte Drehmomentverlauf eher desjenigen vom ZOH. Vor allem beim EROS4 sind im Bereich um $t = 3{,}5s$ deutliche Schwingungen zu sehen und der Drehmomentverlauf weicht anschließend für ungefähr eine Sekunde stark von dem Verlauf bei der Verwendung vom ZOH ab.

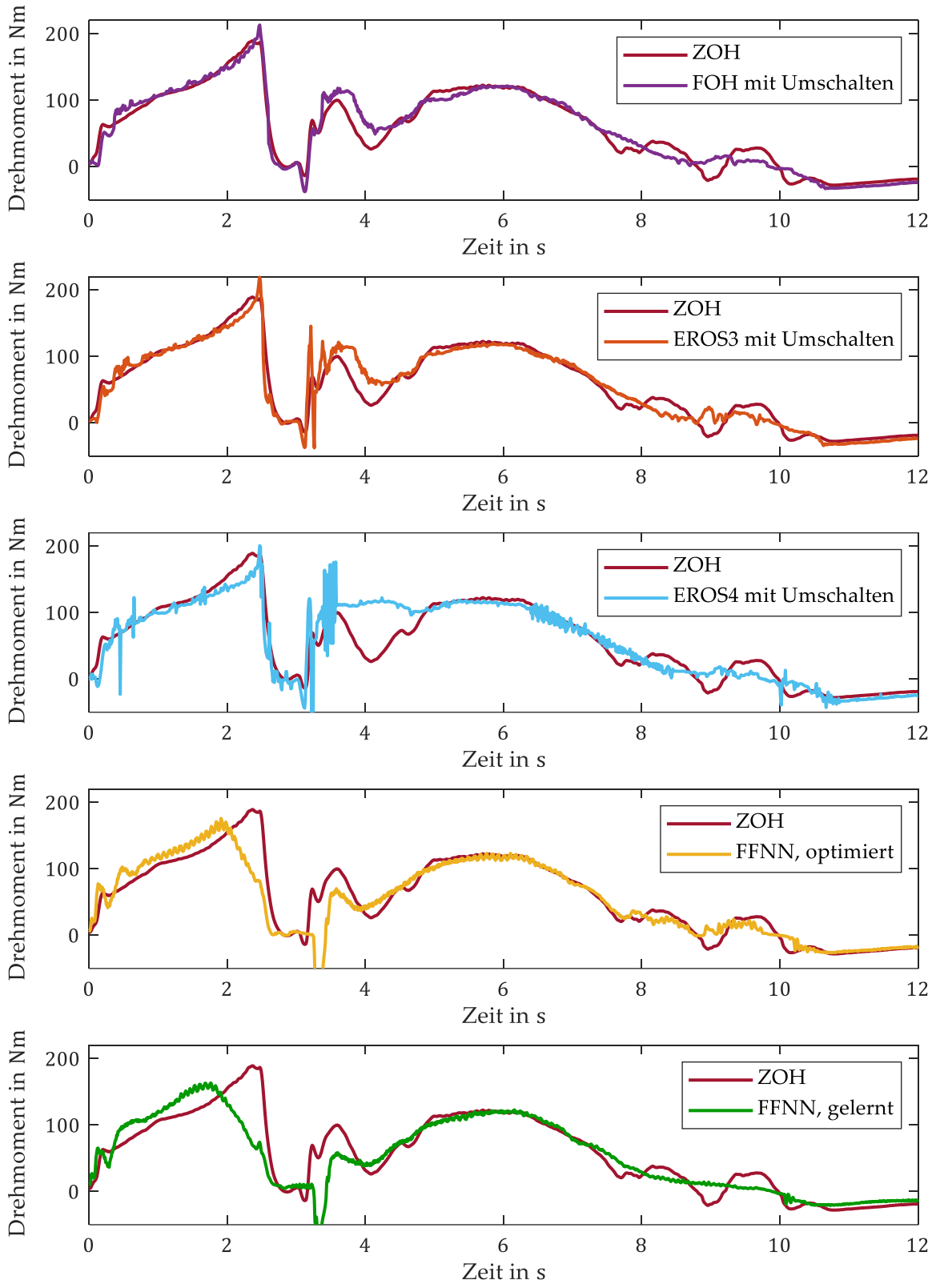


**Abb. 6.7** Vergleich der Anwendung vom ZOH mit verschiedenen anderen Kopplungsalgorithmen für das latenzzeitkompensierte Drehmomentsignal des RVPs

Zur Bewertung dieser Unterschiede im Signalverlauf ist in Abb. 6.8 und Tab. 6.1 der über die Metrik von Sprague und Geers (s. Abschnitt 2.4) berechnete summierte Kopplungsfehler für alle verglichenen Kopplungsalgorithmen dargestellt. Es zeigt sich, dass der kombinierte Kopplungsfehler bei Verwendung des optimierten bzw. des online angepassten FFNNs um 35% bzw. um 40% im Vergleich zum Standardalgorithmus ZOH verringert ist. Auch beim FOH und EROS3 ist eine leichte Verbesserung verglichen mit dem ZOH zu beobachten, während der Kopplungsfehler beim EROS4 um 16,6% höher ist.

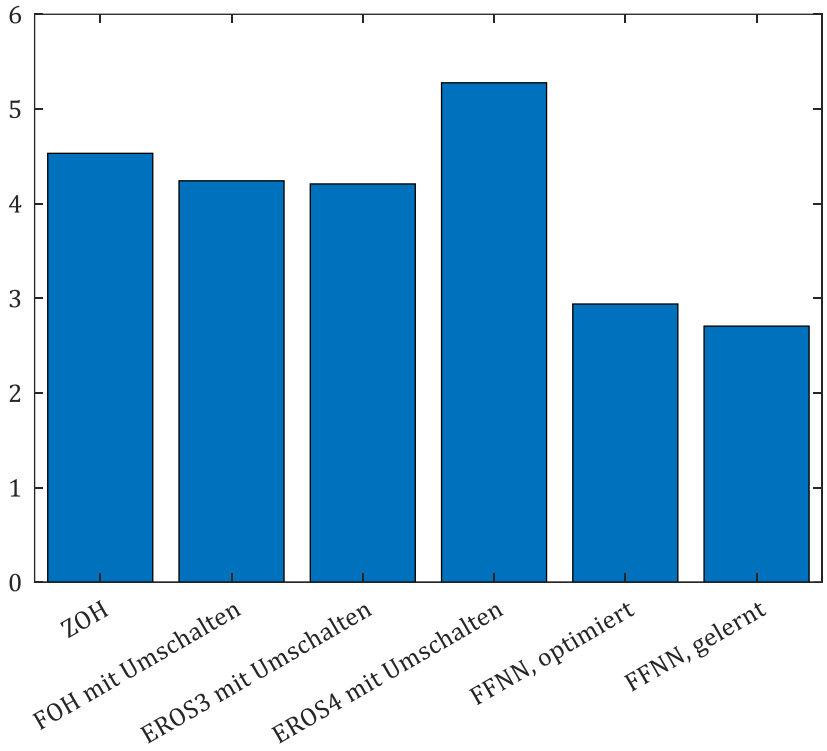


**Abb. 6.8** Balkendiagramm des summierten kombinierten Kopplungsfehlers nach Sprague und Geers ($\cdot 10^2$) für das Drehmomentsignal des RVPs

**Tab. 6.1** Summierter Kopplungsfehler mit Sprague und Geers ($\cdot 10^2$) als Distanzmaß für verschiedene Kopplungsalgorithmen

| | $C_{S\&G}$ | $M_{S\&G}$ | $P_{S\&G}$ |
|---|---|---|---|
| ZOH | 4,53 | 0,01 | 4,53 |
| FOH | 4,24 | 0,79 | 4,17 |
| EROS3 | 4,21 | 0,88 | 4,11 |
| EROS4 | 5,28 | 0,59 | 5,24 |
| FFNN, optimiert | 2,94 | 0,65 | 2,87 |
| FFNN, gelernt | 2,71 | 0,67 | 2,62 |

Zusätzlich ist basierend auf den Voruntersuchungen in der Co-Simulation aus Abschnitt 6.2.2 bewertet, ob die Veränderungen im Drehmomentsignal durch den Einsatz der Kopplungsmethodik tatsächlich für eine Kompensation der Latenzzeiten spricht. Dazu sind in Tab. 6.2 für jeden Kopplungsalgorithmus diejenigen Kriterien gelistet, die in Abschnitt 6.2.2 zur Bewertung des Drehmomentsignals identifiziert wurden. Während sich die Werte für FOH, EROS3 und EROS4 nur geringfügig bzgl. ZOH ändern, sind die Veränderungen für die FFNNs größer und liegen in dem Bereich, der in der vorangegangenen Simulation für eine ideale Kompensation der Latenzzeiten ermittelt wurde. Dort führte eine ideale Kompensation der Latenzzeiten im Vergleich zur nicht kompensierten Latenzzeit (ZOH) zu einem um $-6\%$ veränderten Maximalwert des Moments am ersten Hochpunkt und ein um $-0{,}3s$ verschobenes Auftreten desselben. Die Energie, die über die Drehmoment-Drehzahl Schnittstelle übertragen wird, unterscheidet sich dabei um $-2{,}5\%$. Es sei angemerkt, dass vor allem der Wert des maximalen Moments zwischen mehreren Ausführungen des RVPs mit identischer Konfiguration um zwei bis vier Prozentpunkte schwanken kann, je nachdem ob der Maximalwert des Nutzsignals mit einem Hoch oder Tiefpunkt der überlagerten Schwingung zusammenfällt.

**Tab. 6.2** Systemverhalten repräsentierende Werte für das unveränderte Drehmomentsignal vom Steuerungssystem des Motorprüfstands, gemessen am Eingang der Kopplungsalgorithmen

| | Wert max. Moment bzgl. ZOH in % | Zeitpunkt max. Moment bzgl. ZOH in s | Übertragene Energiemenge bzgl. ZOH in % |
|---|---|---|---|
| FOH | 2,66 | 0,12 | −0,09 |
| EROS3 | 2,67 | 0,12 | −1,07 |
| EROS4 | −6,68 | 0,13 | −0,54 |
| FFNN, optimiert | −12,93 | −0,42 | −4,39 |
| FFNN, gelernt | −8,90 | −0,38 | −2,92 |

**Diskontinuitäten-Erkennung**

Wie oben bereits diskutiert ist die Hinzunahme der Erkennung von Diskontinuitäten für die generischen FOH, EROS3 und EROS4 Algorithmen notwendig, um sie unter den gegebenen Bedingungen mit den Störfrequenzen verwenden zu können. Da das Signal während der gesamten Zeit von den Störfrequenzen überlagert ist, schlägt die Erkennung von Diskontinuitäten sehr häufig an, weshalb ein Großteil der Fahrsequenz ZOH statt des konfigurierten Kopplungsalgorithmus angewandt wird. Dies ist der Grund, weshalb die Ergebnisse für FOH und EROS3 denen vom ZOH so ähnlich sind.

Zur Analyse der Diskontinuitäten-Erkennung zeigt Abb. 6.9 vier Beispiele aus dem Drehmomentsignal bei der Verwendung vom EROS3 als Kopplungsalgorithmus inklusive Erkennung von Diskontinuitäten und Umschalten zu ZOH. Die Bilder zeigen jeweils den Eingang der

FMU bzw. des Kopplungsalgorithmus und den Ausgang desselben. Zusätzlich ist als Referenz das um $k = 6$ Makroschritte verschobene Eingangssignal dargestellt, welches einer perfekten Kompensation der Latenzzeit entspricht.

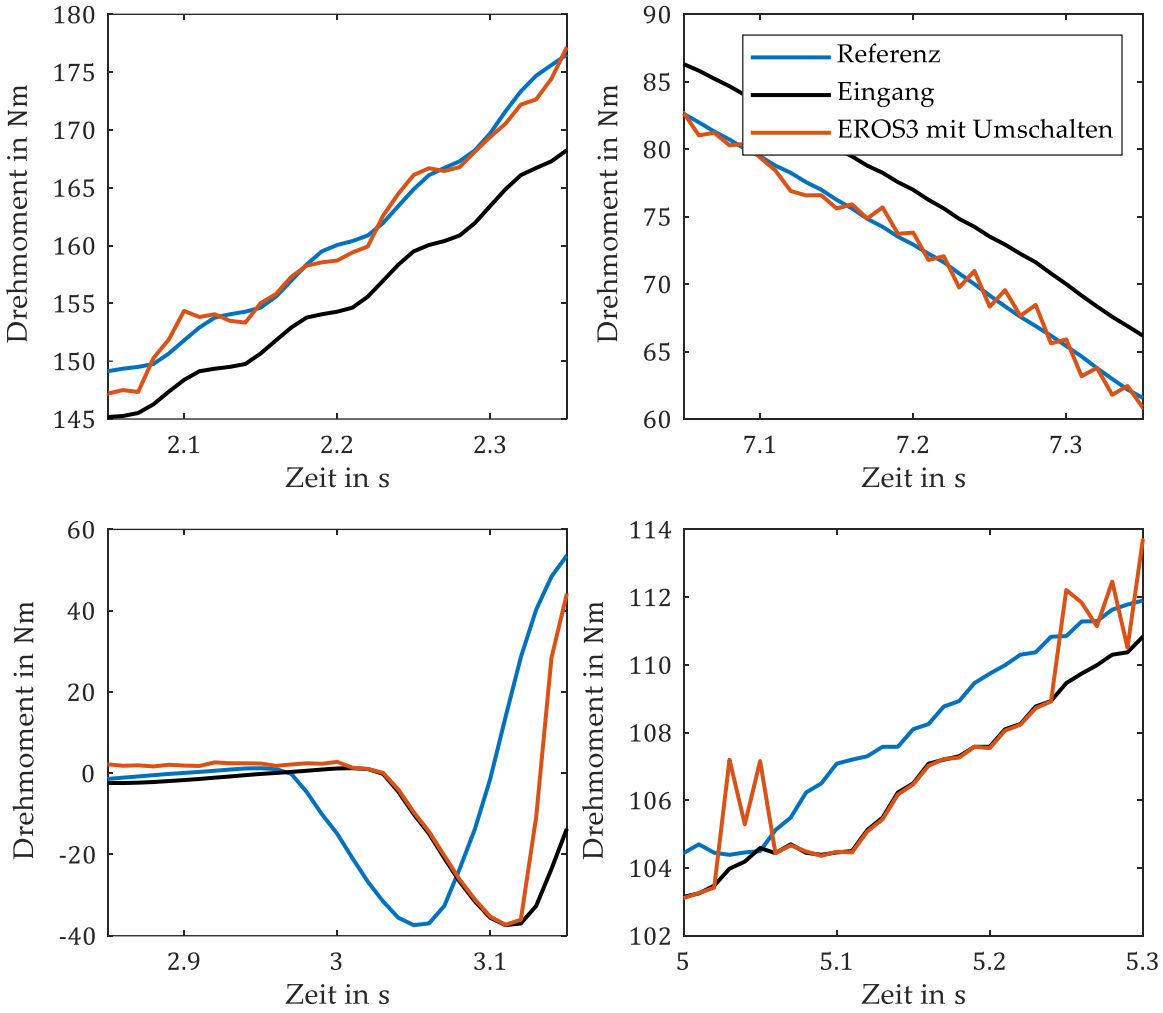


**Abb. 6.9** Beispiele für die Erkennung von Diskontinuitäten anhand EROS3 für das Drehmomentsignal des RVPs. Oben: Keine Diskontinuitäten erkannt. Unten: Diskontinuitäten erkannt und Kopplungsalgorithmus umgeschaltet.

In den oberen beiden Abbildungen sind Beispiele gezeigt, in denen die Diskontinuitäten-Erkennung nicht anschlägt. In diesen Fällen ist dies korrekt, da hier die Änderungsrate im Signalverlauf so groß ist, dass die Amplitude der überlagerten Störfrequenzen die Extrapolation vom EROS3 nur wenig beeinflusst, wodurch das Extrapolationsergebnis sehr gut mit der Referenz übereinstimmt und der Kopplungsfehler gering ist. Im unteren linken Beispiel schlägt die Diskontinuitäten-Erkennung bei ungefähr $t = 3s$ an und es wird anschließend ZOH verwendet, wodurch der Ausgang des Algorithmus dessen Eingang entspricht. Dieses Verhalten ist korrekt, da ohne ein Umschalten des Kopplungsalgorithmus die plötzliche starke Änderung im Signalverlauf zu einer extremen Unterschätzung des Tiefpunkts bei $t = 3{,}05s$ führen

würde. Das untere rechte Beispiel zeigt einen Verlauf, bei dem die Amplitude der Störfrequenzen bezüglich der Änderungsrate des Nutzsignals gerade so groß ist, dass die Diskontinuitäten-Erkennung mal anschlägt, und mal nicht. Es ist zu sehen, dass die Störfrequenzen vom EROS3 etwas verstärkt werden, wenn die Diskontinuitäten-Erkennung nicht aktiv ist. Entsprechenden Stellen sind beim EROS4 die Ursache für die deutlichen Schwingungen, wie sie in Abb. 6.7 gezeigt sind, da EROS4 den Frequenzbereich, in dem die Störungen liegen doppelt so stark verstärkt wie EROS3. Das untere rechte Beispiel offenbart somit, dass die Diskontinuitäten-Erkennung für das gegebene Signal nicht optimal funktioniert, da hier der Signalverlauf durchgängig als nicht für den konfigurierten Kopplungsalgorithmus geeignet eingestuft werden sollte. Ursache dafür ist, dass die Störfrequenzen über einen längeren Zeitraum das Nutzsignal überlagern und sich daher nicht so plötzlich auf das Frequenzspektrum des Signals auswirken und somit nicht immer zu einer erkennbaren Diskontinuität führen, wie es bei einem vereinzelten „Sprung“ im Signal der Fall wäre, für dessen Erkennung die Methode in erster Linie entwickelt ist.

### 6.2.4 Diskussion der Ergebnisse

Die technischen Randbedingungen am untersuchten RVP eines Hybridfahrzeugs sind eine große Herausforderung für die Kompensation von Latenzzeiten. Zum einen ist die Verzögerung mit $0{,}06s$ für eine physikalische Schnittstelle eines hochdynamischen Systems sehr hoch und zum anderen ist das Nutzsignal mit Störfrequenzen überlagert. Trotzdem zeigen die Ergebnisse aus Abschnitt 6.2.3, dass die in dieser Arbeit entwickelte Kopplungsmethodik am realen System funktioniert und das Systemverhalten des RVPs verbessert. Es ist gezeigt, dass bei Verwendung der Kopplungsmethodik der Kopplungsfehler um bis zu 40% reduziert wird. Basierend auf den vorangegangenen Untersuchungen an einer MIL-Simulationsumgebung lässt sich außerdem ableiten, dass die Verletzungen der Energieerhaltung an der kritischen Kopplungsschnittstelle zwischen dem Verbrennungsmotor und der Fahrzeugsimulation des RVP durch den Einsatz der Kopplungsmethodik ebenfalls deutlich verringert werden. Dadurch wird die Glaubwürdigkeit der Aussagen zu verschiedenen Hybridstrategien, die mit diesem RVP erzielt werden, erhöht. Es ist zu erwarten, dass die Ergebnisse besser auf reale Hybridfahrzeuge übertragbar sind und somit belastbarer und wertvoller werden. Dies verkürzt und verbilligt den Entwicklungsprozess für optimierte Hybridstrategien. Die Hauptergebnisse bezogen auf die einzelnen Teile der entwickelten Kopplungsmethodik, welche sich aus den Ergebnissen am RVP ergeben, sind im Folgenden zusammengefasst.

**Erfolgreiche optimale Auslegung des Kopplungsalgorithmus**

Die Anwendbarkeit der Methode zur optimalen Auslegung eines Kopplungsalgorithmus aus Abschnitt 4.2 bestätigt sich am realen System. Das wenige Systemwissen, welches zur Auslegung benötigt wird, ist mit geringem Aufwand aus Messungen der Koppelsignale abzuleiten

und der resultierende Kopplungsalgorithmus verringert den Kopplungsfehler unter den herausfordernden technischen Randbedingungen deutlich besser als alle untersuchten generischen Algorithmen. Außerdem wird bei Verwendung des optimierten und trainierten Kopplungsalgorithmus das Systemverhalten des durch den RVP repräsentierten Hybridfahrzeugs in einer Weise beeinflusst, dass es eher dem des realen Fahrzeugs entspricht.

**Online Lernen des FFNN-Kopplungsalgorithmus verbessert Latenzzeitkompensation**

Die in Abschnitt 3.5 entwickelte Methodik der Übertragung eines generischen als ARIMA($p$,$d$,0) klassifizierten Kopplungsalgorithmus in ein FFNN zusammen mit dessen in Abschnitt 6.1 implementierten Anpassung zur Laufzeit, ist ebenfalls erfolgreich am realen System einsetzbar. Da das online trainierte FFNN wesentlich mehr Parameter zur Verfügung hat als zur optimalen Auslegung verwendet werden, passt es sich noch besser auf das Koppelsignal an, wodurch sowohl der Kopplungsfehler weiter verringert als auch das Gesamtsystemverhalten des RVPs nochmal verbessert wird.

**Diskontinuitäten-Erkennung erleichtert Verwendung generischer Kopplungsalgorithmen**

Die Erkennung von Diskontinuitäten inklusive des Umschaltens des Kopplungsalgorithmus wie sie in Abschnitt 5.3 eingeführt ist, erfüllt ihren Zweck am realen System. Sie ermöglicht die Verwendung generischer Algorithmen wie FOH und EROS3 auf ein Signal, dessen Bandbreite für die gegebene Verzögerung und Makroschrittweite deutlich außerhalb des Gültigkeitsbereichs dieser Algorithmen liegt. Dies funktioniert über weite Strecken sehr gut, obwohl die Diskontinuitäten-Erkennung in erster Linie zur Anwendung auf Signale mit vereinzeln auftretenden Diskontinuitäten und nicht für ständig auftretende Diskontinuitäten im abgetasteten Signal aufgrund einer hochfrequenten überlagerten Schwingung entwickelt ist. Trotz dieser Unzulänglichkeit verringert sich der Kopplungsfehler bei der Verwendung der Diskontinuitäten-Erkennung für FOH und EROS3 im Vergleich zum Standardalgorithmus ZOH, während er für diese Kopplungsalgorithmen ohne eine vorangehende Erkennung von Diskontinuitäten größer ist.

**Methode zur Bewertung der Extrapolationsgüte an diesem RVP nicht sinnvoll anwendbar**

Die in Abschnitt 5.2 entwickelte Methode zur online Bewertung der Extrapolationsgüte bewertet das Ergebnis von jedem der untersuchten Kopplungsalgorithmen bei ihrem Einsatz auf diesen RVP als nicht ausreichend. Dies ist zwar korrekt, da die Bandbreite des Drehmomentsignals für die gesamte untersuchte Fahrsequenz außerhalb des Gültigkeitsbereichs der Kopplungsalgorithmen liegt. Allerdings wird dadurch kein Unterscheidungsmerkmal geschaffen, um zwischen kleineren Verletzungen der Fehlertoleranz, mit denen trotzdem eine Verbesserung des Systemverhaltens durch Latenzzeitkompensation erreicht werden kann, und großen Verletzungen, wie sie zum Beispiel beim EROS4 bei $t = 3{,}5s$ (s. Abb. 6.7) vorkommen und das

Systemverhalten negativ beeinflussen, zu differenzieren, wodurch die Methode keinen Mehrwert liefert. Um für diesen RVP einen Nutzen aus der online Bewertung der Extrapolationsgüte zu ziehen, müsste daher der Gültigkeitsbereich der Algorithmen spezifisch für das Drehmomentsignal angepasst werden. Allerdings würde dies der Anforderung an die Methode widersprechen, nach der sie allgemein und ohne Anpassungen anwendbar sein sollte, weshalb der Ansatz signalspezifischer Gültigkeitsbereiche in dieser Arbeit nicht weiterverfolgt wird.

**EROS3 ist der beste der untersuchten generischen Kopplungsalgorithmen**

Mithilfe des entwickelten EROS-Verfahren aus Abschnitt 3.2 lassen sich mehrere Kopplungsalgorithmen ableiten, von denen EROS3 und EROS4 in dieser Arbeit näher untersucht sind. An mehreren Stellen dieser Arbeit zeigt sich, dass der verbleibende Kopplungsfehler beim Einsatz vom EROS4 zwar etwas geringer als beim EROS3 ist, wenn sie auf Signale innerhalb ihres Gültigkeitsbereichs angewandt werden, diese Verbesserungen aber minimal sind (vgl. Tab. 3.2 in Abschnitt 3.4). Dagegen zeigt sich bei der Analyse im Frequenzbereich in Abschnitt 4.1, dass EROS3 im Vergleich zu EROS4 eine bessere Balance zwischen Amplituden- und Phasenfehler sowie ein größeres Stabilitätsgebiet für das dort untersuchte System eines Zweimassenschwingers aufweist. Weiterhin führt die sehr hohe Amplitudenverstärkung vom EROS4 bei der Anwendung am produktiv genutzten RVP zu einer Verstärkung der Störfrequenzen, selbst wenn deren Amplitude so gering ist, dass die Diskontinuitäten-Erkennung nicht anschlägt. Außerdem benötigt EROS4 deutlich mehr vergangene Signalwerte als EROS3, weshalb er nach einer erkannten Diskontinuität erst nach einem längeren Zeitraum wieder verwendbar ist. Zusammengefasst gleichen die Nachteile vom EROS4 die minimal bessere Kompensation der Latenzzeiten unter idealen Bedingungen verglichen mit EROS3 nicht aus, weshalb EROS3 der beste der untersuchten EROS-Algorithmen ist und daher bevorzugt verwendet werden sollte.

# 7 Zusammenfassung und Ausblick

Virtuelle Prototypen werden bereits früh im Entwicklungsprozess mechatronischer Systeme in Form detaillierter Simulationsmodelle entwickelt und sollten, um Zeit und Kosten zu sparen, in allen Phasen des Entwicklungsprozesses weiterverwendet werden, ohne Änderung an ihnen vornehmen zu müssen. Für die effiziente Entwicklung vernetzter, domänenübergreifender Produkte bedeutet dies, dass sie mit anderen Prototypen gekoppelt werden müssen, um das Verhalten interagierender Teilsysteme abzubilden. In späteren Phasen des Entwicklungsprozesses, wenn erste Komponenten des Gesamtsystems bereits als reale Hardwarekomponente verfügbar sind, entstehen daher Real-Virtuelle Prototypen (RVP). Diese Prototypenkopplungen finden über eine Netzwerkverbindung statt, da die einzelnen Prototypen typischerweise auf unterschiedlicher Hardware ausgeführt werden, die räumlich voneinander getrennt sein kann. Der Datenaustausch innerhalb eines RVPs unterliegt zwangsläufig technisch bedingten Netzwerkeffekten, die das Systemverhalten von RVPen verfälschen oder sogar deren stabilen Betrieb verhindern können, wodurch deren Anwendungsmöglichkeiten eingeschränkt werden.

In der vorliegenden Arbeit wurde daher eine Kopplungsmethodik entwickelt, um den negativen Einfluss von Latenzzeiten auf das Systemverhalten von RVPen zu kompensieren, welche als der dominanteste Netzwerkeffekt identifiziert wurden. Alle entstandenen Teilaspekte der Kopplungsmethodik sind dabei entweder generisch anwendbar oder folgen, falls eine Parametrierung notwendig ist, der Vorgabe nur Informationen zu benötigen, die direkt aus den Koppelsignalen des RVPs bestimmbar sind und kein Systemwissen über die gekoppelten Prototypen voraussetzen. Dadurch ist die Kopplungsmethodik auch dann verwendbar, wenn die einzelnen Prototypen nur als Black-Box vorliegen. Dies ist insbesondere in der Industrie häufig der Fall, da dadurch der Austausch von Prototypen zwischen Unternehmen bei gleichzeitigem Schutz deren geistigen Eigentums ermöglicht wird.

Den Kern der Kopplungsmethodik bildet ein neuartiges Verfahren zur Extrapolation im Fehlerraum (EROS) aus dem zwei lineare Kopplungsalgorithmen konstruiert wurden, die generisch bei RVPen anwendbar sind. Ein detaillierter Vergleich der entwickelten mit bestehenden Kopplungsalgorithmen aus der Literatur zeigt, dass solche Algorithmen die in Abschnitt 3.1 definierten Anforderungen am besten erfüllen, die genau wie die neuartigen EROS-Algorithmen als ARIMA$(p, d, 0)$ Modelle klassifizierbar sind. Da die Klasse von ARIMA Modelle in vorwärts gerichtete neuronale Netze mit identischem Verhalten überführbar sind, wurden

etablierten Methoden aus dem Bereich des maschinellen Lernens angewandt, um die Kopplungsalgorithmen zum einen um die Eigenschaft der Lernfähigkeit zu erweitern und ihnen zum anderen eine verbesserte Latenzzeitkompensation stark nichtlinearer Koppelsignale zu ermöglichen.

Zur Abschätzung des Einflusses von Kopplungsalgorithmen auf das Systemverhalten von RVPen wurde der Kopplungsprozess inklusive Netzwerkeffekten und Kopplungsalgorithmus im Frequenzbereich analysiert und daraus Gültigkeitsbereiche für die untersuchten Kopplungsalgorithmen in Abhängigkeit von den Frequenzanteilen der Koppelsignale und den Latenzzeiten festgelegt. Basierend auf der allgemeinen Frequenzbereichsanalyse des Kopplungsprozesses wurde in Abschnitt 4.2 ein Verfahren entwickelt, um einen Kopplungsalgorithmus optimal für die Latenzzeitkompensation eines bestimmten Koppelsignals auszulegen. Weiterhin wurde anhand von Beispielsystemen, bei denen eine Übertragungsfunktion der gekoppelten Prototypen verfügbar ist, demonstriert wie diese Analyse für Stabilitätsuntersuchungen von RVPen nutzbar ist. Allerdings wurde dies in dieser Arbeit nur für RVPen gezeigt, welche aus zwei Prototypen bestehen und in einer der beiden Übertragungsrichtungen nur ein Signal austauschen (s. Abschnitt 4.3). Daher sollten die Stabilitätsuntersuchungen in weiterführenden Arbeiten erweitert werden, um die Analyse größere Prototypennetzwerke mit einer größeren Anzahl an Koppelsignalen zu ermöglichen und damit ihren praktischen Nutzen zu erhöhen. Ein möglicher Ansatz dafür könnte im Bereich der Mehrgrößenregelung über das Verfahren der Singulärwertzerlegung gefunden werden.

Alle Kopplungsalgorithmen, die effektiv die Latenzzeiten in Koppelsignalen kompensieren können führen aufgrund des Wasserbetteffekts, welcher in Abschnitt 4.2.1 diskutiert wurde, zu einer Amplitudenverstärkung höherer Frequenzen. Dadurch waren diese Kopplungsalgorithmen in der Vergangenheit nur verwendbar, wenn im gesamten Signalverlauf keine Frequenzen vorkommen, die außerhalb ihres Gültigkeitsbereichs liegen, wodurch beispielsweise Signale ausgeschlossen waren in denen vereinzelt Diskontinuitäten vorkommen. Um den Anwendungsbereich von Kopplungsalgorithmen zu vergrößern und nicht im Voraus bereits Wissen über eventuell vorhandene Diskontinuitäten im Signalverlauf zu benötigen, wurden in dieser Arbeit zwei Verfahren entwickelt, die in Kombination mit als ARIMA$(p, d, 0)$ klassifizierbaren Kopplungsalgorithmen verwendbar sind. Das erste Verfahren, dessen Funktionsweise in Abschnitt 5.3 beschrieben wurde, erkennt Diskontinuitäten in Koppelsignalen von RVPen, bevor ein Kopplungsalgorithmus angewandt wird. Dies ermöglicht ein rechtzeitiges und temporäres Umschalten des ausgewählten Kopplungsalgorithmus auf einen Algorithmus, der die hohen Frequenzanteile infolge der Diskontinuität nicht verstärkt, wodurch große Kopplungsfehler verhindert werden. Das zweite Verfahren hat zum Ziel online den Kopplungsfehler zu bestimmen und damit die Extrapolationsgüte zu bewerten. Wird die Güte als nicht ausreichend bewertet, kann eine Warnung ausgegeben und gegebenenfalls die Ausführung des RVPs gestoppt werden.

Für die Implementierung der gesamten Kopplungsmethodik, welche hauptsächlich aus Kopplungsalgorithmus, dessen online Anpassung, der Diskontinuitäten-Erkennung und der Bewertung der Extrapolationsgüte besteht, wurde der FMI-Standard verwendet. Dadurch ist die Anwendung der Kopplungsmethodik unabhängig eines bestimmten Simulationswerkzeugs oder Prüfstand-Steuerungssystems für RVPen aus den unterschiedlichsten Branchen, in denen vernetzte mechatronische Systeme entwickelt werden möglich, wie es zum Beispiel in der Fahrzeugindustrie oder der Luft- und Raumfahrtindustrie der Fall ist.

In dieser Arbeit wurde das Potenzial der Kopplungsmethodik für die Kompensation von Latenzzeiten durch den Einsatz an einem produktiv genutzten RVP unter realen Bedingungen demonstriert. Der untersuchte RVP stellt ein Hybridfahrzeug dar und besteht aus zwei realen und einem virtuellen Prototyp. Es zeigte sich, dass vor allem der im Frequenzbereich optimal ausgelegte Kopplungsalgorithmus den durch Latenzzeiten verursachten Kopplungsfehler deutlich verringerte. Anhand einer zusätzlich verfügbaren rein virtuellen Repräsentation des RVPs wurde weiterhin bestätigt, dass die Reduktion des Kopplungsfehlers außerdem das Systemverhalten des RVPs und damit die in Zukunft mit dem RVP erzielten Ergebnisse bezüglich optimaler Hybridstrategien verbessert. Diese Resultate konnten durch die oben beschriebene online Anpassung des Kopplungsalgorithmus nach dessen Konvertierung in ein neuronales Netz nochmals verbessert werden. Der Nutzen des Verfahrens zur Erkennung von Diskontinuitäten bestätigte sich ebenfalls am produktiv genutzten RVP. Trotz hochfrequenter Störungen im Koppelsignal konnte dadurch auch mit den generischen Kopplungsalgorithmen der Kopplungsfehler verringert werden. Allerdings führte das Verfahren zur online Bewertung der Extrapolationsgüte an diesem produktiv genutzten RVP nicht zur erhofften Überwachung der Kopplungsalgorithmen, da sich das aus der Frequenzbereichsanalyse der Kopplungsalgorithmen abgeleitete generische Kriterium zur Bewertung der Extrapolationsgüte als nicht zielführend erwies (s. Abschnitt 6.2.4). In zukünftige Arbeiten sollte dieses Kriterium weiter verbessert werden, um auch unter herausfordernden technischen Randbedingungen online eine aussagekräftige Bewertung der aktuellen Extrapolationsgüte zu erhalten.

Zusammengefasst leistet die in dieser Arbeit entwickelte Kopplungsmethodik, welche für den Einsatz in der Industrie geeignet ist, einen Beitrag zur durchgängigen Verwendung von Simulationsmodellen im Entwicklungsprozess domänenübergreifender mechatronischer Systeme, indem sie die Kopplung von Simulationsmodellen mit bereits vorliegenden Hardwarekomponenten des Gesamtsystems zu einem RVP durch eine Kompensation des negativen Einflusses von Netzwerkeffekten verbessert.

# Anhang

**Laplace-Transformation des SMBs**

Ausgehend von Gl. (3-33) mit eingesetztem $e_{n-1} = \hat{y}_{n-1} - y_{n-1}$ folgt

$$\hat{y}(t) = \hat{y}_n + (y_{n-k} - y_{n-k-1} - \lambda T(\hat{y}_{n-k} - y_{n-k}))\frac{(t-t_n)}{\Delta T} \text{ für } t_n \le t < t_{n+1}, \qquad (0\text{-}1)$$

dessen Laplace-Transformation aufgrund der abschnittsweisen Definition mit $t_n = n\Delta T$ als

$$\hat{Y}(s) = \sum_{n=0}^{\infty} \int_{n\Delta T}^{(n+1)\Delta T} \hat{y}(t)e^{-st}dt \qquad (0\text{-}2)$$

gegeben ist (vgl. Gl. (4-16)). Nach Aufteilung des Integrals und Anwendung des Verschiebungssatzes gilt

$$\begin{aligned}\hat{Y}(s) = &\sum_{n=0}^{\infty} \int_{n\Delta T}^{(n+1)\Delta T} \hat{y}_n e^{-st}dt \\ &+ \int_{n\Delta T}^{(n+1)\Delta T} \left(y_n e^{-ks\Delta T} - y_n e^{-(k+1)s\Delta T} - \lambda T(\hat{y}_n e^{-ks\Delta T} - y_n e^{-ks\Delta T})\right)\frac{(t-t_n)}{\Delta T}e^{-st}dt.\end{aligned} \qquad (0\text{-}3)$$

Die Lösung der Integrale erfolgt analog zu Gl. (4-17) und Gl. (4-18), wodurch sich

$$\begin{aligned}\hat{Y}(s) = &\frac{1-e^{s\Delta T}}{s}\underbrace{\sum_{n=0}^{\infty}\hat{y}_n\, e^{-sn\Delta T}}_{\hat{Y}_a(s)} - \frac{1-(1+s\Delta T)e^{-s\Delta T}}{s^2\Delta T}\lambda\Delta T e^{-ks\Delta T}\underbrace{\sum_{n=0}^{\infty}\hat{y}_n\, e^{-sn\Delta T}}_{\hat{Y}_a(s)} \\ &+ \frac{1-(1+s\Delta T)e^{-s\Delta T}}{s^2\Delta T}\left(1-e^{-s\Delta T}+\lambda\Delta T\right)e^{-ks\Delta T}\underbrace{\sum_{n=0}^{\infty}y_n\, e^{-sn\Delta T}}_{Y_a(s)}\end{aligned} \qquad (0\text{-}4)$$

ergibt. Nach Gl.(4-7) gilt für die Laplace-Transformierte des abgetasteten geschätzten Signals die Beziehung $\hat{Y}_a(s) = \frac{1}{\Delta T}\hat{Y}(s)$.

Die Übertragungsfunktion des SMBs folgt nach Auflösen der Gl. (0-4) nach

$$G_{SMB}(s) = \frac{\hat{Y}(s)}{Y(s)} \text{ zu} \qquad (0\text{-}5)$$

$$G_{SMB}(s) = \frac{\left(1-(1+s\Delta T)e^{-s\Delta T}\right)\left(1-e^{-s\Delta T}+\lambda\Delta T\right)e^{-ks\Delta T}}{s^2\Delta T^2-(1-e^{-s\Delta T})s\Delta T+(1-(1+s\Delta T)e^{-s\Delta T})\lambda\Delta T\, e^{-ks\Delta T}}.$$

Dabei handelt es sich um eine transzendente Funktion mit Totzeiten im Nenner. Trotzdem ist die Anzahl der Polstellen endlich, da der Summand $s^2\Delta T^2$ die anderen Summanden für große $s$ dominiert. Unabhängig der Parameter weist die Übertragungsfunktion eine doppelte Polstelle bei $s = 0$ auf. Die Lage der gegebenenfalls weiteren Polstellen hängt von den Netzwerkeffekten $\Delta T$ und $k$ sowie dem Designparameter $\lambda$ ab und wird hier aufgrund der Komplexität des Nenners von $G_{SMB}(s)$ numerisch bestimmt.

Für die in dieser Arbeit beispielhaft an mehreren Stellen verwendeten Netzwerkeffekte mit $\Delta T = 0{,}02$ und $k = 3$ ergeben sich bei Betrachtung des Stabilitätsgebiets des kontinuierlichen SMB $0 < \lambda < \bar{\lambda} = \pi/2\Delta T k$ weitere komplex konjugierten Polpaare. Eines liegt im Bereich $|s| \approx \lambda$ und ist nur für $\lambda < 0{,}89\bar{\lambda}$ stabil. Die weiteren Polpaare sind betragsmäßig größer und im gesamten betrachteten Bereich stabil. Daher ist der Stabilitätsbereich des diskreten SMBs kleiner als der des kontinuierlichen.

Die Grenzwerte von $G_{SMB}(s)$ ergeben sich mittels der Regel von de L'Hospital zu

$$\begin{aligned} \lim_{s\to 0-} G_p(s) &= \lim_{\omega\to 0+} G_p(s) = 1 \text{ und} \\ \lim_{s\to -\infty} G_p(s) &= \lim_{s\to +\infty} G_p(s) = 0. \end{aligned} \tag{0-6}$$

# Literaturverzeichnis


AARSETH, J., A.H. LIEN, O. BUNES, Y. CHU, V. ÆSØY, R.-R. MARINE und A. HELGE, 2014. A Hardware-In-The-Loop Simulator For Offshore Machinery Control System Testing. In: *ECMS*, S. 57-63.

ABEL, D. und A. BOLLIG, 2006. *Rapid control prototyping:* Springer. ISBN 3540295240.

AGIWAL, M., A. ROY und N. SAXENA, 2016. Next generation 5G wireless networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, **18**(3), 1617-1655. ISSN 1553-877X.

ALAN V. OPPENHEIM, ALAN S. WILLSKY und S. HAMID NAWAB, 1996. *Signals & systems (2nd ed.):* Prentice-Hall, Inc. ISBN 0138147574.

ALBERS, A., J. REINEMANN, J. FAHL und T. HIRSCHTER, 2019. Augmented Reality for Product Validation: Supporting the Configuration of AR-Based Validation Environments. In: *International Conference on Human-Computer Interaction:* Springer, S. 429-448.

ALBERS, A., M. BEHRENDT, J. SCHROETER, S. OTT und S. KLINGLER, 2013. X-in-the-loop: A framework for supporting central engineering activities and contracting complexity in product engineering processes. In: *DS 75-6: Proceedings of the 19th International Conference on Engineering Design (ICED13), Design for Harmonies, Vol. 6: Design Information and Knowledge, Seoul, Korea, 19-22.08.*, S. 391-400. ISBN 1904670490.

ALFONSO, J., J.M. RODRIGUEZ, C. BERNAD, V. BELIAUTSOU, V. IVANOV und J.A. CASTELLANOS, 2022. Geographically distributed real-time co-simulation of electric vehicle. In: *2022 8th International Conference on Control, Decision and Information Technologies (CoDIT)*, S. 1002-1007.

AMINIKHANGHAHI, S. und D.J. COOK, 2017. A survey of methods for time series change point detection. *Knowledge and information systems*, **51**(2), 339-367. ISSN 0219-1377.

AUGSBURG, K., S. GRAMSTAT, R. HORN, V. IVANOV, H. SACHSE und B. SHYROKAU, 2011. *Investigation of brake control using test rig-in-the-loop technique.* SAE Technical Paper.

BAILLIEUL, J. und P.J. ANTSAKLIS, 2007. Control and communication challenges in networked real-time systems. *Proceedings of the IEEE*, **95**(1), 9-28. ISSN 0018-9219.

BAUMANN, P., L. MIKELSONS, O. KOTTE und D. SCHRAMM, 2019a. Analyzing the coupling process of distributed mixed real-virtual prototypes. In: P. FUSTER-PARRA, Ó.V. SIERRA und P. GERIL, Hg. *Modelling and simulation 2019: the European Simulation and Modelling Conference 2019 / ESM '19,* S. 251-258. ISBN 978-94-92859-09-9.

BAUMANN, P., M. KRAMMER, M. DRIUSSI, L. MIKELSONS, J. ZEHETNER, W. MAIR und D. SCHRAMM, 2019b. Using the Distributed Co-Simulation Protocol for a Mixed Real-Virtual Prototype. In: *Proceedings of 2019 IEEE International Conference on Mechatronics.* Ilmenau, Germany: IEEE Industrial Electronics Society.

BAUMANN, P., O. KOTTE, L. MIKELSONS und D. SCHRAMM, 2024. Enhancing the Coupling of Real-Virtual Prototypes: A Method for Latency Compensation [online]. *Electronics*, **13**(6). Electronics. Verfügbar unter: doi:10.3390/electronics13061077

BAUMANN, P., R. SAMLAUS, L. MIKELSONS, T. KUHN und J. JAHIC, 2019c. Towards virtual validation of distributed functions. In: *Proceedings of the 2019 Summer Simulation Conference:* Society for Computer Simulation International, S. 1-12.

BAUMANN, P., R. SAMLAUS, L. MIKELSONS, T. KUHN, J. JAHIC und D. SCHRAMM, 2019d. Ein Beitrag zur Simulation vernetzter virtueller ECUs. In: *Fachtagung Mechatronik,* S. 13-18.

BEMPORAD, A., 1998. Predictive control of teleoperated constrained systems with unbounded communication delays. In: *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No. 98CH36171):* IEEE, S. 2133-2138. ISBN 0780343948.

BEN KHALED-EL FEKI, A., L. DUVAL, C. FAURE, D. SIMON und M. BEN GAID, 2017. CHOPtrey: contextual online polynomial extrapolation for enhanced multi-core co-simulation of complex systems. *Simulation*, **93**(3), 185-200. ISSN 0037-5497.

BENAJES, J., A. GARCÍA, J. MONSALVE-SERRANO und S. MARTÍNEZ-BOGGIO, 2019. Optimization of the parallel and mild hybrid vehicle platforms operating under conventional and advanced combustion modes. *Energy Conversion and Management*, **190**, 73-90. ISSN 0196-8904.

BENEDIKT, M., 2012. *Eine Kopplungsmethode für die nicht-iterative Co-Simulation:* PhD thesis.

BENEDIKT, M., D. WATZENIG und A. HOFER, 2013. Modelling and analysis of the non-iterative coupling process for co-simulation. *Mathematical and Computer Modelling of Dynamical Systems*, **19**(5), 451-470. ISSN 1387-3954.

BENEDIKT, M., D. WATZENIG, J. ZEHETNER und A. HOFER, 2013. Macro-step-size selection and monitoring of the coupling errof for weak coupled subsystems in the frequency-domain. In: *Proceedings of the International Conference on Computational Methods for Coupled Problems in Science and Engineering*, S. 1-12.

BENEDIKT, M., H. STIPPEL und D. WATZENIG, 2010. *An adaptive coupling methodology for fast time-domain distributed heterogeneous co-simulation.* SAE Technical Paper.

BERNHART, W., 2018. Rennen um die Poleposition im künftigen (auto-) mobilen Ökosystem. *ATZextra*, **23**(5), 12-15. ISSN 2195-1454.

BLOCHWITZ, T., M. OTTER, J. AKESSON, M. ARNOLD, C. CLAUß, H. ELMQVIST, M. FRIEDRICH, A. JUNGHANNS, J. MAUß, D. NEUMERKEL, H. OLSSON und A. VIEL, 2012. Functional Mockup Interface 2.0: The Standard for Tool independent Exchange of Simulation Models. In: *Proceedings of the 9th International MODELICA Conference:* Linköping University Electronic Press, S. 173-184. ISBN 1650-3740.

BLOOMFIELD, P., 2004. *Fourier analysis of time series: an introduction:* John Wiley & Sons.

BOSCHERT, S. und R. ROSEN, 2016. Digital twin—the simulation aspect. In: *Mechatronic futures:* Springer, S. 59-74.

BOUMANS, M., A. GALLET und U. SCHULMEISTER, 2017. *XDomain Simulation as Backbone for Vehicle System Engineering*. 10th Graz Symposium Virtual Vehicle, 2017.

BOX, G.E.P. und G.M. JENKINS, 1970. *Time series analysis: forecasting and control.* San Francisco: Holden-Day.

BRANDT, A., 2023. *Noise and vibration analysis: signal analysis and experimental procedures:* John Wiley & Sons. ISBN 1118962184.

BUSCH, M., 2012. *Zur effizienten Kopplung von Simulationsprogrammen:* kassel university press GmbH. ISBN 3862192962.

BUSCH, M., 2016. Continuous approximation techniques for co-simulation methods: Analysis of numerical stability and local error. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, **96**(9), 1061-1081. ISSN 0044-2267.

CALE, J.L., B.B. JOHNSON, E. DALL'ANESE, P.M. YOUNG, G. DUGGAN, P.A. BEDGE, D. ZIMMERLE und L. HOLTON, 2018. Mitigating communication delays in remotely connected hardware-in-the-loop experiments. *IEEE Transactions on Industrial Electronics*, **65**(12), 9739-9748. ISSN 0278-0046.

CELLIER, F.E. und E. KOFMAN, 2006. *Continuous system simulation:* Springer Science & Business Media. ISBN 0387261028.

CHEN, T., I. GOODFELLOW und J. SHLENS, 2015. Net2net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641.* arXiv preprint arXiv:1511.05641.

CHENG, C., A. SA-NGASOONGSONG, O. BEYCA, T. LE, H. YANG, Z. KONG und S.T.S. BUKKAPATNAM, 2015. Time series forecasting for nonlinear and non-stationary processes: A review and comparative study. *Iie Transactions*, **47**(10), 1053-1071. ISSN 0740-817X.

COOK, A.A., G. MISIRLI und Z. FAN, 2019. Anomaly detection for IoT time-series data: A survey. *IEEE Internet of Things Journal*, **7**(7), 6481-6494. IEEE Internet of Things Journal.

DORFFNER, G., 1996. Neural networks for time series processing. *Neural network world.* Neural network world.

DRENTH, E., 2016. Robust co-simulation methodology of physical systems. In: *9th Graz Symposium Virtual Vehicle*.

DRONKA, S., 2004. *Die Simulation gekoppelter Mehrkörper-und Hydraulik-Modelle mit Erweiterung für Echtzeitsimulation:* Shaker. ISBN 3832229809.

DSPACE, 2020. *SCALEXIO - Modulares Echtzeitsystem* [online] [Zugriff am: 13. Februar 2020]. Verfügbar unter: https://www.dspace.com/de/gmb/home/products/hw/simulator_hardware/scalexio.cfm

DÜSER, T., 2010. *X-in-the-Loop-ein durchgängiges Validierungsframework für die Fahrzeugentwicklung am Beispiel von Antriebsstrangfunktionen und Fahrerassistenzsystemen:* IPEK.

EIDSON, J. und K. LEE, 2002. IEEE 1588 standard for a precision clock synchronization protocol for networked measurement and control systems. In: *Proceedings of the 2nd ISA/IEEE Sensors for Industry Conference,* S. 98-105.

EILERS, S. und C. MÜLLER-SCHLOER, 2005. Mixed Virtual/Real Prototypes for Incremental System Design – A Proof of Concept. In: *Embedded Computer Systems: Architectures, Modeling, and Simulation:* Springer Berlin Heidelberg, S. 465-474. ISBN 978-3-540-31664-0.

ERSAL, T., M. BRUDNAK, A. SALVI, J.L. STEIN, Z. FILIPI und H.K. FATHY, 2009. Development of an Internet-distributed hardware-in-the-loop simulation platform for an automotive application. In: *ASME 2009 Dynamic Systems and Control Conference:* American Society of Mechanical Engineers Digital Collection.

ERSAL, T., M. BRUDNAK, A. SALVI, Y. KIM, J.B. SIEGEL und J.L. STEIN, 2014. An iterative learning control approach to improving fidelity in internet-distributed hardware-in-the-loop simulation. *Journal of dynamic systems, measurement, and control*, **136**(6), 61012. ISSN 0022-0434.

ETAS, 2020. *LABCAR - Modulares HiL-Testsystem für automotive Steuergeräte* [online] [Zugriff am: 13. Februar 2020]. Verfügbar unter: https://www.etas.com/de/anwendungen/applications_labcar_component_overview.php

FATHY, H.K., Z.S. FILIPI, J. HAGENA und J.L. STEIN, 2006. Review of hardware-in-the-loop simulation and its prospects in the automotive area. In: *Modeling and simulation for military applications:* International Society for Optics and Photonics, 62280E.

FRITZSON, P., 2010. *Principles of object-oriented modeling and simulation with Modelica 2.1:* John Wiley & Sons. ISBN 0470937610.

FUJIMOTO, R., 2015. Parallel and distributed simulation. In: *Winter Simulation Conference (WSC):* IEEE, S. 45-59. ISBN 1467397431.

GALL, H.A., 2001. Zur simulation von Haftreibung und mechanischen Anschlägen. *ASIM Nachrichten*, **1**, 4-9. ASIM Nachrichten.

GAN, C., R. TODD und J. APSLEY, 2014. Mitigating time delays: an evaluation of their impact using a simulation model of an aircraft power system demonstrator facility. *IEEE Industry Applications Magazine*, **21**(2), 44-53. ISSN 1077-2618.

GE, X., M.J. BRUDNAK, J.L. STEIN und T. ERSAL, 2014. A norm optimal iterative learning control framework towards internet-distributed hardware-in-the-loop simulation. In: *American Control Conference:* IEEE, S. 3802-3807. ISBN 1479932744.

GE, X., Y. ZHENG, M.J. BRUDNAK, P. JAYAKUMAR, J.L. STEIN und T. ERSAL, 2017. Analysis of a model-free predictor for delay compensation in networked systems. In: *Time Delay Systems:* Springer, S. 201-215.

GEIMER, M., T. KRÜGER und P. LINSEL, 2006. Co-simulation, gekoppelte simulation oder simulatorkopplung. *O+ P Zeitschrift für Fluidtechnik*, **50**(11-12), 572-576. O+ P Zeitschrift für Fluidtechnik.

GLUMAC, S. und Z. KOVACIC, 2019. Relative consistency and robust stability measures for sequential co-simulation. In: *Proceedings of the 13th International Modelica Conference:* Linköping University Electronic Press.

GOMES, C., C. THULE, D. BROMAN, P.G. LARSEN und H. VANGHELUWE, 2018. Co-simulation: a survey. *ACM Computing Surveys (CSUR)*, **51**(3), 49. ISSN 0360-0300.

GROSS, J.L., 1967. Real time hardware-in-the-loop simulation verifies performance of Gemini computer and operational program. *Simulation*, **9**(3), 141-148. ISSN 0037-5497.

GÜHMANN, C., 2005. Model-based testing of automotive electronic control units. In: *3rd International Conference on Materials Testing: Test*.

GUILLO-SANSANO, E., A.J. ROSCOE, C.E. JONES und G.M. BURT, 2014. A new control method for the power interface in power hardware-in-the-loop simulation to compensate for the time delay. In: *49th International Universities Power Engineering Conference (UPEC):* IEEE, S. 1-5. ISBN 147996557X.

GÜNTHER, F.C., 2017. *Beitrag zur Co-Simulation in der Gesamtsystementwicklung des Kraftfahrzeugs:* Technische Universität München.

GUPTA, M., J. GAO, C.C. AGGARWAL und J. HAN, 2013. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and data Engineering*, **26**(9), 2250-2267. ISSN 1041-4347.

HANSEN, S.T., C.Â.G. GOMES, M. NAJAFI, T. SOMMER, M. BLESKEN, I. ZACHARIAS, O. KOTTE, P.R. MAI, K. SCHUCH und K. WERNERSSON, 2022. The FMI 3.0 Standard Interface for Clocked and Scheduled Simulations. *Electronics*, **11**(21), 3635. Electronics.

HATLEDAL, L.I., H. ZHANG, A. STYVE und G. HOVLAND, 2019. FMU-proxy: A Framework for Distributed Access to Functional Mock-up Units. In: *Proceedings of the 13th International Modelica Conference:* Linköping University Electronic Press.

HIMMLER, A., 2014. *From Virtual Testing to HIL Testing-Towards Seamless Testing.* SAE Technical Paper.

HINES, K. und G. BORRIELLO, 1997. Selective focus as a means of improving geographically distributed embedded system co-simulation. In: *Proceedings 8th IEEE International Workshop on Rapid System Prototyping Shortening the Path from Specification to Prototype:* IEEE, S. 58-62. ISBN 0818680644.

HIRZ, M., W. DIETRICH, A. GFRERRER und J. LANG, 2013. Overview of virtual product development. In: *Integrated Computer-Aided Design in Automotive Development:* Springer, S. 25-50.

HOLIŠ, R., V. BOBÁL und J. VOJTĚŠEK, 2017. Real-time digital control of time-delay systems: From smith predictor to MPC. In: *International Conference on Engineering, Technology and Innovation (ICE/ITMC):* IEEE, S. 254-263. ISBN 1538607743.

HOLZINGER, F.R. und M. BENEDIKT, 2014. Online instantaneous frequency estimation utilizing empirical mode decomposition and Hermite splines. In: *22nd European Signal Processing Conference (EUSIPCO):* IEEE, S. 446-450. ISBN 0992862612.

HUANG, N.E., Z. WU, S.R. LONG, K.C. ARNOLD, X. CHEN und K. BLANK, 2009. On instantaneous frequency. *Advances in adaptive data analysis*, **1**(02), 177-229. Advances in adaptive data analysis.

ISERMANN, R., J. SCHAFFNIT und S. SINSEL, 1999. Hardware-in-the-loop simulation for the design and testing of engine-control systems. *Control engineering practice*, **7**(5), 643-653. ISSN 0967-0661.

ITEA3, 2020. *UPSIM - Unleash Potentials in Simulation* [online] [Zugriff am: 23. April 2020]. Verfügbar unter: https://itea3.org/project/upsim.html

IVANOV, V., K. AUGSBURG, C. BERNAD, M. DHAENS, M. DUTRÉ, S. GRAMSTAT, P. MAGNIN, V. SCHREIBER, U. SKRT und N. VAN KELECOM, 2019. Connected and Shared X-in-the-Loop Technologies for Electric Vehicle Design. *World Electric Vehicle Journal*, **10**(4), 83. World Electric Vehicle Journal.

JANSCHEK, K., 2010. Regelungstechnische Aspekte. *Systementwurf mechatronischer Systeme: Methoden–Modelle–Konzepte*, 655-758. ISSN 35407887.

JEON, J.-H., J.-Y. KIM, H.-M. KIM, S.-K. KIM, C. CHO, J.-M. KIM, J.-B. AHN und K.-Y. NAM, 2010. Development of hardware in-the-loop simulation system for testing operation and control functions of microgrid. *IEEE Transactions on Power Electronics*, **25**(12), 2919-2929. ISSN 0885-8993.

JIANG, J., Y. LI, S.H. HONG, M. YU, A. XU und M. WEI, 2019. A Simulation Model for Time-sensitive Networking (TSN) with Experimental Validation. In: *24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA):* IEEE, S. 153-160. ISBN 1728103037.

JUGO, J., 2001. On the stability of time-delay systems using Nyquist criterion. In: *European Control Conference (ECC):* IEEE, S. 2717-2722. ISBN 395241736X.

JUNGHANNS, A., J. MAUSS und M. SEIBT, 2014. Faster Development of AUTOSAR compliant ECUs through simulation. In: *ERTS 2014 - Embedded Real Time Software and Systems*.

JUNGHANNS, A., R. SERWAY, T. LIEBEZEIT und M. BONIN, 2012. Building virtual ECUs quickly and economically. *ATZelektronik worldwide*, **7**(3), 48-51. ISSN 2192-9092.

KAUNE, S., K. PUSSEP, C. LENG, A. KOVACEVIC, G. TYSON und R. STEINMETZ, 2009. Modelling the internet delay space based on geographical locations. In: *17th Euromicro International Conference on Parallel, Distributed and Network-based Processing:* IEEE, S. 301-310. ISBN 0769535445.

KIM, K.-D. und P.R. KUMAR, 2012. Cyber–physical systems. A perspective at the centennial. *Proceedings of the IEEE*, **100**(Special Centennial Issue), 1287-1308. ISSN 0018-9219.

KINGMA, D.P. und J. BA, 2014. Adam. A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.* arXiv preprint arXiv:1412.6980.

KLEIN, S., F. XIA, K. ETZOLD, J. ANDERT, N. AMRINGER, S. WALTER, T. BLOCHWITZ und C. BELLANGER, 2018a. Electric-Motor-in-the-Loop: Efficient Testing and Calibration of Hybrid Power Trains. *IFAC-PapersOnLine*, **51**(31), 240-245. ISSN 2405-8963.

KLEIN, S., P. GRIEFNOW, D. GUSE, F. XIA und J. ANDERT, 2018b. Virtual 48V Mild Hybridization. Efficient Validation by Engine-in-the-Loop. In: *SAE World Congress Experience WCX 2018*.

KLEIN, S., R. SAVELSBERG, F. XIA, D. GUSE, J. ANDERT, T. BLOCHWITZ, C. BELLANGER, S. WALTER, S. BERINGER, J. JOCHHEIM und N. AMRINGER, 2017. Engine in the Loop. Closed Loop Test Bench Control with Real-Time Simulation. *SAE International Journal of Commercial Vehicles*, **10**(1). ISSN 1946-3928.

KRAMMER, M., C. SCHIFFER und M. BENEDIKT, 2021. ProMECoS: A process model for efficient standard-driven distributed co-simulation. *Electronics*, **10**(5), 633. Electronics.

KRAMMER, M., M. BENEDIKT, T. BLOCHWITZ, K. ALEKEISH, N. AMRINGER, C. KATER, S. MATERNE, R. RUVALCABA, K. SCHUCH, J. ZEHETNER, M. DAMM-NORWIG, V. SCHREIBER, N. NAGARAJAN, I. CORRAL, T. SPARBER, S. KLEIN und J. ANDERT, 2018. The Distributed Co-Simulation Protocol for the Integration of Real-Time Systems and Simulation Environments. In: *Proceedings of the 50th Computer Simulation Conference:* Society for Computer Simulation International.

KRAMMER, M., P. FERNER und D. WATZENIG, 2019. Clock Synchronization in Context of the Distributed Co-Simulation Protocol. In: *IEEE International Conference on Connected Vehicles and Expo (ICCVE):* IEEE, S. 1-6. ISBN 1728101425.

KÜBLER, R. und W. SCHIEHLEN, 2000. Two methods of simulator coupling. *Mathematical and Computer Modelling of Dynamical Systems*, **6**(2), 93-113. ISSN 1387-3954.

KUHN, T., T. FORSTER, T. BRAUN und R. GOTZHEIN, 2013. FERAL—Framework for simulator coupling on requirements and architecture level. In: *11th ACM/IEEE International Conference*.

KWON, W.H., S.H. HAN, J.S. LEE und H.S. KIM, 1998. Real-Time Software-In-the-Loop Simulation for Control Education. *Submitted to IEEE Trans. Education.* Submitted to IEEE Trans. Education.

LACHENMAIER, S., L. CROSS, C. FERRARA, A. GREIS, M. WÜST und D. NABER, 2020. Virtual powertrain–Vehicle simulation on the engine test bench with an implemented P2 topology. In: *20. Internationales Stuttgarter Symposium: Automobil-und Motorentechnik:* Springer, S. 377-392. ISBN 3658309946.

LAUSS, G.F., M.O. FARUQUE, K. SCHODER, C. DUFOUR, A. VIEHWEIDER und J. LANGSTON, 2015. Characteristics and design of power hardware-in-the-loop simulations for electrical power systems. *IEEE Transactions on Industrial Electronics*, **63**(1), 406-417. ISSN 0278-0046.

LAWRENCE, D.A., 1993. Stability and transparency in bilateral teleoperation. *IEEE transactions on robotics and automation*, **9**(5), 624-637. ISSN 1042-296X.

LAWRENCE, D.P.Y., C. GOMES, J. DENIL, H. VANGHELUWE und D. BUCHS, 2016. Coupling petri nets with deterministic formalisms using co-simulation. In: *Symposium on Theory of Modeling and Simulation (TMS-DEVS):* IEEE, S. 1-8. ISBN 1510823212.

LEE, E.A., 2015. The past, present and future of cyber-physical systems: A focus on models. *Sensors*, **15**(3), 4837-4869. Sensors.

LEE, M.H., H.M. LEE, K.S. LEE, S.K. HA, J.I. BAE, J.H. PARK, H.G. PARK, H.J. CHOI und H.H. CHUN, 2011. Development of a hardware in the loop simulation system for electric power steering in vehicles. *International journal of Automotive technology*, **12**(5), 733. ISSN 1229-9138.

LI, P., 2017. *On the Numerical Stability of Co-Simulation Methods.* PhD thesis.

LIU, A., W. ZHANG, L. YU, S. LIU und M.Z.Q. CHEN, 2015. New results on stabilization of networked control systems with packet disordering. *Automatica*, **52**, 255-259. ISSN 0005-1098.

LIU, M., I. DASSIOS, G. TZOUNAS und F. MILANO, 2020. Model-Independent Derivative Control Delay Compensation Methods for Power Systems. *Energies*, **13**(2), 342. Energies.

LU, S., M.Y. ZHANG, T. ERSAL und X.J. YANG, 2019. Workload Management in Teleoperation of Unmanned Ground Vehicles: Effects of a Delay Compensation Aid on Human Operators' Workload and Teleoperation Performance. *International Journal of Human–Computer Interaction*, 1-11. ISSN 1044-7318.

LUNZE, J., 2013. *Regelungstechnik 1:* Springer. 10.

MAAS, A.L., A.Y. HANNUN und A.Y. NG, 2018. Rectifier nonlinearities improve neural network acoustic models. In: *Proc. icml*.

MAAS, N., B. HESSE, M. KOPPERS und D. SCHRAMM, 2014. Simulator setup according to use case scenarios-A human-oriented method for virtual development. In: *IEEE/ASME 10th International Conference on Mechatronic and Embedded Systems and Applications (MESA):* IEEE, S. 1-6. ISBN 1479922803.

MELAIKA, M., S. MAMIKOGLU und P. DAHLANDER, 2019. *48V mild-hybrid architecture types, fuels and power levels needed to achieve 75g CO2/km.* 0148-7191.

MEYER, T., J. KRAFT und B. SCHWEIZER, 2021. Co-simulation: Error estimation and macro-step size control. *Journal of Computational and Nonlinear Dynamics*, **16**(4), 41002. ISSN 1555-1415.

MIHALIČ, F., M. TRUNTIČ und A. HREN, 2022. Hardware-in-the-loop simulations: A historical overview of engineering challenges. *Electronics*, **11**(15), 2462. Electronics.

MIKELSONS, L. und R. SAMLAUS, 2017. Towards Virtual Validation of ECU Software using FMI. In: *Proceedings of the 12th International Modelica Conference, Prague, Czech Republic:* Linköping University Electronic Press, S. 307-311. ISBN 1650-3740.

MIKELSONS, L., M. BAUMANN, O. KOTTE und P. BAUMANN, 2023. Method and device for synchronizing a simulation with a real-time system. United States Patent.

MILLS, D., J. MARTIN, J. BURBANK und W. KASCH, 2010. *Network time protocol version 4: Protocol and algorithms specification*.

NAGHSHTABRIZI, P. und J.P. HESPANHA, 2005. Designing an observer-based controller for a network control system. In: *Proceedings of the 44th IEEE Conference on Decision and Control:* IEEE, S. 848-853. ISBN 0780395670.

NANNAPANENI, R. und R.V. KULKARNI, 2018. Time Series Forecasting using Artificial Neural Networks. *SASTech-Technical Journal of RUAS*, **17**(2), 33-36. ISSN 2249-5924.

NATORI, K., T. TSUJI, K. OHNISHI, A. HACE und K. JEZERNIK, 2009. Time-delay compensation by communication disturbance observer for bilateral teleoperation under time-varying delay. *IEEE Transactions on Industrial Electronics*, **57**(3), 1050-1062. ISSN 0278-0046.

OPPENHEIM, A.V., 1999. *Discrete-time signal processing:* Pearson Education India. ISBN 8131704920.

PETRIDIS, K., 2014. *Synchrone und asynchrone Verfahren zur gekoppelten Simulation mechatronischer Systeme:* VDI-Verlag. ISBN 3183454203.

PETZOLD, L., 1983. Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations. *SIAM journal on scientific and statistical computing*, **4**(1), 136-148. ISSN 0196-5204.

PROTOPAPADAKIS, E., A. VOULODIMOS und N. DOULAMIS, 2017. An investigation on multi-objective optimization of feedforward neural network topology. In: *8th International Conference on Information, Intelligence, Systems & Applications (IISA):* IEEE, S. 1-6. ISBN 1538637316.

PÜTZ, A., A. ZLOCKI, J. BOCK und L. ECKSTEIN, 2017. System validation of highly automated vehicles with a database of relevant traffic scenarios. *situations*, **1**, 19-22. situations.

QTRONIC, 2018. FMU SDK: Free [Software]. Version 2.0.6 [Zugriff am: 16. März 2024]. Verfügbar unter: https://github.com/qtronic/fmusdk

RAUTENBERG, P., P. WEBER, J.P. DEGEL, S. HÄHNLEIN, F. GAUTERIN, T. KOCH, M. DOPPELBAUER und M. GOHL, 2023. Electrified Powertrain Development: Distributed Co-Simulation Protocol Extension for Coupled Test Bench Operations. *Applied Sciences*, **13**(4), 2657. Applied Sciences.

REINOLD, P., N. MEYER, D. BUSE, F. KLINGLER, C. SOMMER, F. DRESSLER, M. EISENBARTH und J. ANDERT, 2019. Verkehrssimulation im Hardware-in-the-Loop-Steuergerätetest. In: *Simulation und Test 2018:* Springer, S. 253-269.

SADJINA, S. und E. PEDERSEN, 2016. Energy conservation and coupling error reduction in non-iterative co-simulations. *arXiv preprint arXiv:1606.05168.* arXiv preprint arXiv:1606.05168.

SADJINA, S., L.T. KYLLINGSTAD, M. RINDARØY, S. SKJONG, V. ÆSØY und E. PEDERSEN, 2019. Distributed co-simulation of maritime systems and operations. *Journal of Offshore Mechanics and Arctic Engineering*, **141**(1), 11302. ISSN 0892-7219.

SADJINA, S., L.T. KYLLINGSTAD, S. SKJONG und E. PEDERSEN, 2017. Energy conservation and power bonds in co-simulations: non-iterative adaptive step size control and error estimation. *Engineering with Computers*, **33**(3), 607-620. ISSN 0177-0667.

SARHADI, P. und S. YOUSEFPOUR, 2015. State of the art: hardware in the loop modeling and simulation with its applications in design, development and implementation of system and control software. *International Journal of Dynamics and Control*, **3**(4), 470-479. ISSN 2195-268X.

SAURAV, S., P. MALHOTRA, V. TV, N. GUGULOTHU, L. VIG, P. AGARWAL und G. SHROFF, 2018. Online anomaly detection with concept drift adaptation using recurrent neural networks. In: *Proceedings of the acm india joint international conference on data science and management of data,* S. 78-87.

SCHIERZ, T., M. ARNOLD und C. CLAUß, 2012. Co-simulation with communication step size control in an FMI compatible master algorithm. In: *Proceedings of the 9th International MODELICA Conference:* Linköping University Electronic Press, S. 205-214. ISBN 1650-3740.

SCHMIDHUBER, J., 2015. Deep learning in neural networks: An overview. *Neural networks*, **61**, 85-117. ISSN 0893-6080.

SCHREIBER, V., V. IVANOV, K. AUGSBURG, M. NOACK, B. SHYROKAU, C. SANDU und P.S. ELS, 2018. Shared and Distributed X-in-the-Loop Tests for Automotive Systems. Feasibility Study. *IEEE Access*, **6**, 4017-4026. ISSN 2169-3536.

SCHWEIGER, G., C. GOMES, G. ENGEL, I. HAFNER, J. SCHOEGGL, A. POSCH und T. NOUIDUI, 2019. An empirical survey on co-simulation: Promising standards, challenges and research needs. *Simulation Modelling Practice and Theory.* ISSN 1569-190X.

SHIMMYO, S., Y. SAITO, T. NOZAKI und K. OHNISHI, 2019. Symmetric Operational Force Compensator for Bilateral Teleoperation under Time Delay Based on Power Flow Direction. In: *Proceedings of 2019 IEEE International Conference on Mechatronics.* Ilmenau, Germany: IEEE Industrial Electronics Society, S. 700-705.

SPEEDGOAT, 2020. *Hardware-in-the-Loop Testing (HIL) for Real-Time Plant Simulation* [online] [Zugriff am: 13. Februar 2020]. Verfügbar unter: https://www.speedgoat.com/applications-industries/applications/plant-simulation-hil

SPRAGUE, M.A. und T.L. GEERS, 2004. A spectral-element method for modelling cavitation in transient fluid–structure interaction. *International Journal for Numerical Methods in Engineering*, **60**(15), 2467-2499. ISSN 0029-5981.

STEINBRINK, C., S. LEHNHOFF, S. ROHJANS, T.I. STRASSER, E. WIDL, C. MOYO, G. LAUSS, F. LEHFUSS, M. FASCHANG und P. PALENSKY, 2017. Simulation-based validation of smart grids–status quo and future research trends. In: *International Conference on Industrial Applications of Holonic and Multi-Agent Systems:* Springer, S. 171-185.

STETTINGER, G., J. ZEHETNER, M. BENEDIKT und N. THEK, 2013. Extending Co-Simulation to the Real-Time Domain. In: *SAE 2013 World Congress & Exhibition:* SAE International400 Commonwealth Drive, Warrendale, PA, United States.

STETTINGER, G., J. ZEHETNER, M. BENEDIKT, H. KOKAL, M. PAULWEBER, M. WIERSE und B. TOYE, 2014a. Control of an engine test-bench via hardware-software-co-simulation. In: *Internationales Stuttgarter Symposium Automobil-und Motorentechnik:* Springer Fachmedien Wiesbaden GmbH, S. 869-880.

STETTINGER, G., M. BENEDIKT, M. HORN und J. ZEHETNER, 2015. Modellbasierte Echtzeit-Co-Simulation. Überblick und praktische Anwendungsbeispiele. *e & i Elektrotechnik und Informationstechnik*, **132**(4-5), 207-213. e & i Elektrotechnik und Informationstechnik.

STETTINGER, G., M. BENEDIKT, M. HORN, J. ZEHETNER und C. GIEBENHAIN, 2016. Control of a magnetic levitation system with communication imperfections [online]. A model-based coupling approach. *Control engineering practice*, **58 (2017)**, 161-170. ISSN 0967-0661. Verfügbar unter: doi:10.1016/j.conengprac.2016.10.009

STETTINGER, G., M. BENEDIKT, M. TRANNINGER, M. HORN und J. ZEHETNER, 2017. Recursive FIR-Filter design for fault-tolerant real-time co-simulation. In: *25th Mediterranean Conference on Control and Automation (MED):* IEEE, S. 461-466. ISBN 1509045333.

STETTINGER, G., M. HORN, M. BENEDIKT und J. ZEHETNER, 2014b. A model-based approach for prediction-based interconnection of dynamic systems. In: *53rd IEEE Conference on Decision and Control:* IEEE, S. 3286-3291. ISBN 1467360902.

STETTINGER, G., M. HORN, M. BENEDIKT und J. ZEHETNER. Model-based coupling approach for non-iterative real-time co-simulation. In: , S. 2084-2089.

SUKSOMBOON, K., N. MATSUMOTO, S. OKAMOTO, M. HAYASHI und Y. JI, 2017. Erlang-k-based packet latency prediction model for optimal configuration of software routers. In: *2017 IEEE Conference on Network Softwarization (NetSoft):* IEEE, S. 1-9. ISBN 1509060081.

SZTRIK, J., 2016. *Modeling and Analysis of Information Technology Systems.* Saarbrücken: GlobeEdit. ISBN 9783639734409.

TANDON, A., M.J. BRUDNAK, J.L. STEIN und T. ERSAL, 2013. An observer based framework to improve fidelity in internet-distributed hardware-in-the-loop simulations. In: *ASME 2013 Dynamic Systems and Control Conference:* American Society of Mechanical Engineers Digital Collection.

TEJADO, I., J. SERRANO, E. PÉREZ, D. TORRES und B.M. VINAGRE, 2016. Low-cost hardware-in-the-loop testbed of a mobile robot to support learning in automatic control and robotics. *IFAC-PapersOnLine*, **49**(6), 242-247. ISSN 2405-8963.

TERHELL, D., 2014. Windows and Real-Time. *The NT Insider*, **3**(20), 10-20. The NT Insider.

THE MODELICA ASSOCIATION, 2019. Distributed Co-Simulation Protocol (DCP), (1.0.0).

THIERRY S. NOUIDUI AND MICHAEL WETTER, 2018. Simulator to FMU: A python utility to support building simulation tool interoperability. In: *Proceedings of the 2018 Building Performance Analysis Conference and SimBuild, Chicago, IL.*

THUMMERER, T., J. KIRCHER und L. MIKELSONS, 2021. NeuralFMU: towards structural integration of FMUs into neural networks. *arXiv preprint arXiv:2109.04351.* arXiv preprint arXiv:2109.04351.

TIBBA, G., C. MALZ, C. STOERMER, N. NAGARAJAN, L. ZHANG und S. CHAKRABORTY, 2016. Testing Automotive Embedded Systems Under X-in-the-loop Setups. In: *Proceedings of the 35th International Conference on Computer-Aided Design.* New York, NY, USA: ACM, 35:1--35:8. ISBN 978-1-4503-4466-1.

TRANNINGER, M., G. STETTINGER, M. BENEDIKT und M. HORN, 2018. Diagnosis of interconnected systems via well tuned model-based coupling algorithms. In: *IFAC-PapersOnLine*.

TRANNINGER, M., T. HAID, G. STETTINGER, M. BENEDIKT und M. HORN, 2016. Fault-tolerant coupling of real-time systems: A case study. In: *3rd Conference on Control and Fault-Tolerant Systems (SysTol):* IEEE, S. 756-762. ISBN 1509006583.

TRUONG, C., L. OUDRE und N. VAYATIS, 2020. Selective review of offline change point detection methods. *Signal Processing*, **167**, 107299. Signal Processing.

TSAY, R.S. und G.C. TIAO, 1984. Consistent estimates of autoregressive parameters and extended sample autocorrelation function for stationary and nonstationary ARMA models. *Journal of the American Statistical Association*, **79**(385), 84-96. ISSN 0162-1459.

TURLEA, A., 2019. Model-in-the-Loop Testing for Cyber Physical Systems. *ACM SIGSOFT Software Engineering Notes*, **44**(1), 37. ISSN 0163-5948.

VAN TENDELOO, Y. und H. VANGHELUWE, 2017. An introduction to classic devs. *arXiv preprint arXiv:1701.07697.* arXiv preprint arXiv:1701.07697.

VANGHELUWE, H.L.M., 2000. DEVS as a common denominator for multi-formalism hybrid systems modelling. In: *Cacsd. conference proceedings. IEEE international symposium on computer-aided control system design:* IEEE, S. 129-134. ISBN 0780365666.

VDI/VDE 2206, 2021. Entwicklung Mechatronischer und Cyber-Physischer Systeme.

VEKIĆ, M.S., S.U. GRABIĆ, D.P. MAJSTOROVIĆ, I.L. ČELANOVIĆ, N.L. ČELANOVIĆ und V.A. KATIĆ, 2012. Ultralow latency HIL platform for rapid development of complex power electronics systems. *IEEE Transactions on Power Electronics*, **27**(11), 4436-4444. ISSN 0885-8993.

WACHENFELD, W. und H. WINNER, 2015. Die freigabe des autonomen fahrens. In: *Autonomes Fahren:* Springer, S. 439-464.

WEI, T., C. WANG, Y. RUI und C.W. CHEN, 2016. Network morphism. In: *International Conference on Machine Learning*, S. 564-572.

WENXU, N.I., K. SONG und T. ZHANG, 2017. *Analysis of geographically distributed vehicle powertrain system validation platform based on X-in-the-Loop theory.* SAE Technical Paper.

WISSEL, D. von, Y. JORDAN, D. VON, W. RENAULT, A. DOLHA und J. MAUSS, 2018. Full Virtualization of Renault's Engine Management Software and Application to System Development. *arXiv preprint arXiv:1802.06841.* arXiv preprint arXiv:1802.06841.

WÜNSCH, G., 2008. *Methoden für die virtuelle Inbetriebnahme automatisierter Produktionssysteme:* Utz. ISBN 3831607958.

WYMORE, A.W., 2018. *Model-based systems engineering:* CRC press. ISBN 1351431080.

XIA, F., P. GRIEFNOW, S. KLEIN, R. THARMAKULASINGAM, A. BALAZS, M. THEWES und J. ANDERT, 2017. Crank Angle Resolved Real-Time Engine Modeling for HiL Based Component Testing. In: *19th APAC 2017 SAE China Congress & Exhibition Shanghai,* S. 1663-1670.

YANG, S.H., X. CHEN, L.S. TAN und L. YANG, 2005. Time delay and data loss compensation for Internet-based process control systems. *Transactions of the Institute of Measurement and Control*, **27**(2), 103-118. ISSN 0142-3312.

YOKOKOHJI, Y., T. IMAIDA und T. YOSHIKAWA, 1999. Bilateral teleoperation under time-varying communication delay. In: *Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients (Cat. No. 99CH36289):* IEEE, S. 1854-1859. ISBN 0780351843.

ZEHETNER, J., G. STETTINGER, H. KOKAL und B. TOYE, 2014. Echtzeit-Co-Simulation für die Regelung eines Motorprüfstands. *ATZ-Automobiltechnische Zeitschrift*, **116**(2), 40-45. ISSN 0001-2785.

ZHANG, Y., S. LU, Y. YANG und Q. GUO, 2018. Internet-distributed vehicle-in-the-loop simulation for HEVs. *IEEE Transactions on Vehicular Technology*, **67**(5), 3729-3739. ISSN 0018-9545.

ZHENG, Y., M.J. BRUDNAK, P. JAYAKUMAR, J.L. STEIN und T. ERSAL, 2018. A Predictor-Based Framework for Delay Compensation in Networked Closed-Loop Systems. *IEEE/ASME Transactions on Mechatronics*, **23**(5), 2482-2493. ISSN 1083-4435.

ZIMMERMANN, W. und R. SCHMIDGALL, 2006. *Bussysteme in der Fahrzeugtechnik:* Springer. ISBN 3834891886.

# Publikationen des Autors

BAUMANN, P., O. KOTTE, L. MIKELSONS und D. SCHRAMM, 2024. Enhancing the Coupling of Real-Virtual Prototypes: A Method for Latency Compensation. *Electronics*, **13**(6). Verfügbar unter: doi:10.3390/electronics13061077

BAUMANN, P., L. MIKELSONS, O. KOTTE und D. SCHRAMM, 2019a. Analyzing the coupling process of distributed mixed real-virtual prototypes. In: P. FUSTER-PARRA, Ó.V. SIERRA und P. GERIL, Hg. *Modelling and simulation 2019: the European Simulation and Modelling Conference 2019 / ESM '19*, S. 251-258. ISBN 978-94-92859-09-9.

BAUMANN, P., M. KRAMMER, M. DRIUSSI, L. MIKELSONS, J. ZEHETNER, W. MAIR und D. SCHRAMM, 2019b. Using the Distributed Co-Simulation Protocol for a Mixed Real-Virtual Prototype. In: *Proceedings of 2019 IEEE International Conference on Mechatronics.* Ilmenau, Germany: IEEE Industrial Electronics Society.

BAUMANN, P., R. SAMLAUS, L. MIKELSONS, T. KUHN und J. JAHIC, 2019c. Towards virtual validation of distributed functions. In: *Proceedings of the 2019 Summer Simulation Conference:* Society for Computer Simulation International, S. 1-12.

BAUMANN, P., R. SAMLAUS, L. MIKELSONS, T. KUHN, J. JAHIC und D. SCHRAMM, 2019d. Ein Beitrag zur Simulation vernetzter virtueller ECUs. In: *Fachtagung Mechatronik,* S. 13-18.

**DOI:** 10.17185/duepublico/82233
**URN:** urn:nbn:de:hbz:465-20240809-145847-8