Beiträge zum Stuttgarter Maschinenbau

Caren Dripke

Verteilte Interpolation: Bewegungssynchronisierung in dezentral gesteuerten Mehrachssystemen

stuttgarter
maschinenbau
interdisziplinär und vielfältig

Universität Stuttgart
Institut für Steuerungstechnik
der Werkzeugmaschinen und
Fertigungseinrichtungen (ISW)

Universität Stuttgart

Beiträge zum Stuttgarter Maschinenbau **Band 4**

Herausgeber: Prof. Dr.-Ing. Oliver Riedel
Prof. Dr.-Ing. Alexander Verl
Jun.-Prof. Dr. rer. nat. Andreas Wortmann

Caren Dripke

Verteilte Interpolation: Bewegungssynchronisierung in dezentral gesteuerten Mehrachssystemen

Fraunhofer Verlag

Kontaktadresse:
Institut für Steuerungstechnik der Werkzeugmaschinen
und Fertigungseinrichtungen ISW
Seidenstr. 36
70174 Stuttgart
info@isw.uni-stuttgart.de
https://www.isw.uni-stuttgart.de

Titelbild: Caren Dripke

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.de abrufbar.

ISSN: 2750-655X
ISBN: 978-3-8396-1810-3

D 93

Zugl.: Stuttgart, Univ., Diss., 2021

Druck und Weiterverarbeitung:
Fraunhofer Verlag, Mediendienstleistungen

Für den Druck des Buches wurde chlor- und säurefreies Papier verwendet.

Nobelstraße 12
70569 Stuttgart
verlag@fraunhofer.de
www.verlag.fraunhofer.de

als rechtlich nicht selbständige Einheit der

Fraunhofer-Gesellschaft zur Förderung
der angewandten Forschung e.V.
Hansastraße 27 c
80686 München
www.fraunhofer.de

Geleitwort

Die deutsche Wirtschaft ist weltweit bekannt für ihren Anlagen- und Maschinenbau. Dabei ist die Universität Stuttgart mit ihren beiden Maschinenbaufakultäten – unter deren Dach sich 42 Institute befinden – die größte universitäre Einrichtung für den Maschinenbau in Deutschland. Unsere wissenschaftliche Exzellenz stützt sich dabei auf unsere zahlreichen Promovierenden und ihre hervorragenden Dissertationen. Viele dieser Dissertationen entstehen in lokaler, nationaler und internationaler Zusammenarbeit mit renommierten Universitäten und außeruniversitären Einrichtungen wie dem Deutschen Zentrum für Luft- und Raumfahrt, der Fraunhofer-Gesellschaft und der Max-Planck-Gesellschaft. Dabei reicht das inhaltliche Spektrum der Dissertationen von Biotechnik, Energietechnik, Fahrzeugtechnik, Kybernetik und Systemtechnik, Produktentwicklung und Konstruktionstechnik, Produktionstechnik bis hin zur Verfahrenstechnik und stützt sich auf die sechs Forschungsschwerpunkte Advanced Systems Engineering, Autonome Produktion, Software-Defined Manufacturing, Resiliente Versorgung, Biointelligenz und Dekarbonisierung der Industrie. Die Ergebnisse aus den Dissertationen zielen darauf ab, kunden-, produkt-, prozess- und mitarbeiterorientierte Technologie zielgerichtet und zeitnah zu entwickeln und anzuwenden.
Viele der im Rahmen der Forschungsarbeiten an den Instituten entstandenen Dissertationen werden in diesen »Beiträgen zum Stuttgarter Maschinenbau« veröffentlicht. Die beiden Fakultäten des Stuttgarter Maschinenbaus wünschen den Promovierenden, dass ihre Dissertationen aus dem Bereich des Maschinenbaus in der breiten Fachwelt als maßgebliche Beiträge wahrgenommen werden und so den Wissensstand auf ein neues Niveau heben.

Für den Stuttgarter Maschinenbau

Stefan Weihe
Prodekan Fakultät 4

Oliver Riedel
Prodekan Fakultät 7

Vorwort der Herausgeber

Innerhalb der Reihe »Beiträge zum Stuttgarter Maschinenbau« berichtet das Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen an der Universität Stuttgart (ISW) über seine Forschungsergebnisse. Das Institut beschäftigt sich in vielfältiger Form mit Steuerungs- und Automatisierungstechnik sowie dem Einsatz von modernen Methoden des Informationsmanagements. Dabei stehen Grundlagenforschung und anwendungsorientierte Entwicklung in einem stetigen Austausch, wodurch ein kontinuierlicher Technologietransfer in die Praxis sichergestellt wird.

Die am ISW entstandenen Dissertationen werden damit unter erweitertem Namen und inzwischen in vierter Generation in der bewährten Konzeption, die der Gründer des ISW Prof. Stute und sein Nachfolger Prof. Pritschow 1972 begonnen haben, durch die heutige Institutsleitung fortgesetzt.

Frau Caren Dripke M.Sc. möchten wir für die geleistete Arbeit danken, dem Verlag für die Aufnahme dieser Schriftenreihe in sein Angebot und der Druckerei für die saubere und zügige Ausführung. Möge das Buch von der Fachwelt gut aufgenommen werden.

Alexander Verl

Oliver Riedel

Verteilte Interpolation:
Bewegungssynchronisierung in dezentral gesteuerten Mehrachssystemen

Von der Fakultät Konstruktions-, Produktions- und Fahrzeugtechnik
der Universität Stuttgart
zur Erlangung der Würde einer
Doktor-Ingenieurin (Dr.-Ing.)
genehmigte Abhandlung

Vorgelegt von

Caren Dripke, M.Sc.
aus Leonberg

Hauptberichter:	Prof. Dr.-Ing. Dr. h.c. mult. Alexander Verl
Mitberichter:	Prof. Dr.-Ing. Jörg Krüger
Tag der mündlichen Prüfung:	29.11.2021

Institut für Steuerungstechnik der Werkzeugmaschinen
und Fertigungseinrichtungen der Universität Stuttgart

2022

Vorwort

Diese Arbeit entstand während meiner Tätigkeit als wissenschaftliche Mitarbeiterin am Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW) der Universität Stuttgart.

Herzlich bedanken möchte ich mich bei meinem Doktorvater Herrn Prof. Dr.-Ing. Dr. h.c. mult. Alexander Verl für die wissenschaftliche Betreuung der Arbeit und für die Übernahme des Hauptberichts, sowie die vertrauensvolle Zusammenarbeit am Institut.

Darüber hinaus gilt mein Dank Herrn Prof. Dr.-Ing. Jörg Krüger, Inhaber des Lehrstuhls Industrielle Automationstechnik am Institut für Werkzeugmaschinen und Fabrikbetrieb (IWF) an der TU Berlin, für die freundliche Übernahme des Mitberichts. Ebenso danke ich Herrn Prof. Dr.-Ing. Hans-Christian Reuss für die Übernahme des Prüfungsvorsitzes.

Bei Matthias Keinert, Friedemann Groh und Mihai Dragan möchte ich mich für die inhaltliche Durchsicht der Arbeit und die wertvollen fachlichen Anregungen bedanken.

Ich danke allen Kolleginnen und Kollegen am ISW, die mich während meiner Zeit am Institut begleitet haben. Fachliche wie auch nicht-fachliche Diskussionen, Projekte und Unternehmungen haben die Zeit am Institut zu einer wertvollen Erfahrung gemacht, die mich sehr geprägt hat.

Bedanken möchte ich mich auch bei den Studierenden, die zum Gelingen der Arbeit beigetragen haben. Genannt seien hier im Besonderen Sonja Laicher, Daniel Schöbel und Pirmin Stanglmeier.

Vielen Dank an André Kempf, mit dem ich die letzten beiden Jahre bis hin zur Finalisierung der Dissertation im regelmäßigen Austausch stand. Die gemeinsamen Termine brachten den notwendigen Fokus und unterteilten das große Unterfangen Dissertation in handhabbare Zwischenziele. Es freut mich sehr, dass wir beide den Abschluss nun auch fast zeitgleich geschafft haben!

Ein besonderes Anliegen ist es mir, zum Abschluss meiner Oma Martha Weber zu danken. Bei ihr durfte ich, abgeschottet von Ablenkungen, konzentriert an der Erstellung der Schriftfassung arbeiten. Neben einer fantastischen kulinarischen Versorgung war sie mir in allen Zeiten eine motivierende moralische Stütze. Danke für die großartige Zeit!

Caren Dripke

Kurzfassung

Integrierte Miniatursteuerungen sind Technologiebefähiger für Industrie 4.0. Die Verwendung solcher Steuerungen in Automatisierungskomponenten ermöglicht es, diese als intelligente Elemente in rekonfigurierbaren Fertigungsstationen und -zellen einzusetzen und die Zeitaufwände für die Anpassung einer zentralen Steuereinheit bei Rekonfigurationen zu reduzieren, indem die integrierten Miniatursteuerungen als dezentrale Recheneinheiten die Durchführung der Prozesse übernehmen. Solche intelligenten Komponenten bieten ihre Funktionen über das Produktionsnetzwerk an und können, kombiniert mit weiteren Automatisierungskomponenten auch höherwertige Funktionen realisieren.

Ein Spezialfall der dezentralen Ansteuerung mit solchen höherwertigen Funktionen ist die gemeinsame synchronisierte Bewegung von positionierenden Automatisierungskomponenten. In klassischen Automatisierungssystemen wird diese Aufgabe meist von einer zentralen Bewegungssteuerung berechnet, koordiniert und durchgeführt. Bei einer Rekonfiguration fallen dann aber hohe Änderungsaufwände an. Daher wird in dieser Arbeit betrachtet, welche alternativen Systemarchitekturen zur Interaktion zwischen den Miniatursteuerungen geeignet sind, um eine verteilte Bewegungssteuerung umzusetzen. Die agentenbasierte Systemarchitektur ist vielversprechend für den Anwendungsfall der Rekonfiguration und wissenschaftlich interessant für die Ableitung geeigneter Agentenverhalten.

Es folgt die Analyse des Systems als ein Multiagentensystem und eine Ableitung von Interaktionsmustern, die eine Bewegungssynchronisierung zwischen positionierenden Automatisierungskomponenten durchsetzen können. Diese Interaktionsmuster werden auf die Trajektorienplanung jeder positionierenden Automatisierungskomponente übertragen und ein Agentenverhalten sowie ein geeignetes Kommunikationsprotokoll abgeleitet. Die Validierung des Konzepts wird sowohl simulativ durchgeführt, als auch in einem Versuchsstand mit kommerziellen Achskomponenten realisiert.

Das Systemverhalten der agentenbasierten Ansteuerung mit Miniatursteuerungen kann für verschiedene Anwendungsfälle eine Systemarchitektur und Interaktionsmuster liefern, um synchronisierte Bewegungen in einem gemeinsamen Arbeitsraum ohne eine zentrale Steuerung umzusetzen. Rekonfigurationen des Systems, also die Integration weiterer, veränderter oder andersartiger Achsen, können mit wenigen Anpassungen im Agentenverhalten der Miniatursteuerung der betroffenen Achse umgesetzt werden.

Abstract

Integrated miniature controllers are technology enablers for Industry 4.0. The integration of such controllers in automation components makes the use in reconfigurable stations and cells possible and saves time adapting a central control unit in case of reconfiguration. Such components offer their elementary functions via the production network and realize higher-value functions in combination with other automation components.

A special case of such higher-value functions is the joint synchronized movement of positioning automation components. In current automation systems this is usually calculated, coordinated and executed by a central motion control. Since a reconfiguration in such a central control system requires significant modification efforts, this thesis examines which alternative control architectures are suitable for the interaction between the miniature controls in order to implement a distributed motion control. The agent-based architecture is identified as promising for the use case of reconfiguration and scientifically interesting in terms of the deduction of agent behaviors.

An analysis of the system as a multi-agent system follows, as well as the derivation of interaction patterns that can enforce motion synchronization between positioning automation components. The interaction patterns are transferred to the trajectory planning of each positioning automation component and an agent behavior and a suitable communication protocol is presented. The validation of the concept was carried out both in simulation and in a demonstrator with commercial axis components.

The system behavior of agent-based control with miniature controllers can provide a system architecture and interaction patterns for specific applications in order to implement synchronized movements in a common workspace without a central controller. In case of a reconfiguration of the system, the integration of additional, modified or different axes can be realized with few adjustments in the agent behavior of the miniature control of the affected axis.

Inhaltsverzeichnis

Formelzeichen

a	Beschleunigung
A	Fläche
c	Skalierungsfaktor
C	Rotationsmatrix
f	Faktor
H	homogene Transformationsmatrix
$\vec{\imath}, \vec{\jmath}, \vec{k}$	Koordinatenrichtungen der Achsen x,y,z
k	Laufparameter
K_a, K_v, K_p	Verstärkungsfaktoren
K	Korrekturterm für die approximierte Zeitdauer
l	Länge
L	Weglänge
$\Delta\langle n\rangle$	Schrittweite in Achskoordinate der Achse $\langle n\rangle$
P	Punkt im Raum
p	Position
$\mathcal{P}_{\langle n\rangle}$	Projektion des Referenzvektors auf die Koordinatenrichtung der Achse $\langle n\rangle$
$\vec{r}$	Referenzvektor
${}^{h}\vec{r}$	homogener Referenzvektor
$r_{\langle n\rangle}$	Koordinaten in den Koordinatenrichtungen
R	Kreisradius
t	Zeit
T	Zeitdauer
T_s	Zeitdauer für Segment S
T^*	approximierte Zeitdauer
T°	Konsens der Zeitdauer
τ	Substitutionsvariable zur Zeitintegration
θ	Kreiswinkel
Δu	Schrittweite entlang der Bahn
$\vec{v}$	Geschwindigkeitsvektor
v	Geschwindigkeit

$v_{\langle n \rangle}$	Geschwindigkeit in Achskoordinaten der Achse $\langle n \rangle$
v^*	approximierte gemittelte Geschwindigkeit
$v^\circ_{\langle n \rangle}$	Konsensgeschwindigkeit der Achse $\langle n \rangle$
v^+	angepasste Geschwindigkeit
ω	Kreiswinkelgeschwindigkeit

Häufig verwendete Indizes:

i	Laufindex der Segmente
n	Laufindex der Achsen
S	Segment
römische Zahlen	Laufindex der Phasen eines Beschleunigungsprofils
Start	erste Sollposition
Ziel	letzte Sollposition
max	Maximum
min	Minimum
ist	aktueller Wert
neu	aktualisierter Wert
Ipo	Interpolationstakt
Rest	Restdauer im Segment

Abkürzungen

CC	Cross-Coupling; Kreuzkopplung
CCD	Cyclic Coordinate Descent; zyklische Koordinatenannäherung
CNC	Computerized Numerical Control; Computerbasierte numerische Steuerung
CPPS	cyberphysisches Produktionssystem
CPS	cyberphysisches System
CPU	Central Processing Unit; Zentrale Recheneinheit
DCS	Distributed Control System; verteiltes Prozessleitsystem
DEVEKOS	**D**urchgängiges **E**ngineering für sichere, **ve**rteilte und **ko**mmunizierende Mehrkomponenten**s**ysteme (Projektakronym; Förderprojekt des Bundesministeriums für Wirtschaft und Energie)
DP	Dynamic Programming; dynamische Programmierung
DPS	Distributed Production System; verteiltes Produktionssystem
FIFO	First-In-First-Out
FPGA	Field-Programmable-Gate-Array; programmierbare digitale Bausteine
HLC	High-Level-Controller; übergeordnete Steuerung
HMS	Holonic Manufacturing System; holonisches Produktionssystem
HW	Hardware
IAS	Integrated Assembly Solutions; VDMA Fachabteilung
KGT	Kugelgewindetrieb
LDA	Lineardirektantrieb
LLC	Low-Level-Controller; untergeordnete Steuerung
MAS	Multi-Agenten-System
MC	Motion Control; Bewegungssteuerung
NC	Numerical Control; Numerische Steuerung
OSI	Open-Systems-Interconnection-Modell; Referenzmodell für Netzwerkprotokolle
PEAS	Performance Measure, Environment, Actuators, Sensors; Beschreibungsmodell für Multi-Agenten-Systeme
PTP	Point-to-Point; Punkt-zu-Punkt
QoP	Quality-of-Performance; Leistungsqualität
QoS	Quality-of-Service; Servicequalität
SOA	Service-Oriented Architecture; dienste-orientierte Architektur

SPS	speicherprogrammierbare Steuerung
SW	Software
UDP	User Datagram Protocol
ZRA	Zahnstangenritzelantrieb

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Seit der Jahrtausendwende ist die Produktionstechnik vom Trend der Massenproduktion und langen Produktlaufzeiten hin zu personalisierter, regionalisierter Produktion mit kleinen Chargen und kürzeren Produktlaufzeiten (Koren 2010, S. 2) geprägt. Um die häufige Veränderung bei den anzufertigenden Produkten umsetzen zu können, werden Produktionsabläufe statt in starren Fertigungsketten mit wandelbaren, rekonfigurierbaren Produktionszellen aufgebaut (Koren et al. 1999). Solche rekonfigurierbaren Produktionssysteme sollen neue Produkte, Technologien oder Funktionen innerhalb kürzester Zeit in bestehende Produktionsabläufe integrieren können, vgl. (Monostori et al. 2016, S. 625). Der Schritt zu rekonfigurierbaren Systemen ist notwendig, um die hohen Investitionskosten aufgrund der traditionell langen Lebenszyklen von Produktionsmitteln auszugleichen (Kircher 2011, S. 15; Hees 2017, S. 2). Zum Konzept der wandlungsfähigen, flexiblen oder rekonfigurierbaren Produktionstechnik wird umfassende Forschung betrieben, zu nennen sind hier z. B. (Krüger 2007; Pachow-Frauenhofer 2012; Meling 2013; Hees 2017; Stehle et al. 2017).

In rekonfigurierbaren Produktionssystemen werden System- und Maschinenstruktur nach Bedarf neu konfiguriert. Dies bedeutet, dass Mechanik, Elektrik und Steuerungssoftware sowie das Produktionskonzept anpassbar gestaltet werden müssen (Pantförder et al. 2014, S. 151; Hees 2017, S. 1). Für Produktionszellen oder -stationen bedeutet dies, dass Teilkomponenten in Ausprägung, Kapazität oder Rolle im Produktionsprozess verändert werden. Um die Einsatzdauer solcher Komponenten auch bei kurzen Produktlebenszyklen zu maximieren, werden Komponenten häufig wiederverwendet. Mittelfristig entstehen so Produktionssysteme mit der Beteiligung unterschiedlichster Technologien, Hersteller, und internen Kommunikationsmechanismen (Meling 2013, S. 31). Die Kompatibilität zwischen diesen Systemen ist die Grundvoraussetzung um in einem rekonfigurierten Produktionssystem die Produktionsprozesse durchzusetzen (Leitão 2009, S. 988).

Befähiger für rekonfigurierbare Produktionssysteme und wiederverwendbare Komponenten sind cyberphysische Produktionssysteme (CPPS), also intelligente Produktionsmittel, die miteinander in Austausch stehen und dadurch auf interne sowie externe Veränderungen reagieren können (Monostori et al. 2016, S. 623ff). Die horizontale und vertikale Vernetzung zwischen CPPS löst die bisher starren Kommunikationsstrukturen in Produktionssystemen auf, vgl. (Pantförder et al. 2014, S. 145).

Ermöglicht wird die Vernetzung und Intelligenz von Produktionsmitteln durch die Miniaturisierung von Elektronik in eingebetteten Systemen. So können mechanische und elektrische Systeme mit Computersystemen ausgestattet werden, die eine direkte Ansteuerung in Echtzeit erlauben (Yoong et al. 2014, S. 5; Monostori et al. 2016, S. 623). Im Forschungsprojekt DEVEKOS (Durchgängiges Engineering für sichere, verteilte und kommunizierende Mehrkomponentensysteme) des Bundesministeriums für Wirtschaft und Energie werden Mehrkomponentensysteme untersucht, die aus intelligenten Automatisierungskomponenten bestehen. Mit dem Fokus der Rekonfigurierbarkeit wird ein Systemkonzept erarbeitet, in

dem die intelligenten Komponenten, ausgestattet mit individuellen Miniatursteuerungen, ohne zentrale Steuerungsinstanzen miteinander agieren. Zum aktuellen Zeitpunkt haben Entwicklungen nach diesem Konzept Prototypenstatus. Doch standardisierte Schnittstellen für solche Systeme werden bereits im VDMA, Fachabteilung Integrated Assembly Solutions (IAS), vorangetrieben (Axmann 2019; Zimmermann et al. 2019). Das Systemverständnis einer DEVEKOS-Komponente wird nachfolgend im Detail vorgestellt.

1.1 Systemverständnis der DEVEKOS-Komponente

Das Systemverständnis der DEVEKOS Komponente basiert auf der Annahme, dass Automatisierungsprozesse durch eine Kombination von elementaren Funktionen wie z. B. dem Greifen, Bewegen oder Positionieren aufgebaut werden können, s. (Helbig 2017). Diese Elementarfunktionen werden von Automatisierungskomponenten übernommen und ausgeführt. Wenn ein Prozessablauf über elementare Grundfunktionen geplant wird, kann die Steuerung der Komponenten direkt aus dem geplanten Prozessablauf und den Elementarfunktionen abgeleitet werden und die Auswahl der eingesetzten Komponenten erst danach finalisiert werden. Die Interaktion kann auf der Ebene des Prozessablaufs dann umgesetzt werden, indem die Komponenten sich über ihre elementaren Funktionen identifizieren und diese selbstständig ausführen.

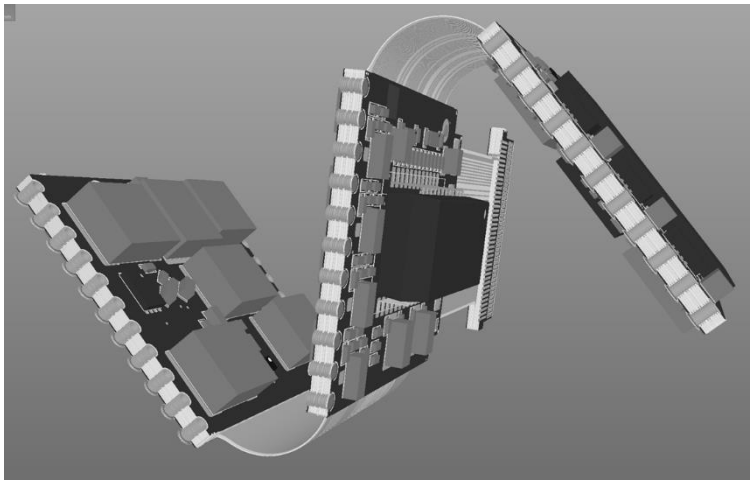

Abbildung 1-1: Schemazeichnung einer Miniatursteuerung „EMC 201 Effectuator“ (Meister 2018).

Im Projekt DEVEKOS werden aktuell Prototypen der Miniatursteuerungen als gefaltete Platinen entwickelt, die Ausmaße von 2,5 Kubikzentimetern nicht überschreiten, s. Abbildung 1-1. Diese Miniatursteuerung bietet neben Anschlüssen für die Kommunikation im Produktionsnetzwerk eine integrierte Endstufe für die Ansteuerung von positionierenden elektrischen Achsen sowie ein frei programmierbares FPGA (Field-Programmable-Gate-Array) und CPU (Central Processing Unit). Die kleine Bauform ermöglicht die Integration direkt in der Automatisierungskomponente und verringert dadurch Verkabelungs- und Konfigurationsaufwand bei der Verwendung in einem Produktionssystem, s. Abbildung 1-2.

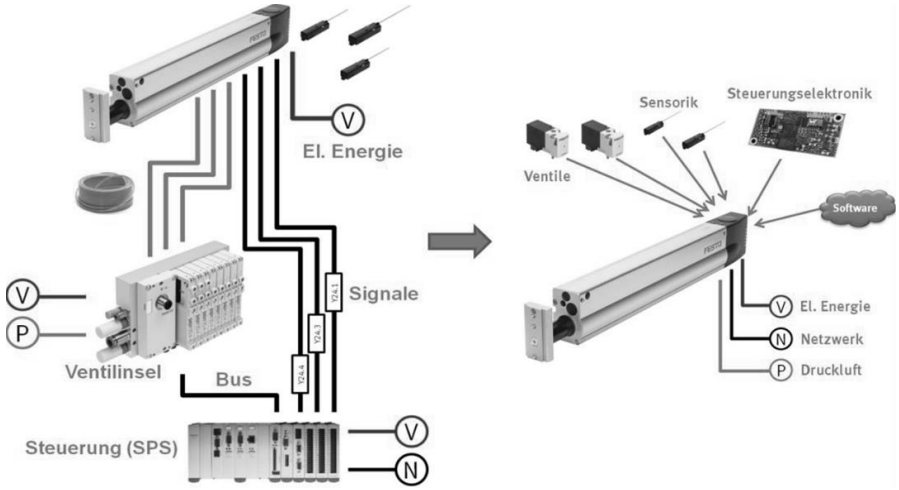

Abbildung 1-2: Koventionelle und integrierte Automatisierungskomponente (links: konventionelle Verkabelung; rechts: integrierte Miniatursteuerung) am Beispiel eines Pneumatikzylinders (Henning et al. 2015, S. 4).

Elementarfunktionen einzelner Komponenten können zu höherwertigen Funktionen kombiniert und angeboten werden, s. Abbildung 1-3. Mehrere Achsen können z. B. mit einem Greifer zu einem Handhabungsmodul kombiniert werden. Die Elementarfunktionen der Achsen und Greifer können zu den höherwertigen Funktionen wie dem Aufnehmen, Ablegen oder Bewegen eines Werkstücks kombiniert werden. Diese höherwertigen Funktionen kann wiederum ein Handhabungsmodul in Kombination mit weiteren Funktionen anderen Komponenten und Komponentenverbünde innerhalb einer Montagestation so kombinieren, dass ein Fügeprozess umgesetzt werden kann. Der Zusammenschluss mehrerer Stationen findet wiederum eine Ebene höher in einer Fertigungszelle statt. Die Interaktion über solche Elementar- und/oder zusammengesetzten höherwertigen Funktionen benötigt abstrakte Schnittstellen für Komponenten, die interne Abläufe und Kommunikation zur Ausführung der Elementaroptionen kapseln, s. (Yoong et al. 2014, S. 2).

Wenn für die Definition der Funktionen klare herstellerunabhängige und standardisierte semantische Vorgaben und Aufrufe existieren, kann in einfacher Art und Weise und mit geringer Komplexität für den Anwender ein Prozessablauf mit intelligenten, vernetzten Komponenten definiert werden (Lenkenhoff 2018, S. 61) und damit die Komplexität der Ansteuerung der Funktionen in den Komponenten aus Anwendersicht gekapselt bleiben.

Die Verwendung von Miniatursteuerungen direkt in den Automatisierungskomponenten und die Definition von Prozessabläufen über Elementarfunktionen ermöglichen es, auf zusätzliche zentrale Einheiten, deren einzige Aufgabe die Durchführung und Überwachung des Prozessablaufs ist, zu verzichten. Stattdessen sollen Komponenten durch Interaktion miteinander diese Durchführung selbstständig vorbereiten, anstoßen, überwachen und ausführen.

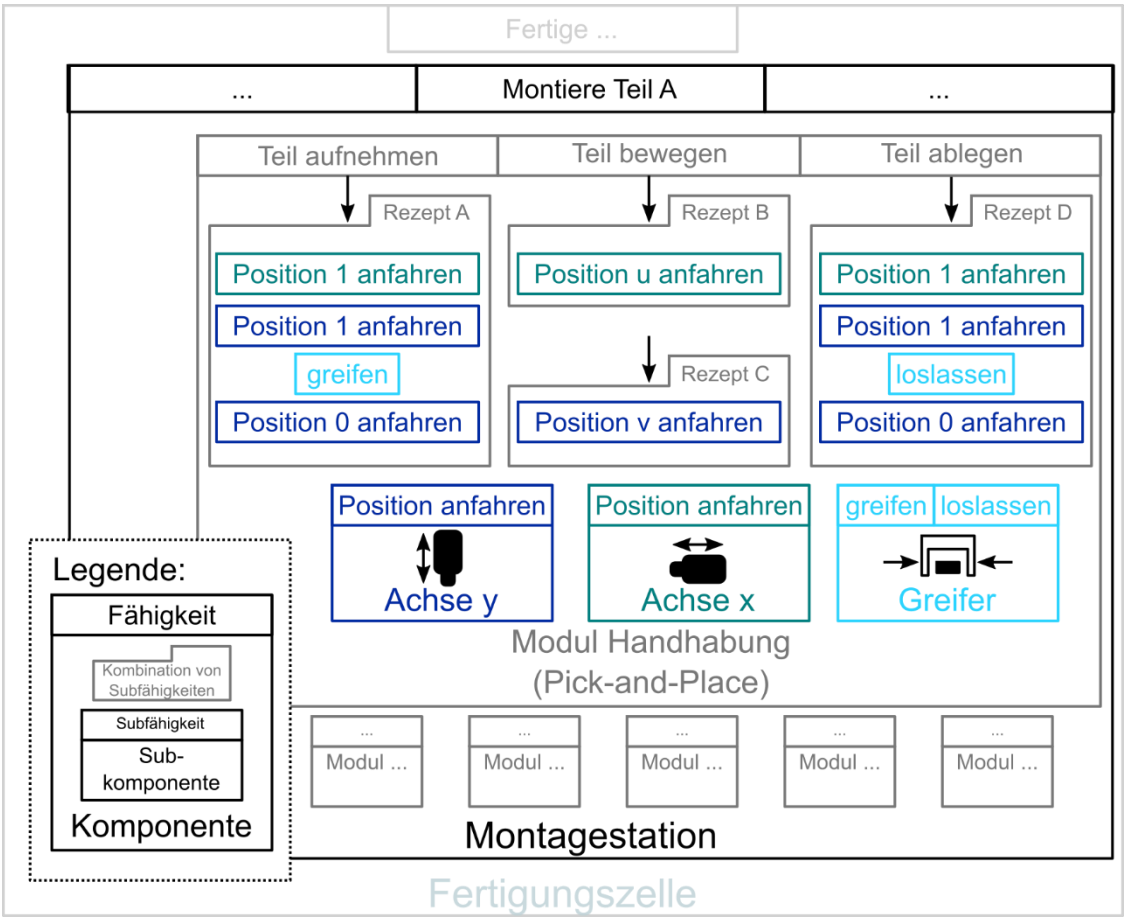

Abbildung 1-3: Kombination von Elementarfunktionen zu höherwertigen Funktionen.

Das DEVEKOS-Systemverständnis folgt damit der Definition des Holonic Manufacturing Systems (HMS; holonisches Produktionssystem). Dort wird der Begriff des Holons verwendet um das Zusammenspiel von Produktionseinheiten auf verschiedenen hierarchischen Ebenen zu beschreiben. Ein Holon kann nach (Lüth 1998, S. 29–32; Leitão 2009, S. 984) als eine Einheit mit Echtzeitanforderungen und einer Verbindung zu physischen Einheiten verstanden werden. Ein HMS entsteht durch die Hierarchie verschiedener zusammenarbeitender Einheiten.

Einen Holon zeichnet aus, dass er autonom auf Umweltbedingungen entsprechend seiner Verhaltensregeln reagiert. Ebenfalls kann er in Kooperation mit weiteren Holonen höherwertige Holone formen (Lüth 1998, S. 31; Leitão 2009, S. 983), was einer höherwertigen, zusammengesetzten Fähigkeit entspricht. Die Zusammensetzung von höherwertigen Fähigkeiten aus atomaren Komponentenfähigkeiten, s. Abbildung 1-3, bildet dieses Prinzip ab. Die Komponenten Achse und Greifer sind nach dieser Definition atomare Holone, das Handhabungsmodul, die Montagestation oder auch die Fertigungszelle sind zusammengesetzte Holone, deren höherwertige Fähigkeiten durch Elementarfunktionen der atomaren Holone zusammengesetzt werden.

1.2 Problemstellung und Handlungsbedarf

Vor allem in Bezug auf positionierende Achsen ist der Verzicht auf eine zentrale Steuerungseinheit ein Paradigmenwechsel. Ansätze bezüglich der Integration von Antriebstechnik direkt an den Motoren sind als dezentrale Antriebsverstärker seit einigen Jahren bereits am Markt verfügbar. Der Trend zur Dezentralisierung ermöglicht die Verringerung der Anzahl der benötigten und zu verkabelnden Schaltschrankkomponenten für die Achsansteuerung. Die Leistungseinspeisung bleibt allerdings bei aktuellen Ansätzen an einer zentralen Stelle verankert, ebenso wie auch die Bewegungssteuerung als Steuereinheit (Beckhoff 2017; SEW-Eurodrive 2018). Gleichzeitig werden von anderen Anbietern sehr kompakte Bewegungssteuerungen auf den Markt gebracht (Struck 2017, S. 50). Komponenten und Ausstattung für eine dezentrale Ansteuerung direkt an der Automatisierungskomponente sind also prinzipiell auf dem Markt vorhanden. Bisherige Systemarchitekturen für rekonfigurierbare Produktionstechnik wie OSACA, Hümnos, oder Plug&Produce, vgl. (Meling 2013, S. 49–51), betrachten jedoch lediglich die Möglichkeit, dass an einer zentralen Steuerung durch eine Rekonfiguration andere Komponenten als zu Beginn angeschlossen werden. Die Betrachtung eines Verzichts auf die zentrale Steuerung und die Verwendung der Rechenkapazitäten von dezentral vorhandenen Miniatursteuerungen erfolgt nicht, obwohl eine große Vereinfachung bei Rekonfigurationen erreicht werden könnte, wenn keine zentrale Datenhaltung angepasst werden muss. Dies könnte erreicht werden, indem die Datenhaltung und Steuerabläufe direkt auf den dezentralen Komponenten stattfinden.

Speziell der Bereich der Montage hat an der Produktion einen hohen Kostenanteil von bis zu 70% und muss bei kürzeren Produktzyklen rekonfigurierbar geplant werden (Pachow-Frauenhofer 2012, S. 6). Positionierende Bewegungen sind ein Grundprinzip der Montage, speziell die Anwendung Pick-and-Place (Aufnehmen und Platzieren von Teilen) ist ein klassischer Prozessschritt. Hierbei interagieren mehrere positionierende Achsen in einem Mehrkomponentensystem (Mehrachssystem) miteinander. Neben der freien Positionierung ohne Vorgabe einer Bahn im Raum, ist in vielen Montageanwendungsfällen zur Verhinderung von Kollisionen oder für spezielle Fügebewegungen die Bewegung des Produkts auf einer definierten Bahnvorgabe umzusetzen. Dies bezeichnet man allgemein als synchrone Bewegungssteuerung der beteiligten Achsen und ist eine Aufgabe, die bisher von einer zentralen Steuerung koordiniert wird.

Wenn innerhalb des DEVEKOS-Systemverständnis auf zusätzliche Komponenten wie die zentrale Steuerung verzichtet wird und Komponenten höherwertige Funktionen in Mehrkomponentensystemen durch Interaktion aus Elementarfunktionen durchführen sollen, ergibt sich für die Betrachtung von positionierenden Komponenten (Achsen) im System folgender Handlungsbedarf:

Handlungsbedarf:

Positionierende Komponenten mit Miniatursteuerung sollen im DEVEKOS-Systemverständnis ohne eine zentrale Steuerung synchronisierte Bewegungen umsetzen können.

1.3 Anforderungen und Zielsetzung

Aus dem Handlungsbedarf folgt die Zielsetzung dieser Arbeit:

> Gesamtziel der Arbeit:
>
> Entwicklung einer Bewegungssynchronisierung für dezentral gesteuerte Mehrachssysteme

An diese Bewegungssynchronisierung sollen ausgehend von der dargestellten Notwendigkeit für die Konzeption von rekonfigurierbaren Produktionsmitteln und innerhalb des DEVEKOS-Systemverständnisses Anforderungen gestellt werden, um die Bewertung der Zielerreichung zu ermöglichen.

> Vorgehen:
>
> Definition der Anforderungen an das Gesamtziel der Arbeit

Anforderungen an eine Bewegungssynchronisierung für dezentral gesteuerte Mehrachssysteme:

- folgend aus dem Gesamtziel der Arbeit:
  a) Umsetzung einer Bewegungssynchronisierung
- folgend aus dem DEVEKOS-Systemverständnis:
  b) Einklang mit dem DEVEKOS-Systemverständnis
  c) Gleichwertige Verteilung der Rechenleistung
- folgend aus der Motivation rekonfigurierbarer Produktionssysteme:
  d) Angemessene Aufwände bei Rekonfigurationen
- folgend aus den vorgestellten Miniatursteuerungen:
  e) Angemessene Realisierungskomplexität der Lösung

Die hier vorgestellten Anforderungen werden in Kapitel 3.2 detailliert aufgeschlüsselt und die Kriterien zur Zielerreichung vorgestellt.

Für das weitere Vorgehen werden zur Erreichung des Gesamtziels Zwischenziele definiert:

Die Definition einer geeigneten Systemarchitektur ist der Ausgangspunkt für die weitere Erarbeitung einer Bewegungssynchronisierung, die vor allem auf die flexiblen Verwendungsmöglichkeiten der Miniatursteuerungen eingeht.

> Zwischenziel 1:
>
> Eine geeignete Systemarchitektur ist ausgewählt, um die Bewegungssynchronisierung auf den vorhandenen Miniatursteuerungen zu integrieren.

Innerhalb der Systemarchitektur soll dann ein Mechanismus konzipiert werden, der die Synchronisierung der Bewegungen ermöglicht.

> Zwischenziel 2:
>
> Ein verteilter Synchronisierungsmechanismus ist konzipiert.

Der konzipierte Synchronisierungsmechanismus wird nachfolgend validiert. Um die Rekonfiguration und Erweiterbarkeit des Systems sicherzustellen, wird

anschließend die Schnittstellendefinition festgelegt, nach der weitere Komponenten in das System integriert werden können.

> Zwischenziel 3:
>
> Der verteilte Synchronisierungsmechanismus ist validiert.
>
> Die Schnittstellen zur Erweiterbarkeit und Rekonfiguration sind definiert.

Nach der obigen Definition von Anforderungen und Zwischenzielen werden im folgenden Schritt I der Stand der Technik und Forschung dargestellt und die Grundlagen der Bewegungssynchronisierung erläutert, aus der Systemkonzepte für das Zwischenziel 1 abgeleitet wurden. Das Vorgehen für die weiteren Zwischenziele, s. Abbildung 1-4, wird nach der Entwurfsentscheidung für eine Systemarchitektur in Kapitel 3.4 dargelegt. In Schritt II wird der Synchronisierungsmechanismus vorgestellt. Es folgen im Schritt III die Umsetzung und Anwendungsfallbetrachtung, auf deren Basis die Schnittstellendefinition aufgearbeitet wird. Die notwendigen Grundlagen für die einzelnen Schritte werden jeweils zu Beginn der Themenkapitel vorgestellt.

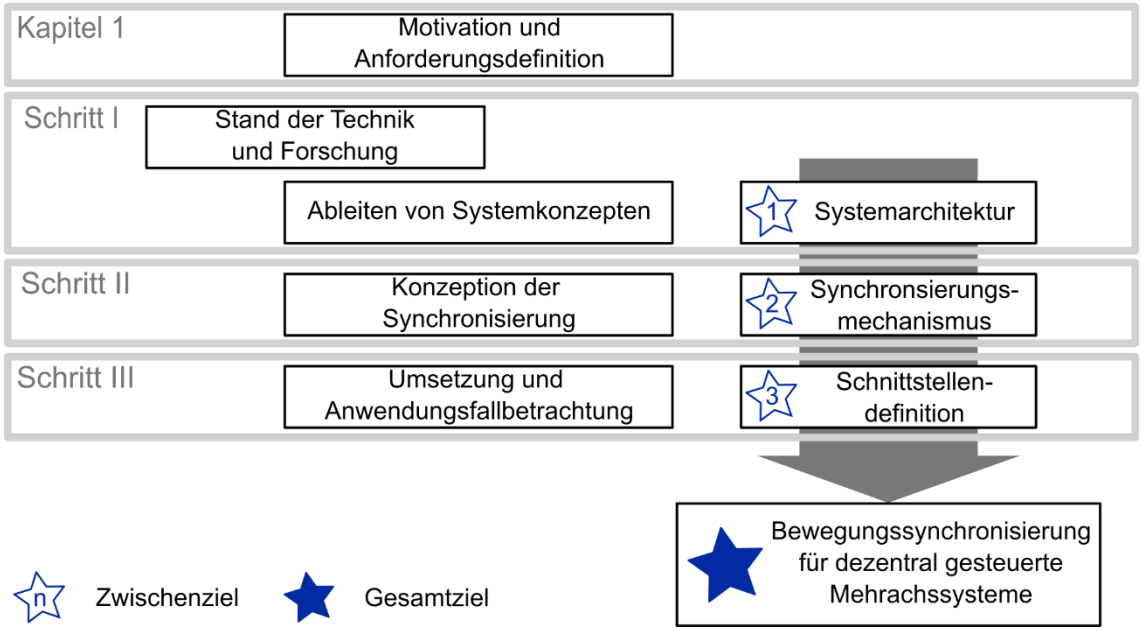

Abbildung 1-4: Zwischenzieldefinition und weiteres Vorgehen.

2 Stand der Technik und der Forschung

Im nachfolgenden Kapitel wird der Stand der Technik und Forschung von positionsgesteuerten Achsen und deren Bewegungssynchronisierung dargestellt. Es werden die Grundlagen positionsgesteuerter Achsen erläutert, Hintergründe zur Bewegungssynchronisierung vermittelt und etablierte Methoden und Systeme zur Bewegungssynchronisierung sowie Forschungsarbeiten zur Bewegungssynchronisierung erläutert.

Vorgehen:

Erarbeitung des Stands der Technik und der Forschung

2.1 Komponenten einer positionsgesteuerten Achse

Diese Arbeit beschäftigt sich mit der Bewegungssynchronisierung in dezentral gesteuerten Mehrachssystemen. Positionsgesteuerte Achsen werden in der Automatisierungstechnik, speziell in der industriellen Produktionstechnik in verschiedenen technologischen Ausprägungen und unterschiedlichen Einsatzszenarien verwendet. In einem Mehrachssystem werden mehrere solcher Achsen in einem gemeinsamen Verbund für eine Automatisierungsaufgabe eingesetzt. Verbünde von Achsen werden meist durch eine zentrale Steuerung gesteuert. Anhand dieses zentralisierten Szenarios sollen die wesentlichen Komponenten einer positionsgesteuerten Achse nachfolgend erläutert werden.

Achsen werden in der Automatisierungstechnik hauptsächlich mittels elektrischer Motoren angetrieben. Sonderbauformen ermöglichen es auch über Verbrenner, Pneumatik oder Hydraulik Achsen anzutreiben. Diese Arbeit beschränkt sich auf die Betrachtung von elektrischen Antrieben, deren für die Arbeit relevante Komponenten in Abbildung 2-1 dargestellt und im Folgenden detailliert vorgestellt werden. Das in Kapitel 1 validierte Konzept zur Bewegungssynchronisierung kann aber ebenso mit Adaptionen auf alternative Antriebsformen übertragen werden.

2.1.1 Mechanischer Aufbau der Antriebsarten

Je nach technologischer Ausprägung wird die Motorbewegung, ausgelöst durch elektrische Energie, die vom Antriebsverstärker bereitgestellt wird, auf unterschiedlichem Weg in eine Antriebsbewegung überführt. Bei rotatorischen Antrieben wird die Bewegung der Motorwelle mit einem Getriebe und einer Kupplung auf einen Gelenkarm oder eine Gelenkverbindung übertragen.

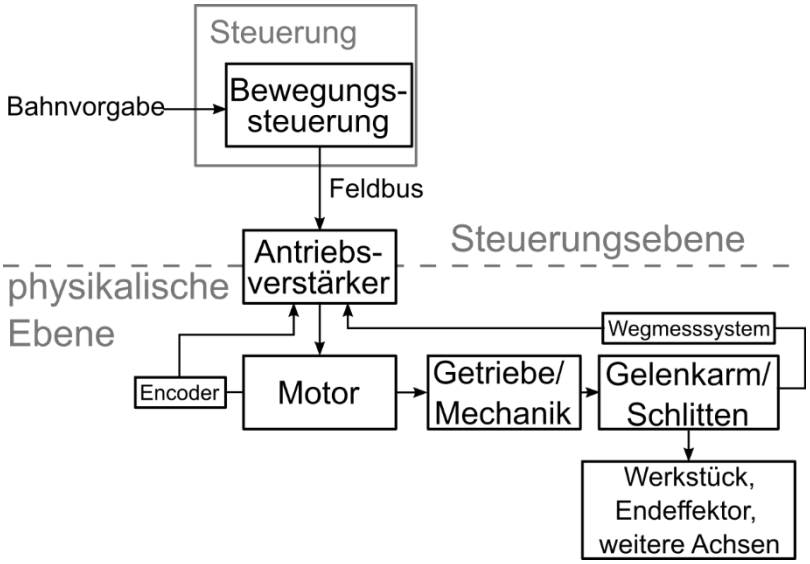

Abbildung 2-1: Komponenten einer positionsgesteuerten Achse.

Für lineare Bewegungen gibt es mehrere Umsetzungsarten: Beim Kugelgewindetrieb (KGT) wird die rotative Bewegung einer Motorwelle mittels Kupplung und einer Gewindestange auf eine mechanische Achse übertragen. Bei Zahnstangenritzelantriebe (ZRA) rollt die rotative Bewegung der Motorwelle über ein Zahnrad auf einer Zahnstange ab, um eine translative Bewegung des Schlittens zu erzeugen. Lineardirektantriebe (LDA) erzeugen ein veränderliches Magnetfeld, sodass ein Stator mit Schlitten auf dem Magnetfeld linear verfährt. In den betrachteten intelligenten Automatisierungskomponenten werden hauptsächlich LDA und KGT eingesetzt, in der Fördertechnik kommen oft auch ZRA zur Anwendung. Die Antriebsarten unterscheiden sich neben der mechanischen Bauform vor allem im dynamischen Verhalten (Lehner 2004, S. 18). Das unterschiedliche Übertragungsverhalten soll hier allerdings nicht näher betrachtet und nachfolgend als im gleichen Maße verlässlich für die Durchsetzung von Sollwerten angenommen werden.

2.1.2 Antriebsverstärker und Lageregelung

Der elektrische Motor erhält über einen Antriebsverstärker die notwendige Energie um von einer Steuerung vorgegebene Positionen im Antrieb zu erreichen. Der Antriebsverstärker erhält über einen Encoder und/oder ein abtriebsseitiges Messsystem Positionsdaten des Motors bzw. der Achse. Ein Kaskadenregelkreis sorgt im Antriebsverstärker bei Abweichung der Istwerte von den Sollwerten für die Durchsetzung der vorgegebenen Position und Verringerung des Schleppfehlers, s. Abbildung 2-2. Der Kaskadenregelkreis besitzt für die Position, die Geschwindigkeit sowie die Beschleunigung (nicht abgebildet; entspricht näherungsweise dem Motorstrom) einzeln konfigurierte Regler. Bei der Auslegung des Systems werden die Regler mit ihren Proportionalfaktoren K_a, K_v, K_p sowie möglichen Integrator- oder Differentialfaktoren von der innersten zur äußersten Kaskade schrittweise ausgelegt. Der Lage- oder Geschwindigkeitsregelkreis läuft in einer Taktrate von unter 1 Millisekunde (Dürkop 2016, S. 17).

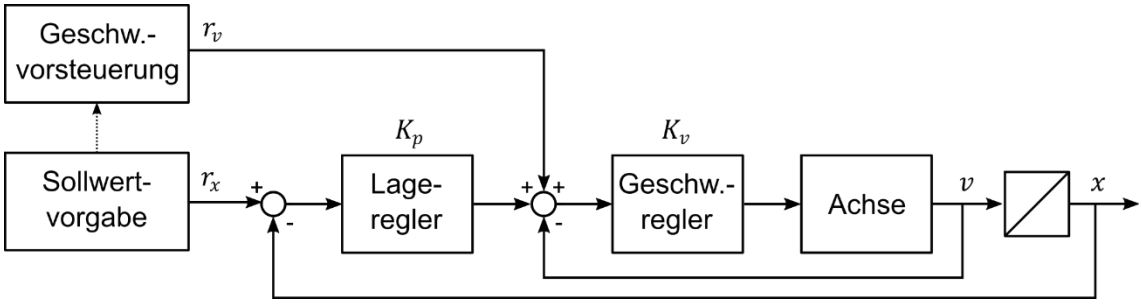

Abbildung 2-2: Blockschaltbild des Kaskadenreglers mit Geschwindigkeitsvorsteuerung.

Eine Vorsteuerung der Geschwindigkeit oder Beschleunigung (nicht abgebildet) kann das Folgeverhalten des Systems zusätzlich verbessern. Hierzu werden aus dem Sollwertverlauf die zu erwartenden Geschwindigkeiten und Beschleunigungen abgeleitet und als Vorgabe an die innere Kaskade des Regelkreises verwendet. In speziellen Anwendungen (Rundtische, Druckwalzen o.ä.) kann statt einer vorgegebenen Position auch eine Geschwindigkeitsregelung umgesetzt sein. In diesem Fall entfällt die äußerste Kaskade des Regelkreises und statt einer Sollposition wird eine Sollgeschwindigkeit umgesetzt. Es werden statt dem Kaskadenregelkreis nur in Sonderfällen andere regelungstechnische Ansätze verwendet, sodass im Folgenden der Kaskadenregler als Standard zur Durchsetzung von Sollwertvorgaben bei Achsen vorausgesetzt wird.

2.1.3 Bewegungssteuerung

Bewegungssteuerungen werden eingesetzt, wenn eine oder mehrere Achsen in definierter Art und Weise verfahren werden sollen. Der definierte Verlauf der Position wird als Bahn oder Bahnvorgabe bezeichnet. Um die Sollwerte dieser Bahn für den Positionsregelkreis der Achse zu generieren, müssen diese in der Bewegungssteuerung (Motion Control: MC) vorbereitet werden. Hier werden Systeme eingesetzt, die numerische Funktionen in Taktzeiten von unter 1 Millisekunde berechnen können (Seitz 2015). Bahnvorgaben werden durch stetige mathematische Funktionen beschrieben, die bestimmten Randbedingungen unterliegen und aus denen Sollwerte für die Achsen bestimmt werden müssen. Die Sollwertgenerierung nennt man auch Interpolation und teilt diese in zwei Hauptformen ein. Zum einen müssen Sollwertsprünge vermieden und stattdessen glatte Verläufe der Vorgabewerte generiert werden, dies nennt man Feininterpolation. Weiter werden Bewegungen meist in Kreisbögen, Geraden oder Polynomkurven definiert, die entlang des Kurvenverlaufs ausgewertet werden müssen. Die Definition dieser Segmente und Kurvenstücke nennt man Grobinterpolation. Erste numerische Steuerungen führten die Grobinterpolation noch über eine elektrische Schaltlogik für lediglich die grundlegendsten Kurvenformen durch (Papaioannou 1979). Inzwischen sind Computeralgorithmen für die Berechnung und Auswertung verantwortlich. Neben der geometrischen Betrachtung der Bahn muss auch die dynamische Durchführung (Trajektorie) vorberechnet werden, s. Kapitel 5.1. Achsen, die gemeinsam eine Manipulations- oder Bearbeitungsaufgaben umsetzen, werden als synchrone Achsen bezeichnet und werden innerhalb eines Achsverbundes mit dynamisch gleichbedeutenden Sollwerten versorgt, um eine

möglichst hohe Bahntreue zu erreichen. Sogenannte asynchrone Achsen sind meist Hilfsachsen (z. B. in Werkzeugmaschinen für den Werkzeugwechsel) und werden ohne direkte Abhängigkeit zu anderen Bewegungen angesteuert (Kircher 2011). Synchronisierungsspezifische Aspekte der Steuerung werden in Kapitel 2.2 betrachtet. Die Trajektorienplanung, die die zeitliche Durchsetzung der Sollwerte betrachtet, wird in Kapitel 5.1 im Detail erläutert. Im Nachfolgenden bezeichnet der Begriff der Interpolation in allen Fällen Umsetzungen von Feininterpolationen, da die Bahnvorgabe bereits als Sollwerte, also nach der Grobinterpolation, zur Verfügung steht.

In der PLCopen Motion Control Spezifikation (PLC Open 2011) werden grundlegende Funktionen einer MC wie Statusinformationen, koordinierte Bewegungsabläufe oder Referenzabläufe in Funktionsbausteinen der (DIN EN 61131-3) spezifiziert, sodass diese auch in speicherprogrammierbaren Steuerungen (SPS) unabhängig von der eingesetzten Achshardware integriert werden können (van der Wal 2001). Einige Hersteller bieten auch einen Zugriff auf zusätzliche, vordefinierte Funktionen der MC. Teilweise wird durch MC-Hersteller eine Anbindung dieses Funktionsumfangs an spezielle Softwarebibliotheken ermöglicht, sodass die Funktionen der MC auch über Softwareanwendungen durch einen externen Trigger gestartet und ausgeführt werden können (LinMot 2017; Molano et al. 2018)

In Werkzeugmaschinen mit vielen bewegten Achsen ist die eingesetzte Bewegungssteuerung in eine computerbasierte numerische Steuerung (CNC; Computerized Numerical Control) eingebettet, die zusätzliche Steueralgorithmen, vor allem die Transformation einer Bahndefinition auf Achskoordinaten, bietet. Teilweise werden dem Nutzer Abläufe innerhalb der Steuerung offengelegt, sodass zusätzliche Algorithmen an speziellen Zeitpunkten im Ablauf der Berechnungen integriert werden können (Schröder 2007). Die Eingriffsmöglichkeiten sind jedoch sehr eingeschränkt und benötigen Expertenwissen in den Zusammenhängen der Parameter aller ablaufenden Algorithmen. Dadurch ist eine umfangreiche Rekonfiguration der angesteuerten Achsen in CNC-Systemen nicht möglich (Kircher 2011, S. 16).

2.1.4 Feldbussystem

Da zentrale Steuerungen unterschiedlich viele Achsen steuern können, sind die kommerziellen Systeme modular aufgebaut. Die zentrale Steuerung läuft auf einer baulich vom Motor getrennten Einheit (Seitz 2015, S. 185). Die Antriebsverstärker sind direkt mit den Motoren und der Steuerung verbunden. Antriebsverstärker und zentrale Steuerung kommunizieren über sogenannte Feldbusse unter Verwendung spezifischer Feldbusprotokoll. Diese übertragen die benötigten Informationen zu den erwarteten Bewegungen der Motoren in einem spezifizierten Format. Die ersten Entwicklungen von Feldbusprotokollen begannen bereits 1975. Zur Jahrtausendwende wurden in IEC 61158 (DIN EN IEC 61158-1) die grundlegenden Struktur der Kommunikationsschichten definiert; aufgeteilt nach Diensten und Protokollen. Aktuelle Feldbusse basieren auf dieser Struktur. Implementierungskonzepte sind in der IEC 61784 (EN IEC 61784-1) nachträglich 2002 zusammengefasst worden (Klasen et al. 2010, S. 15).

Seit 1975 sind viele weitere Feldbussysteme konzipiert und implementiert worden, die nicht zwangsläufig bereits bestehende Technologien abgelöst haben (Klasen et al. 2010, S. 5). Heutige Feldbusse nutzen als „Industrial Ethernet"-Systeme die physikalischen Schichten des Open-Systems-Interconnection-Modells (OSI; Referenzmodell für Netzwerkprotokolle), allerdings erweitert um spezielle Nachrichtenformate und Echtzeit- sowie Sicherheitsaspekte. Hierdurch ergibt sich der Vorteil, dass unabhängig von der verwendeten Feldbustechnologie eine allgemein verwendbare Ethernet-Schnittstelle verbaut werden kann (Beckhoff 2010, S. 2).

In der industriellen Produktion liegt das Hauptaugenmerk auf schneller und verlässlicher Kommunikation, um auch funktionale Sicherheit zu gewährleisten (Klasen et al. 2010, S. 5). Dabei laufen Feldbuskommunikation und Steuerungsprozesse meist in einer Zeittaktung von 1 – 10 Millisekunden (Dürkop 2016, S. 17; Lesi et al. 2019, S. 711). Zahlreiche Erweiterungen, wie zum Beispiel Real-Time-Ethernet mit Publisher-Subscriber-Verfahren (Beckhoff 2010), setzen diese Kommunikation in echtzeitfähigen Systemen um.

Aus den in den 1980ern entwickelten Technologien für industrielle Kommunikationssysteme haben sich zwischenzeitlich achtzehn verschiedene Feldbusprotokolle bzw. Protokollfamilien entwickelt (Klasen et al. 2010, S. 20–24), neue Ethernet-basiert Feldbusse werden laufend in der Norm IEC 61784 integriert (Klasen et al. 2010, S. 16). Aktuelle kommerzielle Lösungen kombinieren die Verkabelung der Kommunikationstechnik zusätzlich mit der Leistungsversorgung in einem gemeinsamen Hybridkabel, sodass der Aufwand zur Inbetriebnahme von Achssystemen weiter verringert wird (Sercos News 2017).

2.2 Grundlagen der Bewegungssynchronisierung

In Anwendungsfällen, in denen mehrere Achsen in einem gemeinsamen Arbeitsraum verfahren, gibt es verschiedenen Stufen der Synchronisierung und daraus entstehend unterschiedliche Effekte in der Bahntreue im Anwendungsfall Pick-and-Place mit (fein-)interpolierenden Achssystemen:

1. Sequentielle, ereignisbasierte Bewegungssteuerung
2. Asynchrone Bewegungssteuerung
3. Synchrone Bewegungssteuerung

Für die ersten beiden Bewegungsschemata sind bereits Ansteuerkonzepte für dezentrale Systeme vorhanden, die im Folgenden vorgestellt werden.

2.2.1 Sequentielle, ereignisbasierte Bewegungssteuerung

Bei der sequentiellen, ereignisbasierten Bewegung werden alle Achsbewegungen strikt nacheinander durchgeführt. Es entstehen eckige Bahnverläufe und es kommt zu einem Halt in der Ecke der Bahn, s. Abbildung 2-3 (Geschwindigkeitsprofil vereinfacht). Zwar ist die Bahn der entstehenden Bewegung klar definiert, dieses Vorgehen ist jedoch weder zeit- noch wegoptimal.

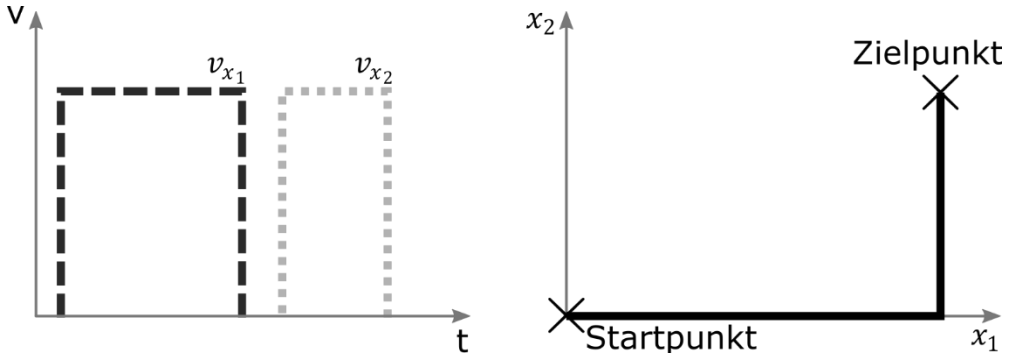

Abbildung 2-3: Resultierende Bahn einer vereinfachten zweidimensionalen sequentiellen Bewegungssteuerung.

Die sequentielle, ereignisbasierte Bewegungssteuerung für Miniatursteuerungen wird unter anderem durch den Application Composer (3S 2016), einer Entwicklung im Forschungsprojekt OPAK, angeboten. Komponenten verschiedener Technologien können in der dort verwendeten Ablaufsprache in ihren Funktionalitäten angesprochen und dadurch die Fähigkeiten von Komponenten (z. B. Positionierung einer Achse) ausgelöst werden. Vor- und Nachbedingungen definieren die Transitionen zwischen den Auslösezeitpunkten der verschiedenen Komponenten.

2.2.2 Asynchrone Bewegungssteuerung

Bei der asynchronen Bewegungssteuerung werden die Bewegungen aller Achsen gleichzeitig ausgelöst. Je nach vorhandener Dynamik und zu bewältigender Positionsdifferenz beenden einzelne Achsen ihre Bewegung früher als andere, wodurch eine gekrümmte Kurve entsteht, die mit einer Bewegung parallel zur zeitlich länger verfahrenden Achse endet. Speziell bei veränderlichen Dynamiken oder Positionsdifferenzen kann der Verlauf einer Bahn daher nicht im Vorhinein bestimmt werden. Die Durchführung ist zwar zeitoptimal, da jede Achse mit der maximalen ihr verfügbaren Geschwindigkeit verfährt, führt jedoch zu nicht vorher bestimmten Bahnkurven, s. Abbildung 2-4, die im Allgemeinen nicht wegoptimal sind.

Die asynchrone Bewegungssteuerung kann als eine Anwendung verteilt auf mehreren Ressourcen mit IEC 61499 (DIN EN 61499) umgesetzt werden. Hierfür werden in den IEC 61499-Funktionsblöcke definiert, in denen Algorithmen zur Aktivierung der Fähigkeiten der Komponenten hinterlegt sind. Beim Erhalt von Eingangssignalen werden über einen internen Zustandsautomaten, dem Execution Control Chart, die Algorithmen mit den anliegenden Parametern und ggf. vorhandenen internen Variablen ausgeführt (Yoong et al. 2014, S. 11), s. auch Abbildung 2-5.

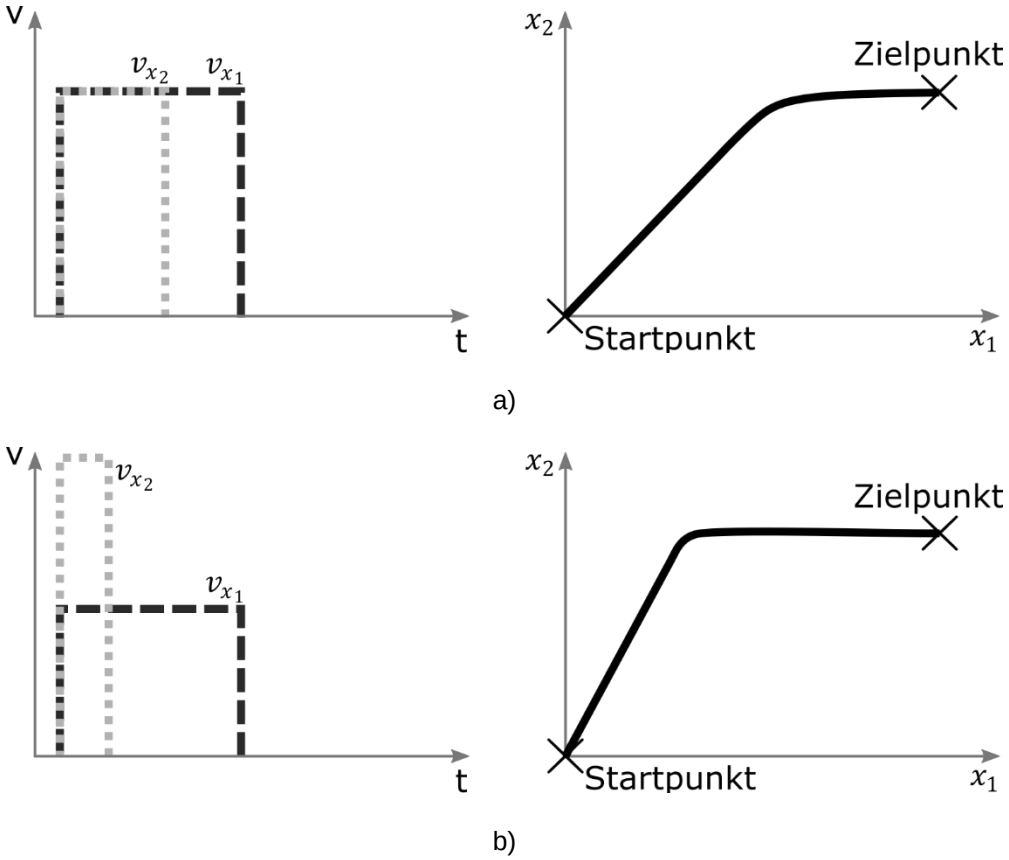

Abbildung 2-4: Resultierende Bahnen einer vereinfachten zweidimensionalen asynchronen Bewegungssteuerung. a) für gleiche und b) für unterschiedliche Verfahrgeschwindigkeiten v_x, v_y beider Achsen.

Die IEC 61499 wurde explizit zur Entwicklung von dezentralen Steuerungssystemen erarbeitet und im Jahr 2005 verabschiedet. Seither gibt es eine große Anwendergruppe, die den Einsatz der Norm vorantreibt. Die größten kommerziellen Anbieter haben diese Form der Definition von Steuerungsabläufen allerdings bisher nicht in ihre Engineeringumgebungen oder Produkte integriert (Hirsch 2010, S. 1). Im Gegensatz zur SPS-Programmierung über IEC 61131-3 (DIN EN 61131-3), die durch die zyklischer Ausführung von Diagrammen und Skripten zur Umsetzung der Steuerungslogik definiert wurde, definiert das Konzept der IEC 61499 keinen Zyklus oder eine Zeitdefinition. Ein Ereignis wie der Erhalt eines Eingangssignals ist genau zu einem einzigen Zeitpunkt verfügbar, unabhängig davon, ob darauf eine Handlung abgeleitet wurde oder nicht und ist damit vergleichbar mit einer Transition in eines Zustandsautomaten (Yoong et al. 2014, S. 68). Dadurch ist keine Gleichzeitigkeit von parallel ausgeführten Berechnungen sichergestellt, sondern sie hängt direkt von der Implementierung der gleichzeitig auszuführenden Algorithmen ab (Yoong et al. 2014, S. 10). Weiter ist auch keine Komposition der Funktionsblöcke vorgesehen, um Funktionsblöcke in hierarchischen Beziehungen anzuordnen. Die fehlenden Vorgaben bezüglich der Komposition und des Zeitverhaltens bei parallelen Ereignissen führen dazu, dass Algorithmen in verschiedenen Funktionsblöcken unterschiedlich interpretiert werden können und es keine klare Vorgabe zu synchron ausgelösten Funktionsblöcken gibt (Yoong et al. 2014, S. 66).

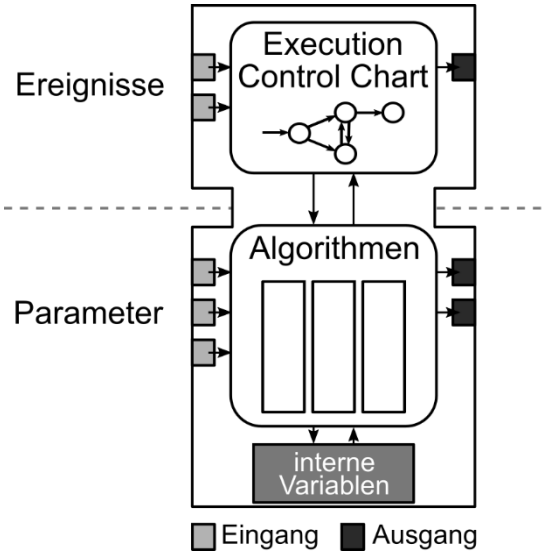

Abbildung 2-5: Funktionsbaustein der IEC 61499(DIN EN 61499) nach (Hirsch 2010, S. 20).

Bei Anwendung der IEC 61499 kann also im Optimalfall eine Aktivierung von Komponentenfähigkeiten zeitgleich ausgelöst werden, eine Synchronisierung während der Ausführung ist allerdings nicht möglich, da zwischen Funktionsblöcken während der Ausführung der Algorithmen kein Bezug möglich ist.

Es gibt vergleichsweise viele umgesetzte Anwendungsfälle der IEC 61499, obwohl nur ein einziger kommerzieller Anbieter (NXTcontrol 2018) sowie eine Open Source Initiative (Eclipse Foundation 2007) eine Engineeringumgebung anbieten. Eine Umsetzung eines Pick-and-Place Produktionsanwendungsfalls mit einer Greifbewegung inklusive Manipulation des Werkstücks nach IEC 61499 mit Petrinetzen ist beispielsweise von (Ivanova-Vasileva et al. 2008) vorgestellt worden. Die Ausführung des Bewegungsablaufs wird allerding sequentiell durchgesetzt. Eine weitere Anwendung wird von (Kollegger et al. 2016) vorgestellt, die das Konzept verwenden um die Komplexität der Aktivierungsalgorithmen bzw. der Steuerungsabläufe durch die Funktionsblöcke für die Anwendersicht zu kapseln. Dieser Ansatz ähnelt der angestrebten Verwendung der Elementarfunktionen bei DEVEKOS-Komponenten. Eine allgemeine Systematik zum Systementwurf mit IEC 61499 wird allerdings weder durch die Norm noch durch die Anwendergruppe vorgegeben (Hirsch 2010, S. 3).

2.2.3 Synchrone Bewegungssteuerung

Bei der synchronen Bewegungssteuerung werden die vorhandenen Dynamiken der Achsen nicht maximal ausgenutzt, sondern so skaliert, dass alle beteiligten Achsen die Bewegung gemeinsam beginnen und beenden. Dadurch entsteht bei linearen Achsen eine Gerade im Raum, s. Abbildung 2-6. Zumeist wird die angestrebte Bahn zuvor definiert und dann die Dynamiken und Bewegungen der einzelnen Achsen so angepasst, dass diese in abgestimmter Dynamik entlang der Bahn verfahren. Dieser Ansatz ist ebenso wie die asynchrone Bewegungs-

steuerung zeitoptimal, wenn bei der langsamsten Achse die Dynamik voll ausgenutzt wird.

Die synchrone Bewegungssteuerung ist die Grundfunktionalität einer zentral gesteuerten MC. Eine synchrone Bewegungssteuerung bei der Verwendung von dezentralen Miniatursteuerungen ist bisher jedoch weder kommerziell noch in Forschungsszenarien vorhanden. Diese Arbeit geht in Kapitel 1 auf mögliche Systemkonzepte für dezentrale Systemarchitekturen ein.

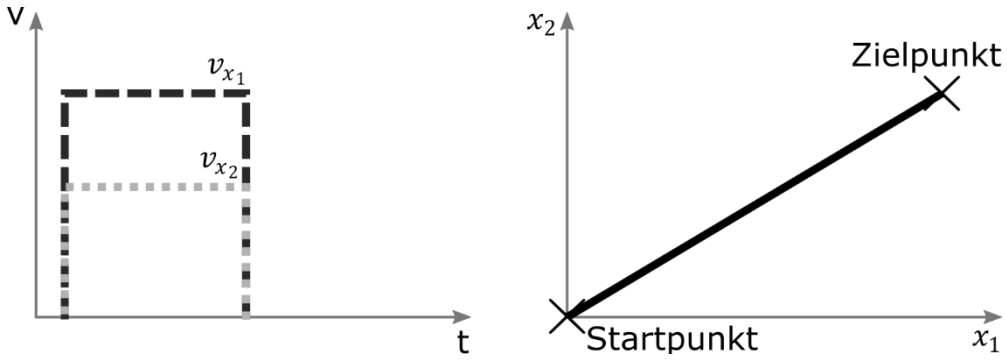

Abbildung 2-6: Resultierende Bahn einer vereinfachten zweidimensionalen synchronen Bewegungssteuerung.

2.3 Methoden und Systeme zur Bewegungssynchronisierung

In den meisten automatisierten Montageanwendungen werden Systeme mit zwei und mehr Achsen eingesetzt. Meist teilen sich diese Achsen einen gemeinsamen Arbeitsraum und die Anwendung wird als Kombination mehrerer Achsbewegungen realisiert. Daher ist die Synchronisierung solcher Bewegungsachsen miteinander ein geläufiges Automatisierungsproblem, siehe Kapitel 2.2, für das es die unterschiedlichsten etablierten Lösungsansätze gibt, die nachfolgend vorgestellt werden. Synchronisierung bezeichnet im Allgemeinen einen zeitlichen Abgleich von Vorgängen, nachfolgend ist damit die Abstimmung zwischen Bewegungen der Achsen entweder über eine gemeinsame Zeitachse oder aber über Positionen von weiteren Achsen gemeint. Ein Anwendungsbeispiel für sogenannte Gleichlaufsysteme mit synchronisierten Bewegungen ist die „fliegende Säge“ bei dem ein auf einem Transportband verfahrenes Schnittgut durch eine auf einem Schlitten verfahrene Säge senkrecht zur Bewegungsrichtung zerteilt wird. Hierfür muss die Bewegung des Schlittens während des Sägevorgangs synchron zur Bewegung des Transportbandes laufen (Seitz 2015). Eine kompakte Übersicht über die bekanntesten der nachfolgend vorgestellten Synchronisierungsmethoden und -systeme ist in (Pérez-Pinal et al. 2004) zu finden.

2.3.1 Königswellen mit Kurvenscheiben

Sogenannte Kurvenscheiben wurden bereits in vorchristlicher Zeit für definierte Bewegungen von Hebeln in Automaten verwendet. Hierbei werden auf einer rotierenden Welle Scheiben aufgebracht. Ein Hebel drückt auf die Kontur der Scheibe und wird bei Drehung der Welle entsprechend der Scheibenkontur

ausgelenkt, s. Abbildung 2-7. Dieses Prinzip wird auch heute noch in Rundtakt- oder Linienfertigungsanlagen mit hohen Taktraten verwendet, da die Königswelle im laufenden Betrieb wenig fehleranfällig ist. Werden auf der rotierenden Welle mehrere Scheiben für die Betätigung mehrerer Hebel befestigt, bestimmt die Rotationsgeschwindigkeit der Welle die Bewegung aller Hebel und wird als Königswelle bezeichnet. Bei dieser Art der Synchronisierung muss bei Konzeption und Inbetriebnahme viel Aufwand in die korrekte Konstruktion aller Kurvenscheiben fließen. Alle Achsbewegungen müssen zu Hebelbewegungen auf den Scheiben zurückprojiziert werden (Heine 2015). Zusätzlich müssen aufgrund der materialtechnischen Eigenschaften der Königswelle und Kurvenscheiben sowie der hohen wirkenden Kräfte auch Verformungen wie z. B. die Torsion der Königswelle antizipiert und bei der Inbetriebnahme iterativ optimiert werden. Eine Rekonfiguration wird bei solchen Produktionssystemen im Allgemeinen aufgrund der hohen Aufwände ausgeschlossen. Bei Kollisionen innerhalb der Maschine, z. B. mit dem gehandhabten Produkt, ist eine Verklemmung der mechanischen Bauteile häufig (Gutt 2014, S. 238). Es ist in den meisten Kollisionsfällen ein hoher manueller Wartungsaufwand für die Wiederaufnahme des Betriebs zu erwarten.

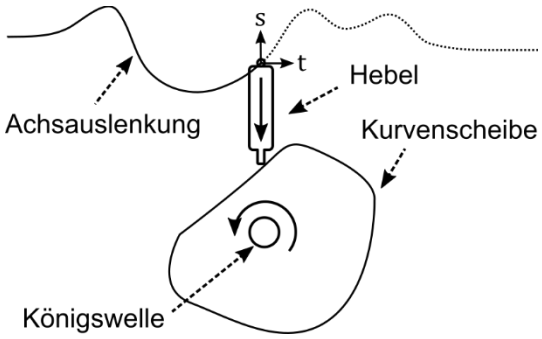

Abbildung 2-7: Definition von Achsbewegungen über die Kurvenscheibenkontur der rotierenden Königswelle.

2.3.2 MC und CNC-Systeme

Moderne Produktionssysteme, die mithilfe von synchronisiert bewegten Achsen zum Beispiel Fertigungstechnologien wie Hochgeschwindigkeitsfräsen anbieten verwenden CNCs um die Ansteuerung der Antriebe miteinander zu synchronisieren. Die Interaktion wird meist durch Bahndefinitionen textuell in definierten Programmen (G-Code) nach DIN 66025/ISO 6983 (DIN 66025-1) vorgegeben.

Integriert in CNCs ist die MC, die für Anwendungen außerhalb von Werkzeugmaschinen auch als eigenständige Steuerung mit geringerem Funktionsumfang erhältlich ist (Jetter 2010; Siemens 2013; zub 2018). Sie berechnet die Trajektorien zu vorgegebenen Sollpositionen, also die Bewegungsabläufe einzelner Achsen unter Beachtung der Achsdynamiken. MCs haben im allgemeinen verschiedene Funktionsmodi, sodass neben einer Positionssteuerung auch eine Geschwindigkeitssteuerung, Punkt-zu-Punkt Steuerungen, sowie in manchen Fällen auch eine Kraftsteuerung (LinMot 2020, S. 882) ermöglicht wird. Einige Anbieter bieten eine MC auch innerhalb von Antriebsverstärkern mit Softwareschnittstellen an (LinMot 2017). Eine MC verwendet explizite mathematische Zusammenhänge

zwischen den bewegten Achsen zur Generierung der Trajektorien. Eine CNC dagegen berechnet aus einer Bahndefinition basierend auf einer Transformation die resultierende Achsbewegungen.

Die MC übernimmt als zentrale Recheneinheit die Berechnung der Korrekturterme sowie der umzusetzenden Trajektorien und leitet die Sollwerte per Feldbus an die Antriebsverstärker weiter. Die Antriebsverstärker implementieren den Lageregelkreis der Achsen, s. Abbildung 2-8. MCs sind heute sehr leistungsfähig und ermöglichen schnelle Achsverfahrbewegungen mit hoher Synchronität und geringer Bahnabweichung für verschiedenste Einsatzzwecke. Da jedoch nicht in allen Anwendungsfällen eine solche hierarchische Struktur gewünscht oder die Rechenleistung der MC oder CNC benötigt bzw. deren Kosten begründbar sind gibt es weitere Methoden die ebenfalls eine Synchronisierung von Achsbewegungen umsetzen.

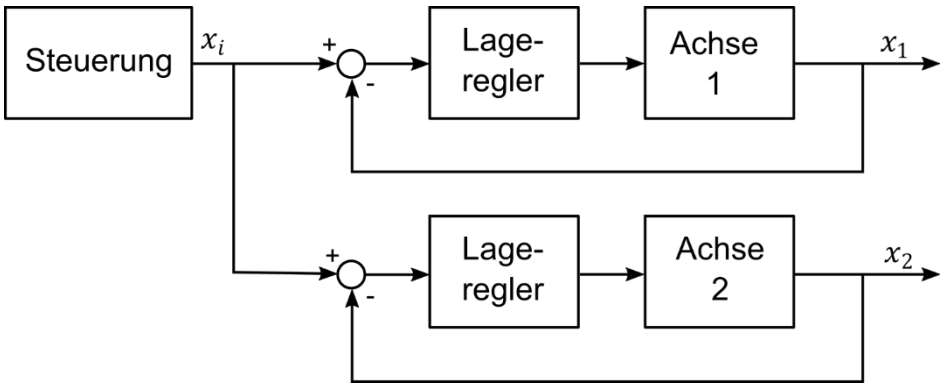

Abbildung 2-8: Blockschaltbild der Sollwertvorgabe der zentralen Steuerung (MC/CNC) an die Achsen, nach (Tan et al. 2007, S. 105).

2.3.3 Leader-Follower-Architekturen[1]

Leader-Follower-Architekturen werden eingesetzt, wenn mehrere positionierende Achsen explizit im Gleichlauf verfahren sollen, also dieselben Sollwerte erhalten. Ein bekannter Anwendungsfall für Leader-Follower-Architekturen sind Portalsysteme, in denen eine Last von zwei Achsen getragen wird, die synchron verfahren werden müssen um ein Verkanten des Portals in den Führungen zu vermeiden (Yao 2015, S. 1). Hier werden aus Kostengründen oft Leader-Follower-Architekturen statt der zentralen Koordination in einer MC oder CNC zur Synchronisierung eingesetzt (Tan et al. 2007, S. 101 - 128).

Diese sind auch ohne eine zentrale Steuereinheit umsetzbar, indem man die Position bzw. Geschwindigkeit der Leaderachse direkt als Sollposition bzw. -geschwindigkeit der Followerachse verwendet, s. Abbildung 2-9. Für diese Architektur sollte die Followerachse eine gleichwertige oder höhere Dynamik als die Leaderachse haben, da ansonsten das Folgeverhalten abweichen kann. Weiter

[1] Die in technischen Dokumentationen verwendeten und in der nahen Vergangenheit kritisierten Begriffe „Master“ und „Slave“ (**Oberhaus 2018**) wurden in dieser Arbeit an allen Stellen durch die neutraleren Begriffe „Leader“ und „Follower“ ersetzt.

werden Störungen in der Leaderachse direkt an die Followerachse weitergegeben und gegebenenfalls verstärkt. Ein Abweichen der Followerachse von den Sollwerten, zum Beispiel durch auftretende Lasten, wird auf der Leaderachse durch die fehlende Rückkopplung nicht detektiert, sodass hier hohe Synchronisierungsabweichungen möglich sind (Pérez-Pinal et al. 2004).

Neben der Referenz auf realen Achsen als Leader, können in vielen Systemen auch virtuelle Leitachsen definiert werden (Bosch Rexroth 2007, S. 559).

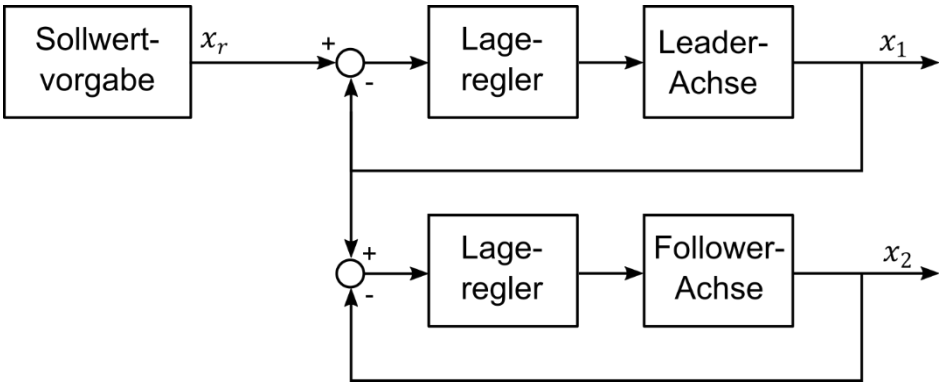

Abbildung 2-9: Blockschaltbild einer Leader-Follower-Struktur nach (Tan et al. 2007, S. 104).

2.3.4 Camming und Sollwertelisten

Wenn Achsen nicht im Gleichlauf synchronisiert werden sollen, wie bei Leader-Follower-Architekturen, sondern unterschiedliche Sollwertvorgaben erhalten, kann durch die Definition einer sogenannten Camming-Funktion (abgeleitet vom englischer Begriff der Kurvenscheibe aus Kapitel 2.3.1, dem cam) auch eine digitale Kurvenscheibe als Funktionsbeschreibung dieser Sollwertverläufe definiert werden, s. Abbildung 2-10. Der Zusammenhang zwischen einer Leaderachsenposition und der entsprechenden Sollwerte für die Followerachsen kann entweder mit einem Übersetzungsverhältnis vergleichbar zu einem mechanischen Getriebe oder als mathematische Funktion beschrieben werden. Die Camming-Funktion wird oft in MCs angeboten (Beckhoff 2020), die Beziehung kann jedoch auch ohne eine MC durch funktionsbasierte Skalierung der Sollwerteübergabe einer Followerachse umgesetzt werden.

Einige Anbieter von MC-Systemen bieten die Verwendung von Sollwertlisten zur Synchronisierung bewegter Achsen an. Diese kommen zum Einsatz, wenn eine Leader-Follower-Kopplung wie in Kapitel 2.3.3 beziehungsweise elektronische Kurvenscheibe/Königswelle nur schwer oder nicht mathematisch beschrieben werden kann. Stattdessen werden in diesem Fall explizit die Sollpositionen der Followerachsen zu jeder Position der Leaderachse angegeben. Zwischen diesen Positionen wird durch die MC eine Interpolation in Geschwindigkeit und Position durchgeführt.

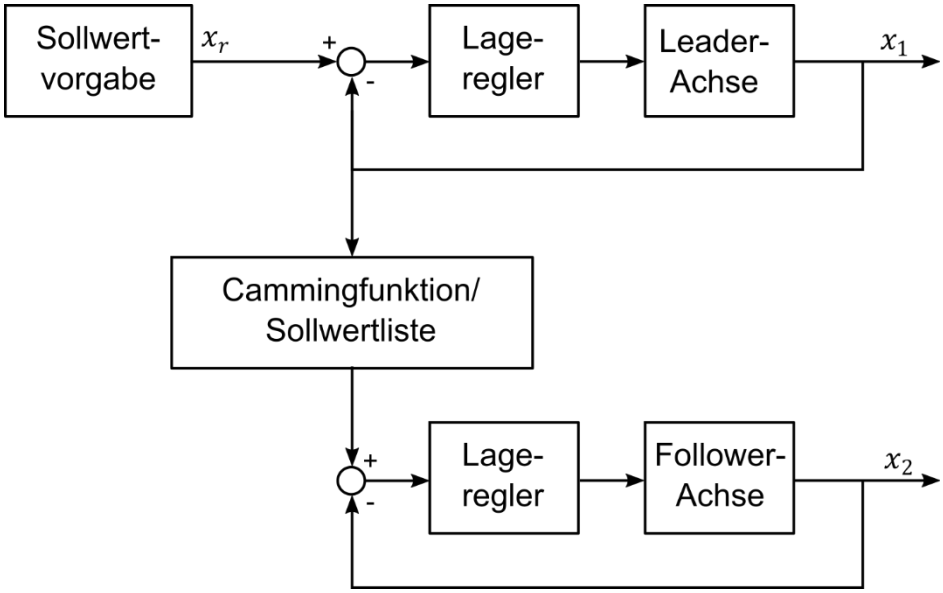

Abbildung 2-10: Blockschaltbild der Cammingfunktion/Sollwertliste als Zusammenhang zwischen der Bewegung einer Leader- und Followerachse.

Wird keine zyklische oder wiederkehrende Bahnbewegung ähnlich einer Kurvenscheibe angestrebt, so bietet beispielsweise die Firma Beckhoff in ihrer MC die Funktion der NC-FIFO-Achsen (FIFO: First-In-First-Out) an. Sie ermöglicht es, die Sollwertlisten ohne Referenz auf einer Leaderachsposition, sondern über ein SPS-Programm schrittweise, in einer durch den Anwender zu bestimmenden Geschwindigkeit, abzuarbeiten (Jetter 2010; Beckhoff 2020).

Die genannten Konzepte bieten also die Möglichkeit eine Bewegungssynchronisierung in Einzelschritten der Bahndefinition abzuarbeiten. Diese Durchführung findet in der zentralen Recheninstanz der MC auf Basis der dort bekannten Dynamik der Achsen statt.

2.3.5 Cross-Coupling

Auch bei der synchronen Vorgabe von Sollwerten durch eine zentrale MC durch Camming/Sollwertlisten ist speziell bei Achsen unter Last oder im Prozesseingriff ein Abweichen der Istpositionen von den vorgegebenen Sollpositionen gegeben. Da die Achsen jedoch komplett getrennte Lageregelkreise besitzen, werden solche Abweichungen nicht durch die weiteren, an der Bewegung beteiligten, Achsen bzw. deren Regelkreise berücksichtigt. Diese Ursache für Bahnabweichungen ist vergleichbar mit der fehlenden Rückkopplung von Sollpositionsabweichungen der Followerachse bei Leader-Follower-Architekturen.

Erstmals 1957 von Sarachik und Ragazzini (Sarachik et al. 1957) im asymmetrischen Fall vorgestellt, erweitert Koren 1980 das Cross-Coupling (CC; Kreuzkopplung) Konzept zum symmetrischen Fall für die Verwendung in Werkzeugmaschinen (Koren 1980, S. 265). CC koppelt die Lageregelkreise der Achsen durch eine zentrale Reglerinstanz, sodass die Sollwertvorgaben direkt in Abhängigkeit der Positionsabweichungen beider Achsen angepasst werden, s. Abbildung 2-11. Um eine hohe Synchronität und damit starke Kopplung der beiden

Achsen zu erreichen, werden sehr hohe Verstärkungen K_{pcc} im CC-Regler angewendet, die ihrerseits Fehler und Rauschen verstärken (Pérez-Pinal et al. 2004, S. 1671) .

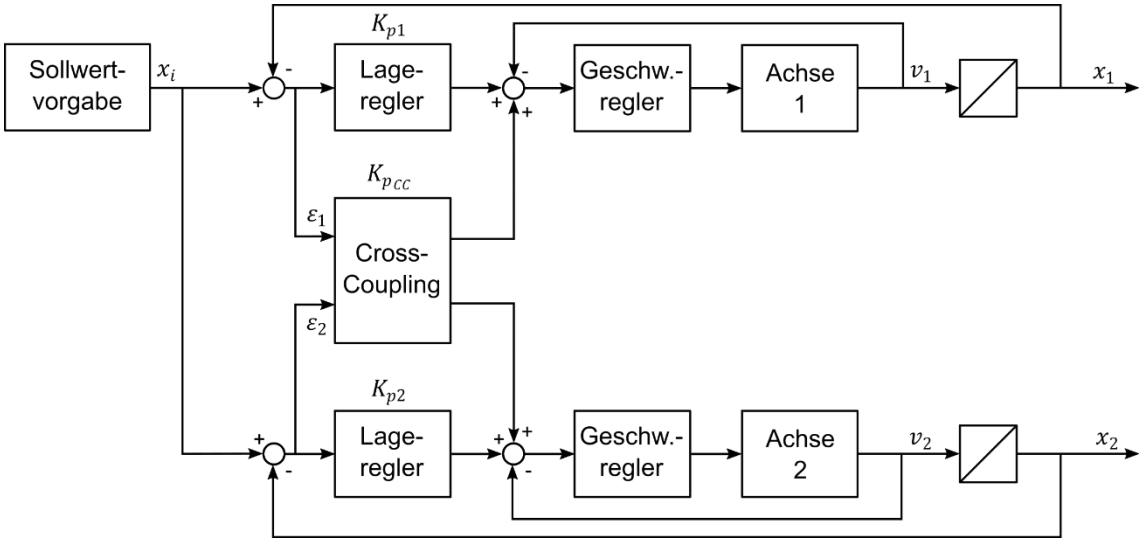

Abbildung 2-11: Blockschaltbild der Cross-Coupling-Synchronisierung nach (Srinivasan et al. 1990, S. 228).

Eine Erweiterung des CC-Konzepts auf mehr als zwei beteiligte Achsen ist nur in Einzelfällen verfolgt worden und wird meist durch die Kopplung von mehreren CC-geregelten Achspaaren (Jeong et al. 2008) oder die Betrachtung begrenzt vieler benachbarten Achsen umgesetzt (Sun et al. 2007).

2.4 Forschungsarbeiten zur Bewegungssynchronisierung

Die nachfolgenden Ansätze wurden im Gegensatz zu den Konzepten aus Kapitel 2.3 hauptsächlich wissenschaftlich betrachtet und analysiert und sind (noch) nicht in kommerziellen Systemen etabliert. Diese Forschungsaktivitäten werden von Defiziten in der Bahnsynchronisierung beziehungsweise der Digitalisierung von Produktionssystemen bis auf Komponentenebene motiviert. Sie zielen entweder darauf ab in einem zentralen modellbasierten Ansatz die Bahnabweichungen zu minimieren oder die Synchronisierung über dezentralisierte intelligente Komponenten zu realisieren.

2.4.1 Bahnregelung

Während bei CC-Reglern lediglich die Achspositionsfehler mittels eines proportionalen Verstärkungsfaktors K_p in die Sollwertvorgabe der gekoppelten Achse eingehen, wird in der Bahnregelung aus den Achspositionsfehlern der Bahnfehler im dreidimensionalen Raum bestimmt und aufgrund des Bahnverlaufs und den kinematischen Ausrichtungen der Achsen individuelle Korrekturterme für die Sollwertvorgaben bestimmt, s. Abbildung 2-12. Vorteilhaft ist, dass so auch bei unterschiedlichen Achsdynamiken eine synchronisierte Durchführung der Bahn möglich ist, da die Korrekturterme auf den aktuellen Bahnfehlern beruhen. Dies

erlaubt es, statt bisher identisch ausgelegten Lageregelkreisen für alle Achsen, die Achsen individuell auf maximale Dynamik zu regeln (Pritschow et al. 1990, S. 89).

Ein Bahnregler benötigt allerdings die Bahndefinition als Vorgabe um entsprechend des weiteren Bahnverlaufs optimale Sollwertvorgaben zu berechnen. Es bestehen Ansätze der Bahnregelung für die Geraden- und Kreisbogeninterpolation, deren Parametrierung den zur Programmierung verwendeten G-Code-Sätzen entnommen werden können (Huan 1982). Eine Erweiterung des Verfahrens auf parametrisch beschriebene Bahnen wie Splines, Ellipsen und andere wird im Ansatz durch (Huan 1982, S. 86) nur im Ausblick genannt. Sollte der Bahnfehler nicht als Funktionsausdruck der Positionsfehler der beteiligten Achsen ausgedrückt werden können, ist eine Umsetzung als Bahnregelung nicht möglich (Li et al. 2017, S. 2263). Bei einer Bahnregelung von mehr als zwei Achsen werden die Funktionsausdrücke außerdem sehr komplex (Huan 1982, S. 90–91), sodass der Berechnungsaufwand im Vergleich gegenüber konventionellen Steuerung mit Interpolator und Lageregelung zunimmt.

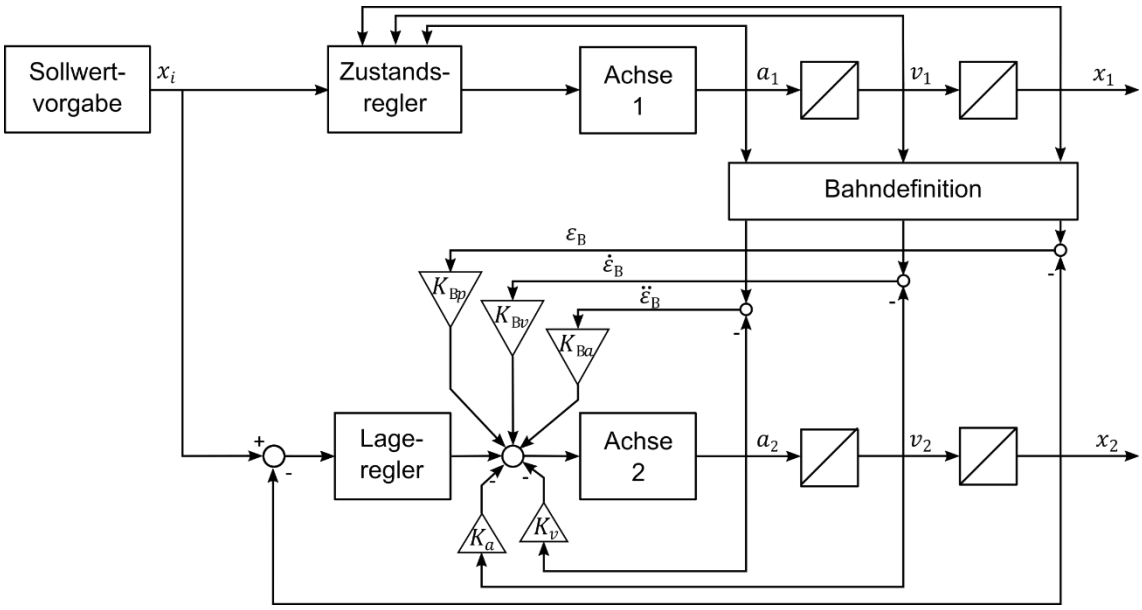

Abbildung 2-12: Blockschaltbild des Bahnreglers für zwei Achsen nach (Pritschow et al. 1990, S. 89).

Da für die Bahnregelung eine Achse als Leaderachse mit einer Zustandsregelung definiert wird und die Bahnfolgefehler auf einer Projektion der Bewegungsrichtung dieser Achse basieren, muss bei einer Richtungsumkehr der Leaderachse eine Glättung der Korrekturterme für die Followerachsen vorgenommen werden, um Unstetigkeiten zu vermeiden. Bei Bahnabschnitten, für die in der Leaderachse keine Bewegung vorgesehen ist, kann die Projektion aufgrund des Stillstandes nicht durchgeführt werden, weshalb ein Wechsel der Leaderachse empfohlen wird (Pritschow et al. 1990, S. 90). Auch aktuelle Forschungsarbeiten betrachten weiterhin eine möglichst optimale Bahnregelung als Funktionalität auf der zentralen Steuerung (Zhang et al. 2013, S. 341). Der Ansatz wird allerdings selten auf kommerziellen Systemen implementiert, vgl. (Pruschek 2009, S. 19).

2.4.2 Scheduling von Bewegungsausführungen auf eingebetteten Low-Level-Controllern

Aktuelle Forschungsarbeiten konzentrieren sich im Gegensatz zum zentralen Ansatz der Bahnregelung häufig auf die Synchronisierung mehrerer dezentral angesteuerten Komponenten. Häufig sind Mikrocontroller, FPGA oder andere miniaturisierte Systeme im Einsatz, die sowohl die Schnittstellen, die Sollwertberechnung sowie die Endstufe des Antriebsverstärkers der Achsen in einem System anbieten.

Die Forschungsarbeiten von Lesi (Lesi 2020) basieren auf seriell aufbauenden Achsen, die jeweils mit einer Mikrocontroller-Steuereinheit ausgerüstet wurden. Laut (Lesi et al. 2019) verhindern CNCs durch ihre hohe Komplexität und monolithische Struktur die einfache Rekonfiguration von Produktionskomponenten. In seinem Ansatz konzentriert sich Lesi auf die Synchronität der Ansteuersignale und trennt die Prozesskette einer CNC zwischen einem High-Level-Controller (HLC; übergeordnete Steuerung) und einem Low-Level-Controller (LLC; untergeordnete Steuerung), wobei der LLC auf allen Achs-Mikrocontrollern umgesetzt wird. Da es in den ersten Schritten der Prozesskette keine enge Kopplung oder den Austausch von achsspezifischen Funktionen gibt, kann die CNC-Funktionalität nach seiner Annahme in unabhängigen Modulen auf der Ebene der Achsen parallel ausgeführt werden (Lesi et al. 2016). Alle Funktionen, die direkt die Ansteuerung der Achsen betreffen oder echtzeitkritisch sind, werden der LLC übertragen. Die Trajektorienplanung sowie das Scheduling des Produktionsprozesses wird in der HLC ausgeführt (Lesi et al. 2019, S. 711). Die Steuerungslogik wird mittels Petrinetzen aufgebaut, sodass die HLC die untergeordneten Prozesse in den LLCs anstößt (Jakovljevic et al. 2019). Sichtbar ist im betrachteten Pick-and-Place Anwendungsfall (Lesi et al. 2019), dass die Ansteuerung der Achsen dezentral umgesetzt wird. Es wird allerdings lediglich eine Sequenz von Bewegungen aller Achsen ausgeführt, beziehungsweise Bewegungen zeitgleich initiiert. Eine Synchronisierung über den Verlauf der Bewegung wird nicht betrachtet und findet daher nicht statt. Der Ansatz ist mit der asynchronen Bewegung aus Kapitel 2.2.2 vergleichbar.

Speziell bei der vorangehend umfangreich dargestellten Abweichung von vorgegebenen Sollwerten durch lasttragende oder weniger dynamische Achsen muss hier also eine Bahnabweichung erwartet werden, die nicht durch die dezentralen Ansteuerungen ausgeglichen wird. Die von Lesi angenommene lose Kopplung der achsenspezifischen Funktionen bei CNC-Prozessen ist für einen konkreten Fall der einfachen Pick-and-Place Anwendung verwendbar, lässt sich aber bei der Erweiterung auf rekonfigurierbare Produktionsmittel mit veränderlicher relativer Positionierung oder gar dem Einsatz von rotativen Achsen nicht halten, da in diesen Fällen die CNC bereits in den Prozessen der HLC die Berechnung der Transformationen, also der kinematischen Kopplung übernimmt.

2.4.3 Betrachtung der Netzwerkverzögerungen zur synchronen Ansteuerung von dezentralen Mehrachssystemen

Ein weiterer aktueller Forschungsansatz in Bezug auf die synchrone Ansteuerung wird von Xu et al. präsentiert (Xu et al. 2017). Die Autoren adressieren die netzwerktechnischen Herausforderungen, die bei der Synchronisierung von dezentralen Mehrachssystemen zu behandeln sind.

Bei der Synchronisierung von Bewegungsanwendungen über ein Kommunikationsnetzwerk entsteht eine hohe Auslastung, da eine große Zahl von Vor- und Rückgabewerten schnell und periodisch ausgetauscht werden müssen. Die Weiterentwicklungen in der Netzwerktechnik konzentrieren sich meist auf den Quality-of-Service (QoS; Servicequalität), wohingegen die Entwicklung von Steuer- und Regelkonzepten den Fokus auf den Quality-of-Performance (QoP; Leistungsqualität) legt. Ersteres adressiert die Verlässlichkeit der Kommunikation, während der zweite Ansatz speziell Verzögerungen, Ausfälle und andere Kommunikationsfehler zusammenfasst (Xu et al. 2017, S. 116).

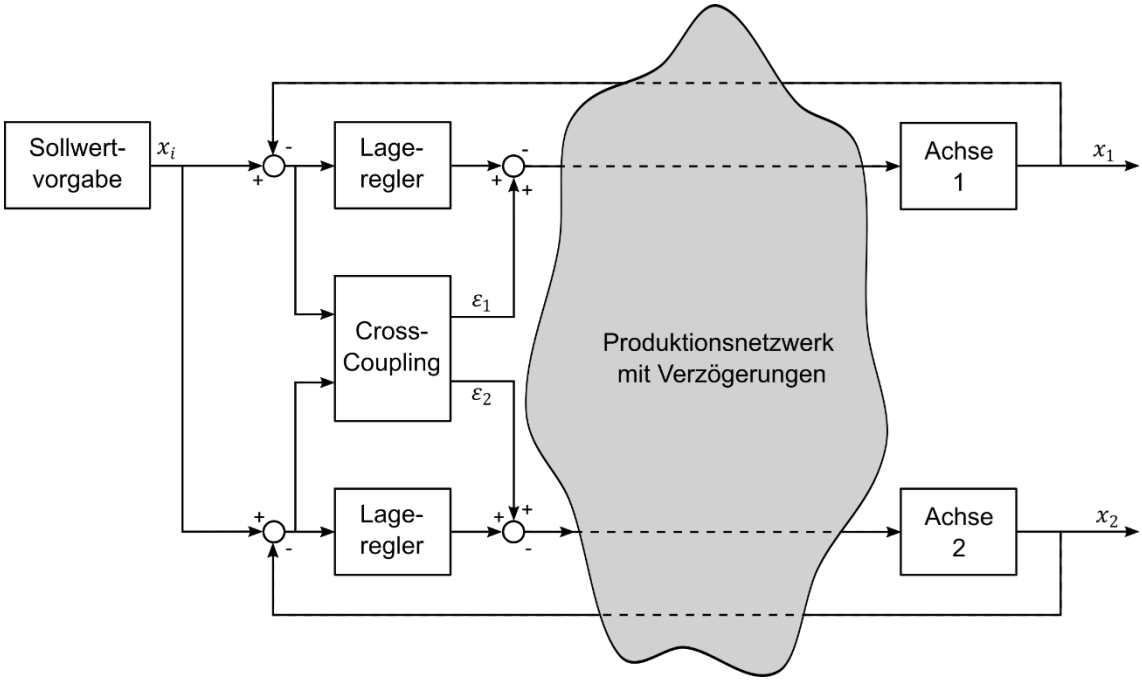

Abbildung 2-13: Blockschaltbild des Cross-Couplings über das Produktionsnetzwerk nach (Xu et al. 2011, S. 4380).

Xu et al. betrachteten QoP Ansätze für Regelkonzepte bereits in 2011 und schlugen ein CC-Konzept mit Zeitstempeln vor (Xu et al. 2011). Einzelne Achsen sind in diesem Ansatz mit einem individuell eingestellten Lageregelkreis ausgestattet und kommunizieren Istpositionen mit einem Zeitstempel über das Kommunikationsnetzwerk, s. Abbildung 2-13. Die zentrale CC-Instanz ordnet die Ist- und Sollwerte beider Achsen entsprechend der Zeitstempel ein und leitet Korrekturterme ab, die wiederum über das Netzwerk den Lageregelern der Achsen zur Verfügung gestellt werden. Es kann eine verbesserte Bahntreue im Gegensatz zu einem CC-Regler, der über eine Netzwerkkommunikation Daten der Achsen ohne Zeit-

stempel erhält, erreicht werden. Sollte ein Netzwerk mit nahezu gleichbleibenden Kommunikationsverzögerungen vorliegen, ist allerdings keine höhere Bahntreue durch die Verwendung der Zeitstempel zu erwarten.

In einem alternativen Ansatz betrachtet dieselbe Forschergruppe eine Leader-Follower-Struktur ohne zentrale Steuerung und berücksichtigt die zu erwartende Netzwerkverzögerung in den Reglern der Followerachsen. Sie weisen die asymptotische Stabilität der Reglerstruktur inklusive des Netzwerks nach (Xu et al. 2017). Im Gegensatz zum vorherigen Ansatz (Xu et al. 2011) wird außer der Vorgabe der Referenztrajektorie keine zentrale Einheit benötigt und alle Regler sind dezentral implementiert, s. Abbildung 2-14. Der Ansatz ist dadurch auch auf beliebig viele Teilnehmer erweiterbar. In der simulativen und experimentellen Validierung wird jedoch sichtbar, dass aufgrund der verketteten Regler bei der jeweils letzten Achse der Kette Positionsfehler nur zeitverzögert ausgeregelt werden können und keine Rückkopplung der Fehler an die vorherigen Achsen existiert. Weiter werden die Regler zwar nach einem allgemeinen Schema entwickelt, sie werden aber je nach Position der Achse in der Leader-Follower-Kette individuell parametrisiert. Die theoretische Erweiterbarkeit auf beliebig viele Teilnehmer wird in der Realisierung vor allem durch die skalierende Zeitverzögerung in den Reglern und der damit begrenzten, erreichbaren Leistungsqualität stark eingeschränkt.

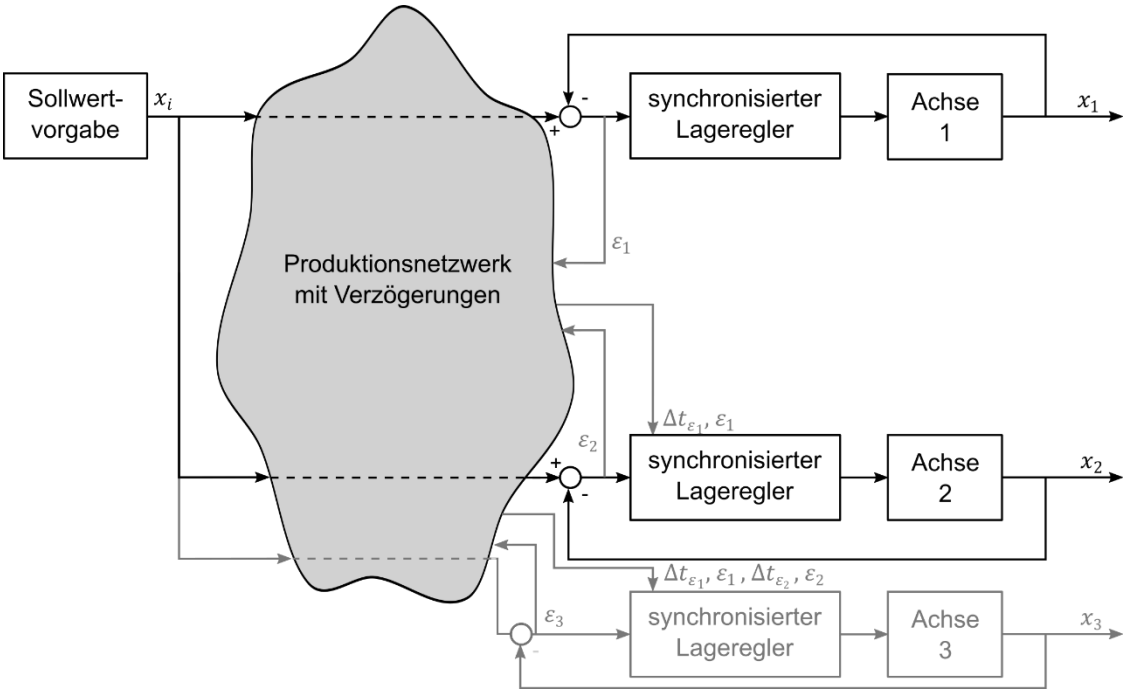

Abbildung 2-14: Blockschaltbild der über das Netzwerk synchronisierten Lageregler nach (Xu et al. 2017, S. 118).

Molano verwendet den verketteten Leader-Follower-Ansatz von Xu et al. bei der Ansteuerung von Achsen mit Antriebsverstärkern verschiedener Hersteller und erweitert das Reglerkonzept um eine Vorsteuerung (Molano et al. 2018). Ein ähnliches Konzept verfolgt (Yook et al. 2002). Zusätzlich zu den über das Netzwerk kommunizierten Soll- und Istwerten implementiert Yook Zustandsschätzer in den

einzelnen Achsen, die über die Netzwerkkommunikation aktualisiert werden. Dadurch verringert sich die Kommunikationslast und das System kann auch bei kurzzeitigen Ausfällen oder Verzögerungen über den Zustandsschätzer stabil weiterbetrieben werden.

2.5 Fazit aus der Betrachtung der Grundlagen sowie des Stands der Technik und Forschung

Nach Darstellung des Stands der Technik für positionierende Achskomponenten und Betrachtung der aktuell eingesetzten Methoden und Systeme zur Bewegungssynchronisierung lässt sich festhalten, dass die synchrone Bewegungssteuerung entsprechend der Anforderungen aus Kapitel 1.3 bereits in unterschiedlicher Art und Weise für zentral gesteuerte Systeme umgesetzt ist. Systeme ohne zentrale Instanz, wie die Leader-Follower-Architektur oder das Cross-Coupling setzen, entgegen der Anforderungen für diese Arbeit, entweder eine explizite Hierarchie um und/oder sind nicht beliebig erweiter- oder rekonfigurierbar.

Verschiedene Forschungsansätze der vergangenen Jahre betrachten die Bewegungssynchronisierung unter Einsatz dezentral gesteuerter Systeme. Entweder wird ein sehr umfangreiches Systemwissen vorausgesetzt, oder es werden ebenfalls Hierarchien zwischen den dezentralen Steuerungen definiert, was die Rekonfiguration deutlich erschwert. Der Forschungsansatz von Lesi (Lesi 2020) vermeidet Hierarchien und konzentriert sich auf die Verteilung von möglichst viel Steuerungsfunktionalität auf die dezentralen LLC, stellt aber im Ergebnis nur eine exakte Synchronisierung des Bewegungsbeginns sicher, ohne die Synchronisierung während der Bewegung zu überprüfen und ggf. Bahnabweichungen zur Laufzeit auszuregeln.

Als Forschungslücke ist also die Notwendigkeit einer synchronen Bewegungssteuerung zur Verwendung auf und mit dezentralen Miniatursteuerungen festzuhalten, die bisher weder kommerziell noch in Forschungsszenarien vorhanden ist. Diese muss so entwickelt werden, dass

- nur wenig Systemwissen vorausgesetzt werden muss,
- keine Hierarchiedefinition benötigt wird, die Rekonfigurationen erschwert,
- laufend während der Bewegung eine Synchronisierung sichergestellt wird.

3 Systemarchitekturen zur Bewegungssynchronisierung

Das in Kapitel 1.3 vorgestellte Zwischenziel 1 einer Systemarchitektur zur Integration von Bewegungssynchronisierung auf Miniatursteuerungen soll im Nachfolgenden erläutert werden. Dazu wird eine Begriffsklärung zu dezentralen und verteilten Systemarchitekturen durchgeführt. Anschließend werden Randbedingungen des Anwendungsfalls festgehalten. Nachfolgend werden verschiedene Ansätze erläutert und mit dem Stand der Technik und Forschung in Kontext gesetzt. Abschließend folgen eine Bewertung der Ansätze anhand der Anforderungskriterien sowie die Ableitung der weiteren Vorgehensweise.

Zielsetzung:

Auswahl einer Systemarchitektur zur Integration von Bewegungssynchronisierung auf Miniatursteuerungen (Zwischenziel 1)

Vorgehen:

Ableiten von Systemkonzepten

3.1 Begriffsklärung zu zentralen, parallelen, dezentralen und verteilten Systemarchitekturen

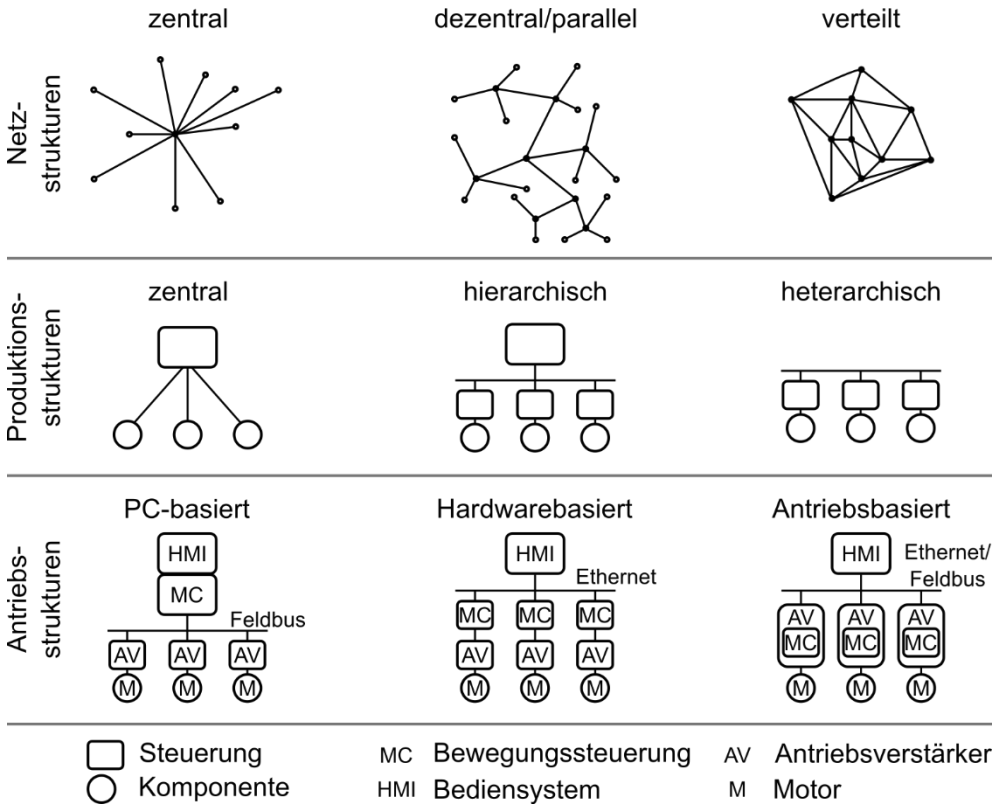

Abbildung 3-1: Begriffsklärung der Strukturen von Systemarchitekturen: Netzstrukturen nach (Özbek 2019), Entsprechung der Produktionsstrukturen nach (Bussmann et al. 2004, S. 41), Entsprechung der Strukturen für die Ansteuerung von bewegten Achsen nach (Seitz 2015, S. 185).

Im für diese Arbeit betrachteten Anwendungsfall von positionsgesteuerten Achsen mit Miniatursteuerungen nach den obigen Kriterien wird eine gleichzeitige Interaktion dezentraler Komponenten angestrebt. Die Begrifflichkeiten zentral, dezentral und verteilt werden in der Literatur je nach Fachrichtung und Anwendergruppe unterschiedlich verwendet, s. Abbildung 3-1.

Vor allem die Begriffe dezentral und verteilt werden in mehrdeutigen, sich teilweise widersprechenden Bedeutungen verwendet (Özbek 2019), weshalb in dieser Arbeit die Begriffe im nachfolgenden Verständnis getrennt werden[2], s. Abbildung 3-2:

- Zentral (englischer Begriff: central[3]):
  Es gibt eine einzelne Einheit C, die mit allen weiteren gleichartigen Einheiten $a_1 \dots a_n$ verbunden ist. Alle Berechnungen werden auf der zentralen Einheit ausgeführt. Entscheidungen werden an die weiteren Einheiten kommuniziert. Es gibt keine Kommunikation zwischen den Einheiten $a_1 \dots a_n$.
  Anwendungsbeispiel: Eine zentrale Steuerung berechnet alle Achssollwerte und erhält die Istdaten der untergeordneten Achsen. Bei Bedarf passt die Steuerung die zentralen Vorgabewerte aufgrund der Istdaten aller Achsen an.
- Parallel (englischer Begriff: decentral/parallel; Bedeutung nach dem Verständnis von u.a. (Ishida 1994; Rinschede 1995, S. 66; Fowler et al. 2002):
  Es gibt mehrere gleichartige Einheiten $a_1 \dots a_n$. Gleichartige Berechnungen werden auf allen Einheiten durchgeführt. Entscheidungen werden in und für die einzelnen Einheit getroffen. Es gibt allerdings keine Kommunikation zwischen den Einheiten.
  Anwendungsbeispiel: Alle Achssollwerte werden von mehreren Antriebsverstärkern eigenständig aufgrund von MC-Funktionen berechnet und umgesetzt. Es findet während der Berechnung aber keine Betrachtung von weiteren Achsen im Verbund statt, da keine Kommunikation zwischen den Antriebsverstärkern existiert.
- Berechnungsverteilt (englischer Begriff: distributed (Cormen et al. 2009, S. 781)):
  Es gibt mehrere gleichartige Einheiten $a_1 \dots a_n$. Alle Einheiten kommunizieren unmittelbar miteinander. Berechnungen werden in Teilprobleme von einzelnen Einheiten gelöst, wobei alle notwendigen Kontextinformationen bereits vorliegen. Ergebnisse aller Teilprobleme werden kommuni-

[2] Die in den Referenzen verwendeten Begriffe werden im restlichen Dokument entsprechend der von den Autoren beabsichtigte Bedeutung in ihren zitierten Aussagen nach der nachfolgenden Definition angepasst, sofern die Notwendigkeit hierfür bestehen sollte.

[3] Der hier beschriebene zentrale Fall wurde in den neunziger Jahren bereits ebenfalls als „distributed interpolation“ (verteilte Interpolation) bezeichnet, da die Interpolation zwischen Stützpunkten in achsbezogenen Koordinaten direkt an der Lageregelung statt in der zentralen CNC auf Bahnkoordinaten durchgeführt wurde (Decotignie 1991, S. 772–773). Das Konzept der Verschiebung von Berechnungsintelligenz in Richtung des Antriebsverstärkers wurde in allen weiteren bekannten Quellen jedoch als dezentral (decentralized) beschrieben, weshalb es sich bei der Verwendung des Begriffes verteilt (distributed) hier um einen Einzelfall zu handeln scheint.

ziert und Entscheidungen aufgrund dieser Ergebnisse durch und für die einzelnen Einheit getroffen.
Anwendungsbeispiel: Die Achssollwerte werden abschnittsweise von mehreren Steuerungen vorberechnet, danach an alle Achsen kommuniziert (verteilt) und umgesetzt.

- Verteilt (englischer Begriff: decentral; Bedeutung nach dem Verständnis von u.a. (Monostori et al. 2006) bzw. distributed; Bedeutung nach dem Verständnis von u.a. (Ishida 1994, S. 61)):
  Es gibt mehrere gleichartige Einheiten $a_1 \dots a_n$. Alle Einheiten kommunizieren mittelbar oder unmittelbar miteinander. Berechnungen werden entweder in gleicher Weise auf allen Einheiten durchgeführt oder es werden Teilprobleme von spezifischen Einheiten gelöst, wobei in beiden Fällen die kommunizierte Information der anderen Einheiten in Betracht gezogen werden. Entscheidungen werden in und für die einzelnen Einheiten getroffen.
  Anwendungsbeispiel: Alle Achssollwerte werden eigenständig im Antriebsverstärker berechnet. Es werden die Stati mit weiteren Achsen des Verbunds ausgetauscht und bei Bedarf die Vorgabewerte aufgrund der vorhandenen Informationen angepasst.

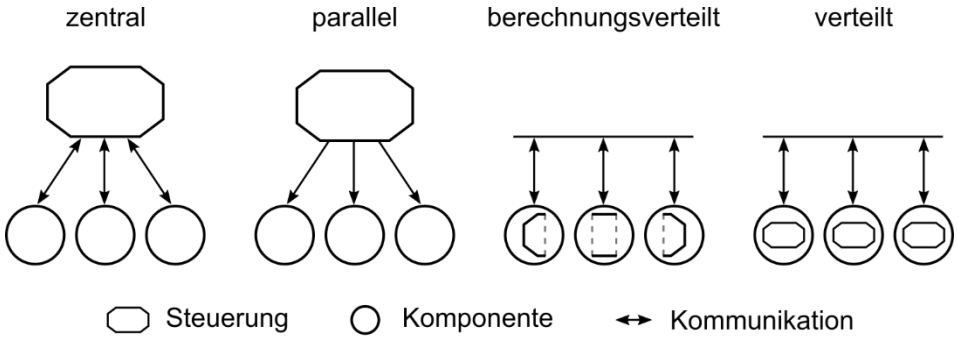

Abbildung 3-2: Verwendung der Begriffe von zentral, parallel, berechnungsverteilt und verteilt in dieser Arbeit.

Die verteilte Produktionssteuerung hat nach (Leitão 2009, S. 981) fünf Charakteristiken, die verteilte Lösungen von einer zentralen Steuerung unterscheiden. Speziell die Eigenschaften unter Punkt 5 werden durch (Lesi et al. 2019, S. 711) umfangreich erörtert:

1. Es wird das komplexe Gesamtproblem in mehrere Unterprobleme unterteilt, die den verteilten Steuerungseinheiten zugeschrieben werden.
2. Jede verteilte Steuerungseinheit hat eigenständige Ziele, Wissen und Fähigkeiten, die gekapselt werden, aber keine Einsicht über das globale Gesamtsystem.
3. Entscheidungen, die das Gesamtsystem betreffen, werden von mehr als einer Steuerungseinheit in kollaborativer Zusammenarbeit getroffen.
4. Die verteilten Steuerungseinheiten können mit einzelnen Automatisierungskomponenten verknüpft sein.
5. Verteilte Steuerungseinheiten sind rekonfigurierbar, robust, schnell einsetzbar, lernfähig und wiederverwendbar.

Auf dieser Basis und den genannten Charakteristiken soll eine geeignete Systemarchitektur zur Bewegungssynchronisierung gefunden werden. Wie aus den Ansätzen in Stand der Technik und Forschung bereits sichtbar wurde, befinden sich Ansätze zur Synchronisierung in einem Spannungsfeld zwischen zentralen, parallelen und verteilten Ansätzen.

In zentralen Konzepten werden häufig modellbasierte Lösungen erarbeitet, die Modelle und Stati aller Komponenten zentral sammeln und aufgrund der Gesamtsituation Entscheidungen und Vorgaben beschließen und durchsetzen. Grundvoraussetzung hierfür ist, dass das Systemwissen vorhanden und modellierbar ist, sowie dass alle Komponenten in ausreichender Taktung aktualisierte Stati erhalten und zurücksenden. Änderungen in zentralen Systemen sind selten vorgesehen, die Ansätze werden meist spezifisch für einen Anwendungsfall gelöst.

Parallele Ansätze trennen die Ansteuerung zu einem frühen Zeitpunkt in der Prozesskette. Aufgrund fehlender Kopplung der parallelen Ansätze kann aber speziell in Fehlerfällen das Gesamtsystem nicht zufriedenstellen reagieren.

Verteilte Ansätze hingegen erfahren einen Aufstieg, seit auch Komponenten mit digitalen Steuerungen ausgerüstet werden. Speziell durch die Digitalisierung und Industrie 4.0 Aktivitäten steigt die Intelligenz der Komponenten bzw. verbauter Steuerungen stark an. Der verteilte Ansatz benötigt eine funktionierende Kommunikation und neuartige Ansätze in Regelprinzipien, da kein zentrales Systemwissen vorliegt und Entscheidungen lokal mit eingeschränktem Wissen bzw. Istwerten aus der Umgebung getroffen werden müssen. Die Vorteile liegen vor allem in der Rekonfigurierbarkeit und Wiederverwendbarkeit von verteilten Systemansätzen, da sie nicht auf eine bestimmte Anzahl von Teilnehmern oder Vorbedingungen beschränkt sind, so wie der Möglichkeit zu echter Parallelität der Steuerungsabläufe (Hirsch 2010, S. 59). Ein erhöhtes Kommunikationsaufkommen im Produktionsnetzwerk (Correll et al. 2013) sowie die Herausforderung bisher zentrale Steuerungskonzepte in eine meist komplexe, verteilte Systemarchitektur zu integrieren sind Nachteile des verteilten Systemkonzeptes (Hirsch 2010, S. 59).

3.2 Gegenüberstellung von Systemkonzepten zur synchronen Bewegungsteuerung auf Miniatursteuerungen

Zur Umsetzung der synchronen Bewegungssteuerung nach Kapitel 2.2.3 auf dezentralen Miniatursteuerungen soll eine geeignete Systemarchitektur gefunden werden. Dazu sollen Konzepte vorgestellt werden, die folgende Anforderungskriterien (s. Kapitel 1.3) erfüllen können:

1. Umsetzung einer Bewegungssynchronisierung:
   Umsetzung der vorgegebenen Bahn mit möglichst geringer Bahnabweichung, inklusive Konzepte zur Korrektur von Abweichungen, die im Prozessablauf durch veränderliche Achsdynamiken, Lasten und weiteren Komplikationen verursacht werden
2. Einklang mit dem DEVEKOS-Systemverständnis:
   Durchführung möglichst aller komponentenbezogenen Berechnungen in der Miniatursteuerung, ohne Definition einer Hierarchie innerhalb der

Komponentenverbünde. Die Systemarchitektur soll mit möglichst geringen Erweiterungen am DEVEKOS-Systemkonzept umsetzbar sein und sich an der verteilten Systemarchitektur mit Fähigkeitsschnittstellen der Komponenten nach Konzept eines HMS orientieren.

3. Gleichwertige Verteilung der Rechenleistung:
   Verwendung der vorhandenen Rechenleistung über alle beteiligten Komponenten hinweg. Die Komponenten sollen möglichst in gleicher Höhe mit Rechenlast beaufschlagt sein.
4. Angemessene Aufwände bei Rekonfigurationen:
   Das Entfernen oder Hinzufügen von Komponenten sowie die Rekonfiguration des Aufbaus der vorhandenen Komponenten soll möglichst geringe Aufwände in der Rekonfiguration des Steuerungssystems verursachen.
5. Angemessene Realisierungskomplexität:
   Die Realisierungskomplexität der Implementierung soll für die spätere Übertragbarkeit auf die Miniatursteuerungen sowie in industrielle Anwendungen und Produkte möglichst geringgehalten werden.

Allgemeine Anforderungen an ein solches Lösungskonzept wurde bereits in (Dripke et al. 2017) vorgestellt. Es gelten weiter folgende Annahmen für den Anwendungsfall bzw. das dezentrale Steuerungssystem:

- Es existieren 1 bis n Achsen. Diese können unterschiedliche Dynamikgrenzen besitzen, von verschiedenen Herstellern stammen oder unterschiedliche Antriebsarten verwenden. In ihrer Reaktion auf Sollwertvorgaben soll aber ein gleichartiges Folgeverhalten erwartet werden können.
- Die Achsen bewegen sich in einem gemeinsamen Arbeitsraum. Die Achsen sind rotatorisch oder linear verfahrend. Der kinematische Zusammenhang im Mehrachssystem wird für alle Achsen als bekannt vorausgesetzt.
- Jede Achse kennt ihre eigene Achsposition. Alle Achsen besitzen eine eigene Miniatursteuerung. Alle Achsen sind mit dem gemeinsamen Kommunikationsnetzwerk verbunden.
- Die Sollwertvorgabe der Bahn wird außerhalb des Systems generiert und zur Ausführungszeit als Gesamtziel an den Achsverbund kommuniziert. Abweichungen von der vorgegebenen Bahn aufgrund von getragenen Lasten, falsch hinterlegter Dynamik, erhöhter Ausführungszeit der Algorithmen oder Ausfälle der Kommunikation sind zu berücksichtigen.
- Die Dynamik der Achsen ist aufgrund von Last oder anderen Einwirkungen zu bestimmten Zeiten veränderlich. Ohne Last zeigen die Achsen ein gleichartiges Dynamikverhalten. Maximale Dynamiken können jedoch voneinander abweichen.

Entsprechend der Begriffsdefinitionen aus Kapitel 3.1 wurden verschiedene Systemkonzepte für den konkreten Anwendungsfall der synchronen Bewegungssteuerung auf Basis von Konzepten aus dem Stand der Technik oder Forschung entwickelt. In Kapitel 3.3 werden die Konzepte anhand des Kriterienkatalogs überprüft.

3.2.1 Konzept mit zentraler Bewegungssynchronisierung (zentrale Systemarchitektur)

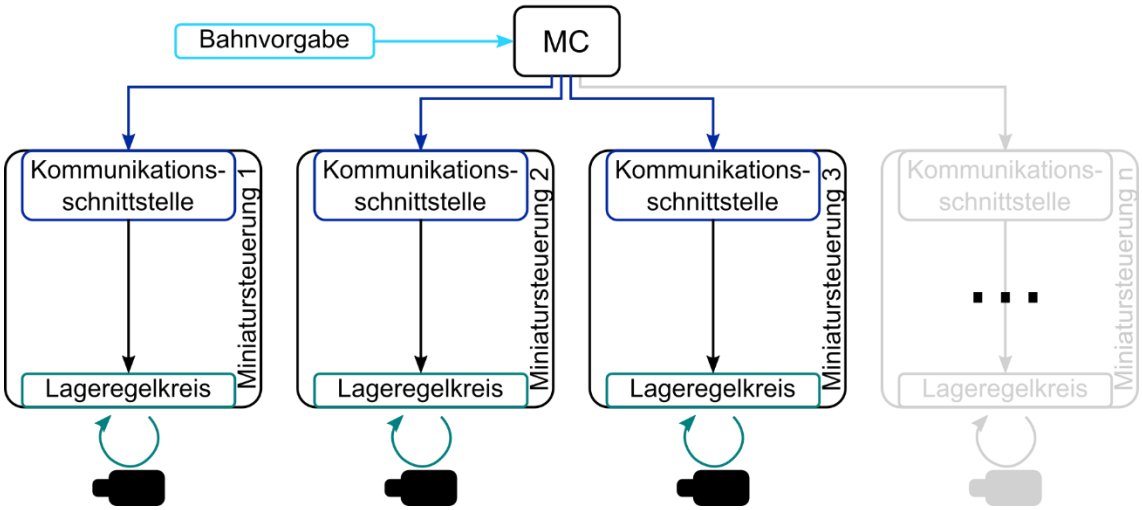

Abbildung 3-3: Systemarchitektur mit zentraler Bewegungssynchronisierung.

Stand der Technik zur Steuerung von Achsverbünden ist die Verwendung einer zentralen MC, s. Abbildung 3-3. Eine zentrale MC verwendet eine Vielzahl von Informationen der Achsen für die Berechnung der Vorgabewerte, wie die Verfahrwege, Referenzpositionen, Endlagen, maximale Geschwindigkeits- und Beschleunigungswerte der Achsen (Lesi et al. 2019).

Umsetzung einer Bewegungssynchronisierung:

Das Kriterium der synchronen Bewegungssteuerung ist durch eine MC in jedem Fall erfüllt und deckt sich mit dem bereits vorhandenen Funktionsumfang.

Angemessene Aufwände bei Rekonfigurationen:

Rekonfigurationen sind mit einem hohen Aufwand bei der Anpassung der zentralen Parametersätze verbunden (Lesi et al. 2016). Anpassungen der Parametersätze reichen jedoch in vielen Fällen nicht aus, um Änderungen im System widerzuspiegeln. Kommerzielle MCs lassen als geschlossene monolithische Systeme keine strukturellen Änderungen (Konfiguration) durch den Anwender zu (Kircher 2011, S. 20). Eine selbstadaptierende, „plug-and-play“-fähige Steuerung wurde in der Dissertation von Kircher geprüft und lässt sich nur unter bestimmten Einschränkungen umsetzen. So müssen die zu integrierenden Achsmodule bereits vorparametriert sein und ein spezifisches Informationsmodell mitliefern. Die Steuerung hingegen muss die konfigurierbaren Elemente offenlegen und in ihrem Informationsmodell ablegen (Kircher 2011, S. 130). Eine Rekonfiguration ist unter diesen Vorbedingungen für Steuerung und Maschinenmodule möglich.

Angemessene Realisierungskomplexität:

Ebenso ist die Realisierungskomplexität akzeptabel, wenn auch eine modulbasierte Rekonfigurationsschnittstelle nach Vorschlag von (Kircher 2011) gewisse Aufwände benötigen würde.

Einklang mit dem DEVEKOS-Systemverständnis / Gleichwertige Verteilung der Rechenleistung:

Anders sieht es jedoch mit den weiteren Anforderungen nach Kapitel 3.2 aus. Die MC als eine weitere übergeordnete Einheit ohne direkte Zuordnung zu einer spezifischen Komponente im System zu integrieren bricht mit dem Systemgedanken aus DEVEKOS, da die Steuerungslogik direkt über die an den Komponenten bereitgestellten Miniatursteuerungen angeboten und ausgeführt werden sollte. Eine gleichwertige Verteilung der Rechenleistung auf alle Miniatursteuerungen wird ebenfalls nicht erreicht, wenn eine zentrale MC alle Berechnungen durchführt und die Miniatursteuerungen lediglich Vorgabewerte umsetzen. Die Aufwände einer Rekonfiguration wurden oben bereits als Engpass bestehender Systeme genannt. Auch mit den vorgeschlagenen Erweiterungen durch (Kircher 2011) sind strukturelle Anpassungen nur eingeschränkt möglich und können daher absehbar nicht alle Rekonfigurationsszenarien abdecken.

Varianten der Umsetzung:

Ein möglicher Verzicht auf die getrennte Hardware einer zentralen Steuerung kann durch die Integration der Bewegungssteuerung in die Miniatursteuerungen aller interpolierenden Achskomponenten erreicht werden. Aufgrund steigender Kapazität der miniaturisierten Elektronik kann bereits heute MC-Funktionalität in kleinbauende leistungsfähige Systeme verbaut werden (Grigoriev et al. 2016; FANUC 2018). Eine mögliche Herangehensweise wäre die Implementierung einer MC in jeder Miniatursteuerung von interpolierenden Achskomponenten. Dann wäre jede einzelne der Miniatursteuerungen in der Lage den Achsverbund zu koordinieren. In einer parallelen Ausführung durchläuft jede Miniatursteuerung mit MC in Referenz zur vorgegebenen Bahn die Prozesskette der MC-Funktionalitäten und verwendet nur diejenigen Ergebnisse, die als Vorgabe für die spezifische Achse benötigt werden. Sofern alle Komponenten identische MC-Funktionalitäten implementiert haben und synchron ausgeführt werden, sind die Ergebnisse in allen Systemen vergleichbar und eine Durchführung einer bahntreuen Bahn ist möglich. Wenn zur Laufzeit Abweichungen von den angenommenen Dynamiken und Folgeverhalten auftreten kommt es allerdings ebenso wie im Falle von Lasten, Änderungen in der Dynamik oder bei unterschiedlichen MC-Implementierungen oder -Laufzeiten zu Bahnabweichungen.

3.2.2 Konzept mit leaderbasierter Bewegungssynchronisierung (parallele Systemarchitektur)

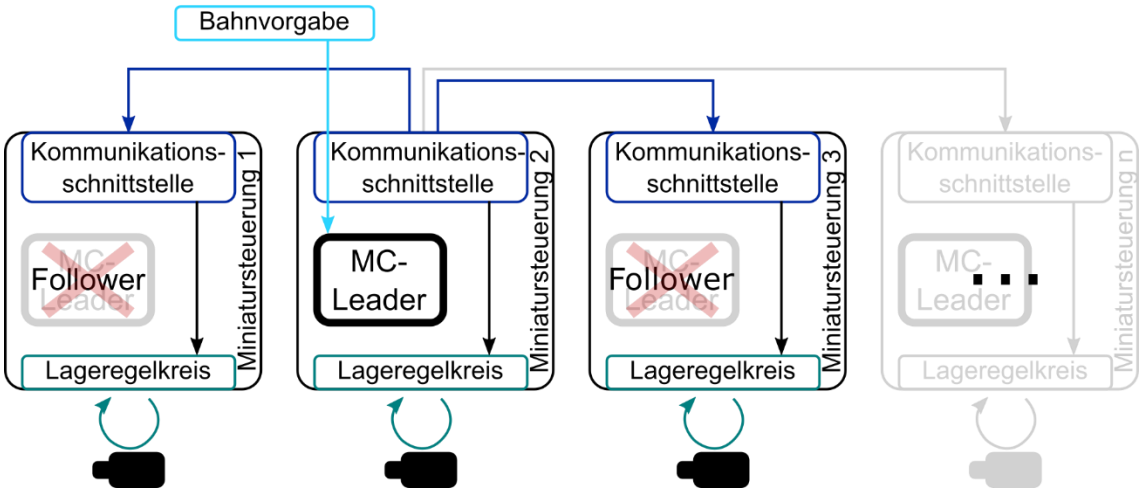

Abbildung 3-4: Systemarchitektur mit leaderbasierter Bewegungssynchronisierung.

Statt mehrere MC-Steuerungen dieselben Berechnungen ausführen zu lassen, kann auch in einer einzelnen der Miniatursteuerungen im Achsverbund die Berechnung durchgeführt werden und die weiteren Miniaturkomponenten als Follower, also reine Antriebsverstärker, betrieben werden, s. Abbildung 3-4.

Umsetzung einer Bewegungssynchronisierung:

In dieser leaderbasierter Steuerung kann allerdings nur bei stetiger Rückkommunikation der Follower zum Leader eine Bahntreue ähnlich der Verwendung einer externen MC erwartet werden.

Angemessene Realisierungskomplexität:

Das Leader-Follower-Konzept kann durch die Verwendung etablierter Feldbusprotokolle mit geringen Anpassungen der Kommunikationsschnittstellen durchgeführt werden.

Einklang mit dem DEVEKOS-Systemverständnis:

Dass keine externe Hardware verwendet wird, sondern die Fähigkeit innerhalb einer Komponente zur Verfügung steht, kommt dem DEVEKOS-Systemgedanken entgegen. Eine standardisierte Darstellung von Fähigkeiten, die die Komplexität der MC-Funktionalitäten nach außen kapseln, fällt aufgrund des sehr unterschiedlichen Leistungsumfangs verschiedener MCs allerdings schwer. Sollten Komponenten von verschiedenen Herstellern mit unterschiedlichen Implementierungen von MC-Funktionalitäten ausgestattet werden, kann die Leistung des Gesamtproduktionssystems je nach Wahl des MC-Leaders unterschiedlich ausfallen.

Gleichwertige Verteilung der Rechenleistung:

Die gleichwertige Verteilung der Rechenleistung ist nicht gegeben, da die Folgerachsen lediglich als Antriebsverstärker betrieben werden und nur die Leaderachse Berechnungen durchführt. Die prototypische Miniatursteuerung besitzt

weiter nur begrenzten Speicherplatz und Rechenleistung, der für eine MC unter Umständen nicht ausreichen könnte. Der benötigte Speicherplatz und die Rechenleistung ist in vielen Fällen von der Implementierung der MC bzw. ihrem Funktionsumfang abhängig und kann dadurch für verschiedene Hersteller ebenfalls unterschiedliche Voraussetzungen notwendig machen.

Angemessene Aufwände bei Rekonfigurationen:

Da eine MC als Knowhow-intensives Produkt hohe Lizenzkosten hat, ist eine Instanziierung auf allen Komponenten wirtschaftlich nicht sinnvoll, wenn nur jeweils eine Komponente im Verbund aktiv genutzt wird. Das Engineering bei einer Rekonfiguration ist ähnlich umfangreich, wenn nicht durch die notwendigen Eingriffe auf mehreren Systemen sogar höher, wie bei einer externen, zentralen MC einzuschätzen.

Varianten der Umsetzung:

Da vor allem in bestehenden Automatisierungssystemen untergeordnete Fertigungszellen meist bereits mit Bewegungssteuerungen ausgestattet sind, ist es ebenso möglich, diese Bewegungssteuerungen bis auf eine Ausnahme als Followerkomponenten zu definieren. Eine Leaderkomponente kann damit die gekoppelte Ausführung aller Bewegungssteuerungen vorgeben. Der Stand der Technik zeigt hier bereits verschiedene Umsetzungen. Grigoriev (Grigoriev et al. 2016) stellt ein dezentrales Steuerungskonzept für Produktionssysteme vor, die Komponenten mit großer räumlicher Entfernung beinhalten. Es werden räumlich nahe Automatisierungskomponenten über einen Schaltschrank in einer hierarchischen Zwischenebene wie einen Antriebsverstärker gekoppelt. Diese Zwischenebene ist über das Netzwerk an eine zentrale CNC angeschlossen und verteilt Vorgabewerte an untergeordnete Komponenten weiter. Er geht zusätzlich darauf ein, dass die dezentral gruppierten Komponenten teilweise von unterschiedlichen Herstellern und damit mit unterschiedlichen Kommunikationsprotokollen ausgestattet sein können, was auch (Molano et al. 2018) thematisiert. Die zentrale CNC muss daher verschiedene Kommunikationsschnittstellen gleichzeitig anbieten um alle Verbünde integrieren zu können. Dafür definiert Grigoriev Abstraktionsebenen für die Antriebsverstärker, SPS, Kommunikationskanäle zu Eingabegeräten sowie den Programmeingaben (Grigoriev et al. 2013, S. 297).

Wenn mehrere Achsverbünde durch Buskoppler oder einen zentralen Antriebsverstärker gekoppelt werden, fällt diesem System die Funktion des Leaders für diesen Achsverbund zu. Sollen mehrere dieser Leader in einem Produktionsnetzwerk Daten austauschen, müssen Multimastermechanismen eingesetzt werden um die Hierarchie im Produktionsnetzwerk zwischen den Leadern festzulegen (Beckhoff 2010). Dieser Fall tritt auch auf, wenn mehrere Steuerungen in einem Netzwerk miteinander gekoppelt werden sollen. Auch in diesem Fall wird festgelegt, welche Steuerung die Leaderfunktion übernimmt und welche Steuerung die Vorgaben des Leaders in den internen Berechnungen verwendet. Es werden auch die Verfahrsätze zwischen den Steuerungen synchronisiert (Bauder 1992), s. Abbildung 3-5. Eine Umsetzungskonzept der leaderbasierten Bewegungssynchronisierung mit Miniatursteuerungen wurde in (Dripke et al. 2018) vorgestellt.

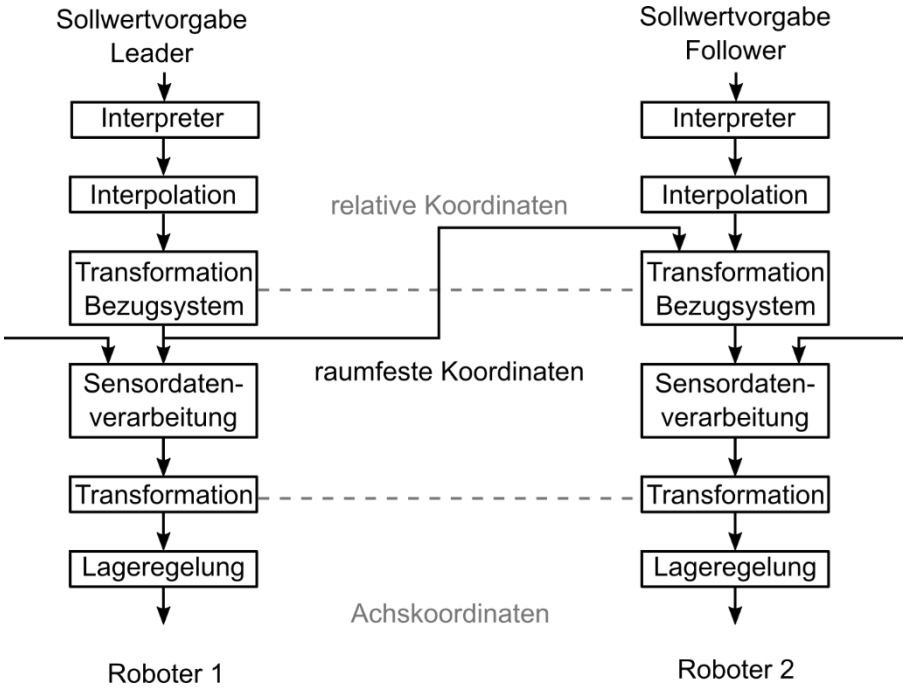

Abbildung 3-5: Synchronisierung der Verfahrsätze zwischen zwei Steuerungen, in diesem Beispiel für Roboter, nach (Bauder 1992).

Vorteile dieser Systemarchitektur sind, dass keine zusätzliche Komponente benötigt wird, das Leader-Follower-Verfahren bereits in vielen Anwendungen etabliert ist und die Anbindung weiterer Follower meist nur geringe Aufwände in der Konfiguration des Engineerings benötigt. Allerdings werden nur jeweils die Leaderkomponenten für die Berechnung und Durchsetzung der Vorgabewerte verwendet und Miniatursteuerungen in den Followerkomponenten bleiben ungenutzt. Weiter muss die Wahl des Leaders so durchgeführt werden, dass alle Followerkomponenten der vorgegebenen Dynamik folgen können oder bestenfalls noch dynamischer sind, um auf die Vorgabewerte zeitnah zu reagieren und Schleppfehler zu vermeiden. Dafür muss ein hohes Vorwissen über das System bekannt sein. Vor allem bei der Rekonfiguration und der Einkopplung weiterer vorher unbekannten Followerkomponenten kann es daher zu Bahnabweichungen kommen, die innerhalb der Leader-Follower-Struktur nicht rückgekoppelt und ausgeregelt werden.

3.2.3 Konzept mit partitionierter Bewegungssynchronisierung (berechnungsverteilte Systemarchitektur)

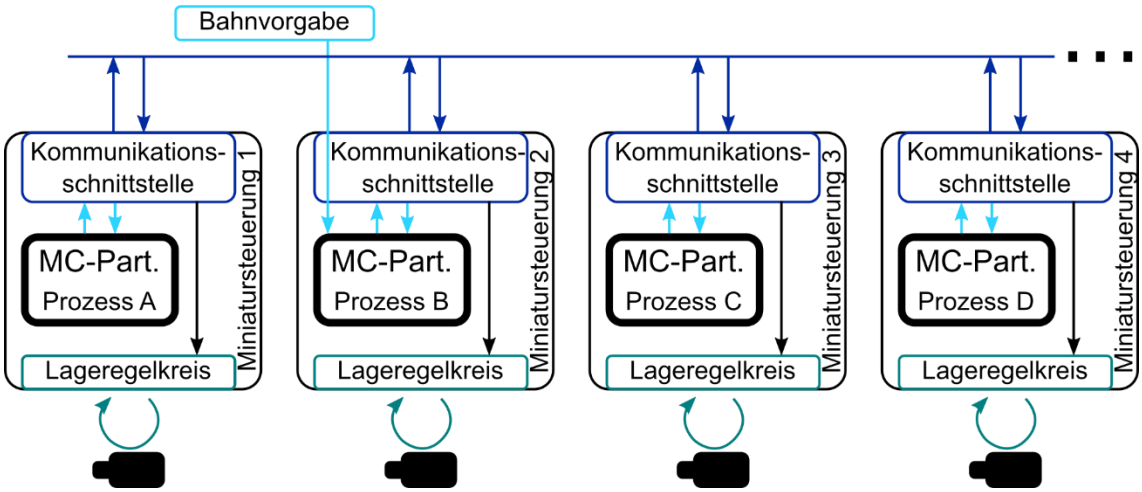

Abbildung 3-6: Systemarchitektur mit partitionierter Bewegungssynchronisierung.

Im vorhergehenden Kapitel war der komplette Berechnungsablauf der Bewegungssynchronisierung auf einer als Leader definierten Komponente durchgeführt worden. Es stellt sich die Frage, ob die weiteren Miniatursteuerungen der Follower ebenfalls zur Berechnung von Teilschritten und –prozessen nach einem berechnungsverteilten Ansatz herangezogen werden können, s. Abbildung 3-6. Die Partitionierung der Bahngenerierung zur parallelen Berechnung auf verschiedenen Kernen in einem Multicoresystem ist Fokus von aktuellen Forschungsarbeiten. In einer Fallstudie wurde die Berechnungen von Funktionsauswertung, Geschwindigkeit und Beschleunigungsverlauf sowie der Bogenlänge eines B-Splines gezeigt (Kaiser et al. 2015). Neben der Partitionierung der Splineberechnung wurden auch weitere CNC-Funktionen wie der Interpreter, die Look-Ahead-Funktion oder die Dynamikberechnung als parallelisierbare Prozesse der CNC identifiziert (Keinert et al. 2014).

Statt wie (Lesi et al. 2019) die CNC-Funktionen in ihrem Ablauf ab einem definierten Punkt parallel auf der LLC durchzuführen, werden die Prozesse an verschiedene Kerne zugewiesen und nach der Berechnung wieder synchronisiert. Prozesse, die weiter partitionierbar sind können auch auf zwei oder mehr Kernen verteilt werden. Teilweise können durch überlappende Datensätze Berechnungen in mehrere Prozesse zerteilt und abschließend im Masterthread an den überlappenden Datenpartitionen synchronisiert werden (Keinert 2020), s. Abbildung 3-7.

Einklang mit dem DEVEKOS-Systemverständnis / Gleichwertige Verteilung der Rechenleistung:

Der Vorteil von verteilten Ansätzen ist die Ausnutzung der verfügbaren Speicher- und Rechenkapazitäten der verfügbaren Steuerungen.

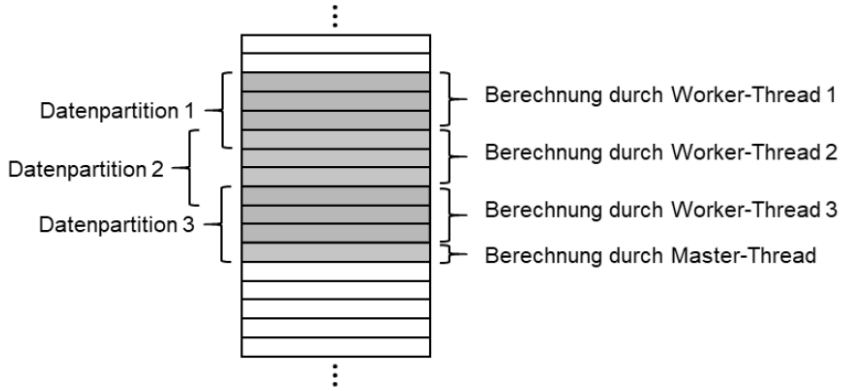

Abbildung 3-7: Überlappende Datenpartitionen bei der Verteilung der Berechnungsprozesse auf mehreren Kernen (Keinert 2020, S. 111).

Umsetzung einer Bewegungssynchronisierung:

Eine Bewegungssynchronisierung ist bei korrekter Umsetzung identisch zu einer zentralen CNC möglich. Die Betrachtungen von Keinert zeigen, dass Prozesse in der CNC partitionierbar sind. Dedizierte Prozesse können über eine abschließende Synchronisierung der Ergebnisse sogar auf mehreren Recheneinheiten verteilt werden.

Angemessene Aufwände bei Rekonfigurationen:

Eine Rekonfiguration des Systems könnte daher solche Prozesse auch auf neu eingekoppelte Achsen übertragen oder aber Prozesse nach Bedarfen neu verteilen.

Angemessene Realisierungskomplexität:

Es bleibt allerdings unklar, wie in der Anwendung eine Rekonfiguration auf die Partitionierung übertragen wird, da bisher nur statische Partitionierung durchgeführt und erprobt wurde. Viel grundlegender ist jedoch das Problem der Verteilung von solchen partitionierten Prozessen über ein Netzwerksystem. Die vorliegende Partitionierung wurde für die Verwendung auf einem einzelnen Rechensystem mit mehreren Kernen erarbeitet. Dort sind die Konzepte von geteiltem und verteiltem Speicher (shared/distributed memory) bereits umfangreich erprobt und werden in verschiedenen Applikationen angewendet (Cormen et al. 2009). Speziell die häufige Synchronisierung über den verteilten Speicher der Prozesse ist eine Grundvoraussetzung in der Parallelisierung. Diese über ein Produktionsnetzwerk abzuwickeln bedeuten Zeitverzögerungen und eine hohe Netzwerklast, vgl. (Cormen et al. 2009, S. 777), sodass die Übertragung des Konzeptes von Multicoresystemen auf verteilte, über ein Produktionsnetzwerk gekoppelte Miniatursteuerungen eine sehr hohe Realisierungskomplexität mit sich bringt.

Varianten der Umsetzung:

Neben der Partitionierung der Bahnberechnung werden auch Konzepte zur Umsetzung der inversen Kinematik in verteilten Steuerungen vorgestellt. Der Ansatz sieht für jedes Gelenk eine Steuerungseinheit nach IEC 61499 vor. Die Autoren kombinieren den Ansatz des Dynamic Programming (DP; dynamische Programmierung: Aufteilen eines Optimierungsproblems in Teilprobleme) und des Cyclic

Coordinate Descent (CCD; zyklische Koordinatenannäherung: iterative Annäherung an die Zielposition durch alle an der Bewegung beteiligten Achsen), in dem sich schrittweise die einzelnen Achsen in ihren Achskoordinaten den Vorgabewerten nähern (Steinegger et al. 2016, S. 1).

3.2.4 Konzept mit agentenbasierter Bewegungssynchronisierung (verteilte Systemarchitektur)

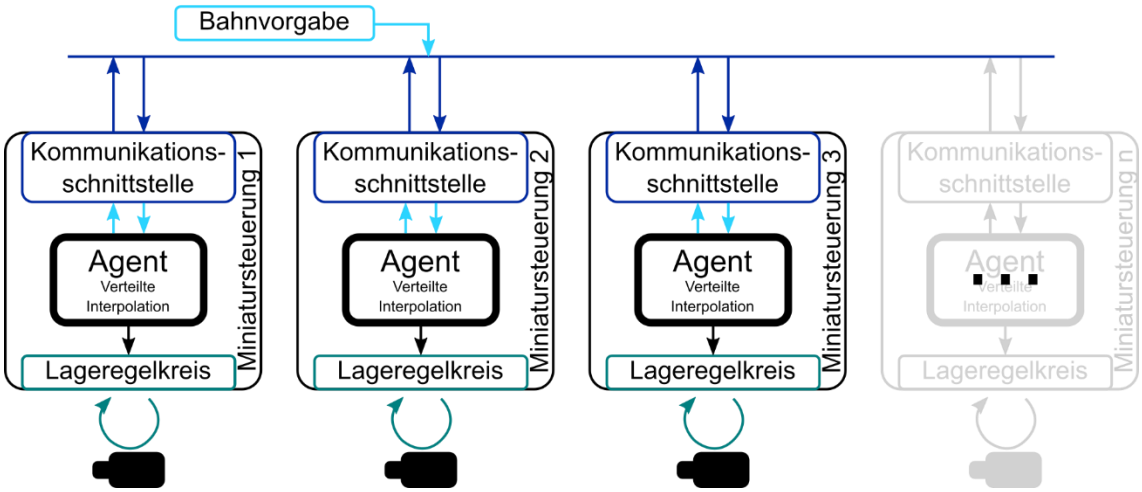

Abbildung 3-8: Systemarchitektur mit agentenbasierter Bewegungssynchronisierung.

Um die Realisierungskomplexität zu begrenzen und keine (Neu-)Verteilung der Rechenprozesse bei einer Rekonfiguration zu erfordern, wird ein agentenbasiertes Konzept für die Systemarchitektur mit verteilten Miniatursteuerungen vorgeschlagen, s. Abbildung 3-8.

Einklang mit dem DEVEKOS-Systemverständnis:

Die Betrachtung der Komponenten als Agenten folgt aus dem HMS Ansatz, der das Grundkonzept des DEVEKOS-Systemverständnisses definiert. Eine Komponente soll nach dem HMS-Verständnis autonom auf Umweltbedingungen entsprechend ihrer Verhaltensregeln reagieren. Diese Definition deckt sich mit der eines Agenten. Die Herausforderung liegt in der Definition der internen Verhaltensregeln. Nach außen wird diese Komplexität jedoch für den Anwender und die weiteren Komponenten nicht offengelegt, sondern durch einfache Schnittstellen, die Fähigkeiten, dargestellt. Da auf allen Komponenten die Verhaltensregeln der Fähigkeiten implementiert werden, handelt es sich um eine verteilte Systemarchitektur, die zusätzliche Vorteile bietet.

Gleichwertige Verteilung der Rechenleistung:

Steuerungssysteme auf Agentenbasis können die Dynamik komplexer Systeme beherrschen und gleichzeitig die Flexibilität und Fehlertoleranz steigern (Merdan et al. 2012, S. 53). Jede Miniatursteuerung nach diesem Ansatz fungiert als eigenständiger Agent mit Bewegungssteuerung nach der antriebsbasierten Struktur aus Abbildung 3-1 (rechts unten). Die Agenten setzen die Bewegungsbefehle in den zugeordneten Achsen um und tauschen ereignisbasiert über das

Produktionsnetzwerk Informationen aus, sodass die Interpolation des Gesamtverbundes aller Achsen eine möglichst hohe Bahntreue einhält. Die Koordination über das Netzwerk wird dabei über drei Hauptaufgaben der Agenten realisiert: Bewegungsvorgaben erhalten, Synchronisierung mit den weiteren Agenten, Vorgabewerte an den Achsen durchsetzen, vgl. (Otto et al. 2014; Lesi et al. 2016).

Angemessene Aufwände bei Rekonfigurationen:

Eine agentenbasierte Systemarchitektur ist sehr gut erweiterbar, da die Agenten der Komponenten identisch mit definierten Grundfunktionen und Schnittstellen implementiert werden. Bei der Rekonfiguration muss eine neu eingebrachte Komponente lediglich mit allen weiteren Komponenten im System gekoppelt werden.

Angemessene Realisierungskomplexität:

Die Umsetzung als agentenbasiertes System kann auf der Miniatursteuerung entsprechend der verfügbaren Rechenkapazität konzipiert werden. Der Funktionsumfang im Vergleich zu einer MC ist deutlich eingeschränkt, kann aber explizit den Anforderungen des Anwendungsfalls angepasst werden.

Ein geeignetes Austauschformat zur Kooperation muss definiert werden, auf dessen Basis die Agenten Entscheidungen zur Umsetzung der Vorgabewerte treffen können, vgl. (Lian et al. 2002). Die Schnittstelle und der Funktionsumfang der Agenten zur Erweiterung um beliebig viele Achsen muss definiert werden um Flexibilität und Wiederverwendbarkeit zu erreichen (Hirsch 2010, S. 68). Dabei muss die Funktion einer Bewegungssteuerung so ersetzt werden, dass eine synchrone Bewegung aller Achsen umgesetzt wird und bei Trägheiten oder Veränderungen in Dynamik der Achsen die Vorgabewerte reaktiv anpasst werden, um die Bahnabweichungen zu minimieren (Xu et al. 2017, S. 117).

Umsetzung einer Bewegungssynchronisierung:

Eine agentenbasierte Systemarchitektur ist bisher in der Produktionstechnik für die synchrone Bewegungssteuerung nicht umgesetzt worden. Bisher gibt es ebenfalls keine systematischen Entwicklungsmethoden für verteilte Steuerungssysteme (Jakovljevic et al. 2019). Geeignete Ansätze können jedoch aus der Theorie der Multiagentensysteme abgeleitet und adaptiert werden. Die verteilte Systemarchitektur und Kapselung der Komplexität erfüllen die Kriterien der Zieldefinition und versprechen eine hohe Flexibilität des Systems auch bei Rekonfigurationen. Es ist zu prüfen, welchen Umfang an Funktionen eine agentenbasierte Systemarchitektur umsetzen kann und wie die Randbedingungen zur Definition der Funktionen und Schnittstellen sind.

3.3 Bewertung der Lösungsansätze anhand der Anforderungskriterien

In Kapitel 2.3 wurden Methoden und Systeme zur Realisierung einer Bewegungssynchronisierung vorgestellt. Diese basieren in den meisten Fällen auf zentralen Ansätzen und lassen sich nicht auf die Verwendung in verteilten Miniatursteuerungen übertragen. In Tabelle 3-1 sind die Forschungsansätze aus Kapitel 2.4 und die vorgestellten Systemkonzepte aus Kapitel 3.2 bezüglich der

Kriterien aus Kapitel 3.2 bewertet. Die Hintergründe zur Bewertung werden nachfolgend erläutert.

Anforderung	**Bahnregelung** (Huan 1982) (2.4.1)	**Scheduling** (Lesi et al. 2019) (2.4.2)	**Netzwerkverzögerung** (Xu et al. 2017) (2.4.3)	zentrales Konzept **zentral** (3.2.1)	paralleles Konzept **leaderbasiert** (3.2.2)	berechn.-verteiltes Konzept **partitioniert** (3.2.3)	verteiltes Konzept **agentenbasiert** (3.2.4)
Umsetzung einer Bewegungssynchronisierung	●	○	◕	●	◕	●	●
Einklang mit DEVEKOS-Systemverständnis	◑	●	◑	○	◑	●	●
Gleichwertige Verteilung der Rechenleistung	◑	●	◔	○	○	●	●
Angemessene Aufwände bei Rekonfigurationen	○	●	◕	○	●	◑	●
Angemessene Realisierungskomplexität	◑	◑	◕	●	◕	○	◑

○ - nicht erfüllt ◔,◑,◕ - teilweise erfüllt ● - erfüllt

Tabelle 3-1: Vergleich der Anforderungserfüllung der Konzepte aus dem Stand der Technik und Forschung sowie Gegenüberstellung mit den vorgestellten Systemkonzepten.

Die bestehenden Ansätze aus dem Stand der Forschung konnten nicht alle Kriterien der Zielerreichung abdecken. Es wurden drei Lösungen vorgestellt:

1. Bahnregelung (Huan 1982):
   Beim Ansatz der Bahnregelung wird die Projektion der Leaderbewegung auf die Dynamik der Followerachse berechnet und dieses Ergebnis als Vorsteuerung der Followerachse übergeben. Die Rechenkapazität kann dezentral verteilt werden, falls die Projektion in der Miniatursteuerung einer Followerachse durchgeführt wird. Da allerdings die Funktionsbeschreibung der Bahn bekannt und der Bahnfehler pro Achse funktional darstellbar sein muss, wird umfangreiches Systemwissen im Vorfeld benötigt, sodass Erweiterungen auf weitere Achsen hohe Aufwände generieren. Die Hierarchie durch Leader- und Followerachsen bringt zusätzliche Einschränkungen mit sich, da bei Stillstand der Leaderachse keine Ansteuerung der Followerachsen abgeleitet werden kann. Eine Übertragung des Bahnregelungskonzeptes auf die Miniatursteuerungen wird aufgrund dieser Einschränkungen nicht angestrebt.
2. Scheduling (Lesi et al. 2019):
   Der Ansatz der verteilten Ansteuerung von Lesi betrachtet bereits die Verwendung von Miniatursteuerungen. Der Fokus liegt auf der synchronen Ausführung der Bahnbefehle. Er analysiert den Ablauf der Funktionen in

einer CNC, um auf eine zentrale Steuerungseinheit zu verzichten und die Berechnungen ab einem sinnvollen Ablaufschritt auf der LLC direkt an den Achsen durchzuführen. Dadurch nutzt er die zur Verfügung stehende Rechenleistung sinnvoll aus. Eine kinematische Kopplung der Achsbewegungen betrachtet er allerdings nicht, sodass lediglich ein gleichzeitiger Bewegungsbeginn, aber keine synchrone Bewegung umgesetzt wird.

3. Netzwerkverzögerung (Xu et al. 2017):
   Xu et al. betrachten die Umsetzung von synchronen Bahnen bei verteilten Steuerungen, verwenden jedoch ein Leader-Follower-Konzept ohne Rückkopplung oder Verwendung der Rechenleistung der Followerachsen. Das Konzept betrachtet die Netzwerkverzögerung der Kommunikation im Regleralgorithmus, lässt sich aber durch die vorgeschlagene Struktur nicht auf beliebig viele Achsen übertragen, da dann die Verzögerungen der Vorgaben ebenfalls sehr hoch werden.

Auf Basis der Ansätze aus dem Stand der Technik und der Forschung wurden Systemkonzepte für die Umsetzung einer synchronen Bewegungssteuerung auf den Miniatursteuerungen diskutiert. Vier Lösungen wurden vorgestellt:

1. Systemarchitektur mit zentraler Bewegungssynchronisierung (zentral):
   Die Verwendung einer MC, wie sie aktuell in vielen Anwendungen eingesetzt wird, verfolgt mit der zentralen zusätzlichen Recheneinheit nicht dem DEVEKOS Systemgedanken und verzichtet auf die Verwendung der Rechenkapazität der Miniatursteuerungen. Rekonfigurationen benötigen hohe Konfigurationsaufwände.
2. Systemarchitektur mit leaderbasierter Bewegungssynchronisierung (parallel):
   Mehrere Lösungskonzepte mit der Verwendung von Leader-Follower-Architekturen, unter anderem auch zur Kopplung von mehreren Steuerungen, setzen synchrone Bewegungen um. Durch die Struktur der Hierarchie fehlt allerdings oft die Rückkopplung von Informationen der Followerachsen. Verfügbare Rechenkapazitäten der Miniatursteuerungen werden für Followerachsen nicht verwendet.
3. Systemarchitektur mit partitionierter Bewegungssynchronisierung (berechnungsverteilt):
   Die vorgestellte Partitionierung der CNC-Funktionalitäten ist ein vielversprechendes Konzept, da keine Berechnungen doppelt ausgeführt werden und das System jeweils nach verfügbarer Kapazität ideal ausgenutzt werden kann. Allerdings ist die Erweiterbarkeit im laufenden Betrieb unklar und die technischen Voraussetzungen zur Übertragbarkeit der Multicoresysteme auf ein Netzwerkproduktionssystem bisher nicht vorhanden.
4. Systemarchitektur mit agentenbasierter Bewegungssynchronisierung (verteilt):
   Das Agentenkonzept verspricht eine Umsetzung der synchronen Bahn, bei der die gestellten Kriterien komplett erfüllt werden. Ein agentenbasiertes System verspricht Flexibilität, hohe Performanz durch Nutzung der verteilten Systemarchitektur und gute Erweiterbarkeit durch einfache und klar definierte Schnittstellen. Für die Umsetzung müssen die Randbedingungen der Anwendungsfalls der verteilten Interpolation aus Sicht eines

Agentensystems bewertet werden. Zur Entwicklung der Agenten stehen verschiedene Vorgehensmodelle und Entwurfsmuster zur Verfügung. Ebenfalls ist die synchrone Bewegung eine weit verbreitete Anwendung für Agentensystem. Die Analyse des Anwendungsfalls und der Übertragbarkeit von bestehenden Konzepten wird zur Spezifikation des Multiagentensystems benötigt.

3.4 Auswahl eines Lösungsansatzes und Vorstellung der weiteren Vorgehensweise

Im ersten Teil der Arbeit ist die Anforderungsdefinition, sowie der Stand der Technik und Forschung umfassend dargestellt. Über den Vergleich verschiedener Systemkonzepte wurde die Entscheidung für eine Systemarchitektur zur Integration von Bewegungssynchronisierung auf Miniatursteuerungen motiviert und damit das Zwischenziel 1 umgesetzt.

Zielerreichung:

Auswahl der agentenbasierten Systemarchitektur zur Integration von Bewegungssynchronisierung auf Miniatursteuerungen (Zwischenziel 1)

Zur Erfüllung der vorgestellten Kriterien ist die Umsetzung eines Agentenkonzeptes zur Bewegungssynchronisierung mit verteilten Steuerungen von besonderem wissenschaftlichem Interesse und verspricht eine vollständige Erfüllung der Kriterien zur Zieldefinition: Ein Agentenkonzept ist bisher in der Produktionstechnik noch nicht für synchrone Bewegungssteuerung angewendet worden, setzt aber das Konzept der HMS konsequent bis in positionierende Automatisierungskomponenten fort. Das Potential der agentenbasierten Methodik soll im Folgenden aufgezeigt werden. Ebenfalls werden die Grenzen des Funktionsumfangs evaluiert und Anwendungsfälle für den Einsatz der verteilten Interpolation eingegrenzt.

Forschungsfrage:

Welches Potential und welchen Funktionsumfang kann eine agentenbasierte Bewegungssynchronisierung in Mehrachssystemen aufzeigen?

Als Basis der wissenschaftlichen Analyse wird die Theorie zu Multiagentensystemen, deren bisherige Anwendungen im produktionstechnischen Umfeld sowie Ansätze zur Bewegungssynchronisierung in Multiagentensystemen auf die Übertragbarkeit für die Interpolation untersucht und darauf eine Bewegungssynchronisierung für dezentral gesteuerte Mehrachssysteme abgeleitet.

Gesamtziel der Arbeit:

Entwicklung einer Bewegungssynchronisierung für dezentral gesteuerte Mehrachssysteme

Das Gesamtziel der Arbeit wird im Folgenden verkürzt als verteilte Interpolation bezeichnet.

Die weitere Vorgehensweise zur Beantwortung der Forschungsfrage wird als Detaillierung von Abbildung 1-4 im Folgenden erläutert:

Es folgt im zweiten Teil der Arbeit die Analyse der Aufgabe und des Systems im Kontext eines Multiagentensystems (MAS) in Kapitel 3. Weiter wird in Kapitel 1 der Transfer von MAS-Algorithmen zu koordinierter Bewegung auf die verteilte Interpolation betrachtet. Das zweite Zwischenziel des Synchronisierungsmechanismus wird auf Basis der vorhergehenden Analysen erarbeitet. Im dritten Teil der Arbeit werden das Interaktionskonzept, die Agentenlogik und die Kommunikationsstruktur in Kapitel 1 validiert, sowie ein Anwendungsfall zur Rekonfiguration, Erweiterung und Integration von kommerziellen MC-Komponenten betrachtet um das dritte Zwischenziel der Schnittstellendefinition zur Erweiterbarkeit und Rekonfiguration umzusetzen. Abschließend folgt eine Bewertung von Potential und Funktionsumfang der verteilten Interpolation und damit die Beantwortung der Forschungsfrage in Kapitel 6.3, s. Abbildung 3-9.

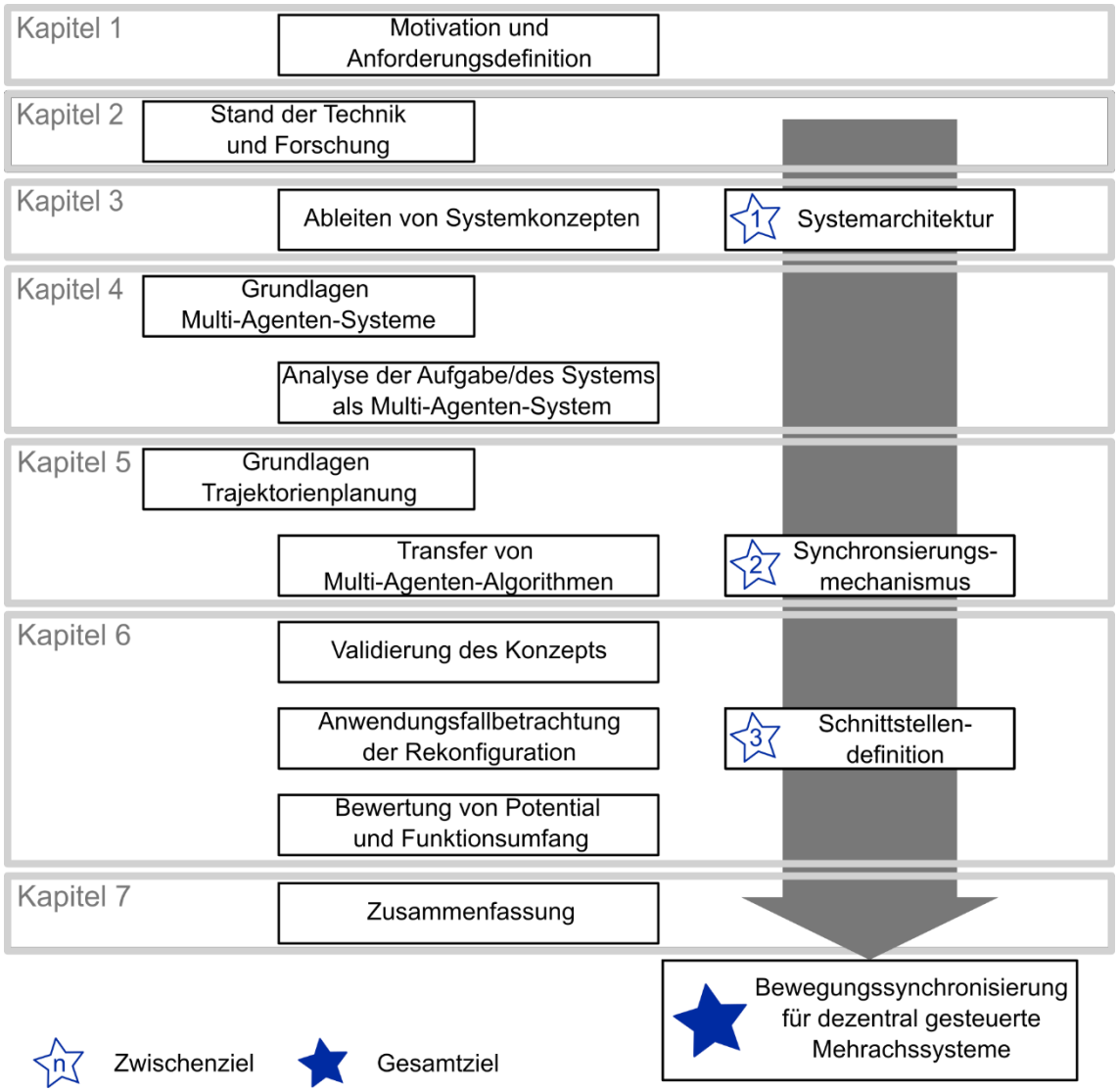

Abbildung 3-9: Vorgehensweise und Aufbau der weiteren Arbeit.

4 Systemanalyse der verteilten Interpolation als Multiagentensystem

In diesem Kapitel wird die Analyse vorgestellt, wie die verteilte Interpolation als Multiagentensystem (MAS) umgesetzt werden kann. Dazu werden im Folgenden die Grundlagen zu MAS erläutert, bisherige Anwendungen von MAS in der Produktionstechnik vorgestellt sowie eine Analyse entlang bekannter Taxonomien und Vorgehensmodelle durchgeführt. Abschließend werden Interaktionsmuster von MAS zur koordinierten Bewegung analysiert und die Übertragbarkeit auf die verteilte Interpolation bewertet.

Vorgehen:

Klärung der Grundlagen zu Multiagentensystemen,

Analyse der Aufgabe und des Systems im Kontext eines MAS

4.1 Grundlagen zu Multiagentensystemen

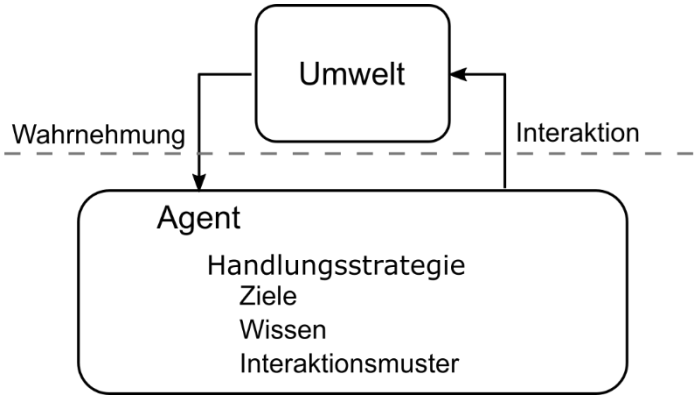

Abbildung 4-1: Struktur eines Agenten, angelehnt an (Schöbel 2019).

Ein Agent ist eine (Software-)Einheit, die aufgrund der ihr zur Verfügung stehenden Informationen und ihrer internen Zustände über die nächste Aktion/Handlung entscheidet, vgl. Abbildung 4-1. Agenten stellen in der Softwaretechnik neben der strukturierten Programmierung und der objektorientierten Entwicklung ein weiteres Entwurfsmuster dar und sind bereits seit mehreren Jahrzehnten in der Softwaretechnik im Einsatz (Parunak 1997, S. 70). Im Falle von technischen Systemen ist ein Agent mit Sensoren zur Erfassung des eigenen und unter Umständen des Umgebungszustandes sowie mit Aktoren zur Interaktion mit der Umgebung ausgestattet, vgl. (Lüth 1998, S. 65–67).

MAS kennzeichnen Systeme mit mehrfach vorhandenen, intelligenten Agenten, die gemeinsam eine Aufgabe lösen können, ohne dass dabei eine zentrale Instanz die Koordination des Gesamtsystems übernimmt. Die Agenten haben in den meisten Fällen kein komplettes Systemwissen, sondern verfügen über nur begrenztes Wissen zur Umwelt. Eine Kommunikation findet nur mit wenigen benachbarten Agenten statt, sodass lokales Wissen über die Umwelt und die Zustände benachbarter Agenten verfügbar gemacht wird. Die eingesetzten Agenten

in einem MAS sind entweder aufgabenspezifisch unterschiedlich oder alle identisch aufgebaut.

MAS ohne Hierarchiedefinition und mit einer hohen Anzahl an sehr ähnlichen oder identischen Agenten nennt man Schwärme. Vorbilder aus der Natur sind beispielsweise Vogel- und Fischschwärme. Diese funktionieren in ihrer Interaktion meist nach sehr einfachen Prinzipen, es gelingen aber trotzdem auch komplexere Bewegungen und Verhaltensweisen als Schwarm. Diese Eigenschaft, die erst bei der Interaktion von vielen Agenten im Schwarm ersichtlich wird, nennt man Schwarmintelligenz, vgl. (Fritsch 2009, S. 24).

Ein Schwarm muss keine Mindestgröße haben, um als solcher bezeichnet zu werden. Sobald das Verhalten von mehreren Agenten Eigenschaften der Schwarmintelligenz zeigt, kann man auch bereits drei Agenten in einer Einheit als Schwarm bezeichnen (Hamann 2018, S. 5). Vom Verhalten eines natürlichen Schwarms auf das zu implementierende Verhalten eines technischen Schwarms zu schließen, ist eine sehr große Herausforderung (Parunak 1997). Vielfach werden komplexe Interaktionen verwendet, die einer Skalierung entsprechend des natürlichen Vorbilds nicht Stand halten. Die technische Implementierung eines Schwarms, z. B. in der Schwarmrobotik, kann weitere Einschränkungen des Schwarmverhaltens bringen, die in einem allgemeinen, rein softwarebasierten MAS nicht vorliegen, weil z. B. Technologien verwendet werden, die nur begrenzt auf beliebig viele Teilnehmer übertragbar sind (z. B. Bluetooth, vgl. (Hamann 2018, S. 9)).

Da das Themengebiet der MAS hauptsächlich durch Publikationen in englischer Sprache geprägt ist, werden nachfolgend zu den deutschen Übersetzungen die englischen Entsprechungen der Schlüsselbegriffe genannt.

4.1.1 Interaktion in Multiagentensystemen

Die Interaktionslogik zwischen den Agenten definiert das globale Verhalten des MAS. Möglichst einfache lokale Verhaltensprinzipien, die eine globale Aufgabe lösen können, müssen definiert werden. Hier gibt es verschiedene Herangehensweisen: Es können in den Agenten identische Strategien implementiert sein, oder eine vorher definierte Aufgabenteilung bedarf unterschiedlicher Strategien. Es gibt allerdings keine allgemeingültige Vorgehensweise zur Ableitung der lokalen Strategien aus den Anforderungen für die globale Lösung. Eine Auswertung der Interaktion der lokalen Strategien ist unabdinglich, da teilweise Effekte im globalen Verhalten des MAS nicht vorhergesehen werden können.

Es werden reaktive (reactive), hybride (hybrid) und erwägende (deliberative) Agenten unterschieden, vgl. (Correll et al. 2013, S. 9). Reaktive Agenten besitzen ein Kostenfunktional, dessen Auswertung direkt auf vorhandenen externen und internen Informationen die Handlungen ableitet. Hybride Agenten haben abzählbar viele Verhaltensweisen, die je nach Situationsanalyse zur Auswahl der Handlung verwendet werden. Erwägende Agenten besitzen dagegen umfangreiche Zustandsmodelle und können einen kontinuierlichen Handlungsspielraum darstellen. Das Wissen (knowledge), das jeder Agent in einem gewissen Umfang besitzt, ist in der Regel beschränkt auf ein eingeschränktes Umgebungswissen,

d.h. ein Wissen über weitere Agenten, deren Zustände und Verhalten sowie die Umwelt, vgl. (Ishida 1994, S. 63). Dieses Wissen kann durch Kommunikation (communication) erweitert werden. Kommunikation ist die Übergabe von Informationen, die einem Agenten, aber nicht notwendigerweise allen weiteren Agenten bekannt ist (Lüth 1998, S. 78). Der Kommunikationskanal ist von essentieller Bedeutung für die Zusammenarbeit von Agenten, wenn eine gemeinsame Aufgabe gelöst werden muss.

Unter Beachtung von vorhandenem Wissen und Kommunikation, sind Konzepte wie die Kooperation, Koordination, Kollaboration und die Konsensfindung die Grundlage der Interaktion. Es wird nachfolgend die Trennung der Begrifflichkeiten erläutert, vgl. (Lüth 1998, S. 77–83; Shen et al. 2003, S. 147–177; Bussmann et al. 2004, S. 27–36), da auch in diesem Themengebiet die Bezeichnungen, speziell der Begriff der Kooperation in unterschiedlichen, teilweise widersprüchlichem Verständnis, vgl. (Doran et al. 1997, S. 309; Shen et al. 2003, S. 167), verwendet werden.

- Kooperation (cooperation) ist die Zusammenarbeit in einem gemeinsamen Arbeitsraum. Dieser Arbeitsraum wird meist über eine gemeinsame Umwelt definiert. Einzelne Agenten kommunizieren ihren eigenen Status innerhalb dieser Umwelt und folgen einem gemeinsamen Ziel, vgl. auch (Olfati-Saber et al. 2007, S. 217).
- Koordination (coordination) bedeutet in der einfachsten Definition, dass aufgrund der Aktionen eines oder mehrerer anderen Agenten ein Agent eigene Aktionen ableitet, die im direkten Zusammenhang stehen, vgl. (Emery et al. 2002). Es wird auch die zeitliche Abstimmung der Aktionen darunter verstanden, die meist über einen zentralen Leader geschieht, der die Koordination leitet.
- Konsensfindung (consensus negotiation) bedeutet, dass mehrere Agenten ihren Wissenstand miteinander abgleichen, gegebenenfalls korrigieren und darauffolgend Aktionen durchführen, die davon abweichen können, was vor der Konsensbildung geplant war. Ein Konsensalgorithmus bestimmt den Ablauf der Interaktion bzw. Verhandlung, mit der die Information ausgetauscht wird, bis ein Konsens erreicht ist, vgl. (Shen et al. 2003, S. 195; Olfati-Saber et al. 2007, S. 215). Zur Konsensfindung gibt es zahlreiche Verhandlungsverfahren, vgl. (Lüth 1998, S. 220–238; Olfati-Saber et al. 2007, S. 216; Ren et al. 2008, S. 25–152; Dong 2016, S. 33 - 51; Russell et al. 2016, S. 610 - 691; Hamann 2018, S. 129 - 163).
- Kollaboration (collaboration) ist in der einfachsten Definition die gleichzeitige Arbeit an einem gemeinsamen Ziel. Hier ist die Kooperation implizit vorgegeben und eine Koordination nicht notwendig (Emery et al. 2002, S. 3008). In anderen Quellen ist dagegen die koordinierte Kooperation mit zuvor abgestimmten Aktionen gemeint. In vielen Fällen wird die Begrifflichkeit Kooperation und Kollaboration gleichbedeutend verwendet, wenn ein koordinierendes Element in der Kooperation zu Konsensfindung eingesetzt wird (Ren et al. 2008).

Es wird sichtbar, dass sehr unterschiedliche Begriffsdefinitionen für Interaktionen in MAS existieren. Nachfolgend wird daher für die Zusammenarbeit im MAS der Begriff der Kooperation verwendet um in dieser Arbeit alle vorhergehenden

Begriffe mit dem Verständnis zu kapseln, dass an einem gemeinsamen Ziel, mit zusammenhängenden Aktionen und einem koordinierenden Element (zum Beispiel der Kommunikation) in einem gemeinsamen Arbeitsraum gearbeitet wird.

4.1.2 Vor- und Nachteile von Multiagentensystemen

Die Vorteile und Nachteile von MAS sind in der Literatur umfangreich diskutiert, vgl. (Fritsch 2009, S. 20; Hamann 2018, S. 6) sowie mit zentral gesteuerten System verglichen worden, vgl. (Parunak 1994, S. 4; Parunak 1997, S. 90; Ren et al. 2008; Leitão 2009, S. 984). Tabelle 4-1 zeigt die grundlegenden Charakteristiken von MAS im Vergleich zu zentralen Produktionssteuerungen auf.

	MAS	***zentrale Produktionssteuerung***
Rahmenbedingung für den Einsatz in der Produktionstechnik	bei geringen und mittleren Auftragsgrößen mittlerer Varianz	für hohe Auftragsgrößen und geringe Varianz
Struktur	**Heterarchie, ebene Struktur**	Hierarchie, Client-Server-Struktur
Datenhaltung	**geringe Umfänge, dezentral**	hohe Umfänge, zentral
Systemarchitektur	**wandelbare, dynamische Systemarchitektur**	starre, statische Systemarchitektur
Entwicklungsansatz	**bottom-up**	top-down
Systemleistung	**erhöhte Systemleistung durch Parallelisierung oder Synergien in Agenten**	Bottleneck, wenn Berechnungskapazität ausgereizt ist
Wandlungsfähigkeit	**hohes Aufgabenspektrum, adaptiv**	nur Aufgaben, die konkret spezifiziert/vorbereitet sind, keine Wandlungsfähigkeit
Skalierbarkeit	**Skalierbar auf beliebig viele Teilnehmer**	Skalierbar nur bis zu einer vordefinierten Anzahl an Teilnehmern
Rekonfigurierbarkeit/Redundanz	**Redundanz/Austauschbarkeit einzelner Agenten**	Zentrales System bietet keine Redundanz, Austausch von Komponenten ist aufwändig
Fehlertoleranz	**Erhöhte Fehlertoleranz Robustheit bei Ausfall einzelner Komponenten**	schlechte Fehlertoleranz, Systemausfall bei Ausfall einzelner Komponenten
Kommunikationsstruktur	m-n Kommunikation	1-n Kommunikation
Kommunikationsaufwand	hoch, zwischen allen Agenten des Systems	**gering, nur zwischen zentraler Steuerung und Komponenten**
Effizienzmaß	**Effizienz durch Flexibilität**	Effizienz durch Spezialisierung
Komplexität der implementierten Logik	**einfache lokale Agentenlogiken**	komplexe Gesamtlogik, schwer verständlich und wartbar
Zieldefinition	dezentrale Ziele als Teilziele des Systemziels	**zentral, einzelnes Ziel**

Koordination	dezentral, Interaktionsmuster benötigt	**zentral, Berechnungen der zentralen Steuerung**
Stabilität/Vorhersagbarkeit	je nach Systemgröße nicht im Voraus bestimmbar	**sehr hoch**
Schnittstellen	umfangreiche Definition entsprechend des Anwendungsfalls nötig	**Standardschnittstellen**
Berechnungsaufwand	**lokal geringer Aufwand in der Durchführung**	zentral hoher Aufwand in der Durchführung
	fett markiert: Vorteil / passend zum spezifizierten Anwendungsfall	

Tabelle 4-1: Beurteilung der Vor- und Nachteile von MAS gegenüber zentralen Produktionssteuerungssystemen.

Herausforderungen bei der Entwicklung von MAS für die Produktionstechnik stellen vor allem die folgenden Charakteristiken:

- Agenten und deren Verhalten werden individuell und für spezifische Anwendungen entwickelt. Oft kommen hier iterative Ansätze zum Einsatz. Klassische, allgemeingültige Vorgehensmodelle für die Entwicklung monolithischer Gesamtsysteme können nicht direkt auf dieses Vorgehen übertragen werden, da das Agenten- und das Systemverhalten nur in wenigen Fällen direkt voneinander abgeleitet werden können. Siehe hierzu das nachfolgende Kapitel 0.
- Die Zieldefinition und Koordination der Aktionen sind bei zentralen Systemen einfach und nachvollziehbar definier- und umsetzbar. Für die dezentralen Systeme und die Koordination der Agenten müssen Teilziele aus den Systemzielen abgeleitet werden und klare Schnittstellen definiert werden. Dabei muss darauf geachtet werden, dass dezentral nur eine begrenze Rechenkapazität zur Verfügung steht. Gleichzeitig besteht das Risiko, dass die Stabilität des Gesamtsystems nicht in allen Fällen im Voraus bestimmt werden kann.
- Ein hoher Kommunikationsaufwand benötigt die Betrachtung der Bandbreite, Art und Zuverlässigkeit des Kommunikationsmechanismus, sowie die Auswahl eines Kommunikationsprotokolls, sodass der Kommunikationsaufwand an den Anwendungsfall anpassbar wird.

Im Folgenden wird nach einer Vorstellung des bisherigen Einsatzes von MAS in Produktionssystemen ein Vorgehensmodell für die Entwicklung von MAS in Kapitel 4.3 ausgewählt und die Analyse des MAS entlang des Modells durchgeführt. Für die Entwicklung des Agentenverhaltens werden bestehende Ansätze zur Bewegungssynchronisierung in MAS in Kapitel 4.4 vorgestellt.

4.2 Multiagentensysteme in der Produktion

Agentenbasierte Ansätze sind sowohl seit den 1980er Jahren als auch im Zusammenhang mit Digitalisierung und Industrie 4.0 in der Produktionsforschung in die Anwendung überführt worden. Vor der Jahrtausendwende wurden diese Systeme unter Begrifflichkeiten wie Distributed Production System (DPS; verteiltes Produktionssystem (Ishida 1994, S. 61)) oder Distributed Control System (DCS; verteiltes Prozessleitsystem (Wei et al. 2010, S. 16)) erforscht und umgesetzt.

Mit der Digitalisierung und Industrie 4.0 wurde stattdessen der Begriff des cyberphysischen Produktionssystem geprägt. Er hebt den hohen Vernetzungsgrad der cyber-physischen Systeme untereinander hervor, wobei das einzelne CPS eine intelligente Produktionseinheit ist, die durch einen Agenten über eine Schnittstelle ansprechbar sein kann (Pantförder et al. 2014, S. 146). Auch der Begriff des agilen Produktionssystems (Agile Manufacturing) ist dem Ansatz der Prozess- und Produktionsplanung mithilfe MAS zuzuschreiben (Lim et al. 2003, S. 379).

Die Verwendung von MAS-Ansätzen in der Produktionstechnik erfordert die Erarbeitung von Steuerungsalgorithmen, die von klassischen zentralen Ansätzen abweichen. Es müssen Algorithmen für Konsensfindung, Kooperation und Koordination neu entwickelt werden (Monostori et al. 2016, S. 637). Ein Vorteil der Modularisierung der Verantwortlichkeiten in Agenten ist, dass durch die klare Schnittstellenbeschreibung Modularität auch auf das Gesamtsystem oder nachfolgende Produktionssystementwicklungen einfacher übertragen werden kann (Shen et al. 2003, S. 47). Einige Umsetzungen, in denen für produktionstechnische Anwendungsfälle mit MAS solche Algorithmen entwickelt wurden, werden nachfolgend vorgestellt.

Grundsätzlich wird beim Einsatz vom MAS in der Produktionstechnik die Verantwortung der einzelnen Produktionsschritte an die Agenten der CPS übergeben, vgl. (Monostori et al. 2016, S. 626). Dadurch wird die Koordination dezentralisiert, sodass Informationen jeweils nur denjenigen Agenten vorliegen, die im Ablauf auf das notwendige Wissen zugreifen, siehe hierzu auch Abbildung 4-2. Die Referenzarchitektur wird auch als marktbasierter Ansatz (market based approach) bezeichnet und folgt dem PPR-Modell (Produkt, Prozess, Ressource), s. (Bracht et al. 2018, S. 62–74). In diesem Modell werden die Entitäten in der Prozessgestaltung mittels Prozessbeschreibungen verknüpft. In der Entsprechung als Agentenmodell werden dem Produkt, dem Prozess und der Ressource jeweils Agenten zugeordnet. Die Prozessbeschreibung dient als gemeinsame Wissensbasis für die Interaktion der Agenten. Mit diesem Modell folgt die Auslastung der vorhandenen Produktionsressourcen/Ressourcenagenten der Nachfrage durch den Prozessagenten nach (Shen et al. 2003, S. 199).

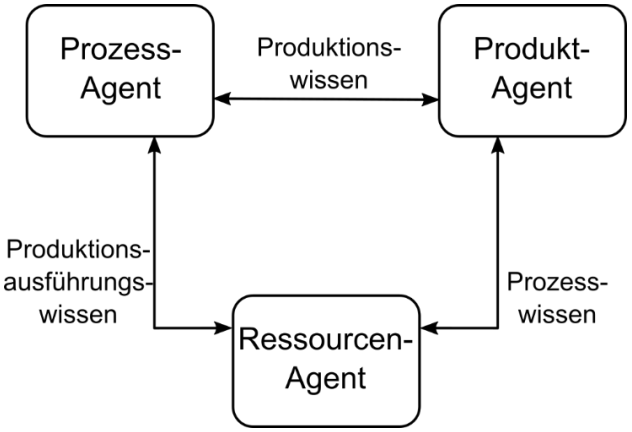

Abbildung 4-2: Referenzarchitektur für ein Agentensystem in HMS für die Trennung von Agentenfunktionalität in der Produktionstechnik nach (Monostori et al. 2016, S. 626).

Um die Auslastung der Produktionsressourcen auch bei stark schwankender Nachfrage bestmöglich zu verteilen und die Redundanz der Ressourcen im Fehlerfall zu nutzen, sind produktionstechnische MAS oft mit dem Fokus auf die Ressourcenverteilung aufgebaut. Im Anwendungsfall Produktion 2000+ (Bussmann et al. 2004, S. 44–45) wurde ein flexibles Transportsystem aufgebaut, das Werkstücke je nach Auslastung den redundant verfügbaren CNC-Maschinen zuliefert. Auch das Projekt Holomobiles (Bussmann et al. 2004, S. 46–49) verwendet fahrerlose Transportfahrzeuge um redundante Maschinen bei Bedarf über Pufferspeicher und Verteilung des Pufferspeicherinhalts auszulasten und Ausfälle zu kompensieren. Auch das Konzept von Pantförder (Pantförder et al. 2014) und Regulin (Regulin et al. 2017), in dem das CPPS zu jedem anderen CPPS über Agentenschnittstellen kommuniziert hat einen ähnlichen Fokus wie Produktion 2000+. Zusätzlich zu den Ressourcenagenten werden jeweils noch Agenten mit Kunden/Auftragssichten sowie ein zentraler Agent zur Koordination instanziiert, die dann die auftragsbasierte Interaktion der Produktionssysteme koordinieren (Pantförder et al. 2014, S. 155). Auch (Lim et al. 2003) verwendet vier Agenten zur Produktionsplanung: Einen Produktagenten, einen Aufgabenagenten, einen Prozessagenten und einen Ressourcenagenten. Die Implementierung solcher Agenten kann auch auf einer SPS-Steuerung umgesetzt werden (Leitão 2009, S. 987). Es braucht hierzu keine dedizierte externe oder zusätzliche Hardware. Zusammenfassend kann festgehalten werden, dass ähnlich der Referenzarchitektur in Abbildung 4-2 in verschiedenen Anwendungen mehrere Agenten mit bestimmten, voneinander unterschiedlichen Aufgabenbereichen definiert wurden.

Neben dem Anwendungsfall der Prozess- und Produktionsplanung wurde auch die Fabrikstrukturplanung als Planungs-/Koordinationsaufgaben, z. B. zur Planung eines Fabrikstrukturlayouts aufgrund von Betriebsmitteleigenschaften, Flächenverfügbarkeit und Auftragslaufwegen als MAS vorgestellt (Harms 2004, S. 105). Alle vorgestellten Ansätze konzentrieren sich hauptsächlich auf die Integration von Prozess- und Produktionsplanung oder flexiblen Produktionssystemen mit MAS (Lim et al. 2003, S. 379; Trentesaux 2009, S. 971).

Aus den vorangehenden Ausführungen wird ersichtlich, dass die Bewegungssynchronisierung bisher in der Produktionstechnik kein Einsatzgebiet für MAS war. Stattdessen konzentrierten sich der Großteil der Aktivitäten auf die Prozessplanung. Die Agentenfunktionalität wird hauptsächlich zur Planung bzw. Koordination als Ersatz für menschliche Aufwände oder Steuerungsabläufe verwendet. Echtzeitfähige bzw. synchronisierte Anwendungen werden mit den hier vorgestellten Anwendungen jedoch noch nicht erreicht, vgl. (Ishida 1994, S. 61), da sie den Echtzeit bzw. Synchronisierungsaspekt nicht in den Fokus rücken. Die Umsetzung von MAS für die Bewegungssynchronisierung stellt jedoch besondere Herausforderungen an die Echtzeit, da Berechnungen und Interaktionen der Agenten zu ausführbaren, stetigen Bewegungen führen müssen.

Die Modularität kann ebenfalls nicht nach Art der Referenzarchitektur aus Abbildung 4-2 abgeleitet werden, denn aufgrund von unterschiedlich ausgelegten Agenten können Konflikte im Systemverhalten entstehen, wenn die geteilte Information nicht in gleicher Art und Weise interpretiert wird (Shen et al. 2003, S. 206).

Daher muss der Fokus bei der Systemkonzeption auf die Interaktion zwischen den Agenten gelegt werden.

Eine weitere Herausforderung ist die Zusammenführung von agentenbasierten Ansätzen mit Service-orientierten Architekturen (SOA), die bereits eine Schnittstelle für die Dienste verschiedener Ressourcen über das Produktionsnetzwerk anbieten (Leitão 2009, S. 989).

Ein Benchmark für die erreichte Leistungsfähigkeit von verteilten Steuerungssystemen ist ebenfalls noch nicht umgesetzt, da durch sehr unterschiedliche Ausprägung der Systemarchitekturen kein direkter Vergleich zu einem zentral gesteuerten System möglich ist (Leitão 2009, S. 988; Brambilla et al. 2013, S. 32).

4.3 Analyse des Anwendungsfalls als Multiagentensystem

Für die Umsetzung eines Agentenkonzeptes wird der Anwendungsfall der verteilten Interpolation innerhalb der Domäne der MAS eingeordnet. Hierfür wird die technische Entsprechung der verwendeten Systeme mit den Grundkonzepten erläutert, eine Einordnung in Taxonomien vorgenommen sowie Vorgehensmodelle für MAS vorgestellt und die Analyse entlang eines geeigneten Modells durchgeführt. Auf dieser Grundlage werden dann im nachfolgenden Kapitel 4.4 weit verbreitete Methoden zur Interaktion und Konsensfindung sowie Interaktionsmuster für koordinierte Bewegungen betrachtet und deren Übertragbarkeit diskutiert.

4.3.1 Technische Entsprechung des Anwendungsfalls

Den vorgestellten Grundlagen eines allgemeinen MAS folgend, werden nachfolgend die Grundstrukturen eines MAS und deren vergleichbare Komponente im technischen System für die Übertragung auf die Umsetzung der verteilten Interpolation geklärt. Da das Agentenkonzept in eine reale Anwendung integriert wird, entstehen Einschränkungen in Bezug auf umsetzbare Systemarchitekturen (vgl. Kapitel 1) sowie verwendbare Entwurfsmuster und Vorgehensmodelle. Tabelle 4-2 schlüsselt die Einordnung im Detail auf.

MAS Komponenten und Konzepte	***Technische Entsprechung für die verteilte Interpolation***
Agent	Softwareeinheit in der Miniatursteuerung mit implementierter Verhaltenslogik
Sensor	Encoder des Antriebs/Motors, Wissen über die Position im Raum
Aktor	Antrieb/Motor; Veränderung der Position im Raum
Umwelt	Arbeitsraum des Systems, im dreidimensionalen Raum. Der einzelne Agent agiert meist in eindimensionalen Achsräumen innerhalb dieses Raums

Wissen	Sensordaten (Position im Raum); Metadaten des eigenen Zustandes (montierte Ausgangsposition, Bewegungsprinzip der Achse, Bewegungsumfang der Achse, erwartbare Dynamik der Achse), über Kommunikation empfangene Information
Kommunikationsstruktur	Produktionsnetz verbindet alle Komponenten, Kommunikation ist unter allen Komponenten möglich, Untergruppen (z. B. des Achsverbundes) können als Sub-Kommunikationsnetz definiert werden.
Kategorisierung als MAS	zutreffend, mehrere identische (homogene) Komponenten, gemeinsam interagierend
Kategorisierung als Schwarm	zutreffend; aufgrund der zu erwartenden Rekonfiguration kann keine aufgabenspezifische Trennung der Agentenlogiken vorgenommen werden, daher ist eine identische Umsetzung in den Ansteuerungen aller Antriebe vonnöten
Kategorisierung als Holon	zutreffend; mehrere Achsen bieten im Verbund als übergeordnetes Holon die mehrdimensionale bewegungssynchrone Interpolation an, werden als Positioniersystem angesprochen. In Ausnahmefällen noch Kommunikation/Anfragen an die individuellen Achsen/Holone; Echtzeitanforderungen gegeben, ansonsten stockt die Bewegungsausführung und -synchronisierung; Verbindung zu physikalischer Einheit ist vorhanden, s.o.

Tabelle 4-2: Einordnung der technischen Entsprechung der Komponenten und Konzepte des MAS für den Anwendungsfall der verteilten Interpolation.

4.3.2 Einordnung in Taxonomien für Multiagentensysteme

Es gibt sehr viele Taxonomien für MAS. Im Folgenden werden nur einzelne Taxonomien mit Relevanz für die spätere Umsetzung aufgegriffen, eine Übersicht weiterer Taxonomien für MAS ist beispielweise in (Fritsch 2009, S. 26; Brambilla et al. 2013, S. 4) zu finden.

4.3.2.1 Taxonomie der Kooperation nach (Doran et al. 1997)

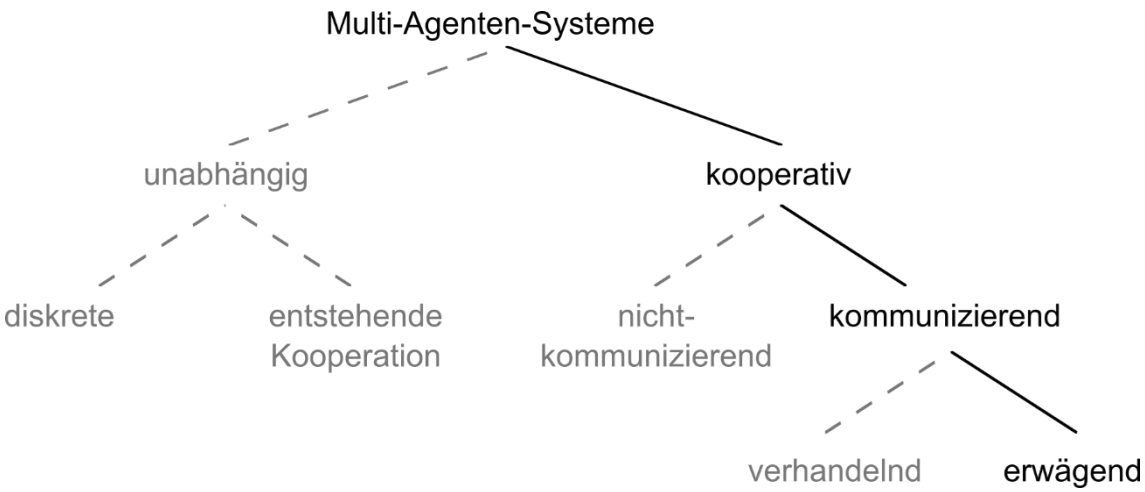

Abbildung 4-3: Taxonomie der Kooperation nach (Doran et al. 1997, S. 310). Dunkel markiert: Einordnung der verteilten Interpolation als MAS.

Doran (Doran et al. 1997) teilt in seiner umfangreichen Betrachtung zur Kooperation MAS in unabhängig (independent) oder kooperativ (cooperative) agierende Systeme ein, s. Abbildung 4-3. Unabhängige Komponenten in der Bewegungssynchronisierung entsprechen der Definition einer parallelen Ansteuerung nach Kapitel 3.2. Innerhalb der kooperierenden Systeme unterscheidet Doran die kommunizierenden (communicative) von nicht-kommunizierenden (non-communicative) Systemen. Nichtkommunizierende Systeme entsprechen der Umsetzung in einer Systemarchitektur mit zentraler Steuerung aus Kapitel 3.2.1. Innerhalb der kooperierenden, kommunizierenden Systeme unterscheidet er die erwägenden (deliberative) und verhandelnden (negotiating) Systeme. Dabei kooperieren die erwägenden Systeme, indem sie ihre Aktionen gemeinsam planen und die Stati der weiteren Agenten in Betracht ziehen. Die verhandelnden Systeme gleichen durch eine auf Wettbewerb ausgerichtete Komponente eine Leader-Follower-Architektur, vgl. Kapitel 3.2.2. Entsprechend folgt entlang der Taxonomie von Doran: Der Anwendungsfall der verteilten Interpolation ist ein erwägendes, kommunizierendes, kooperatives MAS.

4.3.2.2 Taxonomie der Abhängigkeiten (Bussmann et al. 2004, S. 29)

Bussmann klassifiziert die Abhängigkeit zwischen Agenten bei der Kooperation in drei Kategorien: einseitige Abhängigkeit (unilateral dependency); gegenseitige Abhängigkeit (mutual dependency) und reziproke Abhängigkeit (reciprocal dependency. Die einseitige Abhängigkeit bedeutet, dass ein Agent A nur aufgrund von Aktionen, Status oder Informationen des Agenten B eigene Handlungsentscheidungen fällen kann, Agent B dagegen ist nicht von Aktionen, Status oder Informationen des Agenten A abhängig. Die gegenseitige Abhängigkeit bedeutet, dass bei beiden Agenten für dieselbe Handlungsentscheidung die Aktionen, Stati oder Informationen des anderen Agenten bekannt sein müssen. Die reziproke Abhängigkeit bedingt, dass Agent A für eine individuelle Handlungsentscheidung

auf Aktionen, Status oder Informationen des Agenten B angewiesen ist, der wiederum für seine individuellen Handlungsentscheidungen zur Erreichung eines unterschiedlichen Ziels Aktionen, Status oder Information des Agenten A benötigt. Da in der verteilten Interpolation keine Aufgabenteilung für die Agenten vorgenommen werden soll, sind keine unterschiedlichen Handlungsentscheidungen zu treffen. Die gegenseitige Abhängigkeit ist jedoch gegeben, da nur aufgrund der Stati der anderen Achsen die weitere individuelle Achsbewegung festgelegt werden kann.

4.3.2.3 Taxonomie der Agentencharakteristiken nach (Gokulan 2011)

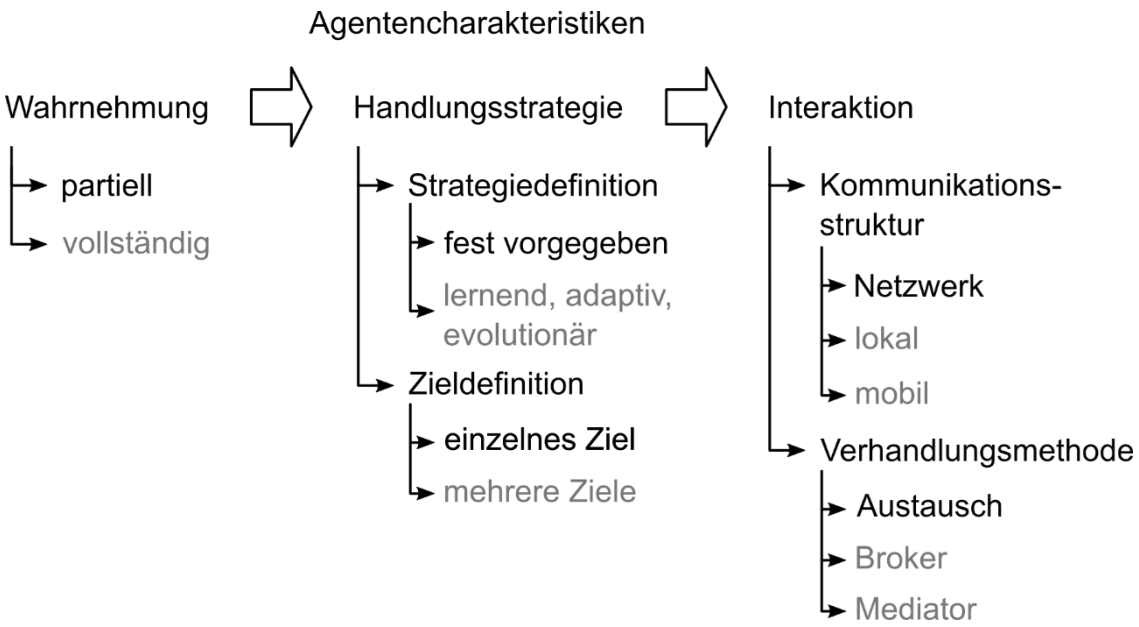

Abbildung 4-4: Taxonomie der Agentencharakteristiken nach (Gokulan 2011, S. 21). Dunkel markiert: Einordnung der verteilten Interpolation als MAS.

Gokulan (Gokulan 2011) betrachtet neben der Architektur und den Protokollen von MAS auch die Agentencharakteristiken in einer Taxonomie, s. Abbildung 4-4. Dabei orientiert er sich an dem klassischen Agentenaufbau mit Wahrnehmung der Umwelt (perception), Ableiten einer Handlungsstrategie (reasoning) und die Interaktion mit der Umwelt (action), s. auch Abbildung 4-1. Die Wahrnehmung der Umwelt kann partiell oder vollständig geschehen. Die vorhandenen verteilten Steuerungen können prinzipiell alle vorhandenen Informationen austauschen, erfassen jedoch einzeln nur ihren eigenen Zustand. Die partielle Wahrnehmung durch einzelne Agenten kann also durch die Zusammenführung über einen Informationsaustausch verbessert werden. Informationen, die den Steuerungen nicht zur Verfügung stehen, können allerdings nicht erfasst werden, weshalb für andere Einsatzzwecke der verteilten Steuerungen, wie beispielweise einer Kollisionsvermeidung, weitere Sensorik vonnöten ist um eine vollständige Erfassung zu gewährleisten oder Strategien zum Umgang mit nur partieller Erfassung der Umwelt erarbeitet werden müssen.

Für die Ableitung der Handlungsstrategie unterscheidet Gokulan mehrere lernende und evolutionäre Ansätze. Innerhalb der verteilten Interpolation soll die

Handlungsstrategie allerdings fest vorgegeben sein, damit die Bahntreue nicht von der Einsatzzeit und Vorerfahrungen des Achsverbundes abhängig ist. Ebenso unterscheidet er Handlungsstrategien, die mehrere Ziele statt nur einem einzelnen verfolgen. Das Ziel der Bewegungssynchronisierung kann zwar ebenfalls in verschiedene Dimensionen aufgebrochen werden (z. B. Bahntreue, Ausnutzen der Dynamik, minimale Abarbeitungszeit, minimale Kommunikationsbedarfe), diese Unterziele werden jedoch im Zusammenhang miteinander als ein globales Optimierungsziel verfolgt. Die Handlungsstrategie muss anpassungsfähig (adaptive) auf Änderungen in den Randbedingungen des Gesamtziels reagieren, die sich auch während der Durchführung der Bahnbewegung ergeben.

Für die Interaktion betrachtet Gokulan speziell die Kommunikationsstruktur sowie das Austauschformat der Agenten. Bei der Kommunikationsstruktur, die zwischen den Agenten verfügbar ist, handelt es sich um das Produktionsnetzwerk, wodurch eine verlässliches Austauschformat und eine verlässliche Austauschrate angenommen werden kann, weshalb kein Broker oder Mediator benötigt wird um Informationen den Agenten zugänglich zu machen. Kommunikationsstrukturen und der Austausch der Positionsinformationen aller Agenten ist nach (Correll et al. 2013, S. 84; Hamann 2018, S. 6) in der Schwarmrobotik eine essentielle Technologieentscheidung für die Ableitung der Handlungsstrategien sowie der Kooperation der Agenten miteinander.

4.3.2.4 Taxonomie der Grundmechanismen für die Kommunikation

Die Kommunikationsstrukturen und -architekturen von MAS sind sehr unterschiedlich aufgebaut. Es werden in verschiedenen Quellen jedoch Grundmechanismen empfohlen:

- Die grundlegende Kommunikationsprinzipien der Agenten sind die Anfrage (request), die Antwort (reply) sowie die Information (inform). Eine mögliche Erweiterung des Kommunikationskonzepts ist die Verwendung von Ausschreibung (call-for-bids) und Angeboten (bids) (Shen et al. 2003).
- Die Verwendung eines Netzwerkbussystems zur Kommunikation zwischen Agenten erhöht die Flexibilität, Effizienz sowie die Zuverlässigkeit der Kommunikation. Zusätzlich ist sie in der Implementierung eine Kostenersparnis gegenüber individuell entwickelten Kommunikationssystemen (Lian et al. 2002)
- Sollte kein Netzwerkbussystem existieren, das sich über den kompletten Agentenverbund erstreckt, kann durch Nachbar-zu-Nachbar-Kommunikation (daisy-chain) auch Informationen von anderen Agenten weitergeleitet werden (Ren et al. 2008, S. 21). Dies ermöglicht auch die Verknüpfung von Subnetzwerken.

4.3.3 Auswahl des Vorgehensmodells und der Agentenarchitektur

Es gibt sehr viele unterschiedliche Ansätze, Entwurfsmuster (design patterns) und Vorgehensmodelle (development models) für MAS. Diese sind oft auf spezifische Anwendungsfälle zugeschnitten, mit dem Fokus benötigte Agentenfähigkeiten zu identifizieren und die Agenten jeweils pro Fähigkeit zu entwickeln (Shen et al. 2003, S. 374; Bussmann et al. 2004, S. 53–115; Fritsch 2009, S. 26). Hierbei werden für die Verhaltensdefinition Zustandsmaschinen, physikalisch-basierte oder verhaltensbasierte Beschreibungsmethoden verwendet. Zweite basieren auf der mathematischen Beschreibung von Zusammenhängen zwischen Agenten, um resultierende Aktionen festzulegen (Brambilla et al. 2013, S. 6). Verhaltensbasierte Beschreibungsmethoden definieren mehrere Verhaltensregeln, die durch einen gewichteten Durchschnitt resultierende Aktionen bestimmen und damit mehrere Ziele gleichzeitig verfolgen können (Beard et al. 2001).

Andere Ansätze versuchen mittels Lernansätzen den Agenten lediglich die Zieldefinition vorzugeben, sodass diese ihre Handlungsstrategie eigenständig entwickeln. Ebenso gibt es Ansätze, um ein Gesamtsystem auf Basis des Agentenverhaltens (mikroskopisches Level) oder des Systemverhaltens (makroskopisches Level) modellbasiert zu entwickeln. Das makroskopische Verhalten kann meist durch Differentialgleichungen beschrieben werden (Foderaro et al. 2014, S. 149). Daraus das mikroskopische Verhalten abzuleiten ist allerdings sehr schwierig. Gleichfalls kann bei der Definition des mikroskopischen Verhaltens das makroskopische Verhalten nicht antizipiert werden, weshalb weiterhin aktiv Forschung für den Zusammenhang zwischen individuellen Agentenverhalten und dem Verhalten eines Schwarms betrieben wird (Shen et al. 2003, S. 33). Die meisten Ansätze konzentrieren sich auf die Konzeption lediglich einer der beiden Verhaltensweisen (Brambilla et al. 2013, S. 10). Neben diesen beiden Ebenen nennt Brambilla auch die Analyse durch die Umsetzung auf realen Robotern als Forschungsmethode. Da nicht alle Verhaltensweisen sinnvoll in Simulationen getestet werden können und durch die technologischen Einschränkungen eine veränderte Robustheit und Reaktionsfähigkeit des Konzepts erwartet werden muss (Brambilla et al. 2013, S. 14), kann so das makroskopische Systemverhalten validiert werden.

Viele Ansätze konzentrieren sich auf die mikroskopische Definition des Agentenverhalten, was im Gegensatz zu der klassischen Systementwicklung der Produktionstechniksteuerungen mit hierarchischem top-down-Ansatz steht und stattdessen an Vorbildern in der Natur orientiert ist (Parunak 1997, S. 70). Forschungen in der Schwarmrobotik fokussieren sich daher meistens auf das mikroskopische Level und betrachten das Schwarmverhalten bei der Implementierung möglichst einfacher Agentenlogiken. Auch Schwarmroboter ohne dedizierte Recheneinheit können durch Adaptionen des Agentenverhaltens unterschiedliches Schwarmverhalten wie das Umzingeln, das Treffen aller Agenten an einem spezifizierten Ort oder das Ausschwärmen zeigen (Johnson et al. 2015). Es soll daher für den vorliegenden Anwendungsfall ebenfalls die Erarbeitung des Agentenverhaltens auf mikroskopischer Ebene verfolgt werden.

Vorgehensmodelle geben kein konkretes Entwurfsmuster vor, sondern definieren den Prozess für die Erarbeitung des Agenten- und Systemverhaltens. Hierfür wurden zwei geeignete Vorgehensmodelle näher analysiert:

- Das DACS (Designing Agent-based Control Systems) Vorgehensmodell gibt für den Agentenentwurf eine Abfolge an Analyseschritten vor, vgl. (Bussmann et al. 2004, S. 119). Demnach sollte das Produktionssteuerungsproblem analysiert werden, indem Eingänge, Ausgänge, Ressourcen und Abweichungen und Störungen im Prozess festgehalten werden. In einem folgenden Schritt werden die daraus resultierenden Steuereingriffe abgeleitet und analysiert. Die Steuereingriffe werden gruppiert und auf verschiedene Agenten aufgeteilt. Nachdem die Agenten und ihre Steuereingriffe feststehen, wird ein Interaktionsprotokoll zwischen den Agenten definiert.
- Auch der DESIRE-Ansatz (DESIRE: DEsign and Specification of Interacting REasoning components) gibt ein Vorgehensmodell für die Konzeption von Agentenstrukturen in MAS vor (Brazier et al. 1997). Ähnlich wie im DACS-Ansatz wird nach der Aufgabenspezifikation ein Informationsaustausch definiert, die Unteraufgaben werden in eine Reihenfolge gebracht, delegiert und abschließend die Wissensstrukturen modelliert.

Beide Vorgehensmodelle sind in der Grundstruktur geeignet um den Ansatz der verteilten Interpolation zu konzipieren. Die Agenten werden jedoch, abweichend zu den Annahmen der DACS und DESIRE Vorgehensmodelle nach obiger Kategorisierung als Schwarmagenten mit homogenen Strukturen konzipiert. Deshalb entfällt die in beiden Vorgehensmodellen enthaltene Verteilung unterschiedlicher Aufgabencluster auf mehrere Agenten. Die Analyse des MAS der verteilten Interpolation wird in Kapitel 4.3.4 dem DACS-Vorgehensmodell entsprechend durchgeführt.

Auf Ebene der Agenten sind Varianten der nachfolgend verwendeten Agentenarchitektur, s. Abbildung 4-5, als Entwurfsmuster in großer Übereinstimmung mit den Konzepten des DESIRE-Ansatzes in der Literatur empfohlen und erfolgreich validiert worden (Ishida 1994, S. 62; Shen et al. 2003, S. 113; Leitão 2009, S. 982; Gokulan 2011; Russell et al. 2016, S. 1005). Es wird in dieser Arbeit bei der Konzeption der Agentenverhaltens auf der abgebildeten Agentenarchitektur aufgebaut und eine Agenten- und Ablaufstruktur vorgeschlagen, die in Kapitel 5.2.3 auf die Anwendung übertragen wird.

Die Grundstruktur besteht aus den Elementen der Wahrnehmung der Umwelt (perception), Ableiten einer Handlungsstrategie (reasoning) und die Umsetzung dieser in Interaktionen (action), wobei die Wahrnehmung der Umwelt Eingänge aus den Sensoren des Agenten sowie der Kommunikationsschnittstelle erhält. Ebenso werden Aktionen über Aktoren ausgeführt und über die Kommunikationsschnittstelle an alle weiteren Agenten kommuniziert. Die Ableitung der Handlungsstrategie wird unter Einfluss des lokalen Wissens (knowledge) bzw. des Ziels (goal) durchgeführt und lässt sich in die drei Teilschritten Interpretation der Information (interpretation), Entscheidungsfindung (decision making) und Planung (planning) unterteilen, vergleiche auch die Entscheidungshierarchie nach (Brazier et al. 1997, S. 73). Die Ausführung der geplanten Entscheidungen

oder Handlung werden dann über die Interaktion kommuniziert, bzw. durchgeführt. Auch der Verknüpfung-Auswahl-Ausführung-Prozess (Match-Select-Act) von (Ishida 1994, S. 68) folgt diesem Muster. Hierbei wird nach der Verarbeitung der eingegangenen Information die Verknüpfung mit dem lokalen Agentenwissen durchgeführt. Die Auswahl des Agentenverhaltens umfasst dabei sowohl die Entscheidungsfindung als auch die Planung der Interaktion, sodass die Ausführung der Interaktion klarer von der Planung getrennt ist.

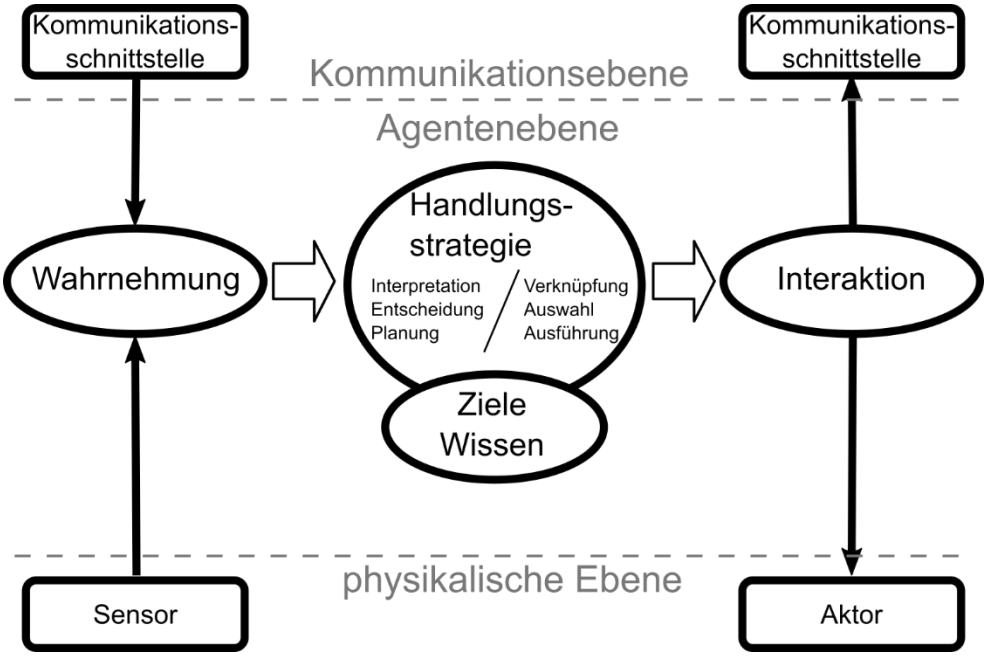

Abbildung 4-5: Entwurfsmuster der Agenten- und Ablaufstruktur.

4.3.4 Analyse der Aufgabeneigenschaften

Für ein umfassendes Verständnis der Aufgabeneigenschaften der verteilten Interpolation wird die Aufgabe entsprechend des DACS Vorgehens analysiert. Das Vorgehensmodell ist in Abbildung 4-6 dargestellt.

4.3.4.1 Spezifikation des Produktionssteuerungsproblems

Die Voraussetzung für die Durchführung der Analyse der Steuereingriffe ist die Spezifikation des Produktionssteuerungsproblems. Hierfür werden die Betriebsbedingungen (Tabelle 4-3), die Prozessdefinition (Tabelle 4-4), sowie die Ziele und Anforderungen (Tabelle 4-5) für die synchrone Bewegungssteuerung strukturiert erarbeitet:

Betriebsbedingungen

Veränderungen im System	Rekonfiguration der Anzahl oder Art der beteiligten Achsen
	Veränderung der vorgegebenen Bahn
	Beschränkung des Arbeitsraums einzelner Achsen
Störungen im System: intern	Schleppabstand durch: - verzögerndes Achsverhalten - Last auf den Achsen - veränderte Dynamik der Achsen
Störungen im System: extern	Schleppabstand anderer Achsen
	Ausfall der Kommunikation

Tabelle 4-3: Betriebsbedingungen nach DACS.

Prozessdefinition

Schnittstelle: Eingänge	parametrisch beschriebene Bahnkurve
	(mittelbar: Sollpositionen im Raum; Sollpositionen der Einzelachsen; Ergebnis der Grobinterpolation)
	Istposition der Achse
Schnittstelle: Ausgänge	Positionsveränderungen der Achse
	Kommunikation
Ressource/ Prozessinhalt	Kommunikationsdurchführung
	Ansteuerung der Achse (Bahnverfolgung, Dynamik)
	Miniatursteuerung für die Ableitung von Handlungsstrategien
	Interaktion mit positionierenden (weiteren Achsen) und nichtpositionierenden (Greifer o.ä.) Komponenten

Tabelle 4-4: Prozessdefinition nach DACS.

Ziele und Anforderungen

Ziele	Bahntreue auf der vorgegebenen Bahn
Anforderungen	zeitoptimale Durchführung
	hohe Dynamik
	geringe Kommunikationslast
	Adaption bei Störungen

Tabelle 4-5: Ziele und Anforderungen nach DACS.

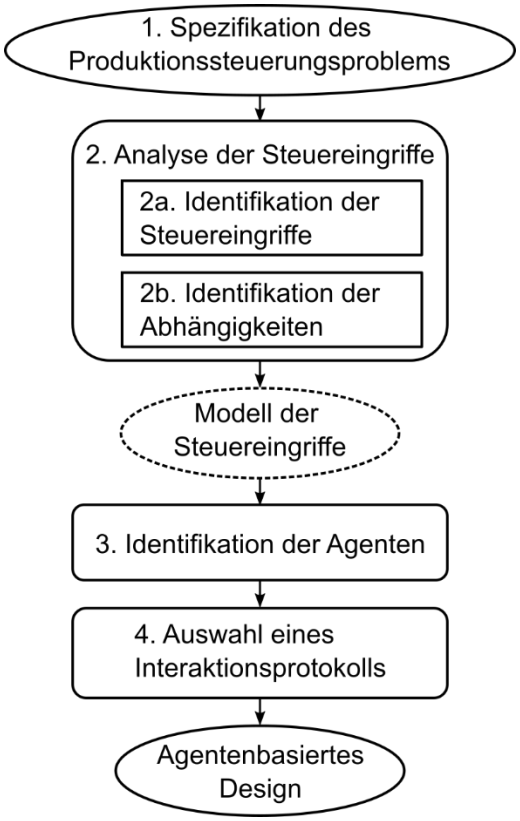

Abbildung 4-6: Vorgehensmodell der DACS Analyse nach (Bussmann et al. 2004, S. 124).

Eine Lösung des Produktionssteuerungsproblems ist entlang der Spezifikation nach DACS in der Lage, über die zur Verfügung stehenden Schnittstellen unter den Prozessbedingungen das Systemverhalten hinsichtlich der Ziele und Anforderungen zu optimieren (Bussmann et al. 2004, S. 124). Im nächsten Schritt, s. Abbildung 4-6, werden die Steuereingriffe und deren gegenseitige Abhängigkeiten identifiziert.

4.3.4.2 Analyse der Steuereingriffe

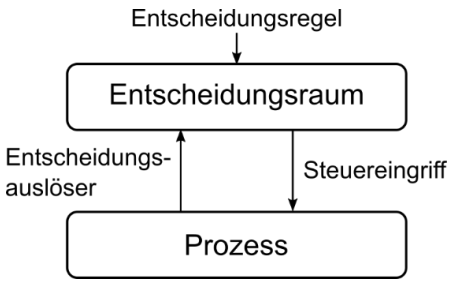

Abbildung 4-7: Zusammenhang zwischen Entscheidungsauslöser, Entscheidungsraum, Entscheidungsregel und Steuerungseingriff nach (Bussmann et al. 2004, S. 125).

Die Analyse der Steuereingriffe im zweiten Schritt befasst sich mit den möglichen Entscheidungen die von der Agentensteuerung innerhalb der Handlungsstrate-

gie (reasoning) getroffen werden können und sollten. Hierfür werden vier Elemente zur Analyse von wirkenden Entscheidungen (effectoric decision) angeführt: Der Entscheidungsauslöser (decision trigger), der Entscheidungsraum (decision space), die Entscheidungsregel (decision rule) und der Steuerungseingriff zur Ausführung der Entscheidung (control interface), s. Abbildung 4-7.

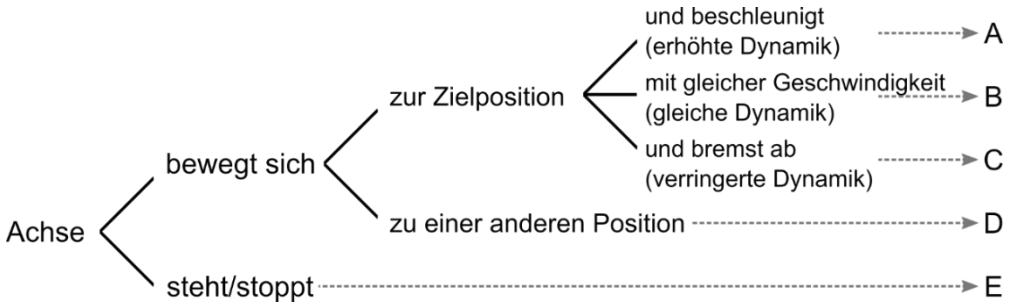

Abbildung 4-8: Handlungsraum einer einzelnen Achse.

Innerhalb der verteilten Interpolation existiert der Handlungsraum für eine einzelne Achse, der in Abbildung 4-8 dargestellt ist. Es folgt auf unterster Ebene eine Aufteilung des Entscheidungsraums in drei Elementarentscheidungen: Beschleunigung, gleichbleibende Bewegung oder Abbremsen. Die Steuerungsschnittstelle ist in allen Fällen die Ansteuerung der Achse durch die Vorgabe von Dynamik bzw. Lagesollwerten. Es folgt daher folgende Aufstellung der Entscheidungsauslöser und Entscheidungsregeln in Tabelle 4-6.

Wenn ***(Auslöser)***	***Dann*** ***(Regel)***
interne Entscheidungsauslöser (Sensorwerte)	
Zielposition nicht erreicht, kein Schleppfehler	B
Zielposition nicht erreicht, Überschwingen des Lagesollwertes	C
Zielposition nicht erreicht, Schleppfehler	A
Zielposition erreicht, Ende der Bahnvorgabe erreicht	E
Zielposition erreicht, weiterer Verlauf der der Bahnvorgabe bekannt	D
unklarer Status, interner Fehler	E
externe Entscheidungsauslöser (Kommunikation)	
weitere Achsen kommunizieren normale Bedingungen	B
weitere Achsen kommunizieren geringere verfügbare Dynamik	C
weitere Achsen kommunizieren höhere verfügbare Dynamik	A
weitere Achsen kommunizieren Erreichen der Zielposition	E
weitere Achsen kommunizieren unklaren Status/Fehler	E
weitere Achsen kommunizieren veränderte Zielposition/weiteren Verlauf der Bahnvorgabe	D

Tabelle 4-6: Entscheidungsauslöser und Entscheidungsregeln.

Als Abschluss der Analyse der Steuereingriffe werden nach DACS die Abhängigkeiten zwischen den Steuereingriffen betrachtet. Es lassen sich Entscheidungsauslöser aus lokaler Information (interne Entscheidungsauslöser) wie Sensorwerten oder Wissen des Agenten von globaler, über die Kommunikation erhaltener Information (externe Entscheidungsauslöser) trennen. Speziell sind hierbei die externen Entscheidungsauslöser zu betrachten, die eine Abhängigkeit zu weiteren Agenten aufzeigen. Interne Entscheidungsauslöser können ohne Abhängigkeit bzw. Austausch mit weiteren Agenten ausgewertet und entsprechend der Entscheidungsregel umgesetzt werden. Bei den externen Entscheidungsauslösern fällt auf, dass die Entscheidungsregeln innerhalb der weiteren Agenten gespiegelt und verstärkt werden. Beispielweise wird im Fall einer reduzierten Dynamik in einem anderen Agenten, durch den betrachteten Agenten ebenfalls ein Abbremsvorgang begonnen und die Dynamik reduziert. Durch die identische Umsetzung der Agenten muss speziell diese Verstärkung betrachtet werden, um das Systemverhalten zu stabilisieren und einen Effekt zu verhindern, der entweder ein Überschwingen oder den Stillstand des Systems zur Folge hat. Zusätzlich ist zu beachten, dass im Allgemeinen die Achsen unterschiedliche Vorgabewerte erhalten, weshalb das Verhalten der anderen Achsen nicht im Vorfeld aus den eigenen achsbezogenen Sollwerten antizipiert werden kann.

4.3.4.3 Identifikation der Agenten

Im dritten Schritt der DACS-Analyse werden die Steuereingriffe geclustert und auf verschiedene Agenten aufgeteilt. Dieser Schritt wird hier übersprungen, da aufgrund der Anforderung möglicher Rekonfigurationen identische Agenten definiert werden sollen.

4.3.4.4 Auswahl eines Interaktionsprotokolls

Der letzte Schritt betrachtet die Auswahl eines Interaktionsprotokolls. Darunter versteht (Bussmann et al. 2004, S. 162) diejenigen Steuereingriffe und Entscheidungen, die nur in Abhängigkeit zwischen den Agenten über Kommunikation durchgeführt werden können. Der Autor empfiehlt die Wiederverwendung oder Adaption von bestehenden Interaktionsmustern und gibt eine Kategorisierung zum Abgleich von Interaktionsmustern vor. Daher werden nachfolgend infrage kommende Interaktionsmuster vorgestellt und deren Übertragbarkeit diskutiert. Die Analyse der Interaktionsmuster nach DACS folgt in Kapitel 4.5.

4.3.4.5 Analyse des Anwendungsfalls

Zusätzlich zur DACS Analyse wurde für den Anwendungsfall auch eine Analyse des Systems nach der PEAS-Beschreibung (Performance Measure, Environment, Actuators, Sensors) (Russell et al. 2016, S. 40–46) durchgeführt, s. (Dripke et al. 2019). Die Erkenntnisse decken sich im weitesten Sinne mit Ergebnissen der DACS Analyse.

Es wird weiter festgestellt:

- Leistungsmaß (Performance measure): entspricht Zielen und Anforderungen des Produktionssteuerungsproblems, s. Tabelle 4-5.
- Umweltdefinition (Environment): Das System agiert im dreidimensionalen Raum. Die Agenten agieren in eindimensionalen, meist linear unabhängigen Achsräumen, es gibt keine definierten Hindernisse
- Aktoren (Actuators) / Sensoren (Sensors): entspricht Prozessdefinition des Produktionssteuerungsproblems, s. Tabelle 4-4

Die Aufgabeneigenschaften werden innerhalb der Dimensionen nach (Russell et al. 2016, S. 42–44) wie folgt bewertet:

- partielle oder volle Informationsabdeckung:
  volle Informationsabdeckung (fully observable) über Sensorinformation oder Kommunikation mit weiteren Agenten, vgl. Kapitel 4.3.2. Durch Beobachteransätze können auch Kommunikationsausfälle überbrückt und die Zustände der weiteren Agenten approximiert werden (Sackenreuther 2019; Quapil 2020).
- Single oder Multi-Agentensystem:
  MAS, vgl. Kapitel 4.3.2
- deterministische oder stochastische Umwelt:
  Es wird von Determinismus im System- und Agentenverhalten ausgegangen.
- Episoden oder Sequenzen im Agentenverhalten:
  sequentielle Abläufe; Agentenentscheidungen können von Zuständen der Vergangenheit abhängig sein.
- statische oder dynamische Umwelt:
  Eine dynamische Interaktion zwischen den Agenten bedeutet, dass eine Aktualisierung der Zustände aller Agenten notwendig ist, bzw. dass durch die diskreten Sollwerte der Bahnvorgabe alleine nicht auf das konkrete Verhalten einzelner Agenten geschlossen werden kann.
- diskrete oder kontinuierliche Zustandsänderungen und Aktionen:
  Agenten können als Aktion kontinuierliche Bewegungen ausführen. Eine Änderung des Zustandes entsprechend Tabelle 4-6 findet jedoch diskret statt. Die Kopplung zeit- bzw. ereignisbasierter Zustandsräume wird bereits durch (Parunak 1997, S. 74) adressiert.
- bekannte oder unbekannte Systemzusammenhänge:
  Die Systemzusammenhänge der Agentenbewegungen sind speziell in Fehlerfällen nicht modelliert, daher sind nicht alle Zusammenhänge bekannt.

4.4 Interaktion, Konsensfindung und koordinierte Bewegungen in Multiagentensystemen

Essentiell für das Systemverhalten eines MAS ist die Interaktion zwischen den Agenten. Für die verteilte Interpolation muss eine Koordination aller Agenten nach den jeweils gültigen Voraussetzungen und Randbedingungen der individuellen Agenten durchgeführt werden. Es werden nachfolgend mehrere Modelle zur

Interaktion und Konsensfindung vorgestellt, sowie die Übertragbarkeit der synchronisierten Bewegung in MAS auf die verteilte Interpolation nach einem DACS-Analyseansatz dargestellt.

4.4.1 Interaktionsmuster und Konsensfindung

Interaktionen zwischen Agenten können neben der bereits erläuterten Informationserfassung durch Sensorwerte und Informationsaustausch über die Kommunikation auch physikalische Effekte, wie den Austausch von Kräften, Momenten, Energie, und andere umfassen (Lüth 1998, S. 78; Fritsch 2009, S. 30–33). Dieser Austausch findet in einem Informations- oder physischen Raum statt, den sich mehrere Agenten teilen. Grundmuster solcher Interaktionen sind in naher Verwandtschaft zu den in Kapitel 4.3.2 genannten Kommunikationsprinzipien zu finden und umfangreich in der Literatur erläutert, s. (Lüth 1998, S. 83–92). Bekannte Formen sind die Dienstleistungsbeziehungen (client-server) und Erzeuger-/Verbraucherbeziehungen (publish-subscribe).

Brambilla teilt Interaktionen nach verschiedenen Interaktionsmustern ein: den räumlich-organisierenden, navigierenden sowie entscheidungsfindenden Verhalten (Brambilla et al. 2013, S. 3). Arai dagegen klassifiziert die Interaktionsmuster entsprechend ihrer Anwendung in kartographische Verhalten, Transport- oder Manipulationsverhalten sowie der koordinierten Bewegung (Arai et al. 2002, S. 655). Die Bewegungssynchronisierung kann als über diese Interaktionsmuster überspannend betrachtet werden, da sowohl räumlich/kartographisch ein bestimmter Punkt eingenommen werden soll (Sollposition), als auch für den Weg zu diesem Punkt eine definierte Bahnvorgabe existiert. Entscheidungsfindende Verhalten sind benötigt, wenn die Agenten eine Strategie für den Umgang mit unerwarteten Abweichungen von der Bahn abgleichen müssen. Nach Arai handelt es sich bei bewegten Achsen um eine koordinierte Bewegung. Sollten Objekte gehandhabt werden, ist dasselbe Verhalten als Manipulationsverhalten einzuordnen.

Obwohl die verteilte Interpolation in ihren Aspekten Charakteristiken aller Interaktionsmuster zeigt, kann der Schwerpunkt auf die Durchführung der koordinierten Bewegung mit Zusammenhang zu den räumlichen Interaktionsmustern gelegt werden, die auch (Brambilla et al. 2013, S. 21) unter den navigierenden Interaktionsmustern kategorisiert. Verhaltensbasierte MAS können durch die Anpassung der Interaktionsmuster auch mehrere Ziele gleichzeitig verfolgen und zwischen ihnen abwägen, vgl. (Balch et al. 1998, S. 926).

Die Konsensfindung ist innerhalb der koordinierten Bewegung notwendig, wenn kein einzelner Agent als führendes Element (leader) fungiert (Beekman et al. 2008, S. 20; Ren et al. 2008; Brambilla et al. 2013, S. 26). Eine Konsensfindung bedeutet, dass alle Agenten das eigene Wissen über ihren Status, der Aufgabe und der Umwelt benachbarten Agenten mitteilen. Das Wissen wird aufgrund der erhaltenen Informationen nach einer definierten Regel aktualisiert (update law), sodass das Wissen in allen Agenten zu einem gemeinsamen Zustand konvergiert, vgl. (Ren et al. 2007, S. 71). Dieses Wissen muss dabei explizit

synchronisiert, d.h. in einem zeitlichen Zusammenhang, ausgetauscht werden, um Mehrdeutigkeiten zu vermeiden (Hamann 2018, S. 6).

Weit verbreitete Interaktionsmuster der koordinierten Bewegung werden durch Zusammenführung (flocking), Ausrichtung (alignement), die Formationsbewegung (formation control), sowie das Rendez-Vous dargestellt, vgl. (Ren et al. 2007, S. 71). Diese Ansätze sollen in den nachfolgenden Kapiteln vorgestellt und diskutiert werden. Eine Übersicht über weitere Anwendungsfälle bietet auch (Arai et al. 2002, S. 657–658; Olfati-Saber et al. 2007; Brambilla et al. 2013, S. 23–24; Hamann 2018, S. 65 - 87; Schöbel 2019, S. 14–17).

4.4.2 Interaktionsmuster Zusammenführung (flocking)

Zusammenführung bedeutet die Annäherung aller Agenten eines Schwarms, sodass die Abstände zwischen den Agenten minimiert sind. Dieses Verhalten ist an biologischen Vorbildern wie Fisch- oder Vogelschwärmen orientiert, vgl. (Brambilla et al. 2013, S. 23), und wird angewendet, wenn die Agenten eine bestimmte Position einnehmen oder in hoher Anzahl an einer gemeinsamen Position sein sollen. Auch in der Bewegungssynchronisierung sollen alle Achsen auf eine räumlich fest definierte Position gelangen.

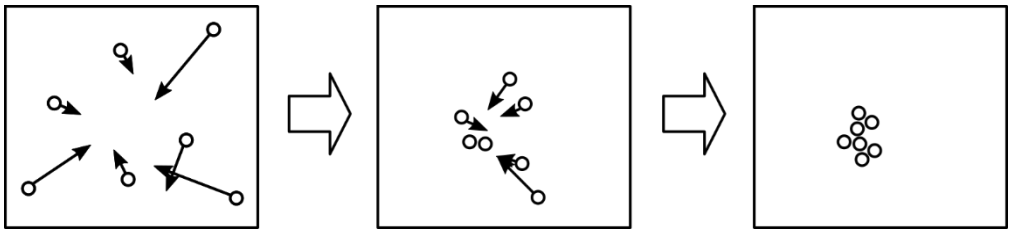

Abbildung 4-9: Interaktionsmuster Zusammenführung (flocking) durch Bewegung auf die mittlere Position zwischen den beiden nächsten Nachbarn.

Umgesetzt wird dieses Interaktionsmuster meist durch Potentialfeldmethoden, sodass Agenten sich entweder ihren Nachbarn (bei beliebiger Zielposition) oder der definierten Zielposition durch eine künstliche Anziehungskraft (artificial potential) nähern (Leonard et al. 2001, S. 2969; Frazzoli et al. 2002, S. 116), s. auch Abbildung 4-9. Es kann statt der räumlichen Annäherung auch im Vorfeld eine Konsensfindung der Zielposition für die Zusammenführung gefunden werden (Olfati-Saber et al. 2007, S. 219). Auch serielle Kinematiken können über die Methode CCD (s. Kapitel 3.2.3) in vergleichbarer Weise zu einer Zielposition bzw. -konfiguration gebracht werden, vgl. die Umsetzung von (Schneider et al. 2012).

4.4.3 Interaktionsmuster Ausrichtung (alignment)

Ausrichtung bedeutet, dass sich die Agenten gegenüber den weiteren Agenten, der Umgebung oder einem führenden Agenten ausrichten und ihre eigene Position auf Basis der möglicherweise veränderlichen Referenzposition einnehmen, wie in Abbildung 4-10 zu sehen. Dieses Interaktionsmuster kommt zur Anwendung, wenn veränderliche oder im Vorfeld nicht fest definierbare Positionen eingenommen werden müssen. Innerhalb der Bewegungssynchronisierung muss

ebenfalls für die Durchführung einer Bahn ihre räumliche Repräsentanz verfolgt werden.

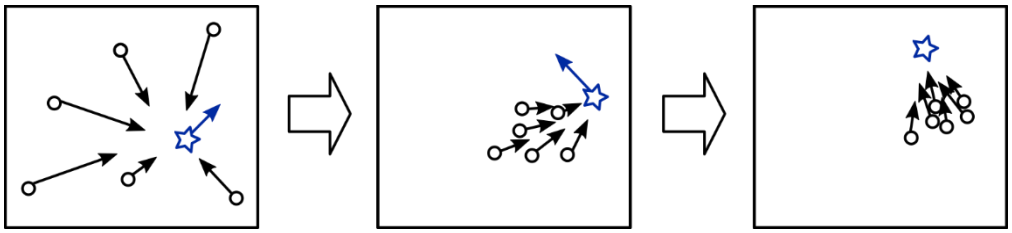

Abbildung 4-10: Interaktionsmuster Ausrichtung (alignment) nach der Bewegung eines Führers.

In der Umsetzung wird meist ein Führer-Mitläufer-Ansatz (leader-follower), vergleichbar zur hierarchischen Leader-Follower-Architektur aus Kapitel 3.2.2 umgesetzt (Beard et al. 2001). Das Folgeverhalten wird entweder ähnlich der Zusammenführung mit physikalisch-basierten Regelungen (Brambilla et al. 2013, S. 23) oder durch eine Kopplung über wirkende Kräfte von transportierten Objekten (Huntsberger et al. 2004, S. 84–87) umgesetzt. Der Führer kann auch als künstliche Einheit (virtual leader) eingesetzt werden, die allen (existierenden) Agenten Bewegungen vorgibt (Leonard et al. 2001). Neben der räumlichen Kopplung über Relativpositionen kann auch eine dynamische Kopplung, wie eine Geschwindigkeitsangleichung umgesetzt werden (Olfati-Saber et al. 2007, S. 218). Für die Ansteuerung von Kinematiken wurde bereits die koordinierte Bewegung einer Plattform mit dem darauf montierten Manipulator nach diesem Interaktionsmuster umgesetzt (Egerstedt et al. 2000, S. 3480).

4.4.4 Interaktionsmuster Formationsbewegung (formation control)

Eine häufig eingesetzte Erweiterung des Interaktionsmusters der Ausrichtung ist die Formationsbewegung. Hier positionieren sich die Agenten entlang einer vorgegebenen Struktur wie in Abbildung 4-11 und halten diese auch bei Bewegungen des Schwarms ein, s. Abbildung 4-12. Eine Bewegungssynchronisierung könnte ebenfalls als eine feste Struktur entlang der Bahn mit Relativbewegungen für alle Achsen interpretiert werden.

Formationsbewegungen werden mit definierten Relativpositionen zu den Nachbaragenten, virtuellen Strukturen oder mitbewegten Koordinatensystemen umgesetzt (Lewis et al. 1997; Desai et al. 1998; Kang et al. 2000; Beard et al. 2001; Russell Carpenter 2002; Murray 2007, S. 572; Olfati-Saber et al. 2007, S. 219; Ren et al. 2008, S. 181–245; Schneider et al. 2012). In allen Agenten werden dann identische Regelalgorithmen implementiert, damit das Folgeverhalten möglichst ähnlich ist.

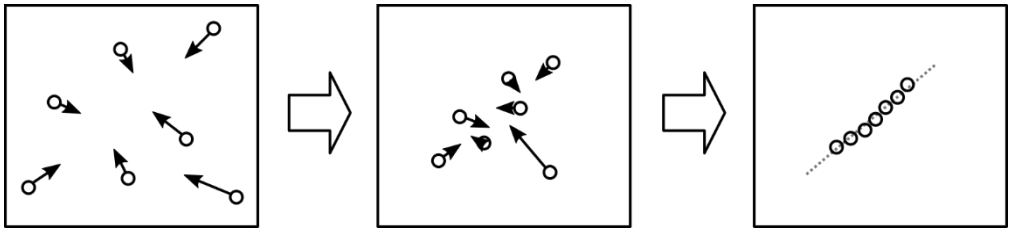

Abbildung 4-11: Interaktionsmuster Formation (formation) der Agenten entlang einer Geraden im Raum.

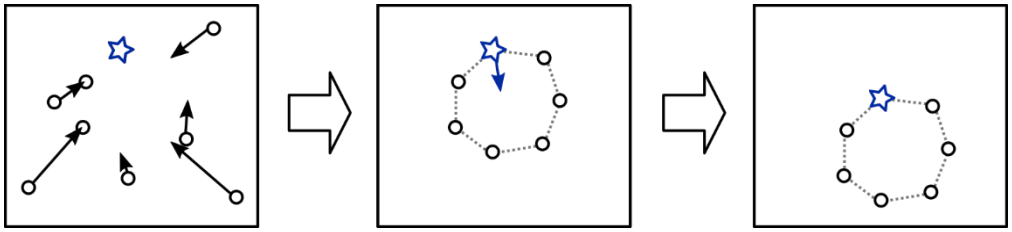

Abbildung 4-12: Interaktionsmuster Formationsfolgeverhalten (formation following) durch Einnehmen der Formation und Folgeverhalten der Bewegungen des Führers.

4.4.5 Interaktionsmuster Rendez-Vous (rendez-vous)

Das Rendez-Vous ist ein spezielles Interaktionsmuster der Zusammenführung der Agenten an einer definierten Position zu einem definierten Zeitpunkt, s. Abbildung 4-13. Hierbei ist der Zeitrahmen für das Erreichen des Zielpunkts meistens sehr eng gefasst, sodass Agenten bereits in der Anfahrt auf den Zielpunkt die Dynamik so ausregeln müssen, dass die Rendez-Vous-Position nicht zu früh oder zu spät erreicht wird, vgl. (Murray 2007, S. 578; Olfati-Saber et al. 2007, S. 218). In den meisten Fällen wird die Pfad- und Dynamikplanung zum Zielpunkt durch die Agenten individuell durchgeführt und benötigt nur geringe Abstimmung zwischen den Agenten (Lin et al. 2007, S. 2123). Dieses Konzept deckt sich mit dem Ansatz der Bewegungssynchronisierung insofern, dass eine asynchrone Bewegung, vgl. Kapitel 2.2.2, durch das gleichzeitige Erreichen des Zielpunkts implizit verhindert wird. Umsetzungen des Rendez-Vous-Interaktionsmusters gibt es beispielsweise in der Drohnensteuerung (McLain et al. 2000; Murray 2007, S. 573).

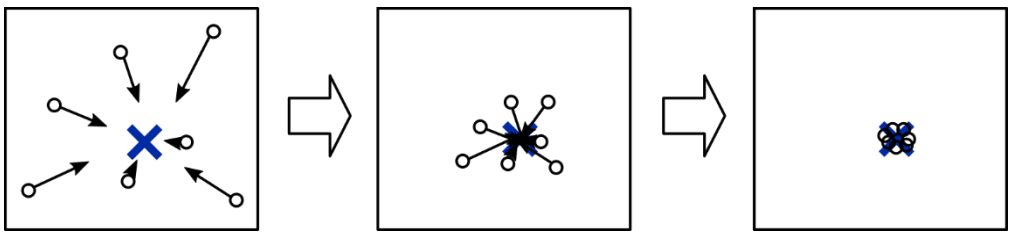

Abbildung 4-13: Interaktionsmuster Rendez-Vous (rendez-vous) an einer Zielposition mit gleichzeitigem Eintreffen aller Agenten.

4.5 Bewertung der Übertragbarkeit der Interaktionsmuster

In den vorhergehenden Kapiteln wurden Grundlagen von MAS und Anwendungsfälle innerhalb der Produktionstechnik vorgestellt. Anschließend wurde eine Einordnung in Taxonomien und Analyse des MAS anhand einsetzbarer Entwurfsmuster und Vorgehensmodelle vorgenommen. Abschließend wurden Interaktionsmuster für die koordinierte Bewegung vorgestellt. Eine Übertragbarkeit dieser Interaktionsmuster wird wie folgt bewertet:

- Die Zusammenführung erreicht nur eine einzelne, in vielen Fällen sogar beliebige, nicht vorher bestimmte Position im Raum. Daher ist eine Bewegung entlang einer definierten Bahn durch dieses Interaktionsmuster nur unzureichend nachzubilden.
- Die Ausrichtung ist ein Interaktionsmuster, das die Agentenarchitektur aus Kapitel 3.2.2 nachempfindet. Aufgrund der Aufwände für die Rekonfiguration des Führers/Leaders sowie der fehlenden Rückkopplung des Mitläufer-/Follower-Verhaltens ist dieses Interaktionsmuster nicht für die Umsetzung der Bewegungssynchronisierung geeignet.
- Die Formationsbewegung scheint für die Umsetzung der Bewegungssynchronisierung geeignet, wenn auch zu klären bleibt, wie das Konzept auf einen Bahnverlauf übertragen werden kann, da die Achsbewegungen in unterschiedlichen Bezugssystemen stattfinden und daher eine Bewegung im Raum unterschiedliche Anforderungen an die Bewegungen der Achsen im Achsraum stellt. Bisherige Forschungsarbeiten betrachten meist nur die Vorgabe einer identischen Bahnvorgabe an alle beteiligten Agenten, vgl. (Xiao et al. 2006, S. 1160).
- Das Rendez-Vous Interaktionsmuster ist speziell wegen der impliziten Synchronisierung durch das gleichzeitige Erreichen der Zielposition interessant und ermöglicht die unabhängige Pfad- und Dynamikplanung der einzelnen Agenten. Es bleibt die Definition einer Rendez-Vous-Position, sodass die linear unabhängigen Achsen im Raum einer Bahn folgen, zu klären.

Eine Analyse der Interaktionsmuster nach DACS wird nach neun Kriterien durchgeführt. Die von (Bussmann et al. 2004, S. 162-179) vorgeschlagenen Antwortoptionen finden sich jeweils in eckigen Klammern. Kriterien für die Analyse der Interaktionsmuster sind:

1. Anzahl der beteiligten Agenten: Wie viele Agenten sind beteiligt? Ändert sich die Zahl über die Zeit?
   [bestimmte Anzahl n; feste Anzahl; veränderliche Anzahl]
2. Sind lokale und nicht-lokale Einschränkungen kompatibel oder werden lokale Entscheidungen durch nicht-lokale Präferenzen eingeschränkt, bzw. sind sie entgegengesetzt zueinander?
   [eingeschränkt; kompatibel; entgegengesetzt]
3. Sind Präferenzen des Systems global (benötigt alle Agenten) oder nicht-lokal (benötigt die Interaktion von mehreren Agenten)?
   [nicht-lokal; global]
4. Anzahl der gemeinsamen Entscheidungen?
   [bestimmte Anzahl n; feste Anzahl; veränderliche Anzahl]

5. Wie viele Agenten müssen für eine Entscheidung übereinstimmen?
   [bestimmte Anzahl n; feste Anzahl; veränderliche Anzahl]
6. Wie ist die Relation der Entscheidungen?
   Kann ein Agent in mehreren Entscheidungen teilnehmen?
   [überlappend],
   In wie vielen Entscheidungen muss ein Agent beteiligt sein?
   [komplette Abdeckung]
7. Wie ist die Rollenzuteilung zwischen den Agenten?
   [keine; feste; veränderliche Rollen]
8. Wie ist die Information verfügbar? Werden Alternativen oder Präferenzen kommuniziert?
   [Alternativen; Präferenzen]
9. Ist eine Delegation der Entscheidung möglich? Kann ein Agent die Entscheidung an einen anderen Agenten abgeben?
   [ja; partiell; nein]

Die Analyse der Interaktionsmuster nach DACS, inklusive der Analyse des Anwendungsfalls der verteilten Interpolation sind in Tabelle 4-7 aufgeschlüsselt.

Aus der Analyse kann geschlossen werden, dass für die verteilte Interpolation bisher kein Interaktionsmuster oder ein vergleichbarer Anwendungsfall existiert, der für die Bewegungssynchronisierung direkt übertragen werden kann. Vor allem Kategorie 7 der DACS-Analyse ist hervorzuheben, nach der eine veränderliche Rollenverteilung, die in einer Rekonfiguration zwangsläufig erwartet werden muss, über die verschiedenen Achsagenten durch die Interaktionsmuster nicht dargestellt werden kann. Die Formationsbewegung sowie das Rendez-Vous-Interaktionsmuster zeigen vielversprechendes Verhalten und haben in der Analyse der Interaktionsmuster in weiten Teilen Überlappungen zum Anwendungsfall. Sie müssen jedoch speziell auf den Anwendungsfall angepasst werden. Im nachfolgenden Kapitel wird daher nach einer Analyse der Aufgabeneigenschaften ein Interaktionskonzept nach Vorbild dieser beiden Ansätze erarbeitet.

	Zusammenführung (flocking)	***Ausrichtung (alignment)***	***Formation (formation)***	***Rendez-Vous (rendez-vous)***	***verteilte Interpolation***
1: Anzahl der Agenten	veränderliche Anzahl	veränderliche Anzahl	feste Anzahl	veränderliche/feste Anzahl	veränderliche Anzahl
2: Kompatibilität der Präferenzen	kompatibel	eingeschränkt	eingeschränkt	eingeschränkt	eingeschränkt
3: Präferenzen des Systems	nicht-lokal	nicht-lokal	global	nicht-lokal	global
4: Anzahl gem. Entscheidungen	0	0	feste Anzahl	1	feste Anzahl
5: Anzahl der übereinstimmenden Agenten	-	-	alle, feste Anzahl	alle, feste Anzahl	alle, feste Anzahl
6: Relation der Entscheidungen	-	-	überlappende, komplette Abdeckung	-, komplette Abdeckung	überlappende, komplette Abdeckung
7: Rollenzuteilung	**keine**	**feste Rolle**	**feste Rolle**	**keine**	**keine/veränderlich**
8: Verfügbarkeit der Information	-	-	Alternativen/ Präferenz	Alternativen/ Präferenz	Präferenz
9: Delegation der Entscheidung	-	-	partielle Delegation	partielle Delegation	partielle Delegation

Tabelle 4-7: Analyse der Interaktionsmuster nach DACS.

5 Konzeption der verteilten Interpolation

Im nachfolgenden Kapitel werden zuerst die Grundlagen der Trajektorienplanung dargestellt, die Übertragung der Bewegungssynchronisierung auf MAS erläutert und ein Interaktionsmuster aus den oben vorgestellten Ansätzen erarbeitet. Für dieses Interaktionsmuster werden das Agentenverhalten und ein Kommunikationsprotokoll entwickelt und damit ein verteilter Synchronisierungsmechanismus konzipiert.

Zielsetzung:

Konzeption eines Synchronisierungsmechanismus (Zwischenziel 2)

Vorgehen:

Klärung der Grundlagen zur Trajektorienplanung,

Transfer von MAS-Algorithmen zu koordinierter Bewegung auf die agentenbasierte Bewegungssynchronisierung

5.1 Grundlagen zur Trajektorienplanung für synchronisierte Bewegungen

Die nachfolgende mathematische Erläuterung der Trajektorienplanung folgt in großen Teilen der Formelbezeichnung und Erläuterung von (Altintas 2012, S. 201ff). Es wird im Folgenden nur die lineare Interpolation betrachtet. Für die Trajektorienplanung im Fall der Kreisbahn- bzw. Polynomkurveninterpolation sei auf (Altintas 2012, S. 234–245) verwiesen.

Ein Vektor $\vec{r}$ im kartesischen Koordinatensystem wird durch

$$\vec{r} = r_x\vec{\imath} + r_y\vec{\jmath} + r_z\vec{k} \in \mathbb{R}^3 \tag{5-1}$$

definiert, wobei $\vec{\imath}, \vec{\jmath}, \vec{k}$ die linear unabhängigen Koordinatenrichtungen (Achsen) aus dem Ursprung des Koordinatensystems sind. Die Koordinaten r_x, r_y, r_z bezeichnen die projizierten Längen der Strecke $|\vec{r}|$ auf die Achsen des Koordinatensystems. Die Projektion wird durch

$$\mathcal{P}_n\colon \{\vec{r} \longmapsto r_n \mid \mathbb{R}^3 \longmapsto \mathbb{R}\} \tag{5-2}$$

definiert. Ein Vektor zwischen zwei Punkten $P_1(x_1, y_1, z_1), P_2(x_2, y_2, z_2)$ kann durch

$$\overrightarrow{P_1P_2} = (x_2 - x_1)\,\vec{\imath} + (y_2 - y_1)\vec{\jmath} + (z_2 - z_1)\vec{k} \tag{5-3}$$

dargestellt werden.

Es wird nachgehend angenommen, dass die Bahnvorgabe für die Bewegungssynchronisierung als Punktmenge im Format

$$[P_0 \quad P_1 \cdots P_{m-1} \quad P_m] = \begin{bmatrix} x_0 & x_1 & x_{m-1} & x_m \\ y_0 & y_1 \cdots & y_{m-1} & y_m \\ z_0 & z_1 & z_{m-1} & z_m \end{bmatrix} = \begin{bmatrix} r_{x0} & r_{x1} & r_{x(m-1)} & r_{xm} \\ r_{y0} & r_{y1} \ldots & r_{y(m-1)} & r_{ym} \\ r_{z0} & r_{z1} & r_{z(m-1)} & r_{zm} \end{bmatrix} \cdot \begin{bmatrix} \vec{\imath} \\ \vec{\jmath} \\ \vec{k} \end{bmatrix} \tag{5-4}$$

als diskrete Sollwerte jeweils mit Start- und Endwerten linearer Interpolationsgeraden (Segmente) $S_i, i = 1 \ldots m$ entlang der Bahn in Weltkoordinaten angegeben sind, wobei P_{i-1} den Startpunkt und P_i den Endpunkt des Segments, s. auch Abbildung 5-1, bezeichnet:

$$\overrightarrow{S_\imath} = \overrightarrow{P_{\imath-1}P_\imath}. \tag{5-5}$$

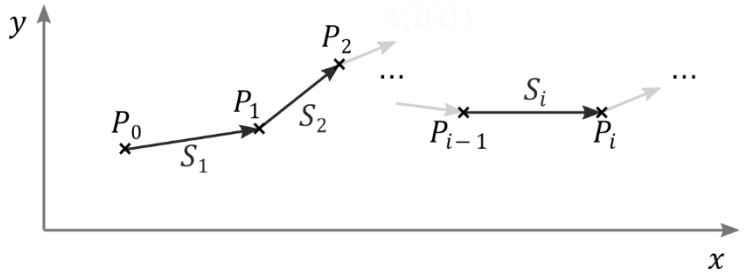

Abbildung 5-1: Segmentdefinition durch die Bahnvorgabe der Sollwerte.

Die geometrische Beschreibung der Bahn ist durch die Bahnvorgabe vollständig definiert. In Bewegungssteuerungen gibt es für eine solche lineare Interpolation zwei verschiedene Ausführungsarten. Punkt-zu-Punkt-Bewegungen bedeuten, dass die Achsen nicht miteinander synchronisiert werden. Für bahntreue lineare Interpolationen wird eine Synchronisierung der Achsgeschwindigkeiten aller beteiligten Achsen dagegen sichergestellt, vgl. (Altintas 2012, S. 197).

5.1.1 Homogene Transformation der Bahn in Achskoordinaten

In den meisten Fällen kann davon ausgegangen werden, dass beteiligte Achsen linear unabhängig sind und dass der Ursprung des Bezugskoordinatensystems der Bahnvorgabe im Schnittpunkt der Bewegungsvektoren der Achsen festgelegt wird. Speziell bei einer Rekonfiguration kann es aber bei nachträglich eingebauten Achsen zu Abweichungen von dieser Annahme kommen. Eine Betrachtung von redundanten, also linear abhängigen Achsen wird in dieser Arbeit nicht speziell vorgenommen, da die Redundanzbetrachtung ein eigenständiges Forschungsgebiet ist. Eine Optimierungsansatz zur Lösung der Redundanz wird beispielsweise durch (Schröder 2007) vorgestellt. Nachfolgend wird davon ausgegangen, dass innerhalb einer kinematischen Kette keine linear abhängigen Achsen eingesetzt werden. Für alle anderen Fällen wird angenommen, dass eine Bewegung innerhalb des Arbeitsraumes von Achsen mit paralleler Achsrichtung auch parallel durch beide/alle Achsen durchgeführt wird.

Jede Achse besitzt einen definierten Arbeitsraum, also eine Bewegungsrichtung und einen Bewegungsbereich. Diese Bewegungsrichtung kann linear oder rotatorisch sein. Sonderbauformen werden im Folgenden nicht näher betrachtet. Sie können aber bei einer bekannten Transformation zum Weltkoordinatensystem ebenfalls in den vorgestellten Ansatz integriert werden. Der Bewegungsbereich liegt bei neu gestalteten Systemen meist so, dass beispielweise die Referenzposition einer Achse (Nullposition der Achse), die zur Bewegung entlang der x-Achse eingesetzt wird, dem Ursprung und ihre Bewegungsrichtung der Orientierung des Elementarvektors $\vec{\imath}$ (Nullposition im Raum) entspricht. Bei rekonfigurierten Systemen kann diese Konvention nicht immer eingehalten werden. Über eine homogene Transformation kann trotzdem das Bezugssystem in Weltkoordinaten in die Achspositionen übertragen werden. Dazu werden die Koordinatenvektoren zu homogenen Koordinaten erweitert:

$$^{h}\vec{r} = \begin{pmatrix} r_x \\ r_y \\ r_z \\ 1 \end{pmatrix} \in \mathbb{R}^4. \tag{5-6}$$

Translationen um einen Vektor $\vec{l} = (l_x, l_y, l_z)$ und Rotationen über die Rotationsmatrix C sind durch die homogene Matrixmultiplikation

$$H \; ^{h}\vec{r} = \begin{pmatrix} & C & & l_x \\ & & & l_y \\ & & & l_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_x \\ r_y \\ r_z \\ 1 \end{pmatrix} \tag{5-7}$$

darstellbar und werden affine Transformationen genannt. Die affinen Transformationen H verändern lediglich die Orientierung einer Vektordarstellung und haben daher keine Auswirkungen auf die Skalierung und Rechenoperationen mit diesen Vektoren. Mit affinen Transformationen können Vektoren aus dem Weltkoordinatensystem in Hilfskoordinatensysteme so übertragen werden, dass der neue Ursprung im Achsnullpunkt liegt.

5.1.2 Dynamikplanung

Durch die Transformation ist die Entsprechung der geometrischen Beschreibung der Bahn in einzelnen Achspositionen fest definiert. Die zeitliche Durchführung der Bahn hat allerdings physische, dynamische und prozessbedingte Grenzen, die in der Dynamikplanung der Trajektorie entlang der Bahn betrachtet werden. Objekte können im Raum durch Trägheit nur begrenzt schnell beschleunigt werden. Bei Achsen folgen durch Bauart und Ansteuerung ebenfalls Grenzen für die Dynamik. Neben der baulichen Begrenzung des Arbeitsraums der Achse gelten die maximalen Geschwindigkeiten, Beschleunigungs-, Verzögerungs- oder auch Ruckgrenzen, die nicht überschritten werden können oder dürfen. Der Prozess einer Maschine hat oft ähnliche Einschränkungen, weshalb die geometrische Bahn hinsichtlich der zeitlichen Durchführung vorab geplant wird um die Grenzen einzuhalten.

Als einfaches Beispiel ist das Geschwindigkeitstrapezprofil für die Bewegung vom Startpunkt P_{Start} zum Zielpunkt P_{Ziel} zu nennen, s. auch Abbildung 5-2. Es ist im Ablauf wie folgt definiert:

1. Die Achse steht still:

$$t < 0\colon\ p_{\text{ist}} = P_{\text{Start}},\ v_{\text{ist}} = 0,\ a_{\text{ist}} = 0$$

2. Die Achse beginnt vom Startpunkt an zu beschleunigen (T_{I}):

$$0 < t < t_1 :\ p_{\text{ist}} = p(t),\ v_{\text{ist}} = v(t), a_{\text{ist}} = a_{\max}$$

3. Die Achse hat die maximale Geschwindigkeit erreicht (T_{II}):

$$t_1 < t < t_2 :\ p_{\text{ist}} = p(t),\ v_{\text{ist}} = v_{\max},\ a_{\text{ist}} = 0$$

4. Die Achse beginnt den Verzögerungsvorgang (T_{III}):

$$t_2 < t < t_Z :\ p_{\text{ist}} = p(t),\ v_{\text{ist}} = v(t),\ a_{\text{ist}} = -a_{\max}$$

5. Die Achse stoppt am Zielpunkt:

$$t_z < t\colon p_{\text{ist}} = P_{\text{Ziel}},\ v_{\text{ist}} = 0,\ a_{\text{ist}} = 0$$

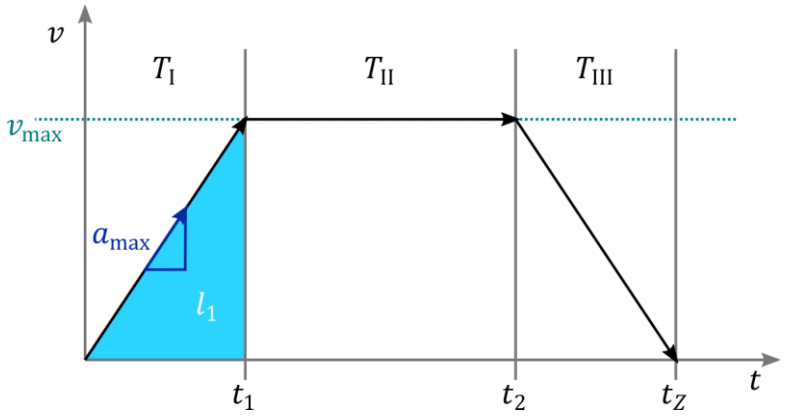

Abbildung 5-2: Geschwindigkeitstrapezprofil nach (Altintas 2012, S. 214).

Sollte aufgrund einer kurzen Segmentlänge oder einer zeitlichen Beschränkung zur Abarbeitung der Bahn die maximale Geschwindigkeit nicht erreicht werden, weil der Verzögerungsvorgang zum vollen Stillstand bereits frühzeitig eingeleitet werden muss ($t_2 \leq t_1$), wird auf Phase T_{III} verzichtet und festgelegt

$$t_1 = t_2 = \frac{t_Z}{2}. \tag{5-8}$$

Bei obigem Beispiel wird von Stillstand der Achse zu Beginn der Bewegung ausgegangen. In vielen Anwendungen wird bei aufeinanderfolgenden Segmenten ein Abbremsen auf vollständigen Stillstand vermieden, sodass zu Beginn des Segmentes die Geschwindigkeit aus dem vorherigen Segment als Ausgangsgeschwindigkeit $v_{ist}(t = 0) = v_0 > 0$ gilt. Die Berechnung der Pfadlänge der Beschleunigung l_1 zwischen $0 < t < t_1$ wird dann statt durch

$$l_1 = \int_0^{t_1} at\, dt = \frac{at_1^2}{2} \tag{5-9}$$

mit der Entsprechung

$$l_1 = \int_{t_0}^{t_1} [v_S + a(t - t_0)]\, dt = \int_0^{\tau_1} [v_S + a\tau]\, d\tau = v_S \tau_a + \frac{a\tau_a^2}{2} \qquad (5\text{-}10)$$

$$\text{wobei } \tau_a = t_1 - t_0 = \frac{v - v_S}{a}$$

dargestellt.

Für die Gesamtlänge des Pfades entlang der Bewegung gilt

$$l = v_S T_\mathrm{I} + \frac{1}{2} a T_\mathrm{I}^2 + v_\mathrm{max} T_\mathrm{II} + v_\mathrm{max} T_\mathrm{III} + \frac{1}{2}(-a) T_\mathrm{III}^2 \qquad (5\text{-}11)$$

$$\begin{aligned} \text{wobei} \quad T_\mathrm{I}&: 0 < t < t_1, \\ T_\mathrm{II}&: t_1 < t < t_2, \\ T_\mathrm{III}&: t_2 < t < t_3. \end{aligned}$$

Da das Trapezprofil in den Übergängen $T_\mathrm{I} \mapsto T_\mathrm{II}$ und $T_\mathrm{II} \mapsto T_\mathrm{III}$ Unstetigkeiten in der Beschleunigung hat, wird oft das umfangreichere Siebenphasenprofil eingesetzt, dass in sieben Phasen vom Stillstand (bzw. Ausgangsgeschwindigkeit) zu Beginn einer Bewegung bis zur maximalen Geschwindigkeit und zurück zum Stillstand (bzw. Restgeschwindigkeit) die Dynamik entsprechend der Grenzen für Geschwindigkeit, Beschleunigung und Ruck definiert, vgl. (Dittrich et al. 1996, S. 61; Altintas 2012, S. 220–229). Andere Ansätze sind die Verwendung von Beschleunigungsprofilen nach Sinusquadrat, Bestehorn, S-Form, usw., die in dieser Arbeit nicht näher betrachtet werden. Es kann allerdings festgehalten werden, dass die Trajektorienplanung allgemein an Komplexität zunimmt, je höher die Anzahl der Randbedingungen ist, die beachtet werden müssen.

5.1.3 Transfer der Dynamikplanung auf die einzelnen Achsen

Aus der Dynamikvorgabe der maximalen Beschleunigungen oder Geschwindigkeiten müssen für die Übermittlung an den Lagesollwertregler die Sollpositionen ermittelt werden. Die Grenzen für die Bahngeschwindigkeit, -beschleunigung und -verzögerung sowie der maximale Ruck stehen in der MC zentral zur Verfügung. Die Trajektorienplanung wird demnach für die Bahn entsprechend dieser Grenzen nach obigen Beschleunigungsprofilen durchgeführt. Die Parameter der Phasen werden bereits vorberechnet, vgl. (Altintas 2012, S. 226), sodass in Echtzeit die Sollwerte der Position, Geschwindigkeit und Beschleunigung generiert werden. Hierfür wird in einer MC die geometrische Bahn in Schritte Δu aufgeteilt, die nach der Taktzeit der Interpolation T_Ipo und der gewünschten Vorschubgeschwindigkeit entlang der Bahn v wie folgt berechnet werden (Altintas 2012, S. 216):

$$\Delta u = v\, T_\mathrm{Ipo} \qquad (5\text{-}12)$$

Für zwei beteiligte, in ihrer Bewegungsrichtung linear unabhängigen, Achsen x, y gilt

$$\overrightarrow{\Delta u} = \Delta x\,\vec{\imath} + \Delta y\,\vec{\jmath} = \begin{pmatrix}\Delta x \\ \Delta y\end{pmatrix}\begin{pmatrix}\vec{\imath} \\ \vec{\jmath}\end{pmatrix} \tag{5-13}$$

und nach (5-12) kann ebenfalls

$$\vec{\mathrm{v}} = \mathrm{v_x}\,\vec{\imath} + \mathrm{v_y}\,\vec{\jmath} \tag{5-14}$$

angenommen werden. Das bedeutet, dass pro Interpolationstakt T_{Ipo} der Vorschub einer an der Bewegung beteiligten Achse der Projektion des Bahnvorschubs auf die Achsdimensionen entspricht. In diesem Sinne sind die Positions- und Geschwindigkeitsvorgabe bei der linearen Interpolation entkoppelt, vgl. (Altintas 2012, S. 213), da der Achsvorschub unabhängig von der Position entlang des Segments in einem festen Verhältnis (entsprechend der Kinematik, bzw. Transformation) zum Bahnvorschub steht. Dies gilt auch für die Trajektorienplanung mittels Trapezgeschwindigkeitsprofil, wenn für jede Phase des Profils eine Schrittweite $\Delta u_{T_{\mathrm{I}}}, \Delta u_{T_{\mathrm{II}}}, \Delta u_{T_{\mathrm{III}}}$ fest definiert wird, nicht jedoch wenn eine kontinuierliche Änderung der Schrittweite während der Phasen $\Delta u_{T_{\mathrm{I}}}(k)$, $\Delta u_{T_{\mathrm{II}}}(k)$, $\Delta u_{T_{\mathrm{III}}}(k)$, abhängig vom Fortschritt k innerhalb des Segments vorliegt[4]. Die angepassten Schrittweiten pro Phase stellen sicher, dass der Lageregelkreis der Achse über den Verlauf der Trajektorie Sollwertsprünge in solcher Art erhält, dass diese aufgrund der Bewegungsdynamik möglichst geringe Anforderungen an den Lageregler stellen, vgl. (Binder 1979, S. 39) und damit einfach durchzusetzen sind.

Bei komplexeren Interpolationsformen wie der Kreisinterpolation gelten statt (5-13) in Abhängigkeit mit dem Kreiswinkel θ und dem Kreisradius R nach

$$\Delta u = R\Delta\theta \tag{5-15}$$

die Achsgeschwindigkeiten als

$$\vec{v} = \begin{pmatrix}v_x \\ v_y\end{pmatrix}\begin{pmatrix}\vec{\imath} \\ \vec{\jmath}\end{pmatrix} = v\begin{pmatrix}-\sin(\frac{v}{R}t) \\ \cos(\frac{v}{R}t)\end{pmatrix}\begin{pmatrix}\vec{\imath} \\ \vec{\jmath}\end{pmatrix} \tag{5-16}$$

und sind damit über die Winkelposition $\theta(t) = \omega t = \frac{v}{R}t$ achsübergreifend gekoppelt. In diesem Fall ist eine Entkopplung der Achsgeschwindigkeiten nur dann möglich, wenn die Kreisbogeninterpolation auf Sehnen linearisiert wird, auf denen dann wieder einzeln (5-14) gilt. Ein Ansatz, der anhand der Bahnkurve die Kopplung der Achsgeschwindigkeiten auch für nicht-lineare Interpolationskurven berechnet und diese verwendet, um den Bahnfehler auszugleichen, ist die in Kapitel 2.4.1 vorgestellte Bahnregelung nach (Huan 1982).

[4] Wenn im allgemeinen Fall die Aufteilung der Bahn in Schritte Δu mit einem gleichbleibenden Vorschub festgelegt wird, bedeutet dies bei einem festen Interpolationstakt, dass Wegdifferenzen, die kein ganzzahliges Mehrfaches dieser Schrittweite sind, mit einer Restdifferenz angegeben werden müssen. Diese Restdifferenz wird im Allgemeinen auf alle Phasen der Beschleunigung aufgeteilt (Altintas **Altintas 2012**, S. 219) und wird daher hier nicht näher betrachtet.

5.1.4 Kinodynamische Bewegungsplanung

Betrachtet man die soeben vorgestellte Trajektorienplanung, wird davon ausgegangen, dass die Dynamik, die für die Bahn vorgegeben wird, auch von allen beteiligten Achsen umgesetzt werden kann. Hierfür wird im Allgemeinen die maximale Bahngeschwindigkeit geringer als die einzelnen maximalen Achsgeschwindigkeiten, sowie die minimale Bahngeschwindigkeit höher als die einzelnen minimalen Achsgeschwindigkeiten gewählt (Binder 1979, S. 39).

$$v_{\max} < \min\left(v_{n,\max} \mid \forall n\right) \tag{5-17}$$

$$v_{\min} > \max\left(v_{n,\min} \mid \forall n\right) \tag{5-18}$$

Eine Überprüfung dieser Annahme ist im allgemeinen Fall einer Bahnbewegung im Raum nichttrivial und ist ein Optimierungsproblem mit den Achsdynamiken sowie der Transformation zwischen Bahn- und Achskoordinaten als Randbedingungen, vgl. (Dittrich et al. 1996, S. 62 - 65). Dadurch kann bei einer sich verändernden Achsdynamik, wie sie als eine Betriebsbedingung in Tabelle 4-3 festgestellt wurde, keine erneute Lösung des Optimierungsproblems der Bahndynamik durchgeführt werden. Es muss in der Folge mit einem Bahnfehler bzw. einem schlechten Bahnfolgeverhalten gerechnet werden. Wenn stattdessen die Achse eine individuelle und nach Bedarf angepasste Dynamikplanung durchführt, kann entsprechend der verfügbaren Dynamik eine maximale, vor allem aber durchführbare Bahndynamik abgeleitet werden. Dieses Konzept wird als kinodynamische Bewegungsplanung bezeichnet, vgl. (Frazzoli et al. 2002, S. 117).

Da innerhalb eines Mehrachssystems davon auszugehen ist, dass beteiligte Achsen unterschiedliche Dynamiken vorweisen und je nach Bahnsegment auch eine unterschiedliche anteilige Bewegung gefordert wird, muss also die Dynamikplanung der Gesamtbahn segmentweise an jeweils der Achse mit der geringsten Dynamik ausgerichtet werden, um eine durchführbare Bahnvorgabe für alle Achsen zu erreichen. Betrachtet man den Zusammenhang (5-12) und (5-13) aus dem Beispiel mit zwei an der Bewegung beteiligten Achsen x und y

$$\Delta u = v\, T_i, \tag{5-19}$$

$$\overrightarrow{\Delta u} = \Delta x\, \vec{\imath} + \Delta y\, \vec{\jmath} = \begin{pmatrix}\Delta x \\ \Delta y\end{pmatrix}\begin{pmatrix}\vec{\imath} \\ \vec{\jmath}\end{pmatrix}, \tag{5-20}$$

ist sichtbar, dass entweder die Schrittweite Δu, die Geschwindigkeit v oder die Interpolationszeit T_i geeignet ist, um die Dynamik in den Achsen auf Ebene der Bahn abzustimmen. Bezogen auf ein Segment $S_i = \overrightarrow{P_{i-1}P_i}$ kann formuliert werden:

$$\Delta S = v_S^* T_s = \int_0^{T_s} a_S(t) t\, dt, \tag{5-21}$$

dabei ist ΔS die Segmentlänge, v_S^* die gemittelte Geschwindigkeit über die Segmentlänge und T_s die Zeitdauer der Bewegung. Da eine synchronisierte Bewegung nach Kapitel 2.2.3 angestrebt wird, also die Bewegung gleichzeitig

begonnen ($t = 0$) und beendet ($t = T_s = T_{S,x} = T_{S,y}$) werden soll, kann mit (5-12) und (5-13)

$$\Delta S = v_S^* T_s = \begin{pmatrix} v_x^* \\ v_y^* \end{pmatrix} T_s \tag{5-22}$$

formuliert werden, wobei

$$v_n^* T_{S,n} = \int_0^{T_{S,n}} a_n(t) t \, dt \tag{5-23}$$

für alle Achsen n gilt. Hierbei bezeichnet $a_n(t)$ die Beschleunigung der Trajektorienplanung der Achse entsprechend ihres Beschleunigungsprofils. Im Falle des Trapezprofils gilt

$$a(T_1) = -a(T_3) = konst., a(T_2) = 0. \tag{5-24}$$

Die Segmentlänge ΔS ist direkt von den Stützpunkten der Bahnvorgabe nach Gleichung (5-4) abhängig. Der Zusammenhang zwischen der gemittelten Segmentgeschwindigkeit v_S^* und den gemittelten Achsgeschwindigkeiten im Segment v_x^*, v_y^* ist direkt abhängig von der Zeitspanne T_s des Segments. Das Verhältnis v_x^* / v_y^* wird durch die Projektionen $\mathcal{P}_i, \mathcal{P}_j$ des Segmentvektors $\overrightarrow{S_l}$ auf die Koordinatenrichtungen $\vec{\imath}, \vec{\jmath}$ bestimmt.

5.2 Trajektorienplanung als Multiagentensystem

Um eine synchronisierte Bewegung als MAS durchzuführen, ist ein zentraler Ansatz der Trajektorienplanung als Bahn und die Ableitung der darauf resultierenden Achstrajektorien nicht geeignet. Stattdessen soll geprüft werden, inwiefern die Trajektorienplanung auch über die kinodynamische Bewegungsplanung koordiniert werden kann. Es soll also umgekehrt durch die Kombination mehrerer Achstrajektorien eine gültige Bahntrajektorie entstehen.

Ein agiles System nutzt neben der globalen Bahnvorgabe auch die Dynamik der Einzelkomponenten bei der Planung der Trajektorien voll aus, vgl. (Frazzoli et al. 2002). Viele Bewegungsanwendungen, für die der Ansatz der Rekonfiguration eine hohe Priorität hat, benötigen nur eine moderate Bahntreue und legen stattdessen Wert auf eine hohe Dynamik der Bewegung. Im Gegensatz zu prozessnahen Anwendungen wie Fräs- oder Laserschneidprozessen, die sich durch eine sehr hohe Konturanforderung im Mikro- oder Nanometerbereich und einem monolithischen Maschinensystem ohne Möglichkeit zur Rekonfiguration auszeichnen, fallen die Anforderungen an die Kontur bahngesteuerter Handhabungsaufgaben in Anwendungen der Montage wie Werkstückmanipulationen, Pick-and-Place und andere prozessferne Bewegungen dagegen in der Größenordnung von Millimetern vergleichsweise moderat aus. Hier sind die vorgegebenen Bahnen im Raum der Kollisionsvermeidung mit anderen Maschinenteilen geschuldet. Die Bewegungsumsetzung mit maximaler Dynamik liegt im Fokus um die Prozesszeiten zu verkürzen.

Es soll daher statt einer exakten mathematischen Invertierung der Bahndynamik auf die Achsdynamik, wie sie die Bahnregelung nach (Huan 1982) durchführt, ein Agentenverhalten mit möglichst einfacher Umsetzung angestrebt werden. Dabei

soll der vorgestellte kinodynamische Zusammenhang aus Kapitel 5.1.4 ausgenutzt werden, nach dem entlang eines linearen Bahnsegments die Dynamikplanung unabhängig von der Position entlang des Segmentes stattfindet, also entkoppelt ist. Weitere Nebenbedingungen sind minimales Systemmodellwissen des Agenten und eine gute Verallgemeinerbarkeit des Ansatzes. Im Weiteren wird das Interaktionskonzept nach der Analyse aus Kapitel 4.5 auf die Trajektorienplanung übertragen, notwendiges Agentenverhalten abgeleitet und eine Kommunikationsstruktur vorgestellt, die diese Anforderungen erfüllen sollen. Die Interaktionsstruktur sowie das Kommunikationsprotokoll wurden in (Schöbel 2019; Dripke et al. 2020; Stanglmeier 2020) in ihren Ansätzen vorgestellt.

5.2.1 Kombination der Interaktionsmuster

In Kapitel 4.4.6 wurde über die DACS-Analyse und die vorgestellten Anwendungsfälle klar, dass sowohl das Interaktionsmuster Formationsbewegung als auch der Rendez-Vous-Ansatz übereinstimmende Charakteristiken für die verteilte Interpolation bieten (vgl. Analyse in Tabelle 4-7). Die Formation hält Agenten in festen, relativ zueinander definierten Abständen. Aus dieser Zwangsbedingung kann sich die komplette Formation nur dann fortbewegen, wenn alle Agenten der Bewegung folgen. Dies entspricht dem Konzept der vorgegebenen Bahn, die nur dann abgefahren werden kann, wenn alle Agenten gleichzeitig entlang ihrer jeweiligen Achskoordinaten verfahren. Die notwendigen Verfahrbewegungen einzelner Achsen lassen sich nach der vorgestellten homogenen Transformation aus Kapitel 5.1.1 von der Bahnvorgabe ableiten. So kann die Bahnvorgabe weiterhin in Weltkoordinaten definiert sein und die Agenten transformieren sich die Bedeutung für ihre individuelle Bewegung in ihre Achsräume. Da die Bahnvorgabe jedoch lediglich Stützpunkte der linearen Interpolation und keine kontinuierliche Wertvorgabe berechnet, muss zwischen den Stützpunkten eine Vorgabe für die Bewegung definiert werden. Statt zwischen den Stützpunkten kontinuierliche Zwischenpunkte für alle Achsen zu erzeugen, was einen hohen Rechen- und Kommunikationsaufwand nach sich zöge, kann das Rendez-Vous-Verfahren für diese Bewegung betrachtet werden. Das Rendez-Vous-Verfahren betrachtet die individuelle dynamische Planung zur Erreichung einer Sollposition und gibt den Agenten die Freiheit über den Zielpunkt zu verhandeln sowie nach Festlegung des Zielpunkts ohne weitere Abstimmungen mit den anderen Agenten eine individuelle Bewegungsplanung durchzuführen. Da für den Fall der linearen Interpolation nach obiger Darstellung die Bewegungsdynamik entkoppelt von der Position der Achsen ist, kann davon ausgegangen werden, dass über eine ähnliche Trajektorienplanung in den Achsen eine implizite Kopplung und eine daraus resultierende Synchronisierung und Bahntreue erreicht werden kann. Das kombinierte Interaktionsmuster der Formationsbewegung und des Rendez-Vous-Verfahrens ist in Abbildung 5-3 dargestellt. Alle Agenten bekommen die Sollposition in Bahnkoordinaten vorgegeben. Diese entspricht einer Formationsposition der Achsen in ihrer eigenen Achskoordinatensystemen, die von allen gleichzeitig eingenommen werden muss, um die Gesamtformation, also die Bahnposition im Raum zu erreichen. Bei Abweichungen von einer oder mehreren Achsen von der benötigten Position ist für das komplette Mehrachssystem die geforderte Position nicht zu erreichen. Um die Formationspositionen einzu-

nehmen, planen alle Achsen individuell ihre Bewegung zur Formationsposition. Vor der Durchführung der Bewegung wird über eine Rendez-Vous-Koordination der Zeitpunkt, zu dem die Formationsposition von allen beteiligten Achsen eingenommen werden soll, festgelegt. Die einzelnen Achsen passen bei Bedarf ihre Bewegungsplanung so an, dass sie den vereinbarten Zeitpunkt für das Erreichen der Formationsposition in ihrem Achsraum einhalten können. Die Bewegungsdurchführung beginnt gleichzeitig und führt durch die implizite Kopplung alle Achsen gleichzeitig in die vorgegebene Formation. Für die nachfolgende Formationsvorgabe beginnt dann die Koordination einer Konsenszeit erneut.

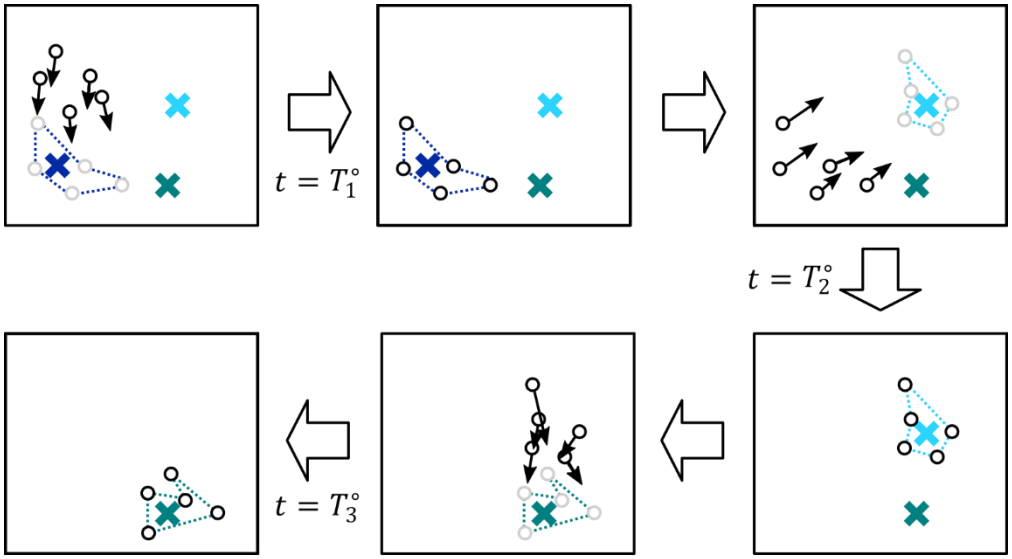

Abbildung 5-3: Interaktionsmuster aus Formation und Rendez-Vous kombiniert: Die Formation ist an den Stützpunkten definiert, dazwischen wird die Bewegung mit individueller Dynamikplanung als Rendez-Vous zur nächsten Stützpunktformation geplant. Die Bewegung zur Formation des folgenden Stützpunktes wird durch den Zustandsautomaten freigegeben, der Rendez-Vous-Zeitpunkt wird durch Konsens festgelegt, die Bewegungsdurchführung geschieht durch den Segmentregler (s. Kapitel 5.2.3).

Eine solche Kombination von Formationsbewegung und Rendez-Vous-Verfahren bedeutet vor allem, dass es keinen Interpolationstakt mit Kommunikation zwischen allen Achsen bedarf, wie er oft für die Bewegungssynchronisierung verwendet wird. Stattdessen kann die Bewegung zu einzelnen Formationen ereignisbasiert jeweils nach erfolgreichem Erreichen der vorherigen Formation geplant werden. Dabei wird die Zeit für die einzelnen Schritte nach Abbildung 5-3 dadurch bestimmt, welchen Zeitbedarf die beteiligten Agenten benötigen, um die nächste Formation nach dem Rendez-Vous-Verfahren einzunehmen. Die Wahl der Stützpunkte und deren räumlicher Abstand bestimmt den Zeitbedarf zwischen zwei Formationen. Es wird ereignisbasiert nur dann kommuniziert, wenn entweder ein neuer Stützpunkt verfolgt wird, ein Zielpunkt für das Rendez-Vous-Verfahren verhandelt wird oder wenn Fehler auftreten, kurz, wenn die in der Aufgabenanalyse in Tabelle 4-6 vorgestellten externen Entscheidungsauslöser oder Kommunikation nach dem Interaktionsmuster notwendig werden. Das Rendez-Vous-Verfahren hat zusätzlich den Effekt, dass die Bewegung selbst ohne zusätzliche Kommunikation durchgeführt werden kann und durch die Randbedingung der

Trajektorienplanung trotzdem eine implizite Kopplung erhalten bleibt. Im Idealfall, dass während der Durchführung der Bahn keine zeitlichen und geometrischen Abweichungen von dem erwarteten Achs- oder Systemverhalten auftreten, kann auf Kommunikation während der Ausführung sogar komplett verzichtet werden[5], was die Kommunikationslast im Produktionssystem zusätzlich senken kann.

5.2.2 Festlegung der Koordinationsvariable

Für das Rendez-Vous-Verfahren wird eine Koordinationsvariable festgelegt, nach der die Agenten jeweils in lokaler Planung ihre Bewegung optimieren. Für das kinodynamische Problem nach Kapitel 5.1.4 bietet sich die Analyse der Variablen aus Gleichung (5-22) an, um die implizite Kopplung der Achsen auszunutzen.

Eine Abstimmung der Segmentgeschwindigkeit über die gemittelten Geschwindigkeiten einzelner Achsen bedeutet jeweils die Berechnung der Achsgeschwindigkeit aus der Bahnvorgabe für alle beteiligten Achsen nach

$$\vec{S} = \begin{pmatrix} r_{x_i} - r_{x_{i-1}} \\ r_{y_i} - r_{y_{i-1}} \end{pmatrix} \begin{pmatrix} \vec{i} \\ \vec{j} \end{pmatrix} = T_s \begin{pmatrix} v_x^* \\ v_y^* \end{pmatrix} \begin{pmatrix} \vec{i} \\ \vec{j} \end{pmatrix} \tag{5-25}$$

$$v_x^* = \frac{r_{x_i} - r_{x_{i-1}}}{r_{y_i} - r_{y_{i-1}}} \, v_y^*. \tag{5-26}$$

Für den Ablauf einer Konsensfindung über eine gemeinsame Bahngeschwindigkeit müsste also jeder Agent pro Konsensfindung die Koordinatentransformation der eigenen Achsgeschwindigkeit in die Bahngeschwindigkeit vornehmen und bei einer Anpassung der Bahngeschwindigkeit erneut die Durchführbarkeit der Bewegung anhand der Transformation auf die eigene Achsgeschwindigkeit prüfen.

Stattdessen kann die Zeitspanne T_s des Segments S_i ohne eine Transformation in Achskoordinaten zur Konsensfindung verwendet werden. Jede Achse n kann für ein Segment S_i die approximiert benötigte Zeit T_n^* bei einer angenommenen mittleren Geschwindigkeit v_n^* durch

$$T_{S,n}^* = \frac{r_{n_i} - r_{n_{i-1}}}{v_n^*} \tag{5-27}$$

feststellen. Dabei werden nur die bereits transformierten zeit- und dynamikunabhängigen Achskoordinaten r_{n_i} der Sollwerte benötigt. Die Konsensfindung der

[5] Ein kompletter Verzicht auf Kommunikation ist für die Durchführung nach dem Interaktionsmuster Rendez-Vous möglich, in den meisten Anwendungsfällen werden allerdings aufgrund von Anforderungen der funktionalen Sicherheit weiterhin sogenannte zyklische Heartbeatsignale kommuniziert, um sicherzustellen, dass alle an der Bewegung beteiligten Systeme weiterhin ansprechbar und funktional sind. Sollten Ausfälle einzelner Achsen unerkannt bleiben kann ansonsten eine Kollision im Bewegungsraum nicht ausgeschlossen werden. Die Heartbeatsignale haben jedoch keinen funktionalen Hintergrund für die Bewegungssynchronisierung, denn sie übertragen keine Applikationsdaten und werden unabhängig von einem spezifischen Interaktionsverhalten benötigt.

Agenten kann unter diesen Voraussetzungen ohne Koordinatentransformationen durchgeführt werden.

Die Konsenszeit

$$T_S^\circ = \max_{\forall \mathrm{n}} \{T_{S,n}^*\} \tag{5-28}$$

kann wiederrum direkt in die benötigte mittlere Achsgeschwindigkeit jeder Achse überführt werden:

$$v_n^\circ = \frac{r_{n_i} - r_{n_{i-1}}}{T_S^\circ}. \tag{5-29}$$

Es wird daher eine Konsensfindung auf Basis der Zeitbedarfsbestimmung nach (5-27) aller Segmente der Bahnvorgabe festgelegt. Die Segmentzeit als Koordinationsvariable ermöglicht den Achsen, entsprechend ihrer individuellen Dynamikplanung, die verfügbare Zeit zum Erreichen des Zielpunkts des Segments miteinander abzustimmen und bei Bedarf (s. Tabelle 4-6) anzupassen. Unter der Annahme, dass die beteiligten Achsen ähnliche Beschleunigungsprofile verwenden und die Bewegung der Achsen lineares Verhalten, d.h. eine zielgerichtete Verfahrrichtung, aufweisen, kann von einer impliziten Kopplung über die Bewegungszeitspanne T_n° gesprochen werden.

Wenn der kinematische Zusammenhang keine Singularitäten aufweist, kann wie beim Ansatz des CCD davon ausgegangen werden, dass eine zielgerichtete Bewegung innerhalb der Achskoordinaten gleichzeitig eine zielgerichtete Bewegung innerhalb der Bahnkoordinaten zur Folge hat. Gleichzeitig ermöglicht die Festlegung der Koordinationsvariable als Zeitbedarf die Erweiterung und Interaktion auch mit nicht-positionierenden Komponenten. So können beispielsweise Prozesse wie Bilderfassung, Zuführungskomponenten, Pressen, Beschriftungslaser, etc. ebenfalls über den Zeitbedarf in die Konsensfindung der Achsen integriert werden. Durch die Konsensfindung eines Maximums aller Koordinationsvariablen pro Segment ist die Anzahl der beteiligten Achsen oder Komponenten nach oben nicht beschränkt. Gleichzeitig ist durch die Abstimmung über die gemeinsame maximale Zeitspanne sichergestellt, dass das Gesamtsystem mit der maximalen, real durchführbaren Dynamik verfährt. Daraus resultiert die Einschränkung, dass bei der Ausnutzung der maximalen Dynamik über die Koordination der Zeitspanne keine direkten Rückschlüsse auf die Bahngeschwindigkeit möglich sind, weshalb dieser Ansatz nicht für Anwendungen geeignet ist, bei denen eine variable Bahngeschwindigkeit prozesstechnische Einbußen zur Folge hat (z. B. Fräs- oder Schweißprozesse).

Die Koordinationsvariablen werden für alle Segmente ähnlich wie die Sollwertvorgabe in einem Vektor zusammengefasst, sodass jeder Eintrag zum entsprechenden Segment der Sollwertvorgabe korrespondiert. Es wird daher ein Koordinationsvektor kommuniziert. Die Konsenszeiten können dann nach Erhalt aller Vektoren als Maxima der Spalten berechnet und erneut in einem Vektor, dem Konsensvektor, gespeichert werden. Die Bedeutungen der Sollwertvorgabe der Formation S_i, der lokalen Koordinationsvektoren $T_{i,n}^*$ sowie des globalen Konsensvektors T_i° sind in Abbildung 5-4 erläutert.

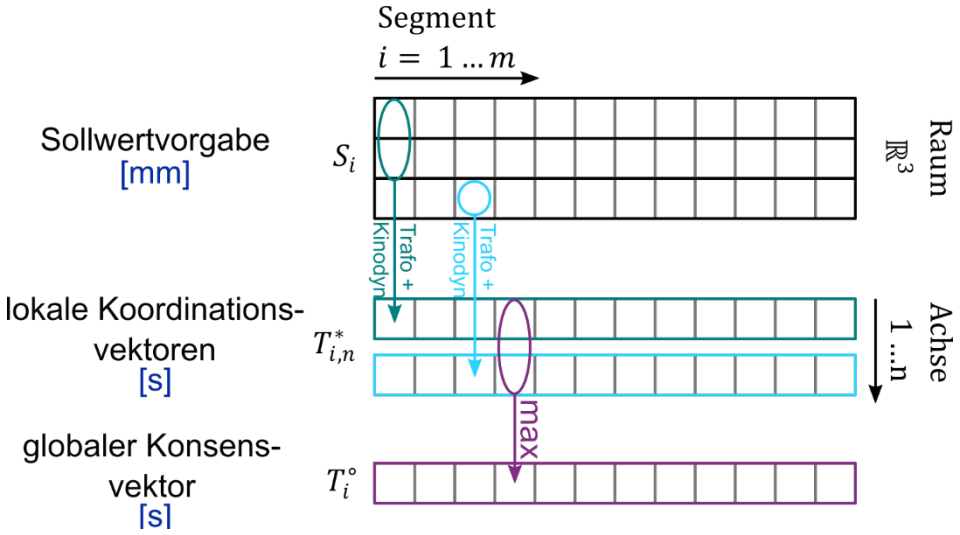

Abbildung 5-4: Bedeutung und Zusammenhang der Sollwertvorgabe, Koordinationsvektoren und Konsensvektor. Die Sollwertvorgabe der Formation S_i ist eine Matrix der Segmente mit Positionsbeschreibungen in $\mathbb{R}^3$, die lokalen Koordinationsvektoren $T^*_{i,n}$ werden pro Achse über die Transformation und kinodynamische Planung berechnet. Der globale Konsensvektor T°_i wird aus den maximalen Einträgen der Koordinationsvektoren pro Segment berechnet.

5.2.3 Agentenverhalten

Damit die Achsen über die oben festgelegte Koordinationsvariable agieren können, wird die Agentenstruktur aus Kapitel 0 auf den Anwendungsfall übertragen. Die Ein- und Ausgänge zur Erfassung der Umwelt und Kommunikation mit weiteren Agenten sowie die Beeinflussung der Umwelt wurden in Kapitel 4.3.4 analysiert. Es steht noch die formale Beschreibung der Handlungsstrategie nach obigem Interaktionsmuster aus.

5.2.3.1 Interaktionsmuster

Das Interaktionsmuster als Kombination der Formationsbewegung und des Rendez-Vous-Verfahrens wird für alle Achsen umgesetzt, wie im Folgenden beschrieben und in Abbildung 5-5 dargestellt. Es folgt dem Grundmuster der Agentenarchitektur aus Abbildung 4-5 nach dem Verknüpfung-Auswahl-Ausführungs-Prozess. Eine vereinfachte Darstellung des Ablaufs findet sich in Abbildung 5-6.

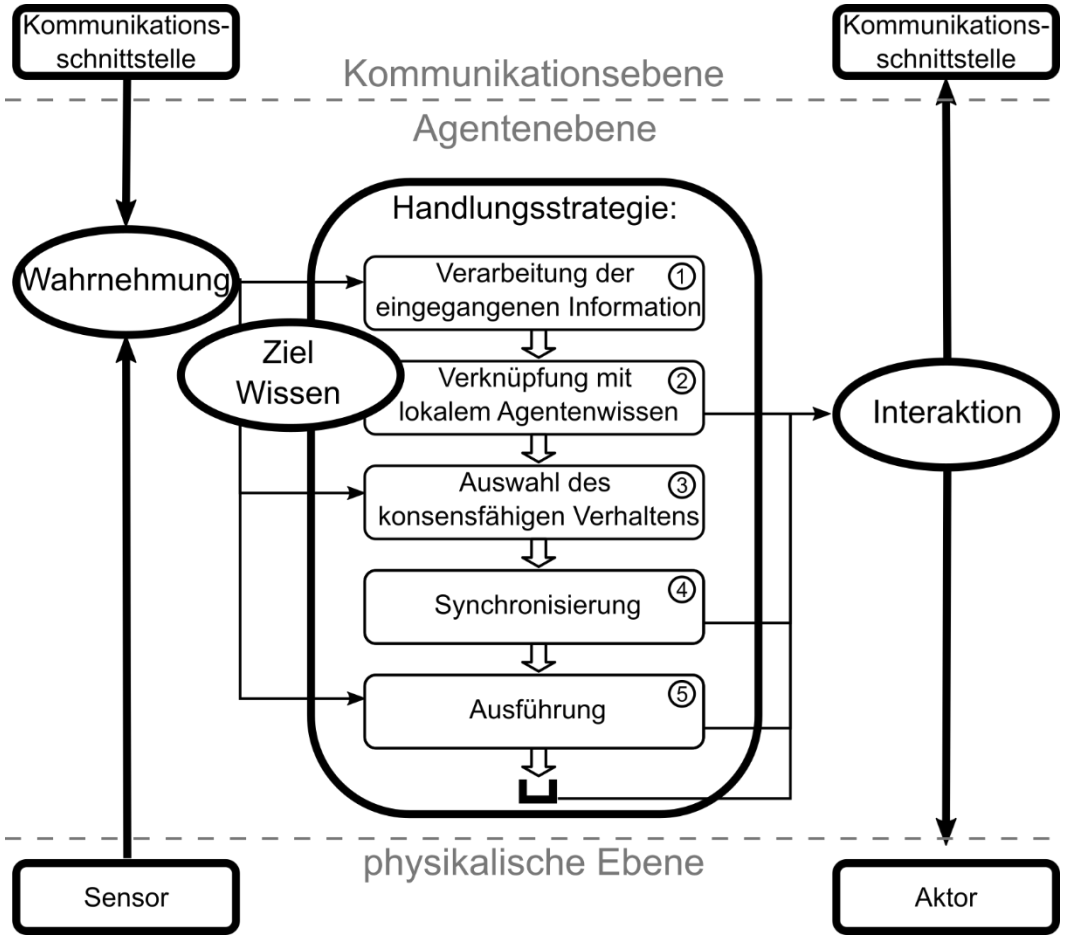

Abbildung 5-5: Agentenstruktur und -verhalten für die verteilte Interpolation.Erweiterter Verknüpfung-Auswahl-Ausführung Prozess angelehnt an (Ishida 1994, S. 68), Weiterentwicklung der Abbildung 4-5.

Init: Initialisierung:
Die Achse registriert sich für eine Bewegung im Verbund und erhält die Bahnvorgabe als Stützpunkte in Weltkoordinaten.
[Definition der Formation $\{S_i \mid i = 1 \dots m\}$]

1. Verarbeitung der eingegangenen Information:
   Die Achse identifiziert über homogene Transformationen und definierte Arbeitsbereiche segmentweise die notwendigen Achsbewegungen.
2. Verknüpfung mit dem lokalen Agentenwissen:
   Die Achse berechnet segmentweise die benötigte Zeit für die Bewegungen.
   [Berechnung der lokalen Koordinationsvariable $T^*_{i,n}$ pro Segment; Koordinationsvektor der Formation]
3. Auswahl des konsensfähigen Verhaltens:
   Die Achse erhält die Koordinationsvektoren aller weiteren Achsen und identifiziert das Maximum der Koordinationsvariable pro Segment.
   [Berechnung des Konsensvektors $T^\circ_i = \max_{\forall n} \{T^*_{i,n}\}, i = 1 \dots m$ der Formation]

4. Synchronisierung:
   Die Achse kommuniziert ihre Bereitschaft zur Bewegung im ersten Segment.
   [Startpunkt S_1 der Formation]

Für $i = 1 \ldots m - 1$ wiederhole:

5. Ausführung:
   Die Achse bewegt sich entsprechend des Konsenses bis zum Ablauf der Konsenszeit gleichmäßig mit der gemittelten Geschwindigkeit zur Zielposition des Segments
   [Rendez-Vous auf Zielposition zur vereinbarten Zeit (Konsens der Koordinationsvariable); Zwischenpunkt S_i der Formation]

dann: Übergang zum nächsten Segment:
Die Achse kommuniziert ihre Bereitschaft zur Planung und Bewegung im nächsten Segment.
[Zwischenpunkt S_{i+1} der Formation]

Für $i = m$:

5. Ausführung:
   Die Achse bewegt sich entsprechend des Konsenses bis zum Ablauf der Konsenszeit gleichmäßig mit der gemittelten Geschwindigkeit zur letzten Zielposition der Bahnvorgabe
   [Rendez-Vous auf Zielposition zur vereinbaren Zeit (Konsens der Koordinationsvariable); Zielpunkt S_m der Formation]

dann: Beenden der Bewegung:
Die Achse kommuniziert das Ende der Bahnvorgabe.

Bei Änderungen der lokalen Achsdynamik (Schritt 2), speziell bei verringerter Dynamik kann jederzeit durch eine erneute Berechnung der Koordinationsvariable im Segment der Austausch eines neuen Koordinationsvektors (Schritt 3) der Konsens auch während der Durchführung der Bewegung (Schritt 5) aktualisiert werden.

Speziell der Planungsschritt (Schritt 2) und die Ausführung (Schritt 5) werden daher nachgehend näher betrachtet.

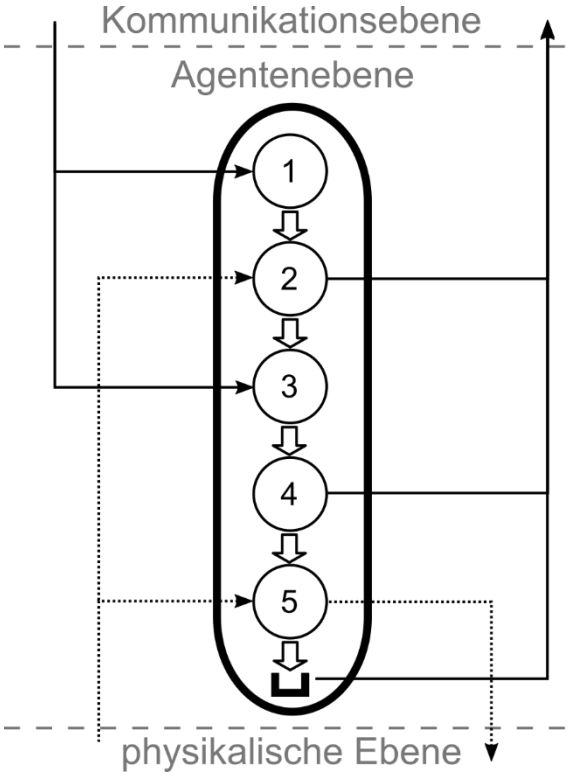

Abbildung 5-6: Vereinfachte Agentenstruktur der Abbildung 5-5 in Schritten des Interaktionsmusters.

5.2.3.2 Planung der lokalen Koordinationsvariablen (Schritt 2)

Die Konsensfindung der Agenten wird über die Koordinationsvariablen T_{n_i} durchgeführt, die für jedes Segment S_i die Konsenszeit $T_{S_i}^{\circ}$ festlegen. Für jedes Segment betrachtet der Agent in lokalen Achskoordinaten r_{n_i} die zu verfahrende Weglänge L. Entweder aus der Beobachtung des eigenen Verhaltens in zeitlich zurückliegenden Bewegungen oder über die gespeicherten Metainformationen hat der Agent Kenntnis über seine eigene Dynamik und kann über eine angenommene mittlere Geschwindigkeit v_n^* durch

$$T_n^* = \frac{r_{n_i} - r_{n_{i-1}}}{v_n^*} \tag{5-30}$$

den individuellen Zeitbedarf T_n^* für die Durchführung eines Segments berechnen.

Die Annahme einer mittleren Geschwindigkeit v_n^* ist eine Vereinfachung, da je nach Bewegungsprofil der Zusammenhang zwischen der benötigten Zeit T_n und der zu verfahrenden Weglänge L an Komplexität zunimmt. Es müssen beispielweise beim sehr einfachen Trapezprofil die Zeitspannen der Phasen $T_{\mathrm{I}}, T_{\mathrm{II}}, T_{\mathrm{III}}$ festgelegt werden. Nach (Altintas 2012, S. 217), ist bei einem gleichbleibenden Interpolationstakt T_{Ipo} unter der Annahme, dass die Beschleunigung $a_{\max}$ und

Verzögerung $-a_{\max}$ gleiche Beträge haben, nach Umformung der Gleichung (5-11) folgende Berechnungen notwendig

$$T_{\mathrm{I}} = \frac{v_{\max} - v_{\mathrm{Start}}}{a_{\max}} \tag{5-31}$$

$$T_{\mathrm{III}} = \frac{v_Z - v_{\max}}{-a_{\max}} \tag{5-32}$$

$$T_{\mathrm{II}} = \frac{L}{v_{\max}} + \left[-\frac{v_{\max}}{a_{\max}} + \frac{1}{v_{\max}}\left(\frac{v_{\mathrm{Ziel}}^2 + v_{\mathrm{Start}}^2}{2a_{\max}}\right)\right]. \tag{5-33}$$

Daraus folgt für die Gesamtdauer einer Bewegungsdurchführung nach dem Trapezprofil

$$\begin{aligned} T &= T_{\mathrm{I}} + T_{\mathrm{II}} + T_{\mathrm{III}} \\ &= \frac{2v_{\max} - v_{\mathrm{Start}} - v_{\mathrm{Ziel}}}{a_{\max}} + \frac{L}{v_{\max}} + \left[-\frac{v_{\max}}{a_{\max}} + \frac{1}{v_{\max}}\left(\frac{v_{\mathrm{Ziel}}^2 + v_{\mathrm{Start}}^2}{2a_{\max}}\right)\right]. \end{aligned} \tag{5-34}$$

Für andere Beschleunigungsprofile wie beispielsweise das Siebenphasenprofil werden diese Zeitberechnungen komplexer und umfassen teilweise die Lösung von Gleichungssystemen oder die numerische Approximation der Zeitspannen der Phasen eines Beschleunigungsprofils, s. (Altintas 2012, S. 224–226). Wird stattdessen die benötigte Zeit mit einer mittleren Geschwindigkeit approximiert, gilt die einfachere Gleichung (5-30).

Diese Vereinfachung ermöglicht es, die Konflikte zwischen Agenten, die durch unterschiedliche Bewertungen der geteilten Information entstehen können, zu vermeiden. Eine einfache Berechnungsvorschrift stellt sicher, dass alle Agenten schnell zu einem Ergebnis mit gleicher Bedeutung kommen. Sollten einzelne Agenten durch erhöhte Rechenkapazitäten noch genauere Zeitbedarfe errechnen können, wird die Information trotzdem in gleicher Weise, als einen approximierten Wert, bewertet und Abweichungen in der Durchführung in gewissem Maße antizipiert.

Anschaulich wird die Annahme einer mittleren Geschwindigkeit v^* durch die Betrachtung der Fläche unter der Geschwindigkeitskurve eines Trapezprofils, s. Abbildung 5-7. Die zurückgelegte Weglänge ist mit

$$l(t) = v(t)t \tag{5-35}$$

definiert und wird durch die Fläche $A\big(l(t)\big)$ unter der Kurve repräsentiert. Für $t = T$ muss die komplette Weglänge $L = l(T)$ zurückgelegt worden sein. Die Annahme einer gemittelten Geschwindigkeit

$$l^*(t) = v^*t \tag{5-36}$$

stimmt dann,

wenn die Flächen unter der Kurve

$$A\big(l(t)\big) = A\big(l^*(t)\big) \text{ für } t = T \tag{5-37}$$

gleich groß sind.

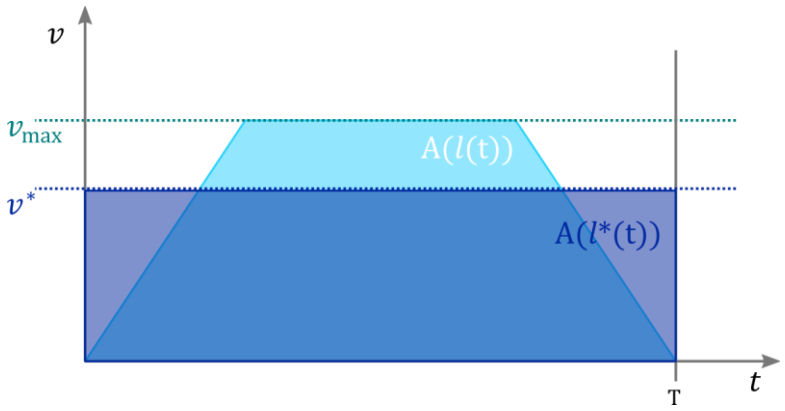

Abbildung 5-7: Verwendung der gemittelten Geschwindigkeit v^* zur Approximierung des Zeitbedarfs eines Segments.

Es wird für die vereinfachte Durchführung der Konsensfindung die Approximation einer mittleren Geschwindigkeit als Faktor $v_n^* = 0.8 \cdot v_{n,\max}$ festgelegt. Dadurch ist für den nachfolgend vorgestellten Segmentregler eine Dynamikreserve vorhanden um Fehlberechnungen des tatsächlichen Zeitbedarfs durch die approximierte mittlere Geschwindigkeit statt der Berechnung eines konkreten Beschleunigungsprofils auszugleichen. Ebenfalls kann durch diese Annahme davon ausgegangen werden, dass unabhängig vom implementierten Beschleunigungsprofil alle Agenten die Berechnung der Koordinationsvariable nach derselben Rechenvorschrift durchführen. Eine Diskussion der Approximationsfehler dieser Annahme wird im Anhang durchgeführt.

5.2.3.3 Durchführung der Konsensbewegung (Schritt 5)

Nachdem im Schritt 3 der Konsensvektor $\{T_i^\circ | i = 1 \dots m\}$ berechnet wurde, gilt

$$T_i^\circ \geq T_{i,n}^*. \tag{5-38}$$

In Segmenten, in denen andere Achsen entweder eine größere Weglänge zurücklegen oder aber eine geringere Dynamik besitzen wird durch den Konsens damit die Zeitspanne zur Durchführung des Segments vergrößert, es gilt dann

$$T_i^\circ > T_{i,n}^*. \tag{5-39}$$

Für die Durchführung der Konsensbewegung muss die Achse daher den Skalierungsfaktor $c_{i,n}$ mit

$$c_{i,n} = \frac{T_i^\circ}{T_{i,n}^*} \geq 1 \tag{5-40}$$

bestimmen.

Daraus folgt die mittlere Sollgeschwindigkeit als

$$v_{i,n}^\circ = \frac{v_{i,n}^*}{c_{i,n}}. \tag{5-41}$$

Wenn die Achse n mit der mittleren Sollgeschwindigkeit $v_{i,n}^\circ$ für das Segment S_i verfährt, verfährt sie in der Zeitspanne T_i° um die gewünschte Weglänge L. Nur

in Fällen, in denen die betrachtete Achse für die zu verfahrende Weglänge von allen an der Bewegung beteiligten Achsen die größte Zeitspanne $T_{i,n}^{*}$ als Koordinationsvariable kommuniziert ist $v_{i,n}^{\circ} = v_{i,n}^{*}$, da $T_i^{\circ} = T_{i,n}^{*}$.

Um sicherzustellen, dass trotz der angesprochenen Approximationsfehler durch die gemittelte Geschwindigkeit und bei Abweichungen von der angenommenen Dynamik der Zielpunkt zum Konsenszeitpunkt erreicht wird, wird im Lageregelkreis der Achse eine schaltende Geschwindigkeitsvorsteuerung eingesetzt, die nachfolgend als Segmentregler bezeichnet wird, s. Abbildung 5-8.

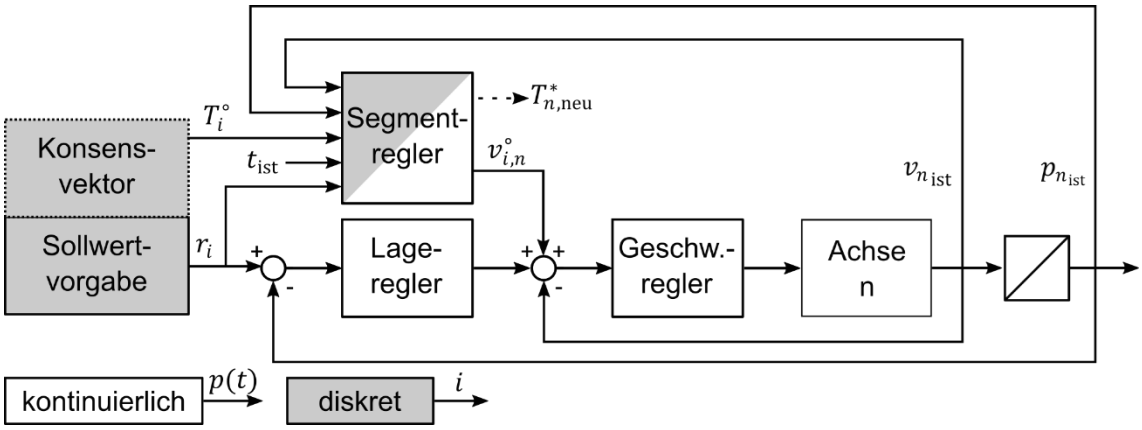

Abbildung 5-8: Blockschaltbild des Segmentreglers als Geschwindigkeitsvorsteuerung.

Dem Segmentregler stehen die Zielzeit T_i°, die Zielposition r_i sowie die aktuelle Zeit t_{ist} zur Verfügung. Im allgemeinen Fall schaltet der Segmentregler $v_{i,n}^{\circ}$ als Geschwindigkeitsvorsteuerung im Lageregelkreis zu. Über den Verlauf des Segments wird außerdem durchgehend die aktuelle Position p_{ist} und Geschwindigkeit v_{ist} beobachtet und die benötigte Restzeit

$$T_{\text{Rest}} = T_i^{\circ} - t_{\text{ist}} = \frac{r_i - p_{\text{ist}}}{v_{\text{ist}}} \tag{5-42}$$

für das Segment berechnet. Sollte

$$T_{\text{Rest}} + t_{\text{ist}} \gg T_i^{\circ} \tag{5-43}$$

gelten, wird die Geschwindigkeitsvorsteuerung angepasst, also die Berechnungsvorschrift umgeschaltet, auf

$$v_{i,n}^{+} = \frac{r_i - p_{\text{ist}}}{T_{\text{Rest}}}, \tag{5-44}$$

wobei gilt $v_{i,n}^{+} \leq v_{n,\max}$.

Sollte die Berechnung nach (5-44) zu $v_{i,n}^{+} > v_{n,\max}$ führen, wird auf Basis der aktuellen Geschwindigkeit v_{ist} eine neue Koordinationsvariable

$$T_{n,\text{neu}}^{*} = \frac{r_{n_i} - p_{n_{\text{ist}}}}{v_{\text{ist}}} \tag{5-45}$$

berechnet, den anderen Agenten kommuniziert und nach Schritt 3 im Agentenverhalten ein neuer Konsens

$$T^{\circ}_{i,\mathrm{neu}} = \max_{\forall \mathrm{n}} \{T^{\circ}_{i}, T^{*}_{i,\mathrm{neu}}\}, i = 1 \dots m \tag{5-46}$$

berechnet, der dann nach Schritt 4 im Segmentregler als neue Zielzeit $T^{\circ}_{i,\mathrm{neu}}$ eingesetzt wird.

Die Toleranz in (5-43) sollte so hoch gesetzt werden, dass innerhalb der Differenzzeit die kompletten Schritte 2 bis 4 inklusive der Verzögerungszeiten der Kommunikation abgehandelt werden können. Speziell während Phasen im Beschleunigungsprofil, die keine konstante Geschwindigkeit haben, sind Abweichungen durch die approximierte mittlere Geschwindigkeit zu erwarten. Ebenfalls muss das nicht ideale Folgeverhalten der Achsen berücksichtigt werden, da die Sollwerte diskret (nicht-kontinuierlich) vorgegeben werden und keine zusätzliche Vorsteuerungen eingesetzt werden und daher ohnehin mit einer verzögerten Reaktion auf Vorgabewerte gerechnet werden muss, s. auch (Binder 1979, S. 36).

Durch die Festlegung der zur Berechnung der Koordinationsvariablen angenommenen Geschwindigkeit von $v^{*}_{n} = 0.8 \cdot v_{n,\max}$ ist dem Segmentregler nach (5-44) die Ausnutzung der restlichen Dynamik zum Ausgleich solcher Fehler ermöglicht. Der Segmentregler ist somit in der Lage, sowohl auf ein nichtideales Folgeverhalten, als auch auf gegebenenfalls auftretende Abweichungen von der approximierten Geschwindigkeit zu reagieren.

Zusammengefasst hat der Segmentregler folgendes Interaktionsmuster:

1. $T_{\mathrm{Rest}} + t_{\mathrm{ist}} \approx T^{\circ}_{i}$: Vorsteuerung des Geschwindigkeitsregelkreises mit $v^{*}_{i,n}$
2. $T_{\mathrm{Rest}} + t_{\mathrm{ist}} \gg T^{\circ}_{i}$, $v^{+}_{i,n} \leq v_{n,\max}$: Vorsteuerung des Geschwindigkeitsregelkreises mit $v^{+}_{i,n}$
3. $T_{\mathrm{Rest}} + t_{\mathrm{ist}} \gg T^{\circ}_{i}$, $v^{+}_{i,n} > v_{n,\max}$: Berechnung einer neuen Koordinationsvariable $T^{*}_{n,\mathrm{neu}}$, Anstoßen einer Synchronisierung aller Agenten nach Schritten 2 bis 4 im Interaktionsmuster.

Damit ist der Segmentregler kein kontinuierlicher Regler im klassischen Sinne, sondern besitzt eine schaltende Steuerungslogik, die je nach Zustand des Agentensystems andere Berechnungen durchführt, beziehungsweise externe Prozesse anstößt. Das bedeutet, dass die Geschwindigkeitsvorgaben zwar kontinuierlich berechnet werden, aber bei bestimmten Schaltbedingungen sich diskret ändern können. Da die Werte der Vorsteuerung sich aber hierbei maximal bis $v^{+}_{i,n} \leq v_{n,\max}$ verändern, bleibt die Stabilität des Positionsregelkreises erhalten.

Zu bemerken ist dies, wenn die Geschwindigkeitsvorsteuerung aus der Definition für das Segment über die gesamte Länge des Segments S_i einen Wert $v^{*}_{i,n} \geq 0$ vorgibt. Das bedeutet speziell für $v^{*}_{i,n} > 0$, dass eine Restgeschwindigkeit bei Erreichen der Zielposition r_i verbleibt, und daher ein Überschwingen der Lage erwartet werden muss, bis der Regelkreis die Achse an der Zielposition zum Stillstand bringt. Da die Bewegungssynchronisierung direkt bei Erreichen einer Zielposition r_i unabhängig von Restgeschwindigkeiten oder weiteren kontinuierlichen Zuständen im Regelkreis diskret die Vorgaben für das Segment S_{i+1} schaltet, wird dadurch entweder direkt die neue Geschwindigkeitsvorgabe $v^{*}_{i+1,n}$

verwendet, oder (falls ein Stillstand der Achse aufgrund von $r_i = r_{i+1}$ erwartet wird) $v^*_{i+1,n} = 0$ gesetzt.

Im Falle, dass die diskrete Schaltung auf die Vorgaben des nachfolgenden Segments der Geschwindigkeitsvorsteuerung exakt zu dem Zeitpunkt durchgeführt wird, zu dem der Zielpunkt des Segments erreicht ist, muss mit einem Überschwingen der Lage und danach für den Zeitraum der Interaktion der Achsen bis zum Start des neuen Segmentes mit Stillstand der Achsen gerechnet werden. Stattdessen kann eine vorgezogene Schaltbedingung bereits vor Erreichen des Zielpunktes des aktuellen Segmentes dafür verwendet werden, dass die diskreten Elemente des Regelkreises, also die Sollwertvorgabe sowohl als auch die Geschwindigkeitsvorsteuerung des Segmentreglers bereits verfrüht die Vorgabewerte des Nachfolgesegmentes als Vorgaben an den Regelkreis geben. Der Zeitpunkt, zu dem alle Achsen die neue Sollwertvorgabe durchsetzen wird wie im Normalverhalten in Schritt 4 des Interaktionsmusters synchronisiert. Dadurch wird einerseits das Überschwingen der Lage und andererseits der Stillstand zwischen der Abarbeitung aufeinanderfolgender Segmente vermieden. Es ist dadurch also implizit ein Überschleifmechanismus implementiert. Der geeignete Zeitpunkt für die vorgezogene Schaltbedingung der diskreten Komponenten des Regelkreises sollte, korrelierend zur Toleranz aus (5-43), entsprechend der Verhandlungszeiten der Agenten ausgerichtet werden und für alle Agenten der Interaktion identisch gewählt werden. Der Ablauf der Agentenstati und der Bewegung bei einer vorgezogenen Schaltbedingung wird nach der Erläuterung des verwendeten Kommunikationsprotokolls in Abbildung 5-10 in der Zusammenfassung des Synchronisierungsmechanismus dargestellt.

5.2.4 Kommunikationsprotokoll

In der Aufgabenanalyse des Anwendungsfalls aus Kapitel 4.3.4 wurde bereits festgehalten, dass alle Achsen über eine Kommunikationsschnittstelle mit dem Produktionsnetzwerk verbunden sind und daher mit allen weiteren Komponenten im Netzwerk Nachrichten und Daten austauschen können. Da industrielle Produktionsnetzwerke inzwischen auch Konzepte für die echtzeitfähige, deterministische Kommunikation parallel zu weiterem Netzwerkverkehr in der Produktion umsetzen, vgl. (Monostori et al. 2016, S. 629), soll hier nicht näher auf die Kommunikationsarchitektur eingegangen werden. Es wird stattdessen angenommen, dass mithilfe von Konzepten aus z. B. der Motion-Arbeitsgruppe der OPC UA Field Level Communications geeignete Kommunikationsstrukturen bereits existieren oder sich im Aufbau befinden, die Funktionalitäten wie Echtzeit, Motionanwendungen oder auch funktionale Sicherheit in konvergenten Netzen über einen herstellerunabhängigen Kommunikationsstandard abdecken, vgl. (Lutz 2020). Ebenso kann davon ausgegangen werden, dass die Uhren der Miniatursteuerungen mithilfe etablierter Methoden wie z. B. der Spezifikation nach IEEE 1588 synchronisiert werden können. Es soll hier deshalb lediglich das Kommunikationsprotokoll spezifiziert werden, dass die Umsetzung der Interaktionsmuster aus Kapitel 5.2.3 ermöglicht.

Die Kommunikationsschnittstelle baut auf der Zustandsmaschine aus Abbildung 5-6 auf. Die Kommunikation in den Schritten des Interaktionsmusters ist in Tabelle 5-1 und in Abbildung 5-9 dargestellt.

Zustand	***eingehende Kommunikation*** von allen weiteren Achsen im Verbund	***ausgehende Kommunikation*** an alle weiteren Achsen im Verbund
Init	[S] Registrierung weitere Achsen im Verbund	[S] Registrierung im Achsverbund
1	[D] Sollwertvorgabe → 1 (von der übergeordneten Aufgabenkoordination)	→ 2
2	[D] Koordinationsvektoren → 3	[D] Koordinationsvektor
3	-	→ 4
4	[S] Bereit zur Bewegung → 5	[S] Bereit für Segment
5		→ 6 // → 2
6	[S] Segment abgeschlossen (→ 1)	[S] Segment abgeschlossen

→ = ausgelöste Transition nach Erhalt aller Informationen, [D] = Datenaustausch, [S] = Statusaustausch

Tabelle 5-1: Kommunikation als Teil des Agentenverhaltens

Das Kommunikationsprotokoll nach (Stanglmeier 2020) enthält standardmäßig in den Nachrichten lediglich den Zustand der Agenten im Interaktionsmuster. Nur bei Bedarf werden Datenpakete, wie die Sollwertvorgaben oder der Koordinationsvektor angehängt. Dadurch ist bei der Durchführung der Bewegung außer bei Abweichungen von der geplanten Dynamik (siehe Schritt 5 → 2, Neuberechnung und Austausch der Kommunikationsvektoren) keine weitere Datenübertragung neben dem Austausch der Stati notwendig. Weiter benötigt die Kommunikation des Status neben den Kopfdateien der Nachricht nur eine feste Größe, während die Datenkommunikation je nach Umfang der Sollwertvorgabe einen deutlich größeren Umfang an Daten versendet. Die reine Statuskommunikation verursacht durch ihre kompakte, gleichbleibende Nachrichtengröße keine hohe Last im Produktionsnetzwerk.

Kopf der Nachricht	***Körper der Nachricht***	
Empfänger-/Versender-identifikation	[S] Status	[D] Daten – *bei Bedarf*
feste Größe	feste Größe	variable Größe

Tabelle 5-2: Aufbau einer Kommunikationsnachricht.

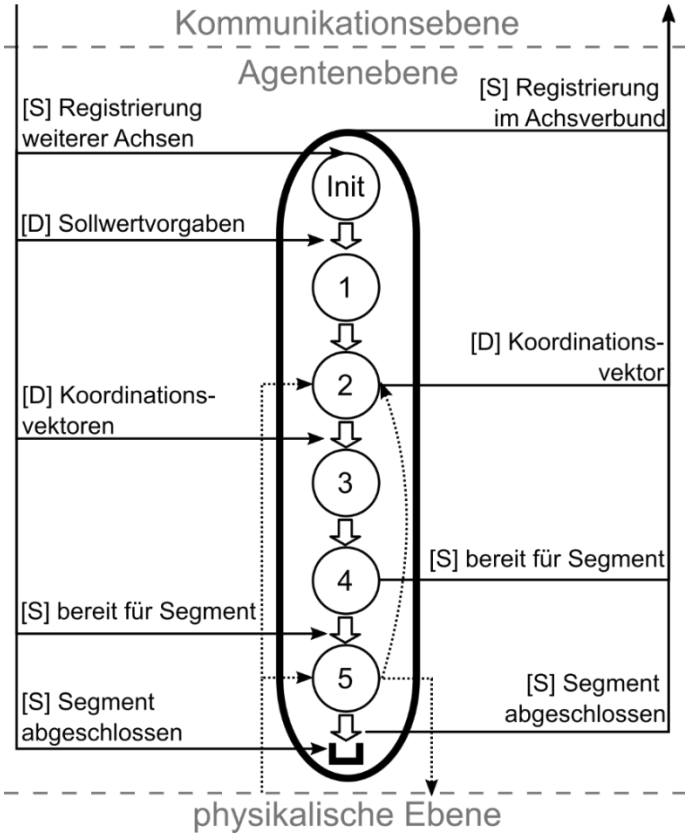

Abbildung 5-9: Agentenstruktur mit Kommunikationsprotokoll basierend auf der vereinfachten Agentenstruktur aus Abbildung 5-6.

5.3 Zusammenfassung des Interaktionsmusters

Das vorgestellte Interaktionsmuster entstand auf Basis der Analysen aus Kapitel 4.3. Eine Kombination der Konzepte der Formationsbewegung und des Rendez-Vous-Verfahrens ermöglicht die synchronisierte Bewegung der Agenten. Das Agentenverhalten ist durch das Interaktionsmusters sowie das Kommunikationsprotokoll spezifiziert. In Abbildung 5-10 wird der Signalverlauf zweier beispielhafter Agenten für den Normalfall, die vorgezogene Schaltbedingung sowie die Neuberechnung während der Bewegung dargestellt.

Dieser Ansatz folgt der Spezifikation des Schwarmverhaltens auf dem mikroskopischen Level nach (Brambilla et al. 2013, S. 3). Das Interaktionsmuster wurde mithilfe simulativer und realer Versuchsumsetzungen in seinem makroskopischen Verhalten untersucht und im Detail angepasst. Eine erfolgreiche Bewegungssynchronisierung aller beteiligten Achsen unter den Betriebsbedingungen ist durch das spezifizierte Interaktionsmuster und Agentenverhalten sichergestellt und wird in Kapitel 1 als Simulationsmodell und am realen Anwendungsfall vorgestellt sowie hinsichtlich von Potential und Einschränkungen evaluiert.

Zielerreichung:
Synchronisierungsmechanismus ist konzipiert (Zwischenziel 2)

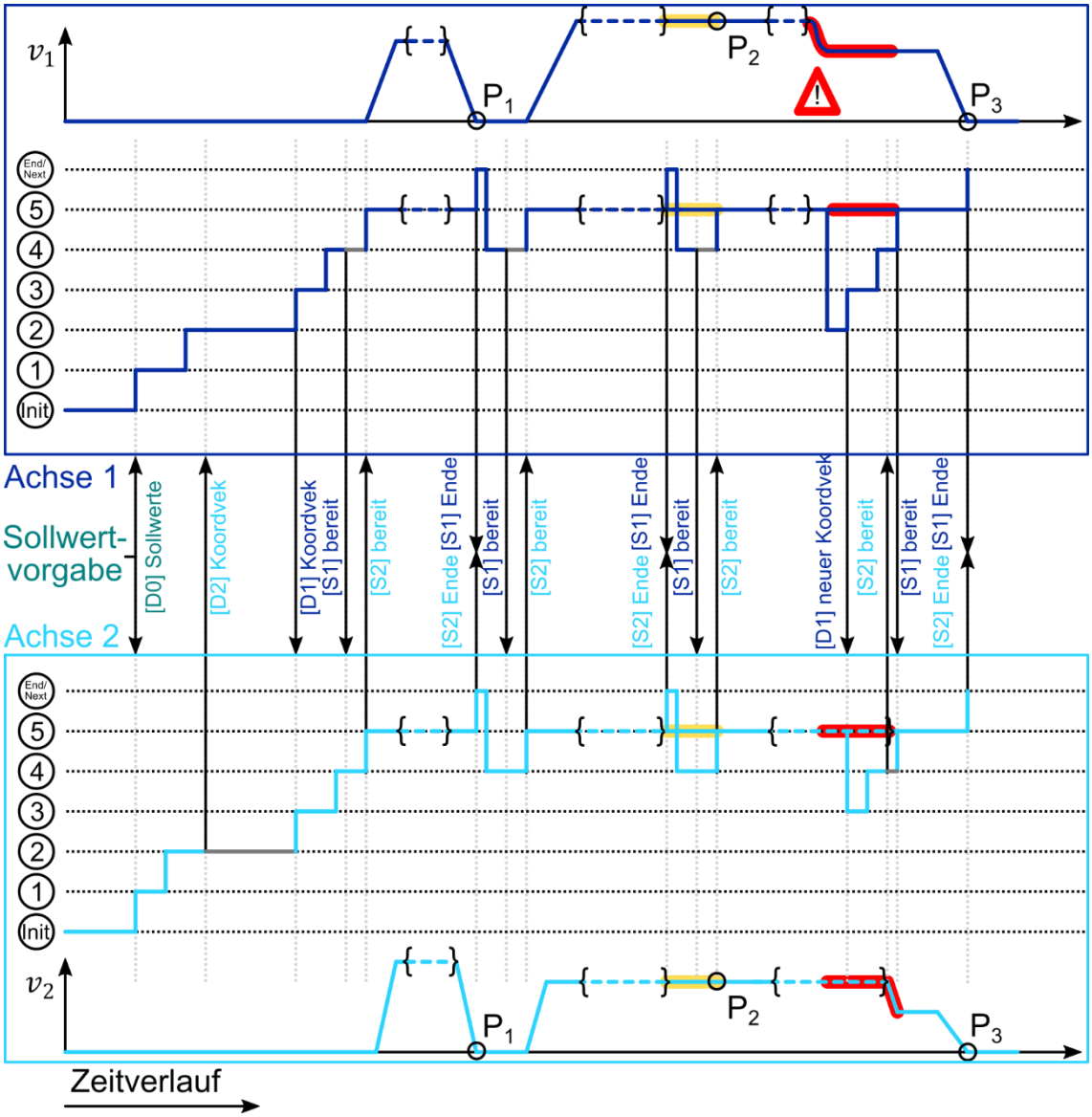

Abbildung 5-10: Signalflussverlauf der Interaktion zweier Agenten. Der Ablauf bis zum ersten Sollwert P_1 beschreibt den regulären Hochlauf und die Durchführung der Interaktion. Nach P_1 wird der nächste Sollwert angefahren, allerdings wirkt nun der vorgezogene Schaltbedingung für das Überschleifen zwischen Punkt P_2 und dem darauffolgenden Segment. Die Agenten tauschen noch während der laufenden Bewegung des Segmentes, markiert in gelb, ihre Bereitschaft für das nachfolgende Segment aus. Im Verlauf des folgenden Segments zum Punkt P_3 ist der Ablauf bei Neuplanung des Koordinationsvektors und damit des Konsens dargestellt. In rot markiert ist der Zeitbereich, bei dem die Bewegung noch nach Plan in den Agenten verläuft und die aktualisierte Konsenszeit noch nicht im Segmentregler wirkend ist. Alle Zeitverhältnisse sind hier nur beispielhaft. In der realen Anwendung sind die benötigte Zeit zum Austausch eines Bereitschaftsstatus oder neu berechnetem Konsens im Vergleich zur Länge der ablaufenden Bewegung um mehrere Größenordnungen kleiner.

6 Realisierung der verteilten Interpolation

Das soeben vorgestellte Interaktionsmuster als Synchronisierungsmechanismus wurde sowohl simulativ als auch auf realen Achsen umgesetzt und in seiner Funktionsweise validiert. Der prinzipielle Funktionsnachweis von Interaktionsmuster, Agentenverhalten und Kommunikationsprotokoll wurde simulativ erbracht. Neben dem korrekten Verhalten der Agenten im Gutfall, wenn die Achsdynamiken die geplanten Trajektorien wie erwartet umsetzen können, wurden auch die in Tabelle 4-3 genannten Betriebsbedingungen betrachtet und überprüft, ob die Entscheidungen im Handlungsraum einzelner Achsen entsprechend der Entscheidungsregeln nach Tabelle 4-6 umgesetzt werden.

Da es durch Simulationsansätze nur selten möglich ist alle Aspekte der Realität abzubilden, wird speziell in der Schwarmrobotik die Übertragung auf reale Roboter als sehr wichtig erachtet. Schwarmverhalten in simulativer und realer Umgebung weichen häufig voneinander ab, vgl. (Brambilla et al. 2013, S. 14). Gründe sind unter anderem Vereinfachungen durch die Simulation, sowie verrauschte Sensorwerte, Kommunikationsverzögerungen oder abweichende Agentendynamik im realen Anwendungsfall. Daher wird in diesem Kapitel nach dem Funktionsnachweis per Simulation und der Schnittstellendefinition der Agenten die Umsetzung der verteilten Interpolation an realen Achsen vorgestellt. Hierfür wurden mithilfe von Softwaretreibern kommerzielle MC-Komponenten in einem Achsverbund mit verteilter Interpolation angebunden. Dabei wurde auf Kommunikationsprotokolle zurückgegriffen, die von den kommerziellen Komponenten angeboten wurde. Für Rekonfigurationen und Erweiterung der Anwendung auf weitere Achsen werden notwendige Anpassungen im Agentenverhalten vorgestellt. Es folgt ein Ausblick auf die Implementierung der verteilten Interpolation auf einer prototypischen Miniatursteuerung. Abschließend erfolgt die zusammenfassende Bewertung von Potential und Funktionsumfang der verteilten Interpolation.

Zielsetzung:

Validierung des Synchronisierungsmechanismus und Definition der Schnittstelle zur Erweiterbarkeit und Rekonfiguration (Zwischenziel 3),

Bewegungssynchronisierung für dezentral gesteuerte Mehrachssysteme: Positionierende Komponenten mit Miniatursteuerung können im DEVEKOS-Systemverständnis ohne eine zentrale Steuerung synchronisierte Bewegungen umsetzen (Gesamtziel).

Vorgehen:

Validierung des Interaktionskonzepts, der Agentenlogik und der Kommunikationsstruktur,

Betrachtung eines Anwendungsfalls zur Rekonfiguration,

Erweiterung und Integration von kommerziellen MC-Komponenten

Bewertung von Potential und Funktionsumfang der Bewegungssynchronisierung

6.1 Validierung von Interaktionsmuster, Agentenverhalten und Kommunikationsprotokoll

Das Interaktionsmuster, Agentenverhalten und Kommunikationsprotokoll wurden über ein Simulationsmodell validiert. Hierfür wurde die Umgebung MATLAB Simulink verwendet. Die umfangreiche Funktionsblockbibliothek bietet viele Bausteine zum Aufbau des Agentenverhaltens und für die Simulation der Achsdynamik sowie ihrer Regelkreise. Simulink bietet sowohl eine zyklustaktbasierte Simulation für die kontinuierliche Dynamik der simulierten Achse, als auch Stateflow-Diagramme, in denen Zustandsautomaten abgebildet werden können.

6.1.1 Simulative Validierung

Das Agentenverhalten nach dem Interaktionsprotokoll wurde in Stateflow-Modellen für jede Achse umgesetzt. Die Agentengrundstruktur ist für alle Achsen identisch und kann daher einfach auf beliebig viele Agenten erweitert werden. Für den Fall einer dreidimensionalen Bewegung, wie in Abbildung 6-1 dargestellt, wurden in der ersten Instanz drei Achsen für die kartesischen Koordinatenrichtungen instanziiert, s. Abbildung 6-2. Der Synchronisierungsmechanismus wurde in Simulink vorerst ohne Kommunikationsnetzwerk zwischen den Agenten aufgebaut, um eine Validierung unabhängig von Effekten aus der Kommunikation zu ermöglichen. Stattdessen wurden in einer Kommunikationskoordination alle Nachrichtendaten in einem lokalen Speicher erfasst, der das Kommunikationsnetz in vereinfachter Form darstellte. Inwiefern eine solche vereinfachte Repräsentation des Kommunikationsnetzwerks Einfluss auf das Systemverhalten der Agenten hat, wird in Kapitel 6.2.3 durch die Umsetzung mit kommerziellen Komponenten überprüft. Die Sollwertvorgabe wird dem Stateflow-Modell zu Beginn der Interaktion vorgegeben. Da das Interaktionsmuster aktuell die lineare Interpolation umsetzt, wurden die Segmentvorgaben als Geradenabschnitte definiert.

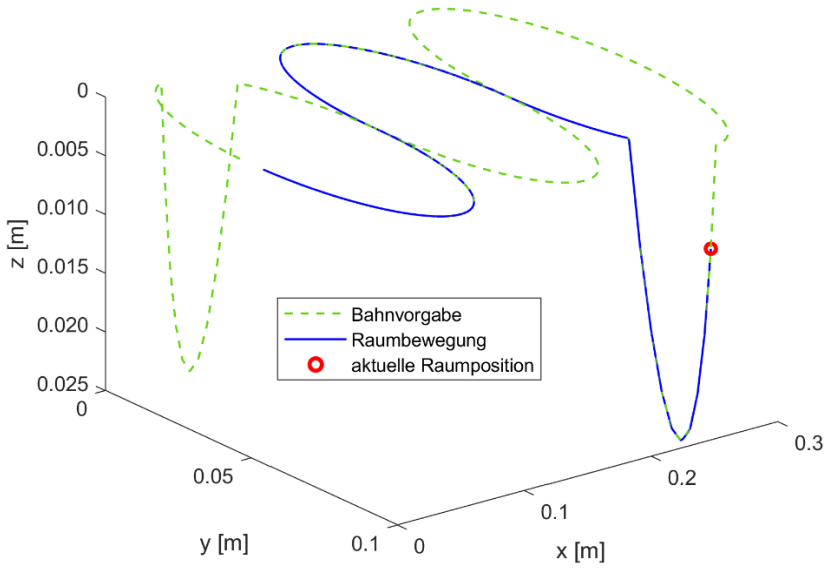

Abbildung 6-1: Sollwertvorgabe im Raum.

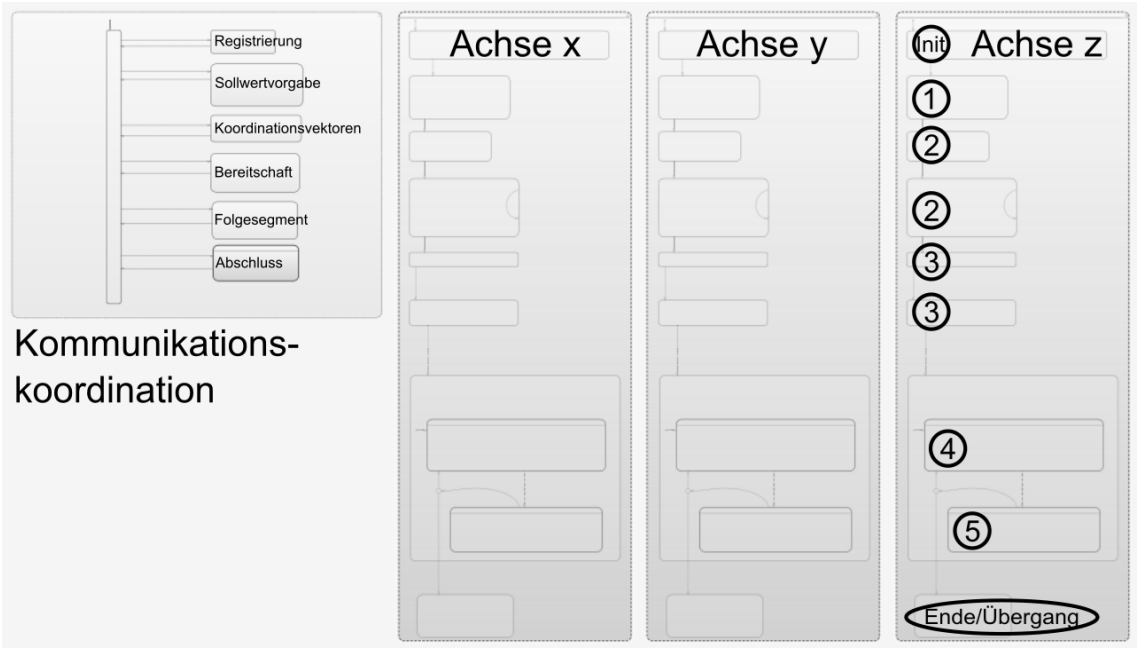

Abbildung 6-2: Kommunikationskoordination und Agentenstruktur in Simulink Stateflow zur Darstellung der Produktionsnetzwerks bzw. des Zustandsautomaten mit allen Schritten des Interaktionsmusters der Agenten nach Abbildung 5-6.

Die Planung der eigenen Koordinationsvariablen wurde innerhalb des Stateflow-Modells entsprechend der vorgegebenen Dynamik der Achse errechnet. In der Ausführung der Bewegung, die im Stateflow-Modell angestoßen wird, wurde eine Kaskadenreglerstruktur und ein Achsmodell zweiter Ordnung verwendet. Hier wurde auch der Segmentregler als Geschwindigkeitsvorsteuerung eingesetzt, s. Abbildung 6-5. Ergebnisse des Segmentreglers wurden wiederrum dem Stateflow-Modell zurückgespielt, das entsprechend des Interaktionsmusters bei Abweichungen über der Toleranzschwelle einen neuen Koordinationsvektor berechnet und kommuniziert.

In den resultierenden Bewegungen der Achsen, siehe Abbildung 6-3 und Abbildung 6-4, wird sichtbar, dass die Zielzeiten der Segmente in unterschiedlich engen Abständen entlang des Verlaufs des Diagramms liegen. Dies spiegelt die resultierenden Segmentzeiten wider, die über die Konsensvariablen koordiniert wurden, siehe Abbildung 6-6. Speziell die z-Achse ist hier hervorzuheben, die aufgrund einer geringeren Dynamik die Bewegung des Gesamtverbundes bremst und die Ausführungsgeschwindigkeit speziell der x- und y-Achse enorm reduziert, siehe Abbildung 6-7. Bei Betrachtung der Geschwindigkeiten der Achsen über den Bahnverlauf wird sichtbar, dass aufgrund der Sollwertvorgabe in Kreisform, die Auslastung der Achsen wechselnd sinkt und steigt, sodass auch die Rolle der Achse mit der geringsten Dynamik je nach Position entlang des Kreisbogens wechselt. Da sichtbar die y-Achse in weiten Teilen des Bahnverlaufs die Geschwindigkeit des Verbunds begrenzt und die Segmentzeiten vorgibt, könnte eine Optimierung des Systems beispielweise einen Ersatz oder eine erhöhte Dynamik dieser Achse vorsehen. Die z-Achse ist zwar mit einer noch geringeren Dynamik dargestellt, besitzt allerdings über den Zeitverlauf nur geringe Anteile an der Bewegung. Der Bahnfolgefehler des Gesamtverbundes an den Ziel-

punkten der Segmente liegt in der Simulation unterhalb von 1mm Abweichung von den Sollwerten.

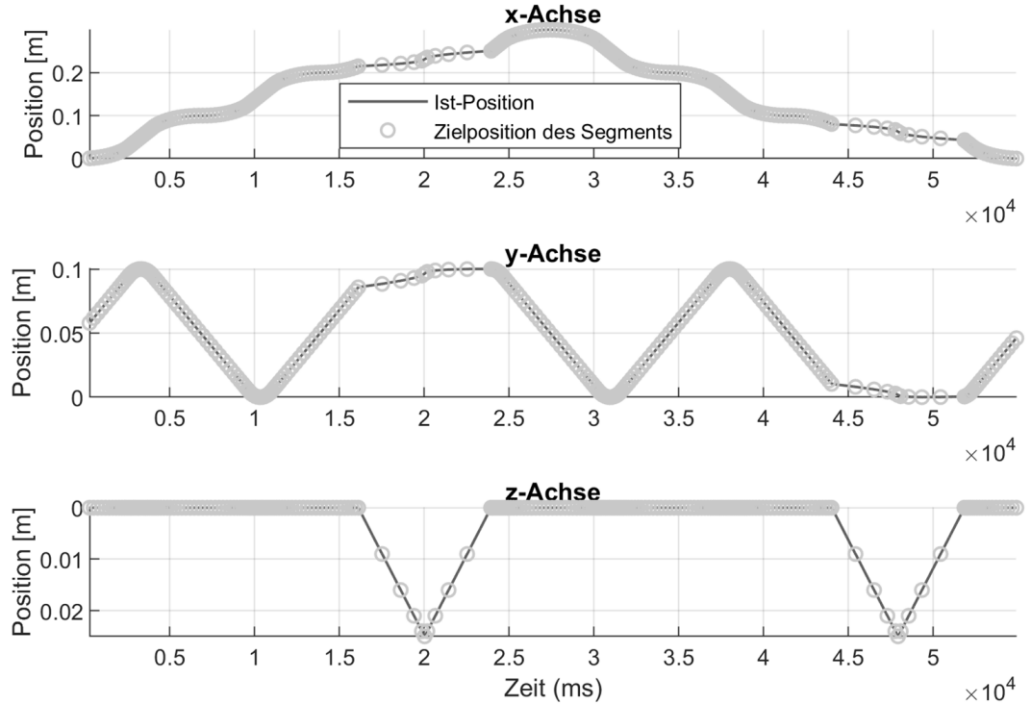

Abbildung 6-3: Positionsverlauf der Achsbewegungen zur Bewältigung der Sollvorgabe mit Zielpositionen der Segmenten im zeitlichen Verlauf.

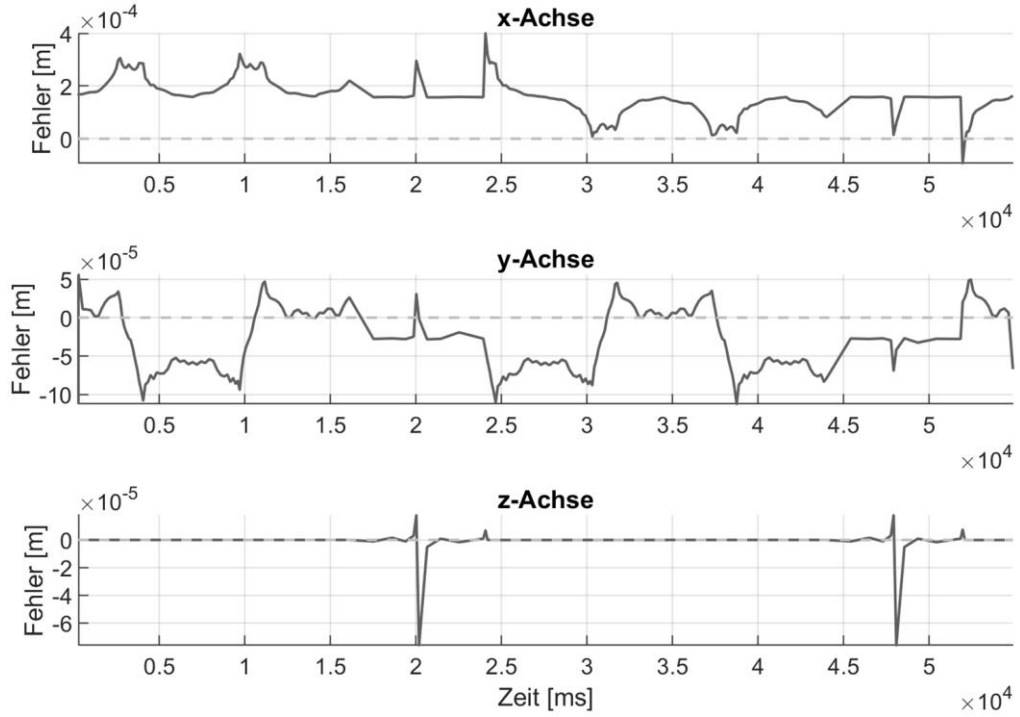

Abbildung 6-4: Folgefehler der einzelnen Achsen im Verbund.

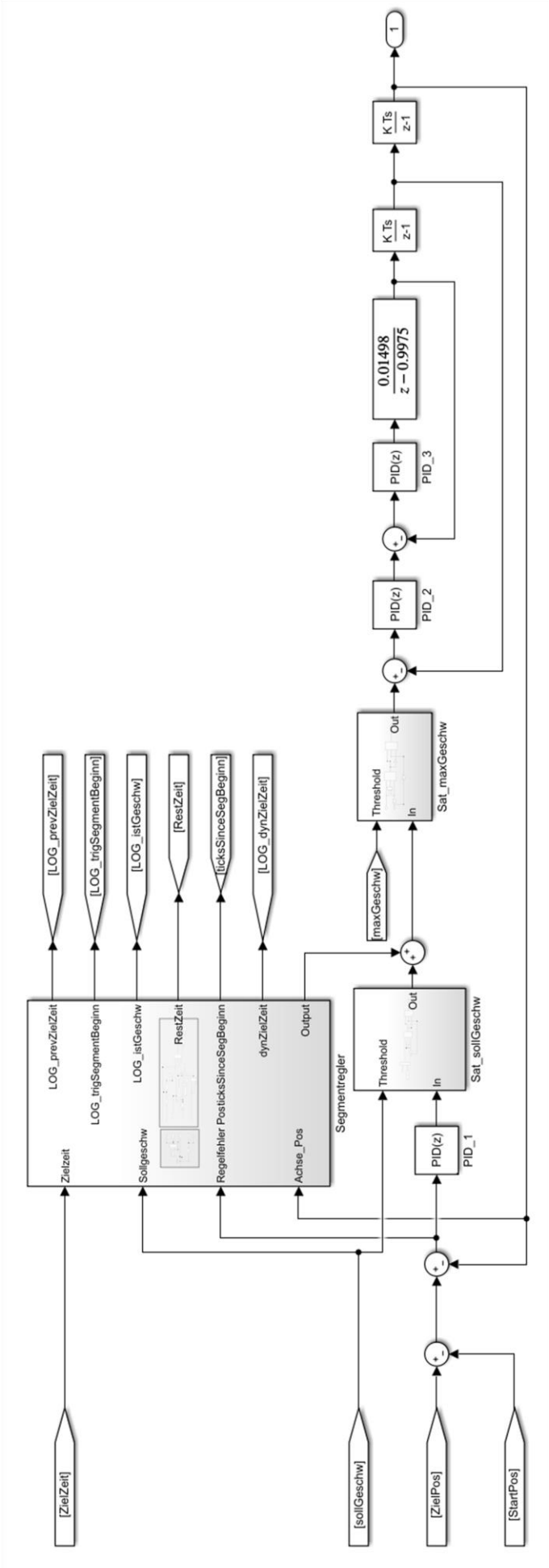

Abbildung 6-5: Simulation der Achsdynamik, der Reglerkaskade und des Segmentreglers.

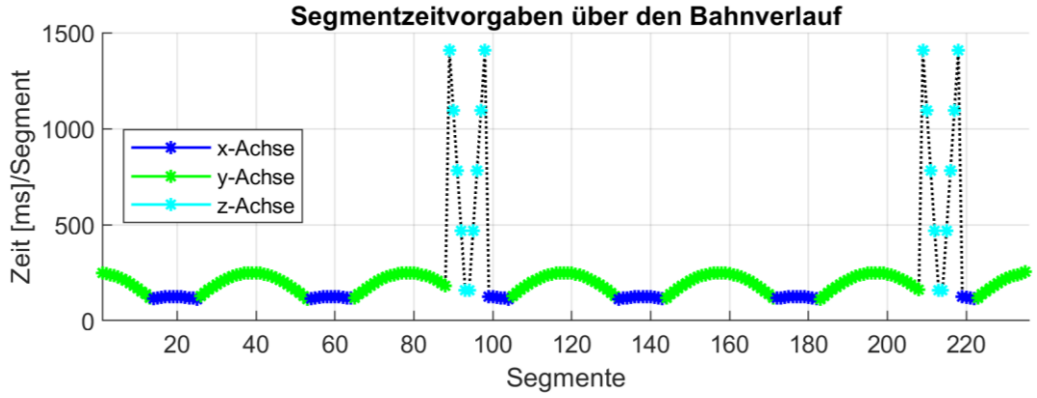

Abbildung 6-6: Resultierende Segmentzeitvorgaben über den Bahnverlauf bei gleichbleibenden Geschwindigkeitsbegrenzungen. Farblich markiert ist, welche Achse, aufgrund der maximalen Segmentzeit innerhalb des Verbunds, die Konsenszeit vorgibt.

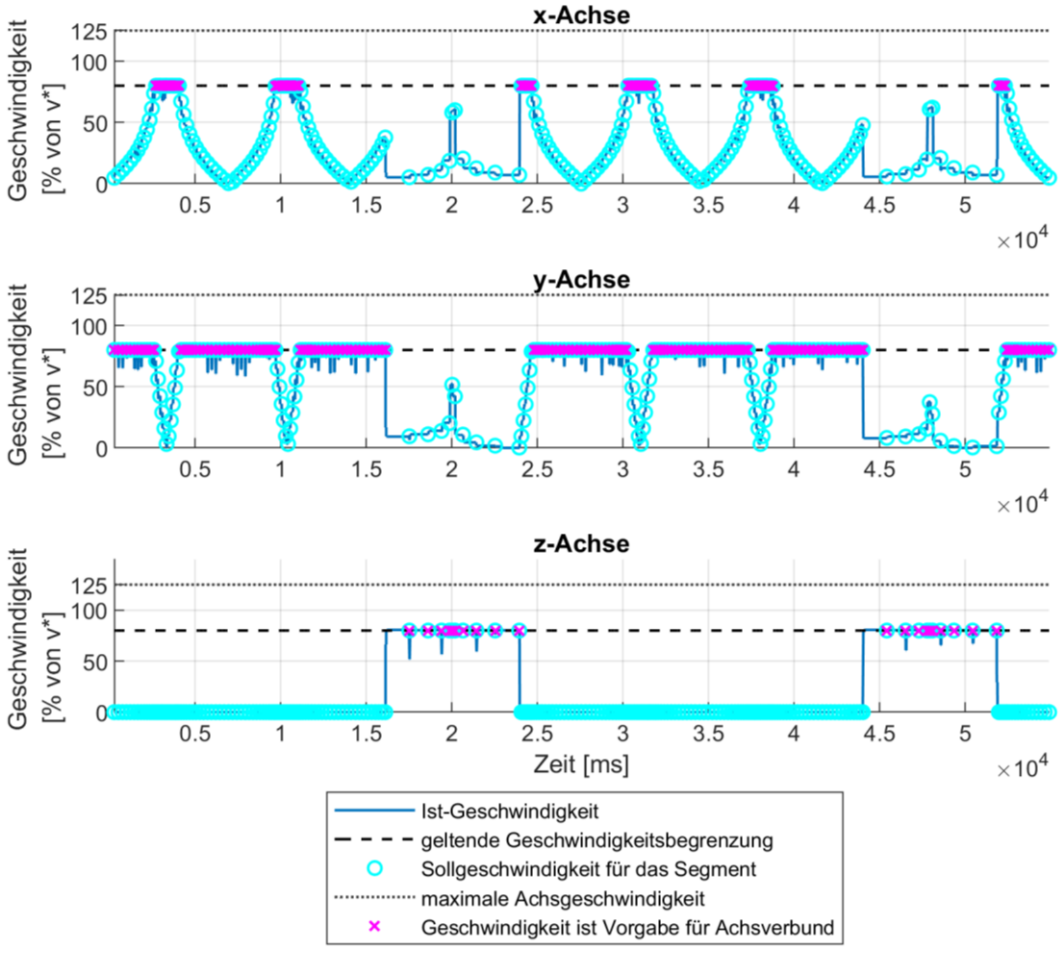

Abbildung 6-7: Geschwindigkeiten der Achsen über den Bahnverlauf. Markiert sind Achsen, die aufgrund der Maximierung der verfügbaren Dynamik die Segmentzielzeit des Verbunds vorgeben.

6.1.2 Funktionsnachweis durch variable Achsdynamiken

Die oben gezeigten Bahnen entstanden, ohne dass im Verlauf der Bewegungsausführung einzelne Achsen aufgrund einer reduzierten Dynamik neue Koordinationsvektoren versendet haben. Es wurden also die Segmente identisch zur Vorplanung, welche zu Beginn der Interaktion erfolgt ist, ausgeführt. Durch den vorgestellten Segmentregler sind die Achsen zusätzlich in der Lage, auch während der Bewegung auf eine veränderte Dynamik (z.B. durch veränderliche Last) nach den Betriebsbedingungen aus Tabelle 4-3 zielgerichtet zu reagieren. Der Segmentregler überprüft den Zeitbedarf für die Umsetzung des Segmentes und korrigiert diesen im Koordinationsvektor bei Bedarf durch z. B. geringere Dynamik aufgrund externer Lasten. Der Nachweis, dass die Achsen untereinander keine feste Hierarchie besitzen, wird dadurch erbracht, dass alle Achsen benutzerdefinierte, variable Achsdynamiken erhalten. Der Benutzer kann so während der Ausführung des Systems die Achsdynamiken reduzieren oder erhöhen und beobachten, wie die Ausführung der Bahn jeweils bei der Achse mit der geringsten Dynamik in Verhältnis zu der zu verfahrenden Weglänge orientiert wird. Diese Achse, die die geringste Dynamik für die geplante Bewegung vorgibt, ist in diesem Moment der virtuelle Leader. Je nach Orientierung des Bahnsegments im Raum und Veränderung der Achsdynamiken wird diese Funktion zwischen allen beteiligten Achsen zur Laufzeit übergeben. Dies gilt auch für den Fehlerfall, wenn eine oder mehrere Achsen sich aufgrund einer Blockade oder Kollision nicht mehr bewegen können. In diesem Fall kommunizieren diese einen sehr hohen bis unendlichen Zeitbedarf für das aktuelle Segment und stoppen dadurch die Bewegung der weiteren Achsen.

Abbildung 6-8: Achsdynamikvorgabe für die beteiligten Achsen.

Für die Darstellung der variablen Achsdynamiken wurde die identische Sollwertvorgabe wie in Abbildung 6-1 verwendet. Entlang des Verlaufs der Bahn wurden allerdings die maximal zulässigen Geschwindigkeiten der Achsen variiert. Hierbei wurde ein Prozentsatz der Achsgeschwindigkeit vorgegeben. Abbildung 6-8 stellt die Geschwindigkeitsvorgabe der drei Achsen dar. Die blau markierten Anteile ihrer jeweiligen Achsgeschwindigkeit entsprechen der Dynamikbegrenzung (0,5 für Achse x, 0,8 für Achse y, 0,2 für Achse z), der rote Zeiger zeigt die tatsächliche Ausnutzung dieses Potentials. Diese Vorgabe wurde über den Zeitverlauf verändert, sodass die Achsen auch während der Bewegungsausführung eine erneute Planung der Segmentzeiten durchführen und diese aktualisierten Segmentzeiten umsetzen mussten. Es wird sichtbar, dass sich die Ausführung der Bahn in der Zeit verändert, s. Abbildung 6-9, die Folgefehler entlang der Trajektorie, s. Abbildung 6-10, allerdings in der gleichen Größenordnung bleiben, sodass die Performanz der Gesamtsystems durch diese variierenden Achsdynamiken nicht eingeschränkt wird.

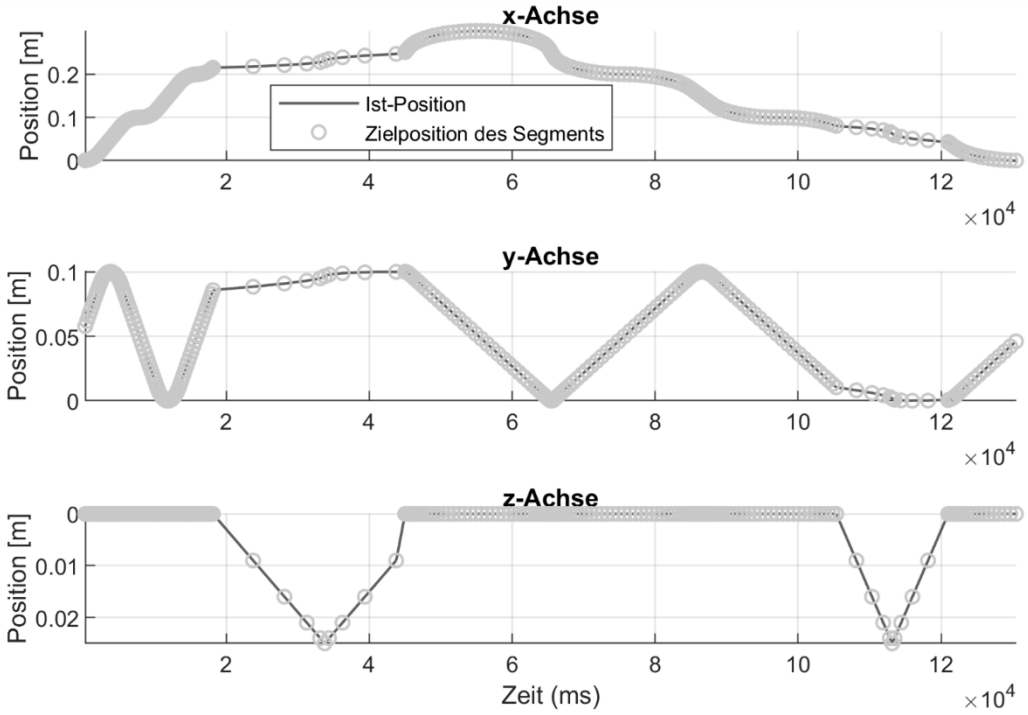

Abbildung 6-9: Positionsverlauf bei variierenden Achsdynamiken.

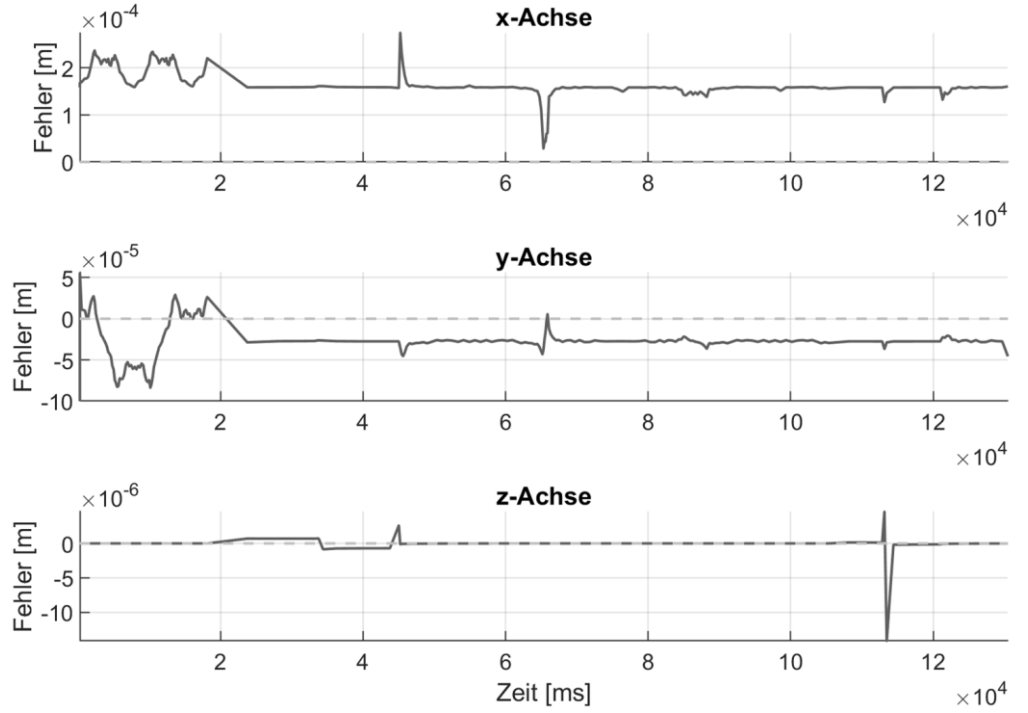

Abbildung 6-10: Folgefehler bei variierenden Achsdynamiken.

Bei der Betrachtung der resultierenden Segmentzeiten in Abbildung 6-12 kann das sich umkehrende Verhältnis der verfügbaren Dynamik zwischen der x- und y-Achse durch die Anteile der Vorgaben der Segmentzeit nachvollzogen werden. Die verfügbaren prozentualen Anteile der Geschwindigkeit sind in einer negierten Skala rechtsseitig aufgetragen und zeigen die Kausalität zwischen der veränder-

ten Geschwindigkeitsvorgabe und der Höhe sowie Zuordnung der resultierenden Segmentzeit. Sichtbar werden die variierten Geschwindigkeitsvorgaben und die achsbezogenen Auswirkungen auf die Bewegung ebenfalls in Abbildung 6-11.

Die variablen Achsdynamiken wurden neben der simulativen Validierung, als Nachweis der Funktionalität des Interaktionsmusters ebenfalls mithilfe einer Benutzerschnittstelle im realen Versuchsaufbau für eine Messedemonstration der verteilten Interpolation umgesetzt. Die Ergebnisse werden im nachfolgenden Kapitel 6.2.3 im Detail vorgestellt.

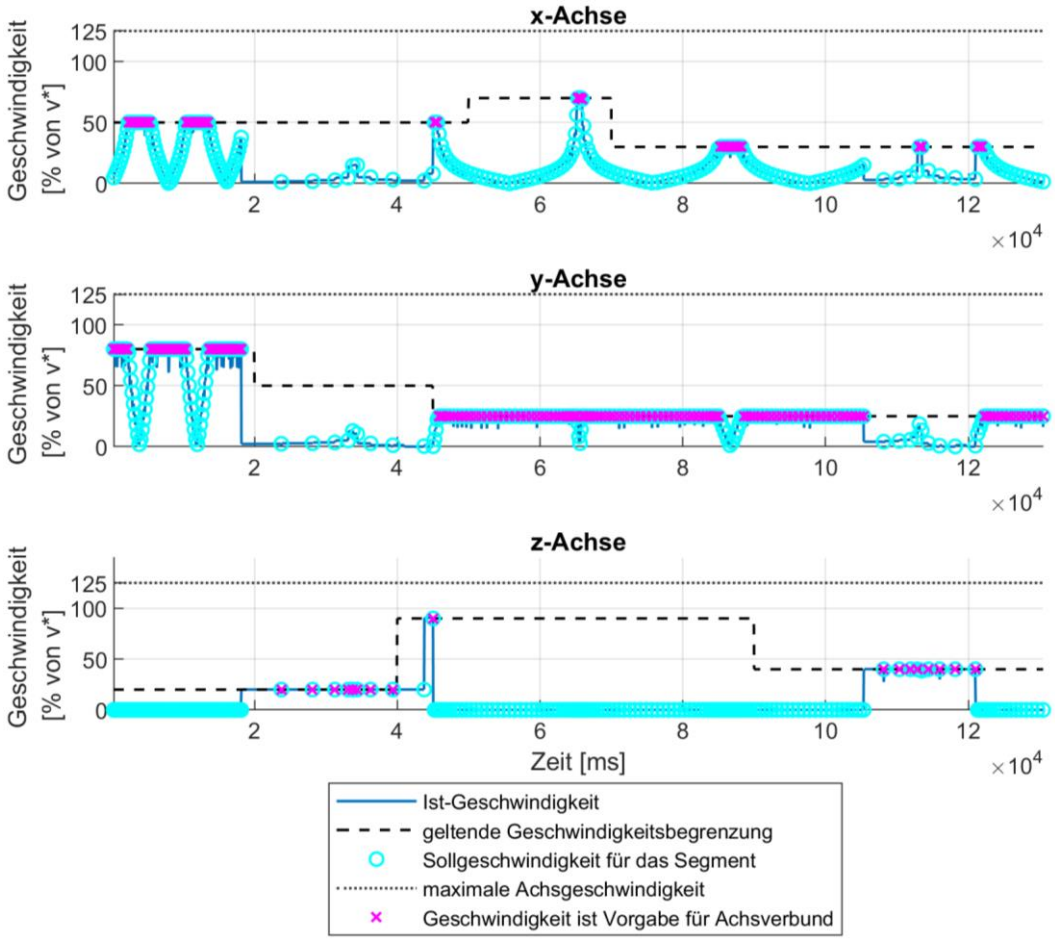

Abbildung 6-11: Betrachtung der Ausnutzung der verfügbaren, variierenden Geschwindigkeit der Achsen über den Bahnverlauf. Markiert sind jeweils die Achsen, die aufgrund der Maximierung der verfügbaren Dynamik die Segmentzielzeit des Verbunds vorgeben.

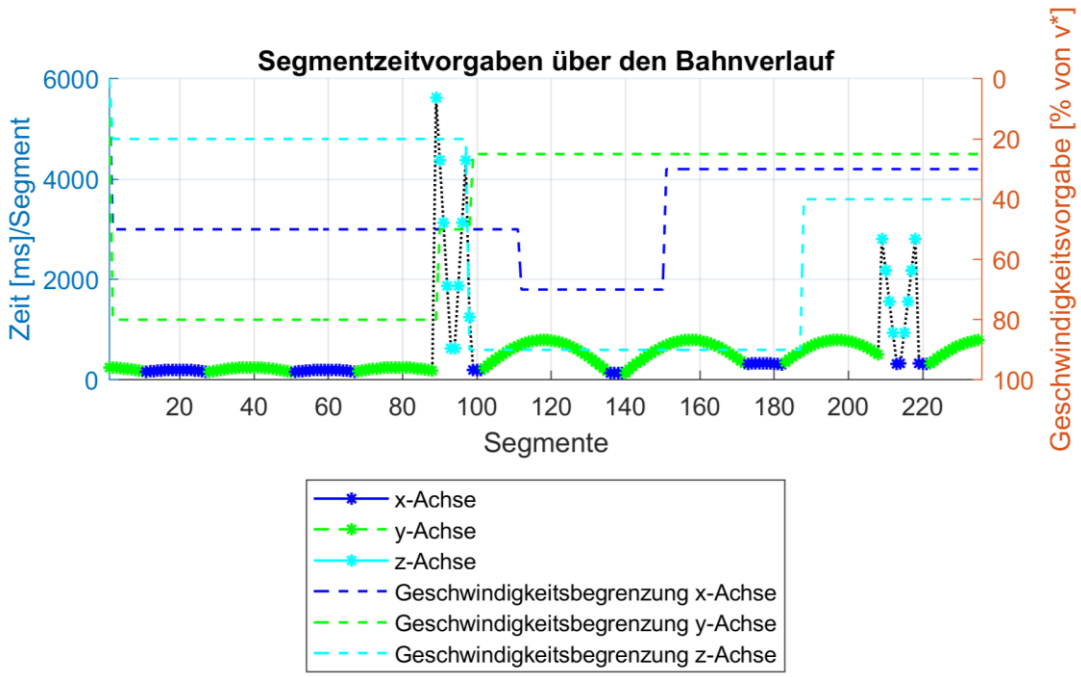

Abbildung 6-12: Resultierende Segmentzeitvorgaben über den Bahnverlauf bei variierenden Geschwindigkeitsbegrenzungen.

6.2 Anwendungsfallbetrachtung

Themen wie die Rekonfiguration, die Erweiterung und die Integration von kommerziellen MC-Komponenten werden nachfolgend anhand von konkreten Anwendungsfällen dargestellt. Hierfür werden zuerst die Anforderungen aus der Rekonfiguration und daraus entstehende Anpassungen im Interaktionsmuster oder Agentenverhalten vorgestellt. Eine Definition der Schnittstelle zur verteilten Interpolation in einem Achsverbund bereitet die Grundlagen für die Integration von kompatiblen kommerziellen Achssystemen in einen verteilt interpolierenden Achsverbund vor. Zum Abschluss wird ein Ausblick auf die Integration der verteilten Interpolation in die prototypische Miniatursteuerung gegeben.

6.2.1 Erweiterbarkeit und Rekonfiguration der verteilten Interpolation

Eine Anforderung der Betriebsbedingungen nach Tabelle 4-3 des betrachteten Anwendungsfalls ist die Rekonfiguration der Anzahl oder Art der beteiligten Achsen. Darunter fallen folgende Fälle:

a) Eine zusätzliche Achse wird hinzugefügt: $n \longmapsto n+1$
b) Eine Achse wird vom System ersatzlos entfernt: $n \longmapsto n-1$
c) Eine Achse wird in ihrer Position, Orientierung oder der Arbeitsraumdefinition verändert: ${}^{h}\vec{n} \longmapsto H \; {}^{h}\vec{n}$
d) Eine Achse wird in ihrer Dynamik oder Reglereinstellung verändert: $v_{n,\max} \longmapsto f \cdot v_{n,\max}$
e) Eine Achse wird durch eine gleichartige Achse von einem anderen Hersteller oder mit einer anderen Aktorik ersetzt

Das Interaktionsmuster, wie auch das Kommunikationsprotokoll, ist aufgrund der Anforderung so spezifiziert worden, dass keine bestimmte Anzahl an beteiligten Achsen festgelegt ist. Durch die Beschreibung der Bahnvorgabe in Weltkoordinaten und die Verwendung der Segmentzeitspanne als Koordinationsvariable, sind beliebig viele Teilnehmer in der Lage die Entsprechung der Bahnvorgabe in lokale Koordinatensysteme zu übertragen und eine Koordinationsvariable zu berechnen, um an einer Bewegung teilzunehmen. Es wird in allen Fällen angenommen, dass der Systemintegrator bei einer Rekonfiguration sicherstellt, dass Achsen derart vorhanden, montiert und parametriert sind, dass eine Bewegung nach der Bahnvorgabe umsetzbar ist. Eine Überprüfung ob alle Segmente der Bahnvorgabe in allen Bewegungsdimensionen durch die beteiligten Achsen umgesetzt werden können, findet nicht statt.

Voraussetzung ist im spezifizierten Interaktionsmuster lediglich, dass zu Beginn einer Bewegungsvorgabe sich alle Achsen, die an der Bewegung teilnehmen, gegenseitig registrieren, damit ab diesem Zeitpunkt die Anzahl der beteiligten Achsen und damit der zu berücksichtigenden Koordinationsvektoren zur Festlegung eines Konsensvektors fest steht, siehe den Init-Schritt des Interaktionsmusters in Kapitel 5.2.3 und die Erläuterung zu den Vektor- und Matrixdefinitionen in Abbildung 5-4. Eine Rekonfiguration während der laufenden Bewegung wird ausgeschlossen. Für alle Kommunikationsschritte aus Tabelle 5-1 wird gewartet, bis alle registrierten Achsen ihren Status bzw. ihre Daten versendet haben. Anpassungen sind bei Rekonfigurationen wie folgt vorzunehmen, s. auch Abbildung 6-13.

a) Sollte eine Achse $n+1$ dem System hinzugefügt werden, registriert sie sich bei allen weiteren beteiligten Achsen, die ab diesem Zeitpunkt auf $n+1$ statt vorher n Statusnachrichten und/oder Datennachrichten pro Schritt warten. Der Konsensvektor wird dann als ein Maximum aus $n+1$ statt n Koordinationsvektoren gebildet. Dieser Vorgang ist bei der Eingliederung beliebig vieler weiterer Achsen in gleicher Art und Weise durchzuführen.
b) Dasselbe gilt umgekehrt für eine Anzahl von $n-1$ statt n beteiligten Achsen.
c) Ändert sich die Position oder Orientierung einer Achse, weil sie neu montiert wird, dann muss die Transformation der Weltkoordinaten auf Achskoordinaten in Schritt 1 des Interaktionsmusters entsprechend der homogenen Transformation H angepasst werden.
d) Wird eine Achse in ihrer Einstellung verändert, z. B. die maximale Dynamik eingeschränkt oder erweitert, muss entsprechend das lokale Agentenwissen angepasst werden, sodass in Schritt 2 des Interaktionsmusters ein korrekter Koordinationsvektor berechnet wird.
e) Wird eine Achse ersetzt, kann dies nach dem DEVEKOS Systemgedanken auch eine gleichartige Achse eines anderen Herstellers, eine andere Aktorik u. ä. sein. In diesem Fall muss sichergestellt sein, dass die Achse entsprechend des Interaktionsmusters und des Kommunikationsprotokolls in das System eingefügt wird. Dafür ist notwendig, dass dieselben Schnittstellen und Grundfunktionen in der Achse umgesetzt werden. Diese werden im folgenden Kapitel zusammengestellt.

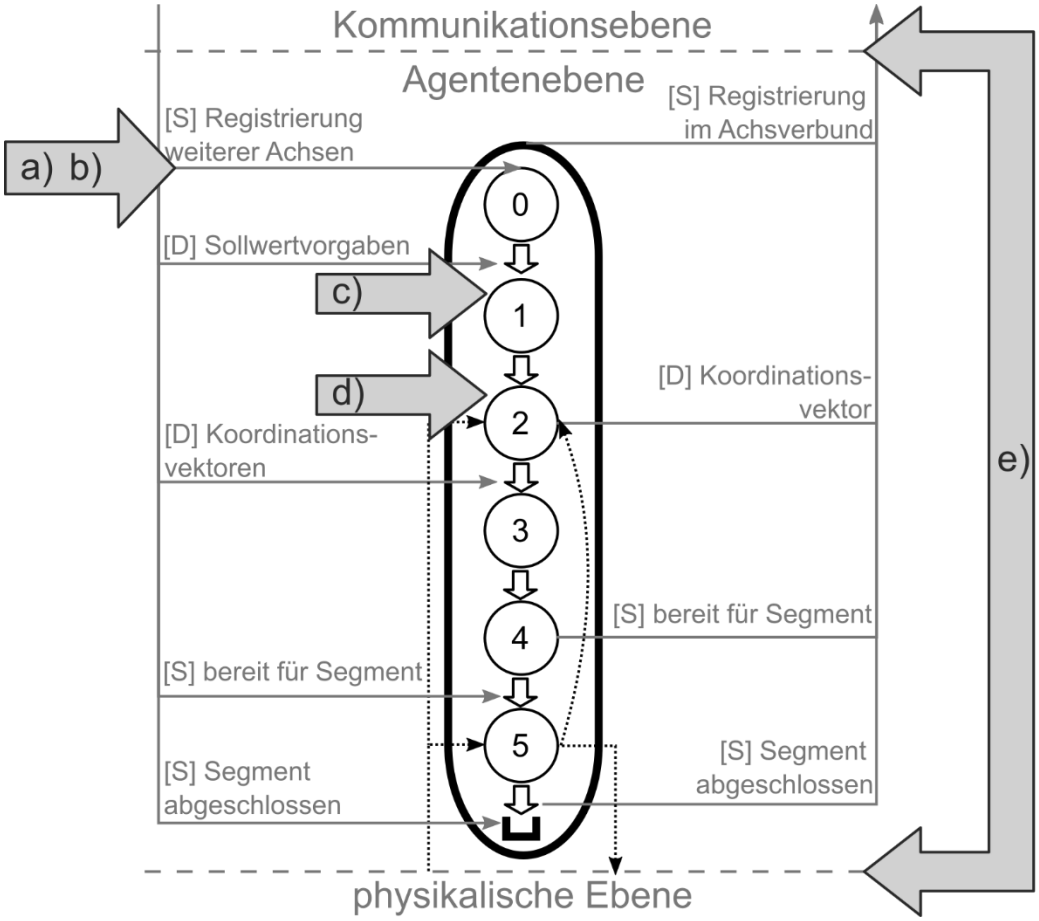

Abbildung 6-13: Rekonfigurationsfälle und Anpassung im Agentenverhalten anhand der vereinfachten Agentenstruktur aus Abbildung 5-6.

6.2.2 Schnittstellen und Funktionen der verteilten Interpolation

Da im DEVEKOS-System die Interaktion zwischen intelligenten Komponenten unabhängig von herstellerspezifischen Protokollen durchgeführt wird, können beliebige Komponenten in das System integriert werden, sofern sie in den Schnittstellen und Grundfunktionen kompatibel sind. Für die verteilte Interpolation wurde eine kompakte Schnittstellen- und Funktionsdefinition gewählt, um eine Erweiterung mit beliebigen weiteren Komponenten für Anwender einfach zu gestalten. Hierbei müssen die zu integrierenden Komponenten über Parameterwissen und Basisfunktionen in ihrem System nach Tabelle 6-1 verfügen. Während der Interaktion in der verteilten Interpolation muss zusätzlich Speicher für das lokale Agentenwissen nach Tabelle 6-2 vorgehalten werden.

Um in den Grundfunktionen und Agentenverhalten kompatibel zu sein, muss eine zu integrierende Komponente weiter die in Tabelle 6-3 erläuterten Rechenoperationen und Fähigkeiten umsetzen. Eine mögliche Umsetzung dieser Fähigkeiten, zum Beispiel des Segmentreglers wurde in Kapitel 5.2.3.3 gezeigt. Dem Integrator bleibt allerdings freigestellt, andere Implementierungen zu verfolgen, insofern das in der Interaktion gezeigte Verhalten identisch zur Spezifikation dieser Schnittstellen bleibt.

	Anforderung
Position und Orientierung	Achsposition und -orientierung in Weltkoordinaten, Arbeitsraum der Achse
Dynamik	maximale Geschwindigkeit
Sensorik	Positionserfassung
Kommunikation	Einbettung in das Produktionsnetzwerk
Zeit	synchronisierte Uhr

Tabelle 6-1: Anforderungen an einen Agenten hinsichtlich Funktionen und Parameter, die als bekannt vorausgesetzt werden.

	lokales Agentenwissen
Achsverbund	Anzahl und Adressen aller beteiligten Achsen
Sollwertvorgabe	Sollwertvorgaben
Konsens	Sollposition und Sollzeiten für alle Sollwertvorgaben
Zustand	aktueller Schritt im Interaktionsmuster, aktuelles Segment der Durchführung

Tabelle 6-2: Zusätzlicher Speicherbedarf durch lokales Agentenwissen

	Agentenverhalten
Kommunikation (Schritte 0,4,6)	Reaktion auf Nachrichten nach Kommunikationsprotokoll Senden von Nachrichten nach Kommunikationsprotokoll
Transformation (Schritt 1)	Transformation der Sollwerte von Weltkoordinaten auf Achskoordinaten
Koordination (Schritt 2)	Berechnung des Zeitbedarfs pro Segment nach $T_n^* = \frac{r_{n_i} - r_{n_{i-1}}}{v_n^*}$
Konsens (Schritt 3)	Berechnung des Konsensvektors nach $T_i^\circ = \max_{\forall n} \{T_{i,n}^*\}, i = 1 \dots m$
Segmentdurchführung (Schritt 5)	Durchführung einer gleichförmigen Bewegung zum aktuellen Konsens mit Segmentregler und Geschwindigkeitsvorsteuerung von $v_n^* = 0.8 \cdot v_{n,\max}$, Laufende Überprüfung der eigenen Bewegung, Kommunikation bei erhöhtem Zeitbedarf nach Kommunikationsprotokoll

Tabelle 6-3: Gefordertes Agentenverhalten

Die hiermit zusammengefasste Schnittstellendefinition und Grundfunktionen eines Agenten ermöglichen es, beliebige positionierende Komponenten für die verteilte Interpolation kompatibel zu entwickeln. Auch Achsen, die keine Miniatursteuerung besitzen, können mithilfe von Softwaretreibern nach der Schnittstellendefinition über externe Rechenhardware in Achsverbünde integriert werden, s. (Dripke et al. 2018) und das nachfolgende Kapitel.

Zielerreichung:

Der verteilte Synchronisierungsmechanismus ist validiert.

Die Schnittstellendefinition zur Erweiterbarkeit und Rekonfiguration ist abgeschlossen (Zwischenziel 3)

6.2.3 Integration des Ansatzes in kommerzielle Antriebskomponenten

Die in Kapitel 1.1 vorgestellte DEVEKOS Komponente ist bisher Fokus der Forschung. Es existieren bisher nur vereinzelt prototypische Umsetzungen von Komponenten mit integrierter Minitatursteuerung. Zur Umsetzung der Interaktionsmuster in einem realen Anwendungsfall musste daher auf kommerzielle Komponenten zurückgegriffen werden, die mithilfe von zusätzlichen Softwaretreibern nach der obigen Schnittstellendefinition erweitert wurden.

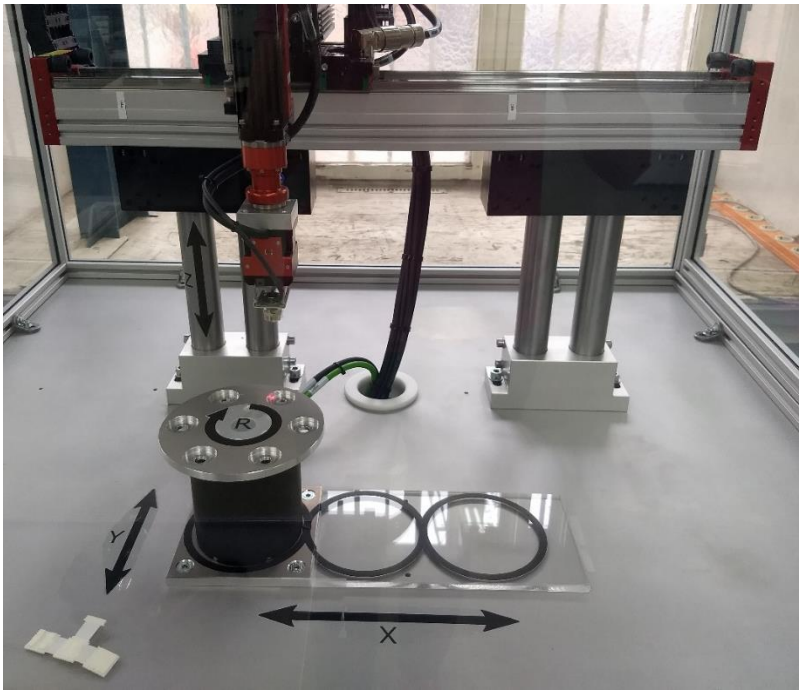

Abbildung 6-14: Versuchsaufbau des Portalachssystems mit Rundtisch.

Für die Realisierung der verteilten Interpolation stand ein Portalachssystem mit 3 linearen Achsen sowie einem Rundtisch zur Verfügung, s. Abbildung 6-14. Die LDA-Achsen werden mit digitalen Antriebsverstärkern angesteuert, die zusätzlich zu den verbreiteten Feldbusprotokollen eine weitere Anwenderschnittstelle mit Grundfunktionen zur Achsansteuerung anbieten. Die Kommunikation mit der

Schnittstelle findet über das von der Komponente angebotene Kommunikationsprotokoll UDP (User Datagram Protocol) statt.

Im Simulink Stateflow-Modell aus Kapitel 6.1.1 wird das simulierte Achsmodell mit der Achsansteuerung der realen Achsen über die Schnittstelle ersetzt. Vorgabewerte an die Achsen sowie der Status der Achse werden mit Funktionen der Anwenderschnittstelle über UDP zwischen Antriebsverstärker und Agenten ausgetauscht. Die Anwenderschnittstellenintegration ist innerhalb MATLAB umgesetzt und wird aus Simulink über den extrinsic coder ausgewertet. Im Simulinkmodell wurden Standardfunktionen von Antriebsverstärkern wie die Abfrage der aktuellen Position oder die Vorgabe von Sollwerten nach (Dripke et al. 2018) umgesetzt, die erst in der extrinsic coder-Funktion in MATLAB auf die konkrete Funktion der Anwenderschnittstellen übertragen werden. Dies lässt eine schnelle Adaption auch an andere Schnittstellen an nur begrenzt vielen Stellen im Programmcode zu. Da der Funktionsumfang des verwendeten Antriebsverstärkers keine Vorgabe einer dynamischen Geschwindigkeitsvorsteuerung anbietet, wird der Segmentregler nur zur Überwachung der benötigten Zeit aufgrund der verfügbaren Dynamik verwendet. Das bedeutet, dass detektierte Dynamikänderungen nicht während der Abarbeitung eines Segments, sondern erst im darauffolgenden Segment wirksam werden. Es ist damit besonders Schritt 5 im Agentenverhalten nach Tabelle 6-3 eingeschränkt, da der Segmentregler als ein nach der Zielzeit agierender Positionsregler keine Standardfunktion ist und nicht durch Standardfunktionen wie Sollwert- oder Geschwindigkeitsvorgaben komplett ersetzt werden kann.

Die Kommunikation über UDP hat den Nachteil, dass bei hoher Kommunikationslast Nachrichten ohne Rückmeldung verloren gehen können, weshalb die Vorgaben und Abfragen nur in einem relativ niedrigen Takt getriggert werden. Weiter lässt Simulink keine parallelen Prozesse zu, was bedeutet, dass das Simulationsmodell, zum Beispiel durch die Grafikdarstellung, die Rechenkapazität zu einem großen Teil auslastet. Es gibt zum aktuellen Zeitpunkt keine Möglichkeit, in Simulink den Prozessen Prioritäten zuzuordnen, damit die Kommunikation weiterhin ausgeführt wird und stattdessen die Grafikausgabe verzögert wird. Daher muss aktuell darauf geachtet werden, das System nicht an die Grenze der Rechenkapazität zu bringen, da ansonsten die Kommunikation zu den Achsen per UDP Ausfälle zeigt und dann starke Einschränkungen im Folgeverhalten zu erwarten sind.

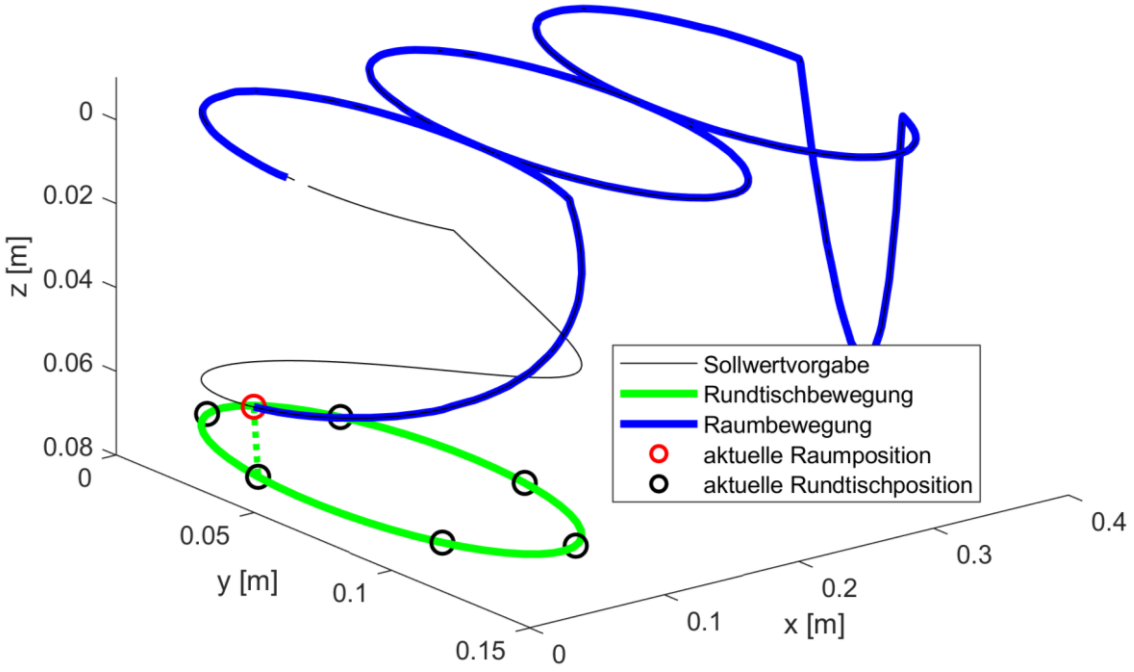

Abbildung 6-15: Sollwertvorgabe im Raum inklusive Rundtischbewegung.

Um die Interaktion in gemeinsamen Arbeitsräumen von Achsen verschiedener Bauart, unterschiedlicher Hersteller und auch anderer Kommunikationsschnittstellen zu zeigen, wurde dem Portalsystem eine Rundtischachse hinzugefügt und die Sollwertvorgabe angepasst, sodass eine der Senkbewegungen des Portals über einer Ablageposition des Rundtisches synchronisiert ist, siehe Abbildung 6-15. Diese rotative Achse ist mit einem Antriebsverstärker eines anderen Herstellers und anderen digitalen Schnittstellen ausgestattet, wurde aber in gleicher Art und Weise parallel zu den linearen Achsen im Stateflow-Modell eingefügt. Die Architektur ist in Abbildung 6-16 dargestellt.

Die Kommunikationsschnittstelle über MATLAB wurde für die Rundtischachse über eine Feldbusverbindung über EtherCAT realisiert. Eine Kopplung der Rundtischachse mit dem Portalsystem wäre über eine zentrale Bewegungssteuerung aufgrund der inkompatiblen Feldbusprotokolle nicht umsetzbar gewesen. Die Grundfunktionen im Simulinkmodell, die über den extrinsic coder mit der Schnittstelle des Antriebsverstärkers gekoppelt wurden, sind identisch zu den Achsen im Portalsystem angeschlossen, jedoch angepasst auf die verfügbare digitale Schnittstelle. Da die Bahnvorgabe in kartesischen Koordinaten definiert ist, musste durch den Agenten der Achse eine Transformation auf die Achskoordinaten, hier Winkelkoordinaten, umgesetzt werden. Zusätzlich wurde ein Arbeitsraum der Achse so definiert, dass eine Interaktion im Achsverbund nur dann stattfindet, wenn das Portalsystem im Arbeitsraum direkt über den Ablagepositionen des Rundtisches verfährt. Koordinationsvariablen wurden also nur dann berechnet, wenn der Arbeitsraum des Rundtisches von der vorgegebenen Bahnkurve überstrichen wurde. In allen anderen Fällen wurde die Koordinationsvariable $T_i^* = 0$ kommuniziert, was eine Entkopplung von den restlichen Achsbewegungen entspricht. Alle weiteren Achsen im Verbund erhielten lediglich in Init-Schritt

des Interaktionsmusters die Registrierung einer weiteren, vierten Achse im interpolierenden Verbund. Folglich entstand der Konsensvektor

$$T_i^{\circ} = \max_{\forall n} \{T_{i,n}^{*}\}, i = 1 \ldots m, n = \{1,2,3,4\} \tag{6-1}$$

als Maximum von vier statt drei Koordinationsvariablen. Es fand keine weitere Anpassung an der Konfiguration oder dem Agentenverhalten der Portalachsen statt.

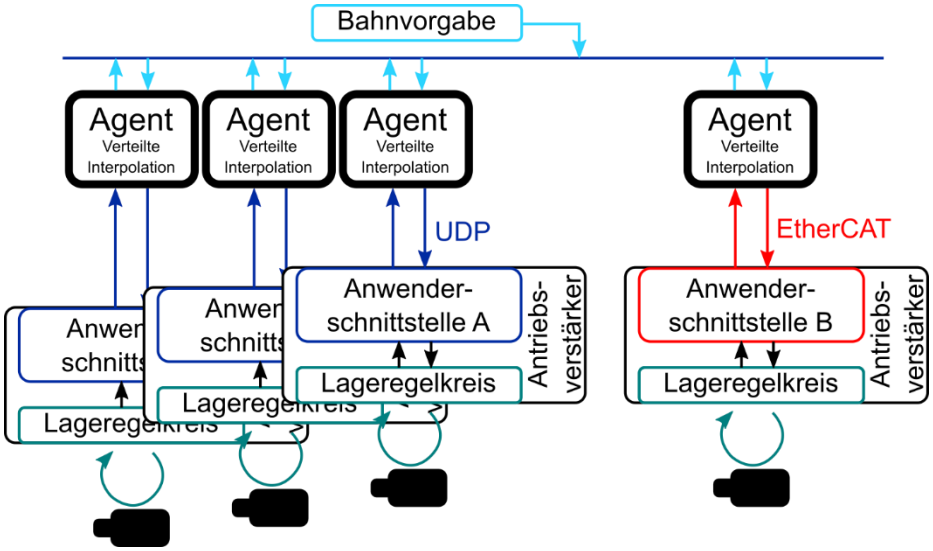

Abbildung 6-16: Integration von Achsen und Antriebsverstärkern verschiedener Hersteller in die verteilte Interpolation mit unterschiedlichen Anwenderschnittstellen und Kommunikationsprotokollen.

Vernachlässigt man die Kommunikationsausfälle aufgrund der Rechenauslastung, behält das System trotz fehlendem Segmentregler ein gutes Folgeverhalten, siehe Abbildung 6-17 und Abbildung 6-18, selbst wenn sich Achsdynamiken während der Bewegungen ändern. Die Kopplung von Rundtischachse und den Portalachsen wird lediglich im Arbeitsraum über dem Rundtisch über die Transformation und Arbeitsraumdefinition im Agenten des Rundtisches aktiviert und bedarf keiner Anpassungen am Interaktionsmuster. Der Bahnfolgefehler beläuft sich bei Ausnutzung der maximalen erlaubten Geschwindigkeit auf bis zu 5 mm für die y-Achse (s. Abbildung 6-18). Das zeugt vom einem schlechteren Folgeverhalten als es in der reinen Simulation der Fall war und korreliert in hohen Abweichungen mit Änderungen der Geschwindigkeitsvorgabe, siehe Abbildung 6-20, weshalb angenommen werden kann, dass mit einer Implementierung eines Segmentreglers in den Achsen eine Verbesserung der Performanz des Gesamtsystems erreicht werden kann. Ansonsten zeigt das System auch in den resultierenden Segmentzeiten, siehe Abbildung 6-19, ein ähnliches Bild wie in der Simulation, sodass je nach Position auf der Sollwertvorgabe unterschiedliche Achsen dem Verbund die Segmentzeit vorgeben.

Die Integration von kommerziellen MC-Komponenten ohne eigene Miniatursteuerung ist auch für die Umsetzung des DEVEKOS-Systems mittelfristig eine sinnvolle Tätigkeit. Der Weg von zentral organisierten zu verteilt arbeitenden Systemen ist ein Paradigmenwechsel, der ohne diese Durchgängigkeit ein sehr hohes

Investitionsrisiko birgt. Können stattdessen auch klassische kommerzielle Komponenten für DEVEKOS-Systeme ausgestattet werden, können die Aufwände beim Wechsel zu einem verteilten System deutlich gesenkt werden. Statt Komponenten eines zentralen Produktionssystems zu verschrotten, können Elemente auch in einem DEVEKOS-System wiederverwendet werden. Die vorhergehende Darstellung zeigt, welche Voraussetzungen dafür nötig sind, beziehungsweise mit welchen Einschränkungen die verteilte Interpolation auch mit Bestandskomponenten möglich ist.

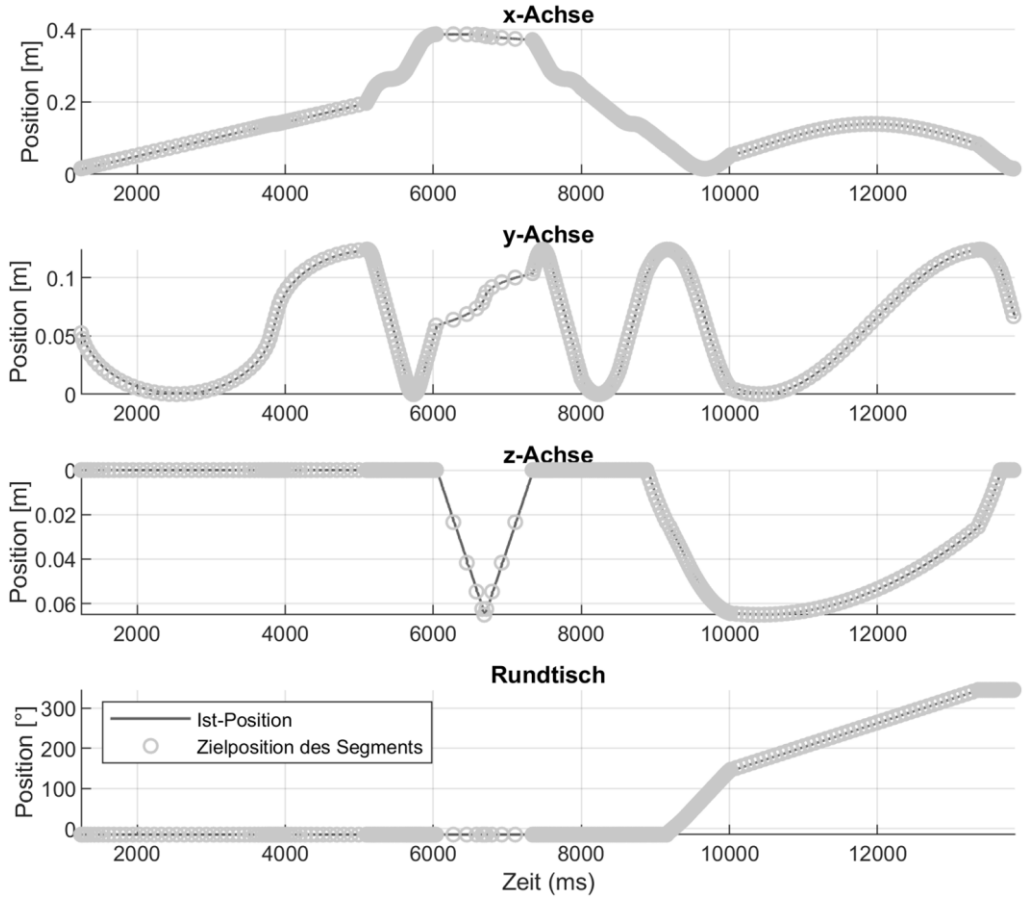

Abbildung 6-17: Positionsverlauf der Achs- und Rundtischbewegungen.

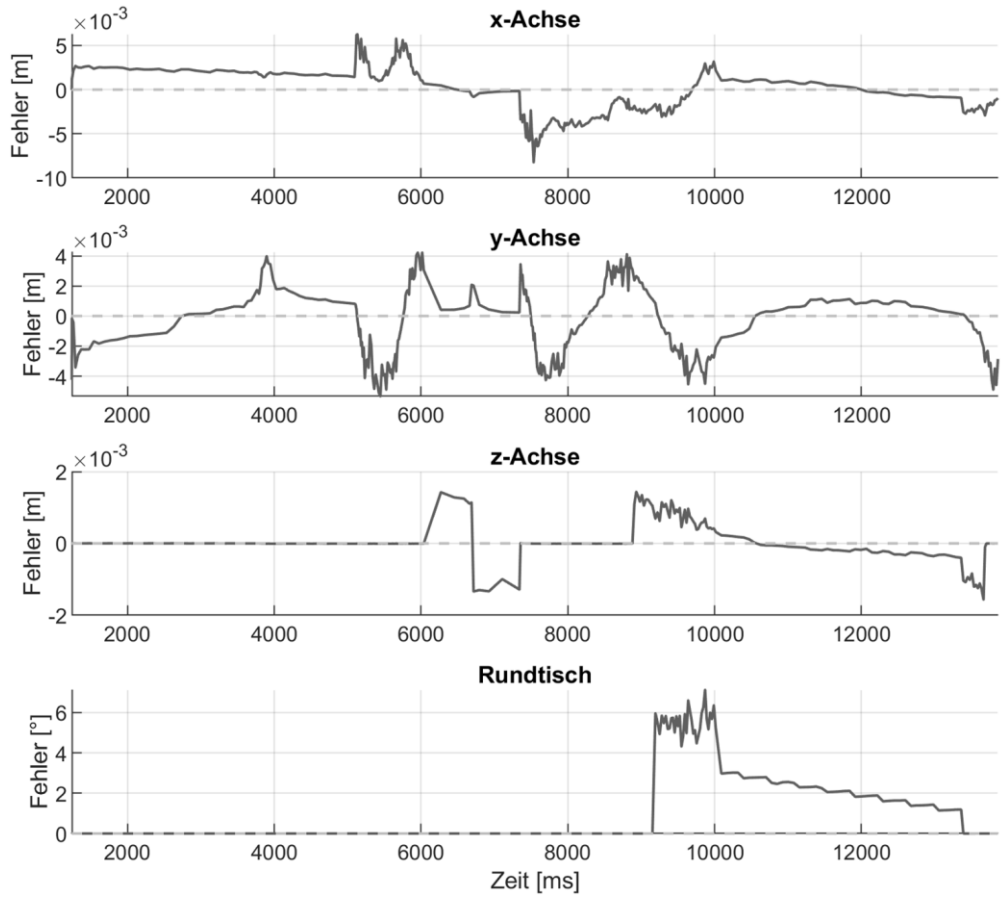

Abbildung 6-18: Folgefehler der kommerziellen Achsen im Verbund.

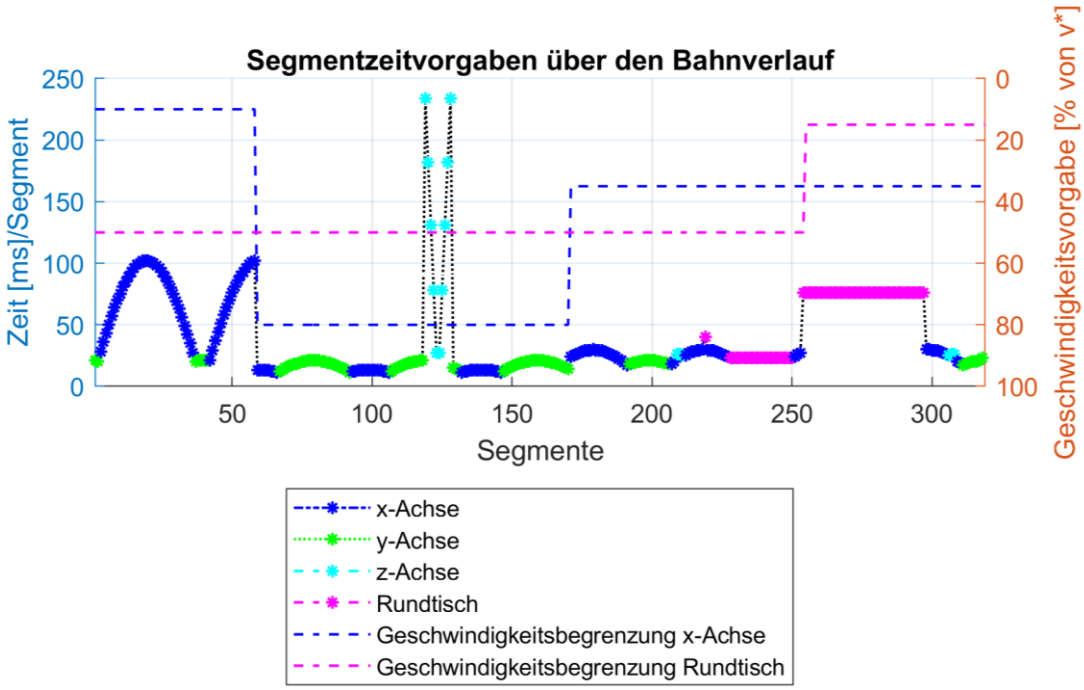

Abbildung 6-19: Resultierende Segmentzeitvorgaben über den Bahnverlauf bei Verwendung der kommerziellen Achsen und des Rundtisches.

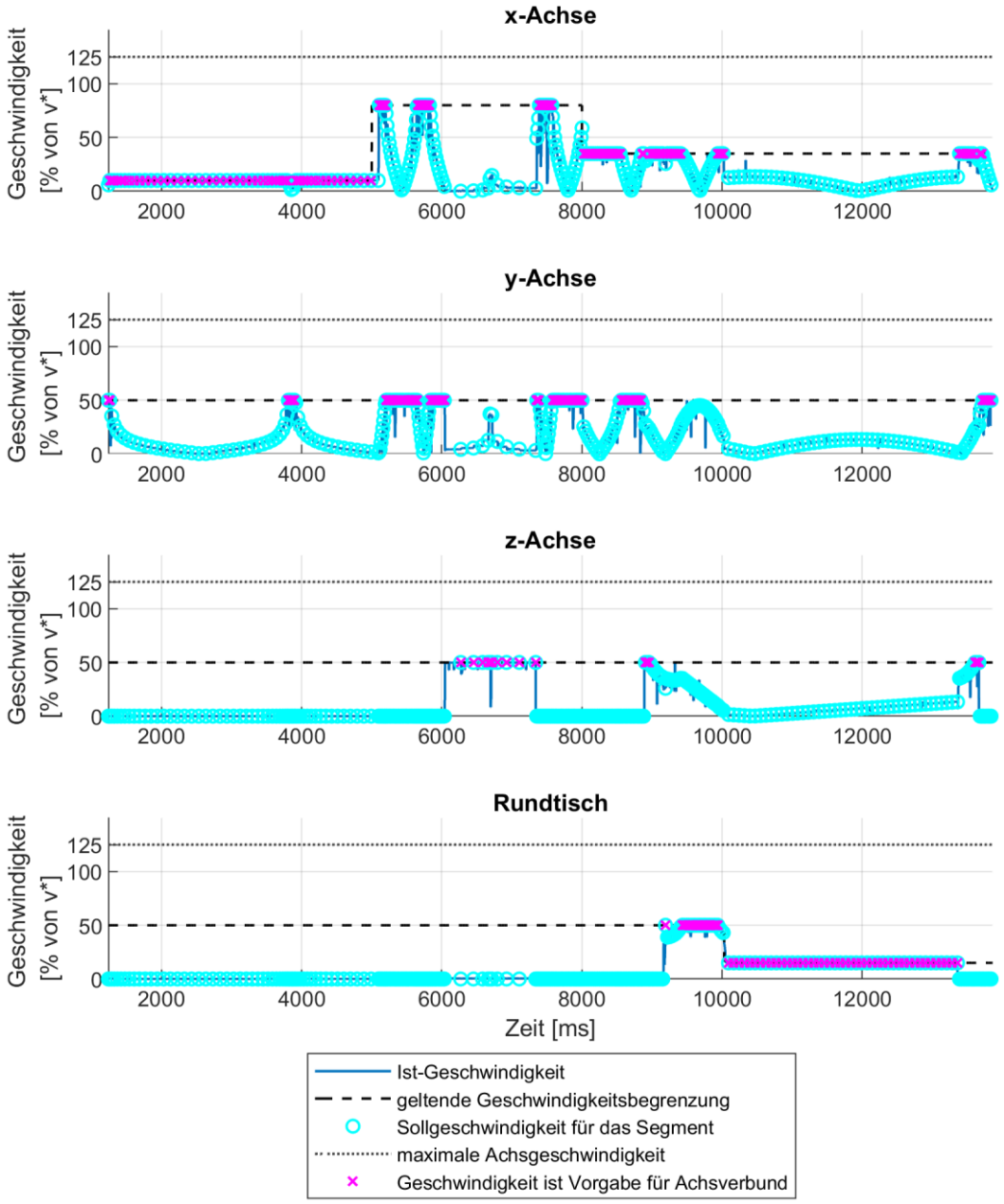

Abbildung 6-20: Geschwindigkeiten der kommerziellen Achsen über den Bahnverlauf. Markiert sind Achsen, die aufgrund der Maximierung der verfügbaren Dynamik die Segmentzielzeit des Verbunds vorgeben.

Abschließend kann festgehalten werden, dass die prototypische Umsetzung des Konzeptes auf kommerziellen Komponenten nur geringe Einschränkungen in der Systemperformanz zeigt, die auf einerseits das Kommunikationsprotokoll und andererseits einen nicht umgesetzten Segmentregler zurückzuführen sind. Speziell die Wahl des Kommunikationsprotokolls kann für industriell eingesetzte Systeme auf zuverlässige Feldbusse oder andere umfangreichere Kommunikationsprotokolle mit höherer Zuverlässigkeit übertragen werden. Die Implementierung eines Segmentreglers ist aufgrund der abgeschlossenen Systeme aktueller kommerzieller Antriebsverstärker absehbar nicht möglich. Hier kann jedoch die prototyp-

isch entwickelte Miniatursteuerung eine Möglichkeit zur benutzerspezifischen Programmierung von Antriebsverstärker und Agentenverhalten aufzeigen.

6.2.4 Integration des Ansatzes in die DEVEKOS-Miniatursteuerung

Die in Kapitel 1.1 vorgestellte Miniatursteuerung in der DEVEKOS-Komponente befindet sich noch im Prototypenstatus. Das Evaluationsboard, das im Projektverlauf zur Verfügung stand, ist in Abbildung 6-21 abgebildet. Die Miniatursteuerung bietet auf der Platine mehrere Elemente (siehe farbliche Markierungen in Abbildung 6-21), die die Umsetzung des Agentenverhaltens, der Achsansteuerung und der Kommunikation vereinfachen:

- Kommunikationsinterface für Ethernet-Kommunikation mit Kommunikationsstack (blau markiert):
  Der Kommunikationsstack kann nach Prinzipien aktueller Produktionsnetzwerke wie der Field Level Communications (FLC) (Lutz 2020) umgesetzt werden und Daten nach dem Protokoll aus Kapitel 5.2.4 übertragen.
- Integrierte MC und Endstufe zur direkten Motoransteuerung mit drei Phasen, Encoder und Hallsensor (gelb markiert):
  Die Ansteuerung des Motors kann mit einem Positionsregler umgesetzt werden und mit einer dynamischen Geschwindigkeitsvorsteuerung über das FPGA erweitert werden.
- FPGA mit integrierter CPU für Logiksteuerung und Programmcode (violett markiert):
  Auf der CPU, beziehungswiese dem FPGA, kann nach der Schnittstellendefinition gemäß Kapitel 6.2.2 folgendes umgesetzt werden, vgl. Tabelle 6-3:
  a) Zusammenführung von lokalem, gespeichertem Wissen, Sensor- und Aktorzustände, Kommunikationssystem und Uhr
  b) Zustandsautomat und Interaktionsmuster des Agentenverhaltens
  c) Berechnungen des Agentenverhaltens der Transformation in Achskoordinaten und Arbeitsraum, Berechnung des Koordinationsvektors, Berechnung des Konsensvektors
  d) Weitergabe der Sollwertvorgaben an MC, Rückmeldung des Segmentreglers über aktuelle Dynamik

Auf diesem ganzheitlichen System existiert keine räumliche Trennung zwischen dem Agenten und der Ausführung der Bewegung in der Aktorik, die in den Umsetzungen mit kommerziellen Achssystemen über Kommunikation überbrückt werden musste und damit zusätzliche Einschränkungen der Funktionalität brachte. Das Kommunikationsnetzwerk zwischen den Agenten kann mit verlässlichen, hochperformanten Kommunikationsprotokollen umgesetzt werden, sodass die Datenrate oder der Datenumfang gegebenenfalls sogar noch gesteigert werden könnte.

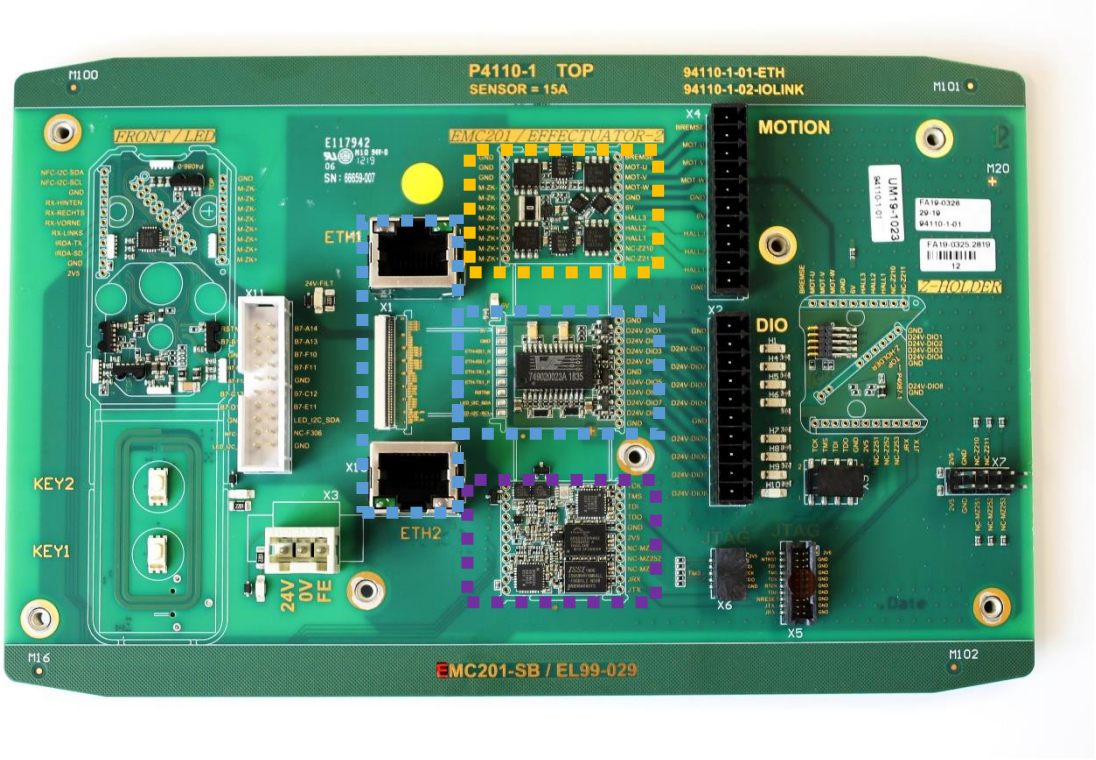

Abbildung 6-21: Prototyp der Miniatursteuerung „EMC 201 Effectuator“ auf einem Evaluationsboard mit Kommunikationsinterface (blau), MC und Endstufe zur Motoransteuerung (gelb) sowie FPGA mit integrierter CPU (violett; auf der Rückseite verbaut).

Der vorhandene Prototyp der Miniatursteuerung wurde mit dem Simulationsmodell aus Kapitel 6.1.1 per UDP-Kommunikation gekoppelt. Der Kommunikationsstack verarbeitet das spezifizierte Datenprotokoll aus Kapitel 5.2.4 und extrahiert die Daten. Das Agentenverhalten wurde in embedded C in der CPU umgesetzt. Die Achsansteuerung der Miniatursteuerung geschieht zum aktuellen Zeitpunkt über ein simuliertes Achsmodell. Der Segmentregler als Vorsteuerung wurde ebenfalls in embedded C realisiert. Eine Kopplung der Bewegungen der Achsen im Simulationsmodell und der simulierten Achse auf der Miniatursteuerung konnte erfolgreich umgesetzt werden.

6.3 Bewertung von Potential und Funktionsumfang der verteilten Interpolation

Der Erfüllungsgrad der Anforderungen an eine Lösung für die verteilte Interpolation aus Kapitel 3.2 wird wie folgt bewertet:

- Umsetzung einer Bewegungssynchronisierung:
  Es wurde eine Umsetzung der vorgegebenen Bahn mit möglichst geringer Bahnabweichung über die agentenbasierte verteilte Interpolation realisiert. Über die Koordinationsvariable kann der Achsverbund auch bei Abweichungen durch veränderlichen Achsdynamiken, Lasten und weiteren Komplikationen eine hohe Bahntreue umsetzen.

- Einklang mit dem DEVEKOS -Systemverständnis:
  Alle Berechnungen für die verteilte Interpolation werden innerhalb der Miniatursteuerung durchgeführt. Alle Achsen im Achsverbund sind gleichwertig ohne Hierarchie miteinander verbunden. Es müssen keine weiteren Komponenten eingesetzt werden.
- Gleichwertige Verteilung der Rechenleistung:
  Alle Komponenten berechnen eigene Koordinationsvektoren sowie den Konsensvektor für die Bewegung. Alle Miniatursteuerungen tragen eine ähnliche Rechenlast.
- Angemessene Aufwände bei Rekonfigurationen:
  Das Entfernen oder Hinzufügen von Komponenten sowie die Rekonfiguration des Aufbaus wurde in Kapitel 6.2.1 erläutert und benötigt für Rekonfigurationen in nur einzelnen Schritten Anpassungen im Agentenverhalten mit geringem Aufwand.
- Angemessene Realisierungskomplexität:
  Die Realisierungskomplexität des Interaktionsmusters wird nach der Schnittstellendefinition aus Kapitel 6.2.2 als mäßig erachtet. Eine Übertragbarkeit in industrielle Anwendungen und Produkte ist über die Umsetzung eines Zustandsautomaten, Erweiterung um Algorithmen für die Berechnung von Koordinationsvektor sowie Konsensvektor und die Umsetzung eines Segmentreglers möglich. Das Agentenverhalten und Kommunikationsprotokoll wurde erfolgreich auf einer prototypischen Miniatursteuerung umgesetzt.

Zur Beantwortung der Forschungsfrage

> Forschungsfrage:
>
> Welches Potential und welchen Funktionsumfang kann eine agentenbasierte Bewegungssynchronisierung aufzeigen?

wird nachfolgend kritisch betrachtet, welcher Funktionsumfang von der realisierten Lösung aktuell nicht abgedeckt wird und diskutiert, unter welchen Voraussetzungen die verteilte Interpolation eingesetzt werden kann.

Die verteilte Interpolation mit Durchführung eines Interaktionsmusters durch Agenten der einzelnen Achsen erreicht die Lösung eines globalen Optimierungsproblems, genauer der bahntreuen synchronen Bewegung. Diese kann auf allen Achsen entsprechend ihrer zur Verfügung stehenden Dynamik umgesetzt werden. Es wird durch die Interaktion der lokalen Agenten eine Bewegung zur Durchführung ausgewählt, die Bahntreue unter minimaler Abarbeitungszeit und das Ausnutzen der zur Verfügung stehenden Dynamik bei geringem Kommunikationsbedarf verspricht. Erhöhter Kommunikationsbedarf entsteht nur dann, wenn aufgrund von Abweichungen eine erneute Koordination der Achsen notwendig wird. Aufgrund des gewählten Interaktionsmusters ist ein veränderter Konsens bereits nach wenigen Berechnungsschritten wirksam. Die Voraussetzung in beiden Fällen ist eine verlässliche und echtzeitfähige Kommunikation.

Der Konsensvektor wird parallel von allen im Achsverbund beteiligten Achsen berechnet. Um sicherzustellen, dass diese Berechnung auch bei sich unterscheidenden Implementierungen zum gleichen Ergebnis kommt, wird hier lediglich die

einfache Bestimmung des Maximums durchgeführt. Die Berechnung des Koordinationsvektors kann entweder, wie vorgeschlagen, mit der Annahme einer mittleren Geschwindigkeit oder nach einer anderen Approximation vorgenommen werden. Solange der implementierte Segmentregler Restdynamik der Achse verwenden kann, um die Konsenszeit auch bei Fehlern in der Zeitspannenberechnung der Koordinationsvariablen einzuhalten, sind abweichende Berechnungen in sich unterscheidenden Implementierungen akzeptabel.

Die Umsetzung der verteilten Interpolation besitzt in der vorgestellten Realisierung folgende implizite Einschränkungen:

- Bahninterpolation als lineare Interpolation:
  Das vorgestellte Interaktionsmuster kann zwischen Sollwerten der Bahnvorgabe lediglich lineare Interpolationen durchführen. Ein Überschleifen über eine vorgezogene Schaltbedingung, s. Kapitel 5.2.3, ermöglicht das kontinuierliche Verfahren auf der Bahn. Die Erweiterung auf weitere Interpolationsmuster wie Kreis- oder Polynomkurven benötigt eine Kopplung der Positionsverläufe, da eine implizite Kopplung ohne Positionsabhängigkeit wie in Kapitel 5.2.1 vorgestellt, nicht gegeben ist.
- Verwendung ähnlicher Beschleunigungsprofile:
  Es wird vorausgesetzt, dass in allen beteiligten Achsen ähnliche Beschleunigungsprofile umgesetzt sind, um die implizite Kopplung zu erhalten. Die Beschleunigungsprofile müssen durch den Segmentregler zeitskalierbar sein. Elektrische Antriebe besitzen in den meisten Fällen sehr ähnliche Beschleunigungsprofile, die lediglich in der Ausprägung der Beschleunigungsphase als Siebenphasenprofil, S-Kurve, sinusquadratische Beschleunigung, usw. voneinander abweichen. Die Vorsteuerung über den Segmentregler wird als ausreichend angenommen um bleibende Bahnabweichungen über den Verlauf des Segments auszuregeln.
- Kinematische Ketten von linear unabhängigen Achsen:
  Im betrachteten Anwendungsfall der Rekonfiguration wird davon ausgegangen, dass in kinematischen Ketten nur linear unabhängige Achsen verbaut werden. Linear abhängige oder redundante Achsen werden nur in feststehenden Bezugssystemen verwendet. Für die Erweiterung des Konzeptes zur Verwendung von linear abhängigen Achsen innerhalb kinematischer Ketten ist die Transformation der Bahnkoordinaten in die Achskoordinaten abhängig von der Position der tragenden linear abhängigen Achsen Voraussetzung. Dies bedeutet zum einen, dass die aktuelle Position der tragenden Achse zu allen Zeitpunkten bekannt sein oder kommuniziert werden muss. Weiter bedeutet dies eine Neubewertung der eigenen Position relativ zu den Bahnkoordinaten bei Veränderung in der Position der tragenden Achsen, mit daraus folgender Neuberechnung des eigenen Koordinationsvektors. Zu klären bleibt, wie die Redundanz der linear abhängigen Achsen auf zwei oder mehr Achsen aufgeteilt wird. Bei linear abhängigen Achsen, die in verschiedenen kinematischen Ketten und damit festen Bezugssystemen verortet sind, ist die Planung einer solchen Redundanz hinfällig, da die parallele Bewegung entlang der Bahn in den meisten Fällen das Ziel der Anwendung ist.

- Annahme der korrekten Kinematisierung des Anwendungsfalls:
  Die Skalierbarkeit des Interaktionsmusters auf beliebig viele oder wenige Teilnehmer ohne explizite Anpassung des Algorithmus hat als Vorbedingung, dass der Achsverbund entsprechend des Anwendungsfalls kinematisiert, also mit Achsen der korrekten Dimensionen, Arbeitsräumen und dynamischen Fähigkeiten ausgestattet worden ist. Es ist keine Kontrollinstanz implementiert, die sicherstellt, dass alle Dimensionen der Bahnvorgabe durch die beteiligten Achsen abgedeckt werden oder die Arbeitsräume der Achsen den Arbeitsraum der Bahn nicht unterschreiten.

Unter den oben genannten Einschränkungen konnte eine agentenbasierte Systemarchitektur mithilfe eines speziell entwickelten Agentenverhaltens erarbeitet und simulativ wie am realen Anwendungsfall gezeigt werden, dass damit eine Bewegungssynchronisierung mit einer Genauigkeit im Millimeterbereich erreicht werden kann. Dabei wurden zwei bestehende Interaktionsmuster von MAS erfolgreich miteinander kombiniert und von Anwendungen der Schwarmrobotik auf die Verwendung mit positionierenden Komponenten, die einen gemeinsamen Arbeitsraum teilen, übertragen. Eine einfache Zustandsmaschine und die Verwendung von approximierten Geschwindigkeiten ermöglichen die einfache Umsetzung des Verhaltens auch auf Miniatursteuerungen mit geringer Rechenkapazität oder die Erweiterung und Integration von weiteren Komponenten. Da als Koordinationsvariable der Zeitbedarf pro Segment festgelegt wurde, ist auch die Ausweitung des Interaktionsmusters auf nichtpositionierende Komponenten denkbar.

Folglich kann festgehalten werden, dass die vorgestellte verteilte Interpolation Funktionen der Bewegungssynchronisierung in verschiedenen Anwendungsfällen umsetzen kann. Die oben genannten Einschränkungen sind unter anderen für den Bereich der wandelbaren Montagelinien, für Gütertransport oder für Handhabungsaufgaben realistisch einzuhalten. Zentrale Ansätze bieten zwar die höchste Gesamtperformance bei geringem Kommunikationsaufwand, aber die verteilte Interpolation ermöglicht eine gute Performanz für echt dezentrale Lösungen. Dabei ist der Rechenbedarf in den Miniatursteuerungen vergleichsweise gering und die Einhaltung der Spezifikation einer kompakten Schnittstelle ermöglicht Rekonfigurationen oder Erweiterungen, die schnell umsetzbar sind. In Sonderfällen, die höherwertige Interpolationen, Redundanzen in den Kinematiken oder die Kombination von unterschiedlichen Antriebsarten einsetzen, sind im Allgemeinen Lösungen zu erwarten, die den zusätzlich über die verteilte Interpolation ermöglichten Aspekt der einfachen Rekonfigurierbarkeit nicht abdecken können.

Zielerreichung des Gesamtziels:

Bewegungssynchronisierung für dezentral gesteuerte Mehrachssysteme: Positionierende Komponenten mit Miniatursteuerung können im DEVEKOS-Systemverständnis ohne eine zentrale Steuerung synchronisierte Bewegungen umsetzen (Gesamtziel).

Als Potential für die verteilte Interpolation können weitere Innovationen bei Produktionsnetzwerken und Miniatursteuerungen genannt werden. Sobald höhere Datenaustauschraten und höhere Rechenlast in den Steuerungen verfügbar sind, können auch explizite Kopplungen der Positionsverläufe über die Kommunikation gelöst werden, sodass beispielweise höherwertige Interpolationen durch die Annäherung an eine Bahnregelung oder die Verwendung von linear abhängigen Achsen innerhalb einer kinematischen Kette inklusive der Klärung ihrer Redundanz umsetzbar wären. Auch die Umsetzung von Beobachtermodellen zur Überbrückung von Phasen ohne Kommunikation wie in (Sackenreuther 2019; Quapil 2020) ist denkbar. Bereits kurzfristig ermöglicht die Synchronisierung von Beschleunigungsprofilphasen; wie im Anhang diskutiert, eine noch höhere Bahntreue und Synchronisierung des Achsverbunds. In allen Fällen sollte die Stabilität des Gesamtsystems bei erhöhter Austauschrate zwischen den Agenten kritisch betrachtet werden.

Die Integrationsfähigkeit von Bestandskomponenten in die verteilte Interpolation nach der definierten Schnittstelle aus Kapitel 6.2.2 ermöglicht den Paradigmenwechsel von aktuellen, zentral gesteuerten Automatisierungssystemen hin zu einer Steuerung von Bewegungssynchronisierung ohne zentrale steuernde Instanz als fließenden Übergang oder in Teilsystemen von bisher hierarchisch gesteuerten Systemen.

7 Zusammenfassung und Ausblick

Wandelbare und rekonfigurierbare Produktionseinheiten sind Schlüsseltechnologien für die Fabrik der Zukunft. Mit aktuellen Entwicklungen der Mikroelektronik, Netzwerk- und Kommunikationstechnik besteht erstmals die Möglichkeit, großflächig Automatisierungskomponenten mit sogenannten Miniatursteuerungen auszustatten. Solche Komponenten können nach Bedarf miteinander zu Verbänden und Subsystemen konfiguriert werden. Diese kapseln die Ansteuerung der Aktorik. Im Verbund führen Komponenten gemeinsame Tätigkeiten wie die definierte Bewegung aus. Da bereits Miniatursteuerungen in jeder Komponente vorhanden sind, soll für die Koordination solcher Bewegungen keine zentrale Steuerungsinstanz verwendet werden, sondern stattdessen die auf den Komponenten verteilt zur Verfügung stehende Rechenkapazität genutzt werden.

Die synchrone Bewegung mehrerer Achsen auf einer Bahn ist ein spezieller Anwendungsfall, der in der Steuerungstechnik in den meisten Fällen durch eine zentrale Koordination umgesetzt wird. Es sollte in dieser Arbeit untersucht werden ob und in welchem Funktionsumfang eine synchrone Bewegungssteuerung ohne zentrale Instanz in einer agentenbasierten Systemarchitektur umgesetzt werden kann. Zusätzlich sollte der Anwendungsfall der Rekonfiguration betrachtet werden, sodass weitere Komponenten möglichst einfach in Verbünde integriert werden können.

Obwohl bereits Forschungsarbeiten im Bereich der Steuerung von positionierenden Achsen ohne zentrale Steuerungsinstanz existieren, geht keine der Arbeiten bisher auf den Faktor der bahntreuen Verfahrbewegung oder der Rekonfiguration des Achsverbunds ein. Es wurden daher mögliche Systemkonzepte für eine verteilte synchrone Bewegungssteuerung aus dem Stand der Technik und Forschung abgeleitet und hinsichtlich ihrer Eignung analysiert. Echte Dezentralität und einfache Rekonfiguration sind nur dann möglich, wenn alle Komponenten ein möglichst identisches Verhalten zeigen und per Netzwerk kommunizieren können. Dieser Ansatz wird verteilte Interpolation genannt und im Verlauf der Arbeit umfassend analysiert, die notwendigen Interaktionsmuster spezifiziert sowie in einer realen Anwendung validiert. Die Analyse des Systemkonzeptes und der zu bewältigenden Aufgabe wurde entlang des DACS-Vorgehensmodells (Designing Agent-based Control Systems) von (Bussmann et al. 2004, S. 119) vorgenommen. Anhand von weit verbreiteten Ansätzen zur Konsensfindung und Bewegungskoordination bei Multiagentensystemen wurde ein Interaktionsmuster für die verteilte Interpolation entwickelt. Hierbei wurde die Kombination von Formationsbewegung und Rendez-Vous-Verfahren mit der impliziten positionsunabhängigen Kopplung von Achsdynamiken bei der linearen Interpolation zwischen Sollwerten kombiniert, um einen Synchronisierungsmechanismus zu konzipieren. Für die daraus resultierende verteilte Interpolation wurde ein schlankes Kommunikationsprotokoll spezifiziert und die Umfänge der Anpassungen bei Rekonfigurationen möglichst geringgehalten. Eine Schnittstellendefinition ermöglicht ebenfalls die Integration von Bestandskomponenten, sodass die verteilte Interpolation mit in einem Achsverbund von Komponenten mit Miniatursteuerung sowie kommerziellen Achsen durchgeführt werden kann.

Abschließend folgte eine Beurteilung des Funktionsumfangs und des Potentials einer solchen verteilten Systemarchitektur für die Interpolation. Die erarbeitete Lösung erfüllt die Anforderungen der Umsetzung einer Bewegungssynchronisierung in Einklang mit dem DEVEKOS-Systemverständnis bei gleichwertiger Verteilung der Rechenleistung. Gleichzeitig sind die Aufwände bei Rekonfigurationen sowie die Realisierungskomplexität vergleichsweise gering. In einer kritischen Analyse wurde die Einschränkung der abgedeckten Anwendungsfälle diskutiert. Diese Einschränkungen basieren hauptsächlich auf der Prämisse der impliziten Kopplung aller Achsen. Explizite Kopplungen, die bei einer komplexeren Bahndefinition, stark unterschiedlichen Beschleunigungsprofilen oder linear abhängigen Kinematiken notwendig wären, benötigen erhöhten Kommunikationsaufwand und Rechenbedarf in der Miniatursteuerung.

Als Aussicht für weitere Forschungsarbeiten ist zu nennen, dass bei der Weiterentwicklung von Produktionsnetzwerkkommunikation und Mikroelektronik solche Entwicklungen in der nahen Zukunft umsetzbar sind. Es können dann auch statt einer identischen Umsetzung des Agentenverhaltens einzelner Komponenten explizit verschiedenen Sichten auf ein gemeinsames, zentrales Systemmodell auf den Miniatursteuerungen integriert werden. Zu entwickeln ist dann ein gemeinsames wandlungsfähiges Systemmodell, aus dem in einfacher Art und Weise Sichten zur Verwendung in einzelnen Komponenten abgeleitet werden können.

8 Literaturverzeichnis

3S 2016
3S-Smart Software Solutions GmbH. *CODESYS® Application Composer: Konfektionieren von IEC 61131-3 Applikationssoftware für Automatisierungssysteme*, 08/2016. Verfügbar unter: https://de.codesys.com/index.php?eID=tx_securedownloads&p=107&u=0&g=0&t=1589818903&hash=781fdc83b3acf217dd67e75a6a61816a2f971d5f&file=/fileadmin/Download/DE/Brochures/CODESYS-Application-Composer-de.pdf
Zugriff am: 17.05.2020

Altintas 2012
Altintas, Yussuf, 2012. *Manufacturing Automation: metal cutting mechanics, machine tool vibrations, and CNC design.*
2nd edition.
New York: Cambridge University Press.
ISBN 978-0-521-17247-9

Arai et al. 2002
Arai, Tamio; Pagello, Enrico; Parker, Lynne E., 2002. Editorial: Advances in multirobot systems.
IEEE Transactions on Robotics and Automation **18** (5), S. 655–661
DOI: 10.1109/TRA.2002.806024

Axmann 2019
Axmann, Etienne, 2019. Real-time publishing with response (2019).
Whitepaper der VDMA Fachabteilung Integrated Assembly Solutions
Verfügbar unter: https://rua.vdma.org/documents/106005/45540632/Whitepaper+Real-time+publishing+with+response+%282019%29.pdf/0e8db974-1d07-dbbe-51ae-4df2e1bdb1e0

Balch et al. 1998
Balch, Tucker; Arkin, Ronald C., 1998. Behavior-based formation control for multirobot teams.
IEEE Transactions on Robotics and Automation **14** (6), S. 926–939

Bauder 1992 Bauder, Manfred, 1992. *Konfigurierbare Robotersteuerung mit allgemeiner Transformation*, Diss., 1992. ISW Forschung und Praxis - Berichte aus dem Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen der Universität Stuttgart 92. ISBN 0-387-55433-5 Verfügbar unter: http://swbplus.bsz-bw.de/bsz029067995cov.htm

Beard et al. 2001 Beard, Randal W; Lawton, Jonathan; Hadaegh, Fred Y., 2001. A coordination architecture for spacecraft formation control. *IEEE Transactions on Control Systems Technology* **9** (6), S. 777–790

Beckhoff 2010 Beckhoff Automation GmbH & Co. KG, 2010. *Realtime-Ethernet mit TwinCAT-Netzwerkvariablen: Application Note DK9321-0110-0024*, Januar 2010. Verfügbar unter: https://download.beckhoff.com/download/document/Application_Notes/DK9321-0110-0024.pdf Zugriff am: 17.04.2020

Beckhoff 2017 Beckhoff Automation GmbH & Co. KG, 2017. *AMP8000 | Distributed Servo Drive system: Distributed servo drives for modular machines*. Verfügbar unter: https://www.beckhoff.ch/default.asp?drive_technology/amp8000.htm?id=3471194472722 Zugriff am: 26.06.2018

Beckhoff 2020 Beckhoff Automation GmbH & Co. KG, 2020. *Handbuch TC3 NC Camming: Version 1.2*, 16.04.2020. Verfügbar unter: https://download.beckhoff.com/download/Document/automation/twincat3/TF5050_TC3_NC_Camming_DE.pdf Zugriff am: 17.04.2020

Beckhoff 2020
Beckhoff Automation GmbH & Co. KG, 2020.
TwinCAT NC: FIFO Achsen.
Beckhoff Information System.
Verfügbar unter: https://infosys.beckhoff.de/index.php?content=./content/1031/tcsampleptp/tcsampleptp_fifoaxis/html/tcsamplefifoaxis_intro.htm&id=
Zugriff am: 07.02.2020

Beekman et al. 2008
Beekman, Madeleine; Sword, Gregory A; Simpson, Stephen J., 2008.
Biological Foundations of Swarm Intelligence.
In: Blum, Christian; Merkle, Daniel (Hrsg.)*:*
Swarm Intelligence.
Berlin, Heidelberg: Springer, S. 3–41
ISBN 978-3-540-74089-6

Binder 1979
Binder, Dietmar, 1979.
Interpolation in numerischen Bahnsteuerungen, Diss.
Berlin, Heidelberg: Springer Berlin Heidelberg.
ISW Forschung und Praxis - Berichte aus dem Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen der Universität Stuttgart 24.
ISBN 978-3-540-09007

Bosch Rexroth 2007
Bosch Rexroth AG, 2007.
Rexroth IndraDrive: Firmware für Antriebsregelgeräte MPH-05, MPB-05, MPD-05, 10/2007.
Verfügbar unter: https://docplayer.org/4839869-Rexroth-indradrive-firmware-fuer-antriebsregelgeraete-mph-05-mpb-05-mpd-05.html
Zugriff am: 17.04.2020

Bracht et al. 2018
Bracht, Uwe; Geckler, Dieter; Wenzel, Sigrid, 2018.
Digitale Fabrik: Methoden und Praxisbeispiele.
2., aktualisierte und erweiterte Auflage.
Berlin: Springer Vieweg.
ISBN 978-3-662-55783-9

Brambilla et al. 2013
Brambilla, Manuele; Ferrante, Eliseo; Birattari, Mauro; Dorigo, Marco, 2013. Swarm robotics: A review from the swarm engineering perspective.
Swarm Intelligence **7** (1), S. 1–41
DOI: 10.1007/s11721-012-0075-2

Brazier et al. 1997 Brazier, Frances M. T; Dunin-Keplicz, Barbara M; Jennings, Nick R; Treur, Jan, 1997. Desire: Modelling Multi-Agent Systems in a Compositional Formal Framework.
International Journal of Cooperative Information Systems **6** (1), S. 67–94
DOI: 10.1142/S0218843097000069

Bussmann et al. 2004 Bussmann, Stefan; Jennings, Nicolas R; Wooldridge, Michael J., 2004.
Multiagent systems for manufacturing control: a design methodology.
Berlin, Heidelberg, New York: Springer Verlag.
ISBN 978-3-642-05890-5
DOI: 10.1007/978-3-662-08872-2

Cormen et al. 2009 Cormen, Thomas H; Leiserson, Charles E; Rivest, Ronald L; Stein, Clifford, 2009.
Introduction to algorithms.
Cambridge, Massachusetts, London, England: MIT press.
ISBN 0262533057

Correll et al. 2013 Correll, Nikolaus; Rus, Daniela, 2013.
Architectures and control of networked robotic systems.
In: Kernbach, Serge (Hrsg.)*: Handbook of Collective Robotics: fundementals and challenges*: CRC press, S. 81–104

Decotignie 1991 Decotignie, J-D, 1991.
Distributed path and speed control in machine-tool axis motion.
In: *Proceedings of the International Conference on Industrial Electronics, Control and Instrumentation (IECON) 1991*, S. 772–777

Desai et al. 1998 Desai, Jaydev P; Ostrowski, Jim; Kumar, Vijay, 1998.
Controlling formations of multiple mobile robots.
In: *Proceedings of the IEEE International Conference on Robotics & Automation,*
Mai 1998, Leuven, Belgien, 2864-2869

Dittrich et al. 1996
Dittrich, Günter; Müller, Rainer, 1996. Bahnplanung.
In: Ameling, Walter (Hrsg.)*: Grundlagen und Komponenten flexibler Handhabungsgeräte im Maschinenbau*.
Weinheim: VCH Verlagsgesellschaft mbH, S. 56–65
ISBN 3-527-27721-8

Dong 2016
Dong, Xiwang, 2016.
Formation and Containment Control for High-order Linear Swarm Systems, Diss.
Heidelberg New York Dordrecht London: Springer Berlin Heidelberg.
Springer Theses, Recognizing Outstanding Ph.D. Research.
ISBN 9783662478356
Verfügbar unter: http://dx.doi.org/10.1007/978-3-662-47836-3DOI: 10.1007/978-3-662-47836-3

Doran et al. 1997
Doran, Jim E; Franklin, SRJN; Jennings, Nicholas R; Norman, Timothy J., 1997. On cooperation in multi-agent systems.
The Knowledge Engineering Review **12** (3), S. 309–314

Dripke et al. 2017
Dripke, Caren; Verl, Alexander, 2017. Challenges in distributed interpolation with multi-components systems in future-oriented manufacturing units.
In: *Proceedings of the 47th International Conference on Computers and Industrial Engineering (CIE) 2017,*
11.10.2017 - 13.10.2017, Lissabon, Portugal

Dripke et al. 2018
Dripke, Caren; Schneider, Ben; Dragan, Mihai; Zoitl, Alois; Verl, Alexander, 2018. Concept of Distributed Interpolation for Skill-Based Manufacturing with Real-Time Communication.
In: *Tagungsband des 3. Kongresses Montage Handhabung Industrieroboter,*
27.02.2018 - 28.02.2018, Erlangen, Deutschland, S. 215–222

Dripke et al. 2018
Dripke, Caren; Laicher, Sonja; Verl, Alexander, 2018.
Axis-in-the-loop simulation: a concept to transfer verification and validation of distributed interpolation algorithms from simulation to interaction with commercial motion control hardware.
In: *Proceedings of the 48th International Conference on Computers and Industrial Engineering (CIE) 2018,*
02.12.2018 - 06.12.2018, Auckland, Neuseeland

Dripke et al. 2019
Dripke, Caren; Ye, Zhongyi; Verl, Alexander, 2019.
Synchronization of a distributed interpolation application via cross-coupled control.
In: *IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC) 2019,*
17.06. - 20.06.2019, Sophia Antipolis, Frankreich

Dripke et al. 2020
Dripke, Caren; Schöbel, Daniel; Verl, Alexander, 2020.
Distributed Interpolation: Synchronization of motion-controlled axes with coordination vector and decentralized segment controllers.
In: *IEEE Advanced Motion Control Worshop (AMC) 2020,*
14.09.2020 - 16.09.2020, Kristiansand, Norwegen

Dürkop 2016
Dürkop, Lars, 2016.
Automatische Konfiguration von Echtzeit-Ethernet.
Berlin: Springer Vieweg.
Technologien für die intelligente Automation 5.
ISBN 978-3-662-54124-1
DOI: 10.1007/978-3-662-54125-8

Eclipse Foundation 2007
Eclipse Foundation, 2007.
Eclipse 4diac: The Open Source Environment for Distributed Industrial Automation and Control Systems.
Verfügbar unter: https://www.eclipse.org/4diac/
Zugriff am: 17.05.2020

Egerstedt et al. 2000 Egerstedt, Magnus; Hu, Xiaoming, 2000. Coordinated trajectory following for mobile manipulation. In: *Proceedings of the 2000 IEEE International Conference on Robotics & Automation,* April 2000, San Francisco, CA, 3479-3484

Emery et al. 2002 Emery, Rosemary; Sikorski, Kevin; Balch, Tucker, 2002. Protocols for collaboration, coordination and dynamic role assignment in a robot team. In: *Proceedings of the 2002 IEEE International Conference on Robotics & Automation,* Mai 2002, Washington, DC, S. 3008–3015

FANUC 2018 FANUC Europe Corporation, 2018. *Power Motion i: CNC- und SPS-Technologie für Hochleistung in Motion Control Anwendungen*, 12/2018. Verfügbar unter: https://www.fanuc.eu/~/media/files/pdf/products/cnc/brochures/mbr-00072-fa%20power%20motion%20i-v3/fanuc-power-motion-de.pdf?la=de Zugriff am: 12.05.2020

Foderaro et al. 2014 Foderaro, Greg; Ferrari, Silvia; Wettergren, Thomas A., 2014. Distributed optimal control for multi-agent trajectory optimization. *Automatica* **50**, S. 149–154 DOI: 10.1016/j.automatica.2013.09.014

Fowler et al. 2002 Fowler, Jeffrey M; D'Andrea, Raffaello, 2002. Distributed control of close formation flight. In: *Proceedings of the 41st IEEE Conference on Decision and Control,* Dezember 2002, Las Vegas, Nevada, USA, 2972-2977

Frazzoli et al. 2002 Frazzoli, Emilio; Dahleh, Munther A; Feron, Eric, 2002. Real-time motion planning for agile autonomous vehicles. *Journal of guidance, control, and dynamics* **25** (1), S. 116–129 DOI: 10.2514/2.4856

Fritsch 2009 Fritsch, Dennis, 2009. *Steuerung selbstorganisierender Multi-Roboter-Systeme für dynamische Sammelaufgaben am Beispiel der Bekämpfung maritimer Ölverschmutzungen*, Diss., 2009. Heimsheim: Jost-Jetter. ISW Forschung und Praxis - Berichte aus dem Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen der Universität Stuttgart 173. ISBN 3939890510

Gokulan 2011 Gokulan, Balaji Parasumanna, 2011. *Distributed multi-agent based traffic management system*, Diss. 2011

Grigoriev et al. 2013 Grigoriev, Sergej Nikolaevich; Martinov, Georgi Martinov, 2013. Decentralized CNC automation system for large machine tools. In: *International Conference on Competitive Manufacturing (COMA'13)*, S. 295–300

Grigoriev et al. 2016 Grigoriev, Sergej N; Martinov, Georgi M., 2016. An ARM-based Multi-channel CNC Solution for Multi-tasking Turning and Milling Machines. In: *7th HPC 2016 – CIRP Conference on High Performance Cutting*, S. 525–528 Verfügbar unter: http://www.sciencedirect.com/science/article/pii/S2212827116301913DOI: 10.1016/j.procir.2016.04.036

Gutt 2014 Gutt, Hans-Joachim, 2014. Entwicklung der Elektrischen Antriebstechnik zur Informations-Aktorik. *e & i Elektrotechnik und Informationstechnik* **131** (7), S. 237–245 Verfügbar unter: https://link.springer.com/content/pdf/10.1007/s00502-014-0223-z.pdfDOI: 10.1007/s00502-014-0223-z

Hamann 2018 Hamann, Heiko, 2018. *Swarm Robotics: A Formal Approach*. Cham: Springer International Publishing. ISBN 978-3-319-74526-8 DOI: 10.1007/978-3-319-74528-2

Harms 2004
Harms, Thomas, 2004.
Agentenbasierte Fabrikstrukturplanung, Diss., 2004.
Garbsen: PZH, Produktionstechn. Zentrum.
Berichte aus dem IFA 02/2004.
ISBN 3936888485

Hees 2017
Hees, Andreas Fabian, 2017.
System zur Produktionsplanung für rekonfigurierbare Produktionssysteme, Diss. 2017.
München: Herbert Utz Verlag 331.
ISBN 3831646767

Heine 2015
Heine, Andreas, 2015.
Ein Beitrag zur kennwertorientierten Entwicklung kurvengesteuerter, ebener Schrittgetriebe, Diss., 2015.
Wissenschaftliche Schriftenreihe der Chemnitzer Montage- und Handhabungstechnik Band 1.
ISBN 9783944640518
Verfügbar unter: https://monarch.qucosa.de/api/qucosa%3A20222/attachment/ATT-0/

Helbig 2017
Helbig, Tobias Bernhard, 2017.
Methode zur Verbesserung der domänenübergreifenden Zusammenarbeit während des Engineering-Prozesses im Sondermaschinenbau, Diss., 2016: Shaker.
ISBN 3844050973

Henning et al. 2015
Henning, Steffen; Niggemann, Oliver; Brandenbourger, Benjamin; Helbig, Tobias, 2015.
Plug-and-Produce für Cyber-Physische Produktionssysteme - Eine Fallstudie im OPAK-Projekt.
In: *Automation 2015*

Hirsch 2010
Hirsch, Martin, 2010.
Systematic design of distributed industrial manufacturing control systems, Diss.: Logos Verlag Berlin GmbH.
Hallenser Schriften zur Automatisierungstechnik 6.
ISBN 3832526072

Huan 1982 Huan, Ji, 1982.
Bahnregelung zur Bahnerzeugung an numerisch gesteuerten Werkzeugmaschinen, Diss., 1982.
Berlin, Heidelberg: Springer Berlin Heidelberg.
ISW Forschung und Praxis - Berichte aus dem Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen der Universität Stuttgart 44.
ISBN 978-3-642-45540-7

Huntsberger et al. 2004 Huntsberger, Terry L; Trebi-Ollennu, Ashitey; Aghazarian, Hrand; Schenker, Paul S; Pirjanian, Paolo; Nayar, Hari Das, 2004. Distributed control of multi-robot systems engaged in tightly coupled tasks.
Autonomous Robots **17** (1), S. 79–92

Ishida 1994 Ishida, Toru, 1994.
Parallel, distributed, and multiagent production systems.
Berlin, Heidelberg [u.a.]: Springer.
Lecture notes in computer science 878.
ISBN 0-387-58698-9
Verfügbar unter: http://swbplus.bsz-bw.de/bsz042445698cov.htm

Ivanova-Vasileva et al. 2008 Ivanova-Vasileva, Ioanna; Gerber, Christian; Hanisch, Hans-Michael, 2008. Basics of Modelling IEC 61499 Function Blocks with Integer-Valued Data Types.
IFAC Proceedings Volumes **41** (3), S. 169–174
DOI: 10.3182/20081205-2-CL-4009.00031

Jakovljevic et al. 2019 Jakovljevic, Zivana; Lesi, Vuk; Mitrovic, Stefan; Pajic, Miroslav, 2019. Distributing Sequential Control for Manufacturing Automation Systems.
IEEE Transactions on Control Systems Technology
DOI: 10.1109/TCST.2019.2912776

Jeong et al. 2008 Jeong, Seok-Kwon; You, Sam-Sang, 2008. Precise position synchronous control of multi-axis servo system.
Mechatronics **18** (3), S. 129–140
DOI: 10.1016/j.mechatronics.2007.10.009

Jetter 2010
Jetter AG, 2010.
Motion Control: Bewegung mit Profil, 12/2010.
Verfügbar unter: https://www.jetter.de/fileadmin/benutzerdaten/jetter-de/download/03_antriebe/motion_control/datenblatt/industrie_jetweb_mc_image_02_101201_d.pdf
Zugriff am: 17.04.2020

Johnson et al. 2015
Johnson, Matthew; Brown, Daniel S., 2015.
Evolving and controlling perimeter, rendezvous, and foraging behaviors in a computation-free robot swarm.
In: *Proceedings 9th European Alliance for Innovation (EAI) International Conference on Bio-Inspired Information and Communication Technologies,*
03-05 Dec 2015, New York City, USA

Kaiser et al. 2015
Kaiser, Benjamin; Keinert, Matthias; Lechler, Armin; Verl, Alexander, 2015.
CNC Tool Path Generation on Multi-Core Processors.
In: *WGP Congress 2015*, S. 339–346

Kang et al. 2000
Kang, Wei; Xi, Ning; Sparks, Andy, 2000.
Formation control of autonomous agents in 3D workspace.
In: *Proceedings of the 2000 IEEE International Conference on Robotics & Automation,*
April 2000, San Francisco, CA, S. 1755–1760

Keinert 2020
Keinert, Matthias, 2020.
Partitionierung von NC-Steuerungen zur Ausführung auf Multicore-Systemen, Diss., 2019.
Stuttgart: Fraunhofer Verlag.
ISW Forschung und Praxis - Berichte aus dem Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen der Universität Stuttgart 103.
ISBN 978-3-8396-1602-4

Keinert et al. 2014
Keinert, Matthias; Kaiser, Benjamin; Lechler, Armin; Verl, Alexander, 2014.
Analysis of CNC software modules regarding parallelization capability.
In: *Proceedings of the 24th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM),* San Antonio, Texas

Kircher 2011 Kircher, Christian, 2011. *Selbstadaptierende NC-Steuerung für rekonfigurierbare Werkzeugmaschinen*, Diss., 2011. Heimsheim: Jost-Jetter. ISW Forschung und Praxis - Berichte aus dem Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen der Universität Stuttgart 185. ISBN 39398907

Klasen et al. 2010 Klasen, Frithjof; Oestreich, Volker; Volz, Michael, 2010. *Industrielle Kommunikation mit Feldbus und Ethernet*. Berlin, Offenbach: VDE-Verlag. ISBN 3800732971

Kollegger et al. 2016 Kollegger, Gernot; Mayer, Horst, 2016. IEC 61499: Das Single Line Engineering. *computer automation* (06) Verfügbar unter: http://www.computer-automation.de/steuerungsebene/steuern-regeln/artikel/130428/

Koren 1980 Koren, Yoram, 1980. Cross-coupled biaxial computer control for manufacturing systems. *Journal of Dynamic Systems, Measurement, and Control* **102** (4), S. 265–272

Koren 2010 Koren, Yoram, 2010. *The global manufacturing revolution: Product-process-business integration and reconfigurable systems*: John Wiley & Sons Vol. 80. ISBN 0470920807

Koren et al. 1999 Koren, Yoram; Heisel, U; Jovane, F; Moriwaki, T; Pritschow, Günter; Ulsoy, G; van Brussel, H., 1999. Reconfigurable Manufacturing Systems. *CIRP Annals* **48** (2), S. 527–540

Krüger 2007 Krüger, Jörg, 2007. Nachhaltige Produktion durch flexible Automatisierung. *ZWF Zeitschrift für wirtschaftlichen Fabrikbetrieb* **102** (6), S. 332–334

Lehner 2004 Lehner, Wolf-Dieter, 2004. *Regelung von Vorschubachsen unter Verwendung der Relativbeschleunigung*, Diss. Heimsheim: Jost-Jetter. ISW Forschung und Praxis - Berichte aus dem Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen der Universität Stuttgart 150. ISBN 3936947457

Leitão 2009 Leitão, Paulo, 2009. Agent-based distributed manufacturing control: A state-of-the-art survey. *Engineering Applications of Artificial Intelligence* **22** (7), S. 979–991 DOI: 10.1016/j.engappai.2008.09.005

Lenkenhoff 2018 Lenkenhoff, Kay, 2018. *Methodenbasierte Modellierung einer cyber-physischen Informationsarchitektur: Ein Beitrag für das durchgängige Engineering*, Diss., 2018. Aachen: Shaker. ISBN 3844060720

Leonard et al. 2001 Leonard, Naomi Ehrich; Fiorelli, Edward, 2001. Virtual leaders, artificial potentials and coordinated control of groups. In: *Proceedings of the 40th IEEE Conference on Decision and Control,* Dezember 2001, Orlando, Florida, USA, S. 2968–2973

Lesi 2020 Lesi, Vuk, 2020. *Cyber-Physical Systems Lab: Reconfigurable Manufacturing Systems Testbeds*. Verfügbar unter: https://cpsl.pratt.duke.edu/lesi Zugriff am: 08.05.2020

Lesi et al. 2016 Lesi, Vuk; Jakovljevic, Zivana; Pajic, Miroslav, 2016. Towards plug-n-play numerical control for reconfigurable manufacturing systems. In: *21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA) 2016*, S. 1–8

Lesi et al. 2019 Lesi, Vuk; Jakovljevic, Zivana; Pajic, Miroslav, 2019.
Synchronization of Distributed Controllers in Cyber-Physical Systems.
In: *24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA),*
10.09.2019 - 14.09.2019, Saragozza, Spain, S. 710–717

Lesi et al. 2019 Lesi, Vuk; Jakovljevic, Zivana; Pajic, Miroslav, 2019.
Distributing Numerical Control for Reconfigurable Manufacturing Systems, Preprint.
Verfügbar unter:
https://cpsl.pratt.duke.edu/sites/cpsl.pratt.duke.edu/files/images/lesi_cnc19.pdf
Zugriff am: 22.11.2019

Lewis et al. 1997 Lewis, M. Anthony; Tan, Kar-Han, 1997. High precision formation control of mobile robots using virtual structures.
Autonomous Robots **4** (4), S. 387–403

Li et al. 2017 Li, Xiangfei; Zhao, Huan; Zhao, Xin; Ding, Han, 2017. Contouring compensation control based on high accuracy contour error estimation for multi-axis motion systems.
The International Journal of Advanced Manufacturing Technology **93** (5-8), S. 2263–2273
DOI: 10.1007/s00170-017-0689-z

Lian et al. 2002 Lian, Feng-Li; Moyne, James; Tilbury, Dawn, 2002. Network design consideration for distributed control systems.
IEEE Transactions on Control Systems Technology **10** (2), S. 297–307

Lim et al. 2003 Lim, M.K; Zhang, Z., 2003. A multi-agent based manufacturing control strategy for responsive manufacturing.
Journal of Materials Processing Technology **139** (1), S. 379–384
DOI: 10.1016/S0924-0136(03)00535-1

Lin et al. 2007 Lin, Jie; Morse, A. Stephen; Anderson, Brian D. O., 2007. The multi-agent rendezvous problem. Part 2: The asynchronous case. *SIAM Journal on Control and Optimization* **46** (6), S. 2120–2147 DOI: 10.1137/040620564

LinMot 2017 LinMot NTI AG, 2017. *LinMot LinUDP Library: DLL Integration Version 2.0.5*, 25.09.2017. Verfügbar unter: www.linmot.com Zugriff am: 18.01.2018

LinMot 2020 LinMot NTI AG, 2020. Industrielle Linearmotoren: Produktkatalog (24) Verfügbar unter: https://shop.linmot.com/data/import/Dokumente/0185-1017-D_1V0_DS_Drives_C1200.pdf

Lüth 1998 Lüth, Tim C., 1998. *Technische Multi-Agenten-Systeme: verteilte autonome Roboter- und Fertigungssysteme*. München, Wien: Carl Hanser Verlag. ISBN 3446194681

Lutz 2020 Lutz, Peter (Hrsg.), 2020. *Working Group FLC Motion*, Kickoff der Working Group. Webmeeting

McLain et al. 2000 McLain, Timothy W; Chandler, Phillip R; Pachter, Meir, 2000. A decomposition strategy for optimal coordination of unmanned air vehicles. In: *Proceedings of the American Control Conference,* Juni 2000, Chicago, Illinois, S. 369–373

Meister 2018 Meister, Eugen, 2018. *Bild der z-Platine "Effectuator"*, Kirchheim u.T.

Meling 2013 Meling, Fabian J., 2013. *Methodik für die Rekombination von Anlagentechnik*, Diss., 2012: Herbert Utz Verlag. Forschungsberichte IWB 279. ISBN 3831643196

Merdan et al. 2012 Merdan, M; Zoitl, A; Koppensteiner, G; Melik-Merkumians, M., 2012. Adaptive Produktionssysteme durch den Einsatz von autonomen Softwareagenten.
e & i Elektrotechnik und Informationstechnik **129** (1), S. 53–58
Verfügbar unter: https://link.springer.com/content/pdf/10.1007%2Fs00502-012-0071-7.pdf-
DOI: 10.1007/s00502-012-0071-7

Molano et al. 2018 Molano, Jacopo Cavalaglio Camargo; Lahrache, Achraf; Rubini, Riccardo; Cocconcelli, Marco, 2018. A new method for motion synchronization among multivendor's programmable controllers.
Measurement **126**, S. 202–214

Monostori et al. 2006 Monostori, L; Váncza, J; Kumara, S.R.T., 2006. Agent-Based Systems for Manufacturing.
CIRP Annals - Manufacturing Technology **55** (2), S. 697–720
DOI: 10.1016/j.cirp.2006.10.004

Monostori et al. 2016 Monostori, L; Kádár, B; Bauernhansl, T; Kondoh, S; Kumara, S; Reinhart, G; Sauer, O; Schuh, G; Sihn, W; Ueda, K., 2016. Cyber-physical systems in manufacturing **65** (2), S. 621–641
DOI: 10.1016/j.cirp.2016.06.005

Murray 2007 Murray, Richard M., 2007. Recent Research in Cooperative Control of Multivehicle Systems.
Journal of Dynamic Systems, Measurement, and Control **129** (5), S. 571–583
Verfügbar unter: http://dynamicsystems.asmedigitalcollection.asme.org/data/Journals/JDSMAA/26405/571_1.pdfDOI: 10.1115/1.2766721

Norm DIN 66025-1 DIN 66025-1:1983-01.
Programmaufbau für numerisch gesteuerte Arbeitsmaschinen: Allgemeines

Norm DIN EN 61131-3 DIN EN 61131-3:2014-06.
Speicherprogrammierbare Steuerungen - Teil 3: Programmiersprachen (IEC 61131-3:2013): Deutsche Fassung EN 61131-3:2013

Norm DIN EN 61499 DIN EN 61499:2014-09. *Funktionsbausteine für industrielle Leitsysteme - Teil 1: Architektur (IEC 61499-1:2012): Deutsche Fassung EN 61499-1:2013*

Norm DIN EN IEC 61158-1 DIN EN IEC 61158-1:2019-04. *Industrielle Kommunikationsnetze - Feldbusse - Teil 1: Überblick und Leitfaden zu den Normen der Reihen IEC 61158 und IEC 61784 (IEC 61158-1:2019): Deutsche Fassung EN IEC 61158-1:2019*

Norm EN IEC 61784-1 EN IEC 61784-1:2019-05. *Industrielle Kommunikationsnetze - Profile - Teil 1: Feldbusprofile (IEC 61784-1:2019).*

NXTcontrol 2018 NXTcontrol GmbH, 2018. *nxtSTUDIO: Das effizienteste Engineering für verteilte Systeme*. Verfügbar unter: https://www.nxtcontrol.com/engineering/ Zugriff am: 17.05.2020

Oberhaus 2018 Oberhaus, Daniel, 2018. ‘Master/Slave’ Terminology Was Removed from Python Programming Language. *VICE,* 13.09.2018 Verfügbar unter: https://www.vice.com/en_us/article/8x7akv/masterslave-terminology-was-removed-from-python-programming-language Zugriff am: 12.09.2020

Olfati-Saber et al. 2007 Olfati-Saber, Reza; Fax, J. Alex; Murray, Richard M., 2007. Consensus and Cooperation in Networked Multi-Agent Systems. *Proceedings of the IEEE* **95** (1), S. 215–233 DOI: 10.1109/JPROC.2006.887293

Otto et al. 2014 Otto, Jens; Henning, Steffen; Niggemann, Oliver, 2014. Why Cyber-physical Production Systems Need a Descriptive Engineering Approach – A Case Study in Plug & Produce. In: *2nd International Conference on System-Integrated Intelligence: Challenges for Product and Production Engineering*, S. 295–302

Özbek 2019 Özbek, Memo, 2019.
Was ist eine dApp (dezentralisierte Applikation)?, 06.03.2019.
Verfügbar unter: https://decentralbox.com/was-ist-eine-dapp/
Zugriff am: 17.05.2020

Pachow-Frauenhofer 2012 Pachow-Frauenhofer, Julia, 2012.
Planung veränderungsfähiger Montagesysteme, Diss.
Garbsen: PZH, Produktionstechn. Zentrum.
Berichte aus dem IFA 01/2012.
ISBN 3943104575

Pantförder et al. 2014 Pantförder, Dorothea; Mayer, Felix; Diedrich, Christian; Göhner, Peter; Weyrich, Michael; Vogel-Heuser, Birgit, 2014.
Agentenbasierte dynamische Rekonfiguration von vernetzen Produktionsanlagen - Evolution statt Revolution.
In: Bauernhansl, Thomas; Hompel, Michael ten; Vogel-Heuser, Birgit (Hrsg.)*: Industrie 4.0 in Produktion, Automatisierung und Logistik: Anwendung-Technologien-Migration*.
Wiesbaden: Springer Vieweg, S. 145–158
ISBN 3658046821

Papaioannou 1979 Papaioannou, Spiros G., 1979. Interpolation algorithms for numerical control.
Computers in industry **1** (1), S. 27–40
DOI: 10.1016/0166-3615(79)90006-X

Parunak 1994 Parunak, Van Dyke, 1994. Applications of distributed artificial intelligence in industry.
Foundations of distributed artificial intelligence **2**

Parunak 1997 Parunak, Van Dyke, 1997. Go to the ant: Engineering principles from natural multi-agent systems.
Annals of Operations Research **75**, S. 69–101

Pérez-Pinal et al. 2004 Pérez-Pinal, Francisco J; Nunez, Ciro; Alvarez, Ricardo; Cervantes, Ilse, 2004.
Comparison of multi-motor synchronization techniques.
In: *Transactions of the The 30th Annual Conference of the IEEE Industrial Electronics Society,,* November 2 - 6, Busan, Korea, S. 1670–1675

PLC Open 2011
PLC Open, 2011.
Function blocks for Motion Control: Part 1.
Verfügbar unter: https://plcopen.org/system/files/downloads/plcopen_motion_control_part_1_version_2.0.pdf
Zugriff am: 17.05.2020

Pritschow et al. 1990
Pritschow, Günter; Rudloff, H., 1990. Bahnregelung im Zustandsraum.
wt Werkstattstechnik **80**, S. 379–382

Pruschek 2009
Pruschek, Peter, 2009.
Verfahren zur anwendungsgerechten Parametrierung der Steuerung und Regelung von Vorschubachsen, Diss., 2008.
Heimsheim: Jost-Jetter.
ISW Forschung und Praxis - Berichte aus dem Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen der Universität Stuttgart 172.
ISBN 3939890413

Quapil 2020
Quapil, Tom, 2020.
Entwicklung und Validierung eines Netzwerk-Regelungskonzepts für ein dezentral gesteuertes Dreiachssystem.
Stuttgart, Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen, Masterarbeit, 2020

Regulin et al. 2017
Regulin, Daniel; Vogel-Heuser, Birgit, 2017.
Agentenorientierte Verknüpfung existierender heterogener automatisierter Produktionsanlagen durch mobile Roboter zu einem Industrie-4.0-System.
In: Vogel-Heuser, Birgit; Bauernhansl, Thomas; Hompel, Michael ten (Hrsg.)*: Handbuch Industrie 4.0 Bd. 2*.
2. Auflage.
Berlin, Heidelberg: Springer Vieweg, S. 93–118
ISBN 978-3-662-53247-8

Ren et al. 2007
Ren, Wei; Beard, Randal W; Atkins, Ella M., 2007. Information consensus in multivehicle cooperative control.
IEEE Control systems magazine **27** (2), S. 71–82

Ren et al. 2008 Ren, Wei; Beard, Randal W., 2008. *Distributed consensus in multi-vehicle cooperative control: Theory and Applications*. London: Springer. ISBN 978-1-84800-014-8 DOI: 10.1007/978-1-84800-015-5

Rinschede 1995 Rinschede, Matthias, 1995. *Koordination dezentraler Leitstände auf Basis objektorientierter Softwaretechnologie*, Diss., 1995. Sinzheim: Pro Universitate. Wissenschaftliche Schriften: Wirtschaftsinformatik. ISBN 3930747359

Russell Carpenter 2002 Russell Carpenter, J., 2002. Decentralized control of satellite formations. *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal* **12** (2-3), S. 141–161 DOI: 10.1002/rnc.680

Russell et al. 2016 Russell, Stuart J; Norvig, Peter, 2016. *Artificial intelligence: a modern approach*. Upper Saddle River, New Jersey: Pearson Education Limited. ISBN 978-0-13-604259-4

Sackenreuther 2019 Sackenreuther, Lisa, 2019. *Entwurf und Validierung eines Beobachtermodells für die dezentrale Interpolation*. Stuttgart, Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen, Bachelorarbeit, 2019

Sarachik et al. 1957 Sarachik, Philip; Ragazzini, John Ralph, 1957. A 2-dimensional feedback control system. *Transactions of the American Institute of Electrical Engineers, Part II: Applications and Industry* **76** (2), S. 55–61

Schneider et al. 2012 Schneider, Frank E; Wildermuth, Dennis; Wolf, Hans-Ludwig, 2012.
Motion Coordination in Formation of Multiple Mobile Robots using a Potential field approach.
In: Parker, Lynne E; Bekey, George; Barhen, Jacob (Hrsg.)*: Distributed Autonomous Robotic Systems*: Springer Science & Business Media, S. 306–313
ISBN 978-4-431-70295-5

Schöbel 2019 Schöbel, Daniel, 2019.
Entwicklung eines dezentralen Steuerungskonzepts für die verteilte Interpolation mit autonomen Agenten.
Stuttgart, Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen, Masterarbeit, 2019

Schröder 2007 Schröder, Thomas, 2007.
Entwicklung und Evaluation von Algorithmen zur zeitoptimierten Bewegungszerlegung bei kinematisch redundanten Werkzeugmaschinen, Diss., 2007
Verfügbar unter: http://archiv.tu-chemnitz.de/pub/2007/0130/

Seitz 2015 Seitz, Matthias, 2015.
Speicherprogrammierbare Steuerungen für die Fabrik-und Prozessautomation: Strukturierte und objektorientierte SPS-Programmierung, Motion Control, Sicherheit, vertikale Integration.
4. überarbeitete und erweiterte Auflage.
München: Carl Hanser Verlag GmbH Co KG.
ISBN 978-3-446-44273-3

Sercos News 2017 Sercos News, 2017. Antriebstechnik der Zukunft: Schaltschranklos, modular und flexibel: Schaltschranklose Antriebstechnik steigert die Produktivität im kompletten Wertstrom.
Sercos News - the automation bus magazine (02/2017)
Verfügbar unter: http://www.sercos.de/downloads/sercos-news/

SEW-Eurodrive 2018 SEW-Eurodrive GmbH & Co KG, 2018. *Controller hardware for decentralized installations*. Verfügbar unter: https://www.sew-eurodrive.de/products/control_technology/controller_hardware/decentralized_controllers/decentralized_controllers.html Zugriff am: 26.06.2018

Shen et al. 2003 Shen, Weiming; Norrie, Douglas H; Barthès, Jean-Paul, 2003. *Multi-agent systems for concurrent intelligent design and manufacturing*: CRC press. ISBN 0203305604

Siemens 2013 Siemens AG, 2013. *SIMATIC: S7-1500 Motion Control*, 10/2013. Verfügbar unter: https://www.automation.siemens.com/salesmaterial-as/interactive-manuals/getting-started_simatic-s7-1500/documents/DE/smc_de.pdf Zugriff am: 07.07.2020

Srinivasan et al. 1990 Srinivasan, Kulkarni; Kulkarni, P. K., 1990. Cross-coupled control of biaxial feed drive servomechanisms. *Journal of Dynamic Systems, Measurement, and Control* **112** (2), S. 225–232 DOI: 10.1115/1.2896129

Stanglmeier 2020 Stanglmeier, Pirmin, 2020. *Anwendungsfallstudie zur dezentralen Interpolation: Integration von kommerziellen Antriebskomponenten und prototypischen DEVEKOS-Achssteuerungskomponenten*. Stuttgart, Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Studienarbeit, 2020

Stehle et al. 2017 Stehle, Thomas; Heisel, Uwe, 2017. Konfiguration und Rekonfiguration von Produktionssystemen. In: *Neue Entwicklungen in der Unternehmensorganisation*: Springer, S. 333–367 ISBN 978-3-662-55426-5

Steinegger et al. 2016 Steinegger, Michael; Plaschka, Nikolaus; Melik-Merkumians, Martin; Schitter, Georg, 2016. A framework for modular and distributable control of reconfigurable robotic systems. In: *IEEE International Conference on Industrial Technology (ICIT)*, S. 848–853

Struck 2017 Struck, Angela, 2017. "Motion, die mehr kann und weniger kostet". *Mechatronik & Engineering* (2)

Sun et al. 2007 Sun, Dong; Shao, Xiaoyin; Feng, Gang, 2007. A model-free cross-coupled control for position synchronization of multi-axis motions: theory and experiments. *IEEE Transactions on Control Systems Technology* **15** (2), S. 306–314 DOI: 10.1109/TCST.2006.883201

Tan et al. 2007 Tan, Kok Kiong; Lee, Tong Heng; Huang, Sunan, 2007. *Precision motion control: design and implementation*. London: Springer Verlag. ISBN 978-1-84800-020-9

Trentesaux 2009 Trentesaux, Damien, 2009. Distributed control of production systems. *Engineering Applications of Artificial Intelligence* **22** (7), S. 971–978 DOI: 10.1016/j.engappai.2009.05.001

van der Wal 2001 van der Wal, Eelco, 2001. Creating Reusable, Hardware Independent Motion Control Applications via IEC 61131-3 and PLCopen Function Blocks for Motion Control. In: *Proceedings of the IEEE International Conference on Advanced Intelligent Mechatronics (ASME) 2001,* Como, Italy, S. 769–774

Wei et al. 2010 Wei, Dong; Ji, Kun, 2010. Resilient industrial control system (RICS), 2010 3rd International Symposium on Resilient Control Systems. In: *3rd International Symposium on Resilient Control Systems 2010*

Xiao et al. 2006 Xiao, Y; Zhu, K. Y., 2006. Optimal synchronization control of high-precision motion systems. *IEEE Transactions on Industrial Electronics* **53** (4), S. 1160–1169 DOI: 10.1109/TIE.2006.878317

Xu et al. 2011 Xu, Xiong; Sheng, Xinjun; Xiong, Zhenhua; Zhu, Xiangyang, 2011. Time-stamped cross-coupled control in networked CNC systems. In: *2011 IEEE International Conference on Robotics and Automation,* 09.05. - 13.05.2011, Shanghai, China, S. 4378–4383

Xu et al. 2017 Xu, Xiong; Gu, Guo-Ying; Xiong, Zhenhua; Sheng, Xinjun; Zhu, Xiangyang, 2017. Development of a decentralized multi-axis synchronous control approach for real-time networks. *ISA transactions* **68**, S. 116–126 DOI: 10.1016/j.isatra.2017.03.012

Yao 2015 Yao, Wu-Sung, 2015. Modeling and Synchronous Control of Dual Mechanically Coupled Linear Servo System. *Journal of Dynamic Systems, Measurement, and Control* **137** (4), S. 41009 Verfügbar unter: https://dynamicsystems.asmedigitalcollection.asme.org/data/journals/jdsmaa/931137/ds_137_04_041009.pdfDOI: 10.1115/1.4028688

Yook et al. 2002 Yook, John K; Tilbury, Dawn M; Soparkar, Nandit R., 2002. Trading computation for bandwidth: Reducing communication in distributed control systems using state estimators. *IEEE Transactions on Control Systems Technology* **10** (4), S. 503–518

Yoong et al. 2014 Yoong, Li Hsien; Roop, Partha S; Bhatti, Zeeshan E; Kuo, Matthew M. Y., 2014. *Model-driven design using IEC 61499: A synchronous approach for embedded and automation systems*. Cham Heidelberg New York Dordrecht London: Springer. ISBN 978-3-319-10520-8 DOI: 10.1007/978-3-319-10521-5

Zhang et al. 2013 Zhang, Li-Bing; You, You-Peng; Yang, Xue-Feng, 2013. A control strategy with motion smoothness and machining precision for multi-axis coordinated motion CNC machine tools. *The International Journal of Advanced Manufacturing Technology* **64** (1-4), S. 335–348 DOI: 10.1007/s00170-012-4019-1

Zimmermann et al. 2019 Zimmermann, Patrick; Axmann, Etienne; Brandenbourger, Benjamin; Dorofeev, Kirill; Mankowski, André; Zanini, Paulo, 2019. Skill-based Engineering and Control on Field-Device-Level with OPC UA.
Whitepaper der VDMA Fachabteilung Integrated Assembly Solutions
Verfügbar unter: https://rua.vdma.org/documents/106005/45540632/Paper+Skill-based+Engineering+and+Control+%282019%29.pdf/cd948e58-8ccf-d2f4-3920-3436987bfe4b

zub 2018 zub machine control AG, 2018.
Antriebe positionieren und synchronisieren, 11/2018.
Verfügbar unter:
https://www.zub.li/zub/pdf/03_ZUB_Imagebrosch_DE.pdf
Zugriff am: 07.07.2020

Anhang

Die approximierte gemittelte Geschwindigkeit

Die approximierte gemittelte Geschwindigkeit zur vereinfachten Berechnung des approximierten Zeitbedarfs T_n^* nach

$$T_n^* = \frac{r_{n_i} - r_{n_{i-1}}}{v_n^*}$$

wird unter der Annahme $v_n^* = 0.8 \cdot v_{n,\max}$ berechnet. Dadurch ist gegeben, dass der approximierte Zeitbedarf T_n^* mit einem Korrekturfaktor K beaufschlagt werden muss um den realen Zeitbedarf T zu ermitteln:

$$T_n^* + K = T.$$

Für das Beispiel des Trapezprofils unter der Annahme von Stillstand zu Beginn und Ende des Segments wird $K = 0$ genau dann, wenn gilt

$$T_{\mathrm{I}} = T_{\mathrm{III}} = \frac{T_{\mathrm{II}}}{3}.$$

In allen anderen Fällen folgt

$$T_{\mathrm{I}} = T_{\mathrm{III}} > \frac{T_{\mathrm{II}}}{3} : K > 0,$$

$$T_{\mathrm{I}} = T_{\mathrm{III}} < \frac{T_{\mathrm{II}}}{3} : K < 0.$$

Herleitung

Dieses Verhältnis kann durch die Flächenbetrachtung des Trapezprofil in Abbildung 5-7 nachvollzogen werden. Es gilt die Fläche unter der Geschwindigkeitskurve ist die zurückgelegte Distanz. Unabhängig ob das Beschleunigungsverhalten als lineare Beschleunigung oder als S-Kurve definiert wird gilt:

$$l_{T_{\mathrm{II}}}(\mathrm{t}) = v_{\max} t$$

$$l_{T_{\mathrm{I}},T_{\mathrm{III}}}(\mathrm{t}) = \frac{1}{2} a_{\max} t^2 = \frac{1}{2} \frac{v_{\max}}{t} t^2 = \frac{1}{2} v_{\max} t$$

Folglich wird also in den Beschleunigungs- und Verzögerungsphasen $T_{\mathrm{I}}, T_{\mathrm{III}}$ des Segments pro Zeiteinheit nur die Hälfte einer Wegeinheit in der Geschwindigkeitsphase T_{II} zurückgelegt.

Für ein Segment mit Zeitbedarf $T = T_n^* = 1$ wird bei Verwendung der approximierten Geschwindigkeit eine Weglänge

$$L = 0{,}8\, v_{\max} \cdot 1 = 2 l_{T_{\mathrm{I}},T_{\mathrm{III}}}(\mathrm{t_b}) + l_{T_{\mathrm{II}}}(\mathrm{t_a})$$

$$0{,}8 = t_b + t_a$$

angenommen.

Es muss gleichzeitig der Zeitbedarf $T_n^* = 1 = 2t_b + t_a$ gelten, weshalb folgt

$$t_a = 0{,}6$$

$$t_b = 0{,}2 = \frac{t_a}{3}.$$

Sollte nicht $K = 0$ gelten, treten folgende Approximationsfehler auf:

$$T = 1 = T_n^* + K$$

$$T_n^* = 2t_b + t_a - K$$

$$L = T_n^* 0{,}8\, v_{\max}$$

daraus folgt

$$K = 0{,}48t_b - 0{,}16t_a.$$

Für Segmente, in denen die zeitlichen Anteile der Beschleunigung und Verzögerung t_b unter dem Faktor $t_b = \frac{t_a}{3}$ des zeitlichen Anteils der Geschwindigkeit t_a liegen, wird damit approximierte Zeitbedarf T_n^* überschätzt

$$K < 0;\ T_n^* > T.$$

Dies gilt ebenfalls, wenn statt der Annahme von Stillstand zu Beginn $v_{\text{Start}} = 0$ und Ende des Segments $v_{\text{Ziel}} = 0$ sichergestellt werden kann

$$|v_{\max} - v_{\text{Start}}| < |v_{\max}|$$

$$|v_{\max} - v_{\text{Ziel}}| < |v_{\max}|.$$

Im anderen Fall wird der approximierte Zeitbedarf T_n^* unterschätzt

$$K > 0;\ T_n^* < T.$$

In diesem Fall wird die approximierte Zeit nicht ausreichen um das Segment zum Zielzeitpunkt zu beenden. Stattdessen wird der Segmentregler nach Überschreiten der Toleranz aus (5-43) erneut den Zeitbedarf T_n^* berechnen. Da die Dynamik der Achse über die Annahme $v_n^* = 0.8 \cdot v_{n,\max}$ allerdings weiterhin unterschätzt wird, wird die neu berechnete Koordinationsvariable allerdings wiederholt nicht durchsetzbar sein. Im gegensätzlichen Fall wird der Segmentregler über die Vorsteuerung der approximierten mittleren Geschwindigkeit die Einhaltung der Zielzeit erreichen können. Eine vorhandene Start- oder Zielgeschwindigkeit $v_{\text{Start/Ziel}}$ kann diesen Effekt teilweise ausgleichen, da in der Folge die Anteile der Zeitbedarfe t_b weiter reduziert werden, davon kann jedoch im Allgemeinen nicht ausgegangen werden.

Lemma:

Die Annahme $v_n^* = 0.8 \cdot v_{n,\max}$ ist nur für Segmente gültig, in denen die Beschleunigungs- und Verzögerungsphasen jeweils maximal ein Drittel der Phase mit gleichbleibender Geschwindigkeit ausmachen.

Bei der Betrachtung von Beschleunigungsprofilen mit zusätzlichen Phasen (Siebenphasenprofil) muss das Verhältnis der Beschleunigungs- und Verzögerungsphasen zur Phase mit gleichbleibender Geschwindigkeit gesondert betrachtet

werden und das Grenzverhältnis bestimmt werden, ab dem eine Zeitüberschätzung eintritt.

In allen anderen Fällen wird die korrekte Berechnung des benötigten Zeitbedarfs empfohlen.

Nebenbemerkung

Es können in der Durchführung der Bewegungen speziell zum Zeitpunkt der Beschleunigung und Verzögerung durch unterschiedliche Dynamiken in den Achsen unterschiedliche Anfahrbewegungen zu einem Schleppabstand bzw. einer Bahnabweichung führen (Binder 1979, S. 40), jedoch wird angenommen, dass die Zeiten, in denen mit konstanter Geschwindigkeit verfahren wird einen weit größeren Einfluss auf die synchronisierte Bewegung hat als die Beschleunigungs- und Verzögerungsphasen.

Alternativ kann auch ein gemeinsames, identisches Beschleunigungsprofil für alle beteiligten Achsen fest definiert werden. Um volle Synchronität zu gewährleisten muss dann über das Rendez-Vous-Verfahren nicht nur die Zeitspanne für die Bewegung über das Segment T_S, sondern ebenfalls die Unterteilung dieser Zeitspanne in die R Phasen $T_{\mathrm{I}}, T_{\mathrm{II}}, T_{\mathrm{III}}, \ldots T_R$ unter der Beziehung

$$T_S = \sum_{P=I \ldots R} T_P$$

in den einzelnen Zeitabschnitten orientiert an der Achse mit dem höchsten Zeitbedarf verhandelt werden.

Publikationsliste

Dripke et al. 2017
Dripke, Caren; Verl, Alexander, 2017.
Challenges in distributed interpolation with multi-components systems in future-oriented manufacturing units.
In: *Proceedings of the 47th International Conference on Computers and Industrial Engineering (CIE) 2017,*
11.10.2017 - 13.10.2017, Lissabon, Portugal

Dripke et al. 2018a
Dripke, Caren; Schneider, Ben; Dragan, Mihai; Zoitl, Alois; Verl, Alexander, 2018.
Concept of Distributed Interpolation for Skill-Based Manufacturing with Real-Time Communication.
In: *Tagungsband des 3. Kongresses Montage Handhabung Industrieroboter,*
27.02.2018 - 28.02.2018, Erlangen, Deutschland, S. 215–222

Dripke et al. 2018b
Dripke, Caren; Laicher, Sonja; Verl, Alexander, 2018.
Axis-in-the-loop simulation: a concept to transfer verification and validation of distributed interpolation algorithms from simulation to interaction with commercial motion control hardware.
In: *Proceedings of the 48th International Conference on Computers and Industrial Engineering (CIE) 2018,*
02.12.2018 - 06.12.2018, Auckland, Neuseeland

Dripke et al. 2019a
Dripke, Caren; Ye, Zhongyi; Verl, Alexander, 2019.
Synchronization of a distributed interpolation application via cross-coupled control.
In: *IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC) 2019,*
17.06. - 20.06.2019, Sophia Antipolis, Frankreich

Dripke et al. 2019b Dripke, Caren; Sun, Yuesheng; Verl, Alexander, 2019.
Framework for the Closed-Form Calculation of Forward and Inverse Kinematics for Basic Kinematics in Reconfigurable Multi-Component Systems.
In: *24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA),*
10.09.2019 - 14.09.2019, Saragozza, Spain

Dripke et al. 2020 Dripke, Caren; Schöbel, Daniel; Verl, Alexander, 2020.
Distributed Interpolation: Synchronization of motion-controlled axes with coordination vector and decentralized segment controllers.
In: *IEEE Advanced Motion Control Worshop (AMC) 2020,*
14.09. 2020 - 16.09.2020, Kristiansand, Norwegen

Betreute studentische Arbeiten

Awad 2018

Awad, Joseph, 2018.
Implementation Study on Distributed Interpolation Algorithms for Smoothing and Lookahead in Multi-component Systems.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Bachelor Thesis. Betreut durch: Verl, A. und Dripke, C., 2018

Bader 2020

Bader, Joshua, 2020.
Studie zur Lageabweichung bei dezentraler Ansteuerung in interpolierenden Systemen unter Berücksichtigung verschiedener Beschleunigungsprofile.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Studienarbeit. Betreut durch: Verl, A. und Dripke, C., 2020

Lu 2018

Lu, Ziniu, 2018.
Entwurf und Validierung eines Kommunikationskonzepts für die dezentrale Interpolation.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Studienarbeit. Betreut durch: Verl, A., Dripke, C., und Dragan, M., 2018

Quapil 2020

Quapil, Tom, 2020.
Entwicklung und Validierung eines Netzwerk-Regelungskonzepts für ein dezentral gesteuertes Dreiachssystem.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen, Masterarbeit. Betreut durch: Verl, A. und Dripke, C., 2020

Sackenreuther 2019
Sackenreuther, Lisa, 2019.
Entwurf und Validierung eines Beobachtermodells für die dezentrale Interpolation.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen, Bachelorarbeit. Betreut durch: Verl, A. und Dripke, C., 2019

Schelbert 2018
Schelbert, Lukas, 2018.
Studie zu Konzepten zur Rückstellung und Wiederaufnahme der Arbeit nach Not-Halts und deren Übertragbarkeit auf dezentral gesteuerte Systeme.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Studienarbeit. Betreut durch: Verl, A. und Dripke, C., 2018

Schöbel 2019
Schöbel, Daniel, 2019.
Entwicklung eines dezentralen Steuerungskonzepts für die verteilte Interpolation mit autonomen Agenten.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen, Masterarbeit. Betreut durch: Verl, A. und Dripke, C., 2019

Stanglmeier 2020
Stanglmeier, Pirmin, 2020.
Anwendungsfallstudie zur dezentralen Interpolation: Integration von kommerziellen Antriebskomponenten und prototypischen DEVEKOS-Achssteuerungskomponenten.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Studienarbeit. Betreut durch: Verl, A. und Dripke, C., 2020

Sun 2018

Sun, Yuesheng, 2018.
Reconfigurable kinematics – a framework to automatically generate forward- and backward transformations for basic kinematic structures.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Student Research Project. Betreut durch: Verl, A. und Dripke, C., 2018

Vetter 2019

Vetter, Thorsten, 2019.
Anwendbarkeit von Collective Decision-Modellen aus der Schwarmrobotik auf verteilte Interpolation von Mehrkomponentensystemen.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Bachelorarbeit. Betreut durch: Verl, A. und Dripke, C., 2019

Villar Rio 2017

Villar Rio, Laura, 2017.
Design and Validation of a model-based development process for distributed control algorithms with different target platforms.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Master Thesis. Betreut durch: Verl, A. und Dripke, C., 2017

Warag 2019

Warag, Mouldi, 2019.
Integration von industriellen Achsen an einem Mehrachsdemonstrator zur verteilten Interpolation.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Studienarbeit. Betreut durch: Verl, A. und Dripke, C., 2019

Ye 2018

Ye, Zhongyi, 2018.
Coupling of multiple Simulink real-time-simulations: Interface for communication for distributed interpolation with industrial linear axes.
Stuttgart.
Univ., Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Student Research Project. Betreut durch: Verl, A. und Dripke, C., 2018

Lebenslauf

Persönliches

Name	Caren Dripke
Geburtstag	13.10.1990
Geburtsort	Leonberg
Eltern	Thomas und Martina Dripke

Bildungsweg

1997 – 2001	Grundschule, Leonberg
2001 – 2009	Gymnasium, Leonberg

Studium

2009 – 2012	Technische Kybernetik, B.Sc., Universität Stuttgart
2012 – 2015	Mechatronik, M.Sc., Universität Stuttgart

Berufstätigkeit

2015 – 2017	Wissenschaftliche Mitarbeiterin, Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Universität Stuttgart
2017 – 2020	Gruppenleiterin „Industrielle Steuerungstechnik Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), Universität Stuttgart
2021 – heute	Abteilungsleitung Entwicklung Robotik Lorch Schweißtechnik GmbH, Auenwald

Universität Stuttgart

Integrierte Miniatursteuerungen sind Technologiebefähiger für Industrie 4.0. Die Verwendung solcher Steuerungen in Automatisierungskomponenten ermöglicht es, diese als intelligente Elemente in Fertigungsstationen und -zellen einzusetzen. Solche intelligenten Komponenten bieten ihre Funktionen über das Produktionsnetzwerk an und können, kombiniert mit weiteren Automatisierungskomponenten, auch höherwertige Funktionen realisieren.

Ein Spezialfall der dezentralen Ansteuerung mit solchen höherwertigen Funktionen ist die gemeinsame synchronisierte Bewegung von positionierenden Automatisierungskomponenten. In klassischen Automatisierungssystemen wird diese Aufgabe meist von einer zentralen Bewegungssteuerung berechnet, koordiniert und durchgeführt. In dieser Arbeit wird betrachtet, welche alternativen Systemarchitekturen zur Interaktion zwischen den Miniatursteuerungen geeignet sind, um eine verteilte Bewegungssteuerung umzusetzen. Die agentenbasierte Systemarchitektur wird in dieser Arbeit von anderen Ansätzen abgegrenzt, daraus folgende Anforderungen betrachtet und geeignetes Agentenverhalten abgeleitet.

ISBN 978-3-8396-1810-3

Fraunhofer VERLAG