Arne Thorsten Rüting, M.Sc.

Echtzeitfähige modellprädiktive Planung zeitoptimierter Trajektorien für kinematisch redundante Mechanismen auf industrieller Hardware

Realtime-capable model-predictive planning of time-optimized trajectories for kinematically redundant mechanism on industrial hardware

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

Als elektronische Version frei verfügbar über die Digitalen Sammlungen der Universitätsbibliothek Paderborn.

Satz und Gestaltung: Arne Thorsten Rüting

Echtzeitfähige modellprädiktive Planung zeitoptimierter Trajektorien für kinematisch redundante Mechanismen auf industrieller Hardware

zur Erlangung des akademischen Grades eines
DOKTORS DER INGENIEURWISSENSCHAFTEN (Dr.-Ing.)
der Fakultät Maschinenbau
der Universität Paderborn

genehmigte
DISSERTATION

von
Arne Thorsten Rüting, M.Sc.
aus Lübbecke

Tag des Kolloquiums: 17. Oktober 2024
Referent: Prof. Dr.-Ing. habil. Ansgar Trächtler
Korreferent: Univ.-Prof. Dr.-Ing. Prof. h.c. Dr. h.c. Torsten Bertram

Vorwort

Die vorliegende Dissertation entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter in der Abteilung Scientific Automation des Fraunhofer-Instituts für Entwurfstechnik Mechatronik IEM in Paderborn.

An dieser Stelle möchte ich mich herzlich bei Herrn Prof. Dr.-Ing. habil. Ansgar Trächtler, Direktor des Fraunhofer IEM sowie Inhaber des Lehrstuhls Regelungstechnik und Mechatronik am Heinz Nixdorf Institut der Universität Paderborn, für die Betreuung dieser Dissertation bedanken. Sein fachlicher Rat, die geführten Diskussionen und seine konstruktiven und kritischen Hinweise haben entscheidend zum Gelingen dieser Arbeit beigetragen.

Ebenfalls zu Dank verpflichtet bin ich Herrn Univ.-Prof. Dr.-Ing. Prof. h.c. Dr. h.c. Torsten Bertram, Inhaber des Lehrstuhls für Regelungssystemtechnik der Fakultät für Elektrotechnik und Informationstechnik am Institut für Roboterforschung der technischen Universität Dortmund, für die Übernahme des Korreferats.

Meinen ehemaligen Kolleginnen und Kollegen des Fraunhofer IEM danke ich herzlich für das in allen Jahren tolle Arbeitsklima, die immer gute Zusammenarbeit inklusive der vielen konstruktiven Diskussionen und Anregungen sowie für die sorgfältige Durchsicht dieser Dissertation.
Den Studierenden, allen voran Herrn Eduard Block und Herrn Eduard Wilhelm, die mich im Rahmen ihrer studentischen Arbeiten oder ihrer Tätigkeit als studentische Hilfskraft während meiner Zeit beim IEM unterstützt haben, möchte ich ebenfalls meinen Dank aussprechen.

Ein ganz besonderer Dank gilt meiner Familie, meinen Eltern Birgit und Rolf für ihre Unterstützung und den Rückhalt auf meinem bisherigen Lebensweg und meiner Frau Melanie, meiner Tochter Lara und meinem Sohn Mattis für ihr Verständnis, den geschaffenen Freiraum und die mir immer wieder gegebene Motivation. Diese Unterstützung hat maßgeblich zum Gelingen der Dissertation beigetragen.

Lübbecke, im Oktober 2024 Arne Thorsten Rüting, M.Sc.

Zusammenfassung

Die vorliegende Arbeit befasst sich mit der modellprädiktiven Planung zeitoptimierter Trajektorien für mehrachsige kinematisch redundante Mechanismen. Es werden zunächst grundlegende Begriffe erläutert, bevor der Stand der Wissenschaft und Technik hinsichtlich der Trajektorienplanung von kinematisch redundanten und nicht-redundanten Mechanismen sowie der Ausführung von Optimierungen auf Industriesteuerungen vorgestellt wird. Im Anschluss wird die modellprädiktive Planung erläutert. Diese nutzt die Mehrdeutigkeit bei der Ermittlung der Gelenkpositionen kinematisch redundanter Mechanismen, um die Verfahrzeit zwischen Sollpositionen im Rahmen von Punkt-zu-Punkt-Bewegungen zu minimieren. Anhand eines kinematisch redundanten hybridkinematischen Mechanismus erfolgt zunächst eine simulative Validierung des Ansatzes. Bezogen auf die immer flexibler werdenden Fertigungen im Kontext Losgröße-1 sind jedoch echtzeitfähige selbstoptimierende Steuerungen erforderlich, die sich zur Laufzeit auf wechselnde Aufgaben einstellen und mit weiteren Systemen interagieren. Aus diesem Grund liegt der Fokus auf der echtzeitfähigen Ausführung des Ansatzes auf einer Industriesteuerung, ein entscheidender Punkt, um die erforderliche Flexibilität und industrielle Einsetzbarkeit zu erreichen und gleichzeitig ein signifikanter Unterschied zum Stand der Technik. Hierzu erfolgt die Umsetzung auf das reale System und der Nachweis der echtzeitfähigen Ausführung mit Optimierungszeiten kleiner einer Millisekunde.

Abstract

This thesis deals with the model-predictive planning of time-optimized trajectories for kinematically redundant multi-axes mechanisms. First, basic terms are explained before the current state-of-science and technology with regard to path planning for kinematically redundant and non-redundant mechanisms as well as the execution of optimizations on industrial controllers is presented. Next, the model-predictive planning is explained. Hereby, the ambiguity when calculating the joint positions of a kinematically redundant mechanism is used to minimize the movement time between set points with regard to point-to-point movements. Using a kinematically redundant hybrid mechanism, the approach is first validated with simulations. With regard to the increasing demands of more flexible manufacturing in context of batch size-1, real-time capable self-optimizing controls are necessary. Such controls adjust themselves to changing tasks during runtime and interact with other systems. Therefore, the focus is on the real-time execution of the approach on industrial controllers. This is a crucial point to achieve the required flexibility and industrial applicability, as well as to differ significantly from the state-of-the-art. For this purpose, the approach is implemented on the real system and the ability of a real-time execution with optimization times less than one millisecond is proven.

Vorveröffentlichungen

[SRHT21] Schütz, S.; Rüting, A.; Henke, C.; Trächtler, A.: Echtzeitfähige Planung optimierter Trajektorien für sensorgeführte, kinematisch redundante Mechanismen auf einer Industriesteuerung. *at - Automatisierungstechnik* 69 (2021), Nr. 3, S. 231–241

[SRHT20] Schütz, S.; Rüting, A.; Henke, C.; Trächtler, A.: Echtzeitfähige Planung optimierter Trajektorien für sensorgeführte, kinematisch redundante Mechanismen auf einer Industriesteuerung. *EKA 2020 - Entwurf komplexer Automatisierungssysteme*. Hrsg. von Jumar, U.; Diedrich, C. 2020

[RHT19] Rüting, A.; Henke, C.; Trächtler, A.: Umsetzung einer echtzeitfähigen modellprädiktiven Trajektorienplanung für eine mehrachsige Hybridkinematik auf einer Industriesteuerung. *at - Automatisierungstechnik* 67 (2019), Nr. 4, S. 326–336

[SRHT19] Schütz, S.; Rüting, A.; Henke, C.; Trächtler, A.: Regelung kollaborativer Robotersysteme zur benutzerfreundlichen, flexiblen Fertigung kleiner Losgrößen am Beispiel eines halbautomatischen Schweißvorgangs. *Fachtagung Mechatronik*. 2019

[RHT18] Rüting, A.; Henke, C.; Trächtler, A.: Umsetzung einer echtzeitfähigen Mehrgrößenoptimierung auf einer Industriesteuerung. *Entwurf komplexer Automatisierungssysteme*. Hrsg. von Jumar, U.; Diedrich, C. 2018

[RBT17] Rüting, A.; Block, E.; Trächtler, A.: Modellprädiktive Vorsteuerung für einen kinematisch redundanten hybridkinematischen Mechanismus im Industrieumfeld. *Fachtagung Mechatronik 2017*. Hrsg. von Bertram, T.; Corves, B.; Janschek, K. 2017

[RBT16] Rüting, A.; Blumenthal, L.; Trächtler, A.: Model Predictive Feedforward Compensation for Control of Multi Axes Hybrid Kinematics on PLC. *IECON*. 2016, S. 583–588

[RHWT15] Rüting, A.; Henke, C.; Warkentin, A.; Trächtler, A.: Entwurf eines Tripodbasierten Inspektionssystems - Vom virtuellen Prototypen zum Vorseriensystem. *10. Paderborner Workshop Entwurf Mechatronischer Systeme*. Bd. 343. 2015, S. 113–126

Echtzeitfähige modellprädiktive Planung zeitoptimierter Trajektorien für kinematisch redundante Mechanismen auf industrieller Hardware

Inhaltsverzeichnis

1 Einleitung **1**
- 1.1 Motivation und Zielsetzung 2
- 1.2 Definition der Anforderungen 5
- 1.3 Vorgehensweise und Aufbau der Arbeit 6

2 Grundlagen **9**
- 2.1 Notation 9
- 2.2 Begriffsdefinitionen 10

3 Stand der Wissenschaft und Technik **23**
- 3.1 Optimierungsbasierte Trajektorienplanung für seriell-, parallel- und hybridkinematische kinematisch redundante Mechanismen 23
- 3.2 Optimierungsstrategien für das Zeitverhalten nicht-redundanter Mechanismen 32
- 3.3 Modellbasierte Optimierungs- und Regelungsansätze auf industriell einsetzbarer Hardware 37
- 3.4 Zusammenfassung und Fazit 45

4 Echtzeitfähige modellprädiktive Planung zeitoptimierter Trajektorien . **47**
- 4.1 Grundidee 47
- 4.2 Gütefunktion und Optimierungsproblem 50
- 4.3 Herleitung des Zeitmodells für Punkt-zu-Punkt-Bewegungen 54
- 4.4 Nutzung des Potentials moderner Antriebshardware 57
- 4.5 Referenzverfahren 58

5 Anwendungsszenario 1: Der IEM-Mechanismus **61**
- 5.1 Adaption des Ansatzes 61
- 5.2 Herleitung der Gleichungen der kinematischen Probleme 64
  - 5.2.1 Inverses kinematisches Problem des Tricept 64
  - 5.2.2 Inverses und direktes kinematisches Problem des Portals . . . 78
  - 5.2.3 Simulative Validierung des inversen kinematischen Problems mittels Mehrkörpersimulation 80
- 5.3 Validierung des Zeitmodells für Punkt-zu-Punkt-Bewegungen 84
- 5.4 Mögliche Anwendungsszenarien für den IEM-Mechanismus 93

6 Simulative Umsetzung in *MATLAB* 95
6.1 Wahl der Sollpositionsfolgen 95
6.2 Simulative Ausführung des Referenzverfahrens 96
6.3 Funktionsvalidierung des modellprädiktiven Ansatzes in *MATLAB* unter Nutzung *MATLAB*-eigener Optimierer 98
6.4 Funktionsvalidierung des modellprädiktiven Ansatzes in *MATLAB* unter Nutzung freier Optimierer 101
6.4.1 Analyse verschiedener Optimierungsalgorithmen und ihrer Bibliotheken 101
6.4.2 Ergebnisse der Mehrzieloptimierung mit ausgewähltem Optimierungsalgorithmus 109

7 Umsetzung am realen System 115
7.1 Umsetzung der modellprädiktiven Planung auf SPS und eingebetteter Hardware 115
7.2 Umsetzung der modellprädiktiven Planung auf einer SPS 122

8 Anwendungsszenario 2: HKM bestehend aus Raumportal und Sechsachs-Knickarmroboter 129
8.1 Adaption des Ansatzes 129
8.2 Herleitung der Gleichungen der kinematischen Probleme 132
8.2.1 Inverses kinematisches Problem des Knickarmroboters 132
8.2.2 Inverses und direktes kinematisches Problem des Portals . . . 138
8.2.3 Simulative Validierung des inversen kinematischen Problems mittels Mehrkörpersimulation 138
8.3 Simulative Umsetzung des modellprädiktiven Ansatzes 139

9 Zusammenfassung und Ausblick 143

Literaturverzeichnis 147

Literaturverzeichnis der studentischen Arbeiten 165

Anhang

A1 Datenblätter 169
A1.1 Parallele Antriebe $M_1 \dots M_3$ des Tricept 169
A1.2 Serieller Antrieb M_4 des Tricept 172
A1.3 Antriebe M_5 und M_6 des Portals 174
A1.4 Getriebe der Antriebe M_4, M_5 und M_6 177
A1.5 Universal Robots UR10 180
A1.6 Raspberry Pi 4B 181
A1.7 SPS B&R X20CP1382-RT 184
A1.8 SPS Beckhoff C6030-0060 189

Abkürzungsverzeichnis

B&R	*Bernecker & Rainer*
DHK	*Denavit-Hartenberg*-Konvention
DHP	*Denavit-Hartenberg*-Parameter
DKP	direktes kinematisches Problem
DOF	Degrees of Freedom
FP	Flächenportal
HKM	hybridkinematischer Mechanismus
IEM	Fraunhofer-Institut für Entwurfstechnik Mechatronik
IKP	inverses kinematisches Problem
MKS	Mehrkörpersimulation
MPR	modellprädiktive Regelung
OPC UA	Open Platform Communications Unified Architecture
OPC UA TSN	OPC UA over Time Sensitive Networking
PAC	Programmable Automation Controller
PKM	parallelkinematischer Mechanismus
PTP	Punkt-zu-Punkt
RFID	Radio Frequency Identification
RPI	Raspberry Pi
SKM	seriellkinematischer Mechanismus
SPS	speicherprogrammierbare Steuerung
ST	*strukturierter Text*
TCP	Tool Center Point

Abkürzungsverzeichnis der Optimierer

(L-)BFGS Quasi-Newton	(Limited-memory-)BFGS Quasi-Newton
ALGENCAN	Augmented Lagrangian with GENCAN
ALPSO	Augmented Lagrangian Particle Swarm Optimizer
BFGS	Broyden-Fletcher–Goldfarb-Shannon
BOBYQA	Bound Optimization BY Quadratic Approximation
COBYLA	Constrained Optimization BY Linear Approximation
CONMIN	CONstrained function MINimization
DIRECT	DIviding RECTangles
FSQP	Feasible Sequential Quadratic Programming
GCMMA	Globally Convergent Method of Moving Asymptotes
IPM	Interior Point Method
ISRES	Improved Stochastic Ranking Evolution Strategy
KSOPT	Kreisselmeier–Steinhauser OPTimizery
LINCOA	LINearly Constrained Optimization Algorithm
MFD	Method of Feasible Directions
MIDACO	Mixed Integer Distribution Ant Colony Optimization
MMA	Method of Moving Asymptotes
NEWUOA	NEW Unconstrained Optimization Algoritm
PRAXIS	PRincipal AXIS
Sbplx	Name der Neuimplementierung des Subplex-Algorithmus
SDPEN	Sequential Penalty Derivative-free method
SLSQP	Sequential Least SQuares Programming
SOLVOPT	SOLver for local OPTimization problems
SUBPLEX	SUBspace-searching simPLEX
UOBYQA	Unconstrained Optimization BY Quadratic Approximation

Symbolverzeichnis

Name	Beschreibung	Einheit
${}^{s^*,s}\mathbf{A}(\theta_s)$	Drehmatrix um Winkel θ_s zwischen initialem und ausgerichtetem KS des Körpers s um z-Achse mit $s \in \mathbb{N}$	–
${}^{s-1,s}\mathbf{A}(\theta_s)$	Homogene Drehmatrix um Winkel θ_s zwischen den Körpern $(s-1)$ und s mit $s \in \mathbb{N}$	–
${}^{s-1,s^*}\mathbf{A}$	Drehmatrix zwischen KS des Körpers $(s-1)$ und Initialstellung des KS von Körper s mit $s \in \mathbb{N}$	–
a_i	Beschleunigung bzw. Verzögerung des Antriebs M_i	$\mathrm{mm\,s^{-2}}$
$a_{i,\max}$	Maximale Beschleunigung des Antriebs M_i	$\mathrm{mm\,s^{-2}}$
$a_{i,\min}$	Maximale Verzögerung des Antriebs M_i	$\mathrm{mm\,s^{-2}}$
α	Kardanwinkel des Mechanismus (x-Achse)	°
α_i	*Denavit-Hartenberg-Parameter*: Drehwinkel um x_i zur Überführung von z_{i-1} in z_i	°
β	Kardanwinkel des Mechanismus (y-Achse)	°
δ_i	Winkel zwischen dem Kardan- bzw. Kugelgelenken K_i bzw. S_i des Tricept und der x-Achse des jeweiligen Bezugskoordinatensystems	°
d_i	*Denavit-Hartenberg-Parameter*: minimaler Abstand zwischen den Ursprüngen von K_{i-1} und K_i entlang z_{i-1}	m
$e_i(\Delta q_i)$	Prozentuale Abweichung zwischen der realen und der modellbasierten Verfahrzeit des Antriebs M_i in Abhängigkeit des Verfahrwegs $\Delta q_{i,k+\nu\|k}$	%
$f_1() \ldots f_m()$	Funktionen	–
$\underline{f}$	Vektor von Funktionen $f_1() \ldots f_m()$	–
$g_1() \ldots g_n()$	Funktionen	–
$\underline{g}$	Vektor von Funktionen $g_1() \ldots g_n()$	–
i	Allgemeiner Laufindex, etwa für die Antriebe	–
n	Laufindex für Spalte einer Matrix	–
p	Laufindex für Elemente eines Vektors und Zeile einer Matrix	–
$\mathbf{J}$	*Jakobi-Matrix*	–
$\min(\Delta J_{\mathrm{rel}})$	Abbruchgrenze des Optimierers für die kleinste Änderung des Wertes der Gütefunktion	s
$J_k(\underline{q}_k, \underline{q}_{k+\nu\|k})$	Gütefunktion	s
k	Laufindex der Sollposition mit $1 \leq k \leq N$ $k \in \mathbb{N}$	–

Name	Beschreibung	Einheit
K_i	Kardangelenke des Tricept	–
KS	Kardanschubgelenk	–
M_i	Antriebe	–
$\mathbf{M}$	Beispiel für Matrix	–
${}^{A,B}m_{23}$	Element aus der zweiten Zeile und dritten Spalt der beispielhaften Matrix ${}^{A,B}\mathbf{M}$	–
${}^{A,B}\mathbf{M}$	Beispiel für Transformationsmatrix mit Bezug	–
M_{wi}	Antriebe des aktiven Werkzeugs	–
N	Länge einer Sollpositionsfolge	–
n_{it}	Maximal zulässige Anzahl an Iterationen	–
n_{p}	Prädiktionshorizont	–
ν	Laufindex innerhalb des Prädiktionshorizonts n_{p} mit $1 \leq \nu \leq n_{\mathrm{p}}$ $\nu \in \mathbb{N}$	–
$\vec{n}_{xy}$	Normalenvektor der x-y-Ebene des Werkstücks	–
O_{B}	Ursprung des Koordinatensystems der bewegten Tricept-Plattform	–
O_{KS}	Ursprung des Koordinatensystems des Kardanschubgelenks	–
O_{P}	Ursprung des Portal-festen Koordinatensystems	–
O_{S}	Ursprung des Koordinatensystems der Ebene der Kugelgelenke	–
O_{T}	Ursprung des Tricept-festen Koordinatensystems	–
O_{UR}	Ursprung des UR-festen Koordinatensystems	–
O_{W}	Ursprung des Welt-Koordinatensystems	–
$p_{\mathrm{qmax}}(\underline{q}_{k+\nu\|k})$	Strafterm der maximalen Verfahrwegsgrenze	mm
$p_{\mathrm{qmin}}(\underline{q}_{k+\nu\|k})$	Strafterm der minimalen Verfahrwegsgrenze	mm
q_i	Antriebsposition des Antriebs M_i	mm
Δq_i	Verfahrweg des Antriebs M_i	mm
$\Delta q_{i,\mathrm{acc}}$	Verfahrweg des Antriebs M_i während der Beschleunigung	mm
$\Delta q_{i,\mathrm{b}}$	Grenzlänge des Verfahrwegs, ab der eine Bewegung des Antriebs M_i mit Erreichen der max. Geschwindigkeit erfolgt	mm
$\Delta q_{i,\mathrm{con}}$	Verfahrweg des Antriebs M_i mit konstanter Geschwindigkeit	mm
$\Delta q_{i,\mathrm{dec}}$	Verfahrweg des Antriebs M_i während der Verzögerung	mm
$\Delta q_{i,k+\nu\|k}$	Verfahrweg des Antriebs M_i zum Erreichen der Sollposition mit Index $k+\nu$ in Abhängigkeit der Position mit Index k	mm
$\Delta q_{\mathrm{max}}(\vec{Z}_{\mathrm{TM}})$	Verfahrweg der langsamsten Achse für die aus Sollposition $\vec{Z}_{\mathrm{TM}}$ resultierende Bewegung	mm
$\min(\Delta q_5, \Delta q_6)$	Abbruchgrenze des Optimierers für die kleinste Änderung der Optimierungsvariablen	mm
$q_{i,0}$	Initiale Position des Antriebs M_i	mm
$q_{i,\mathrm{os}}$	Grenze, ab der der Strafterm vor der eigentlichen maximalen bzw. minimalen Verfahrwegsgrenze des Antriebs M_i greift	mm

Name	Beschreibung	Einheit
$q_{i,\max}$	Maximale Position des Antriebs M_i	mm
$q_{i,\min}$	Minimale Position des Antriebs M_i	mm
$\Delta q_{i,\mathrm{r}}$	Resultierender Verfahrweg des Antriebs M_i	mm
$\underline{q}$	Vektor der Antriebspositionen	–
$\underline{\dot{q}}$	Geschwindigkeitsvektor der Antriebe/Gelenke	–
$\underline{q}_k$	Vektor der Antriebspositionen im Schritt k	–
$\underline{q}_{\mathrm{T1},k}$	Vektor der Antriebspositionen des ersten Teilmechanismus im Schritt k	–
$\underline{q}_{\mathrm{T2},k}$	Vektor der Antriebspositionen des zweiten Teilmechanismus im Schritt k	–
q_{wi}	Antriebsposition des Antriebs M_{wi}	mm
$r_i(\vec{Z}_{\mathrm{TM}})$	Übersetzungsverhältnis des virtuellen Getriebes für Antrieb M_i in Abhängigkeit der Sollposition $\vec{Z}_{\mathrm{TM}}$	–
r_{K}	Abstand der Kardangelenke vom Ursprung O_{T} in der x-y-Ebene	mm
${}_{\mathrm{O_T}}\vec{r}_{\mathrm{K}_i}$	Ortsvektor des Kardangelenks K_i in Tricept-Koordinaten	–
${}_{\mathrm{O_T}}\vec{r}_{\mathrm{S}_i}$	Ortsvektor des Kugelgelenks S_i in Tricept-Koordinaten	–
${}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,r}}$	Roll-Winkel des UR10-TCPs	°
R_{r}	Roll-Winkel des UR10-TCPs (Kurzform)	°
${}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,p}}$	Pitch-Winkel des UR10-TCPs	°
R_{p}	Pitch-Winkel des UR10-TCPs (Kurzform)	°
${}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,y}}$	Yaw-Winkel des UR10-TCPs	°
R_{y}	Yaw-Winkel des UR10-TCPs (Kurzform)	°
r_{S}	Abstand der Kugelgelenke vom Ursprung O_{S} in der x-y-Ebene	mm
${}^{s-1,s}\mathbf{R}_h(\vec{Z}_{\mathrm{TT}})$	Homogene Translationsmatrix zwischen den Körpern $(s-1)$ und s mit $s \in \mathbb{N}$ in Abhängigkeit von Ortsvektor $\vec{Z}_{\mathrm{TT}}$	–
${}_{\mathrm{s-1}}\vec{r}_{\mathrm{s}}(\vec{Z}_{\mathrm{TT}})$	Translationsvektor zwischen den Körpern $(s-1)$ und s mit $s \in \mathbb{N}$ in Abhängigkeit von Ortsvektor $\vec{Z}_{\mathrm{TT}}$	–
$r_t(\vec{Z}_{\mathrm{TT}})$	Translation in Richtung der x-Achse des jeweiligen Hilfs-Koordinatensystems abhängig von Ortsvektor $\vec{Z}_{\mathrm{TT}}$	mm
s_{K}	Abstand der Kugelgelenke vom Ursprung O_{S} in z-Richtung	mm
s_{KS}	Abstand zwischen dem Kardanschubgelenk KS und O_{T}	mm
s_{OP}	Konstruktionsbedingter fester Abstand zwischen O_{W} und O_{P} in z-Richtung des Welt-Koordinatensystems	mm
S_i	Kugelgelenke des Tricept	–
$s_{\mathrm{TS},0}$	Abstand zwischen O_{S} und O_{T} bei Initialstellung der Antriebe $M_1 \ldots M_3$	mm
Δs_{TS}	Änderung des Abstands zwischen O_{S} und O_{T}	mm
s_{TT}	Länge des Vektors ${}_{\mathrm{O_2}}\vec{r}_{\mathrm{TT}}$	mm
$T_{\mathrm{calcmax,sb}}$	Maximale Rechenzeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einer SPS	s

Name	Beschreibung	Einheit
$T_{\text{calcmax,sr}}$	Maximale Rechenzeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einem RPI	s
$\overline{T}_{\text{calc,sb}}$	Mittlere Optimierungszeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einer SPS	s
$\overline{T}_{\text{calc,sm}}$	Mittlere Optimierungszeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* in *MATLAB*	s
$\overline{T}_{\text{calc,sr}}$	Mittlere Optimierungszeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einem RPI	s
TM	TCP des Mechanismus	–
$T1$	TCP des ersten Teilmechanismus	–
$T2$	TCP des ersten Teilmechanismus	–
TP	TCP des Portals	–
TT	TCP des Tricepts	–
TUR	TCP des UR 10	–
$TWks$	TCP des Werkstücks	–
$TWzk$	TCP des Werkzeugs	–
$\Delta t_{\max}(\vec{Z}_{\text{TM}})$	Verfahrzeit des langsamsten Antriebs für die Bewegung zum Erreichen von $\vec{Z}_{\text{TM}}$	s
$\Delta t_i(\Delta q_{i,k+\nu\|k})$	Verfahrzeit des Antriebs M_i in Abhängigkeit des Verfahrwegs $\Delta q_{i,k+\nu\|k}$	s
$\Delta t_{i,\text{acc}}(\Delta q_{i,k+\nu\|k})$	Zeit für die Beschleunigung des Antriebs M_i in Abhängigkeit des Verfahrwegs $\Delta q_{i,k+\nu\|k}$	s
$\Delta t_{i,\text{con}}(\Delta q_{i,k+\nu\|k})$	Zeit für eine Bewegung mit konstanter Geschwindigkeit des Antriebs M_i in Abhängigkeit des Verfahrwegs $\Delta q_{i,k+\nu\|k}$	s
$\Delta t_{i,\text{dec}}(\Delta q_{i,k+\nu\|k})$	Zeit für die Verzögerung des Antriebs M_i in Abhängigkeit des Verfahrwegs $\Delta q_{i,k+\nu\|k}$	s
$\Delta t_{i,\text{I}}(\Delta q_{i,k+\nu\|k})$	Verfahrzeit des Antriebs M_i bei Verfahrweg $\Delta q_{i,k+\nu\|k}$ mit Erreichen der max. Geschwindigkeit	s
$\Delta t_{i,\text{e}}(\Delta q_{i,k+\nu\|k})$	Abweichung zwischen der realen und der modellbasierten Verfahrzeit des Antriebs M_i in Abhängigkeit des Verfahrwegs $\Delta q_{i,k+\nu\|k}$	s
$\Delta\overline{T}_{\text{ot}}$	Prozentuale Einsparung des modellprädiktiven Ansatzes mit einem Optimierer der *Optimization Toolbox* in *MATLAB* gegenüber des Referenzverfahrens	%
$\Delta t_{i,\text{r}}(\Delta q_{i,k+\nu\|k})$	Reale Verfahrzeit des Antriebs M_i in Abhängigkeit des Verfahrwegs Δq_i	s
$\Delta\overline{T}_{\text{sb}}$	Prozentuale Einsparung des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einer SPS gegenüber des Referenzverfahrens	%
$\Delta\overline{T}_{\text{sm}}$	Prozentuale Einsparung des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* in *MATLAB* gegenüber des Referenzverfahrens	%
$\Delta\overline{T}_{\text{sm200}}$	Prozentuale Einsparung des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* bei $n_{\text{it}} = 200$ in *MATLAB* gegenüber des Referenzverfahrens	%

Name	Beschreibung	Einheit
$\Delta\overline{T}_{\mathrm{sr}}$	Prozentuale Einsparung des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einem RPI gegenüber des Referenzverfahrens	%
$\Delta t_{i,\mathrm{II}}(\Delta q_{i,k+\nu\|k})$	Verfahrzeit des Antriebs M_i bei Verfahrweg $\Delta q_{i,k+\nu\|k}$ ohne Erreichen der max. Geschwindigkeit	s
θ_s	Drehwinkel zwischen zwei durch rotatorische Gelenke verbundenen Körpern $(s-1)$ und s mit $s \in \mathbb{N}$	°
$T_{\mathrm{optmax,sb}}$	Maximale Optimierungszeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einer SPS	s
$T_{\mathrm{optmax,sr}}$	Maximale Optimierungszeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einem RPI	s
$\overline{T}_{\mathrm{opt,sb}}$	Mittlere Rechenzeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einer SPS	s
$\overline{T}_{\mathrm{opt,sr}}$	Mittlere Rechenzeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einem RPI	s
$\overline{T}_{\mathrm{ot}}$	Mittlere Verfahrzeit des modellprädiktiven Ansatzes mit einem Optimierer der *Optimization Toolbox* in *MATLAB*	s
${}^{O_{\mathrm{T}},O_{\mathrm{S}}}\mathbf{T}$	Transformationsmatrix zwischen KS der Kugelgelenke und tricept-festem KS	–
${}^{s-1,s}\mathbf{T}(\theta, \vec{Z}_{\mathrm{TT}})$	Transformationsmatrix zwischen den Körpern $(s-1)$ und s mit $s \in \mathbb{N}$ in Abhängigkeit von Winkel θ_s und Ortsvektors $\vec{Z}_{\mathrm{TT}}$	–
$\overline{T}_{\mathrm{ref}}$	Mittlere Verfahrzeit des Referenzansatzes für einen Satz der Länge N	s
$\overline{T}_{\mathrm{sb}}$	Mittlere Verfahrzeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einer SPS	s
$\overline{T}_{\mathrm{sm}}$	Mittlere Verfahrzeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* in *MATLAB*	s
$\overline{T}_{\mathrm{sm200}}$	Mittlere Verfahrzeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* bei $n_{\mathrm{it}} = 200$ in *MATLAB*	s
$T_{\mathrm{sm,max}}$	Maximale Verfahrzeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* in *MATLAB*	s
$T_{\mathrm{sm,min}}$	Minimale Verfahrzeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* in *MATLAB*	s
$\overline{T}_{\mathrm{sr}}$	Mittlere Verfahrzeit des modellprädiktiven Ansatzes mit dem Optimierer *Sbplx* auf einem RPI	s
v_i	Geschwindigkeit des Antriebs M_i	$\mathrm{mm\,s^{-1}}$
$v_{i,1}(\Delta q_{i,k+\nu\|k})$	Maximal erreichte Geschwindigkeit des Antriebs M_i bei Verfahrweg $\Delta q_{i,k+\nu\|k}$ mit $v_{i,1} < v_{i,\mathrm{max}}$	$\mathrm{mm\,s^{-1}}$
$v_{i,\mathrm{g}}$	Aus Kopplung an Masterachse resultierende Geschwindigkeit des Antriebs M_i	$\mathrm{mm\,s^{-1}}$
$v_{i,\mathrm{max}}$	Maximale Geschwindigkeit des Antriebs M_i	$\mathrm{mm\,s^{-1}}$
$v_{i,0}$	Initiale Geschwindigkeit des Antriebs M_i	$\mathrm{mm\,s^{-1}}$
$\underline{v}$	Beispiel für allgemeinen Vektor	–

Name	Beschreibung	Einheit
$v_{,2}$	Zweite Komponente eines beispielhaften allgemeinen Vektors $\underline{v}$	–
${}_{\mathrm{A}}v_{\mathrm{B,y}}$	y-Komponente eines beispielhaften geometrischen Vektors ${}_{\mathrm{A}}\vec{v}_{\mathrm{B}}$	mm
$\vec{v}$	Beispiel für geometrischen Vektor	–
${}_{\mathrm{A}}\vec{v}_{\mathrm{B}}$	Beispiel für geometrischen Vektor mit Bezug	–
w_1	Gewichtung der Verfahrzeit	–
w_2	Gewichtung des Strafterm der maximalen Verfahrwegsgrenze	–
w_3	Gewichtung des Strafterm der minimalen Verfahrwegsgrenze	–
w_4	Gewichtung der Arbeitsraum- und Gelenkgrenzen des UR10	–
$\underline{X}$	Zustandsvektor des Mechanismus	–
${}_{\mathrm{O_W}}\vec{\xi}_{\mathrm{TM}}$	Pose des Mechanismus in Weltkoordinaten	–
${}_{\mathrm{O_{UR}}}\vec{\xi}_{\mathrm{TUR}}$	Pose des UR10 bezogen auf sein Basiskoordinatensystem	–
${}_{\mathrm{O_{UR}}}\vec{\xi}_{\mathrm{TURres}}$	Resultierende Pose des UR10 in *RecurDyn* bezogen auf sein Basiskoordinatensystem	–
$\underline{\xi}_{\mathrm{ref}}$	Vektor der festen Bewegungsaufteilung im Ramen des Referenzverfahrens	–
X_{K}	x-Achse des Kardangelenk-Koordinatensystems mit dem Ursprung O_{K}	–
X_{S}	x-Achse des Kugelgelenk-Koordinatensystems mit dem Ursprung O_{S}	–
X_{T}	x-Achse des Tricept-Koordinatensystems mit dem Ursprung O_{T}	–
X_{W}	x-Achse des Welt-Koordinatensystems mit dem Ursprung O_{W}	–
Y_{K}	y-Achse des Kardangelenk-Koordinatensystems mit dem Ursprung O_{K}	–
Y_{S}	y-Achse des Kugelgelenk-Koordinatensystems mit dem Ursprung O_{S}	–
Y_{T}	y-Achse des Tricept-Koordinatensystems mit dem Ursprung O_{T}	–
Y_{W}	y-Achse des Welt-Koordinatensystems mit dem Ursprung O_{W}	–
Z_{K}	z-Achse des Kardangelenk-Koordinatensystems mit dem Ursprung O_{K}	–
Z_{S}	z-Achse des Kugelgelenk-Koordinatensystems mit dem Ursprung O_{S}	–
$\underline{Z}$	Allgemeiner Ortsvektor (ohne explizite Angabe etwa des Bezugs)	–
$\dot{\underline{Z}}$	Zeitliche Änderung des Ortsvektors $\underline{Z}$	–

Name	Beschreibung	Einheit
${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TM}}$	Ortsvektor des Gesamtmechanismus-TCPs in Weltkoordinaten	–
$\vec{Z}_{\mathrm{TM}}$	Ortsvektor des Gesamtmechanismus-TCPs in Weltkoordinaten (Kurzform)	–
$\vec{Z}_{\mathrm{TM,0}}$	Initialer Ortsvektor des Gesamtmechanismus-TCPs in Weltkoordinaten (Kurzform)	–
$\vec{Z}_{\mathrm{TM},k}$	k-ter Ortsvektor des Gesamtmechanismus-TCPs in Weltkoordinaten (Kurzform)	–
$\vec{Z}_{\mathrm{TM,N}}$	N-ter Ortsvektor des Gesamtmechanismus-TCPs in Weltkoordinaten (Kurzform)	–
${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TP}}$	Ortsvektor des Portal-TCPs in Weltkoordinaten	–
${}_{\mathrm{O_W}}Z_{\mathrm{TP,x}}$	x-Koordinate des Portal-TCPs in Weltkoordinaten	mm
${}_{\mathrm{O_W}}Z_{\mathrm{TP,y}}$	y-Koordinate des Portal-TCPs in Weltkoordinaten	mm
${}_{\mathrm{O_W}}Z_{\mathrm{TP,z}}$	z-Koordinate des Portal-TCPs in Weltkoordinaten	mm
${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TMres}}$	Resultierender Ortsvektor des Gesamtmechanismus-TCPs in Weltkoordinaten	–
${}_{\mathrm{O_W}}Z_{\mathrm{TMres,x}}$	Resultierende x-Koordinate des Gesamtmechanismus-TCPs in Weltkoordinaten	mm
${}_{\mathrm{O_W}}Z_{\mathrm{TMres,y}}$	Resultierende y-Koordinate des Gesamtmechanismus-TCPs in Weltkoordinaten	mm
${}_{\mathrm{O_W}}Z_{\mathrm{TMres,z}}$	Resultierende z-Koordinate des Gesamtmechanismus-TCPs in Weltkoordinaten	mm
${}_{\mathrm{O_W}}R_{\mathrm{TM,r}}$	Roll-Winkel des Gesamtmechanismus-TCPs in Weltkoordinaten	°
${}_{\mathrm{O_W}}R_{\mathrm{TM,p}}$	Pitch-Winkel des Gesamtmechanismus-TCPs in Weltkoordinaten	°
${}_{\mathrm{O_W}}R_{\mathrm{TM,y}}$	Yaw-Winkel des Gesamtmechanismus-TCPs in Weltkoordinaten	°
${}_{\mathrm{O_T}}\vec{Z}_{\mathrm{TT}}$	Ortsvektor des Tricept-TCPs in Tricept-Koordinaten	–
$\vec{Z}_{\mathrm{TT}}$	Ortsvektor des Tricept-TCPs in Tricept-Koordinaten (Kurzform)	–
${}_{\mathrm{O_T}}Z_{\mathrm{TT,x}}$	x-Koordinate des Tricept-TCPs in Tricept-Koordinaten	mm
${}_{\mathrm{O_T}}Z_{\mathrm{TT,y}}$	y-Koordinate des Tricept-TCPs in Tricept-Koordinaten	mm
${}_{\mathrm{O_T}}Z_{\mathrm{TT,z}}$	z-Koordinate des Tricept-TCPs in Tricept-Koordinaten	mm
${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TT}}$	Ortsvektor des Tricept-TCPs in Weltkoordinaten	–
${}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,x}}$	x-Koordinate des UR10-TCPs in seinem Basiskoordinatensystem	mm
Z_{TURx}	x-Koordinate des UR10-TCPs in seinem Basiskoordinatensystem (Kurzform)	mm
${}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,y}}$	y-Koordinate des UR10-TCPs in seinem Basiskoordinatensystem	mm
Z_{TURy}	y-Koordinate des UR10-TCPs in seinem Basiskoordinatensystem (Kurzform)	mm
${}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,z}}$	z-Koordinate des UR10-TCPs in seinem Basiskoordinatensystem	mm

Name	Beschreibung	Einheit
Z_{TURz}	z-Koordinate des UR10-TCPs in seinem Basiskoordinatensystem (Kurzform)	mm
${}_{\mathrm{O_W}}Z_{\mathrm{TM,x}}$	x-Koordinate des Gesamtmechanismus-TCPs in Weltkoordinaten	mm
${}_{\mathrm{O_W}}Z_{\mathrm{TM,y}}$	y-Koordinate des Gesamtmechanismus-TCPs in Weltkoordinaten	mm
${}_{\mathrm{O_W}}Z_{\mathrm{TM,z}}$	z-Koordinate des Gesamtmechanismus-TCPs in Weltkoordinaten	mm
Z_{T}	z-Achse des Tricept-Koordinatensystems mit dem Ursprung O_{T}	–
Z_{W}	z-Achse des Welt-Koordinatensystems mit dem Ursprung O_{W}	–

1 Einleitung

Roboter und komplexe Mechanismen werden zunehmend in industrielle Abläufe integriert. Dies geschieht nicht erst im Zuge der vierten industriellen Revolution (kurz Industrie 4.0). Studien etwa der International Federation of Robots [Mül19] gehen jedoch davon aus, dass die Anzahl der jährlich weltweit neu eingesetzten Industrieroboter in den kommenden Jahren steigen wird. Für das Jahr 2023 wird etwa eine Steigerung 7 % gegenüber dem Vorjahr gesehen (siehe Bild 1-1). Die an die Roboter und ihre Steuerung gestellten An-

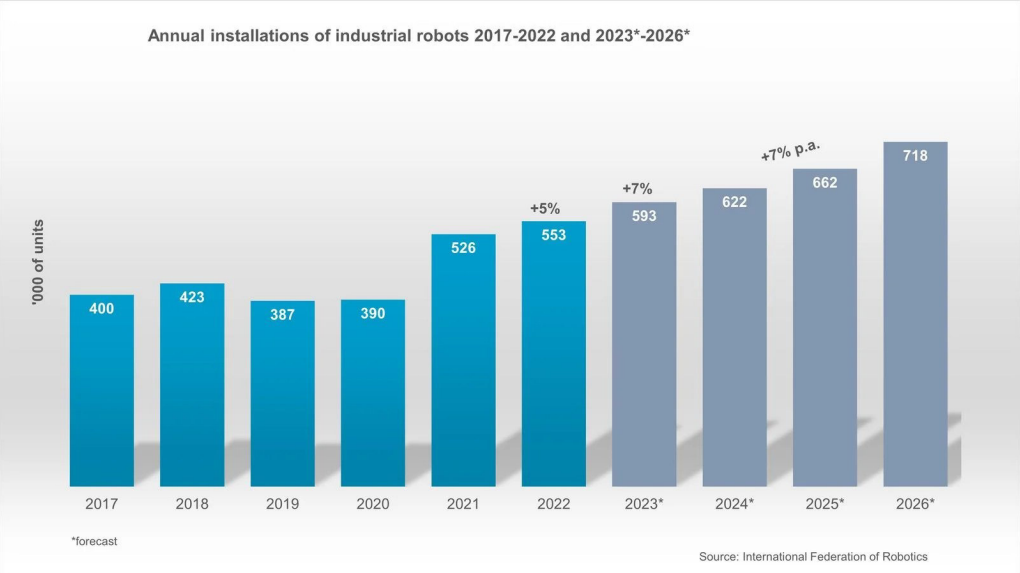


Bild 1-1: *Studie der International Federation of Robots bezüglich der Anzahl an jährlich neu eingesetzten Robotern [Int23]*

forderungen sind dabei einer starken Veränderung unterworfen. Während sie früher für immer wiederkehrende gleichbleibende Tätigkeiten eingesetzt wurden, etwa das Setzen von Schweißpunkten an dem immer gleichen Typ PKW-Karosse, müssen sie heute in immer flexibler werdende Fertigungen im Kontext Losgröße-1 eingebunden werden. Gerade die Vernetzung von Geräten, Maschinen und beweglichen Gegenständen, etwa den Produkten, im Rahmen sogenannter Cyber-Physischer (Produktions-)Systeme spielt nun eine bedeutende Rolle [Bun15]. Bisher wurden Roboter in der Regel weitestgehend losgelöst von umgebenden Anlagen und Systemen per festem, auf einem zum Roboter gehörenden Steuergerät implementierten, Programm auf eine definierte Aufgabe angepasst oder der Arbeitszyklus beziehungsweise die anzufahrenden Sollpositionen wurden per *Teach-In*[1] händisch vorgegeben. Die nun geforderte flexible, robuste oder gar autonome Fertigung mit Losgröße-1 erfordert jedoch selbstoptimierende Steuerungen, die sich automatisch auf wechselnde Aufgaben, unterschiedliche Werkstücke oder variierende Transportwege einstellen und mit weiteren Systemen kommunizieren sowie interagieren. Vom Bundes-

[1] Beim Teach-In werden die einzelnen Stützpunkte einer gewünschten Bewegungsbahn angefahren und für jeden Punkt die zugehörigen Stellung des Endeffektors bzw. die Gelenkpositionen gespeichert [Hau13], S. 236.

ministerium für Wirtschaft und Energie wird dies in [Bun15, S. 8] zu Cyber-Physischen Systemen wie folgt formuliert:

> *„Durch die Vernetzung können Planung und Steuerung von Fertigungs- und Logistik-Prozessen automatisiert und autonomisiert werden.* ***Robotik und Automatisierung wirken dabei zusammen.****“*

Zudem wird in [Bun15, S. 10] die Aussage

> *„****Intelligente Maschinen****,* ***hochflexible Roboter****, Positions- und Bewegungssensoren,* ***Automatisierungstechnik*** *und Leistungselektronik sind die* ***wichtigsten Komponenten*** *der oben beschriebenen* ***Cyber-Physischen Produktions-Systeme****.“*

hinsichtlich der Digitalisierung ausgewählter Branchen getroffen.

1.1 Motivation und Zielsetzung

Wird die im vorherigen Abschnitt beschriebene Entwicklung aus regelungstechnischer Sicht betrachtet, so lassen sich die aufgezeigten Herausforderungen und Aufgaben hinsichtlich der flexiblen automatischen Adaption auf unterschiedliche Aufgaben oder Werkstücke häufig zumindest teilweise als Optimierungsproblem formulieren. Wird beispielsweise der Pfad eines Werkzeugs betrachtet, so reicht im Kontext der flexiblen Fertigung eine einmalige Planung nicht mehr aus. Vielmehr muss für jedes Werkstück ein individueller Pfad beziehungsweise hierfür erforderliche Gelenkbewegungen ermittelt werden. Dies kann mittels einer Optimierung bezüglich eines definierten Kriteriums, etwa der Bearbeitungszeit, erfolgen. Dabei reicht eine vorab durchgeführte Offline-Optimierung aufgrund der dynamische Folge unterschiedlicher Aufgaben nicht aus. Im Kontext einer flexiblen Fertigung sind beispielsweise nur wenige zukünftige Werkstücke bekannt, etwa aufgrund eines Barcode- oder RFID-Lesegeräts, das knapp vor der Bearbeitungsstation montiert ist und nur eine Vorausschau von wenigen Werkstücke ermöglicht. Deren Reihenfolge wird zudem dynamisch durch vorhergehende Fertigungsschritte beeinflusst. Somit muss die Optimierung online während des Fertigungsprozesses erfolgen, um auf die dynamische Folge der Werkstücke und ihre spezifischen Eigenschaften reagieren zu können.

Doch nicht nur die Ansteuerung der Roboter ändert sich, auch immer neue kinematische Strukturen finden ihren Weg in die Industrie. Während der erste industriell eingesetzte Roboter *Unimate* [Hau13] von der Struktur ein seriellkinematischer Mechanismus (SKM) war, wurde 1962 von Gough und Whitehall ein parallelkinematischer Mechanismus (PKM) in Form eines Hexapoden vorgestellt [Gal16]. PKM weisen dabei eine verglichen mit SKM höhere Steifigkeit, Genauigkeit und Dynamik auf, weswegen sie zunehmend in industriellen Anwendungen eingesetzt werden [LRDW12]. Unter anderem finden sie Anwendung in Handhabungsaufgaben (engl. Pick and Place) [BB10], Fräszentren [PD12], im Medizinbereich [LRDW12] oder bei Sonderlösungen wie hochdynamischen Achsprüfständen [KOF+16]. Allerdings haben sie den Nachteil eines relativ kleinen Arbeitsraumes [Koc98]. Um diese Einschränkung zu kompensieren, wurde 1988 von Thornton der *Tetrabot* vorgestellt [Tho88; ZQC+16], eine Kombination aus einem PKM und einem SKM mit jeweils drei Freiheitsgraden (engl. Degrees of Freedom (DOF)). Solche Kombinationen werden auch als hybridkinematischer Mechanismus (HKM) bezeichnet. Um die Arbeitsräume von

SKM, PKM und HKM weiter zu vergrößern, werden die Mechanismen zunehmend mit weiteren Achsen kombiniert. Ein Beispiel dafür ist ein Sechsachs-Knickarmroboter, der zur Vergrößerung des Arbeitsraumes mit seiner Basis auf einer oder mehreren zusätzlichen Linearachsen montiert ist. Der resultierende Mechanismus erfüllt die Anforderungen an den Arbeitsraum, jedoch weist er nun sogenannte kinematische Redundanz auf. Um die daraus resultierenden Eigenheiten und Herausforderungen zu verstehen, kann das menschliche Bewegungsverhalten betrachtet werden. Laut Prof. Dr. rer. nat. Alexandra Reichenbach [Rei13], Professorin für Psychologie und Informatik, können

> *... natürlichen **Bewegungen**, zum Beispiel das Ergreifen eines Glases, ... durch eine **Vielzahl von Bewegungskombinationen und Gelenkstellungen** von Rumpf und distalen (körperfernen) Gliedmaßen erreicht werden. ... Diese **Redundanz des menschlichen Bewegungsapparates** bedingt sich durch die Tatsache, dass der **Körper mit weit mehr Freiheitsgraden ausgestattet ist als benötigt** werden, um in unserer dreidimensionalen Welt ein Ziel zu erreichen. Einen neurologisch gesunden Menschen stattet die beschriebene biomechanische Redundanz mit einem **enormen Flexibilitätsgrad** aus. Wird die Bewegung eines Gelenks kurzfristig eingeschränkt oder wird ein Gelenk suboptimal angesteuert, können andere reflexartig kompensieren, was die Leichtigkeit und Varianz alltäglicher Bewegungen charakterisiert. Auf der anderen Seite stellt diese **Flexibilität** das Gehirn auch vor die **Herausforderung aus der Vielzahl möglicher Bewegungen die gerade passende auszuwählen**.*

Der menschliche Körper wird somit als redundant angesehen. Abhängig davon, wie eine Person eine Aktion ausführen möchte, beispielsweise möglichst schnell oder mit geringem (Energie-) Aufwand, werden die vorhandenen (redundanten) Bewegungsmöglichkeiten etwa der Arme und Beine kombiniert. Ebenfalls wird dabei berücksichtigt, welche Bewegungen oder Aktionen als nächstes geplant sind. Zudem kann flexibel auf äußere Einflüsse oder sich ändernde Aufgaben reagiert werden. Allerdings wird hierzu neben den Vorteilen der Flexibilität auch angemerkt, dass gerade die Handhabung dieser Mehrdeutigkeit der Bewegungen eine große Herausforderung für das menschliche Gehirn darstellt. Wird das menschliche Bewegungsverhalten mit der Bewegung eines kinematisch redundanten Mechanismus verglichen, so sind verschiedene Parallelen zu sehen. Wird beispielsweise die Hand analog zum Tool Center Point (TCP) des Mechanismus angesehen, so gibt es in beiden Fällen für eine Aktion bzw. eine Sollposition verschiedene mögliche Stellungen der Gelenke. Technisch gesehen ist das inverse kinematische Problem (IKP) eines solchen Mechanismus nicht mehr eindeutig lösbar. Dies stellt eine große Herausforderung für die Ansteuerung und Regelung solcher Mechanismen dar. Es gilt hinsichtlich der redundanten Antriebe festzulegen, welchen Anteil an der Gesamtbewegungen jedes Gelenk übernimmt. Wie beim Menschen kann diese Wahl abhängig von einem zusätzlichen Ziel bzw. einem Gütemaß, etwa hinsichtlich der Minimierung der erforderlichen Zeit, erfolgen. Es liegt somit ein Optimierungsproblem vor.

Wissenschaftliche Konzepte, die auf dem Prinzip der Redundanz aufsetzen, sind dabei nicht neu. Sowohl im Bereich der PKM (etwa [NSS+02], [MH12]) als auch im Bereich der SKM werden sie schon länger genutzt. Jedoch stellt die Regelung solcher Systeme und die effiziente Nutzung der durch die Redundanz geschaffenen Möglichkeiten immer noch ein Forschungsfeld da. Eine sehr gute Möglichkeit, um die durch Redundanz vorhandenen Mehrdeutigkeit des IKP zu nutzen, bieten Optimierungsansätze. Mit diesen ist es möglich

das Bewegungsverhalten des Systems durch Nutzung der Mehrdeutigkeit unter Berücksichtigung mehrerer Kriterien zu optimieren. Sind beispielsweise zukünftige Sollpositionen bekannt, so kann dieses Wissen ebenfalls bei der Wahl einer optimierten Gelenkstellung einbezogen werden. Bezogen auf die erforderliche Flexibilität im Kontext Industrie 4.0 und dem Einsatz für Losgröße-1 kann diese Optimierung jedoch nicht wie in bisherigen Ansätzen offline vorab erfolgen. Vielmehr muss sie online zur Laufzeit ausgeführt werden. Dabei muss industrielle Hardware anstelle teurer Echtzeit-Entwicklungssysteme und Prototyping-Systeme eingesetzt werden, um hierdurch einen entscheidenden Schritt in Richtung industrieller Akzeptanz zu gehen und zudem die Vernetzung mit umgebenden (Automatisierungs-)Systemen im Sinne Cyber-Physischer Systeme zu ermöglichen. Gattringer et al. treffen hierzu in der Veröffentlichung [GRN13] zu nicht redundanten SKM die Aussage:

> *„**Most importantly**, all **procedures are evaluated** in **practical experiments** on a standard robot **with industrial control hardware** and the recorded measurements are presented, **a step often missing in publications** in this area.“*

Ähnlich hierzu sind die Aussagen von T. E. Hufnagel in seiner Dissertationsschrift zu Regelungskonzepten für parallelkinematische Mechanismen mit Aktorredundanz [Huf13]:

> *„Ein wichtiger Punkt ist die Umsetzung der Regler auf einer rein industriellen Plattform.“*

und von Reiter et al. bezogen auf die Planung zeitoptimaler Trajektorien für redundante SKM [RMG18]:

> *„**Time-optimal motion control** will only **find industrial applications** if the optimal motions can actually be **performed** by **standard industrial robots**. This is **not ensured by any optimal motion planning scheme proposed up to now**.“*

Allerdings stellt die echtzeitfähige Ausführung der Optimierung insbesondere für kinematisch redundante Mechanismen auf Industriehardware einen entscheidenden Unterschied zum aktuellen Stand der Wissenschaft und Technik dar. Hier besteht entsprechender Entwicklungsbedarf, um die aufgezeigten Herausforderungen zu meistern. In der vorliegenden Arbeit wird aus diesem Grund die Entwicklung eines in Echtzeit auf einer Industriesteuerung ausführbaren Ansatzes vorgestellt. Dieser nutzt die kinematische Redundanz und die damit verbundene Mehrdeutigkeit bei der Lösung des IKP zur Optimierung der Bewegungen hinsichtlich definierter Kriterien. Dabei wird vorrangig das zeitliche Verhalten des Mechanismus betrachtet mit dem Ziel, den für Punkt-zu-Punkt (PTP)-Bewegungen (engl. point-to-point) zwischen Sollpositionen erforderlichen Zeitaufwand zu minimieren und damit die Wirtschaftlichkeit des jeweiligen Mechanismus zu erhöhen. Gleichzeitig wird durch die Echtzeitfähigkeit des Ansatzes und die dadurch erreichte Ausführbarkeit zur Laufzeit ein Einsatz bei Fertigungs-Szenarien im Kontext von Losgröße-1 ermöglicht. Hierzu gilt es den erforderlichen Rechenaufwand weitestmöglich zu reduzieren. Zudem liegt ein weiterer Augenmerk auf der Implementierbarkeit des Ansatzes auf industrieller Hardware anstelle von teuren Echtzeit-Entwicklungssystemen und Prototyping-Systemen, um hierdurch eine industrielle Akzeptanz zu schaffen. Dazu soll der entwickelte Ansatz zudem auf unterschiedliche Mechanismen adaptierbar und im Rahmen eines Retrofit nachträglich in bestehende Systeme integriert werden können. Am Beispiel eines unter Vernachlässigung der Orientierung anwendungsspezifisch kinematisch redundanten HKM des

Fraunhofer IEMs bestehend aus einem sogenannten Tricept (parallelkinematischer Tripod mit zusätzlicher serieller Achse am TCP) und einem Flächenportal erfolgt die Umsetzung und Analyse des Ansatzes. Zusätzlich wird die Adaption auf einen weiteren redundanten Mechanismus als Kombination zweier SKM (Raumportal und daran montierter Sechsachs-Knickarmroboter) vorgestellt, um die Variabilität des Ansatzes zu verdeutlichen.

Die sich ergebenden Anforderungen an den zu entwickelnden Ansatz sind im folgenden Abschnitt zusammengefasst.

1.2 Definition der Anforderungen

Es ergeben sich die nachstehenden Anforderungen an den Ansatz, die es im Rahmen der vorliegenden Arbeit zu erfüllen gilt:

Handhabung der Unbestimmtheit des IKP kinematisch redundanter Mechanismen
Ein entscheidendes Merkmal kinematisch redundanter Mechanismen und gleichzeitig eine hohe Anforderung an ihre Regelung ist die Unbestimmtheit des IKP und die Nutzung der Redundanz zur zielgerichteten Beeinflussung des (Bewegungs-) Verhaltens. Um das Bewegungsverhalten eines kinematisch redundanten Mechanismus entsprechend definierter Vorgaben zu optimieren, muss der Ansatz diese Eigenschaft zwangsläufig berücksichtigen.

Onlinefähigkeit
Wie in der Einleitung und Motivation beschrieben, soll der zu entwickelnde Ansatz im Kontext der Losgröße-1-Fertigung eingesetzt werden. Aufgrund der dynamischen Reihenfolge der Werkstücke und damit der Sollpositionen des Mechanismus reicht eine vorab durchgeführte Offline-Optimierung nicht aus. Stattdessen muss der Ansatz zur Laufzeit ausgeführt werden und auf die variierenden Werkstücke und ihre jeweiligen Anforderungen reagieren.

Ausführbarkeit auf SPS und industrielle Akzeptanz
Im Fachmedium *elektrotechnik AUTOMATISIERUNG* [Ine19] wird die speicherprogrammierbare Steuerung (SPS) als

> *„... Der Urknall für die dritte und vierte industrielle Revolution...“*

und

> *„... Schlüsseltechnologie moderner Industriekonzepte ...“*

bezeichnet. Sie stellt eine entscheidende Komponente moderner Maschinen und Anlagen dar. Aus diesem Grund ist eine Umsetzung der angestrebten modellprädiktiven Planung zeitoptimierter Trajektorien auf einer SPS für den industriellen Einsatz unumgänglich. Dies deckt sich beispielsweise mit den Aussagen von Reiter et al. [RMG18] (siehe Abschnitt 1.1), die eine Umsetzung auf industrieller Hardware als entscheidend bezeichnen.

Echtzeitfähigkeit
Bei einem Einsatz des Ansatzes auf einer SPS als Echtzeitsystem muss dieser die daraus resultierenden Anforderungen erfüllen. Hierzu ist es erforderlich, dass die Zeit für die Ausführung des Ansatzes eine definierte Zeitschwelle nie überschreitet

(*harte Echtzeit*, siehe Abschnitt 2.2). Ansonsten kann es zu einem Absturz der SPS und damit des gesamten Prozesses kommen. Diese Zeitschwelle wird in der vorliegenden Arbeit als ≤ 1 ms definiert. Generell ist die erforderliche Zykluszeit des Ansatzes abhängig vom eingesetzten Mechanismus bzw. der Zykluszeit des jeweiligen Prozesses, somit können auch Zeiten im Bereich von Sekunden oder Minuten ausreichen. Da der im Rahmen der Arbeit vorgestellte Ansatz jedoch flexibel auf unterschiedliche kinematisch redundante Mechanismen mit unterschiedlichen zeitlichen Anforderungen adaptierbar sein soll, wird hier eine Zykluszeit von ≤ 1 ms (*Steuerungsechtzeit*, siehe Abschnitt 2.2) angestrebt.

Retrofit
Als Retrofit wird im Rahmen der vorliegenden Arbeit verstanden, dass der zu entwickelnde Ansatz auch für bereits bestehende Systeme und Mechanismen eingesetzt werden kann. Die Herausforderung wird darin gesehen, dass diese oftmals im produktiven Betrieb sind und somit zum einen eine zeitlich aufwendige Anpassung der Steuerungssoftware oder der Hardware nicht durchführbar ist. Zum anderen weisen die eingesetzten SPS nicht zwangsläufig die erforderlichen Leistungsparameter auf. Hierfür soll eine Möglichkeit geschaffen werden, um den Ansatz ohne tiefgehenden Eingriff in die bestehende Software und ohne Austausch der SPS einsetzen zu können.

Nutzung der Möglichkeiten moderner Automatisierungssysteme
Heutige Automatisierungssysteme bieten umfangreiche Funktionalitäten. Um die SPS im Sinne der echtzeitfähigen Ausführung des Ansatzes von Aufgaben zu entlasten, gilt es diese Möglichkeiten der Automatisierungssysteme zielgerichtet einzusetzen. Ein Beispiel hierfür ist die Auslagerung der Positions- und Geschwindigkeitsregelung der Antriebe von der SPS auf die Antriebsumrichter.

Kommerzielle Nutzbarkeit
Eine Anforderung im Rahmen der vorliegenden Arbeit ist die industrielle Akzeptanz. Damit einhergehend wird angestrebt, dass die Möglichkeit gegeben ist den Ansatz perspektivisch etwa als fertigen Baustein oder als Bibliothek für eine SPS vermarkten zu können. Um dies zu erreichen, muss von Beginn an, etwa bei der Wahl des Optimierers oder einer fertigen Bibliothek für die Optimierung, auf die Lizenzierungsbedingungen geachtet werden. Im idealen Fall wird nur auf Bibliotheken oder ähnliches zurückgegriffen, die ohne Lizenzgebühren und ohne die Forderung der Offenlegung des umgebenden Quellcodes eingesetzt werden können.

Adaptierbarkeit
Aufgrund der Vielzahl an möglichen kinematisch redundanten Mechanismen gilt es den Ansatz derart zu gestalten, dass er auf unterschiedliche Mechanismen adaptiert werden kann.

1.3 Vorgehensweise und Aufbau der Arbeit

In der vorliegenden Arbeit werden in Kapitel 2 zunächst wichtige Punkte der im Rahmen der Arbeit verwendeten Notation sowie relevante Begriffe und Definitionen vorgestellt, die das Verständnis der folgenden Kapitel erleichtern sollen. Darauf folgend wird in Kapitel 3 die Analyse des aktuellen Stands der Wissenschaft und Technik für im Rahmen der Arbeit

relevante Themenfelder dargelegt. Die Quellen werden in Form einer Kurzzusammenfassung vorgestellt, wobei sich die Reihenfolge am Erscheinungsjahr sowie thematischen Zusammenhängen verschiedener Quellen orientiert. Abgeschlossen wird der Abschnitt mit einer Zusammenfassung und einem Fazit.

Kapitel 4 beschäftigt sich mit dem entwickelten Ansatz als eigentlichen Kern der Dissertation. Hierbei wird zunächst die Grundidee vorgestellt, bevor näher auf einzelne Elemente wie die Gütefunktion und das Optimierungsproblem sowie das für den Vergleich der Leistungsfähigkeit des Ansatzes genutzte Referenzverfahren eingegangen wird.

Im Rahmen eines in Kapitel 5 beschriebenen ersten Anwendungsszenarios wird der Ansatz auf einen anwendungsbezogen kinematisch redundanten Mechanismus, bestehend aus einem Tricept und einem Flächenportal, adaptiert. Es werden der eingesetzte Mechanismus inklusive der Herleitung der Gleichungen des kinematischen Problems sowie die Validierung der Gleichungen mittels Mehrkörpersimulation und mögliche Anwendungsszenarien des Mechanismus vorgestellt.

Zur Absicherung der Funktionsfähigkeit des Ansatzes wird entsprechend des Vorgehens einer modellbasierten Entwicklung zunächst eine Umsetzung in der Software *MATLAB* durchgeführt, die in Kapitel 6 beschrieben ist. Neben der Generierung der eingesetzten Sollpositionsfolgen und der Darlegung der Ergebnisse wird auch die Wahl des Optimierers beschrieben, bei der der Fokus auf der Implementierbarkeit auf Industriesteuerungen unter Berücksichtigung von Lizenzbedingungen liegt.

Kapitel 7 beinhaltet die Umsetzung am realen System, wobei zwei unterschiedliche Ansätze beschrieben werden. Zum einen erfolgt die Umsetzung auf einer Kombination aus eingebettetem System und einer speicherprogrammierbaren Steuerung (SPS) mit einer auf der Open Platform Communications Unified Architecture (OPC UA) basierenden Kommunikation. Zum anderen wird die vollständige Implementierung auf einer SPS dargestellt.

Um die Leistungsfähigkeit des vorgestellten Ansatzes sowie die Adaptierbarkeit zu verdeutlichen, wird in Kapitel 8 die Umsetzung auf einen weiteren redundanten HKM vorgestellt, dieses mal jedoch als Kombination aus einem Sechsachs-Knickarmroboter sowie einem dreiachsigen Raumportal. Hierdurch sind sowohl Bewegungen in x-, y- als auch z-Richtung durch kombiniertes Verfahren der beiden Teilsysteme möglich, zudem kann die Orientierung des Gesamt-TCP über den Knickarmroboter gestellt werden.

Zum Abschluss wird eine Zusammenfassung der Arbeit sowie ein Ausblick auf mögliche weitere Forschungsfragen gegeben (Kapitel 9).

2 Grundlagen

In diesem Kapitel wird im ersten Abschnitt zunächst die verwendete Notation in Form einer Auflistung vorgestellt. Im zweiten Abschnitt erfolgt eine Erläuterung verschiedener relevanter Definitionen und Begriffe.

2.1 Notation

Die im Rahmen der vorliegenden Arbeit genutzte Notation ist wie folgt definiert:

- Skalare werden kursiv geschrieben und falls erforderlich mit einem kursiven Index versehen, wie etwa

$$a_i \tag{2-1}$$

  mit dem Index i.

- Geometrische Vektoren werden durch einen Pfeil über dem Formelzeichen gekennzeichnet, ein Beispiel hierfür stellt der Vektor

$$\vec{v} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2-2}$$

  dar. Soll etwa angegeben werden, dass ein Punkt B im Koordinatensystem A durch einen geometrischen Vektor $\vec{v}$ beschrieben wird, so kann dies durch

$${}_{\mathrm{A}}\vec{v}_{\mathrm{B}} \tag{2-3}$$

  angegeben werden. Dabei dient der Ursprung des Koordinatensystems A als Bezug. Soll beispielsweise das zweite Element des Vektors ${}_{\mathrm{A}}\vec{v}_{\mathrm{B}}$ genutzt werden, wird dies als

$${}_{\mathrm{A}}v_{\mathrm{B,y}} \tag{2-4}$$

  dargestellt.

- Allgemeine Vektoren wie

$$\underline{v} = \begin{bmatrix} v_1 \\ \vdots \\ v_p \end{bmatrix} = \begin{bmatrix} v_1 & \dots & v_p \end{bmatrix}^T \in \mathbb{R}^{p \times 1} \tag{2-5}$$

  werden durch einen Unterstrich gekennzeichnet. Hier erfolgt die Nutzung etwa des zweiten Elements des Vektors wie folgt:

$$v_2\ . \tag{2-6}$$

- Matrizen werden im Rahmen der Arbeit durch fett geschriebene Großbuchstaben gekennzeichnet:

$$\mathbf{M} = \begin{bmatrix} m_{11} & \cdots & m_{1n} \\ \vdots & \ddots & \vdots \\ m_{p1} & \cdots & m_{pn} \end{bmatrix} \in \mathbb{R}^{p \times n} \,. \tag{2-7}$$

  Werden Transformationsmatrizen verwendet, die eine definierte Überführung zwischen zwei Koordinatensystemen A und B beschreiben, so werden analog zur Schreibweise bei Vektoren die Bezugskoordinatensysteme angegeben:

$${}^{A,B}\mathbf{M} = \begin{bmatrix} m_{11} & \cdots & m_{1n} \\ \vdots & \ddots & \vdots \\ m_{p1} & \cdots & m_{pn} \end{bmatrix} \in \mathbb{R}^{p \times n} \,. \tag{2-8}$$

  Ähnlich zu Vektoren erfolgt die Angabe

$${}^{A,B}m_{2,3} \tag{2-9}$$

  eines Elements der Matrix, beispielsweise aus der zweiten Zeile und der dritten Spalte, falls erforderlich unter Angabe der relevanten Koordinatensysteme.

2.2 Begriffsdefinitionen

In diesem Abschnitt werden verschiedene im Rahmen dieser Arbeit relevante Begriffe aus dem Bereich der Robotik in alphabetischer Reihenfolge erläutert und die zugehörigen Definitionen vorgestellt. Neben den explizit genannten Quellen wurden dafür die studentischen Arbeiten [Blu16], [Cra18] und [Ker18] hinzugezogen.

Arbeitsraum
: Der Arbeitsraum eines Mechanismus ist die Zusammenfassung aller möglichen Punkte bzw. Positionen, die der TCP des Mechanismus durch Ausnutzung der Bewegungsmöglichkeiten der Achsen erreichen kann [Ion03b] (engl. reachable workspace [Web09]). Alle Positionen innerhalb des Arbeitsraumes, bei denen zusätzlich die Orientierung des TCP frei gewählt werden kann, bilden einen Teilraum davon (engl. dexterous workspace) [Web09].

***Denavit-Hartenberg*-Konvention**
: Die nachfolgende Beschreibung orientiert sich an [Web09]: Die sogenannte *Denavit-Hartenberg*-Konvention (DHK) dient der verkürzten Beschreibung der Lage zwischen zwei benachbarten Koordinatensystemen mit vier anstelle der üblicherweise erforderlichen sechs Parameter (drei translatorische und drei rotatorische Parameter). Anwendung findet sie insbesondere im Bereich der Robotik bei seriellkinematischen Mechanismen (vgl. Abschnitt 2.2, Absatz *Mechanismus - Seriellkinematische Mechanismen*), wobei die DHK die Angabe der Lage des Roboter-TCPs einschließlich der Lage der durch Körper miteinander verbundenen Gelenke ermöglicht. In ihrer Grundform ist die DHK ausschließlich für offene kinematische Ketten nutzbar, in denen jeder Körper maximal einen Vorgänger sowie einen Nachfolger haben

darf [DGM+08]. Dies ist bei den seriellkinematischen Mechanismen der Fall. Bei Anwendung der DHK wird sowohl die Basis des Roboters als auch jedes Gelenk mit einem Koordinatensystem versehen, wobei die relative Stellung benachbarter Koordinatensysteme K_{i-1} und K_i mittels vier Parametern, den sogenannten *Denavit-Hartenberg*-Parameter (DHP), beschrieben werden kann. Damit dies gilt, müssen bei der Definition der Koordinatensysteme vorgegebene Regeln zwingend eingehalten werden (siehe beispielsweise [Web09]). Zudem darf jedes Gelenk zwischen zwei Körpern nur einen Freiheitsgrad f (f = 1) aufweisen. Ist dies nicht der Fall, so werden anstelle eines Gelenks mit f > 1 insgesamt f Gelenke mit jeweils nur einem Freiheitsgrad eingesetzt. Diese Gelenke wiederum werden durch Körper verbunden, deren Masse und Maße zu null angenommen werden. In Bild 2-1 sind die wie folgt definierten DHP dargestellt:

- q_i (in der Literatur auch als Θ_i bezeichnet, im Rahmen der Arbeit wird für die Gelenkpositionen jedoch allgemein q_i verwendet): absoluter Drehwinkel um z_{i-1}, um x_{i-1} in x_i zu überführen
- d_i: minimaler Abstand zwischen den Ursprüngen von K_{i-1} und K_i entlang z_{i-1}
- a_i: Abstand der Ursprünge von K_{i-1} und K_i entlang der Achse x_i (immer positiv)
- α_i: Drehwinkel um x_i zur Überführung von z_{i-1} in z_i

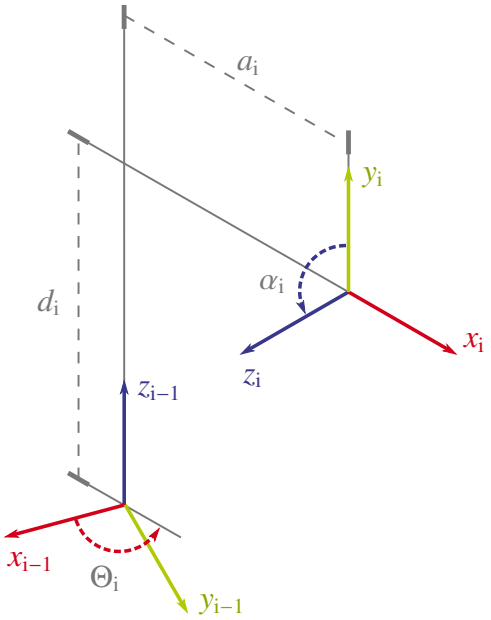


Bild 2-1: *Denavit-Hartenberg-Parameter nach [Web09]*

Für weitere Informationen wird etwa auf [Web09] verwiesen.

Direktes kinematisches Problem

Das direkte kinematische Problem (DKP), auch als Vorwärtstransformation bezeichnet, dient der Berechnung der Pose des TCP eines Mechanismus aus den zugehörigen Gelenkpositionen. Es stellt somit eine Vektor $\underline{f}$ von Funktionen f dar, mittels derer die Gelenkkoordinaten $\underline{q} = [q_1 \ \dots \ q_n]^T$ nach

$$\underline{Z} = \underline{f}(\underline{q}) = \left[f_1(\underline{q}) \dots f_m(\underline{q})\right]^T \tag{2-10}$$

auf den Vektor $\underline{Z} = [Z_1 \ \ldots \ Z_m]^T$ in kartesischen Koordinaten abgebildet werden. Für einen Mechanismus mit sechs DOF gilt dabei $m = 6$ (Bsp.: $\underline{Z} = [X \ Y \ Z \ \alpha \ \beta \ \gamma]^T$), n gibt die Anzahl der Gelenke an [Web09].

Echtzeit

Der Begriff Echtzeit wird etwa in der DIN 44300 definiert. In [Gev06] oder [WB05] werden die Kernaussagen dieser Norm wie folgt zusammengefasst: Echtzeit bedeutet, dass ein Zugriff auf z.B. Daten innerhalb eines vorher definierten Zeitrahmens und in äquidistanten Abständen garantiert sein muss (Rechtzeitigkeit). Auch zufällige Ereignisse müssen innerhalb der festgelegten Zeitschranken bearbeitet werden, sodass die Reaktion unter allen Umständen vorhersagbar ist (spontane Reaktion). Zudem müssen verschiedene Aufgaben mit unterschiedlichen Zeitanforderungen im selben Zeitintervall bearbeitet werden (Gleichzeitigkeit). Dabei ist Echtzeit jedoch nicht gleichzusetzen mit Schnelligkeit, es muss immer eine Grenze etwa in Form einer Zykluszeit angegeben werden. Es wird zwischen *harter Echtzeit* und *weicher Echtzeit* unterschieden. Bei harter Echtzeit ist ein Überschreiten der Zeitgrenzen unter allen Umständen inakzeptabel. Im Gegensatz dazu kann bei weicher Echtzeit ein Überschreiten toleriert werden, solange die dadurch entstehenden Risiken bzw. die *Kosten* eine Grenze nicht verletzen. Technische Prozesse, die etwa Regelungen oder die Integration von Antriebssystemen beinhalten, werden unter dem Gesichtspunkt der harten Echtzeit betrachtet [WB05; Gev06; SW07].

In der vorliegenden Arbeit wird der Begriff für harte Echtzeit mit einer Zeitgrenze von ≤ 1 ms verwendet. Diese Zykluszeit wird beispielsweise bei Bewegungssteuerungen genutzt und ist in [ISG19] als *Steuerungsechtzeit* bezeichnet.

Effektor

Bei dem (End-)Effektor handelt es sich um z.B. ein Werkzeug oder einen Greifer, der von dem Mechanismus durch den Arbeitsraum bewegt wird und etwa mit dem Werkstück interagiert [Web09].

Gelenkkoordinaten

Als Gelenkkoordinaten werden nach [Web09] die Winkel bzw. Positionen rotatorischer und translatorischer Bewegungsachsen bezeichnet. Sind die Gelenkkoordinaten eines Mechanismus bezogen auf eine feste Nullstellung bekannt, so kann mittels des DKP die daraus resultierende Pose des Endeffektors eindeutig bestimmt werden.

Inverses kinematisches Problem

Durch Lösung des IKP, auch als Rückwärtstransformation bezeichnet [Neu06], werden die Positionen der Gelenke eines Mechanismus berechnet, die für eine definierte Pose des TCP erforderlich sind. Es werden über einen Vektor $\underline{g}$ von Funktionen $g_i(\underline{Z})$ mit $i = 1 \ldots n \in \mathbb{N}$ die kartesischen Koordinaten nach

$$\underline{q} = \underline{g}(\underline{Z}) = g_1(\underline{Z}) \ldots g_n(\underline{Z}) \tag{2-11}$$

auf die Gelenkkoordinaten abgebildet, wobei n die Anzahl an Gelenken angibt [Web09].

Kinematik

Durch die Kinematik wird die Beziehung zwischen der Pose des TCP bezogen auf die Basis des Mechanismus und den zugehörigen Gelenken bzw. Aktoren festgelegt.

Je nach Art des Gelenks oder des Antriebs werden Drehwinkel oder translatorische Wege angegeben [Hau13].

Mechanismus

Als Mechanismus wird eine kinematische Kette bezeichnet, in der eins der Glieder als unbewegt erachtet wird und in der Regel als Bezugssystem dient [Ion03a]. Es handelt sich also um ein System verbundener Körper, welches die Übertragung und Umwandlung mechanischer Bewegungen gewährleistet [Kol12]. Es wird dabei zwischen verschiedenen Arten von Mechanismen unterschiedenen:

– Seriellkinematische Mechanismen
Bei einem seriellkinematischen Mechanismus (SKM) wird der TCP durch eine offene kinematische Kette mit der Basis verbunden. Diese Kette ermöglicht die Positionierung des TCP und besteht aus Gliedern und Gelenken beziehungsweise Antriebsachsen. Die Bewegungsachsen (translatorisch und rotatorisch) sind in der kinematischen Kette nacheinander angeordnet, sodass eine Achse alle folgenden Bewegungsachsen tragen und bewegen muss. Jede weitere Achse führt zu einem weiteren Freiheitsgrad (engl. DOF) des Mechanismus [Neu06].

In Bild 2-2 sind drei verschiedene Arten von SKM dargestellt. Bei dem linken Mechanismus handelt es sich um ein Linienportal (a) mit zwei translatorischen Achsen und zwei DOF, in der Mitte um einen vierachsigen Scara-Roboter (b) mit vier DOF und rechts um einen Sechsachs-Knickarmroboter (c) mit sechs DOF.

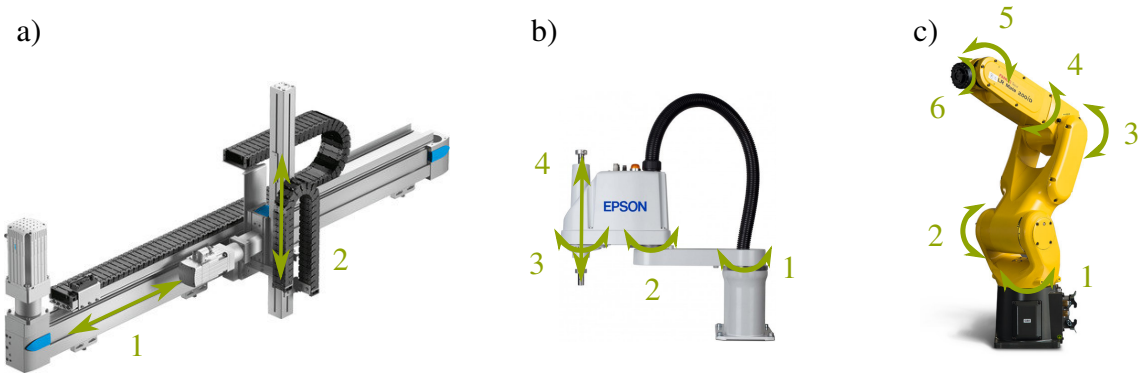


Bild 2-2: Beispiele für SKM nach [Blu16]: a) Linienportal (2 Achsen) [FES17] b) SCARA-Roboter (4 Achsen) [EPS11] c) Sechsachs-Knickarmroboter (6 Achsen) [FAN17]

– Parallelkinematische Mechanismen
Nach Definition von [Ion03b] ist ein parallelkinematischer Mechanismus (PKM) ein

> "Manipulator, bei dem die Bewegungssteuerung des Endeffektors über mindestens zwei voneinander getrennt angetriebene kinematische Ketten zwischen Gestell und Endeffektor erfolgt."

Der Aufbau von PKM wird häufig direkt in der Bezeichnung des Mechanismus angegeben. Ein in der Literatur oft genanter (kinematisch redundanter) PKM ist der 3-(P)RRR-Mechanismus. Die vorne stehende Nummer 3 gibt die Anzahl

der (gleichartigen) seriellen kinematischen Ketten an, die den TCP mit der Basis verbinden [ECB08]. Drehgelenke werden mit R, prismatische Gelenk mit P bezeichnet. Ein Unterstrich, etwa P, bedeutet, dass das Gelenk aktiv ist. Die Klammer wird genutzt um anzugeben, dass dieses Gelenk redundant ist. Die Betrachtung der kinematischen Ketten erfolgt angefangen bei der Basis in Richtung des TCP. Besteht ein Mechanismus aus mehreren kinematischen Ketten, die jedoch unterschiedlich aufgebaut sind, so kann die Gesamtstruktur additiv angegeben werden, etwa 2-(P)RRR + RPP [Mer06; Fd15].

In Bild 2-3 sind drei verschiedenen PKM dargestellt: ein Delta-Roboter mit zwei DOF (a) ein sogenannter Tripod mit drei DOF (b) sowie der hydraulisch aktuierte Hexapod (c) der Fachgruppe Regelungstechnik und Mechatronik des Heinz Nixdorf Instituts Paderborn mit sechs DOF [KOF+16].

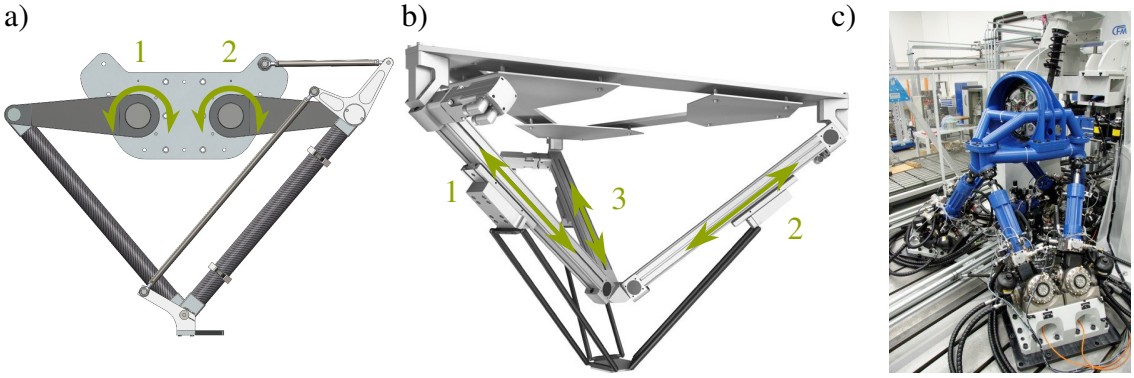


Bild 2-3: *Beispiele für PKM nach [Blu16]: a) Delta-Roboter (2 Achsen) [COD19] b) Tripod (3 Achsen) [Fes18] c) Hexapod (6 Achsen) [Jäk18]*

– Hybridkinematische Mechanismen
  Nach [Sch16] wird ein HKM allgemein als serielle oder parallele Kopplung von SKM und/oder PKM definiert, sodass sich eine Vielzahl an möglichen Kopplungsvarianten ergibt. Einige davon sind in Bild 2-4 dargestellt.
  Quellen wie [TGG98] und [Mil13] schränken die Kopplungsmöglichkeiten ein, indem ein HKM als Kombination einer PKM und zusätzlichen seriellen Komponenten definiert wird. Häufig wird dabei eine PKM mit einer oder mehreren seriellen Komponenten am TCP erweitert, etwa um den Arbeitsraum zu vergrößern. In Bild 2-5 ist als Beispiel für einen HKM ein Tricept der Firma *Tricept PKM* dargestellt. Die drei an der Basis befestigten parallelen Antriebe und die zusätzliche passive Führung sind um zwei zusätzliche seriell angeordnete rotatorische Gelenke ergänzt.

  In der vorliegenden Arbeit wird der Begriff HKM analog zu [Sch16] definiert. Die im Rahmen der Anwendungsszenarien eingesetzten Mechanismen werden zudem der Einfachheit halber kurz als HKM bezeichnet, ohne weitere Unterscheidungen zu machen.

Die Tabelle 2-1 liefert eine Zusammenfassung der Unterschiede zwischen den drei Arten an Mechanismen.

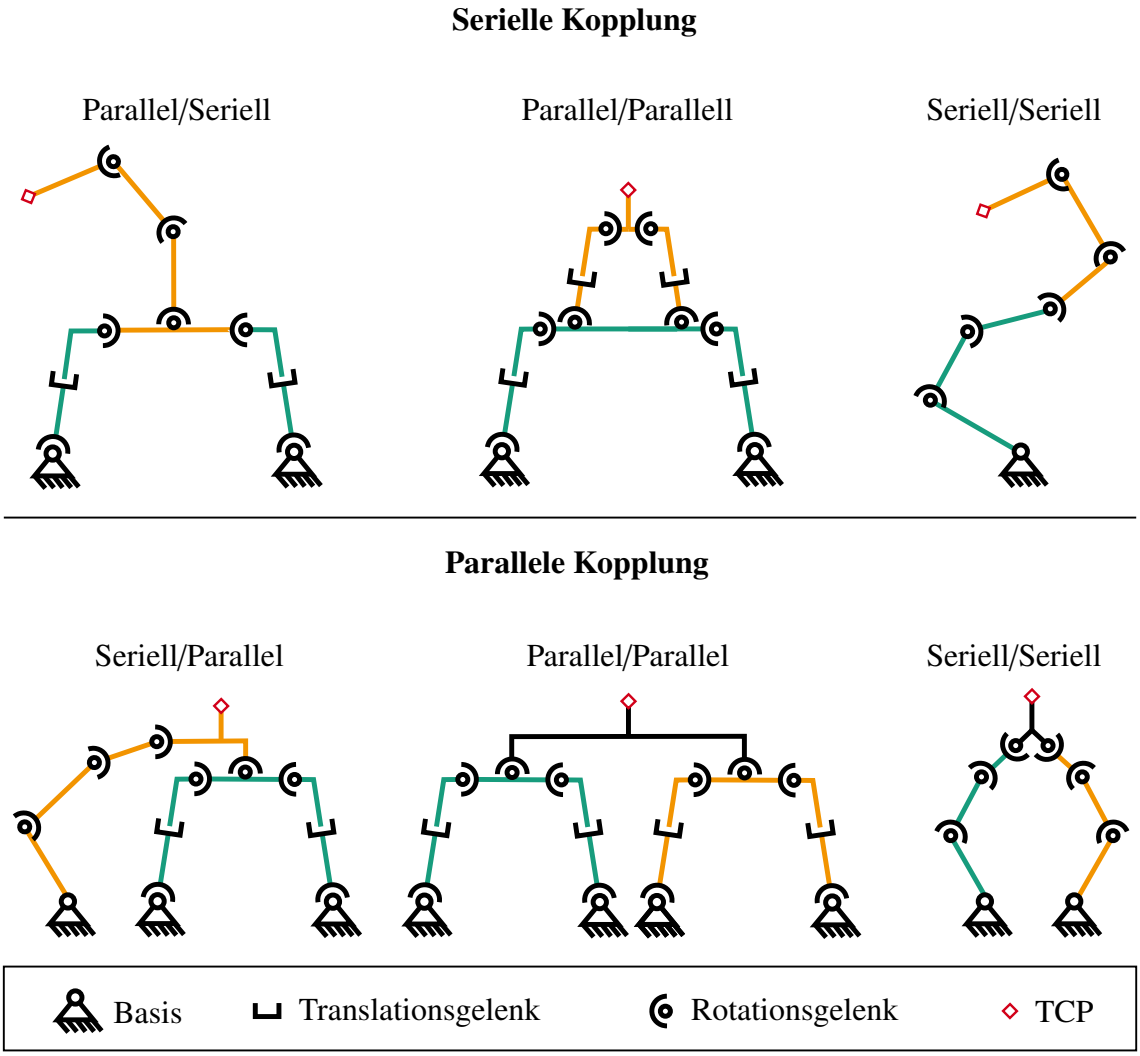


Bild 2-4: *Beispiele für unterschiedliche Kopplungsarten zweier serieller und/oder paralleler Mechanismen (grün: Teilmechanismus 1, orange: Teilmechanismus 2) nach [Sch16]*

Pose

Die Kombination aus der Position sowie der Orientierung des Endeffektorkoordinatensystems wird nach der ISO 8373:2012-03 als Pose bezeichnet [ISO12].

Punkt-zu-Punkt-Bewegung

Bei Punkt-zu-Punkt-Bewegungen (engl. PTP), auch als Punktsteuerung bezeichnet, werden die zeitlichen Sollwerte des Mechanismus in Form von Achssollwerten vorgegeben. Aus diesen resultiert eine Bewegungskurve des TCP, die nicht durch den Anwender vorgegeben wird. Es wird dabei zwischen zwei Varianten unterschieden. Bei asynchronen PTP-Bewegungen verfahren alle Achsen unabhängig von einander, sie kommen somit zu unterschiedlichen Zeitpunkten zum Stillstand. Bei synchronen PTP-Bewegungen hingegen erreichen alle Achsen zeitgleich ihre Zielposition. Hierzu werden abhängig von der Achse mit der höchsten erforderlichen Verfahrzeit (Leitachse) die Geschwindigkeiten der anderen Achsen reduziert [Web09].

Bild 2-5: *Beispiele für einen HKM: Tricept T606 [PKM18]*

Redundanz

Je nach Literaturquelle wird die Redundanz von Mechanismen in die folgenden verschiedene Arten unterteilt:

- Kinematische Redundanz
  Bei kinematischer Redundanz weißt der Mechanismus mehr interne Bewegungsfreiheitsgrade als Freiheitsgrade des TCP auf [ECB07b; BK15]. Dies bedeutet, dass sich Gelenke bewegen können, ohne das damit die Pose des TCP verändert wird. Anders ausgedrückt kann eine Pose des TCP mit mehr als einer Kombination von Gelenkpositionen erreicht werden. Schreiber beschreibt dies in seiner Dissertation [Sch04] als den Fall, in dem „die Dimension des Gelenkraumes größer als die Dimension des Endeffektor-Raums“ ist. Der sogenannte *Grad der Redundanz* gibt dabei die Differenz an zwischen der Anzahl an Gelenken und der Dimension der möglichen Endeffektorbewegungen. Ist dieser Grad größer Null so tritt der beschriebene Fall ein, dass Gelenkbewegungen durchgeführt werden können, ohne damit die Pose des TCP zu verändern [Sch04]. Ein Beispiel ist der in Bild 2-6 dargestellte 2RRR+(P)RRR-Mechanismus [KTHO10], bei dem die kinematische Redundanz durch die serielle Achse M_4 als Erweiterung zu den Achsen $M_1 \ldots M_3$ erzeugt wird.
  Kinematische Redundanz wird etwa genutzt, um Singularitäten zu vermeiden bzw. zu reduzieren, den Arbeitsraum zu vergrößern oder das Bewegungsverhalten zu verbessern. Jedoch ergeben sich für die inverse kinematische Gleichung unendlich viele Lösungen [ECB07b]. Hierdurch wird der Aufwand für die Regelung eines solchen Mechanismus deutlich vergrößert. Es werden entsprechende Ansätze benötigt, um mit dieser Mehrdeutigkeit umzugehen [ECB08].

- Aktorredundanz
  Bei einem Mechanismus mit Aktorredundanz liegen mehr Antriebe vor, als der Mechanismus an internen Bewegungsfreiheitsgraden aufweist [ECB07b; BK15]. Sie kann erreicht werden, wenn entweder passive Gelenke eines bestehenden Mechanismus mit einem Antrieb ausgestattet werden, oder einem Mechanismus eine kinematische Kette hinzugefügt wird, ohne das daraus zusätzliche Freiheitsgrade des TCP resultieren [Mül08] [KTHO10]. In Bild 2-7 ist ein entsprechender redundanter Mechanismus dargestellt.

Tabelle 2-1: *Unterschiede von SKM, PKM und HKM nach [HE98; Koc98; SW98; TMH98; Neu06; Rös11; LRDW12; ZQC+16]*

Eigenschaften	SKM	PKM	HKM
Absolute Genauigkeit	niedrig	hoch	mittel/hoch
Wiederholgenauigkeit	mittel	hoch	mittel/hoch
Größe des Arbeitsraums	groß	klein	mittel/groß
Steifigkeit	niedrig	hoch	mittel/hoch
Eigengewicht bezogen auf Last	hoch	niedrig	niedrig
Dynamik	mittel	hoch	mittel/hoch
Komplexität des DKP	Analytisch mittels der DHK	komplex	komplex
Komplexität des IKP	Abhängig vom Aufbau des Mechanismus	Abhängig vom Aufbau des Mechanismus	Abhängig vom Aufbau des Mechanismus
Antriebe außerhalb des Arbeitsraumes	nein	möglich	möglich

- Anwendungsbezogene Redundanz
  Bei der anwendungsbezogenen Redundanz handelt es sich um eine Sonderform der Redundanz. In der Literatur wird die Bezeichnung verwendet, wenn ein Mechanismus eine spezifische Aufgabe aus mehreren Gelenkstellungen heraus ausführen kann [Cra94; Nik01]. Hier ist die Dimension des für die Aufgabe erforderlichen Endeffektor-Raums kleiner als die tatsächliche Möglichkeit des Mechanismus, zudem ist sie kleiner als die Dimension des Gelenkraumes [Sch04]. Somit kann der selbe Mechanismus bezogen auf eine Aufgabe redundant sein, bezüglich einer anderen jedoch nicht. In [ZRRJ11] wird als Beispiel für anwendungsbezogenen Redundanz ein Roboterarm mit sechs Freiheitsgraden genannt, der Bohrvorgänge ausführen soll. Die anwendungsbezogenen Redundanz entsteht in diesem Fall dadurch, dass das Werkzeug um die eigentliche Bohrerachse beliebig gedreht werden kann, somit werden von den sechs Freiheitsgraden des Roboters nur fünf für die eigentliche Aufgabe benötigt. Dies ist in Anlehnung an [ZRRJ11] in Bild 2-8 dargestellt, wobei die Bohrerachse in z-Richtung des Werkzeug-Koordinatensystems liegt.

Roboterhand

Als Roboterhand werden bezogen auf einen Sechsachs-Knickarmroboter (siehe Bild 2-9) die Achsen vier, fünf und sechs bezeichnet.

Hierbei gibt es zwei grundlegende Formen. Die erste ist die sogenannte Zentralhand (siehe Bild 2-10, a), bei der die drei Handachsen orthogonal zueinander ausgerichtet sind und sich in einem Punkt schneiden. Die zweite Variante ist die Winkelhand

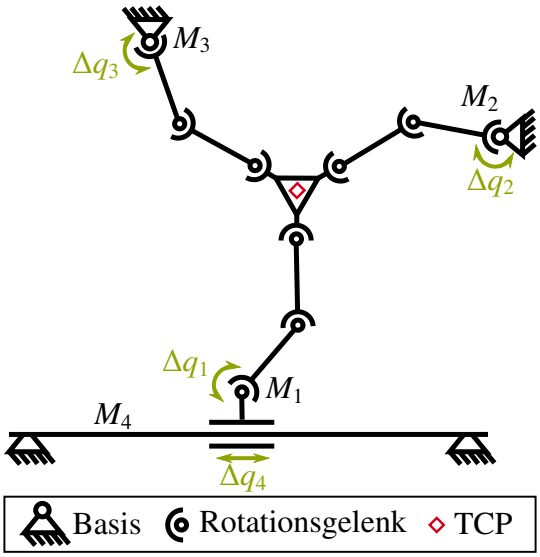


Bild 2-6: *Kinematisch redundanter Mechanismus (2RRR+(P)RRR) nach [KTHO10]*

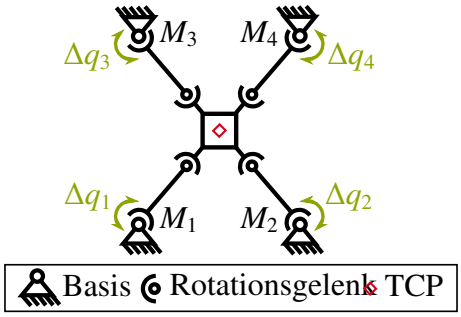


Bild 2-7: *Mechanismus mit Aktorredundanz nach [Koc98]*

(siehe Bild 2-10, b), die weniger kompakt und ohne gemeinsamen Schnittpunkt der drei Achsen aufgebaut ist [Sta09].

Singularitäten und *Jakobi*-Matrix

Die im Folgenden vorgestellte Identifikation von Singularitäten über die Determinante der sogenannten *Jakobi*-Matrix $\mathbf{J}$ oder ihrer Inversen $\mathbf{J}^{-1}$ ist nur bei quadratischen *Jakobi*-Matrizen möglich. Weist etwa der Mechanismus mehr Antriebe als DOF des TCP auf (vgl. Antriebsredundanz), so ist dies nicht mehr der Fall. Hier finden andere Ansätze Anwendung [Rös11], auf die jedoch an dieser Stelle nicht näher eingegangen wird.

Die *Jakobi*-Matrix gibt den Zusammenhang zwischen der Lineargeschwindigkeit sowie der zeitlichen Änderung der Orientierung des Effektors $\dot{\underline{Z}}$ und den Geschwindigkeiten $\dot{\underline{q}}$ der einzelnen Gelenke an. Sie wird mittels partiellen Ableitungen erster Ordnung gebildet (siehe Gleichung (2-12)), wobei eine stetige Differenzierbarkeit der Funktionen $\underline{f}$ (vgl. Abschnitt 2.2, Absatz *Direkte kinematische Gleichung*) vorausgesetzt ist [Sch16]. Dieser Zusammenhang geht jedoch verloren, wenn sich der Mechanismus in einer Singularität bzw. singulären Pose befindet. Im beschriebe-

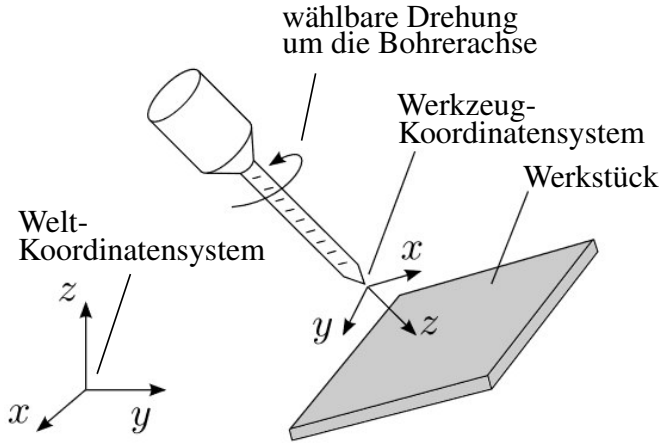


Bild 2-8: *Anwendungsbezogene Redundanz bei Bohrprozess nach [ZRRJ11]*

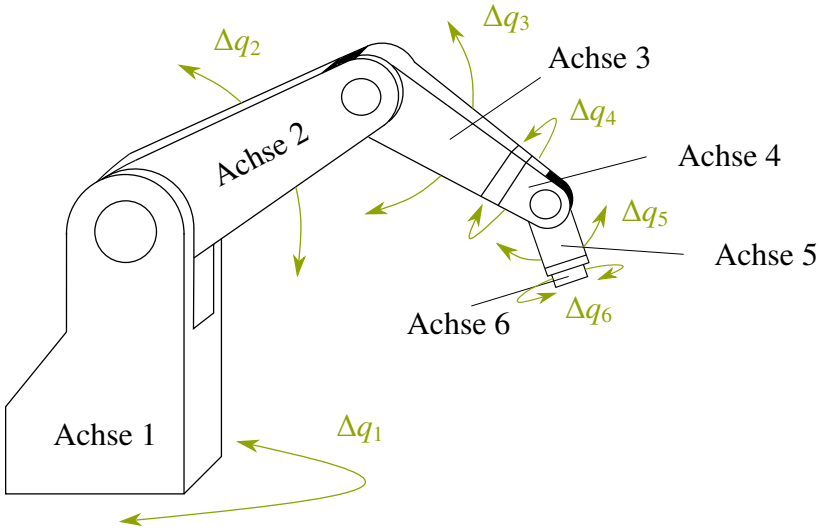


Bild 2-9: *Sechsachs-Knickarmroboter nach [BHS98]*

nen Fall quadratischer Matrizen handelt es sich bei Singularitäten mathematisch betrachtet somit um Sonderfälle bei der Berechnung der *Jakobi*-Matrix. [Web09]

$$\mathbf{J}(q_1,\ldots,q_n) = \begin{pmatrix} \frac{\partial f_1}{\partial q_1} & \cdots & \frac{\partial f_1}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial q_1} & \cdots & \frac{\partial f_n}{\partial q_n} \end{pmatrix} \tag{2-12}$$

Für einen Mechanismus mit sechs DOF gilt hierbei $n = 6$. Prinzipiell enthält die *Jakobi*-Matrix alle erforderlichen Informationen für einen eindeutigen Zusammenhang zwischen $\dot{\underline{Z}}$ und $\dot{\underline{q}}$ nach

$$\begin{aligned} \dot{\underline{Z}} &= \mathbf{J}(\underline{q}) \cdot \dot{\underline{q}} \\ \dot{\underline{q}} &= \mathbf{J}^{-1}(\underline{q}) \cdot \dot{\underline{Z}} \end{aligned} \tag{2-13}$$

Im Bezug auf PKM können drei Arten von Singularitäten unterschieden werden [GA90; Neu06; Rös11; Jun04]:

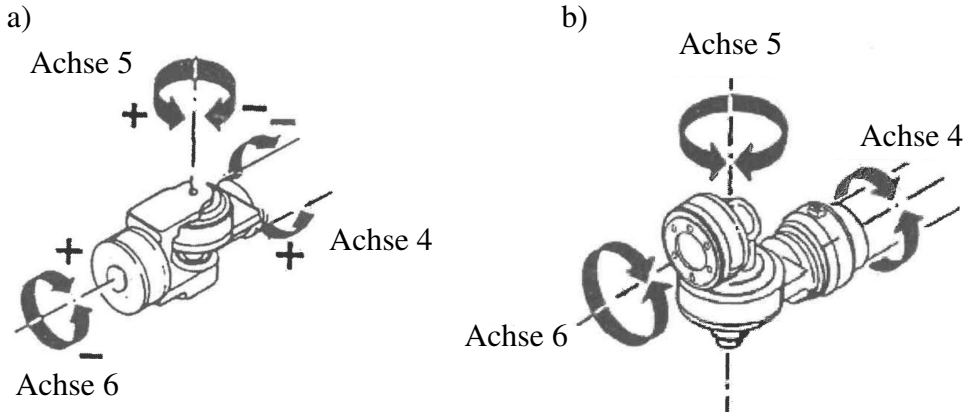


Bild 2-10: *a) Zentralhand und b) Winkelhand eines Sechsachs-Knickarmroboters [Sta09, S. 25]*

Singularität Typ-1
: Wird die Determinante der *Jakobi*-Matrix zu Null ($\det \mathbf{J} = 0$), so liegt eine Singularität des ersten Typs vor. Hierbei ist eine Antriebsbewegung möglich, ohne dass sich der TCP bewegt [Rös11]. Eine oder mehrere kinematische Ketten sind vollständig gestreckt oder zwei Achsen sind kollinear, der TCP verliert mindestens einen DOF [Sch16]. In Bild 2-11, a) ist der Fall einer vollständigen Kettenstreckung dargestellt. Dabei wird bei idealer Betrachtung die Bewegung von M_2 nicht auf den TCP übertragen, vielmehr wird das Antriebsmoment als unendlich hohe Kraft an diesen übertragen [Rös11].

Singularität Typ-2
: Im Fall einer Singularität des zweiten Typs wird die Determinante der inversen *Jakobi*-Matrix zu Null ($\det \mathbf{J}^{-1} = 0$). Hier kann der TCP in eine Richtung keine Kraft aufnehmen, da sich der Mechanismus in einer Klemmlage bezüglich der Antriebe befindet und das Drehmoment der Antriebe nicht auf den TCP übertragen wird [Rös11], siehe Bild 2-11, b). Dadurch sind infinitesimale Bewegungen des TCP ohne Bewegungen der Antriebe möglich [Sch16]. Diese Art der Singularität kann nur bei PKM auftreten, ein DKP von SKM ist immer analytisch lösbar [Sch16].

Singularität Typ-3
: Die in Bild 2-11, c) abgebildete Singularität des Typs 3 stellt eine Kombination der Typen 1 und 2 dar ($\det \mathbf{J} = 0$ und $\det \mathbf{J}^{-1} = 0$) [Rös11].

Hinsichtlich Singularitäten bei SKM findet sich in [SSVO09] eine Unterteilung in zwei unterschiedliche Arten:

Grenzsingularitäten
: Diese Art von Singularitäten liegt vor, wenn der serielle Mechanismus entweder vollständig aus- oder eingefahren ist (siehe Bild 2-12, a) und sich der TCP somit an der Arbeitsraumgrenze befindet.

Interne Singularitäten
: Liegt eine Pose vor, bei der die Achsen von zwei oder mehr Gelenken des SKM in einer Linie liegen, so ist die Stellung dieser Gelenke über die inverse

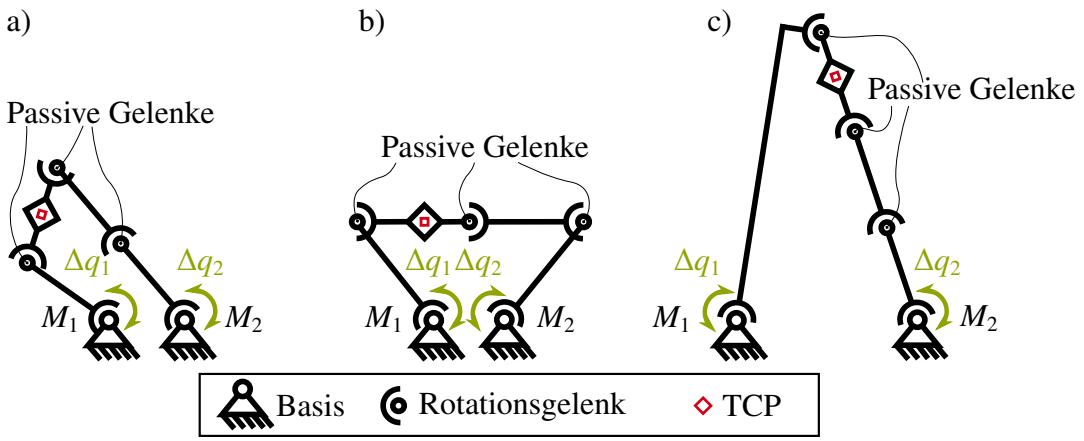


Bild 2-11: *Drei Singularitätstypen bei PKM nach [Rös11]*

kinematische Gleichung nicht mehr eindeutig zu bestimmen. Vielmehr kann die Pose durch unendliche viele Stellungen der betroffenen Gelenke erreicht werden. Dies wird als interne Singularität bezeichnet. Nach [Web09] wird in diesem Fall die Determinante der *Jakobi*-Matrix zu Null ($\det \mathbf{J} = 0$). In Bild 2-12, b) ist dieser Fall anhand eines Sechsachs-Knickarmroboters dargestellt, bei dem die Rotationsachsen der Gelenke vier und sechs in einer Linie liegen.

Tool Center Point

Der Tool Center Point (TCP) stellt einen Punkt mit entscheidender Prozessrelevanz am Effektor dar, wie die Spitze einer Schweißpistole oder eines Fräsers [Web09].

Nach der Erläuterung der notwendigen Begriffe und Definitionen wird im folgenden Kapitel ein Überblick über den Stand der Wissenschaft und Technik gegeben. Es wird dabei zwischen drei im Rahmen der vorliegenden Arbeit relevanten Themenbereichen unterschieden.

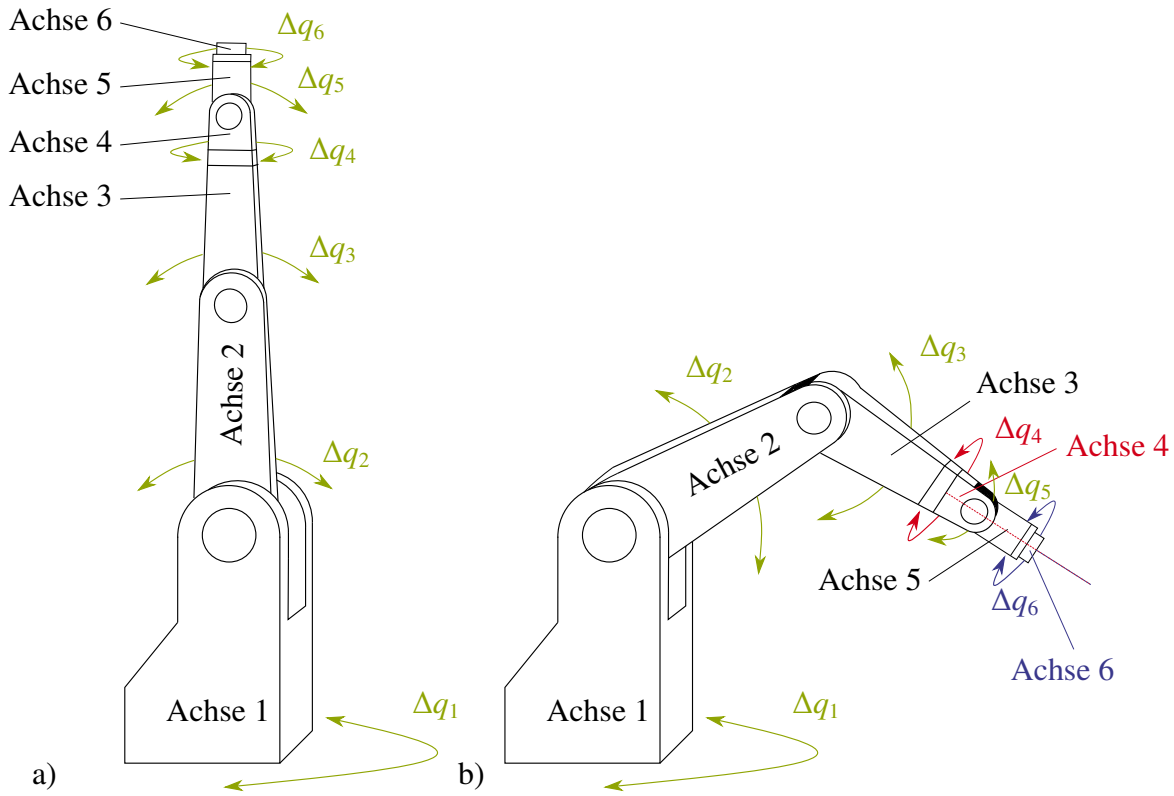


Bild 2-12: a) Grenzsingularität eines sechsachsigen SKM, b) interne Singularität, bei der die Achsen der Gelenke vier und sechs in einer Linie liegen

3 Stand der Wissenschaft und Technik

In diesem Kapitel wird zum einen ein Überblick über den aktuellen Stand der Wissenschaft und Technik hinsichtlich der Planung optimierter Trajektorien für kinematisch redundante Mechanismen gegeben. Zum anderen werden verschiedenen Ansätze für nicht-redundante Mechanismen erläutert. Des Weiteren erfolgt die Vorstellung und Analyse relevanter Quellen bezüglich echtzeitfähiger modellbasierter Optimierungs- und Regelungsansätze auf Industriehardware.

Die Quellen werden in Form einer Kurzzusammenfassung vorgestellt, gegebenenfalls vorhandene Querbeziehungen zwischen den Ansätzen aufgezeigt und die daraus gezogenen Schlüsse am Ende jedes Abschnitts als Zwischenfazit dargelegt. Die Reihenfolge der Quellen orientiert sich an dem Erscheinungsjahr sowie an inhaltlichen Zusammenhängen. Abgeschlossen wird das Kapitel mit einer Zusammenfassung und einem kurzen Fazit.

3.1 Optimierungsbasierte Trajektorienplanung für seriell-, parallel- und hybridkinematische kinematisch redundante Mechanismen

In diesem Abschnitt werden für die vorliegende Arbeit relevante Quellen bezüglich der Planung optimierter Trajektorien für hybridkinematische sowie allgemein kinematisch redundante Mechanismen vorgestellt. Dabei erfolgt keine Einschränkung hinsichtlich des Planungsziels (z.B. Zeitersparnis, Minimierung des Energiebedarfs etc.). Vielmehr ist es das Ziel einen möglichst umfassenden Überblick über Ansätze für die Trajektorienplanung bei redundanten Mechanismen zu geben. Zunächst werden die Veröffentlichungen präsentiert, die explizit redundante HKM adressieren.

Cha et al. stellen in [CLV06] einen Ansatz zur Nutzung kinematischer Redundanz bei einem redundanten HKM mittels lokaler Optimierung vor. Der Mechanismus besteht aus einem mehrachsigen SKM mit drei Freiheitsgraden, an dessen TCP zudem ein PKM in Form eines Hexapoden mit sechs Freiheitsgraden befestigt ist. Da der SKM sowohl mit seinem eigenen Gewicht als auch dem des PKM belastet wird, sollen durch Optimierung seine zum Abfahren einer gegebenen Trajektorie erforderlichen Bewegung minimiert werden. Dazu wird der Verfahrweg des SKM als Gütekriterium genutzt und in Nebenbedingungen die Einhaltung der Antriebs- und Gelenkgrenzen des PKM sichergestellt. Die dafür erforderlichen kinematischen Gleichungen des Mechanismus werden auf Basis geometrischer Zusammenhänge hergeleitet. Um zu vermeiden, dass die Antriebe des PKM die minimalen und maximalen Positionsgrenzen erreichen und durch den Optimierer dann starke Änderungen der einzelnen Antriebspositionen vorgegeben werden, werden die Positionsverläufe geglättet. Sobald ein oder mehrere Antriebe des PKM einen vorgegebenen minimalen oder maximalen Grenzwert der Position erreichen, wird durch einen zusätzlichen Faktor in der Gütefunktion ausnahmsweise nicht die Bewegung des SKM sondern die des PKM fokussiert und somit minimiert. Dadurch werden die erforderlichen Verfahrwege der Antriebe des PKM reduziert und diese daran gehindert ihre Positionsgrenzen zu erreichen. Die Reduzierung der Bewegung des PKM muss durch den SKM kompensiert werden. Eine simulative Validierung zeigt die Funktionsfähigkeit des Ansatzes, liefert aber

keine Aussagen zur erreichbaren Zykluszeit. Echtzeitfähigkeit ist hier kein betrachtetes Kriterium.

Viele weitere Ansätze beschäftigen sich mit der Optimierung von redundanten Mechanismen, ohne dabei explizit HKM zu adressieren. Da es nach der in Kapitel 2 gegebenen Definition jedoch viele Arten von HKM gibt, welche nicht zwangsläufig als solche benannt werden, und um zudem einen möglichst umfassenden Überblick über das Thema zu vermitteln, werden im Folgenden weitere relevante Veröffentlichungen bezüglich kinematisch redundanter Mechanismen ohne Einschränkung auf HKM vorgestellt.

Seereeram et al. stellen in [SW93] die Planung kontinuierlicher Gelenkbewegungen für kinematisch redundante Mechanismen als zeitlich begrenztes nichtlineares Regelungsproblem dar, welches in eine statische nichtlineare Wurzelsuche überführt und mittels des *Newton-Raphson-Verfahrens* (siehe [DH08, S. 97ff]) gelöst wird. Ziel ist es Pfade für die einzelnen Gelenke zu finden, sodass der TCP einem aufgabenbezogenen Pfad folgt. Dabei gilt es weniger einen optimalen denn einen möglichen Weg ohne Kollision mit Hindernissen zu ermitteln. Berücksichtigt werden die Hindernisse und Gelenkgrenzen als Ungleichheitsnebenbedingungen sowohl im Gelenkraum als auch im kartesischen Raum. Die Validierung erfolgt simulativ in *MATLAB* anhand verschiedener SKM. Es wird eine globale Optimierung durchgeführt, bei der die gesamte durchzuführende Aufgabe und damit die vollständige Trajektorie als vorab bekannt angesehen und als Gesamtes betrachtet wird.

Bei dem 1997 von Hirakawa et al. in [HK97] vorgestellten Ansatz für redundante SKM wird eine Optimierung genutzt, um die erforderliche Energie beim Abfahren einer vorgegebenen Trajektorie zu minimieren und gleichzeitig die Genauigkeit der TCP-Position zu gewährleisten. Hierzu werden unter Annahme der Gelenktrajektorien als B-Splines (siehe [PBP02]) die Summe der Energien der einzelnen Gelenke beim Abfahren der Trajektorie minimiert. Die freien Parameter der B-Splines werden mittels Optimierung derart gewählt, dass die Gütefunktion als gewichtete Summe der Differenz von Soll- und Istposition des TCP sowie der erforderlichen elektrischen Energie der Antriebe minimiert wird. Dabei befindet sich der Mechanismus sowohl in der Start- als auch in der Endposition in der Ruhelage. Die Antriebe werden als Gleichstrommotoren angenommen, mittels der auf der Jakobi-Matrix basierenden Dynamikgleichung des Roboters kann die Energie berechnet werden. Die Optimierung erfolgt mittels des Gradientenverfahrens, für eine Trajektorie mit zehn Positionen werden auf einem Computer mit einer 120 MHz-CPU laut den Autoren 7,51 min benötigt, Echtzeitfähigkeit ist kein betrachtetes Kriterium.

Von McAvoy et al. wird in [MSS00] ein ähnlich wie bei Hirakawa et al. auf B-Splines aufsetzender Ansatz vorgestellt. Jedoch erfolgt die Optimierung hier mittels genetischer Algorithmen, um eine energieoptimale und möglichst exakte Punkt-zu-Punkt-Bewegung von SKM im Rahmen von Pick and Place-Anwendungen zu erzielen. Dabei wird angenommen, dass Start- und Zielpose des Mechanismus bekannt sind. Die für das Verfahren erforderlichen Gelenkbewegungen werden als B-Splines vierter Ordnung angesehen. Um eine ideale Bewegung zwischen Start- und Zielpose zu erzielen, wird die Trajektorie des TCP als Gerade angenommen, welche in eine definierte Anzahl gleicher Abschnitte unterteilt wird.

Durch Nutzung genetischer Algorithmen werden verschiedene mögliche Gelenktrajektorien erzeugt und anhand der jeweiligen Ergebnisse einer Gewichtungsfunktion bewertet. Ziel ist es Gelenktrajektorien zu ermitteln, mittels der die Zielpose entlang der Geraden möglichst exakt erreicht wird, gleichzeitig aber auch die auf Basis der erforderlichen Antriebsmomente ermittelte erforderliche Energie minimiert wird. Die Validierung erfolgt simulativ anhand eines kinematisch redundanten planaren SKM mit drei Freiheitsgraden. Die Gewichtung zwischen der Abweichung und der Energie innerhalb der Gütefunktion hat dabei laut der Autoren einen entscheidenden Einfluss auf das Systemverhalten. So kann etwa eine stärkere Gewichtung der Energie dazu führen, dass der TCP die Gerade bei der Bewegung verlässt. Über die Rechenzeit bei der simulativen Validierung wird keine Angaben gemacht.

Von Basile et al. wird in [BC03] ein Offline-Ansatz vorgestellt, der die kinematische Redundanz von SKM nutzt, um gegebene Trajektorien möglichst schnell abzufahren. Basis bildet die gewichtete pseudoinverse Jakobi-Matrix des Mechanismus. Durch die Gewichtung innerhalb der Gütefunktion wird die Bewegung von Gelenken mit einem schlechteren Verhältnis von Trägheit zu Drehmoment stärker bestraft. Ziel ist es die Möglichkeiten des Mechanismus bezüglich Beschleunigung und Verzögerung möglichst weit auszunutzen. Dazu wird eine lineare statische Optimierung mit einer quadratischen Gütefunktion genutzt. Am Beispiel eines nicht weiter beschriebenen planaren Mechanismus mit drei Freiheitsgraden wird simulativ die Funktionsfähigkeit des Ansatzes nachgewiesen.

Ebenfalls anhand eines planaren Mechanismus mit drei Gelenken präsentieren Ma et al. in [MW04] einen Ansatz für kinematisch redundante Mechanismen, der die zeitoptimale Verfolgung vorgegebener Pfade ermöglicht. Dabei werden die Drehmomentgrenzen der einzelnen Gelenke berücksichtigt. Der Ansatz basiert auf der Unterteilung des Gelenkraums in redundante und nicht-redundante Koordinaten (engl. joint space decomposition), die Trajektorienplanung erfolgt unter Einsatz der *Phasenraumanalyse* (engl. Phase-plane analysis) und linearer Programmierung. Abschließend wird die simulative Validierung des Ansatzes anhand eines seriellkinematischen Mechanismus mit einem Grad der Redundanz (siehe 2, Abschnitt *Kinematische Redundanz*) von eins vorgestellt. Ergebnis ist, dass durch den Ansatz zwei der insgesamt drei Gelenke immer mit maximalem Drehmoment betrieben werden und somit die Geschwindigkeit der Pfadverfolgung gesteigert wird. Abhängig von der Initialstellung des Mechanismus sind mit dem Ansatz jedoch nicht alle Bereiche des Arbeitsraumes abgedeckt, sodass hier je nach Anwendung entsprechende Anpassungen vorgenommen werden müssen.

Ata et al. stellen in [AM06] einen Ansatz für das möglichst exakte Abfahren einer gegebenen Solltrajektorie vor. Dazu wird diese durch zusätzliche Stützpunkte in Abschnitte gleicher Länge unterteilt. Für jeden der erzeugten Stützpunkte werden der erwartete Positionsfehler, die erforderlichen Gelenkbewegungen sowie die Gelenk- und TCP-Geschwindigkeiten (die jeweils konstant sein sollen) berücksichtigt. Im Rahmen einer globalen Optimierung der Gelenkwinkel mittels genetischem Algorithmus (siehe etwa [BL04]) werden diese Werte für die Gütefunktion über alle Positionen gewichtet summiert. Die Optimierung erfolgt zweistufig, wobei die Ergebnisse des genetischen Algorithmus als Eingänge für einen *Pattern Search-Algorithmus* (siehe etwa [HJ61]) genutzt werden, der nur die Positionsabweichung berücksichtigt und somit die Genauigkeit des Mechanismus verbessern soll. Die simulative Validierung in *MATLAB* erfolgt anhand eines planaren SKM mit drei Gelenken.

In [CLV07a] stellen CHA ET AL. einen auf lokaler Optimierung basierenden Ansatz zur Vermeidung von Singularitäten bei einem kinematisch redundanten parallelkinematischen 3(P)RRR-Mechanismus [GA88] vor. Ziel ist es die Bewegung der Antriebe mittels Optimierung derart zu wählen, dass beim Abfahren einer vorgegebenen Trajektorie Singularitäten des Typs 2 (siehe Kapitel 2, Absatz *Singularitäten und* JAKOBI-*Matrix*) vermieden werden. Der Indikator für eine solche Singularität ist eine zu Null werdende Determinante der inversen JAKOBI-Matrix. Durch ein geeignetes Gütekriterium soll dies im Rahmen einer lokalen Optimierung vermieden werden. Die Validierung erfolgt simulativ anhand eines kinematisch redundanten 3-RPRR-Mechanismus. In [CLV07b] wird von gleichen Autoren der Ansatz zudem anhand eines 3-RRPR-Mechanismus sowie eines 3-RPRPR-Mechanismus simulativ erprobt.

Von EBRAHIMI ET AL. wird in [ECB07b] ebenfalls ein Ansatz zur Vermeidung von Singularitäten für kinematisch redundante planare PKM mittels lokaler Optimierung vorgestellt. Der Ansatz basiert auf der Ermittlung eines Faktors, des sogenannten *Normalised scaled incircle radius NSIR*. Dieser gibt an, wie nah der Mechanismus einer Singularität ist. Der Faktor dient als Gütekriterium und wird über geometrische Zusammenhänge (u.a. der Stellung der kinematischen Ketten zueinander) und die JAKOBI-Matrix ermittelt. Über einen nicht näher erläuterten Optimierungsansatz werden die Stellungen der linearen Achsen eines redundanten planaren 3(R)PRR-Mechanismus derart gewählt, dass der TCP einer vorgegebenen Trajektorie folgt und dabei Singularitäten vermieden beziehungsweise die Abstände zu Singularitäten maximiert werden. In [ECB07a] und [ECB08] wird der Ansatz durch die gleichen Autoren aufgegriffen und um ein zweites Gütekriterium erweitert. Dieses basiert auf der Kondition[2] der JAKOBI-Matrix als das Verhältnis zwischen dem größten und dem kleinsten Eigenwert der Singulärwertzerlegung. Je näher dieser Wert bei eins ist, desto näher ist die Genauigkeit der TCP-Position an der möglichen Genauigkeit der Antriebe. Die Ansätze werden simulativ validiert und anhand der Ergebnisse verglichen.

In [CEB08] stellen CARRETERO ET AL. einen globalen Ansatz zur Offline-Optimierung der Bewegung eines kinematisch redundanten planaren PKM beim Abfahren einer gegebenen Trajektorie vor. Dabei wird die gesamte Solltrajektorie mit dem Ziel betrachtet, die Gelenktrajektorien redundanter Antriebe derart zu wählen, dass das Bewegungsverhalten des Mechanismus etwa bezüglich der Vermeidung von Singularitäten optimiert wird. Mittels Lösung des IKP werden für jeden Antrieb alle möglichen Positionen, mit denen der TCP noch der Solltrajektorie folgt, ermittelt. Diese möglichen Trajektorien der Antriebe werden durch Hilfspunkte unterteilt und als Polylinien[3] nachgebildet. Die Lage der Hilfspunkte und damit die Trajektorien der Antriebe gilt es nun im Sinne des definierten Ziels zu optimieren. Dabei muss die Anzahl der eingesetzten Hilfspunkte gut gewählt werden, zu wenige verschlechtern das Ergebnis, zu viele erhöhen den Rechenaufwand stark. Werden optimierte Hilfspunkte gefunden, werden mit ihnen die Polylinien der Antriebe durch Polynome höherer Ordnung ersetzt. Die simulative Validierung erfolgt anhand eines 3-RPRR-Mechanismus unter Nutzung des NSIR-Ansatzes aus [ECB07b] in *MATLAB*. Der gleiche Ansatz, ebenfalls mit dem NSIR-Gütefunktional zur Vermeidung von Singularitä-

[2] Die Kondition der JAKOBI-Matrix gibt an, welchen Einfluss Gelenkfehler auf die Position des TCP haben. In diesem Zusammenhang wird auch von einem Fehler-Verstärkungsfaktor gesprochen.[Mer07]

[3] Der Begriff Polylinie wird im Bereich von Computer-Aided Design (CAD) verwendetet. Er bezeichnet eine beliebige Folge von zusammenhängenden und als zusammengehörendes Objekt angesehene Linien- und Kreisbogensegmente [Har98].

ten, wird von CARRETERO ET AL. auch in [CEB12] vorgestellt. Ergänzt wird hier angegeben, dass die Optimierung auf einem Computer mit einer 2,66 GHz-CPU bei 21 Hilfspunkten ungefähr 20 s benötigt.

ALBA-GÓMEZ ET AL. stellen in [APW08] einen Ansatz zur Trajektorienplanung von PKM in Anwendungen vor, in denen die Orientierung des TCP vernachlässigt werden kann. Dadurch sind diese Mechanismen bezogen auf ihre Anwendung redundant. Diese Umstand wird genutzt, um auf Basis einer globalen Optimierung die Kondition der JAKOBI-Matrix (vgl. [ECB07a] und [ECB08]) zu verbessern und dadurch gegebene Trajektorien möglichst genau abzufahren. Aufsetzend auf sogenannten *feasibility maps* [WCR93], die für den gesamten Pfad die Kondition für alle möglichen Orientierungen des TCP enthalten, wird mittels der Optimierung die Orientierung des TCP derart gewählt, dass die Kondition der JAKOBI-Matrix möglichst optimal wird. Gleichzeitig werden Begrenzungen wie die maximale Änderungsrate der Orientierung berücksichtigt. Es wird dabei angenommen, dass für den Mechanismus durchgehend der gleiche Arbeitsmodus[4] (engl. working mode) genutzt wird. Wie in [APW07] dargestellt, ist jedoch auch ein Wechsel des Arbeitsmodus möglich, etwa wenn aufgrund von Singularitäten eine vorgegebene Trajektorie ansonsten nicht abgefahren werden kann. Die Validierung erfolgt anhand eines planaren 3-RRR-Mechanismus, wobei keine Angaben zur verwendeten Rechenhardware gemacht werden. In [RPW16] werden die Ansätze aus [APW08] und [APW07] von REVELES ET AL. aufgegriffen und erweitert. Hintergrund ist, dass das Umschalten zwischen Arbeitsmodi am realen Mechanismus zu Vibrationen führt, die es zu vermeiden gilt.

Von KOTLARSKI ET AL. wird in [KTHO10] ein Ansatz zur optimierten Ausnutzung von kinematischer Redundanz bei PKM bezogen auf die Vermeidung von Singularitäten und die gleichzeitige Verbesserung der Positioniergenauigkeit vorgestellt. Ausgangspunkt ist eine durch Singularitäten führende Trajektorie. Da die Positioniergenauigkeit in den Bereichen um die Singularitäten sinkt, gilt es diese zu vermeiden. Durch eine geeignete Positionierung zusätzlicher redundanter Achsen soll der Mechanismus derart umkonfiguriert werden, dass die Singularitäten in Bereiche außerhalb der vorgegebenen Trajektorie verschoben werden. Als Optimierungskriterium wird dabei die Minimierung des Positionsfehlers genutzt, welcher approximiert wird durch die auf der JAKOBI-Matrix basierenden Geschwindigkeitsgleichung des Mechanismus. Hinsichtlich der lokalen ableitungsfreien Optimierung werden fünf verschiedene Strategien vorgestellt: einmalige Änderung vor Durchführung der geplanten Bewegung, diskrete bzw. abschnittsweise Optimierung (siehe auch [KAH08]) während der Bewegung, bedarfsabhängige diskrete bzw. abschnittsweise Optimierung während der Bewegung, kontinuierliche Optimierung und semi-kontinuierliche Optimierung. Am Beispiel eines redundanten 3-(P)RRR-Mechanismus [GA88] werden die Strategien simulativ miteinander verglichen. In [KHO11] werden diese Ansätze von KOTLARSKI ET AL. wieder aufgegriffen und an einem realen Aufbau des redundanten 3-(P)RRR-Mechanismus experimentell validiert. Für die Regelung wird ein leistungsstarkes Linux-basiertes System eingesetzt, welches aus *MATLAB* heraus programmiert wird. Die erforderlichen Rechenzeiten werden nicht genannt.

In [AIS11] stellen AYTEN ET AL. einen Ansatz für energieoptimierte/drehmomentreduzierte Punkt-zu-Punkt-Bewegungen von SKM vor. Dabei werden die Trajektorien als B-Splines fünfter Ordnung angenommen. Von den neun Parametern dieser Funktion geben jeweils

[4] Ein sogenannter working mode bezeichnet eine der verschiedenen möglichen Lösungen der inversen kinematischen Gleichung [RPW16]

drei den Start- beziehungsweise den Endzustand an, die anderen drei können im Rahmen einer Optimierung bestimmt werden. Die für diese Optimierung erforderliche Gütefunktion berücksichtigt die Summe der Antriebsdrehmomente sowie die Systemgrenzen, wobei die Drehmomente auf Basis der dynamischen Gleichungen des Mechanismus berechnet werden. Die simulative Validierung erfolgt anhand eines SKM mit zwei Freiheitsgraden. Ayten greift diesen Ansatz in [Ayt12] erneut auf und ergänzt ihn um verschiedene Elemente zur Verkürzung der Rechenzeit. So wird unter anderem das dynamische Verhalten nur dann berechnet, wenn alle auf der inversen kinematischen Gleichung basierenden Nebenbedingungen (zulässige Gelenkwinkel u.ä.) erfüllt sind. Der resultierende Ansatz wird an zwei redundanten SKM eingesetzt. Dabei wird die Optimierung in *MATLAB* durchgeführt, die berechneten Trajektorien werden entweder anhand detaillierter Modelle oder realer Mechanismen ohne nähere Spezifikation validiert.

Einen weiteren Ansatz zur Bewegungsplanung kinematisch redundanter PKM stellen Niemann et al. in [NKOM13] vor. Ziel ist es, das Potenzial redundanter PKM unter Echtzeitbedingungen zu nutzen und dafür den Rechenaufwand für ein zugrundeliegendes Optimierungsproblem zu reduzieren. Wie Echtzeit im Kontext der Veröffentlichung verstanden wird, ist nicht näher beschrieben. Im vorgestellten Ansatz gilt es den Abstand zu Singularitäten des Typs 2 zu maximieren und diese Singularitäten damit zu vermeiden. Die Aufgabe des Optimierers besteht darin, die Sollpositionen zusätzlicher linearer Antriebe derart zu wählen, dass die Kondition der homogenisierten Jakobi-Matrix (vgl. [ECB07a], [ECB08], [APW08]) optimiert wird. Um die Rechenzeit zu reduzieren, wird offline eine systematische Reduzierung der erforderlichen Optimierungspunkte, etwa Postionen des TCP und zugehöriger Antriebspositionen, durchgeführt. Dazu werden um die Punkte (kreisförmige) Bereiche definiert, wobei sich jeweils alle Punkte innerhalb eines Kreises bezüglich des Optimierungskriteriums nahezu gleich verhalten. Ziel ist es diese Bereiche so groß wie möglich zu wählen und zudem ihre Anzahl derart zu reduzieren, dass lediglich der Arbeitsraum vollständig abgedeckt ist. Zur Laufzeit werden die Bereiche dann so weit verringert, dass nicht mehr der gesamte Arbeitsraum sondern nur die Solltrajektorie abgedeckt wird. Dadurch wird der Suchbereich des Optimierers sowie infolgedessen die Optimierungszeit reduziert. Anhand eines planaren Mechanismus (3-(P)RRR, [GA88]) mit wahlweise einem, zwei oder drei zusätzlichen redundanten Linearantrieben erfolgt die simulative Validierung. Auf einem Computer mit einer 2,80 GHz-CPU werden je nach Anzahl der redundanten Linearantriebe zwischen 46,19 ms und 90,10 ms je Optimierungsproblem benötigt.

In [VS14] und [VS15] stellen Varalakshm et al. einen auf lokaler Optimierung basierenden Ansatz zur Reduzierung der Antriebsdrehmomente von kinematisch redundanten planaren PKM während der Verfolgung gegebener Trajektorien vor. Durch Nutzung der Jakobi-Matrix können die Antriebsmomente in jedem Punkt ermittelt und im Rahmen der Gütefunktion genutzt werden. Die Optimierung erfolgt mittels binär codierter genetischer Algorithmen mit dem Ziel, die Position eines zusätzlichen linearen Antriebs derart zu wählen, dass eine vorgegebene Trajektorie mit möglichst geringen Antriebsdrehmomenten abgefahren wird. Die Validierung erfolgt simulativ anhand eines redundanten 3-(P)RRR-Mechanismus.

Fontes et al. präsentieren in [FSd14a] ebenfalls einen Ansatz, um kinematische Redundanz bei PKM zur Reduzierung der erforderlichen Antriebsdrehmomente zu nutzen. Als Gütekriterium wird das maximale beim Abfahren einer gegebenen Trajektorie auftretende

Antriebsdrehmoment verwendet. Die grundlegende Idee, die auch von SANTOS ET AL. in [SRD14] präsentiert wird, basiert auf der Minimierung der Gütefunktion durch die Positionierung zusätzlicher redundanter Linearachsen. Die Trajektorien der Linearachsen werden als Polynom fünfter Ordnung angenommen und drei mögliche Wege zur Ermittlung von optimierten Parametern angegeben. Eine Möglichkeit besteht darin, die bestmöglichen Positionen für die Linearachsen offline zu berechnen (siehe auch [FSd14b]). Für eine Online-Berechnung als zweite Variante können im einfachen Fall Start- und Zielposition der Linearachsen als Optimierungsgrößen angesehen werden. Dabei wird angenommen, dass die Trajektorie in Form des Polynoms eine Dauer von 2 s aufweist und die Achsen an ihren Start- und Zielposition in Ruhe sind. Ist eine komplexere Online-Berechnung erforderlich, so können neben den Start- und Zielpositionen die initialen und finalen Beschleunigungen bzw. Verzögerungen ebenfalls optimiert werden. Die simulative Validierung der Ansätze erfolgt anhand von drei planaren redundanten Mechanismen ((P)RRR+2-RRR, 2-(P)RRR+RRR und 3-(P)RRR) im Vergleich mit einem nichtredundanten planaren Mechanismus (3-RRR). In [Fd16] wird der komplexe Online-Ansatz für weitere redundante PKM simulativ in *MATLAB* erprobt und die Ergebnissen mit dem Verhalten antriebsredundanter Mechanismen verglichen.

REITER ET AL. stellen in [RSGM15] aufsetzend auf [MW04] einen Ansatz für die Planung zeitoptimaler Gelenktrajektorien für kinematisch redundante SKM beim Abfahren gegebener Endeffektortrajektorien vor. Hierbei wird die sogenannte *joint space decomposition* genutzt, wobei der Gesamtmechanismus hinsichtlich seiner Antriebe in einen nicht-redundanten und einen redundanten Teilmechanismus unterteilt wird (siehe auch [Wam87]). Mittels eines Optimierungsansatzes erfolgt die Festlegung der Bewegung für die *redundanten* Achsen im Sinne zeitoptimaler Bewegungen, wodurch das inverse kinematische Problem lösbar wird. Um die Kontinuität[5] der Trajektorien zu erhöhen und damit die Zahl an möglichen Einsatzbereichen zu vergrößern werden diese mittels B-Splines abgebildet. Neben der Erhöhung der Kontinuität resultiert aus dem Einsatz der B-Splines ein Optimierungsproblem, das mittels einer *Active Set-Methode* (siehe z.B. [NW06]) gelöst werden kann. Im Rahmen der Veröffentlichung erfolgt die simulative Validierung in *MATLAB* anhand eines planaren Mechanismus mit drei Gelenken, der durch Vernachlässigung der Orientierung des TCP anwendungsspezifisch redundant ist.

In [RMG16] präsentieren REITER ET AL. zwei unterschiedliche Ansätze für die zeitoptimale Verfolgung gegebener Pfade für SKM mit zwei oder mehr kinematisch redundanten Freiheitsgraden. Das erste Verfahren basiert ähnlich wie in [MW04] und [RSGM15] auf einer Unterteilung des Gelenkraumes in redundante und nicht-redundante Koordinaten *joint space decomposition*). Im zweiten Ansatz wird die Redundanz basierend auf den differentiellen kinematischen Gleichungen, welche den Zusammenhang zwischen den Gelenkgeschwindigkeiten und den linearen sowie rotatorischen Geschwindigkeiten des TCP angeben [SSVO09], durch Betrachtung der Gelenkbeschleunigungen im Nullraum genutzt. Beide Verfahren nutzen die *Mehrzielmethode* (engl. direct multiple shooting method, siehe z.B. [SB02]) und werden anhand eines auf einer zusätzlichen seriellen Achse montierten Industrieroboters mit sechs Freiheitsgraden miteinander verglichen. Da die angenommene Anwendung nur fünf Freiheitsgrade erfordert, liegen der Grad der anwendungsspezifischen Redundanz bei zwei. Für die Ansätze werden bei der Berechnung

[5] Die Kontinuität gibt an, bis zu welchen Grad die resultierende Trajektorie ableitbar ist. Die Konitinuität C^3 bedeutet somit eine dreifache Ableitbarkeit.

auf einem Computer mit einer 3,50 GHz-CPU 680 s beziehungsweise 968 s benötigt, wobei sich die erreichten Verfahrzeiten der beiden Ansätze nur um 0,16 % bezogen auf die längere Verfahrzeit unterscheiden. Die Erweiterung um Ansätze zur Berechnung des IKP höherer Ordnung mittels *joint space decomposition* sowie Nutzung des Nullraums stellen Reiter et al. in [RMG18] vor. Die Lösung des nichtlinearen Optimierungsproblems erfolgt mittels der *Ipopt*-Toolbox [WB06] (vgl. Abschnitt 6.4.1). Die angegebenen Rechenzeiten liegen je nach Ansatz und gewünschter Kontinuität der resultierenden Gelenktrajektorien bei minimal 26 s (planarer redundanter SCARA) und 291,5 s (redundanter SKM bestehende aus Sechsachs-Knickarmroboter mit zusätzlicher serieller Achse). Neben der Rechenzeit steigt auch die erforderliche Verfahrzeit des jeweiligen Mechanismus mit Erhöhung der gewünschten Kontinuität.

Zwischenfazit

Zusammenfassend ist an dieser Stelle festzuhalten, dass der Einsatz redundanter Mechanismen und ihre Regelung bzw. die Nutzung der Redundanz zur Beeinflussung des (Bewegungs-) Verhaltens schon viele Jahre Gegenstand der Forschung ist. Wenige Autoren wie Cha et al. [CLV06] adressieren dabei explizit HKM. Die Quellen aus den letzten Jahren zeigen jedoch, dass allgemein im Bereich redundanter Mechanismen noch Bedarf an neuen Ansätzen besteht. Unterschiede gibt es beispielsweise hinsichtlich des Optimierungskriteriums: Während sich Hirakawa et al. [HK97], McAvoid et al. [MSS00] oder Ayten et al. [AIS11; Ayt12] mit der Reduzierung der erforderlichen Energie befassen und Varalakshmi et al. [VS14; VS15] sowie Fontes et al. [FSd14a; Fd16] mit der Reduzierung der Antriebsmomente ein ähnliches Ziel fokussieren, setzen die Ansätze von Cha et al. [CLV07a], Ebrahimi et al. ([ECB07b; ECB07a; ECB08]), Alba-Gómez et al. [APW08], Carretero et al. [CEB08; CEB12; CEB12], Kotlarski et al. [KTHO10; KHO11], Niemann et al. [NKOM13] oder Reveles et al. [RPW16] auf die Nutzung von Redundanz zur Vermeidung von Singularitäten. Weitere Optimierungskriterien sind etwa die Erhöhung der Positioniergenauigkeit ([AM06; ECB07a; ECB08; APW07; APW08; KTHO10; KHO11; RPW16]) oder die im Rahmen der vorliegenden Dissertation fokussierte Optimierung der Verfahrzeit ([BC03; MW04; RSGM15; RMG16; RMG18]).

Der erforderliche Rechenaufwand wird nur für einzelne Ansätze angegeben, dabei erfüllt keiner die im Rahmen der vorliegenden Arbeit definierte Echtzeitfähigkeit mit einer Zykluszeit kleiner 1 ms. Vielmehr erfolgt häufig eine simulative Umsetzung etwa in *MATLAB* ([AM06; ECB07a; ECB08; CEB08; RSGM15; Fd16]). Werden Rechenzeiten angegeben, beziehen sich diese nicht auf Industriesteuerungen sondern in der Regel auf performante Computersysteme, zudem liegen die Zeiten in Bereichen von einigen zehn Millisekunden [NKOM13] bis hin zu mehreren Sekunden [CEB12] oder Minuten ([HK97; RMG16; RMG18]).

Des weiteren ist festzuhalten, dass im Rahmen der vorgestellten Veröffentlichungen unterschiedlich geartete Modelle des jeweiligen Mechanismus eingesetzt werden. Während beispielsweise in [HK97; APW08] oder [KTHO10] die (pseudoinverse) Jakobi-Matrix Anwendung findet, nutzen Ebrahimi et al. in [ECB07b] geometrische Zusammenhänge wie die Stellung der kinematischen Ketten zueinander. Von Reiter et al. wird in [RMG16] hingegen die Nutzung der differentiellen kinematischen Gleichungen des Mechanismus genannt.

Tabelle 3-1 fasst die vorgestellten Quellen sortiert nach dem adressierten Mechanismus zusammen.

Tabelle 3-1: *Zusammenfassung der Quellen zur optimierungsbasierten Trajektorienplanung für kinematisch redundante SKM, PKM und HKM*

Quelle	Optimierungsziel	Mechanismus	Bewegungsart	Rechenhardware	Rechenzeit
[CLV06]	Bewegungsminimierung	HKM	CP	Simulation	-
[SW93]	variabel, Kollisionsvermeidung	SKM	CP	*MATLAB*	-
[HK97]	Energieminimierung	SKM	PTP	Computer	7,51 min für 10 Pos.
[MSS00]	Genauigkeit, Energieminimierung	SKM	PTP	Simulation	0 s
[BC03]	Zeitminimierung	SKM	CP	Simulation	-
[MW04]	Zeitminimierung	SKM	CP	Simulation	-
[AM06]	Genauigkeit	SKM	PTP	*MATLAB*	-
[AIS11]	Energieminimierung	SKM	PTP	Simulation	-
[Ayt12]	Energieminimierung	SKM	PTP	*MATLAB*	-
[RSGM15]	Zeitminimierung	SKM	CP	*MATLAB*	-
[RMG16]	Zeitminimierung	SKM	CP	Computer	680 s / 968 s
[RMG18]	Zeitminimierung	SKM	CP	Computer	26 s / 291,5 s
[CLV07a; CLV07b]	Singularitätsvermeidung	PKM	-	Simulation	-
[ECB07b; ECB07a; ECB08]	Singularitätsvermeidung, Genauigkeit	PKM	CP	Simulation	-
[APW07; APW08]	Singularitätsvermeidung, Genauigkeit	PKM	CP	Simulation	-

[CEB08]	Singularitäts-vermeidung	PKM	CP	Simulation	-
[KTHO10]	Singularitäts-vermeidung, Genauigkeit	PKM	CP	Simulation	-
[KHO11]	Singularitäts-vermeidung, Genauigkeit	PKM	CP	Linux-System, *MATLAB*	-
[CEB12]	Singularitäts-vermeidung	PKM	CP	Computer	20 s bei 21 Hilfspos.
[NKOM13]	Singularitäts-vermeidung	PKM	CP	*MATLAB*	46,19 ms / 90,10 ms
[VS14; VS15]	Drehmoment-minimierung	PKM	CP	Simulation	-
[FSd14a]	Drehmoment-minimierung	PKM	PTP, CP	Simulation	-
[Fd16]	Drehmoment-minimierung	PKM	PTP, CP	*MATLAB*	-
[RPW16]	Singularitäts-vermeidung, Genauigkeit	PKM	CP	Simulation	-

Das in vielen Veröffentlichungen aufgezeigte Potential der gezielten Beeinflussung des Bewegungsverhaltens eines redundanten Mechanismus gilt es aufzugreifen und derart zu adaptieren, dass eine echtzeitfähige Ausführung auf industrieller Hardware ermöglicht wird. Zudem muss der Ansatz derart gestaltet werden, dass nicht alle zukünftigen Sollpositionen oder die abzufahrende Trajektorie für eine Offline-Optimierung ([BC03], [CEB08]) oder einen Ansatz mit Betrachtung des gesamten Pfads ([SW93], [CEB08], [APW08]) a priori bekannt sein müssen. Vielmehr sind insbesondere hinsichtlich der immer flexibler und dynamischer werdenden Fertigung im Kontext von Industrie 4.0 und Losgröße-1 nur wenige zukünftige Sollpositionen für den Mechanismus bekannt, die zudem einer kurzfristigen Änderung unterliegen können. Hierauf muss zur Laufzeit des Systems mittels der Optimierung reagiert werden. Grundlage für einen solchen Ansatz bildet ein Modell des Mechanismus zur Abbildung des relevanten Verhaltens, es wird im Folgenden deshalb von einem *modellbasierten Optimierungsansatz* gesprochen.

3.2 Optimierungsstrategien für das Zeitverhalten nicht-redundanter Mechanismen

Um neben den spezifischen Quellen hinsichtlich redundanter Mechanismen und ihrer Trajektorienplanung auch einen Überblick über Strategien für nicht-redundante Mechanismen zu erlangen, werden in diesem Abschnitt einzelne relevante Quellen hierzu vorgestellt. Aufgrund der Vielzahl an Ansätzen und Veröffentlichungen zu diesen Mechanismen er-

folgt eine Einschränkung auf Ansätze, die das Zeitverhalten der Mechanismen optimieren und somit ein vergleichbares Ziel wie die vorliegende Arbeit aufweisen. Zudem werden Ansätze betrachtet, die entweder den hier betrachteten Veröffentlichungen hinsichtlich der Optimierung des Zeitverhaltens zugrunde liegen oder auf diesen aufbauen.

Entsprechende Forschungen hinsichtlich der Optimierung des Zeitverhaltens nicht-redundanter Mechanismen werden schon viele Jahre betrieben. So stellen etwa im Jahr 1985 Bobrow et al. in [BDG85] einen Ansatz zur zeitoptimalen Regelung von SKM vor. Hier wird insbesondere die eingeschränkte Dynamik der zu dieser Zeit verfügbaren Mechanismen adressiert, deren Auswirkungen auf den produktiven Einsatz beim Abfahren gegebener Trajektorien des TCP reduziert werden sollen. Hierzu werden die Verfahrstrecke und die Geschwindigkeit als Zustandsvektoren angesehen, die nichtlineare Dynamik des Mechanismus sowie die Begrenzungen der Aktoren werden in zustandsabhängige Begrenzungen überführt. Ziel des Ansatzes ist es die Beschleunigungsprofile derart zu wählen, dass daraus ein Geschwindigkeitsprofil für die Antriebe entsteht, welches jederzeit innerhalb der stellbaren Geschwindigkeiten der Antriebe liegt. Dadurch kann der TCP zu jedem Zeitpunkt durch die Antriebe auf der vorgegebenen Bahn gehalten werden. Die Grundlagen dieses Ansatzes wurden bereits 1982 von Bobrow in seiner Dissertation [Bob82] vorgestellt und in [BDG83] von Bobrow et al. weiter ausgeführt.

Von Haschke et al. wird in [HWR08] ein als online-fähig benanntes Verfahren zur Planung zeitoptimaler und ruckbegrenzter Bewegungen eines SKM vorgestellt. Dabei wird die Bewegung des TCP zwischen zwei vorgegebenen Sollpositionen zunächst mittels trapezförmiger Geschwindigkeitsverläufe (Polynom zweiter Ordnung) für alle Freiheitsgrade approximiert. Somit besteht der Geschwindigkeitsverlauf aus drei Phasen: Beschleunigung, konstante Geschwindigkeit und Verzögerung. Ziel ist es, dass der Antrieb mit der langsamsten Verfahrzeit seine maximalen Werte für Beschleunigung und Verzögerung nutzt und bei ausreichend langer Verfahrstrecke mit seiner maximalen Geschwindigkeit verfährt. Die maximalen Geschwindigkeiten der anderen Achsen werden in Abhängigkeit des Geschwindigkeitsverlaufs mit der längsten Dauer angepasst, sodass die Sollposition in alle DOF gleichzeitig erreicht wird. Die initialen Werte etwa für die Geschwindigkeit entsprechen den Werten an der Zielposition der vorhergegangenen Bewegung. Wird vorgegeben, dass die Werte an Start- und Zielposition gleich null sein sollen, liegt eine PTP-Bewegung vor, ansonsten können kontinuierliche Trajektorien abgebildet werden. Ein sehr ähnliches Vorgehen wird von Kröger et al. in [KTW06] vorgestellt. Von Haschke et al. wird dieser Ansatz noch erweitert auf den Einsatz von Polynomen dritter Ordnung, um den Ruck ebenfalls zu berücksichtigen und damit die Bewegungen zu glätten. Validiert wird der Ansatz am Beispiel eines SKM mit sieben DOF, wobei die Generierung der Sollgeschwindigkeitsverläufe auf einem Computer mit einer 3,00 GHz-CPU erfolgt und maximal 0,36 ms benötigt. Diese werden an die Robotersteuerung übergeben, welche die Ermittlung der erforderlichen Gelenkpositionen übernimmt.

van den Broeck et al. präsentieren in [vDS09] einen zweistufigen Ansatz für die Ermittlung zeitoptimaler Punkt-zu-Punkt-Bewegungen allgemeiner *mechatronischer Systeme*. Da hier zeitoptimale Bewegungen adressiert werden, wird die Quelle trotz des fehlenden expliziten Bezugs auf SKM, PKM oder HKM vorgestellt. Im Rahmen des Ansatzes wird zunächst die Anzahl an Zeitschritten N, die das System benötigt, um an einer Sollposition zum Stillstand zu kommen, minimiert. Diese *gemischt-ganzzahlige Optimierung* (engl. auch mixed-integer program, siehe z.B. [Kal13]) mit dem Integer N wird als eine Serie von

quadratischen Programmen mittels einer *Active Set-Methode* (siehe z.B. [NW06]) gelöst. Wird eine vorab definierte minimale Anzahl an erforderlichen Schritten erreicht, das System befindet sich also kurz vor oder bereits am Sollpunkt, wird eine traditionelle modellprädiktive Regelung (MPR) eingesetzt. Diese berücksichtigt neben dem Regelfehler auch Änderungen der Stellgrößen, um die Sollposition schnellstmöglich zu erreichen, dabei aber ein unruhiges Verhalten an der Sollposition zu vermeiden. Für das zu regelnde System wird ein zeitdiskretes Zustandsraummodell höherer Ordnung benötigt. Bei dem in [vDS09] vorgestellten 2DOF-Feder-Masse-Dämpfer-System wird ein Modell fünfter Ordnung genutzt, die Berechnung des Ansatzes in *MATLAB* auf einem Computer mit einer 2,00 GHz-CPU benötigt maximal 7 ms. In [vDS11a] wird der Ansatz auf zwei weitere Systeme angewandt: einen Linearmotor, der ein zeitdiskretes Zustandsraummodell dritter Ordnung benötigt sowie einen Brückenkran mit einem Modell vierter Ordnung. Bei dem ersten System wird ein FPGA-basiertes echtzeitfähiges und *Simulink*-basiertes Testsystem namens *SpeedGoat* eingesetzt, mit dem beim vorliegenden Anwendungsfall Abtastzeiten von minimal 5 ms erreicht werden können. Für das zweite System wurde der Ansatz auf einem ebenfalls aus *Simulink* heraus programmierten *dSPACE*-System implementiert. Hier sind Abtastzeiten von minimal 17 ms möglichen. Diese Ergebnisse werden auch in [vDS11b] für das selbe System vorgestellt.

Wang et al. setzen in [WSSP12] auf dem zeitoptimalen Ansatz von van den Broeck et al. auf und überführen diesen in einen energieoptimalen Ansatz für lineare zeitinvariante Systeme. Dazu wird angenommen, dass das System einen Sollpunkt im Rahmen von Punkt-zu-Punkt-Bewegungen nach einer definierten Anzahl an Zeitschritten erreichen soll. Ist dies möglich, werden gegebenenfalls noch vorhandene zeitliche Freiheiten genutzt, um die Energieeffizienz zu optimieren. Dazu wird die aus dem zeitoptimalen Ansatz resultierende Gütefunktion durch eine Funktion zur Berücksichtigung der Energieverluste der Antriebe ersetzt. Durch Annahme eines ideal geschwindigkeitsgeregelten Antriebs kann dieser als Integrator angesehen werden. Am Beispiel des seriellkinematischen Mechanismus Badminton Robot auf einer nicht näher spezifizierten Rechenhardware werden mit dem Ansatz Zykluszeiten von minimal 10 ms erzielt. Eine Erweiterung des Ansatzes um die Berücksichtigung von nicht modellierten Störungen und Modellungenauigkeiten zur Verbesserung der Positioniergenauigkeit wird in [WS15] vorgestellt, bei gleicher Regelstrecke ändernt sich die Performanz der Regelung nicht.

Von Rymansaib et al. wird in [RIS13] ein Ansatz vorgestellt, um Sollpositionen möglichst schnell abzufahren. Gleichzeitig werden aber Beschleunigungen und Ruck des TCP begrenzt, um die Positioniergenauigkeit von SKM bei Punkt-zu-Punkt-Bewegungen zu verbessern. Dabei werden die Verfahrwege zwischen zwei Positionen als Geraden angenommen. Während andere Verfahren für zeitoptimale Punkt-zu-Punkt-Bewegungen häufig auf trapezförmigen Geschwindigkeitsverläufen mit entsprechend starken Änderungen der Beschleunigung setzen, werden die Geschwindigkeitsverläufen hier als Polynom dritter Ordnung definiert. Durch Einführung eines Skalierungsfaktors für die Verfahrzeit können die Steigung der Geschwindigkeit und damit die Beschleunigungen verändert werden. Durch entsprechende Wahl des Faktors wird sichergestellt, dass Beschleunigung und Ruck immer in definierten Grenzen liegen, diese Grenzen jedoch erreicht werden, um eine möglichst kurze Verfahrzeit zu erzielen. Mittels des Ansatzes werden in *MATLAB* für jede lineare Bewegung der Skalierungsfaktor für die Verfahrzeit sowie die maximal mögliche Geschwindigkeit berechnet. Zudem kann festgelegt werden, ob der TCP an

jeder Position zum Stillstand kommen soll oder die Positionen überschliffen[6] werden. Über eine Schnittstelle zwischen *MATLAB* und einem SKM werden darauf basierend Geschwindigkeitsbefehle an den SKM ausgegeben, die zugehörige Steuerung übernimmt die Berechnung der Gelenkwinkel mittels des IKP. Wird kein Überschleifen genutzt, erhöht der Ansatz laut den Autoren die Genauigkeit beim Anfahren der Sollpositionen. Gegenüber Ansätzen mit trapezförmigen Geschwindigkeitsverläufen werden die Belastungen der Antriebe reduziert, jedoch ist die erforderliche Verfahrzeit länger.

In [GRN13] stellen Gattringer et al. verschiedene Ansätze für die Optimierung des Bewegungsverhaltens von SKM bezüglich Applikationen mit hohen Geschwindigkeitsanforderungen vor. Dabei wird unter anderem auch die optimierte Trajektorienplanung betrachtet. Konkret wird unterschieden zwischen der Pfadverfolgung bei bestehenden Trajektorien und der Ermittlung zeit- oder energieoptimaler Punkt-zu-Punkt-Trajektorien. Im Rahmen dieser Zusammenfassung des Stands der Technik und Wissenschaft wird nur Letzteres betrachtet. Vorgestellt wird ein Ansatz zur Offline-Berechnung basierend auf der dynamischen Gleichung des Roboters. Die verwendete Gütefunktion hängt von den die Dynamik beschreibenden Differentialgleichungen ab. Unter Beachtung der Grenzen des maximalen Drehmoments sowie den maximalen Geschwindigkeiten und Beschleunigungen der einzelnen Gelenke wird offline mittels des *(direktes) Mehrfach-Schießverfahrens* (engl. direct multiple shooting method, ein numerisches Verfahren zur Lösung von Randwertproblemen siehe z.B. [SB00], [Rei12]) eine Optimierung der Verfahrzeit durchgeführt, um die erforderlichen Gelenkpositionsverläufe zu ermitteln. Zudem werden zusätzliche Drehmomente für die Gelenke berechnet, die im Rahmen einer Vorsteuerung zur Verbesserung der Positioniergenauigkeit verwendet werden. Die Berechnung einer solchen Punkt-zu-Punkt-Trajektorie dauert für einen Sechsachs-Roboter auf einem nicht näher spezifizierten Computer näherungsweise 3 min.

Von Springer et al. wird in [SGS14] eine weitere Methode zur Planung nahezu zeitoptimaler Punkt-zu-Punkt-Trajektorien für serielle Mechanismen vorgestellt. Dabei werden offline berechnete zeitoptimale Trajektorien, sogenannte *point-to-point time-optimal trajectories PTPTO* (siehe [GRN13]), mittels Lernalgorithmen in sogenannte *dynamic movement primitives* (siehe z.B. [Sch06]) überführt und eine Trajektorie durch eine Anzahl an Differentialgleichungen dargestellt. Durch geeignete Formulierung der Gleichungen kann eine explizite Zeitabhängigkeit vermieden werden, sodass letztendlich die Sollposition innerhalb gewisser Bereiche verschoben werden kann. Somit ermöglicht dieser Ansatz die Adaption bestehender Pfade auf neue Sollpositionen innerhalb eines festzulegenden Bereichs um die Sollposition der offline berechneten Referenztrajektorie. In [SGS14] wird ein kubischer Bereich von 10 cm Kantenlänge angenommen. Die Ermittlung der hier zeitoptimal ausgelegten Referenztrajektorie muss für jeden Bereich nur einmalig vorab offline durchgeführt werden. Die Adaption auf andere Sollpositionen kann online erfolgen. Jedoch ist zu beachten, dass die adaptierten Trajektorien streng genommen nicht mehr zeitoptimal sind.

[6] Überschleifen bedeutet in der Robotik, dass bei Punkt-zu-Punkt-Bewegungen nicht abgewartet wird, bis der TCP an einer Position zum Stillstand kommt. Das Verfahren zur nächsten Position wird bereits begonnen, wenn eine Mindestgeschwindigkeit oder ein Mindestabstand zur Sollposition unterschritten wird. [Web09]

Zwischenfazit

Wie an den Erscheinungsjahren der vorgestellten Veröffentlichungen zu erkennen ist, handelt es sich bei der Optimierung des Zeitverhaltens unterschiedlicher Mechanismen um ein Themenfeld, an dem bereits lange geforscht wird (siehe etwa [BDG85]). Nichtsdestotrotz ist es aber auch in den letzten Jahren noch von wissenschaftlichem Interesse. Allerdings gilt hier, wie bereits bei den analysierten Quellen hinsichtlich redundanter Mechanismen, dass weder Echtzeitfähigkeit noch der Einsatz auf industrieller Hardware im Fokus stehen. Werden Ausführungszeiten angegeben, so beziehen sich diese wie in [vDS11a] oder [vDS11b] auf Rapid-Prototyping-Systeme, Computer bzw. nicht näher spezifizierte Rechensysteme ([KTW06; WSSP12; GRN13]) oder Simulationen etwa in *MATLAB* ([vDS09; RIS13]). Zudem liegen die Zeiten in der Regel im Bereich von mehreren Millisekunden ([vDS09; vDS11a; vDS11b; WSSP12]) bis hin zu Minuten ([GRN13]). Gattringer et al. [GRN13] sowie Springer et al. [SGS14] kennzeichnen ihre Ansätze bereits explizit als offline auszuführen.

Einen ersten Ansatz Richtung industrieller Einsetzbarkeit liefern Haschke et al. in [HWR08], wobei hier die eigentliche rechenintensive Berechnung auf einem Computer erfolgt und die Ergebnisse direkt über eine Schnittstelle auf die Steuerung des SKM übertragen werden. Für die Übertragung der Daten werden keine Zeiten angegeben, ebenso wenig für die Ausführung des IKP auf der Steuerung. Die Berechnung auf dem Computer benötigt jedoch maximal 0,36 ms. Somit kann eine Aufteilung des Ansatzes auf zwei Systeme, um beispielsweise die SPS von der rechenintensiven Optimierung zu entlasten, als ein möglicher Ausgangspunkt für einen echtzeitfähigen Ansatz angesehen werden. Gleiches gilt für die vorgestellte Reduzierung der Geschwindigkeitsverläufe auf ein Polynom zweiter ([KTW06; HWR08]) oder dritter Ordnung ([HWR08]). Zudem kann in Betracht gezogen werden, den ebenfalls von Kroeger et al. sowie Haschke et al. vorgestellten Ansatz hinsichtlich der Nutzung der maximalen Werte für Beschleunigung, Verzögerung und Geschwindigkeit für die langsamste Achse sowie der davon abhängigen Beeinflussung der anderen Achsen auf kinematisch redundante Mechanismen zu adaptieren.

Ähnlich wie Haschke et al. gehen auch Rymansaib et al. in [RIS13] vor: die Berechnung von Geschwindigkeitsvorgaben für zeit- und genauigkeitsoptimierte Punkt-zu-Punkt-Bewegungen von SKM erfolgt in *MATLAB*. Die Vorgaben werden dann über eine Schnittstelle an den Mechanismus bzw. seine Steuerungseinheit übergeben. Allerdings werden hier keine trapezförmigen sondern exponentielle Geschwindigkeitsverläufe betrachtet, zudem werden keine Rechenzeiten o.ä. angegeben. Somit kann das Potential dieses Ansatzes hinsichtlich der Echtzeitfähigkeit nicht beurteilt werden.

Wie bereits bei den Ansätzen zu redundanten Mechanismen finden auch bei den nichtredundanten Mechanismen unterschiedliche Modelle Anwendung. Während von Haschke et al. in [HWR08] trapezförmige Geschwindigkeitsverläufe in Form eines Polynoms zweiter Ordnung angenommen werden, setzen Rymansaib et al. in [RIS13] hierzu ein Polynom dritter Ordnung ein. Bobrow et al. und van den Broeck et al. hingegen nutzen die Zustandsraumdarstellung [BDG85; vDS09].

Tabelle 3-2 fasst die vorgestellten Quellen chronologisch sortiert zusammen.

Tabelle 3-2: *Chronologische Zusammenfassung der Quellen zu Optimierungsstrategien für das Zeitverhalten nicht-redundanter Mechanismen*

Quelle	Optimierungs-ziel	Mechanismus	Bewegungs-art	Rechen-hardware	Rechenzeit
[Bob82; BDG83; BDG85]	Zeit-minimierung	SKM	CP	-	-
[KTW06]	Zeit-minimierung	SKM	PTP, CP	-	-
[HWR08]	Zeit-minimierung	SKM	PTP, CP	Computer + Roboter-steuerung	0,36 ms
[vDS09]	Zeit-minimierung	-	PTP	*MATLAB*	7 ms
[vDS11a; vDS11b]	Zeit-minimierung	-	PTP	SpeedGoat, dSPACE-System	5 ms, 17 ms
[WSSP12; WS15]	Energie-minimierung	-	PTP	-	10 ms
[RIS13]	Zeit-minimierung	SKM	PTP	*MATLAB*	-
[GRN13]	Zeit-minimierung	SKM	PTP	Computer	3 min
[SGS14]	Zeit-minimierung	PKM	PTP	-	-

Da generell Modelle bei den Ansätzen sowohl für redundante als auch nicht-redundante Mechanismen eine wichtige Rolle spielen, zeitgleich aber die Ausführung des im Rahmen der vorliegenden Arbeit entwickelten Ansatzes auf einer SPS angestrebt wird, werden im folgenden Abschnitt relevante Quellen hinsichtlich modellbasierter Optimierungs- und Regelungsansätze auf industriell einsetzbarer Hardware vorgestellt.

3.3 Modellbasierte Optimierungs- und Regelungsansätze auf industriell einsetzbarer Hardware

Der im Rahmen des hier beschriebenen Dissertationsvorhabens entwickelte Ansatz fokussiert explizit den echtzeitfähigen Einsatz auf einer Industriesteuerung. Basierend auf den vorgestellten Quellen in den Abschnitten 3.1 und 3.2 sowie aufgrund der generell bei kinematisch redundanten Mechanismen vorliegenden Unbestimmtheit des IKP wird dabei ein modellbasierter Optimierungsansatz angestrebt. Deswegen wird als dritter Teilbereich im Stand der Wissenschaft und Technik nun ein Überblick über modellbasierte

Optimierungsansätze auf industriell einsetzbarer Hardware gegeben. Dies erfolgt ohne eine Einschränkung auf Quellen mit explizitem Bezug auf Trajektorienplanung oder kinematische Redundanz. Neben der Implementierung auf einer SPS werden auch einzelne Ansätze für eingebettete Hardware betrachtet, da diese ebenfalls als industriell nutzbar angesehen wird. Aufgrund der ständig steigenden Rechenleistung und um eine Vergleichbarkeit mit den aktuell eingesetzten Systemen insbesondere bezogen auf die Rechenzeit zu erzielen, werden vornehmlich Veröffentlichungen der letzten zehn Jahren betrachtet. Wieder erfolgt die Gliederung anhand des Alters der Quellen sowie gegebenenfalls auftretender inhaltlicher Zusammenhängen.

Einen Ansatz für die Umsetzung einer MPR auf einer SPS am Beispiel der Temperaturregelung eines Heizlüfters stellen Valencia-Palomo et al. in [VHR09] vor. Ein erklärtes Ziel der Veröffentlichung ist in folgendem Zitat zu finden:

> *„... this paper seeks to encourage the academic community to put more effort in designing and promoting simple control algorithms, such as PFC, that can be set up in standard PLC units.“*

Es wird angeregt, dass sich die wissenschaftliche Gemeinschaft mehr mit der Entwicklung einfacher Regelalgorithmen befassen soll, die auf einer SPS ausgeführt werden können. Als Beispiel für diese Regler wird die sogenannte Predictive Functional Control (PFC) genannt, bei der es sich um eine modellprädiktive Regelung handelt [RO09]. SPS (engl. programmable logic controller, PLC) werden dabei als die am besten akzeptierten Computer in der Industrie angesehen, was unter anderem auf die einfache (elektrische und mechanische) Montage, eine Vielzahl an Ein- und Ausgängen oder Kommunikationsschnittstellen sowie die Möglichkeit der Analyse laufender Programme zur Laufzeit zurückgeführt wird. Um die Implementierung einer MPR auf einer SPS umzusetzen, wurden in [VHR09] die Algorithmen und Modelle möglichst einfach gestaltet. Dadurch sollen die Rechenanforderungen gering gehalten werden. Zunächst wird im Rahmen einer praktischen Umsetzung die Temperatur am Ende eines Heizlüfters durch die Drehzahl des Lüfters geregelt, wobei ein Modell erster Ordnung genutzt wird. Im zweiten Beispiel soll die Drehzahl eines Motors geregelt werden, es handelt sich um ein System zweiter Ordnung. In beiden Fällen wird bei der Modellierung auf die Zustandsraumdarstellung zurückgegriffen. Die Gütefunktion beinhaltet die quadratische Abweichung zwischen dem prädizierten Regelgrößenverlauf und dem Sollverlauf sowie zusätzlich die gewichtete Änderung der Stellgröße. Begrenzungen etwa der Stellgrößen werden nicht berücksichtigt, wodurch die Minimierung ohne Optimierer mathematisch gelöst werden kann. Die Implementierung mit einem Prädiktionshorizont von zehn und einem Steuerhorizont von drei Zeitschritten liefert für beide Fälle laut der Autoren gute Regelergebnisse. Es werden dabei auf der SPS Zykluszeiten von maximal 11,64 ms gemessen, was bezogen auf den Prozess als ausreichend schnell angesehen wird. Aufgrund dessen wird von Valencia-Palomo et al. ein entsprechendes Potential für modellprädiktive Ansätze auf industrieller Hardware gesehen, wobei insbesondere die recheneffiziente Gestaltung der Algorithmen als entscheidend für eine ausreichend geringe Zykluszeit angesehen wird.

A. Syaichu-Rohmann und R. Sirius stellen in [SS11] die Entwicklung und Implementierung einer modellprädiktiven Drehzahlregelung eines Gleichstrommotors auf einer SPS vor. Ziel ist, dass der Motor durch die Regelung einer vorgegebenen Drehzahlreferenz bestmöglich folgt. Der Ansatz wurde als Kontaktplan auf einer SPS der Firma Mitsubishi implementiert

und gegen einen PI-Regler mit Anti-Windup verglichen. Die Optimierung wurde als quadratische Programmierung umgesetzt, die erzielten Ergebnisse sind schlechter als die des PI-Reglers. Dass keine besseren Resultate erzielt wurden, liegt laut den Autoren an der für die Rechenzeit entscheidenden Begrenzung auf einen Prädiktionshorizont von zwei Zeitschritten und eine maximale Iterationsanzahl des Optimierers von vier. Die erforderliche Berechnungszeit ist nicht angegeben.

In der Veröffentlichung [RVKF11] stellen Rauová et al. einen Ansatz für eine nach eigenen Angaben echtzeitfähige modellprädikitve Regelung eines Heizlüfters auf einer SPS der Fa. *Siemens* vor. Dabei wird das eigentliche Optimierungsproblem offline auf einem PC gelöst und die Ergebnisse (Systemzustand und jeweils zugehörige optimierte Parameter) als *Lookup-Table* gespeichert. Als Modell für die MPR wird ein lineares zeitdiskretes und zeitinvariantes Modell in Form der Zustandsraumdarstellung genutzt. Nach Lösung des Optimierungsproblems werden die Ergebnisse aus der Lookup-Table in einen binären Suchbaum überführt, der auf die SPS übertragen wird. Hier wird online der aktuelle Systemzustand berechnet und der zugehörige Eintrag des Suchbaums ermittelt. Für den folgenden Regelzyklus werden die hinterlegten optimierten Parameter für den Heizwiderstand und die Lüfterdrehzahl genutzt, bevor der Ablauf im folgenden Zyklus mit der Berechnung des aktualisierten Systemzustands erneut beginnt. Es werden keine Angaben hinsichtlich der erforderlichen Rechenzeiten oder der erreichten Zykluszeit der SPS angegeben, jedoch wird die erzielte Regelgüte als zufriedenstellend bezeichnet.

In [VŠ14] greifen J. Velagić und J. Šabić diesen Ansatz auf und adaptieren ihn auf die Regelung der Temperatur eines Inkubators. Wie bereits bei Rauová et al. wird eine explizite MPR zusammen mit einer Kombination aus Computer und SPS genutzt. Dabei wird das eigentliche Optimierungsproblem offline mittels quadratischer Programmierung mit mehreren Parametern auf einem Computer gelöst und die Ergebnisse in Form von *C*-Code als Lookup-Table gespeichert. Nach Abschluss der offline-Berechnung wird diese auf die SPS übertragen. Hier wird nun online der aktuelle Systemzustand berechnet und darauf basierend der entsprechende Eintrag der Lookup-Table ermittelt. Die zum aktuellen Systemzustand zugehörigen Parameter werden für den nächsten Zyklus genutzt, bevor der Vorgang mit der Neuberechnung des Systemzustands von vorne beginnt. Auch in diesem Fall wird keine Zykluszeit angegeben und die Regelgüte als zufriedenstellend bezeichnet.

Zwei unterschiedliche Umsetzungen für eine MPR unter Nutzung einer SPS stellen Mrosko et al. in [MM11] und [MM12] am Beispiel der Füllstandregelung eines Labortanks vor. Bei der ersten Variante erfolgt die Ausführung der MPR auf einem Computer mit *MATLAB Simulink*, während die Anbindung der Peripherie sowie etwa die Datenerfassung auf einer SPS der Fa. *Siemens* erfolgen. Der Austausch zwischen Computer und SPS wird mittels des Kommunikationsprotokolls OPC realisiert. Benanntes Problem dabei ist, dass die Simulationszeiten der Modelle in *Simulink* nicht unbedingt den Zeiten am realen System entsprechen, wodurch es insbesondere bei dynamischen Prozessen zu Fehlern in der Regelung kommen kann. Durch eine geeignete Parametrierung in *Simulink* kann die zeitliche Abweichung jedoch ausreichend reduziert werden. Im Rahmen der zweiten Variante erfolgt die Ausführung der MPR sowie der Datenerfassung etc. ausschließlich auf der SPS. Beide Ansätze liefern vergleichbare Ergebnisse, es werden jedoch keine Aussagen insbesondere zu der erzielten Zykluszeit der SPS angegeben.

Einen weiteren Ansatz, ebenfalls am Beispiel der Temperaturregelung eines Heizlüfters über die Vorgabe der Lüfterdrehzahl und der Heizleistung, stellen Huyck et al. in [HCL+12] vor. Ziel ist die Adaptierung einer modellprädiktiven Regelung für kleine MIMO-Systeme (engl. Multiple Input Multiple Output) auf eine SPS. Für das Beispiel des Heizlüfters werden der sogenannte *Hildreth QP*-Algorithmus [Hil57] und der *qpOASES*-Algorithmus [FKP+14] auf einer SPS der Fa. *Siemens* implementiert und verglichen. Als Ergebnis wird genannt, dass der *Hildreth QP*-Algorithmus ca. eine Sekunde langsamer auf Änderungen reagiert als der *qpOASES*-Algorithmus und somit die Summe der quadratischen Fehler höher liegt. Dabei benötigt der *Hildreth QP*-Algorithmus maximal ca. 19 ms und der *qpOASES* ca. 125 ms Rechenzeit.

In [HFD+12] wird der Anwendungsfall um einen durch den Code-Generator *CVXGEN* erzeugten Optimierer erweitert. Dieser liegt jedoch ausschließlich in der Sprache *C* vor, die von der eingesetzten SPS nicht unterstützt wird. Somit erfolgt keine Umsetzung auf die reale Hardware und es wurden keine Vergleichsergebnisse erzeugt. Für die beiden Algorithmen *Hildreth QP*-Algorithmus und *qpOASES* lagen die maximalen Rechenzeiten bei ca. 50 ms bzw. ca. 200 ms. Die größere Rechenzeit gegenüber [HCL+12] kann etwa an einer anderen Wahl der maximal erlaubten Anzahl an Iterationen oder der Wahl eines anderen Sollwertverlaufs für die Temperatur begründet liegen, die eingesetzte SPS ist in beiden Beiträgen gleich.

Der *Hildreth QP*-Algorithmus wird zudem von Huyck et al. in [HBM+13] am Beispiel der Temperaturregelung einer prototypischen Destille auf einen sogenannten Programmable Automation Controller (PAC)[7] adaptiert. Die maximal erforderliche Rechenzeit liegt dabei bei ca. 4 ms.

Eine modellprädiktive Positionsregelung für einen Portalkran mit zwei Antrieben zur Verfolgung einer vorgegebenen Trajektorie stellen B. Käpernick und K. Graichen in [KG14a] und [KG14b] vor. Hierbei handelt es sich um eine nichtlineare Regelung, die Stellgrößenbeschränkungen hinsichtlich der Beschleunigung berücksichtigt. Zur Lösung des resultierenden quadratischen Problems wird auf ein Gradientenverfahren gesetzt, dessen Grundlagen bereits in [GK12] anhand eines inversen Doppelpendels beschrieben wurden und dessen Einsatz für die modellprädiktive Regelung des Krans auf einem Rapid Prototyping System der Fa. *dSPACE* von B. Käpernick at al. in [KG13] vorgestellt wird. Die Generierung der erforderlichen Funktionsbausteine für die SPS der Fa. *Festo* erfolgt aus *Simulink* heraus mittels des Moduls *PLC coder*. Der Ansatz wird mit einer Zykluszeit von 2 ms ausgeführt, jedoch liegt die tatsächlich erforderliche Rechenzeit im Bereich von 1 ms. Für eine verbesserte Regelgüte wurde der Ansatz abschließend um eine flachheitsbasierte Trajektorienplanung erweitert, wobei hierzu keine weiteren Rechenzeiten angegeben werden.

Kufoalor et al. stellen in [KRI+14] eine sogenannte eingebettete MPR für einen Unterwasser-Separationsprozess aus dem Bereich der Petrochemie vor, die auf einer SPS ausgeführt wird. Ziel ist die Regelung des Gasvolumenanteils an den Systemausgängen für Gas und Flüssigkeit. Um nachgeschaltete Systeme wie Pumpen und Kompressoren vor Schaden

[7] Ein PAC und eine SPS dienen beide der Ausführung von Automatisierungsaufgaben. Der PAC kombiniert dazu die Eigenschaften und Möglichkeiten einer Computer-basierten Steuerung und einer SPS. Bei der Programmierung können vielfältigere Möglichkeiten etwa hinsichtlich der Programmiersprache genutzt werden, zudem weisen sie in der Regel höhere Rechenleistung auf. [Sha17]

zu bewahren, muss der Gasvolumenanteil in festgelegten Grenzen bleiben, zudem müssen die Betriebsbereiche aller Komponenten und etwa die Grenzen von Ventilen etc. berücksichtigt werden. Es resultiert ein zu lösendes Mehrzielregelungsproblem mit mehreren unterschiedlich gewichteten Regelungszielen und Beschränkungen der Ein- und Ausgänge. Das Systemverhalten wurde anhand von Sprungantworten unter Nutzung eines Simulators für das System aufgenommen und in lineare SISO-Modelle für jedes Ein-/Ausgangspaar überführt. Diese einzelnen Modelle wurden wiederum zu einem MIMO-Gesamtmodell des Systems überlagert. Die Entwicklung und Konfiguration der MPR erfolgte mittels der Software *SEPTIC* (Statoil Estimation and Prediction Tool for Identification and Control) der Fa. *Statoil*. Per automatischer Code-Generierung wurde das Ergebnis von *SEPTIC* zunächst in *C*-Code überführt. Daraus wurde mittels einer *MATLAB*-Toolbox der eigentliche für das Problem spezifische Algorithmus basierend auf einer in [PC11] vorgestellten *primal-dual first-order-Methode* in *ANSI C* generiert und auf der SPS der Fa. *ABB* implementiert. Die maximal benötigte Rechenzeit liegt bei 116,40 ms, was bezogen auf die geforderte Abtastrate von 1 Hz als ausreichend schnell angesehen wird.

Den gleichen Anwendungsfall mit einem sehr ähnlichen Ansatz, allerdings im Rahmen einer Implementierung auf eingebetteter Hardware, stellen Kufoalor et al. in [KAJ+15] vor. Hier wird erneut der Unterwasser-Separationsprozess inklusive der aus der Software *SEPTIC* exportierten MPR betrachtet. Für die Ausführung der Regelung wird die open-source-Hardware *Ethernut 5* eingesetzt, die laut den Autoren kostengünstig ist, aber nur begrenzte Rechenleistung aufweist. Die resultierenden maximalen Rechenzeiten liegen abhängig von den gewählten Einstellungen etwa hinsichtlich der mathematischen Genauigkeit zwischen 690 ms und 1 s.

Basierend auf dem Ansatz von Kufoalor et al. stellen Binder et al. in [BKPJ14] eine eingebettete MPR für eine elektrische Tauchpumpe im Kontext der Fluidförderung vor, die ebenfalls auf einer SPS ausgeführt wird. Der Fokus der Veröffentlichung liegt auf der Implementierung der MPR, die das Ziel hat, die Pumpe immer in einem Betriebsbereich zu betreiben, der eine maximale Haltbarkeit der Pumpe verspricht. Dazu müssen Grenzen etwa der elektrischen Parameter, des aufgebauten Drucks oder der auftretenden Temperaturen eingehalten werden. Es resultiert wiederum ein zu lösendes Mehrzielregelungsproblem mit mehreren unterschiedlich gewichteten Regelungszielen. Die Pumpe wird dazu mit einem Modell dritter Ordnung beschrieben, die Entwicklung und Konfiguration der MPR erfolgte wie in [KRI+14] mittels der Software *SEPTIC* der Fa. *Statoil*. Auch hier wurde das Ergebnis von *SEPTIC* per automatischer Code-Generierung und unter Nutzung einer *MATLAB*-Toolbox in ein quadratisches Programmierungsproblem in ANSI C überführt und auf der SPS implementiert. Die Validierung des Ansatzes erfolgte im Rahmen einer Hardware-in-the-loop Simulation. Dafür wurde die SPS über OPC mit einem Computer verbunden, auf dem ein in *MATLAB* realisiertes Modell des Systems ausgeführt wurde. Die mittlere Rechenzeit der SPS lag in diesem Szenario bei 124,09 ms und die maximale Rechenzeit bei 124,89 ms, was als ausreichend schnell bezogen auf den Anwendungsfall der Tauchpumpe angesehen wird (die geforderte Abtastrate beträgt 1 Hz). Als ausschlaggebend wird hier die maximal erlaubte Anzahl an Iterationen des Optimierers eingeordnet, um entweder die Regelgüte bei gleichzeitigem Anstieg der Rechenzeit zu verbessern (mehr Iterationen) oder die Rechenzeit bei gleichzeitiger Verschlechterung der Regelgüte zu verringern (weniger Iterationen).

Einen weiteren Ansatz für die Ausführung einer MPR auf einer SPS stellen Zöller et al. in [ZRA15] vor. Ziel ist es die Temperatur einer Anlage für das thermische Aufdampfen unter Vakuum auf weniger als 0,2 K genau zu regeln. Hierzu wird das Modell des Temperaturverhaltens basierend auf physikalischen Gleichungen etwa hinsichtlich Temperaturübergängen oder Wärmestrahlung aufgestellt. Die Parametrierung des Modells erfolgt anhand von Messungen, wobei die Wahl der Parameter als Optimierungsproblem formuliert wurde. Dessen Ziel war ein möglichst deckungsgleicher Verlauf von gemessenen und simulierten Werten, die Lösung erfolgt in *MATLAB* mittels des Optimierers *fmincon*. Das resultierende physikalisch motivierte Modell weißt einen relativen Fehler gegenüber den Messwerten von weniger als 6 % auf und beschreibt das Verhalten des Systems über einen großen Arbeitsbereich. Es wird als ausreichend genau angesehen. Für die Ausführung der entwickelten linearen zeitvarianten MPR wurde eine SPS gewählt, die aus *MATLAB Simulink* heraus programmierbar ist. Zur Reduzierung der erforderlichen Rechenzeit wurde der Kontrollhorizont kleiner gewählt als der Prädiktionshorizont (siehe [Dit04]). Die Lösung des Optimierungsproblems erfolgt mittels des Optimierers *qpOASES* [FKP+14]. Mit diesem Ansatz konnte die Zykluszeit der SPS auf 1 s festgelegt werden, die MPR wird aufgrund der geringen Dynamik des Prozesses langsamer aufgeführt. Mittels des Ansatzes wird die Einschwingzeit der Temperatur gegenüber einem PID-Regler reduziert, das Überschwingen zudem von über 5,7 % auf unter 0,3 % reduziert.

Von Pereira et al. wird in [PLA15] ein weiterer Ansatz für eine MPR auf einer SPS vorgestellt, die mittels der Programmiersprache *Strukturierter Text* aus der *IEC 61131-3* implementiert wurde. Unter Einsatz der sogenannten *Nesterov's fast gradient methode* [Nes04] wird mittels der MPR der Wasserstand der Behälter eines Vier-Tank-Systems geregelt. Es soll im Rahmen der Veröffentlichung zum einen gezeigt werden, dass modellprädiktive Regelungen auch für dynamischere Prozesse eingesetzt werden können. Zum anderen wird die Umsetzung auf eine SPS angestrebt, da es sich hier um ein „zuverlässiges und robustes" System und den „am meisten in der Industrie akzeptierten Computer" handelt. Das Systemverhalten wird als lineares quadratisches Modell abgebildet, im Rahmen der MPR werden die Beschränkungen der Eingangsgrößen berücksichtigt. Im Rahmen eines Versuchsaufbaus wurde die auf der SPS ausgeführte MPR über OPC an ein *MATLAB*-Modell des zu regelnden Vier-Tank-Systems angekoppelt. Dabei wurde eine maximale Rechenzeit von 255 ms gemessen. Zudem wurde im Rahmen einer weiterführenden Analyse festgehalten, dass bei einem steigenden Prädiktionshorizont und der damit verbundenen Erhöhung der Anzahl an Optimierungsvariablen die erforderliche Zykluszeit stark ansteigt: Zehn Variablen (Prädiktionshorizont $n_\mathrm{p} = 5$) führen zu einer maximalen Zykluszeit von 10 ms, bei 70 Variablen ($n_\mathrm{p} = 35$) sind es hingegen 186 ms. Die Regelgüte wird von den Autoren als gut beschrieben, jedoch wurden nach eigenen Angaben im Hinblick auf die Rechenzeiten noch nicht alle Möglichkeiten einer SPS ausgenutzt.

Eine MPR für die Regelung industrieller Prozesse stellen Aslam et al. in [AHSZ17] ebenfalls am Beispiel einer Füllstandregelung, zusätzlich jedoch auch anhand der Temperaturregelung einer Flüssigkeit, vor. Die Flüssigkeit wird in einem Tank auf eine definierte Temperatur aufgeheizt, wobei die MPR die Leistung des Heizelements beeinflusst. Des weiteren wird die Flüssigkeit in Flaschen abgefüllt, wobei die Durchflussmenge eines Ablassventils durch die MPR verändert wird. Die Modellierung des Systems erfolgt als Zustandsraummodell, wobei unter anderem das Flüssigkeitsvolumen und die Dichte, der dadurch entstehende Druck oder die spezifische Wärmekapazität der Flüssigkeit berück-

sichtigt werden sowie die Geometrie des Ablaufs oder die Leistung des Heizelements. Die mittels der Programmiersprache *Ladder Logic* auf der SPS implementierte MPR ist laut den Autoren echtzeitfähig, jedoch werden keine Angaben zu den geforderten oder erzielten Zykluszeiten gemacht. Hinsichtlich der erzielten Ergebnisse wird die Regelung als effizient sowohl für die Temperatur- als auch die Füllstandregelung beschrieben.

Zwischenfazit

Wie aus den vorgestellten Quellen zu entnehmen ist, gibt es verschiedene Anwendungsszenarien für modellprädiktive Ansätze und die Nutzung von Optimierung auf industrieller Hardware. Der Einsatz einer SPS wird dabei zum Beispiel in [VHR09] oder [PLA15] dahingehend begründet, dass es sich um eine in der Industrie akzeptierte Komponente handelt. Neben einem einfachen Anschluss und einer Vielzahl an Ein- und Ausgängen sowie Kommunikationsschnittstellen werden Robustheit und Zuverlässigkeit als Merkmale genannt. Für die vorliegende Arbeit wird hieraus der Schluss gezogen, dass die definierte Anforderung bezüglich der Umsetzung des Ansatzes auf eine SPS ein wichtiger Schritt in Richtung industrieller Akzeptanz ist. Allerdings liegen die erzielten Zykluszeiten bei den meisten der vorgestellten Ansätzen mit einigen Millisekunden ([VHR09; HFD+12; HCL+12; HBM+13]) bis in den Bereich von hundert Millisekunden und mehr ([HFD+12; HCL+12; KRI+14; BKPJ14; KAJ+15; ZRA15; PLA15]) über der im Rahmen der vorliegenden Arbeit geforderten Grenze von 1 ms. Es muss an dieser Stelle jedoch die Dynamik des zu regelnden Prozesses berücksichtigt werden. Hierauf bezogen können auch Zykluszeiten von 100 ms und mehr als ausreichend schnell angenommen werden, jedoch sind die Ansätze dann auf Prozesse mit ähnlichen Zeiten beschränkt, beziehungsweise die Adaptierbarkeit auf dynamischere Prozesse muss separat überprüft werden. Um hier eine größere Flexibilität bei der Adaption auf verschiedenen Mechanismen zu erreichen, ist die definierte Grenze in der vorliegenden Arbeit auf 1 ms festgelegt. Jedoch beweisen Käpernick et al. in [KG14a] und [KG14b], dass Zeiten in diesem Bereich möglich sind. Hierzu ist laut [VHR09] insbesondere die recheneffiziente Gestaltung der Algorithmen entscheidend. Dazu können etwa Abstriche in der Genauigkeit der Modelle gemacht werden. So sehen Zöller et al. in [ZRA15] einen relativen Fehler des eingesetzten Modells gegenüber den Messwerten von weniger als 6 % als hinreichend genau an.

Eine weitere Möglichkeit der Zykluszeitreduzierung liegt in der Auslagerung von rechenintensiven und damit zeitintensiven Aufgaben auf leistungsstärkere Systeme, die dann wie etwa in [MM11] und [MM12] per OPC an die SPS angekoppelt werden. Leider werden in [MM11] und [MM12], ebenso wie in [SS11; RVKF11; VŠ14] oder [AHSZ17], keine Zeiten angegeben, sodass die Leistungsfähigkeit nicht beurteilt werden kann. Die Idee der Aufteilung wird jedoch in anderer Form häufig in der petrochemischen Industrie eingesetzt. Als überlagerte Regelung wird eine MPR auf einem Computer/Server ausgeführt und die unterlagerten Regelungen werden auf einer oder mehreren SPS ausgeführt [BKPJ14]. Zudem finden sich auch in anderen Bereichen Ansätze, die auf eine Aufteilung der Berechnung auf zwei Systeme setzen (siehe vorheriger Abschnitt 3.2). Für das Ziel der vorliegenden Arbeit ist eine strikte Trennung in offline- und online-Berechnung aufgrund der erwarteten Variationen der Bauteile beziehungsweise Sollpositionen im Kontext kleiner Losgrößen nicht sinnvoll, da hierdurch auf zur Laufzeit auftretende Änderungen eventuell

nicht adäquat reagiert werden kann. Eine Aufteilung der Online-Berechnung auf zwei Systeme, um mehr Rechenleistung zur Verfügung zu haben, kann jedoch nützlich sein.

Um die Rechenzeit weiter zu reduzieren, sind neben den bereits genannten Punkten auch die Ergebnisse von Pereira et al. in [PLA15] zu berücksichtigen. Hier wurde eine starke Abhängigkeit der Rechenzeit von der Wahl des Prädiktionshorizonts und der damit verbundenen Anzahl der Optimierungsvariablen festgestellt. Somit kommt im Rahmen der vorliegenden Arbeit der Wahl eines geeigneten Prädiktionshorizontes, bezogen auf die Ergebnisse und die erforderlichen Rechenzeit, eine entsprechende Bedeutung zu.

Tabelle 3-3 fasst die Quellen nach der Rechenzeit aufsteigend sortiert zusammen.

Tabelle 3-3: *Zusammenfassung der Quellen zu modellbasierten Optimierungs- und Regelungsansätze auf industriell einsetzbarer Hardware*

Quelle	Optimierungs-/Regelungsziel	System	Rechen-hardware	Rechenzeit
[KG14a; KG14b]	Positionsregelung	Portalkran	SPS	1 ms-2 ms
[HBM+13]	Temperaturregelung	Destille	PAC	4 ms
[PLA15]	Füllstandregelung	4-Tank-System	SPS	10 ms,186 ms
[VHR09]	Temperaturregelung	Heizlüfter	SPS	11,64 ms
[HFD+12]	Temperaturregelung	Heizlüfter	SPS	19 ms, 125 ms
[HCL+12]	Temperaturregelung	Heizlüfter	SPS	50 ms, 200 ms
[KRI+14]	Regelung des Gasvolumenanteils	Unterwasser-Separationsprozess	SPS	116,40 ms
[BKPJ14]	Einhaltung des Betriebsbereichs	Tauchpumpe	SPS	124,89 ms
[KAJ+15]	Regelung des Gasvolumenanteils	Unterwasser-Separationsprozess	eingebettete Hardware	690 ms-1 s
[ZRA15]	Temperaturregelung	Vakuum-Beschichtung	SPS	1 s
[SS11]	Drehzahlregelung	Gleichstrommotor	SPS	-
[RVKF11]	Temperaturregelung	Heizlüfter	Computer + SPS	-
[VŠ14]	Temperaturregelung	Inkubator	Computer + SPS	-
[MM11; MM12]	Füllstandregelung	Labortank	Computer + SPS	-
[AHSZ17]	Füllstand- und Temperaturregelung	Abfüllanlage	SPS	-

3.4 Zusammenfassung und Fazit

Wie in den vorangegangenen Abschnitten 3.1, 3.2 und 3.3 dieses Kapitels dargelegt, gibt es bereits viele Ansätze und Veröffentlichungen, die sich mit der Regelung von SKM, PKM und HKM im Allgemeinen, der optimalen Regelung dieser Mechanismen mit und ohne Redundanz bezogen auf verschiedene Kriterien und der Ausführung modell- und optimierungsbasierter Ansätze auf industrieller Hardware beschäftigen. An den Erscheinungsjahren der Quellen ist zu erkennen, dass es durchaus schon seit Jahren entsprechende Forschungsaktivitäten gibt. Dieses Interesse besteht bis zum jetzigen Zeitpunkt und resultiert in immer neuen Veröffentlichungen und Ansätzen zu den einzelnen Themen. Ein Ansatz, der voll umfänglich das Ziel der vorliegenden Dissertation, eine echtzeitfähige modellprädiktive Planung zeitoptimierter Trajektorien für mehrachsige kinematisch redundante Mechanismen auf industrieller Hardware, widerspiegelt, ist dem Autor jedoch nicht bekannt.

Gerade im Bereich der optimierungsbasierten Trajektorienplanung redundanter Mechanismen wird bei den bisherigen Ansätze in der Regel nur die generelle Funktionsfähigkeit auf Basis von Simulationen oder mittels Laboraufbauten mit *MATLAB* und hochperformanten Prototyping-Einheiten wie *dSPACE*-Systemen analysiert. Eine mögliche Echtzeitfähigkeit der Ansätze sowie die Implementierung auf industrieller Hardware und damit das Potential dieser Ansätze im industriellen Umfeld wird hingegen nicht betrachtet. Die Veröffentlichungen von Kröger et al. [KTW06] und Haschke et al. [HWR08] hinsichtlich der Zeitoptimierung des Bewegungsverhaltens eines SKM ohne Nutzung von Redundanz zeigt jedoch, dass eine echtzeitfähige zeitoptimierte Regelung von Mechanismen durchaus möglich ist. Allerdings wird auch hier auf einen Computer zur Berechnung der Sollvorgaben zurückgegriffen, wobei die Ergebnisse dann an die Steuerung des Mechanismus übergeben werden.

Die in einigen Ansätzen genutzte (offline ausgeführte) Optimierung des gesamten Pfades ist im betrachteten Kontext ebenfalls nicht zielführend, da hierzu alle zukünftigen Sollpositionen des TCP bekannt sein müssen. Aufgrund von kleinen Losgrößen und einer damit verbundenen sehr hohen Variantenvielfalt und Kundenindividualität sowie der erforderlichen Flexibilität der Systeme und Anlagen ist jedoch immer nur eine begrenzte Anzahl zukünftiger Produkte und somit zukünftiger Sollpositionen des Mechanismus bekannt. Dennoch sollten nicht nur der nächste, sondern mehrere zukünftiger Positionen betrachtet werden, um die Optimierung nicht nur bezüglich der nächsten Sollposition auszuführen, sondern den Mechanismus auch in eine „gute“ Ausgangslage für die weiteren zukünftigen Positionen zu bringen.

Eine weitere Auffälligkeit ist, dass die Möglichkeiten heutiger Automatisierungssysteme nicht oder nicht ausreichend genutzt werden. So wird etwa bei vielen Ansätze versucht, Beschleunigung und Ruck in der Optimierung zu berücksichtigen. Moderne Antriebsumrichter bieten dem Anwender jedoch bereits die Möglichkeit entsprechender Vorgaben bezüglich Maximalwerten, Reglertyp (P-, PI-, PID-Regler) für die integrierten Regelschleifen (Strom, Geschwindigkeit, Position), Vorsteuerungen und weiterer Parameter. Somit kann hier im Rahmen der Optimierung die Modellkomplexität und der damit einhergehende Rechenaufwand deutlich reduziert werden.

In der vorliegenden Arbeit liegt der Fokus auf der Entwicklung einer echtzeitfähigen modellprädiktiven Planung zeitoptimierter Trajektorien für mehrachsige kinematisch red-

undante Mechanismen, die auf einer industriellen Steuerung ausgeführt wird und die Möglichkeiten moderner Automatisierungssysteme bestmöglich nutzt. Dazu gilt es zum einen die eingesetzten Modelle und Algorithmen möglichst recheneffizient zu gestalten. Hierbei können beziehungsweise müssen Abweichungen zwischen dem Modell und dem realen Mechanismus zugunsten der Rechenzeit in Kauf genommen werden. Daraus resultiert eine optimierte anstelle einer optimalen Pfadplanung. Kann hierdurch jedoch eine Echtzeitfähigkeit und zudem eine Verkürzung der Verfahrzeit gegenüber einem Vergleichsansatz erreicht werden, sind diese Abstriche in der Genauigkeit wie beispielsweise bei Zöller et al. in [ZRA15] zu akzeptieren. Zum anderen ist es entscheidend, einen geeigneten Optimierungsalgorithmus für das resultierende Optimierungsproblem zu ermitteln, diesen auf den Befehlssatz der Steuerung zu adaptieren und ihn mit dem Ziel der Echtzeitfähigkeit auf der SPS zu implementieren. Als zusätzliches Ziel soll der Ansatz für unterschiedliche kinematisch redundante Mechanismen einsetzbar sein.

Im folgenden Kapitel wird die Grundidee der echtzeitfähigen modellprädiktiven Planung zeitoptimierter Trajektorien für kinematisch redundante Mechanismen vorgestellt.

4 Echtzeitfähige modellprädiktive Planung zeitoptimierter Trajektorien

In diesem Kapitel wird die modellprädiktive Planung zeitoptimierter Trajektorien für mehrachsige kinematisch redundante Mechanismen vorgestellt. Nach der Erläuterung der Grundidee erfolgt eine Beschreibung der eingesetzten Gütefunktion und des Optimierungsproblems sowie des genutzten Zeitmodells. Zudem wird die Nutzung des Potentials moderner Antriebsumrichter beschrieben und das für die spätere Validierung und Bewertung des Ansatzes erforderliche Referenzverfahren vorgestellt.

4.1 Grundidee

Ziel der im folgenden vorgestellten modellprädiktiven Planung für mehrachsige kinematisch redundante Mechanismen ist die Optimierung des zeitlichen Verhaltens des Mechanismus, sodass eine Bewegungsaufgabe in der minimal möglichen Zeit ausgeführt wird. Die gesamte betrachtete Bewegungsaufgabe besteht darin, dass eine Folge von N einzelnen Positionen ${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TM},k}$ mit $k = 1 \ldots N \in \mathbb{N}$ nacheinander als Punkt-zu-Punkt (PTP)-Bewegungen angefahren werden sollen, wobei der TCP in jeder Sollposition zum Stillstand kommt. Bezogen auf die im Kapitel 2 gegebene Definition handelt es sich um eine asynchrone Punkt-zu-Punkt-Bewegung, bei der alle Achsen zu unterschiedlichen Zeiten ihr jeweilige Sollposition erreichen können. ${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TM},k}$ (gekürzt auch $\vec{Z}_{\mathrm{TM,k}}$) ist der Ortsvektor des TCP TM eines Mechanismus im Schritt k bezogen auf ein Weltkoordinatensystem mit dem Ursprung O_W. Es wird davon ausgegangen, dass immer einige der zukünftigen Sollpositionen bekannt sind, diese jedoch kurzfristigen Änderungen unterliegen können. Es gilt für jede Sollposition optimierte Positionen der einzelnen Antriebe zu ermitteln. Die $m \in \mathbb{N}$ Antriebspositionen je Schritt k werden auch als Gelenkkoordinaten bezeichnet, sie sind im Vektor

$$\underline{q}_k = [q_{1,k} \ldots q_{m,k}\ ]^T \in \mathbb{R}^m \tag{4-1}$$

zusammengefasst.

Im Rahmen des Ansatzes wird der eigentliche Mechanismus zunächst gedanklich in zwei Teilmechanismen unterteilt. Der erste stellt einen nicht-redundanten Mechanismus dar, während der zweite aus den zusätzlichen, zur kinematischen Redundanz führenden, Achsen besteht. Dies wird beispielsweise von Reiter at al. in [RSGM15; RMG16; RGM18] auch als *joint space decomposition* bezeichnet (vgl. Kapitel 3). In Bild 4-1 ist dies anhand eines Sechsachs-Knickarmroboters *Denso VS087* [DEN20] mit einer zusätzlichen seriellen Achse der Fa. *Rose+Krieger* [RK 20] dargestellt.

Der Roboter stellt den nicht-redundanten und die serielle Achse den zusätzlichen zur Redundanz führenden Teil dar. Die Basis des Roboters ist im TCP $T2$ der Linearachse montiert. Es liegt ein Mechanismus mit $m = 7$ Achsen vor. Im dargestellten Beispiel können Bewegungen in der x-Richtung des Weltkoordinatensystems sowohl durch die Linearachse als auch den Roboter durchgeführt werden. Das kombinierte Verfahren aller

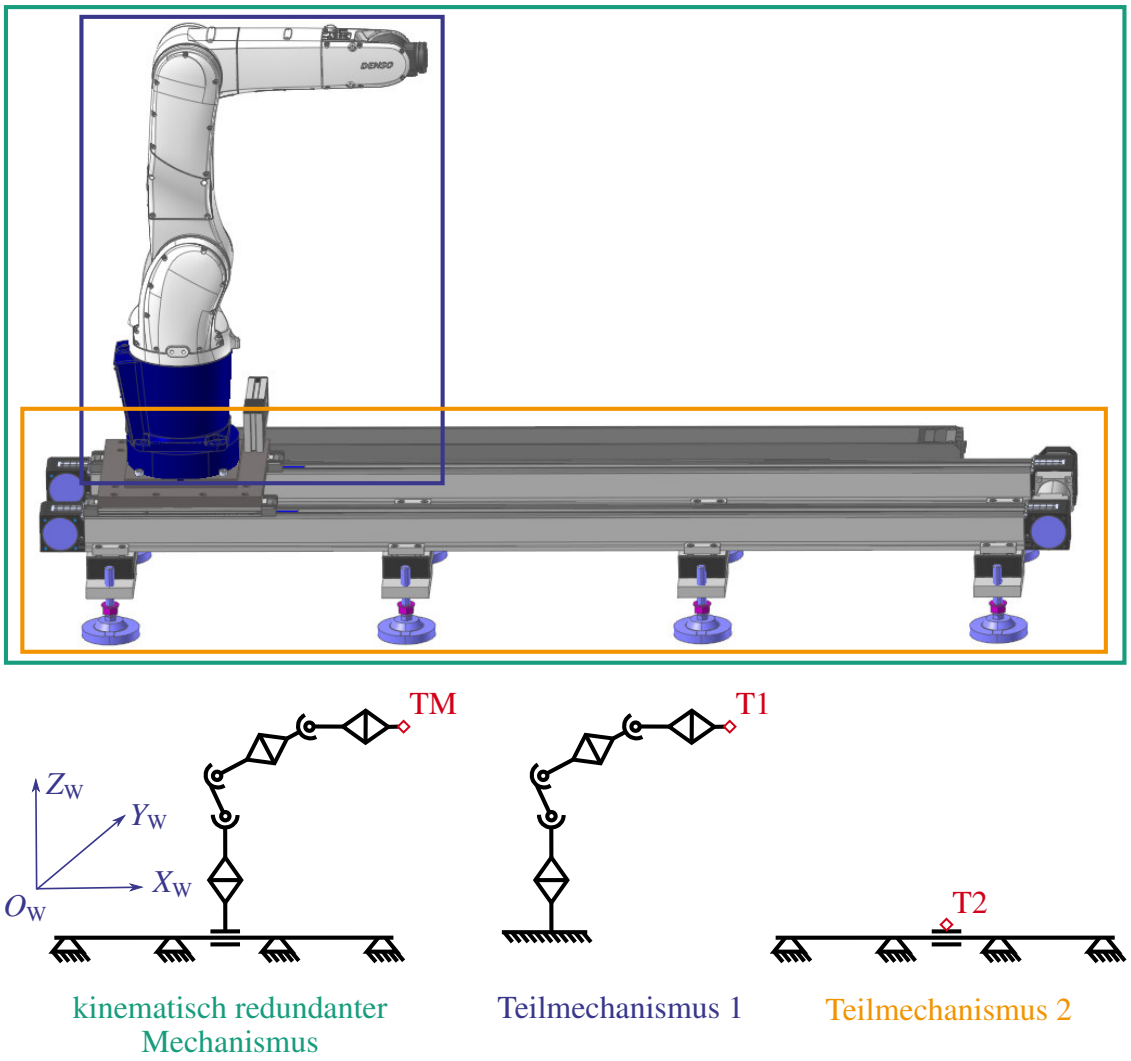


Bild 4-1: *Aufteilung eines kinematisch redundanten Mechanismus bestehend aus Sechsachs-Knickarmroboter und Linearachse*

Achsen des Mechanismus ergibt die Bewegung des TCP.

Der Ortsvektor

$$ {}_{\mathrm{OW}}\vec{Z}_{\mathrm{TM},k} = {}_{\mathrm{T2}}\vec{Z}_{\mathrm{TM},k} + {}_{\mathrm{OW}}\vec{Z}_{\mathrm{T2},k} \text{ mit } TM \equiv T1 \tag{4-2} $$

des kinematisch redundanten Mechanismus kann als Überlagerung der Ortsvektoren der Teilmechanismen dargestellt werden, wobei ${}_{\mathrm{OW}}\vec{Z}_{\mathrm{TM},k}$ in Form der Sollposition bekannt ist. Der modellprädiktive Ansatz sieht vor, die Positionen $\underline{q}_{T2,k} \in \mathbb{R}^r$ der Achsen des zweiten Teilmechanismus optimiert hinsichtlich des Wertes $J_k \in \mathbb{R}$ eines definierten Gütekriteriums zu ermitteln. Dabei gibt $r \in \mathbb{N}$ die Anzahl der Achsen des zweiten Mechanismus an (im dargestellten Beispiel gilt $r = 1$), wobei diese kleiner sein muss als die Anzahl $m \in \mathbb{N}$ der insgesamt beim kinematisch redundanten Mechanismus vorhandenen Achsen ($r < m$). Aus

$\underline{q}_{T2,k}$ kann mittels des DKP wiederum der zugehörige Ortsvektor ${}_{\mathrm{Ow}}\vec{Z}_{\mathrm{T2},k}$ berechnet werden. Durch Umstellung von Gleichung (4-2) nach

$$ {}_{\mathrm{T2}}\vec{Z}_{\mathrm{TM},k} = {}_{\mathrm{Ow}}\vec{Z}_{\mathrm{TM},k} - {}_{\mathrm{Ow}}\vec{Z}_{\mathrm{T2},k} \tag{4-3} $$

kann mit ${}_{\mathrm{Ow}}\vec{Z}_{\mathrm{TM},k}$ und ${}_{\mathrm{Ow}}\vec{Z}_{\mathrm{T2},k}$ der Ortsvektor ${}_{\mathrm{T2}}\vec{Z}_{\mathrm{TM},k}$ des nicht-redundanten Teilmechanismus berechnet und damit wiederum unter Nutzung der Gleichungen des zugehörigen IKP die erforderlichen Antriebs- bzw. Gelenkpositionen $\underline{q}_{T1,k} \in \mathbb{R}^{m-r}$ ermittelt werden. Damit liegen alle erforderlichen Gelenkpositionen $\underline{q}_k \in \mathbb{R}^m$ für eine Sollposition ${}_{\mathrm{Ow}}\vec{Z}_{\mathrm{TM},k}$ des kinematisch redundanten Mechanismus vor.

Erste Ideen hierzu wurden bereits in [Blu16] erarbeitet, zudem wurde das Grundprinzip für den modellprädiktiven Ansatz im Rahmen von Vorveröffentlichungen unter anderem in [RBT16] vorgestellt. In Bild 4-2 ist es unter Betrachtung einer einzelnen Sollposition dargestellt: Zunächst werden durch die Optimierung initiale Werte für die Gelenkpositionen $\underline{q}_{T2,k}$ des zweiten Teilmechanismus vorgegeben ❶, welche mittels des DKP in den Ortsvektor ${}_{\mathrm{Ow}}\vec{Z}_{\mathrm{T2},k}$ überführt werden ❷. Hiermit und mit dem vorgegebenen Ortsvektor ${}_{\mathrm{Ow}}\vec{Z}_{\mathrm{TM},k}$ ❸ wird der Ortsvektor ${}_{\mathrm{T2}}\vec{Z}_{\mathrm{TM},k}$ ❹ berechnet. Die über das IKP des ersten Teilmechanismus ermittelten Gelenkpositionen $\underline{q}_{T1,k}$ werden zusammen mit dem Vektor $\underline{q}_{T2,k}$ in der Gütefunktion berücksichtigt ❺. Hintergrund ist, dass der Mechanismus erst dann in seiner Zielposition angekommen ist, wenn alle Gelenke ihre Zielposition erreicht haben. Den Wert J_k der Gütefunktion ❻ über eine geeignete Wahl von $\underline{q}_{T2,k}$ zu minimieren ist das Ziel des Optimierers. Konvergiert J_k zu einem Minimum (oder greift ein anderes Abbruchkriterium der Optimierung) werden die optimierten Gelenkpositionen $\underline{q}_k$ abschließend ausgegeben ❼.

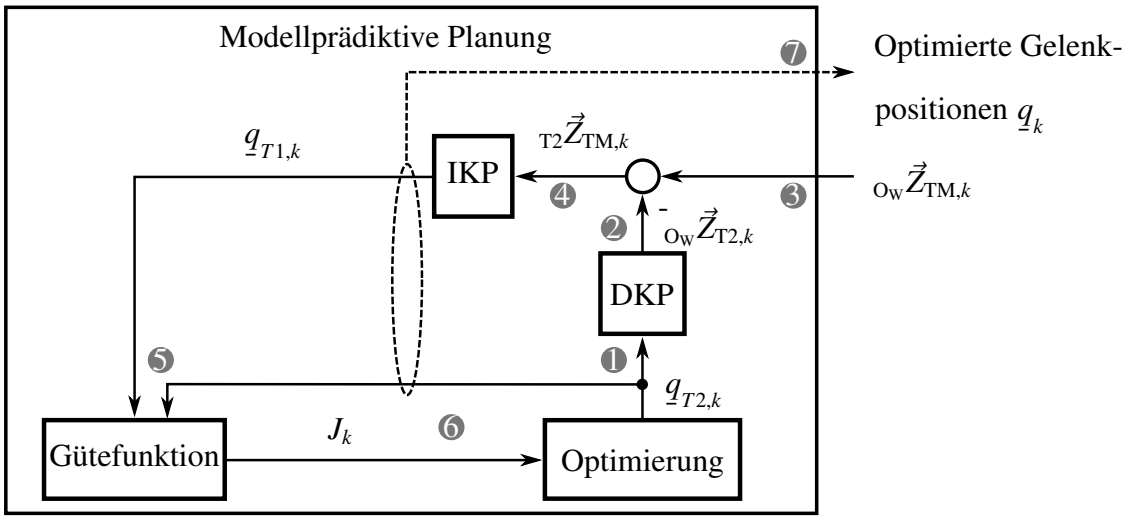


Bild 4-2: *Grundprinzip des modellprädiktiven Optimierungsansatzes bei einer Sollposition*

Für die Ermittlung der optimierten Antriebspositionen des zweiten Teilmechanismus werden bei dem modellprädiktiven Optimierungsansatz nicht nur die nächste Sollposition, sondern die folgenden $n_{\mathrm{p}} \in \mathbb{N}$ Positionen betrachtet. Dieser Bereich an zukünftigen Positionen wird auch als Prädiktionshorizont bezeichnet. Es wird hierdurch erreicht, dass

sich der Mechanismus beim Anfahren einer neuen Position unter Nutzung seiner überzähligen Bewegungsfreiheitsgrade bereits in eine „gute“ Konfiguration für die zukünftigen Sollpositionen bringt und somit eine insgesamt geringere Verfahrzeit erzielt wird. Der Hintergrund hierfür ist wie folgt: es gilt Ergebnisse für die nächste Sollposition zu vermeiden, die nur für diese bzw. die zugehörigen Bewegungen optimiert sind, aufgrund etwa von Achsgrenzen bei den folgenden Sollpositionen aber zu einer signifikanten Verschlechterung des Bewegungsverhaltens führen und somit über eine Folge von Positionen die erforderliche Verfahrzeit negativ beeinflussen. Ist eine optimierte Lösung für einen betrachteten Prädiktionshorizont ausgehend von der aktuellen Position $\vec{Z}_{\mathrm{TM},k}$ gefunden, wird das Ergebnis nur für das Anfahren der Position $\vec{Z}_{\mathrm{TM},k+1}$ genutzt. Danach wird der betrachtete Bereich um einen Schritt in die Zukunft verschoben und die Optimierung beginnt erneut. In Anlehnung an die Nomenklatur modellprädiktiver Regelungen wird dies als Prinzip des gleitenden Horizonts bezeichnet (vgl. [Dit04]). Hierdurch kann etwa auf zur Laufzeit auftretende Änderungen zukünftiger Sollpositionen reagiert werden. Die Ergebnisse der vorhergegangenen Optimierung ($\underline{q}_{T2,k+1} \dots \underline{q}_{T2,k+n_\mathrm{p}}$) können als Startwerte für die ersten $n_\mathrm{p} - 1$ Positionen genutzt werden. Der Prädiktionshorizont sowie das Prinzip des gleitenden Horizonts sind in Bild 4-3 dargestellt.

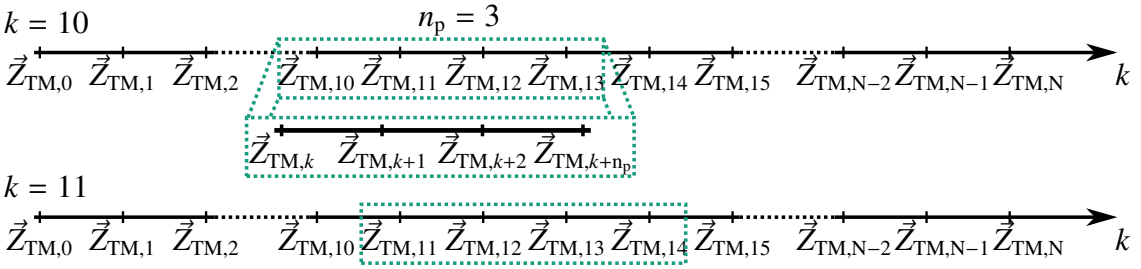


Bild 4-3: *Prinzip des gleitenden Horizonts mit* $n_\mathrm{p} = 3$

Zunächst wird der Bereich von $\vec{Z}_{\mathrm{TM},10}$ bis $\vec{Z}_{\mathrm{TM},13}$ betrachtet, wobei $k = 10$ gilt und somit $\vec{Z}_{\mathrm{TM},10}$ der aktuellen Position entspricht. Zudem wird ein Prädiktionshorizont von $n_\mathrm{p} = 3$ genutzt. Nach erfolgreicher Optimierung werden die entsprechenden Gelenkkoordinaten $\underline{q}_{,10}$ mittels der Gleichungen des IKP berechnet und der Bereich um einen Schritt in die Zukunft verschoben ($k = 11$), sodass die Positionen $\vec{Z}_{\mathrm{TM},11}$ bis $\vec{Z}_{\mathrm{TM},14}$ betrachtet werden. Dieses Vorgehen wird wiederholt, bis eine Folge an Sollpositionen vollständig durchlaufen ist ($k = N$). Es wird angenommen, dass der Mechanismus abschließend in der Position $\vec{Z}_{\mathrm{TM,N}}$ verharrt. Dadurch ergeben sich die letzten Sollpositionen einer Folge zu

$$\vec{Z}_{\mathrm{TM},k+\nu} = \vec{Z}_{\mathrm{TM,N}} \text{ für } k + \nu \geq N,\ k \leq N \text{ und } \nu = 1 \dots n_\mathrm{p} \in \mathbb{N}\,. \tag{4-4}$$

Es wird im Folgenden zunächst die der Optimierung zugrunde liegende Gütefunktion vorgestellt, bevor näher auf die eingesetzten Modelle eingegangen wird.

4.2 Gütefunktion und Optimierungsproblem

Für die zum Zweck der Verfahrzeitminimierung durchzuführende Optimierung wird eine Gütefunktion benötigt, die die Verfahrzeit zwischen der aktuellen Position $\vec{Z}_{\mathrm{TM},k}$ und der

folgenden Position $\vec{Z}_{\mathrm{TM},k+1}$, aber auch zwischen den weiteren Positionen $\vec{Z}_{\mathrm{TM},k+\nu}$ mit $\nu = 1 \ldots n_\mathrm{p} \in \mathbb{N}$ innerhalb des Prädiktionshorizonts n_p berücksichtigt. Die im aktuellen Zeitschritt zu minimierende Gütefunktion

$$\begin{aligned} J_k(\underline{q}_k, \underline{q}_{k+\nu|k}) = &\sum_{\nu=1}^{n_\mathrm{p}} w_1 \exp(-(\nu-1)) \max_{i=1\ldots m} \Delta t_i(\Delta q_{i,k+\nu|k}) \\ &+ w_2 p_{\mathrm{qmax}}(\underline{q}_{k+\nu|k}) \\ &+ w_3 p_{\mathrm{qmin}}(\underline{q}_{k+\nu|k}) \end{aligned} \tag{4-5}$$

beinhaltet die über $\exp(-(\nu-1))$ exponentiell abfallend und zusätzlich mit w_1 gewichteten Verfahrzeiten

$$\Delta t_i(\Delta q_{i,k+\nu|k}) \tag{4-6}$$

des i-ten Antriebs von insgesamt m Antrieben aus der Position $q_{i,k+\nu-1|k}$ um die Strecke

$$\Delta q_{i,k+\nu|k} = q_{i,k+\nu|k} - q_{i,k+\nu-1|k} \tag{4-7}$$

wobei $q_{i,k|k} = q_{i,k}$ gilt. Die Gelenkkoordinaten müssen dabei die Bedingung

$$\vec{Z}_{\mathrm{TM}}(\underline{q}_{k+\nu|k}) = \vec{Z}_{\mathrm{TM},k+\nu} \tag{4-8}$$

erfüllen. Durch die Maximumbildung innerhalb der Gütefunktion (4-5) wird für jede Sollposition nur die Verfahrzeit der langsamsten Achse berücksichtigt. Erst wenn diese Achse ihre Bewegung beendet hat, hat der TCP des Mechanismus die Sollposition $\vec{Z}_{\mathrm{TM},k+\nu}$ erreicht. Die Berechnung der Verfahrzeit wird im Abschnitt 4.3 vorgestellt. Durch die exponentielle Gewichtung hat die für weiter in der Zukunft liegende Sollpositionen erforderliche Verfahrzeit einen geringeren Einfluss auf die Gütefunktion. Gegebenenfalls auftretende Änderungen zukünftiger Sollpositionen wirken sich weniger stark negativ auf die Gesamtverfahrzeit einer Folge aus.

Über die zusätzlichen Terme

$$p_{\mathrm{qmax}}(\underline{q}_{k+\nu|k}) = \sum_{i=1}^{m} \exp\left(\left(\max_{\nu=1\ldots n_\mathrm{p}} (q_{i,k+\nu|k}) + q_{i,\mathrm{os}}\right) - q_{i,max}\right) \tag{4-9}$$

und

$$p_{\mathrm{qmin}}(\underline{q}_{k+\nu|k}) = \sum_{i=1}^{m} \exp\left(q_{i,min} - \left(\min_{\nu=i\ldots n_\mathrm{p}} (q_{i,k+\nu|k}) - q_{i,\mathrm{os}}\right)\right), \tag{4-10}$$

die auch als Strafterme (engl. penalty terms) bezeichnet werden, wird sichergestellt, dass die Aufteilung der Bewegungen unter Einhaltung der Gelenkgrenzen des Mechanismus erfolgt. Dazu wird die Differenz zwischen der Sollposition jedes Antriebs und seiner Maximal- bzw. Minimalposition berechnet und als Exponent der Exponentialfunktion genutzt. Liegt der Antrieb innerhalb seiner Grenzen, so wird der Exponent negativ und der Wert der Exponentialfunktion ist kleiner Eins. Wird die Grenze verletzt, so ist der Exponent größer Null und der Wert der Exponentialfunktion steigt entsprechend stark an. Hintergrund für die Wahl eines exponentiellen Strafterms ist der Umstand, dass „harte“

Grenzen und damit verbundene Unstetigkeiten der Gütefunktion zu Problemen bei der Optimierung führen können. Deswegen werden „weiche“ Grenzen, die ohne Sprung eingreifen, eingesetzt. Über den zusätzlichen Parameter $q_{i,\mathrm{os}}$ wird zudem ein Bereich vor den jeweiligen Grenzen definiert, bei dem der Exponent bereits ohne Verletzung der eigentlichen Maximal- bzw. Minimalposition positiv wird. Somit wird der Exponent bereits bei Sollpositionen innerhalb des Bereichs vor den Grenzen positiv. Je näher sie an der Grenze liegen, desto stärker werden sie durch den Verlauf der Exponentialfunktion bestraft. Die beiden Summen der jeweils m Exponentialterme werden mittels w_2 bzw. w_3 gewichtet. In Bild 4-4 ist der Verlauf der Werte von p_{qmin} und p_{qmax} für einen Antrieb M_i bei Variation der Antriebsposition q_i mit den in Tabelle 5-1 angegebenen Parametern, den Grenzen $q_{i,min} = 387{,}38\,\mathrm{mm}$ und $q_{i,max} = 487{,}38\,\mathrm{mm}$ sowie $q_{i,\mathrm{os}} = 5\,\mathrm{mm}$ und einer Gewichtung von $w_2 = w_3 = 20$ dargestellt. Es ist am Beispiel von p_{qmax} zu erkennen, dass dieser Wert oberhalb der Grenze von $q_{i,max} - q_{i,\mathrm{os}} = 482{,}38\,\mathrm{mm}$ entsprechend stark ansteigt. Erreicht die Antriebsposition die eigentlichen Verfahrgrenze, ist der Wert des Strafterms bereits im vierstelligen Bereich, wodurch er einen signifikanten Einfluss auf den Wert der Gütefunktion hat. Damit ist das Ziel einer „weichen“ sprungfreien Begrenzung der Antriebsposition erreicht.

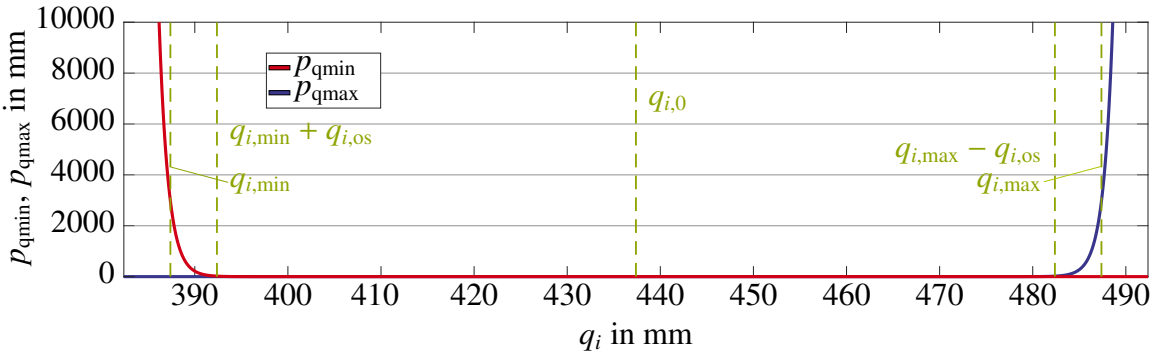


Bild 4-4: Verlauf der Werte von p_{qmin} und p_{qmax} für einen Antrieb M_i bei Variation der Antriebsposition q_i ($w_2 = w_3 = 20$)

Das insgesamt zu lösende Optimierungsproblem

$$\min_{\underline{q}_{T2,k+\nu|k}} J_k(\underline{q}_k, \underline{q}_{k+\nu|k}) \tag{4-11}$$

basiert wie bereits beschrieben auf dem Ansatz, die Antriebspositionen $\underline{q}_{T2,k+\nu|k}$ des zweiten Teilmechanismus für die Sollpositionen innerhalb des Prädiktionshorizontes durch den Optimierer derart bestimmen zu lassen, dass die gewichtete Summe der Verfahrzeiten des Gesamtmechanismus minimiert wird. Wie im Abschnitt 4.4 später genauer erläutert wird, wird zur Reduzierung des Rechenaufwandes auf eine Überwachung der Grenzen für Geschwindigkeit, Beschleunigung und Ruck in Form von Nebenbedingungen verzichtet, hier werden die Möglichkeiten moderner Antriebsumrichter genutzt.

In Bild 4-5 wird das Prinzip der modellprädiktiven Planung bei Berücksichtigung mehrerer zukünftiger Sollpositionen dargestellt. Der Optimierer gibt für alle Positionen $\vec{Z}_{\mathrm{TM},k} \ldots \vec{Z}_{\mathrm{TM},k+\mathrm{n_p}}$ innerhalb des Prädiktionshorizonts die Positionen $\underline{q}_{T2,k+\nu|k}$ des zweiten Teilmechanismus vor. Im ersten Schritt ($k = 1$) werden hierzu Initialwerte des Optimierers für $\underline{q}_{T2,k+\nu|k}$ mit $k = 1$ angestrebt, die keine der Nebenbedingungen verletzten dürfen, da ansonsten

der Optimierer nicht effizient funktioniert. Zusammen mit den Sollpositionen des TCP des Mechanismus werden mittels des IKP die Positionen $\underline{q}_{T1,k+\nu|k}$ der weiteren Antriebe des Mechanismus ermittelt. Aus diesen Verfahrwegen werden über ein Zeitmodell die Verfahrzeiten der sechs Antriebe je Sollposition berechnet und im Rahmen der Gütefunktion für die Berechnung des entsprechenden Wertes $J_k(\underline{q}_k, \underline{q}_{k+\nu|k})$ ausgehend von der aktuellen Position $\vec{Z}_{\mathrm{TM},k}$ des Mechanismus berechnet. Dieser Wert wiederum wird vom Optimierungsalgorithmus zur Ermittlung verbesserter, d.h. zu kürzeren Gesamtverfahrzeiten führenden, Sollpositionen des zweiten Teilmechanismus benötigt. Nachdem durch den Optimierer keine bessere Lösung ermittelt werden kann (Grenze für die minimale Änderung der Optimierungsgrößen oder des Wertes der Gütefunktion unterschritten, maximale Anzahl an erlaubten Iterationen erreicht), werden die Sollpositionen $\underline{q}_{T2,k+1|k}$ ausgegeben und für die Ansteuerung der einzelnen Antriebe genutzt. Entsprechend des Prinzips des gleitenden Horizonts wird der betrachtete Bereich nun um eine Sollposition in die Zukunft verschoben und der Optimierungsvorgang beginnt erneut. Hierbei werden bei allen $\vec{Z}_{\mathrm{TM},k}$ mit $k > 1$ für die ersten $n_\mathrm{p} - 1$ Sollpositionen die Ergebnisse des vorhergegangenen Optimierungsschrittes als Startwerte genutzt. Lediglich für die letzte Sollposition $\vec{Z}_{\mathrm{TM},k+\mathrm{n_p}}$ werden Initialwerte angenommen, die wie bei $k = 1$ so angenommen werden sollten, dass keine Nebenbedingungen verletzt werden.

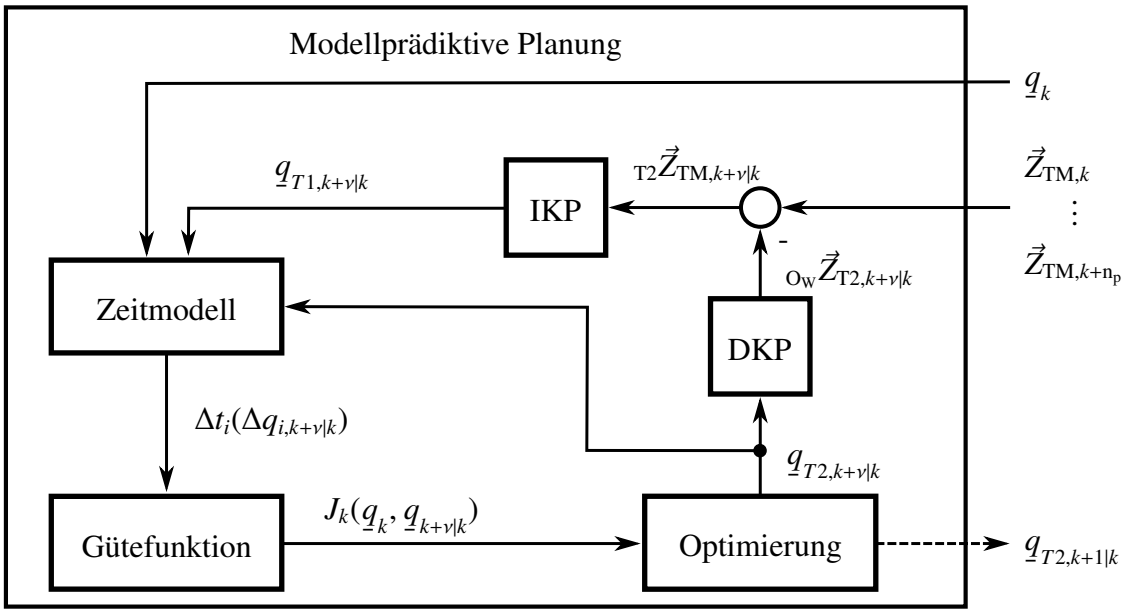


Bild 4-5: *Prinzip der modellprädiktiven Planung unter Betrachtung mehrerer zukünftiger Sollpositionen*

Zusammengefasst wird somit mittels des modellprädiktiven Planungsansatzes für jede Sollposition $\vec{Z}_{\mathrm{TM},k}$ mit $k = 1 \ldots N$ eine individuelle optimierte Aufteilung der insgesamt erforderlichen Achsbewegungen auf die Achsen des kinematisch redundanten Mechanismus ermittelt. Die Optimierung unter Nutzung der Mehrdeutigkeit des IKP erfolgt hinsichtlich der erforderlichen Verfahrzeit, wobei immer n_p zukünftige Sollpositionen im Prädiktionshorizont betrachtet werden. Durch eine exponentiell abfallende Gewichtung haben weiter in der Zukunft liegende Sollpositionen und ihre kurzfristigen Veränderungen einen

geringeren Einfluss auf die Gütefunktion. Neben der Verfahrzeit berücksichtigt die Gütefunktion die Verfahrbereichsgrenzen der Achsen. Die Überwachung der Grenzwerte etwa für Beschleunigungen oder Geschwindigkeiten wird hingegen auf die Antriebshardware ausgelagert.

Im folgenden Abschnitt wird das Zeitmodell für PTP-Bewegungen vorgestellt, welches die Berechnung der im Rahmen der Gütefunktion erforderlichen Verfahrzeiten ermöglicht.

4.3 Herleitung des Zeitmodells für Punkt-zu-Punkt-Bewegungen

Für den angestrebten zeitoptimierten Ansatz wird ein mathematisches Modell benötigt, mit dem die Verfahrzeiten der einzelnen Achsen M_i ermittelt werden können. Dabei werden an dieser Stelle nur asynchrone Punkt-zu-Punkt-Bewegungen betrachtet. Die Aufgabe des Zeitmodells liegt darin, für jede Achse M_i die Verfahrzeit $\Delta t_i(\Delta q_{i,k+\nu|k})$ zu berechnen, die für die Bewegungen zwischen der Position $\vec{Z}_{\mathrm{TM},k+\nu-1}$ und der folgenden Position $\vec{Z}_{\mathrm{TM},k+\nu}$ in Abhängigkeit der vorhergegangenen Positionen ab der aktuellen Position $\vec{Z}_{\mathrm{TM},k}$ erforderlich ist.

Aufgrund der geforderten Recheneffizienz wird wie etwa bei Haschke et al. in [HWR08] ein stark vereinfachtes Zeitmodell eingesetzt, das auf den mathematisch einfachen grundlegenden Gleichungen

$$v_i = a_i \cdot t_i + v_{i,0} \tag{4-12}$$

und

$$q_i = \frac{1}{2} \cdot a_i \cdot t_i^2 + v_{i,0} \cdot t_i + q_{i,0} \tag{4-13}$$

gleichmäßig beschleunigter Bewegungen basiert. Diese werden auch als *Grundgleichungen der Mechanik* bezeichnet ([Web07, S. 34]). Dabei gibt v_i die Verfahrgeschwindigkeit des Antriebs M_i, t_i die Verfahrzeit, q_i die Position, $v_{i,0}$ und $q_{i,0}$ die initiale Geschwindigkeit bzw. die Position zum Zeitpunkt $t = 0\,\mathrm{s}$ und a_i die Beschleunigung bzw. die Verzögerung an. Es ergibt sich ein Modell zweiter Ordnung mit den Begrenzungen

$$a_{i,\min} \leq a_i \leq a_{i,\max}, \qquad |v_i| \leq v_{i,\max} \tag{4-14}$$

bei dem

$$\dot{v}_i = a_i, \qquad \dot{q}_i = v_i \tag{4-15}$$

gilt.

Da eine zeitoptimierte Lösung gesucht wird, gilt nach dem Maximumprinzip von Pontryagin [SL12], dass die Beschleunigung bzw. Verzögerung abschnittsweise maximal oder null ist. Letzteres ist der Fall, wenn aufgrund einer hinreichend langen Verfahrstrecke die maximale Verfahrgeschwindigkeit erreicht wird. Für die Verwendung zur modellprädiktiven Planung zeitoptimierter Trajektorien ist dieses Modell ausreichend genau, zumal der eigentliche Positions- bzw. Geschwindigkeitsverlauf nicht von Bedeutung ist und lediglich die Verfahrzeit im Rahmen der Optimierung benötigt wird.

Das Verständnis von PTP-Bewegungen orientiert sich in der vorliegenden Arbeit an der in Kapitel 2.2 gegebenen Definition. Die Gesamtdauer

$$\begin{aligned}\Delta t_i(\Delta q_{i,k+\nu|k}) =& \Delta t_{i,\mathrm{acc}}(\Delta q_{i,k+\nu|k}) + \Delta t_{i,\mathrm{con}}(\Delta q_{i,k+\nu|k}) \\ &+ \Delta t_{i,\mathrm{dec}}(\Delta q_{i,k+\nu|k})\end{aligned} \tag{4-16}$$

einer solchen Bewegung des Antriebs M_i zwischen den Positionen $\vec{Z}_{\mathrm{TM},k+\nu-1}$ und $\vec{Z}_{\mathrm{TM},k+\nu}$ setzt sich dabei zusammen aus einer Zeit $\Delta t_{i,\mathrm{acc}}(\Delta q_{i,k+\nu|k})$ für die Beschleunigungsphase (engl. acceleration), einer Zeit $\Delta t_{i,\mathrm{con}}(\Delta q_{i,k+\nu|k})$ mit konstanter Verfahrgeschwindigkeit (engl. constant) und einer Verzögerungsphase der Dauer $\Delta t_{i,\mathrm{dec}}(\Delta q_{i,k+\nu|k})$ (engl. deceleration) [Web09]. Der Ruck wird als unendlich groß angenommen und im Folgenden vernachlässigt, dieses Vorgehen wird in der Literatur auch als Rampenprofil bezeichnet [Web09]. In den folgenden Bildern wird aus Gründen der besseren Lesbarkeit auf die explizite Kennzeichnung der Abhängigkeit von der Verfahrstrecke $\Delta q_{i,k+\nu|k}$ des Antriebs sowie die Indizierung mit $k+\nu|k$ zur Kennzeichnung der Zugehörigkeit zur Position an der Stelle $k+\nu$ in Abhängigkeit von der aktuellen Position k verzichtet. Es wird für die Darstellung vielmehr verallgemeinert angenommen, dass immer eine Bewegung zwischen einer Ausgangsposition $\vec{Z}_{\mathrm{TM},k}$ und einer Sollposition $\vec{Z}_{\mathrm{TM},k+1}$ erfolgt.

Für die Bewegung der Antriebe gilt es nun zu unterscheiden, ob der Verfahrweg $\Delta q_{i,k+\nu|k}$ lang genug ist, um die maximale Antriebsgeschwindigkeit $v_{i,\max}$ zu erreichen (Bild 4-6, a)) oder die gesamte Bewegung nur aus Beschleunigungs- und Verzögerungsphase besteht (Bild 4-6, b)).

Die minimal erforderliche Strecke

$$\Delta q_{i,\mathrm{b}} = \underbrace{\frac{1}{2}\cdot a_{i,\max}\cdot\left(\frac{v_{i,\max}}{a_{i,\max}}\right)^2}_{\Delta q_{i,\mathrm{acc}}} + \underbrace{\frac{1}{2}\cdot a_{i,\min}\cdot\left(\frac{v_{i,\max}}{a_{i,\min}}\right)^2 - v_{i,\max}\cdot\frac{v_{i,\max}}{a_{i,\min}}}_{\Delta q_{i,\mathrm{dec}}} = \frac{v_{i,\max}^2}{2}\cdot\left(\frac{1}{a_{i,\max}} - \frac{1}{a_{i,\min}}\right) \tag{4-17}$$

zum Erreichen der maximalen Verfahrgeschwindigkeit ist abhängig von der maximalen Beschleunigung $a_{i,\max}$ bzw. Verzögerung $a_{i,\min}$.

Für den Fall, dass $|\Delta q_{i,k+\nu|k}| > \Delta q_{i,\mathrm{b}}$ ist und somit $v_{i,\max}$ erreicht wird, gilt die Zeit

$$\Delta t_{i,\mathrm{I}}(\Delta q_{i,k+\nu|k}) = \frac{v_{i,\max}}{a_{i,\max}} + \frac{\Delta q_{i,\mathrm{con}}(\Delta q_{i,k+\nu|k})}{v_{i,\max}} - \frac{v_{i,\max}}{a_{i,\min}}\,, \tag{4-18}$$

wobei eine Strecke

$$\begin{aligned}\Delta q_{i,\mathrm{con}}(\Delta q_{i,k+\nu|k}) &= |\Delta q_{i,k+\nu|k}| - \Delta q_{i,\mathrm{acc}} - \Delta q_{i,\mathrm{dec}} \\ &= |\Delta q_{i,k+\nu|k}| - \frac{1}{2}\cdot a_{i,\max}\cdot\left(\frac{v_{i,\max}}{a_{i,\max}}\right)^2 - \frac{1}{2}\cdot a_{i,\min}\cdot\left(\frac{v_{i,\max}}{a_{i,\min}}\right)^2 \\ &= |\Delta q_{i,k+\nu|k}| - \Delta q_{i,\mathrm{b}}\end{aligned} \tag{4-19}$$

mit konstanter Geschwindigkeit $v_{i,\max}$ zurückgelegt wird. Durch Einsetzen der Gleichung (4-19) in Gleichung (4-18) ergibt sich der erste in Gleichung (4-22) dargestellte Fall.

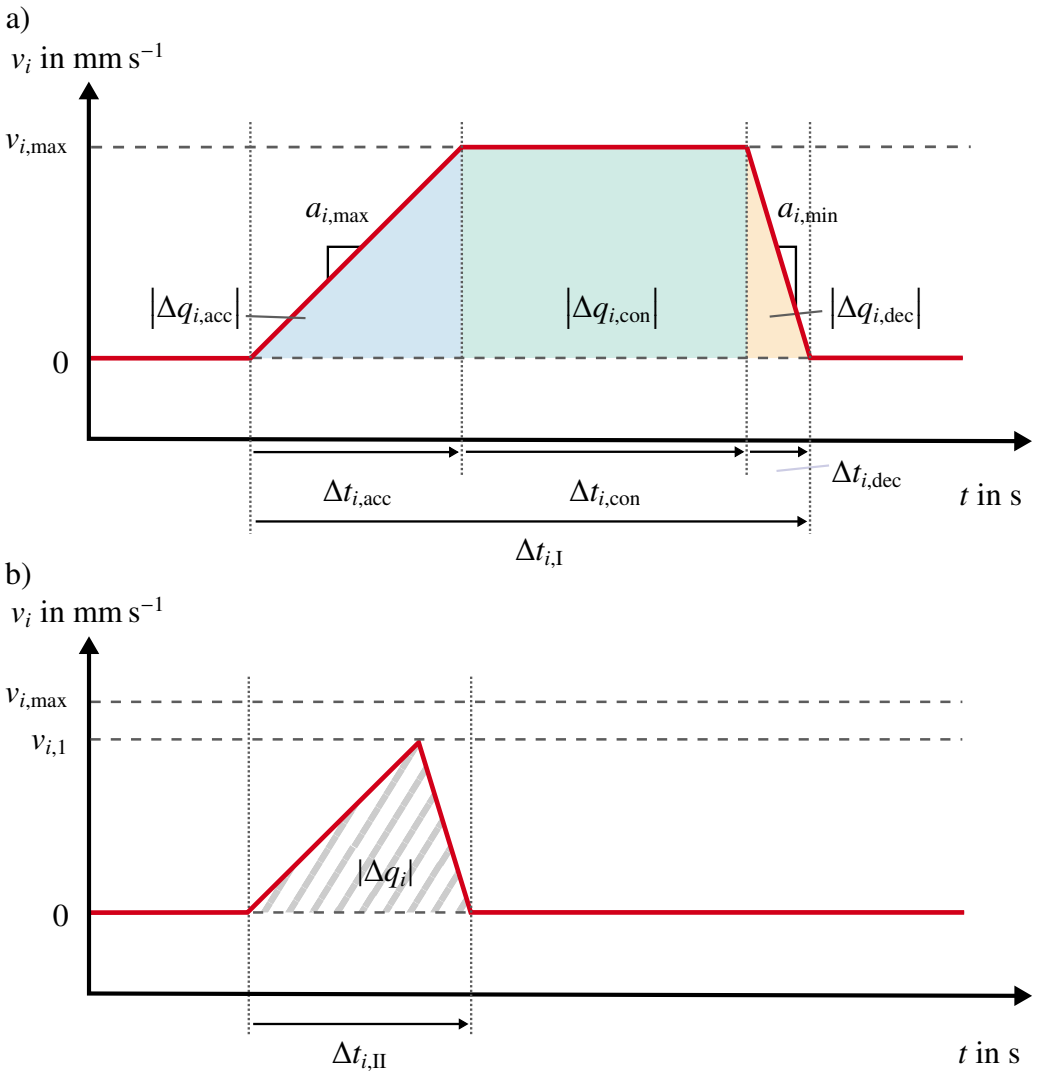


Bild 4-6: *Geschwindigkeitsverlauf des vereinfachten Antriebsmodells mit Erreichen a) und ohne Erreichen b) der Maximalgeschwindigkeit* $v_{i,\max}$

Ist die zurückzulegende Strecke nicht lang genug, um eine Phase der Bewegung mit konstanter Geschwindigkeit zu erreichen ($|\Delta q_{i,k+\nu|k}| \leq \Delta q_{i,\mathrm{b}}$, vgl. Bild 4-6, (b)), ergibt sich die erforderliche Zeit

$$\Delta t_{i,\mathrm{II}}(\Delta q_{i,k+\nu|k}) = \underbrace{\frac{v_{i,1}(\Delta q_{i,k+\nu|k})}{a_{i,\max}}}_{\Delta t_{i,\mathrm{acc}}} - \underbrace{\frac{v_{i,1}(\Delta q_{i,k+\nu|k})}{a_{i,\min}}}_{\Delta t_{i,\mathrm{dec}}}, \tag{4-20}$$

wobei

$$v_{i,1}(\Delta q_{i,k+\nu|k}) = \sqrt{\frac{2|\Delta q_{i,k+\nu|k}|}{\frac{1}{a_{i,\max}} - \frac{1}{a_{i,\min}}}} \tag{4-21}$$

die maximale innerhalb der Verfahrstrecke $\Delta q_{i,k+\nu|k}$ erreichbare Geschwindigkeit angibt.

Zusammengefasst ergibt sich somit das Zeitmodell für Punkt-zu-Punkt-Bewegungen zu

$$\Delta t_i(\Delta q_{i,k+\nu|k}) = \begin{cases} \Delta t_{i,\mathrm{I}}(\Delta q_{i,k+\nu|k}) & \text{für } |\Delta q_{i,k+\nu|k}| > \Delta q_{i,\mathrm{b}} \\ \Delta t_{i,\mathrm{II}}(\Delta q_{i,k+\nu|k}) & \text{für } |\Delta q_{i,k+\nu|k}| \leq \Delta q_{i,\mathrm{b}} \end{cases} . \tag{4-22}$$

Die Validierung dieses Modells erfolgt im Rahmen der Vorstellung eines kinematisch redundanten HKM in Kapitel 5, Abschnitt 5.3.

Im folgenden Abschnitt wird kurz auf die Nutzung der Möglichkeiten moderner Antriebshardware eingegangen, da dies thematisch eng mit dem Zeitmodell beziehungsweise dem zeitlichen Verhalten der Antriebe verbunden ist und zudem einen wichtigen Schritt in Richtung einer reduzierten Rechenzeit darstellt.

4.4 Nutzung des Potentials moderner Antriebshardware

Einen weiteren Schritt in Richtung Echtzeitfähigkeit bildet neben der Nutzung möglichst einfacher und recheneffizienter Modelle die Ausnutzung des Potentials moderner Antriebshardware bzw. Antriebsregler. Diese bieten häufig die Möglichkeit, neben dem Stromregler einen oder mehrere kaskadierte Regler auszuführen, und somit die SPS hiervon zu entlasten. Im Rahmen der vorliegenden Dissertationen werden sowohl der Positionsregler als auch die unterlagerten Regler für Geschwindigkeit und Strom auf dem Antriebsregler ausgeführt. Zudem werden die Werte hinsichtlich ihrer Grenzwerte überwacht, ein Schritt, der somit beispielsweise bezogen auf Ruck und Beschleunigung nicht als Nebenbedingung im Rahmen der Optimierung erfolgen muss. Da die Regelung zudem das Zeitverhalten der Antriebe beim Anfahren einer gegebenen Sollposition beeinflusst, wird an dieser Stelle kurz darauf eingegangen.

In Bild 4-7 ist beispielhaft die vereinfachte Regelstruktur eines Servoverstärkers aus der AX-5000-Serie der Firma Beckhoff [Bec18a] dargestellt. Von dem auf der SPS ausgeführten Programm wird eine absolute Sollposition des Antriebs an den sogenannten *NC-Profilgenerator* (NC: Numerical Control) übergeben, welcher diese auf Basis der voreingestellten Werte für Ruck, Beschleunigung und Sollgeschwindigkeit in ein entsprechendes Positionsprofil überführt. Hieraus werden wiederum zyklisch Sollpositionen mittels des echtzeitfähigen *EtherCAT*-Protokolls an den Servoverstärker übertragen. In diesem werden die Positions-, Geschwindigkeits- und Stromregler als Kaskade ausgeführt. Die einzelnen Regler können dabei anhand unterschiedlicher Parameter an die gegebenen Anforderungen angepasst werden (siehe Ausschnitte der drei Bedienoberflächen in Bild 4-7). Auf der SPS erfolgt lediglich die Generierung des Positionsprofils, von der eigentlichen Regelung wird die Steuerung entlastet. Für mehr Informationen hierzu wird beispielsweise auf [Bec18a] verwiesen.

Trotz der bei aktuellen SPS oder Industrie-PCs, auf denen die Funktionalitäten einer SPS als Software realisiert wird, immer weiter steigenden Rechenleistung wird insbesondere auch im Kontext der vielfältigen Einsatzmöglichkeiten der modellprädiktiven Planung zeitoptimierter Trajektorien eine derartige Entlastung der SPS als wichtiger Schritt in Richtung einer Echtzeitfähigkeit der Berechnung angesehen.

Bevor in Kapitel 6 die simulative Umsetzung des modellprädiktiven Ansatzes erläutert wird, erfolgt zum Abschluss dieses Kapitels die Vorstellung des genutzten Referenzverfahrens.

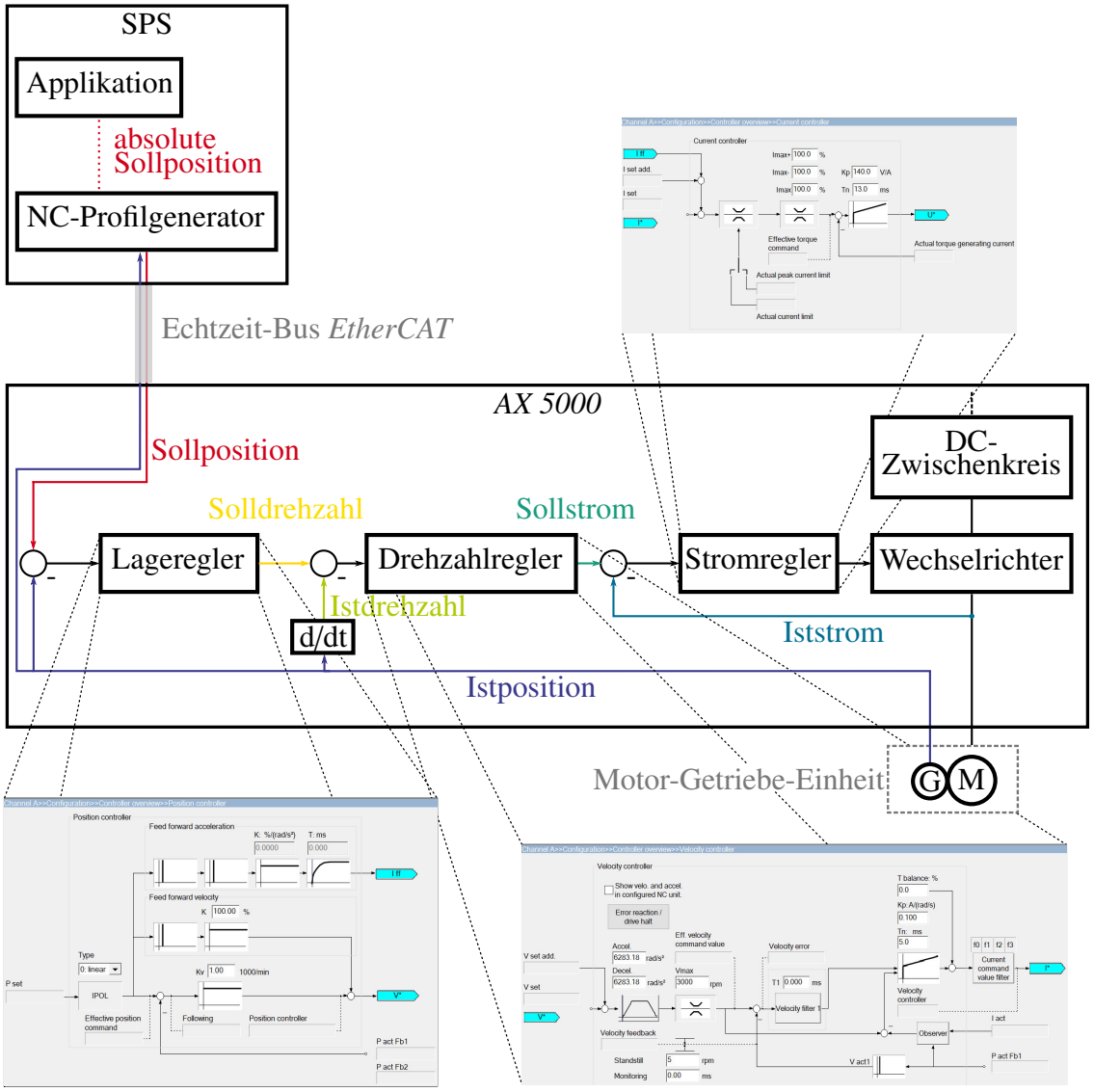


Bild 4-7: *Vereinfachte Reglerstruktur des AX5000 nach [Bec18a]*

Mit diesem werden alle weiteren im Rahmen dieser Arbeit erzielten Ergebnisse hinsichtlich der resultierenden Verfahrzeit verglichen.

4.5 Referenzverfahren

Um die in den folgenden Kapiteln vorgestellten Ergebnisse, die mit dem modellprädiktiven Planungsansatz sowohl simulativ in *MATLAB* als auch durch Messungen am realen Mechanismus erzielt wurden, einordnen und bewerten zu können, wird ein Vergleichsansatz benötigt. Bei diesem wird angenommen, dass über alle zu einer Folge gehörenden Sollpositionen eine konstante prozentuale Aufteilung der insgesamt durchzuführenden Bewegung auf die beiden Teilmechanismen erfolgt. Dies steht im Gegensatz zu dem modellprädiktiven Planungsansatz, bei dem für jede Sollposition eine optimierte Aufteilung ermittelt wird und bildet somit die Möglichkeit Vor- und Nachteile deutlich aufzuzeigen.

Bei der festen Aufteilung wird ein über eine gesamte Folge fester Anteil

$$_{\mathrm{OW}}\vec{Z}_{\mathrm{T2},k} = {}_{\mathrm{OW}}\vec{Z}_{\mathrm{TM},k} \circ \underline{\xi}_{\mathrm{ref}} \text{ mit } \underline{\xi}_{\mathrm{ref}} \in \mathbb{R}^l \tag{4-23}$$

der Sollposition durch den zweiten Teilmechanismus übernommen. Die Aufteilung $\underline{\xi}_{\mathrm{ref}}$ hat dabei die gleich Dimension $l \in \mathbb{N}$ wie der Ortsvektor des Mechanismus. Der Operator $\circ$ gibt das sogenannte *Hadamard-* oder *Schur-Produkt* zur elementweisen Multiplikation von Matrizen an [VA07, S. 13].

Die jeweils für eine gesamte Sollpositionsfolge geltende Aufteilung $\underline{\xi}_{\mathrm{ref}}$ wird mittels Optimierung festgelegt. Die dabei genutzte Gütefunktion

$$J_{\mathrm{ref}}(\underline{q}_0 \dots \underline{q}_N) = \sum_{k=0}^{N-1} \max_{i=1\dots6} \Delta t_i(\Delta q_{i,k+1|k}) \tag{4-24}$$

summiert die Verfahrzeiten der jeweils langsamsten Achse zwischen allen Sollpositionen $\vec{Z}_{\mathrm{TM,k}}$ mit $k = 1 \dots N$ ausgehen von der Initialposition $\vec{Z}_{\mathrm{TM},0}$ einer Folge. Nach

$$\begin{aligned} &\min_{\underline{\xi}_{\mathrm{ref}}} J_{\mathrm{ref}}(\underline{q}_0 \dots \underline{q}_N) \\ &\quad \text{s.t.} \\ &\quad 0 \leq \xi_{\mathrm{ref},l} \leq 1 \\ &\quad \underline{q}_{min} \leq \underline{q}_k \leq \underline{q}_{max} \end{aligned}$$

wird durch den Optimierer die Aufteilung der insgesamt für die Folge erforderlichen Gesamtverfahrzeit minimiert. Die Elemente von $\underline{\xi}_{\mathrm{ref}}$ können dabei zwischen dem Wert Null (keine Bewegung in diese Raumrichtung durch den zweiten Teilmechanismus) und Eins (komplette Bewegung in diese Raumrichtung durch den zweiten Teilmechanismus) liegen. Jedoch dürfen für keine Sollposition die Grenzen der einzelnen Antriebe verletzt werden.

Entgegen der vorgestellten modellprädiktiven Planung wird durch den Referenzansatz somit nicht die einzelne Bewegung zwischen zwei Positionen optimiert aufgeteilt, sondern eine konstante Aufteilung für eine komplette Positionsfolge derart gewählt, dass die insgesamt resultierende Verfahrzeit für die Folge minimiert wird. Es bildet somit näherungsweise den Fall ab, dass ein Prozessexperte mit gegebenenfalls jahrzehntelanger Erfahrung auf Basis seines Wissens eine feste Aufteilung festlegt bzw. diese im Rahmen eines Einfahr-/Einrichtprozesses iterativ ermittelt. Die Ermittlung der in den weiteren Analysen genutzten und auf der festen Aufteilung basierenden Referenzergebnissen wird im Rahmen der simulativen Adaption des Ansatzes auf einen kinematisch redundanten HKM in Kapitel 6, Abschnitt 6.2 beschrieben.

Im folgenden Kapitel werden ein kinematisch redundanter HKM des Fraunhofer IEM sowie die Adaption des modellprädiktiven Ansatzes auf diesen Mechanismus vorgestellt.

5 Anwendungsszenario 1: Der IEM-Mechanismus

Im Rahmen des vorliegenden Kapitels wird die Umsetzung der in Kapitel 4 vorgestellten modellprädiktiven Planung zeitoptimierter Trajektorien für kinematisch redundante mehrachsige Mechanismen auf einen realen Mechanismus vorgestellt. Erste Ansätze hierzu wurden bereits in der studentischen Arbeit [Blu16] sowie in [RBT16] präsentiert.

5.1 Adaption des Ansatzes

Bei dem kinematisch redundanten Mechanismus handelt es sich um eine Eigenentwicklung des Fraunhofer-Instituts für Entwurfstechnik Mechatronik (IEM). Er besteht aus einem Tripod mit den topologisch parallelen Antrieben M_1, M_2 und M_3, der mit einer zusätzlichen seriellen Achse M_4 versehen ist (siehe Bild 5-1) und nach [Sic99] als Tricept bezeichnet wird. Gegenüber SKM weist er den Vorteil auf, dass alle Antriebe außerhalb des Arbeitsraumes liegen und zudem durch die parallelkinematische Grundstruktur eine hohe Nutzlast bezogen auf das Eigengewicht und die Dimensionierung der Antriebe sowie eine hohe Steifigkeit vorliegen. Der HKM wurde ursprünglich für die Positionierung von Sensorik innerhalb eines schlanken, zylindrischen Arbeitsraumes mit innenliegenden konstruktions- und prozessbedingten Hindernissen entwickelt [RHWT15]. Während des Entwurfsprozesses wurden zunächst am Markt verfügbare Mechanismen modellbasiert analysiert. Dabei stellte sich heraus, dass diese die geforderte hohe Reichweite bei gleichzeitig möglichst großen Winkeln des TCP bezogen auf die Basis nicht erfüllen können [RHWT15]. Aus diesem Grund wurde eine modellbasierte Eigenentwicklung durchgeführt, deren Ergebnis der hier vorgestellte HKM ist. Die zusätzliche serielle Achse dient der Erhöhung der Reichweite des Mechanismus. Dazu ist diese fest mit der bewegten Plattform des Tripod verbunden, im Bereich der Basisplattform wird sie mittels eines Kardanschubgelenks geführt. Der TCP TT des HKM befindet sich am Ende der Achse M_4. Um den Arbeitsraum zusätzlich zu vergrößern, wurde der HKM mit der Basisplattform auf ein Flächenportal mit den Achsen M_5 und M_6 montiert, sodass der Ursprung O_T des Bezugskoordinatensystem des Tricept genau im TCP TP des Portals liegt (vgl. Bild 5-2). Das Symbol ≡ wird hier genutzt, um darzustellen, dass zwei Achsen oder die Ursprünge zweier unterschiedlicher Koordinatensystem identisch sind. Der TCP TM des Gesamtmechanismus wird am TCP TT des HKM verortet. Der resultierende Mechanismus, auch als *IEM-Mechanismus* bezeichnet, weist sechs Antriebe auf, wobei für $M_1 \dots M_3$ Hubspindelantriebe mit DC-Motoren der Fa. BST Eltromat (Datenblatt siehe Anhang A1-1), für M_4 ein DC-Servomotor der Fa. Beckhoff (Anhang A1-4) mit Zahnstangenritzel sowie für M_5 und M_6 Zahnriemenachsen, ebenfalls mit DC-Servomotoren der Fa. Beckhoff (Anhang A1-6 und A1-9), eingesetzt werden. Alle Achsen sind mit Absolutencodern ausgestattet, die Positionen der einzelnen Antriebe werden im Vektor

$$\underline{q} = [q_1\ q_2\ q_3\ q_4\ q_5\ q_6\ ]^T \in \mathbb{R}^6 \tag{5-1}$$

zusammengefasst.

Durch den Aufbau des Mechanismus kann nicht nur die Position sondern auch die Orientierung des TCP vorgegeben werden. Das Portal übernimmt dabei die Positionierung in x-

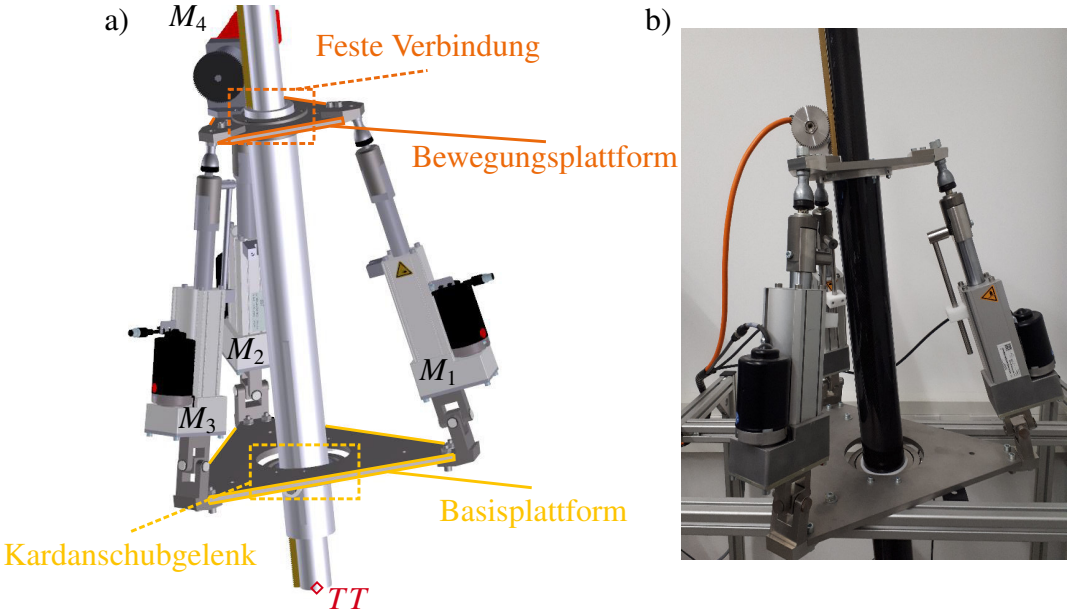


Bild 5-1: *Der Tricept: a) CAD-Darstellung, b) realer Tricept im Robotics Lab des Fraunhofer IEM*

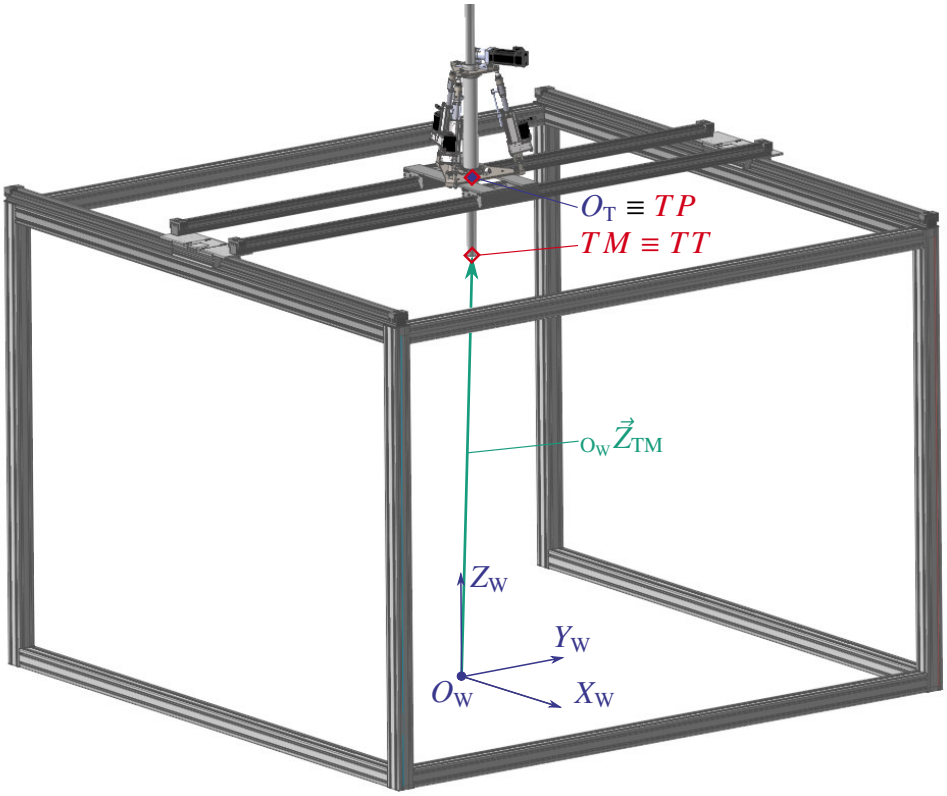


Bild 5-2: *CAD-Darstellung des Gesamtmechanismus aus der Mehrkörpersimulationssoftware RecurDyn*

und y-Richtung, während der Tricept durch den spezifischen Aufbau mittels der Achse M_4 die z-Koordinate, sowie aufgrund der Bewegung der Antriebe $M_1 \dots M_3$ die Kardanwinkel α und β beeinflusst (näheres dazu in Abschnitt 5.2.1). Der Mechanismus hat somit fünf DOF. Es gilt jedoch zu beachten, dass die Werte für α und β direkten Einfluss auf die x- und y-Position haben und somit keine unabhängige Vorgabe von Position und Orientierung bezogen auf den reinen Tricept möglich ist. Für den gesamten Mechanismus bedeutet dies, dass das Portal bei Vorgabe von α und β die daraus durch den Tricept resultierende Bewegung des TCP in x- und y-Richtung kompensieren muss.

In der vorliegenden Arbeit kann aufgrund des geplanten Anwendungsfalles beziehungsweise unter Einsatz eines aktiven Werkzeugkopfes (siehe Abschnitt 5.4) die Orientierung des TCP vernachlässigt werden. Es werden nur drei der fünf vorhandenen DOF genutzt, ein solcher Mechanismus wird als anwendungsbezogen kinematisch redundant bezeichnet (vgl. Kapitel 2.2). Die Position des TCP wird für den resultierenden Mechanismus durch den Positionsvektor

$$ {}_{\mathrm{OW}}\vec{Z}_{\mathrm{TM}} = {}_{\mathrm{OW}}\vec{Z}_{\mathrm{TT}} = \left[{}_{\mathrm{OW}}Z_{\mathrm{TM,x}} \; {}_{\mathrm{OW}}Z_{\mathrm{TM,y}} \; {}_{\mathrm{OW}}Z_{\mathrm{TM,z}} \right]^T \in \mathbb{R}^3 \tag{5-2} $$

bezogen auf ein kartesisches Bezugskoordinatensystem mit dem Ursprung O_{W} beschrieben (siehe Bild 5-2). Für die in Kapitel 4 vorgestellte modellprädiktive Planung wird der IEM-Mechanismus gedanklich in zwei Teilmechanismen unterteilt, den *Tricept* und das *Flächenportal*. Der Tricept stellt den ersten Teilmechanismus dar und kann Bewegungen des TCP in die drei Raumrichtungen ausführen. Das Portal muss diese in x- und y-Richtung nicht mehr ausgleichen, sondern führt als zweiter Teilmechanismus zur kinematischen Redundanz. Die Bewegungen beider Teilmechanismen können in x- und y-Richtung des Weltkoordinatensystems zu einer zielgerichteten Bewegung des TCP überlagert werden. Der Zustandsvektor des Gesamtmechanismus resultiert zu

$$ \underline{X} = \begin{bmatrix} {}_{\mathrm{OW}}\vec{Z}_{\mathrm{TM}} \\ {}_{O\mathrm{W}}\dot{\underline{Z}}_{TM} \end{bmatrix}. \tag{5-3} $$

Wie bereits erläutert, liegt der TCP TM des Gesamtmechanismus im TCP TT des Tricept, zudem befindet sich der Ursprung O_{T} des Tricept-Koordinatensystems im TCP TP des Portals. Das Tricept-feste Koordinatensystem weist dabei die gleiche Ausrichtung der Koordinatenachse auf wie das Welt-Koordinatensystem. Somit lässt sich unter der bereits angegebenen Vernachlässigung der Orientierung der Ortsvektor

$$ {}_{\mathrm{OW}}\vec{Z}_{\mathrm{TM}} = {}_{\mathrm{OT}}\vec{Z}_{\mathrm{TT}} + {}_{\mathrm{OW}}\vec{Z}_{\mathrm{TP}} \text{ mit } TP \equiv O_{\mathrm{T}} \tag{5-4} $$

des Gesamtmechanismus-TCPs durch Addition der Ortsvektoren der Teilmechanimus-TCPs ermitteln, wobei ${}_{\mathrm{OT}}\vec{Z}_{\mathrm{TT}}$ erforderlich ist, um mittels der Gleichungen des IKP des Tricept die Gelenkkoordinaten $q_1 \dots q_4$ zu berechnen. Dem vorgestellten Ansatz folgend werden die Positionen der Achsen q_5 und q_6 durch den Optimierer ermittelt. Mit diesen wiederum kann ${}_{\mathrm{OW}}\vec{Z}_{\mathrm{TP}}$ und hieraus mittels Gleichung (5-4) ${}_{\mathrm{OT}}\vec{Z}_{\mathrm{TT}}$ berechnet werden.

Es ergibt sich das zu lösende Optimierungsproblem

$$ \min_{q_{5,k+\nu|k}, q_{6,k+\nu|k}} J_k(\underline{q}_k, \underline{q}_{k+\nu|k}) \tag{5-5} $$

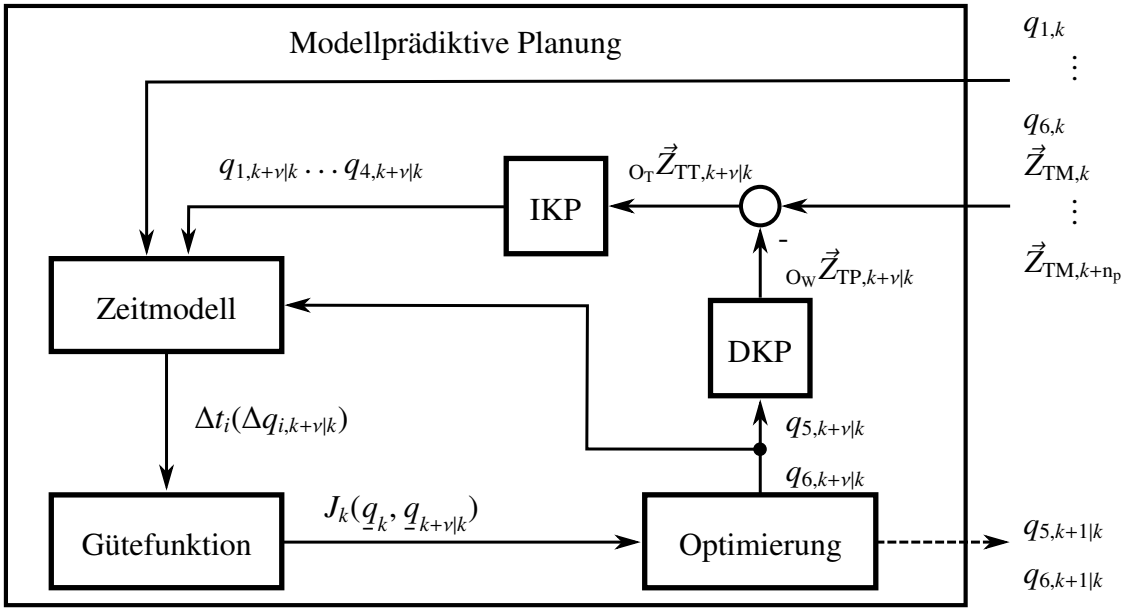


Bild 5-3: *Modellprädiktive Planung zeitoptimierter Trajektorien für den IEM-HKM*

mit der in Gleichung (4-5) beschriebenen Gütefunktion. Aufgrund der Anzahl der Antriebe ergibt sich dabei $m = 6$. In Bild 5-3 ist die resultierende Struktur des adaptierten Ansatzes dargestellt.

Für die Anwendung des modellprädiktiven Ansatzes wird im folgenden Abschnitt zunächst die Herleitung des IKP beschrieben. Zudem wird für das Portal als zweiter Teilmechanismus das DKP erläutert. Anschließend erfolgt die Vorstellung der simulativen Validierung des IKP des Gesamt-Mechanismus mittels Mehrkörpersimulation sowie die Validierung des Zeitmodells aus Abschnitt 4.3 anhand realer Antriebe. Zudem wird eine kurze Beschreibung möglicher Anwendungsszenarien für den Mechanismus gegeben.

5.2 Herleitung der Gleichungen der kinematischen Probleme

Im Folgenden wird die analytische Herleitung der IKP für die beiden Teilmechanismen und des DKP für das Portal sowie die simulative Validierung mittels Mehrkörpersimulation in der Software *RecurDyn* [Fun20] beschrieben.

5.2.1 Inverses kinematisches Problem des Tricept

Der erste Teilmechanismus ist der in Bild 5-4 dargestellte Tricept. Die drei parallelen Antriebe M_1, M_2 und M_3 des Tripods sind mit der Basisplattform über die Kardangelenke K_1, K_2 und K_3 sowie mit der bewegten Plattform über die Kugelgelenke (engl. spherical joint) S_1, S_2 und S_3 verbunden. Die zusätzliche Achse M_4 ist fest mit der bewegten Plattform des Tripod verbunden und im Bereich der Basisplattform durch das Kardanschubgelenk KS geführt. Dieses ermöglicht neben den spezifischen Freiheitsgraden eines Kardangelenks das Ein- und Ausfahren der seriellen Achse, verhindert aber eine Rotation um ihre Längsachse und stabilisiert damit die bewegte Plattform gegenüber der Basisplattform. Die

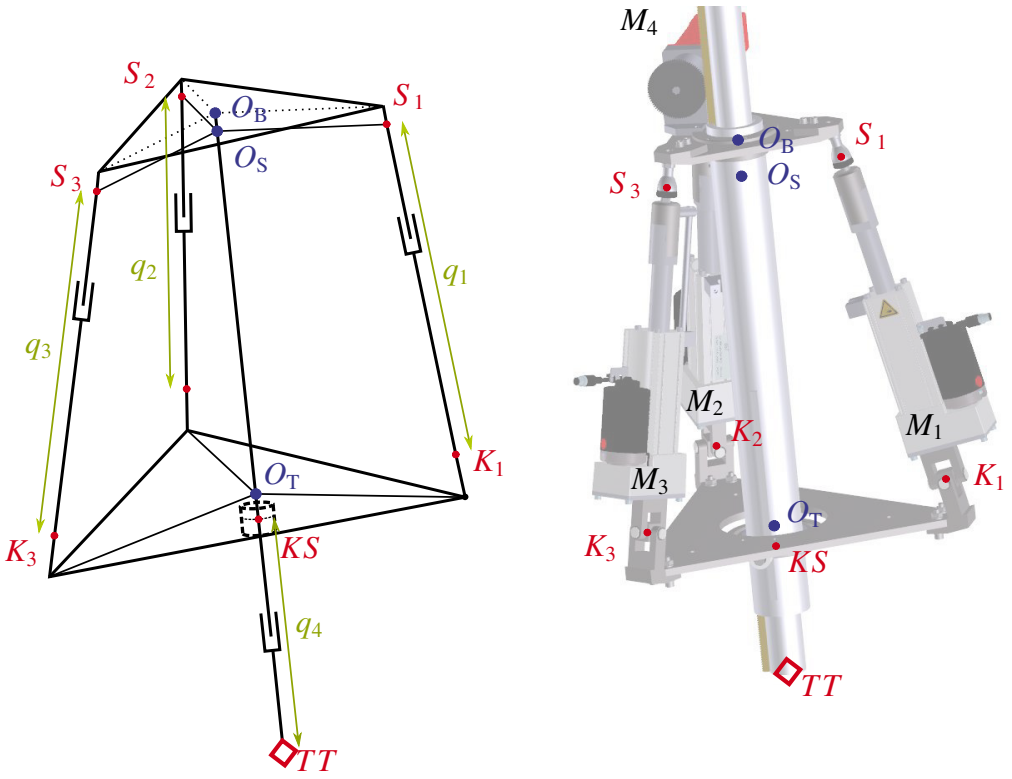


Bild 5-4: *IEM-Tricept (links: schematische Darstellung, rechts: Bild der Konstruktion)*

Position des Tricept-TCP TT lässt sich in kartesischen Koordinaten durch den Ortsvektor

$$ {}_{\mathrm{O_T}}\vec{Z}_{\mathrm{TT}} = \left[{}_{\mathrm{O_T}}Z_{\mathrm{TT,x}} \; {}_{\mathrm{O_T}}Z_{\mathrm{TT,y}} \; {}_{\mathrm{O_T}}Z_{\mathrm{TT,z}} \right]^T \in \mathbb{R}^3 \tag{5-6} $$

bezogen auf den Ursprung O_T des Tricept-festen Koordinatensystems beschreiben. Aufgrund des konstruktiven Aufbaus des Mechanismus und der Führung der vierten Achse mittels des Kardanschubgelenks kann die Position des TCP auch in Kugelkoordinaten (engl. Spherical coordinate, sp) mit den Winkeln α und β des Kardangelenks angegeben werden. Im Folgenden wird ${}_{\mathrm{O_T}}\vec{Z}_{\mathrm{TT}}$ stellvertretend für beide Arten der Positionsvorgabe verwendet, an entsprechenden Stellen findet eine Differenzierung statt. Zudem wird auf eine explizite Kennzeichnung des Bezugs verzichtet und stattdessen nur die Bezeichnung $\vec{Z}_{\mathrm{TT}}$ genutzt.

Bei der Ermittlung der inversen kinematischen Gleichung finden zwei Eigenschaften des Mechanismus Berücksichtigung: Zum einen sind die Antriebe M_1, M_2 und M_3 für die Verkippung sowie den Abstand zwischen bewegter Plattform und Basisplattform zuständig, während M_4 ausschließlich das Aus- und Einfahren des TCPs in Längsrichtung der Achse bewerkstelligt. Zum anderen gilt aufgrund der Konstruktion des Mechanismus für die Gelenkkoordinaten q_1, q_2 und q_3 der Antriebe der Parallelstruktur der in folgender Gleichung beschriebene Zusammenhang (vergleiche Bild 5-5):

$$ q_i(\vec{Z}_{\mathrm{TT}}) = \left| {}_{\mathrm{O_T}}\vec{r}_{\mathrm{S}_i}(\vec{Z}_{\mathrm{TT}}) - {}_{\mathrm{O_T}}\vec{r}_{\mathrm{K}_i} \right| \in \mathbb{R} \text{ mit } i = 1 \ldots 3 \, . \tag{5-7} $$

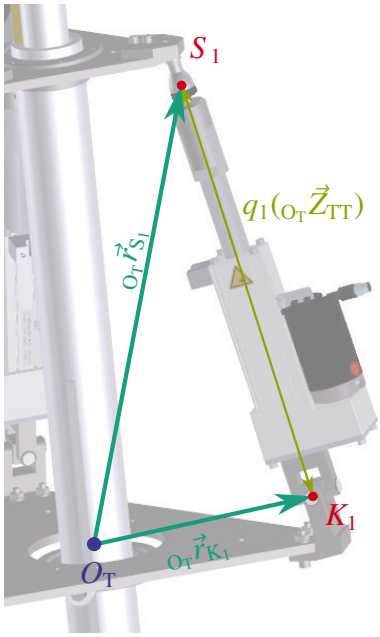


Bild 5-5: *Vektoren zur Ermittlung der Antriebslänge q_1 in Abhängigkeit von $\vec{Z}_{TT}$*

Sind somit die Positionen der Gelenke K_i sowie der von $\vec{Z}_{TT}$ abhängigen Gelenke S_i mit $i = 1 \ldots 3$ in Tricept-Koordinaten bekannt, können die Längen der Antriebe berechnet werden. Die Positionen der Gelenke bezogen auf die jeweilige Plattform (K_i bezogen auf den Mittelpunkt O_T der Basisplattform sowie S_i bezogen auf den Mittelpunkt O_S einer gedachten Ebene der Kugelgelenke[8]) sind bekannt. Die Gelenke liegen jeweils auf einem

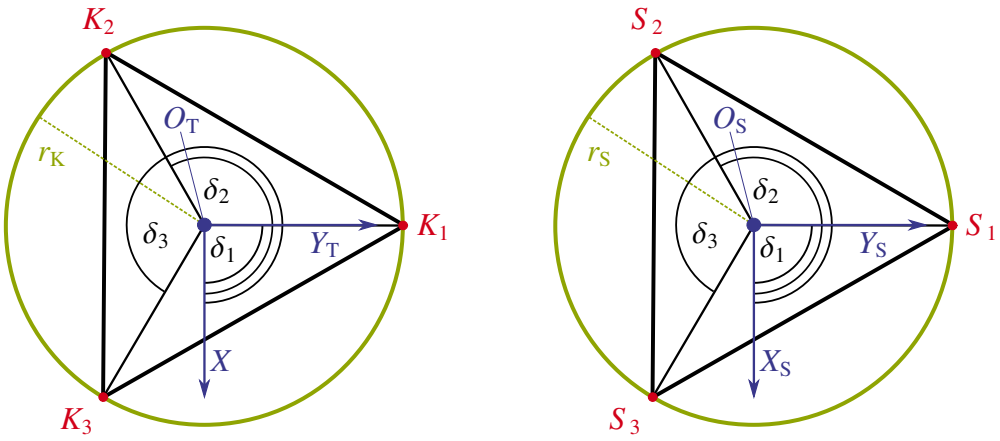


Bild 5-6: *Lage der Kardan- und Kugelgelenke bezogen auf den Mittelpunkt der jeweiligen Plattform*

gedachten Kreis mit dem Radius r_K bzw. r_S (vgl. Bild 5-6) und sind um jeweils $\Delta\delta = 120°$ zueinander verschoben, sodass sich bezogen auf die x-Achse die Winkel

$$\delta_1 = 90° \qquad \delta_2 = 210° \qquad \delta_3 = 330° \tag{5-8}$$

[8] Die Verschiebung zwischen der gedachten Ebene der Kugelgelenke mit dem Koordinatenursprung O_S und der eigentlichen bewegten Plattform mit dem Koordinatenursprung O_B hat keinen Einfluss auf die Berechnung.

ergeben. Für die Gelenke K_i mit $i = 1 \ldots 3$ resultiert daraus der Ortsvektor

$$ {}_{\mathrm{O_T}}\vec{r}_{\mathrm{K}_i} = \begin{bmatrix} r_{\mathrm{K}} \cdot \cos(\delta_i) \\ r_{\mathrm{K}} \cdot \sin(\delta_i) \\ s_{\mathrm{K}} \end{bmatrix} \in \mathbb{R}^3, \tag{5-9} $$

welcher sich direkt berechnen lässt. Der Parameter s_{K} gibt dabei den Abstand in z-Richtung ${}_{\mathrm{O_T}}Z_{\mathrm{TT,z}}$ des Tricept-Koordinatensystems zwischen dem Mittelpunkt des jeweiligen Kardangelenks K_i und dem Ursprung O_{T} an (die drei Kardangelenke weisen konstruktionsbedingt den gleichen konstanten Abstand auf, weswegen auf eine Unterscheidung verzichtet und stattdessen immer s_{K} verwendet werden kann).

Der Ortsvektor

$$ {}_{\mathrm{O_T}}\vec{r}_{\mathrm{S}_i} \tag{5-10} $$

lässt sich nicht direkt darstellen, vielmehr wird eine Transformationsmatrix ${}^{O_{\mathrm{T}},O_{\mathrm{S}}}\mathbf{T}$ zur Überführung des Ortsvektors

$$ {}_{\mathrm{O_S}}\vec{r}_{\mathrm{S}_i} = \begin{bmatrix} r_{\mathrm{S}} \cdot \cos(\delta_i) \\ r_{\mathrm{S}} \cdot \sin(\delta_i) \\ 0 \end{bmatrix} \in \mathbb{R}^3, \tag{5-11} $$

der im Koordinatensystem der gedachten Ebene der Kugelgelenke mit dem Ursprung O_{S} vorliegt, in das Tricept-Koordinatensystem mit Ursprung O_{T} benötigt.

Zur Ermittlung der Transformationsmatrix wird der Tricept in mehrere Teilsysteme unterteilt, wozu in den Gelenken zunächst Hilfskoordinatensysteme platziert und dann die entsprechenden Transformationsmatrizen zwischen diesen Koordinatensystemen ermittelt werden. Dabei wird auf eine in [Kol12] beschriebene Vorgehensweise zurückgegriffen. Diese sieht die Wahl der Hilfskoordinatensystemen in jedem Gelenk derart vor, dass Drehungen zwischen Körpern immer um die z-Achse und translatorische Verschiebungen aufgrund von translatorischen Achsen und Gelenken immer in positive Richtung der x-Achse des jeweiligen Hilfskoordinatensystems erfolgen. Bei der Wahl der Hilfskoordinatensysteme gilt es die positive Richtung der Achsen zu erhalten. Durch Befolgung dieser Konvention kann für die Drehung um den Winkel θ_s zwischen zwei durch ein rotatorisches Gelenk verbundenen Körpern $s-1$ und s mit den Koordinatenursprüngen O_{s-1} und O_s mit $s \in \mathbb{N}$ immer die gleiche Drehmatrix

$$ {}^{O_{s-1},O_s}\mathbf{A}(\theta_s) = {}^{O_{s-1},O_{s^*}}\mathbf{A} \cdot {}^{O_{s^*},O_s}\mathbf{A}(\theta_s) \in \mathbb{R}^{3\times3} \tag{5-12} $$

verwendet werden. Die erste Matrix

$$ {}^{O_{s-1},O_{s^*}}\mathbf{A} = \begin{bmatrix} \cos(x_{s-1}, x_{s^*}) & \cos(x_{s-1}, y_{s^*}) & \cos(x_{s-1}, z_{s^*}) \\ \cos(y_{s-1}, x_{s^*}) & \cos(y_{s-1}, y_{s^*}) & \cos(y_{s-1}, z_{s^*}) \\ \cos(z_{s-1}, x_{s^*}) & \cos(z_{s-1}, y_{s^*}) & \cos(z_{s-1}, z_{s^*}) \end{bmatrix} \in \mathbb{R}^{3\times3} \tag{5-13} $$

gibt die Verdrehung zwischen dem Ausgangskoordinatensystem mit dem Ursprung O_{s-1} und der Initialstellung des Koordinatensystems von Körper s mit dem Ursprung O_{s^*} an. Sie

resultiert aus den Vorgaben der Konvention, nach denen translatorische Verschiebungen zwischen Körpern in positive Richtung der x-Achse (siehe Bild 5-7 a)) und Drehungen um die z-Achse des Hilfskoordinatensystems (siehe Bild 5-7 b)) durchgeführt werden [Kol12]. Somit dürfen zwischen den Achsen der beiden Koordinatensysteme nur Verdrehungen um ein ganzzahliges Vielfaches von $\pi/2$ vorliegen, was einem Vertauschen der Achsen entspricht. Die Elemente der Matrix können somit nur die Werte -1, 0 und 1 annehmen.

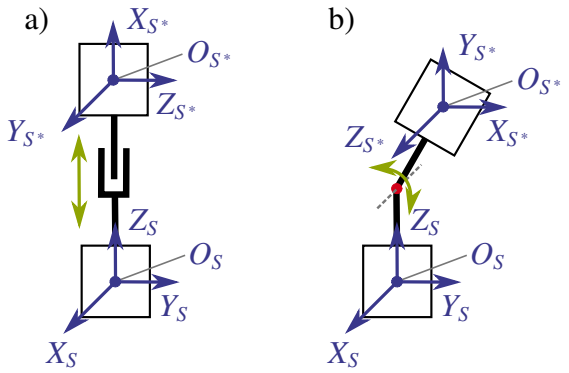


Bild 5-7: *Wahl des Hilfskoordinatensystems mit dem Ursprung O_{S^*}*

Die zweite Matrix

$$ {}^{O_{s^*},O_s}\mathbf{A}(\theta_s) = \begin{bmatrix} \cos(\theta_s) & -\sin(\theta_s) & 0 \\ \sin(\theta_s) & \cos(\theta_s) & 0 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{5-14} $$

kennzeichnet die Verdrehung um die z-Achse des Hilfskoordinatensystem mit dem Ursprung O_{s^*} (siehe z.B. [FF08]) um den Winkel θ_s zwischen den Koordinatensystemen mit den Ursprüngen O_{s^*} und O_s (die z-Achsen der beiden Koordinatensysteme bleiben dadurch deckungsgleich). Es ergibt sich somit die Rotationsmatrix nach Gleichung (5-12) durch Multiplikation der beiden Matrizen aus den Gleichungen (5-13) und (5-14).

Zusätzlich zu einer Verdrehung müssen mögliche Verschiebungen

$$ {}_{\mathrm{O}_{s-1}}\vec{r}_{\mathrm{O}_s}(\vec{Z}_{\mathrm{TT}}) = {}_{\mathrm{O}_{s-1}}\vec{r}_{\mathrm{O}_{s^*}}(\vec{Z}_{\mathrm{TT}}) = \underbrace{\begin{bmatrix} x_{s^*} - x_{s-1} \\ y_{s^*} - y_{s-1} \\ z_{s^*} - z_{s-1} \end{bmatrix}}_{I} + \underbrace{\begin{bmatrix} r_{\mathrm{t}}(\vec{Z}_{\mathrm{TT}}) \\ 0 \\ 0 \end{bmatrix}}_{II} \in \mathbb{R}^3 \tag{5-15} $$

berücksichtigt werden. Diese können zum einen aus der reinen Lage der Hilfskoordinatensysteme zueinander resultieren, etwa bei zwei durch einen Körper verbundenen rotatorischen Gelenken (Gleichung (5-15), I). Zum anderen liegen in dem betrachteten Tricept auch translatorische Gelenke vor, deren vom Ortsvektor $\vec{Z}_{\mathrm{TT}}$ abhängige Verschiebung $r_{\mathrm{t}}(\vec{Z}_{\mathrm{TT}})$ nach Konvention in Richtung der x-Achse des jeweiligen Hilfs-Koordinatensystems erfolgen muss (Gleichung (5-15), II). Um eine einzelne Transformationsmatrix zwischen

zwei Koordinatensystemen zu erhalten, wird die Rotationmatrix um eine vierte Zeile und Spalte erweitert (homogene Matrix)

$$ {}^{s-1,s}\mathbf{A}_h(\theta_s) = \begin{bmatrix} {}^{s-1,s}a_{1,1} & {}^{s-1,s}a_{1,2} & {}^{s-1,s}a_{1,3} & 0 \\ {}^{s-1,s}a_{2,1} & {}^{s-1,s}a_{2,2} & {}^{s-1,s}a_{2,3} & 0 \\ {}^{s-1,s}a_{3,1} & {}^{s-1,s}a_{3,2} & {}^{s-1,s}a_{3,3} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5-16} $$

und die Translation durch die Matrix

$$ {}^{s-1,s}\mathbf{R}_h(\vec{Z}_{\mathrm{TT}}) = \begin{bmatrix} 0 & 0 & 0 & {}_{s-1}r_{s,\mathrm{x}}(\vec{Z}_{\mathrm{TT}}) \\ 0 & 0 & 0 & {}_{s-1}r_{s,\mathrm{y}}(\vec{Z}_{\mathrm{TT}}) \\ 0 & 0 & 0 & {}_{s-1}r_{s,\mathrm{z}}(\vec{Z}_{\mathrm{TT}}) \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5-17} $$

dargestellt [FF08]. Nun kann durch Addition der Matrizen (5-16) und (5-17) die Transformationsmatrix

$$ {}^{s-1,s}\mathbf{T}(\theta, \vec{Z}_{\mathrm{TT}}) = {}^{s-1,s}\mathbf{A}_h(\theta_s) + {}^{s-1,s}\mathbf{R}_h(\vec{Z}_{\mathrm{TT}}) = \begin{bmatrix} & & & \\ & {}^{s-1,s}\mathbf{A}(\theta_s) & & {}_{s-1}\vec{r}_s(\vec{Z}_{\mathrm{TT}}) \\ & & & \\ 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \tag{5-18} $$

ermittelt werden.

Im Folgenden sind die Drehmatrizen und die Verschiebungen sowie die daraus resultierenden Transformationsmatrizen dargestellt und erläutert, wobei sich einzelne Schritte an der Vorgehensweise von Milutinović et al. in [Mil13] orientieren.

In Bild 5-8 ist der Tricept in mehrere Teilsysteme zerlegt und inklusive der zugehörigen Hilfskoordinatensysteme dargestellt. Zunächst wird der Abstand zwischen dem Maschinenkoordinatensystem des Tricepts, dessen Ursprung O_{T} im Mittelpunkt der Basisplatte liegt, und dem Koordinatensystem des Kardanschubgelenks mit dem Ursprung O_{KS} betrachtet. Es findet lediglich eine Verschiebung und keine Rotation statt, es ergeben sich somit die Rotationsmatrizen

$$ {}^{O_{\mathrm{T}},O^*_{\mathrm{KS}}}\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-19} $$

und

$$ {}^{O^*_{\mathrm{KS}},O_{\mathrm{KS}}}\mathbf{A}(0) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-20} $$

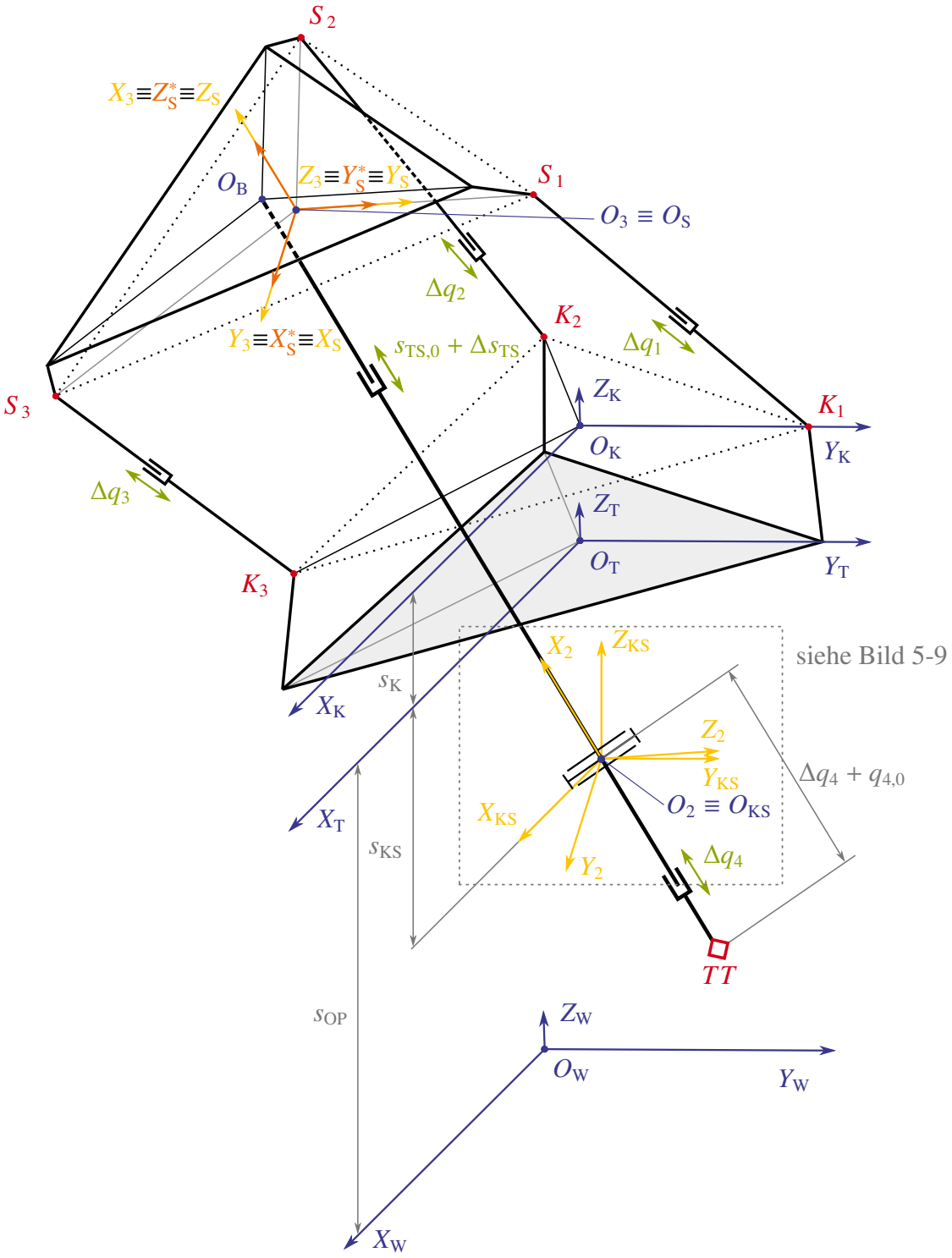


Bild 5-8: *Skizze des Tricept mit den erforderlichen Hilfskoordinatensystemen*

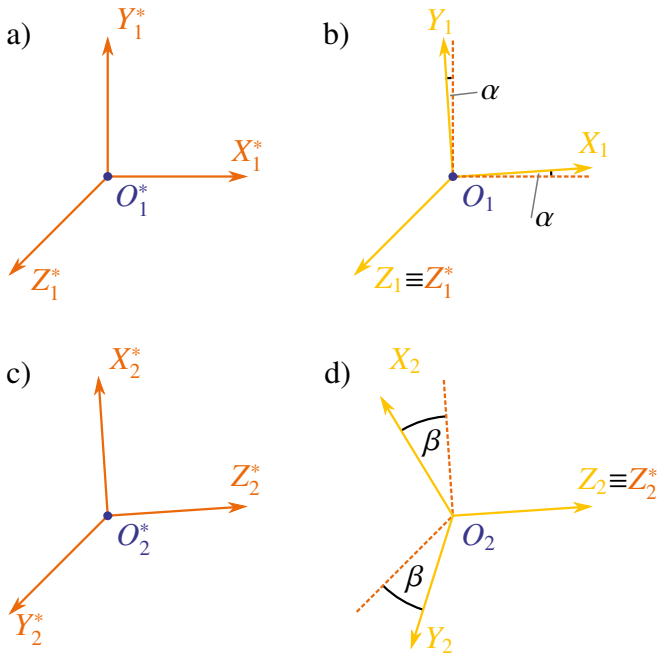


Bild 5-9: *Skizze des Kardanschubgelenks mit den erforderlichen Hilfskoordinatensystemen*

sowie die Translation

$$ {}_{O_T}\vec{r}_{O_{KS}} = \begin{bmatrix} 0 \\ 0 \\ s_{KS} \end{bmatrix} . \tag{5-21} $$

Da es sich um eine konstruktionsbedingte Verschiebung und nicht um die Bewegung eines Gelenks handelt, muss diese nicht in Richtung der x-Achse des Hilfs-Koordinatensystems erfolgen (siehe Gleichung (5-15), I)). Es ergibt sich die Transformationsmatrix

$$ {}_{O_T,O_{KS}}\mathbf{T}(0) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & s_{KS} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5-22} $$

zwischen den beiden Koordinatensysteme.

Als nächstes wird das Kardanschubgelenk als zwei rotatorische Gelenke mit jeweils nur einem rotatorischen Freiheitsgrad angesehen und durch zwei Hilfskoordinatensysteme abgebildet (siehe Bild 5-9). Die Winkel des Kardanschubgelenks werden als α und β bezeichnet. Die translatorische Komponente wird im Rahmen eines dritten Hilfs-Koordinatensystems berücksichtigt. Als erstes wird die Rotation um die x-Achse (Winkel α) berücksichtigt. Da die Rotation nach der vorgestellten Konvention um die z-Achse des Hilfskoordinatensystems erfolgen muss, ergibt sich die erste Rotationsmatrix zu

$$ {}_{O_{KS},O_1^*}\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{5-23} $$

(siehe Bild 5-9, a) und die zweite Matrix unter Berücksichtigung des Winkels α zu

$$ {}^{1^*,1}\mathbf{A}(\alpha) = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-24} $$

(siehe Bild 5-9, b). Es liegt keine Verschiebung vor, sodass sich der Translationsvektor

$$ {}_{\mathrm{O_{KS}}}\vec{r}_{\mathrm{O_1}} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{5-25} $$

ergibt. Insgesamt folgt daraus die Transformationsmatrix

$$ {}_{O_{\mathrm{KS}},O_1}\mathbf{T}(\alpha) \begin{bmatrix} 0 & 0 & 1 & 0 \\ \cos(\alpha) & -\sin(\alpha) & 0 & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5-26} $$

zwischen dem Koordinatensystem des Kardanschubgelenks und dem ersten Hilfskoordinatensystem. Im folgenden Schritt wird die Rotation um die x-Achse des ersten Hilfs-Koordinatensystems (Winkel β) berücksichtigt (siehe Bild 5-9, c und d):

$$ {}_{O_1,O_2^*}\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{5-27} $$

$$ {}_{O_2^*,O_2}\mathbf{A}(\beta) = \begin{bmatrix} \cos(\beta) & -\sin(\beta) & 0 \\ \sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{5-28} $$

Da sowohl das erste als auch das zweite Hilfskoordinatensystem zu Komponenten des Kardanschubgelenks gehören und sich die Rotationsachsen in einer Ebene befinden, liegt mit

$$ {}_{\mathrm{O_1}}\vec{r}_{\mathrm{O_2}} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{5-29} $$

keine Verschiebung vor. Daraus folgt die Transformationsmatrix

$$ {}_{O_1,O_2}\mathbf{T}(\beta) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ \cos(\beta) & -\sin(\beta) & 0 & 0 \\ \sin(\beta) & \cos(\beta) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5-30} $$

Im nächsten Schritt wird der Schubanteil des Kardanschubgelenks berücksichtigt, wobei dieser durch eine Verschiebung zwischen der Basisplattform und der bewegten Plattform entsteht. Für diese Bewegung wird ein Schubgelenk zwischen beiden Plattformen angenommen, dessen Verschiebung nach Konvention in x-Richtung des zugehörigen Hilfskoordinatensystems mit dem Ursprung O_3 (welches in der gedachten Ebene der Kugelgelenke mit Ursprung O_S platziert ist) erfolgen muss. Die aus den beiden Matrizen

$$ {}^{O_2,O_3^*}\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-31} $$

und

$$ {}^{O_3^*,O_3}\mathbf{A} = \begin{bmatrix} \cos(0) & -\sin(0) & 0 \\ \sin(0) & \cos(0) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-32} $$

resultierende Drehmatrix führt zusammen mit der Verschiebung

$$ {}_{O_2}\vec{r}_{O_3}(\vec{Z}_{TT}) = \begin{bmatrix} s_{TS,0} \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} \Delta s_{TS}(\vec{Z}_{TT}) \\ 0 \\ 0 \end{bmatrix} \tag{5-33} $$

zur Transformationsmatrix

$$ {}^{O_2,O_3}\mathbf{T}(\vec{Z}_{TT}) = \begin{bmatrix} 1 & 0 & 0 & s_{TS,0} + \Delta s_{TS}(\vec{Z}_{TT}) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5-34} $$

mit

$$ s_{TS,0} = \left| {}_{O_T}\vec{r}_{O_S} \right| \text{, wenn gilt } q_1 = q_{1,0},\ q_2 = q_{2,0},\ q_3 = q_{3,0}\ . \tag{5-35} $$

Dabei gibt $s_{TS,0}$ den Abstand zwischen Basisplattform und der Ebene der Kugelgelenke an, der sich einstellt, wenn die drei parallelen Antriebe $M_1 \ldots M_3$ in ihrer Initialstellung $q_{i,0}$ sind (vgl. Bild 5-10). Der Parameter $\Delta s_{TS}(\vec{Z}_{TT})$ entsteht durch das Verfahren der drei Antriebe in Abhängigkeit der gewählten Sollposition und der daraus resultierenden Änderung des Abstands zwischen Basis- und Bewegungsplattform, sowie der damit fest verbundenen Kugelgelenke.

Der Ursprung O_3 des dritten Hilfskoordinatensystems befindet sich im Ursprung O_S. Um diesen Punkt herum befinden sich mit einem Abstand r_S, der dem Radius eines gedachten Kreises mit dem Ursprung O_S entspricht, die drei Kugelgelenke zur Verbindung der Antriebe mit der Bewegungsplattform (vgl. Bild 5-6). Um vom letzten Hilfskoordinatensystem in

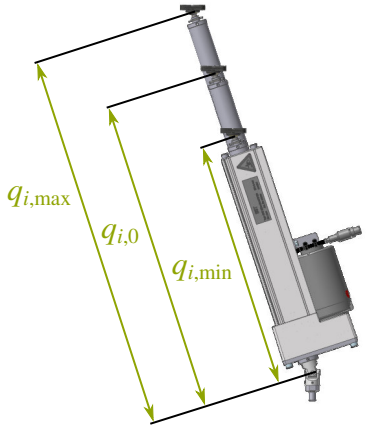


Bild 5-10: *Antrieb in Initialstellung $q_{i,0}$ sowie maximal aus- bzw. eingefahren ($q_{i,\max}$, $q_{i,\min}$)*

das Koordinatensystem der Ebene der Kugelgelenke zu gelangen, muss eine Verdrehung gemäß der Matrix

$$ {}^{O_3,O_S^*}\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{5-36} $$

erfolgen. Es liegt an dieser Stelle kein Drehgelenk vor, sodass die Matrix

$$ {}^{O_S^*,O_S}\mathbf{A} = \begin{bmatrix} \cos(0) & -\sin(0) & 0 \\ \sin(0) & \cos(0) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-37} $$

konstant ist. Da zudem kein Schubgelenk vorliegt ergibt sich der Verschiebungsvektor

$$ {}_{O_3}\vec{r}_{O_S} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \tag{5-38} $$

womit die Transformationsmatrix zu

$$ {}^{O_3,O_S}\mathbf{T} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5-39} $$

resultiert. Durch die Matrizenmultiplikation

$$ {}^{O_T,O_S}\mathbf{T}(\vec{Z}_{TT}) = {}^{O_T,O_{KS}}\mathbf{T}(0) \cdot {}^{O_{KS},O_1}\mathbf{T}(\alpha) \cdot {}^{O_1,O_2}\mathbf{T}(\beta) \cdot {}^{O_2,O_3}\mathbf{T}(\vec{Z}_{TT}) \cdot {}^{O_3,O_S}\mathbf{T}(0) \tag{5-40} $$

der einzelnen Transformationsmatrizen ergibt sich die Transformationsmatrix

$$
{}^{O_\mathrm{T},O_\mathrm{S}}\mathbf{T}(\vec{Z}_\mathrm{TT}) =
$$

$$
\begin{bmatrix}
\cos(\beta) & 0 & \sin(\beta) & \sin(\beta)(s_{\mathrm{TS},0} + \Delta s_\mathrm{TS}(\vec{Z}_\mathrm{TT})) \\
\sin(\alpha)\sin(\beta) & \cos(\alpha) & -\sin(\alpha)\cos(\beta) & -\sin(\alpha)\cos(\beta)(s_{\mathrm{TS},0} + \Delta s_\mathrm{TS}(\vec{Z}_\mathrm{TT})) \\
-\cos(\alpha)\sin(\beta) & \sin(\alpha) & \cos(\alpha)\cos(\beta) & \cos(\alpha)\cos(\beta)(s_{\mathrm{TS},0} + \Delta s_\mathrm{TS}(\vec{Z}_\mathrm{TT})) + s_\mathrm{KS} \\
0 & 0 & 0 & 1
\end{bmatrix}
\tag{5-41}
$$

zwischen dem Ursprung O_T des triceptfesten Koordinatensystems und dem Ursprung O_S als Mittelpunkt der Ebene der Kugelgelenke. Mit Hilfe dieser Transformationsmatrix kann nun die Länge der Antriebe berechnet werden, wozu jedoch die Vektoren aus den Gleichungen (5-9) und (5-11) als homogene Koordinaten dargestellt werden müssen. Nach [FF08] ergeben sich aus den Gleichungen (5-9) und (5-11) die Vektoren

$$
{}_{O_\mathrm{T}}\vec{r}_{\mathrm{K}_i} = \begin{bmatrix} r_\mathrm{K} \cdot \cos(\delta_i) \\ r_\mathrm{K} \cdot \sin(\delta_i) \\ s_\mathrm{K} \\ 1 \end{bmatrix} \in \mathbb{R}^4 \tag{5-42}
$$

sowie

$$
{}_{O_\mathrm{S}}\vec{r}_{\mathrm{S}_i} = \begin{bmatrix} r_\mathrm{S} \cdot \cos(\delta_i) \\ r_\mathrm{S} \cdot \sin(\delta_i) \\ 0 \\ 1 \end{bmatrix} \in \mathbb{R}^4, \tag{5-43}
$$

mit denen die Gelenkkoordinaten

$$
q_1(\vec{Z}_\mathrm{TT}) = \left| {}^{O_\mathrm{T},O_\mathrm{S}}\mathbf{T}(\vec{Z}_\mathrm{TT}) {}_{O_\mathrm{S}}\vec{r}_{\mathrm{S}_1} - {}_{O_\mathrm{T}}\vec{r}_{\mathrm{K}_1} \right| , \tag{5-44}
$$

$$
q_2(\vec{Z}_\mathrm{TT}) = \left| {}^{O_\mathrm{T},O_\mathrm{S}}\mathbf{T}(\vec{Z}_\mathrm{TT}) {}_{O_\mathrm{S}}\vec{r}_{\mathrm{S}_2} - {}_{O_\mathrm{T}}\vec{r}_{\mathrm{K}_2} \right| , \tag{5-45}
$$

$$
q_3(\vec{Z}_\mathrm{TT}) = \left| {}^{O_\mathrm{T},O_\mathrm{S}}\mathbf{T}(\vec{Z}_\mathrm{TT}) {}_{O_\mathrm{S}}\vec{r}_{\mathrm{S}_3} - {}_{O_\mathrm{T}}\vec{r}_{\mathrm{K}_3} \right| \tag{5-46}
$$

berechnet werden (siehe Gleichung (5-7)). Im Folgenden wird auf die explizite Kennzeichnung der Abhängigkeit von $\vec{Z}_\mathrm{TT}$ verzichtet, um die Lesbarkeit zu verbessern.

Aus den Gleichungen (5-44) ergeben sich die Längen der drei Antriebe zu

$$
\begin{aligned}
q_1 = \sqrt{(\sin(\delta_1)\sin(\alpha)r_\mathrm{S} - \cos(\delta_1)\cos(\alpha)r_\mathrm{S}\sin(\beta) + s_\mathrm{KS} + \cos(\alpha)\cos(\beta)(s_{\mathrm{TS},0} + \Delta s_\mathrm{TS}) - s_\mathrm{K})^2} \\
\overline{+(\sin(\delta_1)\cos(\alpha)r_\mathrm{S} + \cos(\delta_1)\sin(\alpha)r_\mathrm{S}\sin(\beta) - \sin(\alpha)\cos(\beta)(s_{\mathrm{TS},0} + \Delta s_\mathrm{TS}) - \sin(\delta_1)r_\mathrm{K})^2} \\
\overline{+(\cos(\delta_1)r_\mathrm{S}\cos(\beta) + \sin(\beta)(s_{\mathrm{TS},0} + \Delta s_\mathrm{TS}) - \cos(\delta_1)r_\mathrm{K})^2}
\end{aligned} ,
\tag{5-47}
$$

$$q_2 = \sqrt{\begin{array}{l}(\sin(\delta_2)\sin(\alpha)r_S - \cos(\delta_2)\cos(\alpha)r_S\sin(\beta) + s_{KS} + \cos(\alpha)\cos(\beta)(s_{TS,0} + \Delta s_{TS}) - s_K)^2 \\ +(\sin(\delta_2)\cos(\alpha)r_S + \cos(\delta_2)\sin(\alpha)r_S\sin(\beta) - \sin(\alpha)\cos(\beta)(s_{TS,0} + \Delta s_{TS}) - \sin(\delta_2)r_K)^2 \\ +(\cos(\delta_2)r_S\cos(\beta) + \sin(\beta)(s_{TS,0} + \Delta s_{TS}) - \cos(\delta_2)r_K)^2\end{array}} \tag{5-48}$$

und

$$q_3 = \sqrt{\begin{array}{l}(\sin(\delta_3)\sin(\alpha)r_S - \cos(\delta_3)\cos(\alpha)r_S\sin(\beta) + s_{KS} + \cos(\alpha)\cos(\beta)(s_{TS,0} + \Delta s_{TS}) - s_K)^2 \\ +(\sin(\delta_3)\cos(\alpha)r_S + \cos(\delta_3)\sin(\alpha)r_S\sin(\beta) - \sin(\alpha)\cos(\beta)(s_{TS,0} + \Delta s_{TS}) - \sin(\delta_3)r_K)^2 \\ +(\cos(\delta_3)r_S\cos(\beta) + \sin(\beta)(s_{TS,0} + \Delta s_{TS}) - \cos(\delta_3)r_K)^2\end{array}} \; . \tag{5-49}$$

Die Winkel δ_1, δ_2 und δ_3 ergeben sich wie in Gleichung (5-8). Mittels der Gleichungen (5-47), (5-48) und (5-49) wird jedoch die Gesamtlänge des jeweiligen Antriebs zwischen beiden Gelenken bestimmt, für die Positionierung der Antriebe werden die erforderlichen Verfahrwege

$$\Delta q_1 = q_1 - q_{1,0} \tag{5-50}$$
$$\Delta q_2 = q_2 - q_{2,0} \tag{5-51}$$
$$\Delta q_3 = q_3 - q_{3,0} \tag{5-52}$$

benötigt, die sich durch Abzug der Initiallänge $q_{i,0}$ des i-ten Antriebs ergeben.

Um die Gleichungen (5-50), (5-51) und (5-52) lösen zu können, werden abschließend die Winkel α und β benötigt. Diese müssen aus den kartesischen Koordinaten (siehe Gleichung (5-6)) berechnet werden. Hierbei wird ausgenutzt, dass die letzte Spalte aus Gleichung (5-41) dem Vektor

$${}_{O_{KS}}\vec{r}_{O_S} = \begin{bmatrix} \sin(\beta)((s_{TS,0} + \Delta s_{TS}(\vec{Z}_{TT})) \\ -\sin(\alpha)\cos(\beta)(s_{TS,0} + \Delta s_{TS}(\vec{Z}_{TT})) \\ \cos(\alpha)\cos(\beta)(s_{TS,0} + \Delta s_{TS}(\vec{Z}_{TT})) + s_{KS} \end{bmatrix} \tag{5-53}$$

entspricht. Der Vektor kann aus den bekannten Parametern jedoch nicht ermittelt werden. Allerdings besteht eine Kollinearität[9] zwischen den Vektoren ${}_{O_{KS}}\vec{r}_{O_S}$ und ${}_{O_{KS}}\vec{r}_{TT}$ [Mil13], wodurch auch die Betrachtung des Vektors ${}_{O_{KS}}\vec{r}_{TT}$ zur Ermittlung der Winkel genutzt werden kann.

[9] Kollinearität zwischen zwei Vektoren $\vec{a}$ und $\vec{b}$ liegt vor, wenn $\vec{a} \times \vec{b} = \vec{0}$ gilt [BSMM06].

Der Vektor

$$ {}_{O_{KS}}\vec{r}_{TT} = {}_{O_T}\vec{Z}_{TT} + {}_{O_{KS}}\vec{r}_{O_T} = \begin{bmatrix} {}_{O_T}Z_{TT,x} \\ {}_{O_T}Z_{TT,y} \\ -{}_{O_T}Z_{TT,z} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ s_{KS} \end{bmatrix} \tag{5-54} $$

ergibt sich aus der Position des Tricept-TCPs TT bezogen auf den Ursprung O_T des Tricept-Koordinatensystems sowie dem Abstand s_{KS} des Kardanschubgelenks von O_T (vgl. Bild 5-11).

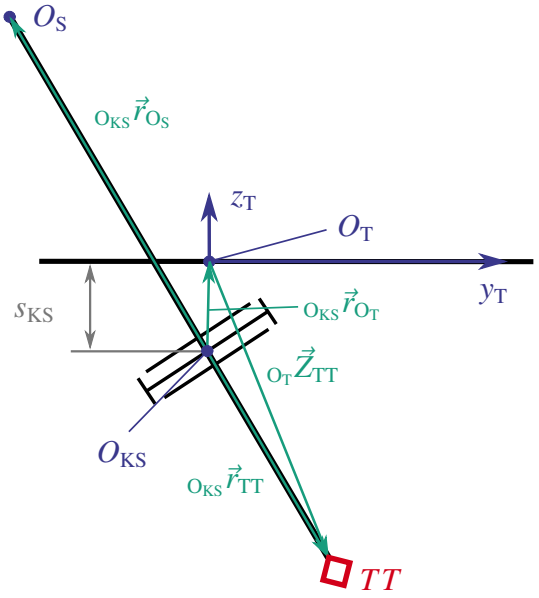


Bild 5-11: *Zweidimensionale Darstellung der Vektoren zur Berechnung von α und β*

Die für die Berechnung der Länge

$$ s_{TT} = \left|{}_{O_{KS}}\vec{r}_{TT}\right| = \sqrt{{}_{O_T}Z_{TT,x}^2 + {}_{O_T}Z_{TT,y}^2 + (-{}_{O_T}Z_{TT,z} + s_{KS})^2} \tag{5-55} $$

des Vektors erforderlichen Koordinaten ${}_{O_T}Z_{TT,x}$, ${}_{O_T}Z_{TT,y}$ und ${}_{O_T}Z_{TT,z}$ sowie der konstruktionsbedingte Abstand s_{KS} sind bekannt. Es ergeben sich somit in Anlehnung an Kugelkoordinaten die Gleichungen

$$ {}_{O_T}Z_{TT,x} = \sin(\beta) s_{TT} \tag{5-56} $$

$$ {}_{O_T}Z_{TT,y} = -\sin(\alpha)\cos(\beta) s_{TT} \tag{5-57} $$

$$ -{}_{O_T}Z_{TT,z} + s_{KS} = \cos(\alpha)\cos(\beta) s_{TT}\ . \tag{5-58} $$

Aus Gleichung (5-56) kann $\beta(\vec{Z}_{TT})$ zu

$$ \beta(\vec{Z}_{TT}) = \arcsin\left(\frac{{}_{O_T}Z_{TT,x}}{s_{TT}}\right) \tag{5-59} $$

berechnet werden. Durch Umstellen der Gleichung (5-57) nach $\cos(\beta)$, Einsetzen in Gleichung (5-58) und Nutzung des trigonometrischen Zusammenhangs $\tan(\alpha) = \frac{\sin(\alpha)}{\cos(\alpha)}$ ergibt sich

$$ \alpha(\vec{Z}_{TT}) = \operatorname{atan2}(-{}_{O_T}Z_{TT,y}, (-{}_{O_T}Z_{TT,z} + s_{KS}))\ . \tag{5-60} $$

Gegenüber dem Arcustangens werden dem Arcustangens2 (atan2 oder arctan2) zwei Argumente übergeben. Mit diesen kann der Wertebereich von 360° abgedeckt werden.

Für die zusätzliche serielle Achse M_4 des Tricept ergibt sich die Bewegung

$$\Delta q_4(\vec{Z}_{\mathrm{TT}}) = s_{\mathrm{TT}}(\vec{Z}_{\mathrm{TT}}) - q_{4,0} - \Delta s_{\mathrm{TS}} \,, \tag{5-61}$$

wobei Δs_{TS} die Änderung des Abstandes zwischen Grundplattform und bewegter Plattform bezogen auf die Initialstellung mit $s_{\mathrm{TS},0}$ und $q_{4,0}$ die Länge des Vektors zwischen dem Kardanschubgelenk und dem TCP bei Initialstellung der Achse M_4 angibt.

Bei Betrachtung der Gleichungen (5-60) und (5-59) zur Ermittlung der Winkel α und β ist zu erkennen, dass beide von ${}_{\mathrm{O_T}}Z_{\mathrm{TT,z}}$ abhängen. Dies äußert sich bei Bewegungen des Mechanismus derart, dass mit steigendem ${}_{\mathrm{O_T}}Z_{\mathrm{TT,z}}$ bzw. zunehmendem Ausfahren der Achse M_4 kleinere Winkel für konstante Werte von ${}_{\mathrm{O_T}}Z_{\mathrm{TT,x}}$ und ${}_{\mathrm{O_T}}Z_{\mathrm{TT,y}}$ benötigt werden oder anders formuliert kleinere Verfahrwege der parallelen Triceptachsen $M_1 \ldots M_3$ erforderlich sind. Aus diesem Grund sind die maximal erreichbaren Positionen des Tricept bezüglich x- und y-Richtung abhängig von der z-Richtung. In Bild 5-12 ist der Arbeitsraum des Tricepts für unterschiedliche Positionen der Achse M_4 dargestellt. Es ist die genannte Vergrößerung des Arbeitsraumes bei zunehmender z-Koordinate ${}_{\mathrm{O_T}}Z_{\mathrm{TT,z}}$ zu erkennen.

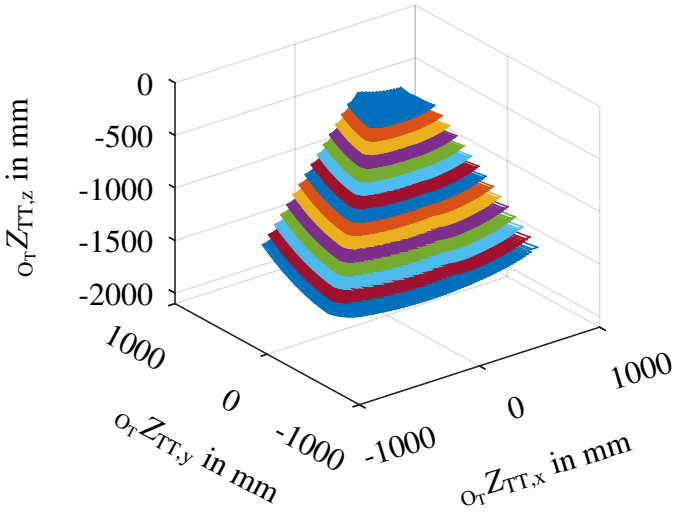


Bild 5-12: *Arbeitsraum des Tricept bei unterschiedlichen Positionen des Antriebs M_4 nach [Blu16]*

Im folgenden Abschnitt wird nun das IKP des Portals vorgestellt.

5.2.2 Inverses und direktes kinematisches Problem des Portals

Der zweite Teilmechanismus ist ein Flächenportal, welches Bewegungen in x- und y-Richtung bezogen auf ein Weltkoordinatensystem mit dem Ursprung O_{W} ermöglicht (siehe Bild 5-13).

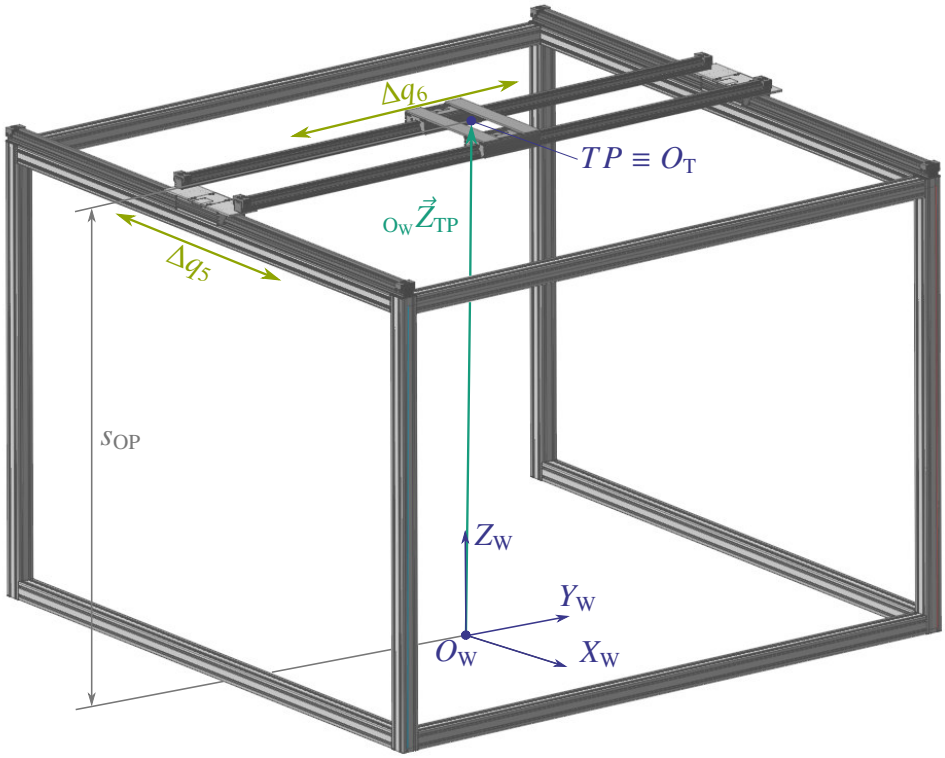


Bild 5-13: *CAD-Darstellung des Portals aus der Mehrkörpersimulationssoftware RecurDyn*

Der Position des TCP wird durch den Ortsvektor

$$ {}_{O_W}\vec{Z}_{TP} = \left[{}_{O_W}Z_{TP,x} \; {}_{O_W}Z_{TP,y} \; s_{OP} \right]^T \in \mathbb{R}^3 \tag{5-62} $$

bezogen auf das Weltkoordinatensystem beschrieben. Die Positionen ${}_{O_W}Z_{TP,x}$ und ${}_{O_W}Z_{TP,y}$ entsprechen durch geeignete Wahl von O_W in der Mitte des Arbeitsraumes des Portals und eine entsprechende Wahl des Nullpunktes der Achsen in der Mitte des jeweiligen Verfahrwegs direkt den Gelenkkoordinaten

$$ q_5({}_{O_W}\vec{Z}_{TP}) = {}_{O_W}Z_{TP,x} \tag{5-63} $$

und

$$ q_6({}_{O_W}\vec{Z}_{TP}) = {}_{O_W}Z_{TP,y} \tag{5-64} $$

der Achsen. Durch die Umkehrung dieses Zusammenhangs ergibt sich das DKP. Die z-Richtung ist bei einem Flächenportal nicht variabel und entspricht in diesem Fall dem konstruktionsbedingten festen Abstand s_{OP} zwischen dem Ursprung O_W und dem TCP TP des Portals in z-Richtung des Welt-Koordinatensystems. Die Position

$$ {}_{O_W}\vec{Z}_{TP} = [0 \; 0 \; s_{OP}]^T \tag{5-65} $$

bedeutet somit, dass beide Portalachsen in ihrer Initialstellung bei Null stehen und der TCP des Portals bezüglich seiner x- und y-Koordinate dem Ursprung des Weltkoordinatensystems entspricht. Die z-Richtung ist jedoch um den Abstand s_{OP} verschoben.

Der Parameter s_{OP} ist konstruktionsbedingt vorgegeben und hat keinen Einfluss auf die Antriebe des Portals, für die Zusammenfassung

$$\begin{bmatrix} {}_{\mathrm{O_W}}Z_{\mathrm{TM,x}} \\ {}_{\mathrm{O_W}}Z_{\mathrm{TM,y}} \\ {}_{\mathrm{O_W}}Z_{\mathrm{TM,z}} \end{bmatrix} = {}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TP}} + {}_{\mathrm{O_T}}\vec{Z}_{\mathrm{TT}} = \begin{bmatrix} {}_{\mathrm{O_W}}Z_{\mathrm{TP,x}} \\ {}_{\mathrm{O_W}}Z_{\mathrm{TP,y}} \\ {}_{\mathrm{O_W}}Z_{\mathrm{TP,z}} \end{bmatrix} + \begin{bmatrix} {}_{\mathrm{O_T}}Z_{\mathrm{TT,x}} \\ {}_{\mathrm{O_T}}Z_{\mathrm{TT,y}} \\ {}_{\mathrm{O_T}}Z_{\mathrm{TT,z}} \end{bmatrix} \tag{5-66}$$

von Tricept und Portal zum kinematisch redundanten Gesamtmechanismus ist er jedoch von hoher Relevanz. Sollpositionen für diesen werden bezogen auf das Weltkoordinatensystem mit dem Ursprung O_{W} angegeben, wobei Bewegungen in z-Richtung des Koordinatensystems ausschließlich durch den Tricept erfolgen. Somit muss mittels des konstruktionsbedingten Parameters s_{OP} eine Umrechnung von ${}_{\mathrm{O_W}}Z_{\mathrm{TM,z}}$ in ${}_{\mathrm{O_T}}Z_{\mathrm{TT,z}}$ durchgeführt werden.

Im folgenden Abschnitt wird zunächst die modellbasierte Validierung der ermittelten IKP vorgestellt, da diese Gleichungen für den modellbasierten Ansatz zwingend erforderlich sind.

5.2.3 Simulative Validierung des inversen kinematischen Problems mittels Mehrkörpersimulation

Um die Gleichungen des IKP noch vor Aufbau des realen Mechanismus zu validieren, wurde ein Mehrkörpermodell basierend auf den CAD-Daten von Portal sowie Tricept in der Software *RecurDyn* genutzt (siehe Bild 5-2). Die Ansteuerung der Mehrkörpersimulation (MKS) erfolgt im Rahmen einer Co-Simulation aus der zu *RecurDyn* gehörenden Softwarekomponente *CoLink* heraus. Hier sind neben der Vorgabe von Sollpositionen und der Gleichungen des IKP auch die Verfahrbereichsgrenzen $q_{i,\min}$ und $q_{i,\max}$ sowie die maximalen Geschwindigkeiten $v_{i,\max}$ der Antriebe $M_1 \ldots M_6$ berücksichtigt. Es wird analysiert, wie weit die vorgegebene Sollposition ${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TM}}$ (vgl. Gleichung (5-2)) von der in der MKS resultierenden Position

$${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TMres}} = \left[{}_{\mathrm{O_W}}Z_{\mathrm{TMres,x}} \; {}_{\mathrm{O_W}}Z_{\mathrm{TMres,y}} \; {}_{\mathrm{O_W}}Z_{\mathrm{TMres,z}} \right]^T \tag{5-67}$$

des TCP abweicht. Damit das IKP eindeutig lösbar ist, wird zusätzlich zu ${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TM}}$ auch ${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TP}}$ für das Portal vorgegeben. Es werden für eine Sollposition drei unterschiedliche Aufteilungen untersucht, wobei

1. die gesamte Bewegung durch den Tricept erfolgt,
2. das Portal die gesamte Bewegung (in x- und y-Richtung des Weltkoordinatensystems) übernimmt und
3. die Bewegung gleichmäßig zu jeweils 50 % (in x- und y-Richtung) auf die beiden Teilmechanismen aufgeteilt wird.

Die für die Simulation erforderlichen konstruktiven Parameter wurden anhand des CAD-Modells ermittelt, sie sind in Tabelle 5-1 aufgeführt.

Als Sollposition wird ${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TM}} = [200\ 400\ 1600]^T$ vorgegeben. Zum einen muss die Bewegung dabei in alle drei Raumrichtungen ausgeführt werden, zum anderen liegt diese Position im Arbeitsraum des Tricept, sodass dieser einzeln betrachtet werden kann. Bei der Analyse ergaben sich folgende Ergebnisse für die drei betrachteten Fälle:

Tabelle 5-1: *Konstruktive Parameter des Mechanismus*

Parameter	Wert
$q_{1,0}$, $q_{2,0}$, $q_{3,0}$	437,38 mm
$q_{4,0}$	255,38 mm
r_K	251,91 mm
r_S	146,37 mm
s_K	61,756 mm
s_{KS}	25,00 mm
s_{OP}	2716,51 mm
$s_{TS,0}$	461,21 mm

1. Zunächst wird der Fall betrachtet, bei dem die gesamte Bewegung durch den Tricept erfolgen muss. In Bild 5-14 sind die zeitliche Verläufe von ${}_{O_W}\vec{Z}_{TM}$ und ${}_{O_W}\vec{Z}_{TMres}$ dargestellt. Während die Sollposition sprungförmig geändert wird, ändert sich die tatsächliche Position des TCP aufgrund der begrenzten Geschwindigkeiten der Antriebe langsamer. Der stationäre Endwert ist nach etwas über 5 s erreicht und entspricht der Sollposition, die gemessenen Abweichungen liegen in allen drei Raumrichtungen deutlich unter 0,1 % bezogen auf den Sollwert. Wie in Bild 5-15 anhand der zeitlichen Verläufe der Antriebsbewegungen $\Delta q_1 \ldots \Delta q_6$ ausgehend von der jeweiligen Initialposition erkennbar ist, bewegen sich nach Vorgabe nur die Antriebe $M_1 \ldots M_4$ des Tricept, die Antriebe M_5 und M_6 des Portals bleiben im Stillstand ($\Delta q_5 = 0$, $\Delta q_6 = 0$). Damit können die Gleichungen des IKP des Tricept als korrekt angesehen werden.

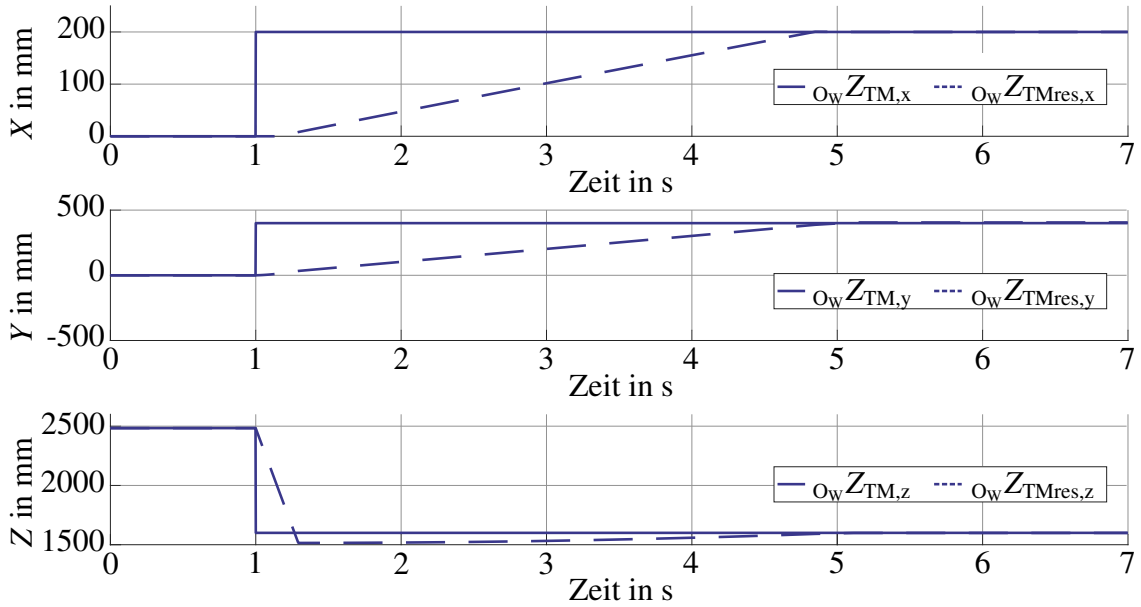


Bild 5-14: *Gegenüberstellung von* ${}_{O_W}\vec{Z}_{TM}$ *und* ${}_{O_W}\vec{Z}_{TMres}$ *bei Bewegung des Tricept*

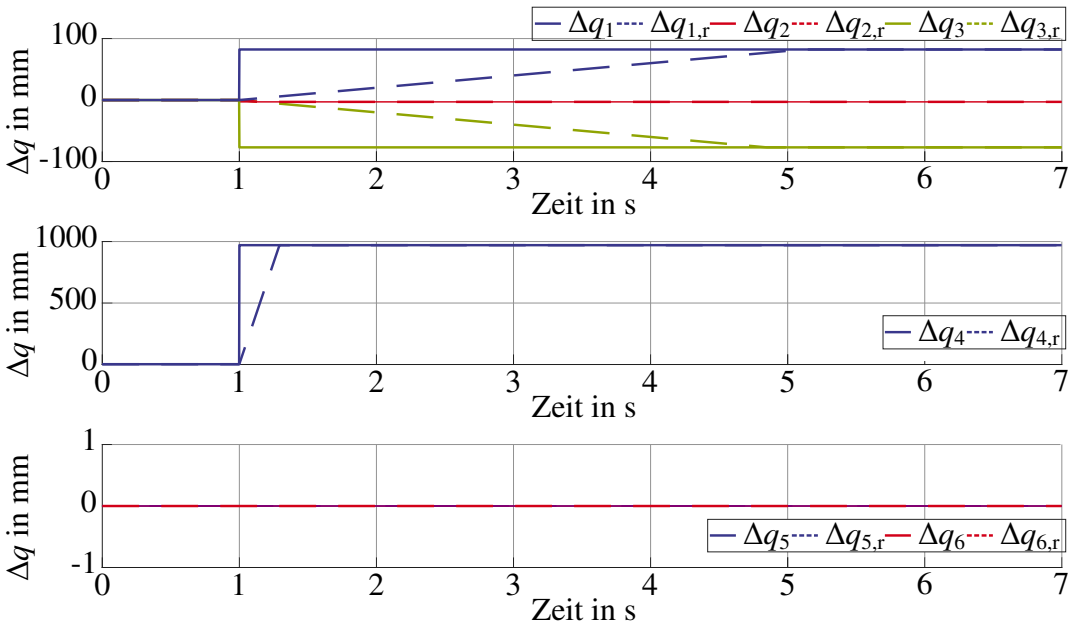


Bild 5-15: *Zeitlicher Verlauf der Antriebsbewegungen $\Delta q_1 \ldots \Delta q_6$ bei Bewegung des Tricept*

2. Im zweiten Fall wird die gesamte Bewegung in x- und y-Richtung durch das Portal ausgeführt und somit die Lösung des zugehörigen IKP validiert. Die Verläufe der Positionen sind in Bild 5-16 dargestellt. Auch hier liegt die stationäre Abweichung zwischen Soll- und Istposition unter 0,1 %. Die Antriebe $M_1 \ldots M_3$ bewegen sich nicht, M_4 wird verfahren, um die vorgegebene z-Position zu erreichen und M_5 sowie M_6 stellen die geforderte x- bzw. y-Position (vgl. Bild 5-17).

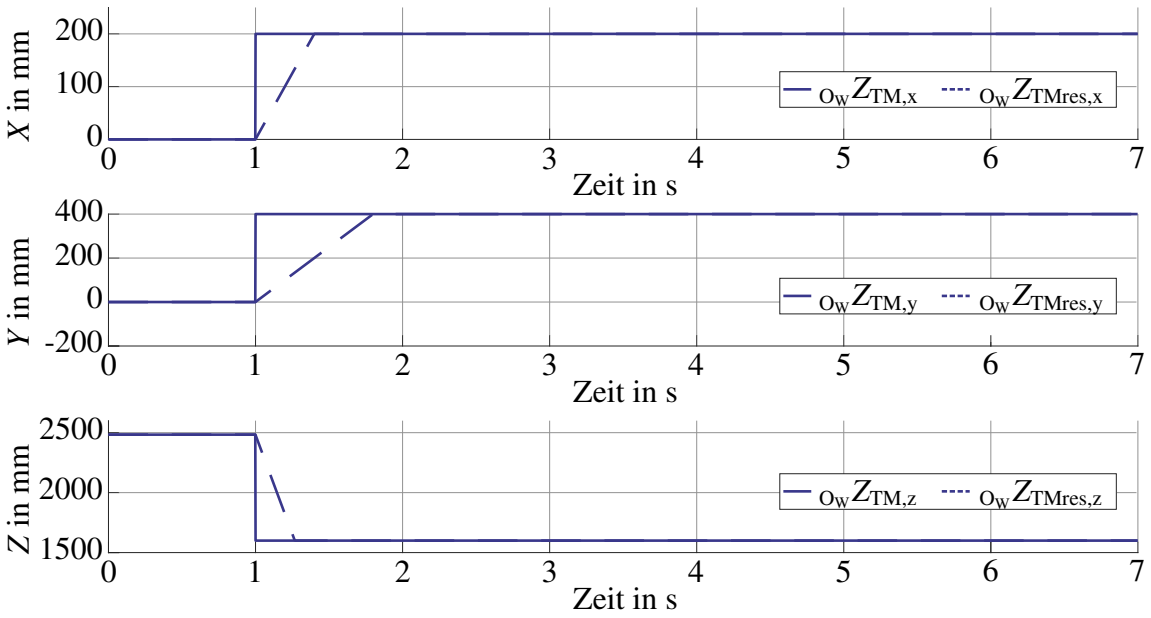


Bild 5-16: *Gegenüberstellung von ${}_{\mathrm{Ow}}\vec{Z}_{\mathrm{TM}}$ und ${}_{\mathrm{Ow}}\vec{Z}_{\mathrm{TMres}}$ bei Bewegung des Portals*

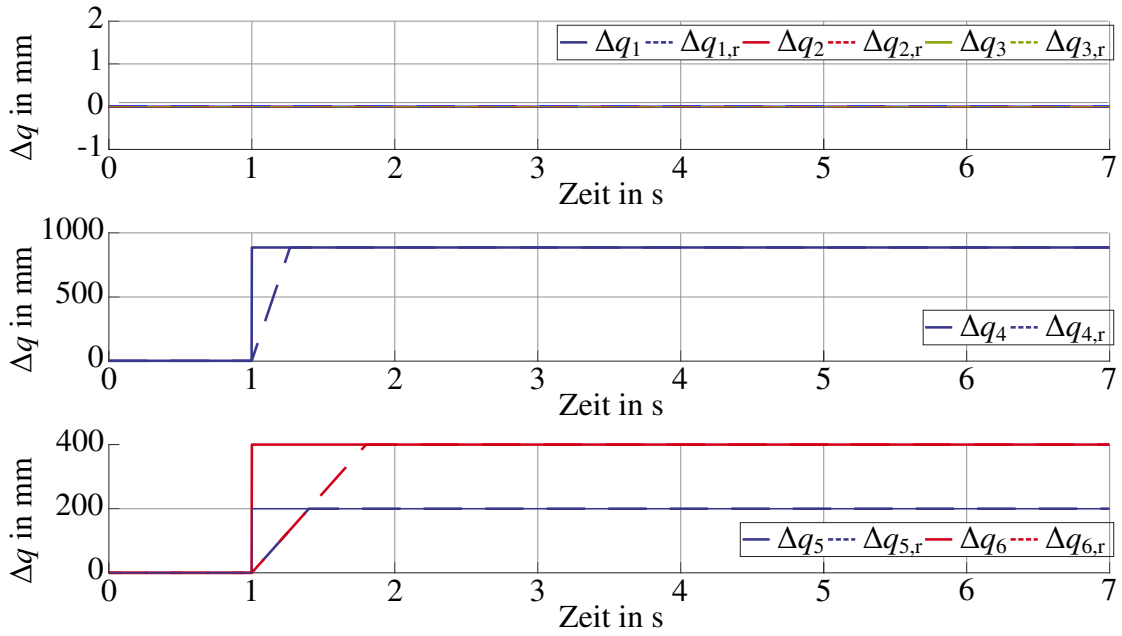


Bild 5-17: *Zeitlicher Verlauf der Antriebsbewegungen $\Delta q_1 \ldots \Delta q_6$ bei Bewegung des Portals*

3. Abschließend wird die kombinierte Bewegung der Teilmechanismen und somit das IKP des Gesamtmechanismus betrachtet. In Bild 5-18 bzw. 5-19 sind die Positionsverläufe des TCP sowie der Antriebe dargestellt. Die insgesamt durchzuführende Bewegung in x- und y-Richtung wird je zur Hälfte durch Portal und Tricept ausgeführt. Der TCP erreicht seine Sollposition mit Abweichungen unter 0,1 %, es werden alle sechs Antriebe des Mechanismus für die Bewegung genutzt. Die Positionsverläufe in x- und y-Richtung sind dabei entgegen den ersten beiden betrachteten Fällen nicht linear, da hier beide Teilmechanismen in Bewegung sind, jedoch mit unterschiedlichen Geschwindigkeiten der Antriebe.

Es lässt sich schlussfolgern, dass die Lösungen der IKP der einzelnen Teilmechanismen sowie die daraus resultierenden Gleichungen des IKP des Gesamtmechanismus korrekt sind. In Bezug auf die konstruktionsbedingten Parameter gilt es am realen Mechanismus zu überprüfen, ob Bauteiltoleranzen oder fertigungsbedingte Abweichungen vorliegen. In diesem Fall müssten die Parameter messtechnisch bestimmt und angepasst werden. In der vorliegenden Arbeit wird vom Idealfall ausgegangen, bei dem der Mechanismus exakt dem CAD-Modell entspricht.

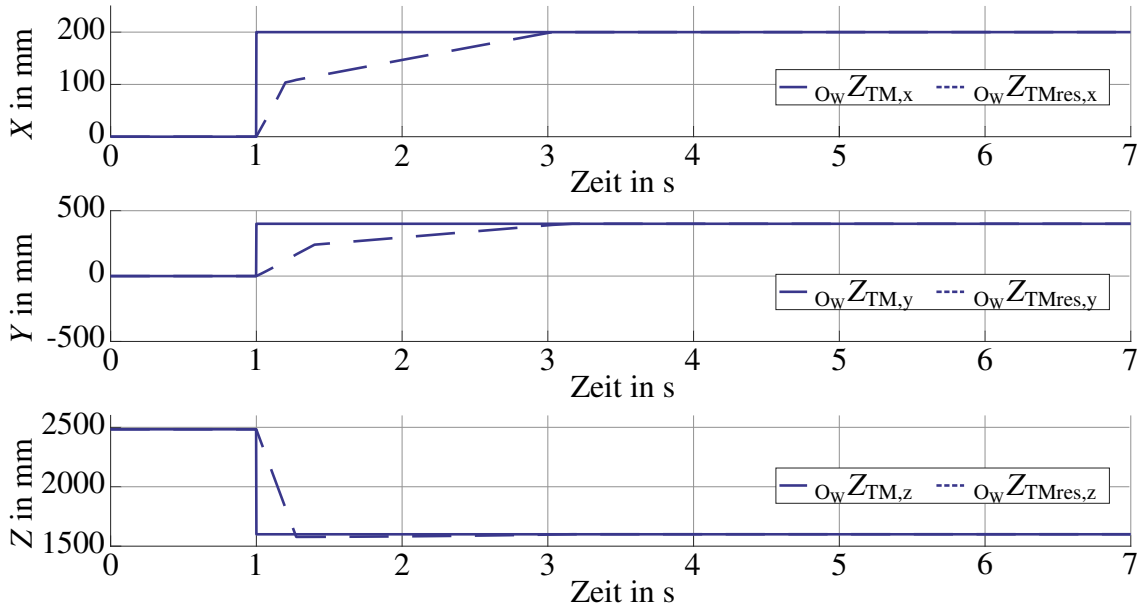


Bild 5-18: *Gegenüberstellung von* $_{OW}\vec{Z}_{TM}$ *und* $_{OW}\vec{Z}_{TMres}$ *bei kombinierter Bewegung (50/50 Aufteilung auf die beiden Teilmechanismen)*

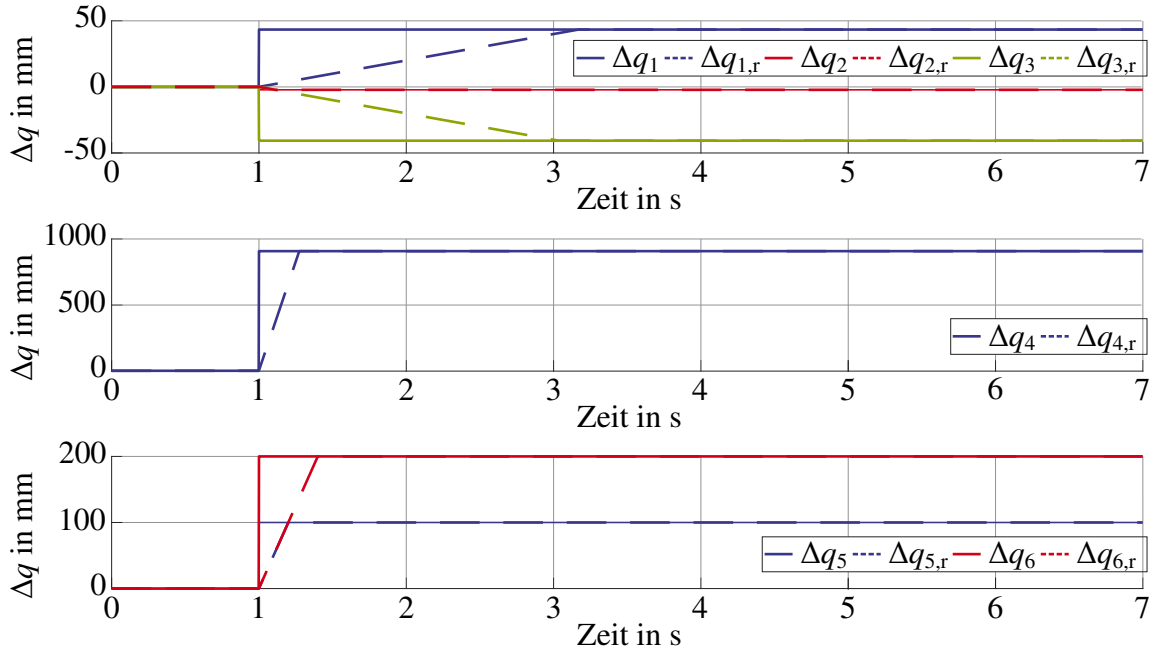


Bild 5-19: *Zeitlicher Verlauf der Antriebsbewegungen* $\Delta q_1 \ldots \Delta q_6$ *bei kombinierter Bewegung von Tricept und Portal*

5.3 Validierung des Zeitmodells für Punkt-zu-Punkt-Bewegungen

In diesem Abschnitt wird das zuvor vorgestellte Modell mit dem zeitlichen Verhalten der realen Antriebe verglichen. Dazu wurde das im weiteren Verlauf dieser Arbeit in Abschnitt 7.2 vorgestellte System mit einer SPS und Antriebsreglern der Fa. Beckhoff [Bec20] eingesetzt. Da die drei parallelen Antriebe $M_1 \ldots M_3$ des Tricept identisch sind, wird hier die exemplarische Validierung an Antrieb M_1 vorgenommen. Zudem wird der Antrieb M_4

betrachtet. Für das Portal stehen nur die Antriebe M_5 und M_6 ohne die entsprechenden Linearachsen zur Verfügung. Hier wird angenommen, dass sich die Antriebe durch ihre Dimensionierung ohne und mit Belastung durch das Portal vergleichbar verhalten, zudem wird hier M_6 stellvertretend für beide Antriebe betrachtet.

Der Vergleich zwischen dem vorgestellten vereinfachten Zeitmodell und den Fahrzeiten der realen Antrieben wird anhand unterschiedlicher Längen der zu verfahrender Strecken sowie positiver und negativer Verfahrrichtung durchgeführt. Die Daten der einzelnen Antriebe sind in Tabelle 5-2 zusammengefasst. Alle Angaben beziehen sich auf die Parametrierung im Rahmen des vorgestellten Systems in kombination mit den eingesetzten Achsen und geben nicht zwangsläufig die tatsächlich möglichen Maximalwerte der einzelnen Antriebe an.

Tabelle 5-2: *Parameter der eingesetzten Antriebe und Achsen*

Antrieb	$v_{i,\max}$	$a_{i,\max}$	$a_{i,\min}$	$\Delta q_{i,\max}$	$\Delta q_{i,\mathrm{b}}$
M_1	$30\,\mathrm{mm\,s^{-1}}$	$180\,\mathrm{mm\,s^{-2}}$	$-180\,\mathrm{mm\,s^{-2}}$	100 mm	5,00 mm
M_2	$30\,\mathrm{mm\,s^{-1}}$	$180\,\mathrm{mm\,s^{-2}}$	$-180\,\mathrm{mm\,s^{-2}}$	100 mm	5,00 mm
M_3	$30\,\mathrm{mm\,s^{-1}}$	$180\,\mathrm{mm\,s^{-2}}$	$-180\,\mathrm{mm\,s^{-2}}$	100 mm	5,00 mm
M_4	$500\,\mathrm{mm\,s^{-1}}$	$3300\,\mathrm{mm\,s^{-2}}$	$-3300\,\mathrm{mm\,s^{-2}}$	2000 mm	75,76 mm
M_5	$100\,\mathrm{mm\,s^{-1}}$	$500\,\mathrm{mm\,s^{-2}}$	$-500\,\mathrm{mm\,s^{-2}}$	2000 mm	20,00 mm
M_6	$100\,\mathrm{mm\,s^{-1}}$	$500\,\mathrm{mm\,s^{-2}}$	$-500\,\mathrm{mm\,s^{-2}}$	2000 mm	20,00 mm

Zunächst erfolgt die Validierung des Modells anhand der jeweils einzeln und unabhängig voneinander betrachteten Antriebe. Dabei wird auf die explizite Kennzeichnung der Abhängigkeit von der aktuellen Position mit Index k verzichtet. Vielmehr wird angenommen, dass immer eine Bewegung der Länge Δq_i zwischen der Initialstellung $q_{i,0}$ des jeweiligen Antriebs als Ausgangsposition und einer Zielposition q_i stattfindet.

In Bild 5-20 ist der relative Verlauf der Antriebsposition des Antriebs M_1 bei einer Verfahrstrecke von $\Delta q_1 = 5\,\mathrm{mm}$ dargestellt. Die Bewegung erfolgt in positive Richtung, was bedeutet, dass der Hubspindelantrieb ausfährt und somit die Masse des Tricept entgegen der Erdbeschleunigung bewegen muss. Der eigentliche Verlauf der Position ist bei der Nutzung des Modells nicht von Bedeutung. Ziel ist es die benötigte Zeit der Bewegung (vom Stillstand über die Bewegungen bis zum erneuten Stillstand) so genau wie möglich zu berechnen. Bezogen auf den dargestellten Fall benötigt der reale Antrieb M_1 die Zeit $\Delta t_{1,\mathrm{r}}(\Delta q_1) = 362\,\mathrm{ms}$, das Modell liefert als Ergebnis $\Delta t_1(\Delta q_1) = 333\,\mathrm{ms}$. Dies führt zu einem Fehler von

$$\Delta t_{1,\mathrm{e}}(\Delta q_1) = \Delta t_{1,\mathrm{r}}(\Delta q_1) - \Delta t_1(\Delta q_1) = 29\,\mathrm{ms}\,. \tag{5-68}$$

Bezogen auf die Verfahrzeit des realen Antriebs liegt der prozentuale Fehler bei

$$e_1(\Delta q_1) = \frac{100 \cdot \Delta t_{1,\mathrm{e}}(\Delta q_1)}{\Delta t_{1,\mathrm{r}}(\Delta q_1)} = 8{,}01\,\%\,. \tag{5-69}$$

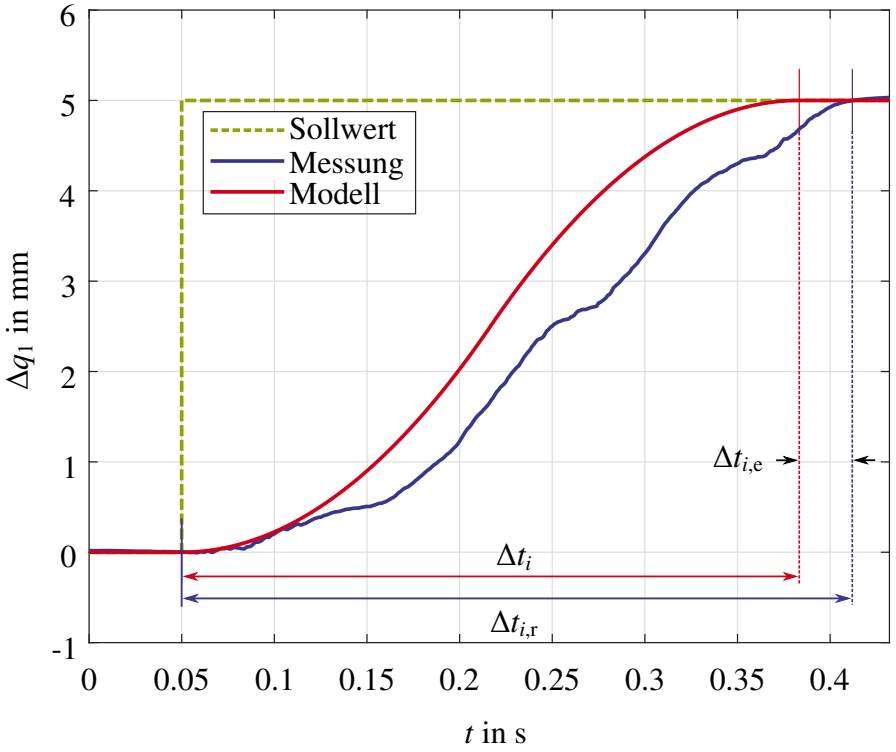


Bild 5-20: *Positionsverlauf des Antriebs M_1 bei einer Strecke $\Delta q_1 = 5\,\mathrm{mm}$*

Bild 5-21 zeigt das Verhalten bei einer Strecke von $\Delta q_1 = 30\,\mathrm{mm}$. Die ermittelten Zeiten ergeben sich zu $\Delta t_{1,\mathrm{r}}(\Delta q_1) = 1{,}161\,\mathrm{s}$ und $\Delta t_1(\Delta q_1) = 1{,}167\,\mathrm{s}$, der prozentuale Fehler beträgt $e_1(\Delta q_1) = -0{,}52\,\%$. Die Abweichung des Modells vom realen Antrieb bei kleinen Verfahrstrecken liegt vornehmlich am Antrieb und der dazu gehörenden Regelung. Die verwendeten Gleichstrommotoren benötigen eine Spannung von mehreren Volt, um das für den Spindelantrieb erforderliche Losbrechmoment aufzubringen. Sobald sich die Spindel bewegt, genügt eine kleinere Spannung zum Aufrechterhalten der Bewegung. Die geforderte Verfahrstrecke $\Delta q_1 = 5\,\mathrm{mm}$ ist so lang wie die nach Gleichung (4-17) berechnete Grenzdistanz $\Delta q_{1,\mathrm{b}}$ (vgl. Tabelle 5-2), jedoch schafft der Regler es nicht, dass tatsächlich $v_{1,\mathrm{max}}$ erreicht wird. Vielmehr ist der reale Antrieb mit näherungsweise $v_{1,\mathrm{r}} = 26\,\mathrm{mm\,s^{-1}}$ langsamer als das Modell. Die Reglerauslegung für die realen Antriebe nach etablierten Verfahren wie dem Betragsoptimum oder dem symmetrischen Optimum (siehe [Föl13]) lieferte am realen Antrieb aufgrund des hier beschriebenen Verhaltens mit einem Losbrechmoment größer dem für die Bewegung erforderlichen Moment keine zufriedenstellenden Ergebnisse. Der genutzte Antriebsregler bietet für solche Antriebe aber die Option, zwei verschiedene Proportionalfaktoren festzulegen [Bec23]. Die Geschwindigkeit, bei der eine Umschaltung erfolgt, kann ebenfalls festgelegt werden. Die Auslegung der Reglerparameter erfolgte empirisch anhand der Analyse verschiedener Positionsvorgaben mit Variation der Faktoren und der Umschaltgeschwindigkeit. Hierbei handelt es sich um eine Vorgehensweise, die etwa von Fa. Beckhoff in Handbüchern zur Antriebshardware beschrieben ist [Bec18a]. Die hier gezeigten Positionsverläufe basieren auf den empirisch ermittelten Parametern.
Bei größeren Strecken wird der Einfluss des Effekts deutlich geringer, der Antrieb erreicht seine maximale Geschwindigkeit und das Modell stimmt ausreichend gut mit dem realen

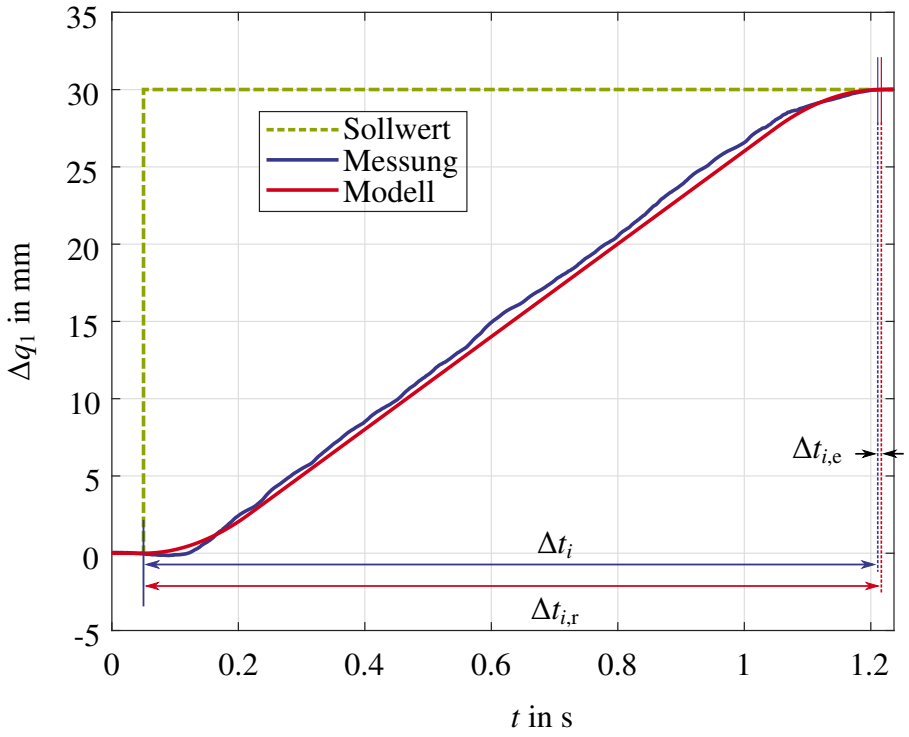


Bild 5-21: *Positionsverlauf des Antriebs M_1 bei einer Strecke $\Delta q_1 = 30\,\mathrm{mm}$*

System überein. Neben den positiven Verfahrwegen wurden auch die Fälle $\Delta q_1 = -5\,\mathrm{mm}$ und $\Delta q_1 = -30\,\mathrm{mm}$ betrachtet, bei denen der Hubspindelantrieb die Last in Richtung der Erdanziehungskraft bewegt. Die Ergebnisse sind in Tabelle 5-3 zusammengefasst. Des Weiteren wurde das Verhalten der seriellen Achse M_4 sowie stellvertretend für das Portal der Achse M_6 analysiert (vgl. Tabelle 5-3). Auch hier sind die Verfahrstrecken jeweils so gewählt, dass die Antriebe einmal $v_{i,\max}$ erreichen und einmal nicht. Bei der seriellen Achse wird aufgrund des Einflusses der Schwerkraft sowohl die positive als auch die negative Verfahrrichtung betrachtet. Beim Portal hingegen wird nur die positive Verfahrrichtung angegeben, durch die Ausrichtung in der x-y-Ebene wirkt die Erdanziehungskraft orthogonal zur Achse und damit zu den beiden möglichen Bewegungsrichtungen, ihr Einfluss ist in beide Verfahrrichtungen näherungsweise gleich.

Wie an den in Tabelle 5-3 dargestellten Ergebnissen zu erkennen ist, sind die Abweichungen zwischen realem Antrieb und Modell aus den bereits beschriebenen Gründen bei den Antrieben $M_1 \ldots M_3$ des parallelen Mechanismus am größten. Generell ist aber festzuhalten, dass das Modell eine für den geplanten Anwendungsfall ausreichende Genauigkeit aufweist, zumal der zu erwartende geringe Rechenaufwand mit berücksichtigt werden muss.

Abschließend wird die Gültigkeit des vorgestellten Modells bei Achsinterpolation analysiert. Dabei werden fünf der sechs Achsen derart beeinflusst, dass sie ihre jeweilige Zielposition $q_i(\vec{Z}_{\mathrm{TM}})$ in Abhängigkeit von der Sollposition $\vec{Z}_{\mathrm{TM}}$ des TCP nicht in der kürzest möglichen Zeit sondern zeitgleich mit der für das Anfahren dieser Sollposition langsamsten Achse (Verfahrzeit $\Delta t_{\max}(\vec{Z}_{\mathrm{TM}})$) erreichen. Dieses Prinzip ist in Bild 5-22 dargestellt, es liegt eine synchrone PTP-Bewegung vor (siehe Kapitel 2.2). Im Beispiel stellt M_2 die langsamste Achse dar, für alle anderen Achsen wird die jeweils innerhalb der Verfahrstrecke

Tabelle 5-3: Ergebnisse der Validierung des Zeitmodells für Punkt-zu-Punkt-Bewegungen

Bezeichnung	Δq_i	$\Delta t_{i,\mathrm{r}}$	Δt_i	$\Delta t_{i,\mathrm{e}}$	e_i
M_1	5 mm	362 ms	333 ms	29 ms	8,01 %
	30 mm	1161 ms	1167 ms	−6 ms	−0,52 %
	−5 mm	355 ms	333 ms	22 ms	6,20 %
	−30 mm	1153 ms	1167 ms	−14 ms	−1,21 %
M_4	50 mm	188 ms	196 ms	−8,2 ms	−4,36 %
	300 mm	697 ms	701,5 ms	−4,5 ms	−0,65 %
	−50 mm	190 ms	196 ms	−6 ms	−3,16 %
	−300 mm	697 ms	701,5 ms	−4,5 ms	−0,65 %
M_6	1 mm	40 ms	39,4 ms	0,6 ms	1,50 %
	50 mm	647 ms	650 ms	−3 ms	−0,46 %

maximal erreichte Geschwindigkeit $v_{i,\mathrm{g}}(\vec{Z}_{\mathrm{TM}}) \leq v_{i,\max}$ soweit reduziert, dass alle Antriebe nach der Zeit $\Delta t_{\max}(\vec{Z}_{\mathrm{TM}})$ an ihrer Sollposition im Stillstand sind. Bei der Umsetzung am realen System erfolgt die Interpolation der Achsen über virtuelle Getriebe. Dabei werden alle sechs Achsen an eine virtuelle Leitachse (vgl. Kapitel 2, Punkt-zu-Punkt-Bewegungen) gekoppelt, diese übernimmt die Parameter maximale Geschwindigkeit, Beschleunigung, Verzögerung und Verfahrstrecke der langsamsten Achse mit der Verfahrzeit $\Delta t_{\max}$. Die Achsen werden mittels eines jeweils individuell und für jede Sollposition $\vec{Z}_{\mathrm{TM}}$ neu berechneten Übersetzungsverhältnisses

$$r_i(\vec{Z}_{\mathrm{TM}}) = \frac{\Delta q_i(\vec{Z}_{\mathrm{TM}})}{\Delta q_{\max}(\vec{Z}_{\mathrm{TM}})} \tag{5-70}$$

an diese Leitachse gekoppelt. Es gilt zu beachten, dass durch das Getriebe zum einen die maximal erreichte Geschwindigkeit

$$v_{i,\mathrm{g}}(\vec{Z}_{\mathrm{TM}}) = r_i(\vec{Z}_{\mathrm{TM}}) \cdot v_{\max}(\vec{Z}_{\mathrm{TM}}) \tag{5-71}$$

des jeweiligen Antriebs M_i bezogen auf die Leitachse festgelegt wird, sich aber auch die Werte für Beschleunigung und Verzögerung in Abhängigkeit des Übersetzungsverhältnisses ändern. Dies wird nach [Web09] auch als *vollsynchrone PTP-Bewegung* bezeichnet.

In Bild 5-23 ist ein beispielhafter Verlauf der sechs Antriebspositionen $q_i(\vec{Z}_{\mathrm{TM,k}})$ bei drei Sollpositionen $\vec{Z}_{\mathrm{TM,k}}$ mit $i = 1 \ldots 6$ und $k = 1 \ldots 3$ ausgehend von einer nicht näher definierten Ausgangsposition $\vec{Z}_{\mathrm{TM,0}}$ dargestellt. Die grün gestrichelten senkrechten Linien geben den Beginn einer Bewegung bzw. das Erreichen der Sollposition durch das Modell an. Die Solldistanzen der einzelnen Antriebe sind in Tabelle 5-4, die daraus resultierenden Übersetzungsverhältnisse $r_i(\vec{Z}_{\mathrm{TM,k}})$ und maximal zulässigen Geschwindigkeiten $v_{i,\mathrm{g}}(\vec{Z}_{\mathrm{TM,k}})$ nach Gleichung (5-71) in Tabelle 5-5 zusammengefasst. Wie hier anhand der Geschwindigkeit zu erkennen ist, weist bei den ersten beiden Sollpositionen $\vec{Z}_{\mathrm{TM,1}}$ und $\vec{Z}_{\mathrm{TM,2}}$ der Antrieb M_3 die längste Verfahrzeit auf, somit erreicht auch nur dieser Antrieb in der Bewegung seine tatsächliche Maximalgeschwindigkeit von $30\,\mathrm{mm\,s^{-1}}$ (vgl. Tabelle 5-2). Die

entsprechenden Übersetzungsverhältnisse des virtuellen Getriebes betragen $r_3(\vec{Z}_{\mathrm{TM},1}) = 1$ und $r_3(\vec{Z}_{\mathrm{TM},2}) = 1$. Bei der dritten Bewegung benötigt der Antrieb M_1 die längste Zeit, somit gilt $r_1(\vec{Z}_{\mathrm{TM},3}) = 1$ und $v_{\max}(\vec{Z}_{\mathrm{TM},3}) = v_{1,\max}$.

Tabelle 5-4: *Vorgaben bei der Modellvalidierung mit Achsinterpolation*

i	$\Delta q_i(\vec{Z}_{\mathrm{TM},1})$	$\Delta q_i(\vec{Z}_{\mathrm{TM},2})$	$\Delta q_i(\vec{Z}_{\mathrm{TM},3})$
1	54,82 mm	−14,95 mm	−25,40 mm
2	5,41 mm	−6,33 mm	5,56 mm
3	−59,90 mm	20,88 mm	19,80 mm
4	−282,12 mm	115,70 mm	128,89 mm
5	−33,26 mm	12,76 mm	7,28 mm
6	47,79 mm	−11,40 mm	−22,69 mm

Tabelle 5-5: *Übersetzungsverhältnisse und Geschwindigkeiten bei der Modellvalidierung mit Achsinterpolation*

i	$r_i(\vec{Z}_{\mathrm{TM},1})$	$r_i(\vec{Z}_{\mathrm{TM},2})$	$r_i(\vec{Z}_{\mathrm{TM},3})$	$v_{i,\mathrm{g}}(\vec{Z}_{\mathrm{TM},1})$	$v_{i,\mathrm{g}}(\vec{Z}_{\mathrm{TM},2})$	$v_{i,\mathrm{g}}(\vec{Z}_{\mathrm{TM},3})$
1	−0,92	−0,72	1,00	27,46 mm s^{-1}	−21,48 mm s^{-1}	−30,00 mm s^{-1}
2	−0,09	−0,30	−0,22	2,71 mm s^{-1}	−9,10 mm s^{-1}	6,57 mm s^{-1}
3	1,00	1,00	−0,78	−30,00 mm s^{-1}	30,00 mm s^{-1}	23,39 mm s^{-1}
4	4,71	5,54	−5,07	−141,3 mm s^{-1}	166,24 mm s^{-1}	152,23 mm s^{-1}
5	0,56	0,61	−0,29	−16,66 mm s^{-1}	18,33 mm s^{-1}	8,60 mm s^{-1}
6	−0,80	−0,55	0,89	23,94 mm s^{-1}	−16,38 mm s^{-1}	−26,80 mm s^{-1}

Tabelle 5-6 beinhaltet die resultierenden Abweichungen zwischen realem Mechanismus und Modell, sowie die prozentualen Fehler nach Gleichung (5-69). Es ist zu beachten, dass bei dem realen Antrieb der Zeitpunkt betrachtet wird, an dem sich dieser das erste mal im Bereich von ±0,05 mm um die Sollposition befindet. In der SPS ist dies ein fest definierter Parameter innerhalb der eingesetzten Programmbausteine zur Ansteuerung der Antriebe. Im Bild 5-23 ist dies durch einen kurzen senkrechten Strich in der Farbe der jeweiligen Linie gekennzeichnet. Der prozentuale Fehler wird bezogen auf die Verfahrzeit des jeweiligen realen Antriebs für die aktuelle Bewegung berechnet. Die erste Bewegung beginnt beispielsweise bei 1 s, der reale Antrieb M_1 erreicht den Sollbereich zum Zeitpunkt 3,26 s, somit bezieht sich der prozentuale Fehler auf die Verfahrdauer 2,26 s. Zusammen mit einer Verfahrdauer von 2,22 s im Modell ergibt sich der absolute Fehler zu $\Delta t_{1,\mathrm{e}}(\vec{Z}_{\mathrm{TM},1}) =$ 40 ms und der prozentuale Fehler zu $e_1(\vec{Z}_{\mathrm{TM},1}) = 1{,}77\,\%$. Die Abhängigkeit vom Ortsvektor $\vec{Z}_{\mathrm{TM},k}$ ergibt sich daraus, dass wiederum q_i von diesem abhängt, siehe Gleichungen (5-68) und (5-69).

Tabelle 5-6: *Absolute und prozentuale Fehler der Verfahrzeiten bei der Modellvalidierung mit Achsinterpolation*

i	$\Delta t_{i,\mathrm{e}}(\vec{Z}_{\mathrm{TM},1})$	$\Delta t_{i,\mathrm{e}}(\vec{Z}_{\mathrm{TM},2})$	$\Delta t_{i,\mathrm{e}}(\vec{Z}_{\mathrm{TM},3})$	$e_i(\vec{Z}_{\mathrm{TM},1})$	$e_i(\vec{Z}_{\mathrm{TM},2})$	$e_i(\vec{Z}_{\mathrm{TM},3})$
1	40 ms	32 ms	40 ms	1,77 %	3,51 %	3,74 %
2	−21 ms	3 ms	7 ms	−0,95 %	0,34 %	0,68 %
3	1 ms	23 ms	41 ms	0,05 %	2,55 %	3,83 %
4	2 ms	−10 ms	−1 ms	0,09 %	−1,15 %	−0,1 %
5	3 ms	2 ms	5 ms	0,13 %	0,23 %	0,48 %
6	6 ms	−7 ms	6 ms	0,27 %	−0,8 %	0,58 %

Die ermittelten relativen Fehler liegen bei maximal 3,83 % für Antrieb M_3 (Sollposition $\vec{Z}_{\mathrm{TM},3}$). Damit ist das Modell auch im achsinterpolierten Fall als ausreichend genau anzusehen, es kann im Rahmen der modellbasierten Optimierung eingesetzt werden.

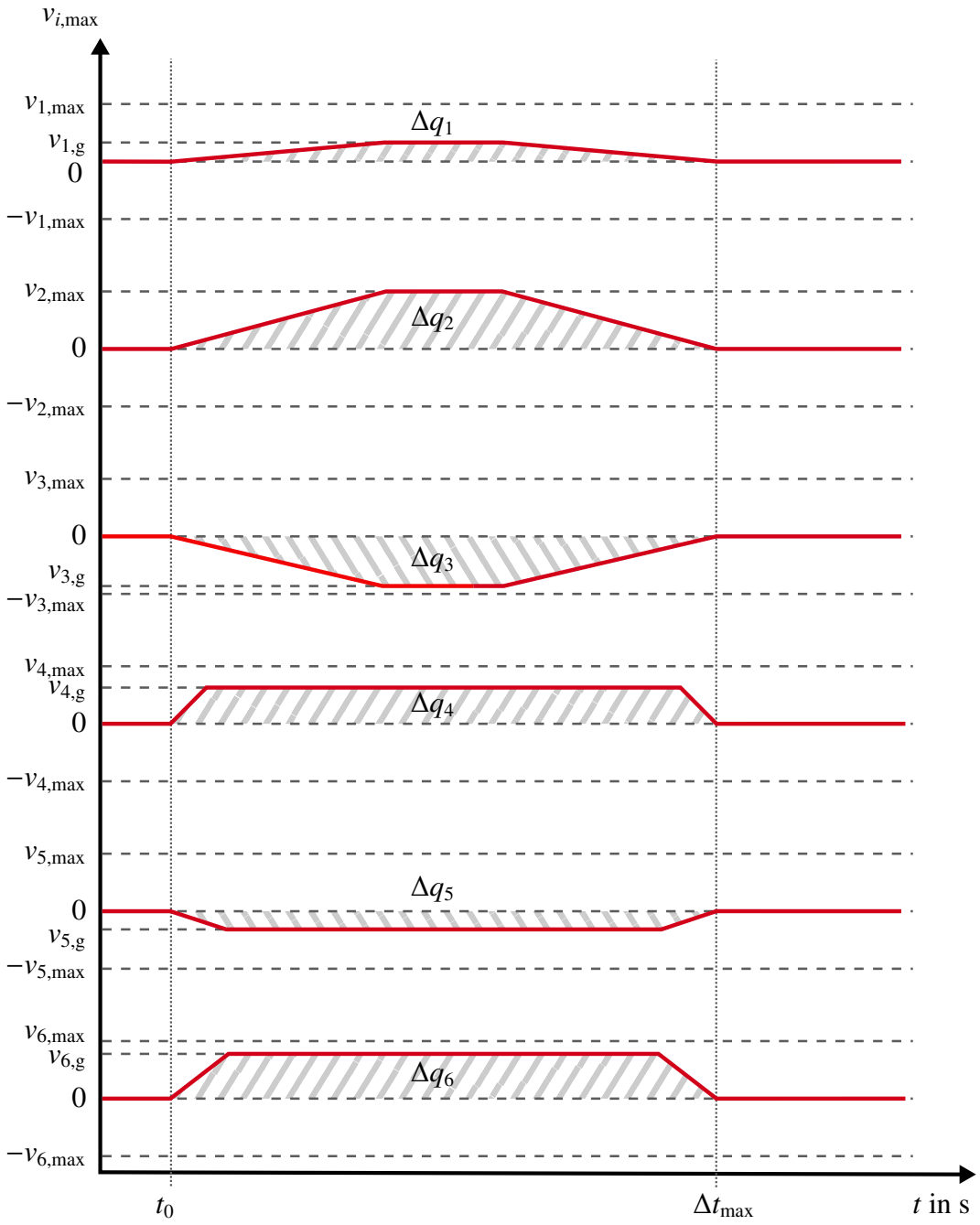


Bild 5-22: *Prinzip des vollsynchronen PTP-Betriebs*

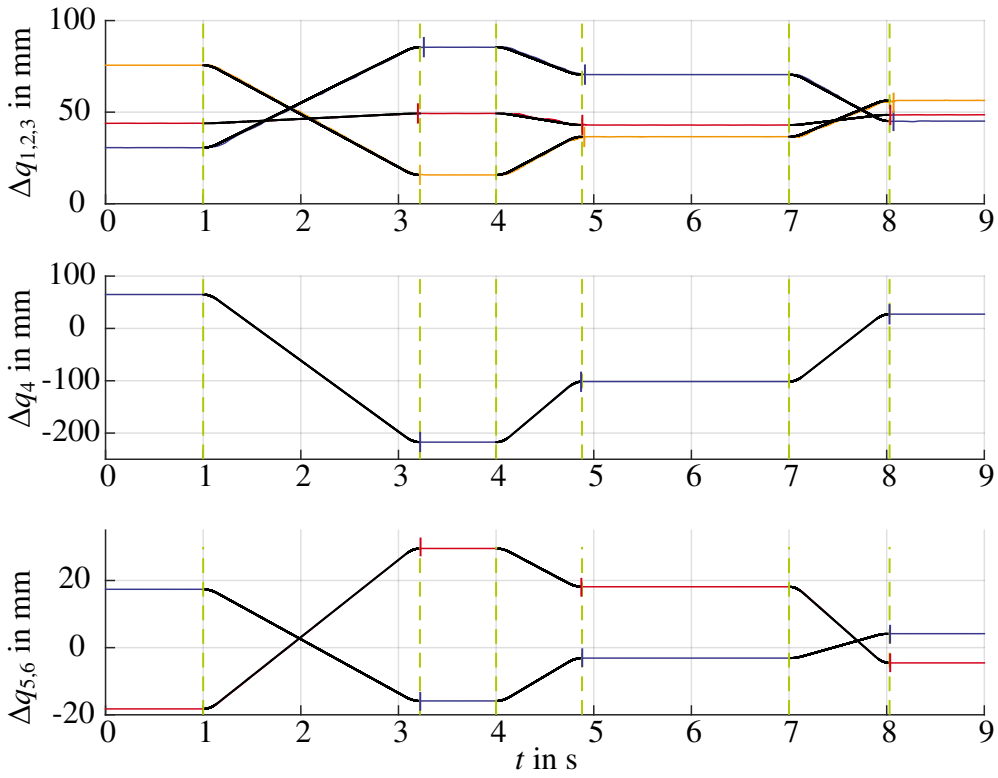


Bild 5-23: *Beispielhafter Positionsverlauf der sechs Antriebe bei vollsynchronem PTP-Betrieb (oben: M_1 (blau), M_2 (rot), M_3 (orange); Mitte: M_4; unten: M_5 (blau), M_6 (rot))*

5.4 Mögliche Anwendungsszenarien für den IEM-Mechanismus

Der IEM-Mechanismus wurde ursprünglich als Inspektionssystem für beengte Arbeitsräume entwickelt, die geprägt sind durch eine Vielzahl von innenliegenden konstruktions- und prozessbedingten Hindernissen [RHWT15]. In dieser Anwendung kann der Mechanismus die Sensorik aufgrund der Hindernisse nicht immer in die erforderliche Orientierung zur zu inspizierenden Stelle bringen. Dies wird vielmehr durch einen aktiven Werkzeugkopf, durch seine Position am TCP des Mechanismus auch als *Endeffektor* oder teilweise auch *End of Arm Tooling* bezeichnet, erreicht. Dieser kann zusätzlich zum eigentlichen Tricept die Sensoren beziehungsweise die gesamte Inspektionseinheit ausrichten.

In der vorliegenden Arbeit wird die Orientierung des TCP bzw. des daran befestigten Endeffektors hinsichtlich der Planung zeitoptimierter Trajektorien vernachlässigt. Für den Fall, dass der durchzuführende Prozesse eine definierte Orientierung erfordert, wird hier ebenfalls eine entsprechende zusätzliche Ausrichtungseinheit benötigt. Diese muss die Differenz zwischen den Winkeln α und β des Tricept und den für den Prozess erforderlichen Winkeln ausgleichen. Mit einem solchen *aktiven* Werkzeug kann der Tricept bzw. der gesamte IEM-Mechanismus in unterschiedlichen Anwendungen eingesetzt werden. Beispielhaft seien hier neben der Inspektion etwa das Setzen von Klebe- oder Schweißpunkten auf Bauteilen genannt, die eine große Variantenvielfalt hinsichtlich der Größe und der Form, aber auch bezogen auf die Positionen der zu bearbeitenden Stellen aufweisen.

Aktive Werkzeugköpfe werden in unterschiedlichsten Anwendungen im industriellen Umfeld eingesetzt und sind je nach Anwendungsfall zum Teil auch als fertige Komponente erhältlich. Ein Beispiel hierfür ist das Greif-Schwenk-Modul *EGS* der Fa. *Schunk*, welches eine Kombination aus Greifer und zusätzlichem Rotationsantrieb darstellt [SCH19b] (siehe Bild 5-24). Dies kann etwa für schnelle Greif- und Rotationsaufgaben genutzt werden, bei denen die Geschwindigkeit der klassischen Robotergelenke nicht ausreicht, um die Prozessanforderungen zu erfüllen und stattdessen eine zusätzliche Achse genutzt wird.

Für den vorgestellten IEM-Mechanismus lassen sich mit entsprechenden Werkzeugköpfen verschiedene Anwendungsszenarien definieren. Da der Fokus der vorliegenden Arbeit jedoch auf der vom konkreten Anwendungsfall losgelösten Planung zeitoptimierter Trajektorien liegt, wird auf weitere mögliche Anwendungen hier nicht näher eingegangen. Im folgenden Kapitel werden vielmehr die Grundidee dieser modellprädiktiven Planung, das dabei resultierende Optimierungsproblem sowie die eingesetzten Modelle vorgestellt.

Bild 5-24: *Greif-Schwenk-Modul EGS der Fa. Schunk [SCH19a]*

6 Simulative Umsetzung in *MATLAB*

Eines der grundlegenden Ziele der vorliegenden Arbeit ist die echtzeitfähige Implementierbarkeit des entwickelten Ansatzes auf industrieller Hardware. Entsprechend dem grundlegenden Gedanken der modellbasierten Entwicklung werden vor der Umsetzung auf reale Hardware jedoch zunächst weiterführende Analysen anhand eines Modells durchgeführt. Hierzu wird im Folgenden die Implementierung in *MATLAB* vorgestellt. Diese dient zum einen dem Nachweis der prinzipiellen Funktionsfähigkeit des Ansatzes, zudem werden sowohl mit dem Referenzverfahren aus Abschnitt 4.5 als auch mit dem modellprädiktiven Ansatz Vergleichsdaten erzeugt. Mit diesen werden die im Rahmen der in Kapitel 7 vorgestellten Implementierung auf der Industriehardware erzielten Ergebnisse verglichen.

Für die Ausführung des Referenzverfahrens und des modellprädiktiven Ansatzes in *MATLAB* werden zunächst die in Kapitel 4 erläuterte Gütefunktion und die Zeitmodelle sowie die in Kapitel 5 vorgestellten Gleichungen der IKP der Teilsysteme Portal und Tricept implementiert. Im Anschluss wird ein geeigneter Optimierer ausgewählt. Dieser muss für nichtlineare Optimierungsprobleme mit mehreren Variablen geeignet sein. Mit der Optimization Toolbox [Mat20] bietet *MATLAB* hier bereits verschiedene Algorithmen an. Da diese jedoch nicht für die Implementierung auf industrieller Hardware geeignet sind, wird im Anschluss eine Analyse weiterer Algorithmen mit dem Fokus auf frei verfügbare und Software-/*MATLAB*-unabhängige (jedoch in *MATLAB* nutzbare) Bibliotheken durchgeführt. Die dabei erzielten Ergebnisse bezogen auf den modellprädiktiven Ansatz werden mit denen der Optimization Toolbox und zudem mit dem Referenzverfahren verglichen. Um hierbei eine Vergleichbarkeit über alle Analysen hinweg zu gewährleisten und zudem die Ergebnisse nicht durch eine gezielte Wahl von Sollpositionen zu beeinflussen, müssen zunächst geeignete Sollpositionsfolgen ermittelt werden. Dies ist im folgenden Abschnitt beschrieben.

6.1 Wahl der Sollpositionsfolgen

Um durch die Wahl der Sollpositionen weder den modellprädiktiven Optimierungsansatz noch den Referenzansatz bewusst zu beeinflussen und eventuell zu bevorzugen, wird bei der Erzeugung der Sollpositionsfolgen auf einen Zufallsgenerator[10] zurückgegriffen. Durch diesen werden in jedem Schritt drei gleichverteilte Zufallszahlen für ${}_{\mathrm{O_W}}Z_{\mathrm{TM,x}}$, ${}_{\mathrm{O_W}}Z_{\mathrm{TM,y}}$ und ${}_{\mathrm{O_W}}Z_{\mathrm{TM,z}}$ erzeugt. Die Grenzen

$$\begin{aligned} -500\,\mathrm{mm} &\leq {}_{\mathrm{O_W}}Z_{\mathrm{TM,x}} \leq 500\,\mathrm{mm} \\ -500\,\mathrm{mm} &\leq {}_{\mathrm{O_W}}Z_{\mathrm{TM,y}} \leq 500\,\mathrm{mm} \\ 1000\,\mathrm{mm} &\leq {}_{\mathrm{O_W}}Z_{\mathrm{TM,z}} \leq 2000\,\mathrm{mm} \end{aligned} \tag{6-1}$$

bilden in x- und y-Richtung die maximalen Verfahrwege der Portalachsen und in z-Richtung die Möglichkeiten des Tricept ab. Prinzipiell ist es möglich, Positionen außerhalb

[10] Zur Erzeugung der Zufallszahlen wurde die Funktion *rand()* in *MATLAB* genutzt, welche gleichverteilte Zufallszahlen ausgibt.

des Verfahrbereichs des Portals durch den Tricept anzufahren, dieser Fall wird bei der Erzeugung der Sollpositionen jedoch vernachlässigt. Es werden jeweils N Zufallszahlen für x-, y- und z-Richtung in der Reihenfolge ihrer Erzeugung zu einer Sollpositionsfolge zusammengefasst. Um bei den späteren Analysen auszuschließen, dass eine einzelne generierte Sollpositionsfolge hinsichtlich Länge oder enthaltenen Sollpositionen für einen der zu vergleichenden Ansätze besser geeignet ist, werden zum einen Folgen der unterschiedlichen Längen $N = 10$, $N = 20$, $N = 50$, $N = 100$, $N = 200$ und $N = 1000$ erzeugt. Zum anderen werden je N 1000 unterschiedliche Folgen generiert.

In Bild 6-1 ist exemplarisch eine Folge mit $N = 100$ Sollpositionen dargestellt, wobei die Zahlen die Reihenfolge angeben. Es ist zu erkennen, dass sich etwa die Sollposition „81“ (roter Kreis) bezogen auf x- und y-Richtung am Rand des Arbeitsraumes befindet, während die Position „100“ (grüner Kreis) nah am Zentrum liegt. Insgesamt liegt eine Verteilung über nahezu den gesamten Arbeitsraum vor.

Um die Sollpositionsfolgen im weiteren Verlauf der vorliegenden Arbeit unterscheiden zu können, werden sie nach dem Schema rN_j benannt, wobei r für *random* (engl. zufällig) steht, N wie bekannt die Länge der Sollpositionsfolge bezeichnet und j angibt, um welche der 1000 Folgen je N es sich handelt. Die dritte Folge für $N = 100$ heißt somit $r100_3$.

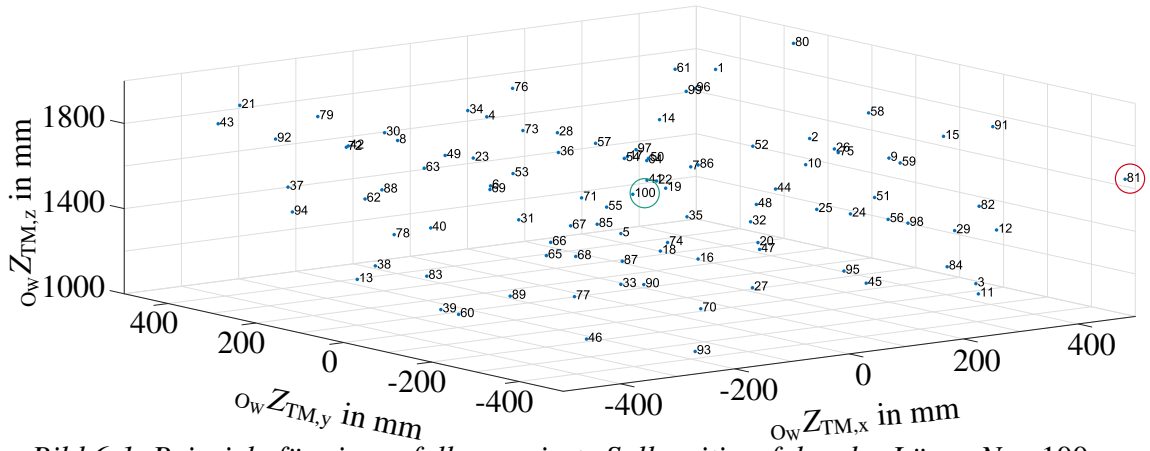


Bild 6-1: *Beispiele für eine zufallsgenerierte Sollpositionsfolge der Länge* $N = 100$

Im Folgenden werden die generierten Folgen im Rahmen der simulativen Umsetzung des Referenzverfahrens eingesetzt.

6.2 Simulative Ausführung des Referenzverfahrens

Wie in Abschnitt 4.5 erläutert, dient das Referenzverfahren dazu, die mittels des modellbasierten Optimierungsansatzes erzielten Ergebnisse einordnen und bewerten zu können. Dazu wird das Verfahren auf alle generierten Sollpositionsfolgen angewendet. Je Folge wird mittels des Optimierers eine für die gesamte Folge feste und hinsichtlich der Gesamtverfahrzeit optimierte Aufteilung $\underline{\xi}_{\text{ref}}$ der Bewegung in x-und y-Richtung zwischen Portal und Tricept ermittelt. Bezüglich des Optimierers wird im Rahmen der simulativen Ausführung des Referenzansatzes auf die *MATLAB*-eigene *Optimization Toolbox* zurückgegriffen. Diese stellt verschiedene Funktionen sowohl für lineare als auch quadratische, ganzzahlige und nichtlineare Optimierungsaufgaben zur Verfügung [Mat20]. Wie bereits eingangs

dieses Kapitels beschrieben, wird ein Optimierer für nichtlineare Optimierungsaufgaben mit mehreren Variablen benötigt, für die Ausführung des Referenzansatzes zusätzlich mit der Möglichkeit der Vorgabe von Nebenbedingungen. Diese Anforderungen erfüllt *fmincon*, wobei innerhalb dieser Funktion zwischen verschiedenen Optimierungsalgorithmen (*interior-point*, *sqp*, *sqp-legacy*, *active-set* und *trust-region-reflectiv*) gewählt werden kann [Mat20]. Im Rahmen der folgenden Analysen wurde die Standardeinstellung *interior-point* beibehalten. Die Tabelle 6-1 fasst die weiteren relevanten Parameter der Optimierung zusammen.

Tabelle 6-1: *Parameter für fmincon bei Ausführung des Referenzansatzes*

Parameter	Wert
Optimierer	fmincon
n_{it}	100000
$\min(\Delta q_5, \Delta q_6)$	1×10^{-12} mm
$\min(\Delta J_{\mathrm{rel}})$	1×10^{-11} s
$\min(q_1 \dots q_3)$	387,38 mm
$\max(q_1 \dots q_3)$	487,38 mm
$\min(q_4)$	400,00 mm
$\max(q_4)$	1800,00 mm
$\min(q_5, q_6)$	−500,00 mm
$\max(q_5, q_6)$	500,00 mm

Mittels n_{it} wird die maximale Anzahl an zulässigen Iterationen des Optimierers angegeben. Durch einen hohen Wert wird verhindert, dass der Optimierer frühzeitig aufgrund der Iterationen beendet und damit eingeschränkt wird. Zusätzlich werden mittels $\min(\Delta q_5, \Delta q_6)$ und $\min(\Delta J_{\mathrm{rel}})$ Abbruchgrenzen hinsichtlich der minimalen Änderung der Optimierungsgrößen q_5 und q_6 sowie der minimalen durch die Variation der Optimierungsgrößen erreichte Änderungen der Gütefunktion definiert. Die entsprechenden Werte werden klein gewählt, um mit dem Optimierer möglichst gute Ergebnisse zu erzielen. Es wird davon ausgegangen, dass sich die beiden Teilmechanismen Tricept und Portal zu Beginn jeder Sollpositionsfolge in ihrer Intitalposition ($q_i = q_{i,0}$) befinden. Zudem wird als Startwert für den Optimierer angenommen, dass das Portal die gesamte Bewegung in x- und y-Richtung übernimmt. Durch die Wahl aller Sollpositionen innerhalb des Arbeitsraumes des Portals ist diese Aufteilung mit den Möglichkeiten der Teilmechanismen vereinbar. Es liegen somit Startwerte der Optimierungsgrößen vor, mit denen die Nebenbedingungen eingehalten werden.

Als relevantes Ergebnis für einen späteren Vergleich mit dem modellprädiktiven Ansatz wird die insgesamt je Folge erforderliche Verfahrzeit betrachtet. Es wird zum einen der Mittelwert der Verfahrzeit

$$\overline{T}_{\mathrm{ref}}(rN) = \frac{\sum_{j=1}^{1000} J_{\mathrm{ref}}(rN_j)}{1000} \tag{6-2}$$

mit J_{ref} siehe Gleichung (4-24) über die jeweils 1000 Folgen rN_j der Länge N mit $j = 1 \ldots 1000$ gebildet. Zum anderen werden die kleinste und größte Verfahrzeit

$$T_{\mathrm{ref,min}}(rN) = \min(J_{\mathrm{ref}}(rN_j)) \text{ mit } j = 1 \ldots 1000 \tag{6-3}$$

und

$$T_{\mathrm{ref,max}}(rN) = \max(J_{\mathrm{ref}}(rN_j)) \text{ mit } j = 1 \ldots 1000 \tag{6-4}$$

ermittelt. Die Tabelle 6-2 fasst die Ergebnisse zusammen.

Tabelle 6-2: *Ergebnisse des Referenzverfahrens*

N	$\overline{T}_{\mathrm{ref}}(rN)$	$T_{\mathrm{ref,min}}(rN)$	$T_{\mathrm{ref,max}}(rN)$
10	33,71 s	17,02 s	55,63 s
50	189,50 s	139,26 s	246,44 s
100	388,58 s	322,81 s	465,92 s
200	788,49 s	686,84 s	887,57 s
1000	4028,76 s	3724,25 s	4286,07 s

An den Abweichungen zwischen den drei Zeiten $\overline{T}_{\mathrm{ref}}$, $T_{\mathrm{ref,min}}$ und $T_{\mathrm{ref,max}}$ je N ist zu erkennen, dass sich die einzelnen Sollpositionsfolgen hinsichtlich der zwischen den Sollpositionen zu verfahrenden Strecken aufgrund der Nutzung des Zufallsgenerator wie erwartet unterscheiden.

Im folgenden Abschnitt wird die simulative Funktionsvalidierung des modellprädiktiven Ansatzes vorgestellt und die dabei erzielten Ergebnisse mit denen aus Tabelle 6-2 verglichen.

6.3 Funktionsvalidierung des modellprädiktiven Ansatzes in *MATLAB* unter Nutzung *MATLAB*-eigener Optimierer

Um die grundlegende Funktionsweise des modellprädiktiven Ansatzes in Bezug auf die Minimierung der Verfahrzeit eines kinematisch redundanten Mechanismus zwischen aufeinanderfolgenden Sollpositionen zu validieren, wird im Rahmen der vorliegenden Arbeit zunächst eine Simulation in *MATLAB* durchgeführt. Wie bei der simulativen Ausführung des Referenzverfahrens (Abschnitt 6.2) wird hierzu der Optimierer *fmincon* der *MATLAB*-eigenen *Optimization Toolbox* mit den in Tabelle 6-3 angegebenen Parametern genutzt. Dabei wurden die Abbruchgrenzen $\min(\Delta q_5, \Delta q_6)$ und $\min(\Delta J_{\mathrm{rel}})$ größer gewählt als beim Referenzverfahren, um sie den tatsächlichen Möglichkeiten des Mechanismus anzupassen. Insbesondere hinsichtlich der Änderung der Portalpositionen als Optimierungsvariablen liegen sie jedoch weiterhin in einem hohen Genauigkeitsbereich. Die Abbruchgrenze der Iterationsanzahl n_{it} wurde bei 100000 belassen, um den Optimierer hier nicht einzuschränken. Auch beim modellprädiktiven Ansatz werden die Teilmechanismen Tricept und Portal zu Beginn jeder Sollpositionsfolge in ihrer Initialstellung verortet. Initial wird

zudem für jede Position innerhalb des Prädiktionshorizontes angenommen, dass das Portal die gesamte Bewegung in x- und y-Richtung übernimmt. Äquivalent zu der Ausführung des Referenzansatzes wird somit sichergestellt, dass mit den Startwerten die Grenzen des Mechanismus eingehalten werden. Nachdem der betrachtete Bereich das erste mal verschoben wurde, werden für die ersten $n_\mathrm{p} - 1$ Sollpositionen die Ergebnisse des vorhergegangenen Optimierungsschrittes als Startwerte genutzt (vgl. Kapitel 4). Lediglich für die letzte Sollposition $\vec{Z}_{\mathrm{TM},k+\mathrm{n_p}}$ wird initial die vollständige Ausführung durch das Portal angenommen.

Tabelle 6-3: *Parameter für fmincon bei Ausführung der modellprädiktiven Planung*

Parameter	Wert
Optimierer	fmincon
n_p	3
n_it	100000
$\min(\Delta q_5, \Delta q_6)$	1×10^{-5} mm
$\min(\Delta J_\mathrm{rel})$	1×10^{-4} s
w_1	50
w_2	10
w_3	10
$q_{i,\mathrm{os}}$	5 mm

Neben den bereits in Tabelle 6-2 vorgestellten Daten werden in Tabelle 6-4 die sich ergebenden mittleren Verfahrzeit $\overline{T}_\mathrm{ot}$ (ot: ***O**ptimization **T**oolbox*) sowie die jeweiligen Minimal- und Maximalwerten $T_\mathrm{ot,min}$ und $T_\mathrm{ot,max}$ je N für die modellprädiktive Optimierung angegeben. Die Berechnung erfolgt analog zu den Gleichungen (6-2), (6-3) und (6-4), wobei anstelle von J_ref aus Gleichung (4-24) die Berechnung der insgesamt erforderlichen Verfahrzeit

$$\sum_{k=0}^{N-1} \max_{i=1\ldots6} \Delta t_i(\Delta q_{i,k+1|k}) \tag{6-5}$$

genutzt werden muss. Zusätzlich wird die prozentuale Einsparung

$$\Delta\overline{T}_\mathrm{ot} = \frac{\overline{T}_\mathrm{ot} - \overline{T}_\mathrm{ref}}{\overline{T}_\mathrm{ref}} \tag{6-6}$$

des Optimierungsansatzes bezogen auf das Referenzverfahren angegeben. Es wird auf die explizite Kennzeichnung der Abhängigkeit der Zeiten von der jeweiligen Sollpositionsfolge und der Länge N aus Gründen der besseren Lesbarkeit verzichtet.

Wie zu erwarten steigen die Verfahrzeiten bei beiden Ansätzen näherungsweise linear mit der Länge N der Sollpositionsfolgen. Die Verfahrzeiten des modellprädiktiven Ansatzes sind jedoch zwischen −20,14 % und −30,80 % geringer als die des Vergleichsansatzes. Der

Tabelle 6-4: *Ergebnisse des Referenzverfahrens und des modellprädiktiven Ansatzes*

N	$\overline{T}_{\text{ref}}$	$\overline{T}_{\text{ot}}$	$T_{\text{ot,min}}$	$T_{\text{ot,max}}$	$\Delta\overline{T}_{\text{ot}}$
10	33,71 s	26,92 s	14,57 s	44,84 s	−20,14 %
50	189,50 s	138,41 s	101,00 s	176,36 s	−26,96 %
100	388,58 s	278,20 s	233,98 s	334,96 s	−28,41 %
200	788,49 s	555,34 s	500,12 s	627,41 s	−29,57 %
1000	4028,76 s	2788,00 s	2614,70 s	2946,40 s	−30,80 %

Grund für diesen großen Unterschied in den Verfahrzeiten liegt vor allem darin begründet, dass bei größeren Variationen der Sollpositionen innerhalb einer Folge die Wahl der festen Aufteilung nachteilig ist. Die maximal möglichen Bewegungen des Tricept-TCP bzw. die Ausdehnung des Arbeitsraumes in x- und y-Richtung hängen stark von der z-Koordinate ${}_{\text{Ow}}Z_{\text{TM,z}}$ der Sollposition und der damit verbundenen Länge der seriellen Achse M_4 ab (vgl. Abschnitt 5.2.1). Liegt eine Sollposition vor, bei der diese nur geringfügig ausgefahren ist (kleiner Wert für q_4 nahe $q_{4,0}$), so kann der Tricept nur kleine Bewegungen in x- und y-Richtung ausführen. In Kombination mit einer großen durchzuführenden Gesamtbewegung in x- und/oder y-Richtung muss ein Großteil davon zwangsläufig durch das Portal übernommen werden. Ist hingegen Achse M_4 aufgrund der geforderten Sollposition weit ausgefahren, so wird der Arbeitsraum des Tricept in x- und y-Richtung vergrößert. Bewegungen in diese beiden Richtungen erfordern geringere Verfahrwege Δq_1, Δq_2 und Δq_3 der parallelen Achsen M_1, M_2 und M_3. Durch den vorgestellten modellbasierten Optimierungsansatz wird diese Eigenschaft des Tricept berücksichtigt und für jede Sollposition einer Folge eine individuelle Aufteilung der Bewegung ermittelt, was zu einer Minimierung der Verfahrzeit führt. Bei dem Vergleichsansatz hingegen sind die Aufteilungen und damit der Vektor $\underline{\xi}_{\text{ref}}$ für alle Sollpositionen einer Folge konstant und maßgeblich abhängig von den Sollpositionen mit gering ausgefahrener serieller Achse bei zeitgleich großen Verfahrstrecken in x- oder y-Richtung. Dies führt insbesondere bei Sollpositionen mit weit ausgefahrener Achse M_4 zu einer vergleichsweise langen erforderlichen Verfahrzeit. Je mehr Sollpositionen eine Folge umfasst, desto ausgeprägter wird der Einfluss dieses Effekts. Somit lässt sich als Zwischenfazit festhalten, dass Ansätze mit festen Aufteilungen bei individuellen Fertigungsaufgaben mit stark unterschiedlichen Verfahrwegen nicht mehr effizient einsetzbar sind, die vorgestellte Optimierung hier jedoch deutliche Reduzierungen der Verfahrzeit ermöglicht.

Die bisher durchgeführte Analyse unter Nutzung von *fmincon* zeigt bereits das Potential des modellprädiktiven Optimierungsansatzes. Um einen Einsatz auf industrieller Hardware zu ermöglichen, wird jedoch ein Optimierer benötigt, der unabhängig von einer bestimmten Software ist und möglichst als Quellcode in der Programmiersprache *C* oder *C++* vorliegt. Im folgenden Abschnitt werden verschiedene Optimierungsbibliotheken und die in ihnen enthaltenen Optimierungsalgorithmen diesbezüglich analysiert.

6.4 Funktionsvalidierung des modellprädiktiven Ansatzes in *MATLAB* unter Nutzung freier Optimierer

Im Rahmen der vorliegenden Arbeit wird ein Optimierungsalgorithmus benötigt, der neben der Lösung des Optimierungsproblems eine Implementierung in Form von *C*- oder *C++*-Code ermöglicht, ein entscheidendes Kriterium für den Einsatz auf einer SPS. Zudem wird aufgrund der Komplexität des Optimierungsproblems inklusive der Gleichungen der kinematischen Probleme und des Modells ein ableitungsfreier Ansatz bevorzugt. Durch die Fallunterscheidung innerhalb des Zeitmodells liegt kein kontinuierliches Modell vor, von der Ableitung ist an dieser Stelle abzusehen.

Im folgenden Abschnitt wird die Analyse verschiedener Open Source-Optimierungsbibliotheken und ihrer Optimierungsalgorithmen vorgestellt, bevor im Anschluss die Implementierung in *MATLAB* und die erzielten Ergebnisse vorgestellt werden.

6.4.1 Analyse verschiedener Optimierungsalgorithmen und ihrer Bibliotheken

Die Tabelle 6-5 zeigt zunächst eine Übersicht über verschiedene Algorithmen aus frei verfügbaren Open Source-Bibliotheken und die entscheidenden Eigenschaften, sie orientiert sich zum Teil an [Abd17]. Die in der Tabelle angegebenen Quellen beziehen sich auf Veröffentlichungen zu den Optimierungsalgorithmen selber, die Bibliotheken werden im Anschluss daran vorgestellt.

Tabelle 6-5: *Auswahl an im Rahmen von Bibliotheken nutzbaren Optimierern*

Algorithmus	Bibliothek	Global/Lokal	Ableitungsfrei	Nebenbedingungen	Linear/Nichtlinear	Literatur
ALGENCAN	TANGO, pyOpt	Lokal	Nein	Bound-Constraints[11]	Nichtlinear	[ABMS08]
ALPSO	pyOpt	Global	Ja	Ja	Nichtlinear	[LY09], [JP11]
BOBYQA	Dlib, NLopt, ZHANG	Lokal	Ja	Bound-Constraints	Nichtlinear	[Pow09]
COBYLA	NLopt, pyOpt, ZHANG	Lokal	Ja	(Un-)gleichheits-nebenbedingungen	Nichtlinear	[Pow94], [Pow07]
CONMIN	OpenMDAO, pyOpt	lokal	Nein	(Nicht-)lineare Ungleichheits-nebenbedingungen	(Nicht-)linear	[Van73]
DIRECT	NLopt	Global	Ja	Bound-Constraints	Nichtlinear	[JPS93]
FSQP	pyOpt	lokal	Nein	(Nicht-)lineare (Un-)Gleichheits-nebenbedingungen	(Nicht-)Linear	[LT96]
filterSD[12]	pyOpt	Lokal	Nein	Lineare Nebenbedingungen	Nichtlinear	-

GCMMA	pyOpt	Lokal	Nein	Bound-Constraints, Nichtlineare Nebenbedingungen	Nichtlinear	[Sva95], [Sva02], [Sva14]
GENCAN[13]	Open Optimization Library	Global	Nein	Bound-Constraints	Nichtlinear	[BM02]
Genetic Algorithm	openGA	Lokal	Ja	-[14]	-	[MAM+17]
IPM	Ipopt	Lokal	Nein	(Nicht-)lineare Nebenbedingungen, Bound-Constraints	Nichlinear	[RB05], [ABO05], [WB06]
ISRES	NLopt	Global	Ja	Bound-Constraints, (Nicht-)lineare Ungleichheitsnebenbedingungen	Nichlinear	[RY00], [RY05]
KSOPT	pyOpt	Lokal	Nein	Ja	Nichtlinear	[Wre89]
(L-)BFGS Quasi-Newton	Dlib	Lokal	Nein	Nein	Nichtlinear	[LN89], [BLNZ95]
LINCOA	ZHANG	Lokal	Ja	Lineare Nebenbedingungen	Nichtlinear	[Pow15]

MIDACO	pyOpt	Lokal	Ja	(Un-) Gleichheits-nebenbedingungen, Bound-Constraints	Nichtlinear (nicht-konvex)	[SEB09]
MMA	pyOpt, NLopt	Lokal	Nein	(Nicht-)lineare Nebenbedingungen, Bound-Constraints	Nichtlinear	[Sva87]
MMFD	pyOpt	Lokal	Nein	Ungleichheits-nebenbedingungen	Nichtlinear	[Van83]
Nelder-Mead Simplex	NLopt	Lokal	Nein	Bound-Constraints	Nichtlinear	[NM65], [She74]
NEWUOA	NLopt, ZHANG	Global	Nein	Nein	Nichtlinear	[Pow04]
PRAXIS	NLopt	Lokal	Ja	Nein	Nichtlinear	[Bre73]
Sbplx (Subplex)	NLopt	Lokal	Ja	Nein	Nichtlinear	[Row90]
SDPEN	pyOpt	Lokal	Ja	(Nicht-)lineare Nebenbedingungen, Bound-Constraints	Nichtlinear	[LLS10]
SLSQP	pyOpt, NLopt	Lokal	Nein	(Un-)Gleichheits-nebenbedingungen	Nichtlinear	[Kra88]

SOLVOPT	pyOpt	Lokal	Nein	(Un-)Gleichheits-nebenbedingungen, Bound-Constraints	Nichtlinear	[KK97]
UOBYQA	ZHANG	Lokal	Ja	Nein	Nichtlinear	[Pow02]

[10] Sogenannte Bound-Constraints (manchmal auch als Box-Constraints bezeichnet) sind Nebenbedingungen die angeben, dass eine Optimierungsvariable x zwischen einem minimalen Wert $x_{\min}$ und einem maximalen Wert $x_{\max}$ liegen muss.

[11] Bei filterSD handelt es sich um einen Algorithmus von Professor R. Fletcher, jedoch ist nicht erwähnt, ob der Name ein Akronym darstellt. Es wird für filterSD auf eine Sammlung an Veröffentlichungen von Prof. Fletcher verwiesen [Hor11].

[12] Die Bedeutung der Bezeichnung GENCAN, und ob es sich dabei um ein Akronym handelt, wird in den betrachteten Quellen nicht erläutert.

[13] Hier waren den Quellen keine eindeutigen Angaben zu entnehmen.

Im Folgenden werden die einzelnen Bibliotheken, deren Optimierer in Tabelle 6-5 aufgelistet sind, kurz vorgestellt.

Dlib

Die Bibliothek *Dlib* stellt unter der *Boost Software Licence*[15] kostenfrei verschiedene Algorithmen u.a. für Optimierungs- und Machine Learning-Aufgaben zur Verfügung. Präferierte Programmiersprache ist *C++*, jedoch ist auch eine Nutzung unter *Python* möglich. [Kin09]

Ipopt

Die in *C++* geschriebene *Ipopt*-Bibliothek (Interior Point OPTimizer) beinhaltet verschiedene Algorithmen für nichtlinear Optimierung und wird unter der *Eclipse Public License* (EPL)[16] bereitgestellt. Neben *C++* können auch Bibliotheken für den Einsatz in den Sprachen *C*, *Fortran*, *Java* und *R* sowie in *MATLAB* oder *AMPL* erzeugt werden. [WB06]

NLopt

Bei *NLopt* (NonLinear OPTimization) handelt es sich um eine open-source-Bibliothek für nichtlineare lokale und globale Optimierung, die Schnittstellen zu den Programmiersprachen *C*, *C++*, *Fortran*, *Python*, *GNU Guile*, *Julia*, *GNU R*, *Lua* und *OCaml* sowie zu *MATLAB* bietet. Die Bibliothek wurde von Steven G. Johnson im Rahmen seiner Tätigkeit am *Massachusetts Institute of Technology* entwickelt und ist unter der *GNU Lesser General Public License*[17] verfügbar [Joh16].

openGA

openGA (OPEN Genetic ALgorithm) ist eine von Mohammadi et al. entwickelte *C++*-Bibliothek für Optimierung basierend auf genetischen Algorithmen. Sie wird unter der *Mozilla Public License* Version 2.0[18] frei zur Verfügung gestellt. [MAM+17]

OpenMDAO

Bei *openMDAO* (OPEN Multidisciplinary Design, Analysis and Optimization) handelt es sich um eine freie, auf *Python* basierende, Plattform für Optimierung. Je nach genutztem Algorithmus bzw. Bereich von *openMDAO* gibt es unterschiedliche Quellen, es wird an dieser Stelle auf die entsprechende Quellenübersicht [ope18] der Webseite verwiesen.

[15] Unter der *Boost Software Licence* bereitgestellte Software darf von jeder Person oder Organisation kostenlos genutzt, verändert, verteilt oder vertrieben werden. In jeder Kopie oder jedem Ableger der Software muss diese Lizenz wiederum enthalten sein. [DFA03]

[16] Die *Eclipse Public License* erlaubt die freie Nutzung, Veränderung und Verbreitung der Software. Dabei gibt ein Copyleft, bei dem zwar neue Module nicht offengelegt werden müssen, veränderte Module der Originalquelle jedoch. Zudem muss die EPL in den Modulen erhalten bleiben. [The17]

[17] Unter der *GNU Lesser General Public License* bereitgestellte Software (open source) kann frei genutzt, verändert und weitergegeben werden, darauf basierende Software darf verkauft werden. Gegenüber der *GNU General Public License* ist hier das Copyleft schwächer. Dem Endnutzer ist nicht der gesamte Quellcode offen zu legen, er muss lediglich die Möglichkeit erhalten, die aus der ursprünglichen *GNU Lesser General Public License*-Software stammenden Teile modifizieren zu können [Fre07b].

[18] Die *Mozilla Public License* weißt, ähnlich wie die *Lesser General Public License*, ein schwaches Copyleft auf. Nutzer können Softwareänderungen durchführen und diese verbreiten, zudem kann eine Kombination mit anderer Software (auch unter anderen Lizenzen) erfolgen. [Moz12], [Moz18]

Open Optimization Library
: Die in *C* geschriebene *Open Optimization Library* beinhaltet verschiedene Algorithmen für Optimierung mit Nebenbedingungen und ist frei unter der *GNU General Public License*[19] nutzbar. [BDV05]

pyOpt
: *pyOpt* ist eine in Python geschriebene Bibliothek für nichtlineare Optimierung mit Nebenbedingungen. Die Bibliothek selber steht unter der *GNU Lesser General Public License*, jedoch gibt es je nach Optimierungsalgorithmus noch weitere zu beachtende Lizenzen, etwa die *Eclipse Public License*. [PJM12]

TANGO
: *TANGO* (Trustable Algorithms for Nonlinear General Optimization) ist eine unter der *GNU General Public License* frei verfügbare *Fortran*-Bibliothek für Optimierer, die im Fachbereich angewandte Mathematik der *State University of Campinas* und dem Fachbereich Computer Science der *University of São Paulo* unter Leitung von *Prof. Martinez* entwickelt wurde. Neben *Fortran* existieren Schnittstellen zu *AMPL*, *C*/*C*++, und *CUTEst*. Für die einzelnen Optimierer existieren unterschiedliche Quellen, es wird an dieser Stelle auf die Webseite des *TANGO*-Projekts [Mar05] verwiesen.

ZHANG
: Hierbei handelt es sich weniger um eine Bibliothek denn um von Assistenzprofessor *Z. Zhang* (*Hong Kong Polytechnic University*, Fachbereich angewandte Mathematik) auf seiner Webseite bereitgestellt Downloads des *FORTRAN*-Quellcodes einzelner Optimierer [Zha17].

Nach der Analyse der verschiedenen Optimierungsbibliotheken wird der Fokus der Betrachtung auf die Bibliothek *NLopt* und die darin enthaltenen Algorithmen gelegt. Zum einen stellt sie unterschiedliche Optimierer zur Verfügung, die hinsichtlich ihrer Eignung für das zu lösende Optimierungsproblem im Folgenden analysiert werden. Zum anderen bietet die *NLopt*-Bibliothek sowohl Schnittstellen zur Nutzung in *C* als auch für *MATLAB*. Dies ist der entscheidende Unterschied etwa zu den Algorithmen *SDPEN* sowie *MIDACO* bzw. zu der diese Algorithmen beinhaltenden Bibliothek *pyOpt*, die lediglich aus *Python* heraus genutzt werden kann. Für den Einsatz auf einer SPS ist *Python* bezogen auf den aktuellen Stand der Technik nicht einsetzbar. Des Weiteren kann die im Rahmen der vorliegenden Arbeit entwickelte modellprädiktive Planung aufgrund der Lizenzierung von *NLopt* auf Basis der *GNU Lesser General Public License* später vermarktet werden, ohne dass der Quellcode vollständig offengelegt werden muss.

Bei der Gegenüberstellung werden die Algorithmen *BOBYQA*, *COBYLA*, *Nelder-Mead Simplex*, *PRAXIS* und *Sbplx* anhand der zufällig generierten Sollpositionsfolgen mit $N = 10$, $N = 50$ und $N = 1000$ miteinander hinsichtlich der erzielten durchschnittlichen Verfahrzeit $\overline{T}_{\mathrm{C}}$ und der dafür erforderlichen durchschnittlichen Rechenzeit $\overline{T}_{\mathrm{calc,c}}$ verglichen. Die Rechenzeit gibt an, wie lange der Optimierer benötigt, um ausgehend von einer aktuellen Position das Optimierungsproblem für den definierten Prädiktionshorizont zu lösen.

[19] Unter der *GNU General Public License* bereitgestellte Software kann frei genutzt, verändert und weitergegeben werden, darauf basierende Software darf verkauft werden. Es gilt jedoch, dass die Rechte ebenfalls weitergegeben werden müssen (Copyleft). Somit muss insbesondere bei kommerzieller Nutzung etwa der Quellcode inklusive der durchgeführten Änderungen frei oder auf Anfrage hin dem Endnutzer zugänglich sein. [Fre07a]

Die Algorithmen *DIRECT*, *ISRES*, *MMA*, *NEWUOA* und *SLSQP* werden nicht weiter betrachtet, da sie entweder nicht ableitungsfrei oder für globale Optimierung gedacht sind. Letzteres erfordert in der Regel jedoch einen höheren Rechenaufwand, was für einen echtzeitfähigen Einsatz auf der SPS hinderlich ist.

Die Ergebnisse des Vergleichs sind in Bild 6-2 dargestellt, die Implementierung von Zeitmodell und Gütefunktion sowie der Aufruf der Optimierer erfolgen dabei in *C*. Dies ist die später auf der SPS erforderliche Programmiersprache, somit sind die erzielten Ergebnisse näher am späteren Zielsystem. Ausgeführt wird die Analyse auf einem Computer mit *Intel i5*-Prozessor (2,7 GHz) mit den in Tabelle 6-6 angegebenen Parametern. Diese sind mit Ausnahme der eingesetzten Optimierer äquivalent zu denen bei der Analyse mit *fmincon*, wodurch die erforderliche Vergleichbarkeit gegeben ist.

Tabelle 6-6: *Parameter für den Vergleich der Optimierer*

Parameter	Wert
Optimierer	BOBYQUA COBYLA Nelder-Mead PRAXIS Sbplx
n_p	3
n_it	100000
$\min(\Delta q_5, \Delta q_6)$	1×10^{-5} mm
$\min(\Delta J_\mathrm{rel})$	1×10^{-4} s
w_1	50
w_2	10
w_3	10
$q_{i,\mathrm{os}}$	5 mm

Es ist in den Resultaten zu erkennen, dass der Algorithmus *Sbplx* [Joh19] (Neuimplementierung des *Subplex*-Algorithmus von T. Rowan [Row90]) bei allen drei Sollfolgenlängen N die beste Verfahrzeit erzielt, dafür allerdings eine höhere Rechenzeit als *BOBYQA*, *Nelder-Mead* und *PRAXIS* benötigt. Da die erforderliche Rechenzeit jedoch unterhalb der angestrebten Zykluszeit von einer Millisekunde bleibt, wird *Sbplx* aufgrund der kürzeren Verfahrzeiten im weiteren Verlauf der Arbeit eingesetzt. Die Schwankungen der Rechenzeiten zwischen den Längen N lassen sich durch die Ausführung auf einem Computer erklären. Hierbei handelt es sich nicht um ein Echtzeitsystem mit fester Zykluszeit, vielmehr werden die Prozesse bzw. ihre Ausführung über sogenannte *Scheduler* geregelt. Ein Prozess läuft damit abhängig von den anderen gleichzeitig ausgeführten Prozessen nur eine bestimmte Zeit, wie lange diese ist, hängt von der Auslastung und den Ressourcen des Computers ab. Belasten mehrere Prozesse das System (etwa Hintergrundprozessse oder Update-Prozesse andere Programme), so steigt die Rechenzeit für den betrachteten modellprädiktiven Ansatz damit an. Für eine Abschätzung der erforderlichen Rechenzei-

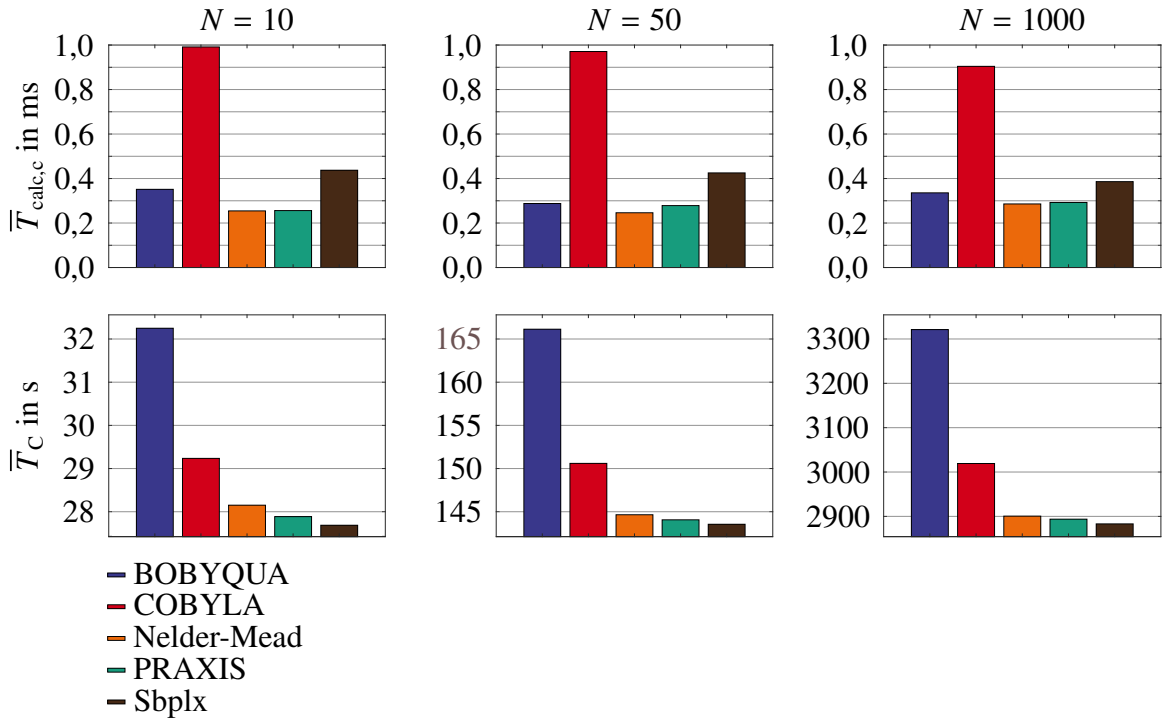


Bild 6-2: *Mittlere Rechenzeit* $\overline{T}_{calc,c}$ *und Verfahrzeit* $\overline{T}_C$ *der ausgewählten Optimierungsalgorithmen für* $N = 10$, $N = 50$ *und* $N = 1000$

ten genügen die vorgestellten Ergebnisse jedoch auch aufgrund der Mittlung über 1000 unterschiedliche Sätze. Die erzielten Verfahrzeiten sind von der Auslastung des Computers unabhängig.

6.4.2 Ergebnisse der Mehrzieloptimierung mit ausgewähltem Optimierungsalgorithmus

Um den ausgewählten Algorithmus *Sbplx* zunächst weiter zu analysieren, wird trotz der im vorherigen Abschnitt vorgestellten Implementierung in *C* nun auf *MATLAB* und die im Rahmen der *NLopt*-Bibliothek angebotene Schnittstelle zurückgegriffen. Hierdurch kann *Sbplx* äquivalent zu *MATLAB*-eigenen Optimierungsalgorithmen wie *fmincon* genutzt werden. Die Ergebnisse können sowohl mit denen der *Optimization Toolbox* als auch mit dem Referenzansatz verglichen werden. Es sind gegenüber *fmincon* lediglich die Algorithmusspezifischen Anpassungen etwa hinsichtlich der Formulierung des Gütefunktionals oder der Grenzen der Optimierung vorzunehmen (Parameter siehe Tabelle 6-7).

In Tabelle 6-8 sind die mittleren Verfahrzeiten $\overline{T}_{sm}$ (sm: ***S**bplx **M**atlab*) sowie die jeweiligen Minimal- und Maximalwerten $T_{sm,min}$ und $T_{sm,max}$ den vorab vorgestellten Ergebnissen von Referenzverfahren und *fmincon* gegenüber gestellt. Die erzielten Einsparungen liegen dabei im Bereich von *fmincon*, somit erscheint *Sbplx* als geeigneter Optimierungsalgorithmus für die vorliegende Anwendung. Es gilt jedoch neben der reinen zeitlichen Einsparung auch zu analysieren, inwieweit eine der elementaren Grundüberlegungen hinsichtlich der hier vorgestellten modellprädiktiven Planung zeitoptimierter Trajektorien validiert werden kann: Bei einer optimierten Aufteilung der insgesamt durchzuführenden Bewegung des

Tabelle 6-7: *Parameter für Sbplx bei Ausführung der modellprädiktiven Planung in MATLAB*

Parameter	Wert
Optimierer	Sbplx
n_p	3
n_it	100000
$\min(\Delta q_5, \Delta q_6)$	1×10^{-5} mm
$\min(\Delta J_\mathrm{rel})$	1×10^{-4} s
w_1	50
w_2	10
w_3	10
$q_{i,\mathrm{os}}$	5 mm

Tabelle 6-8: *Ergebnisse des Referenzverfahrens und des modellprädiktiven Ansatzes mit fmincon und Sbplx in MATLAB*

N	$\overline{T}_\mathrm{ref}$	$\overline{T}_\mathrm{ot}$	$\overline{T}_\mathrm{sm}$	$T_\mathrm{sm,min}$	$T_\mathrm{sm,max}$	$\Delta\overline{T}_\mathrm{ot}$	$\Delta\overline{T}_\mathrm{sm}$
10	33,71 s	26,92 s	27,78 s	14,76 s	46,05 s	−20,14 %	−17,59 %
50	189,50 s	138,41 s	143,46 s	104,51 s	179,09 s	−26,96 %	−24,30 %
100	388,58 s	278,20 s	288,61 s	247,19 s	347,19 s	−28,41 %	−25,73 %
200	788,49 s	555,34 s	576,34 s	515,98 s	654,02 s	−29,57 %	−26,91 %
1000	4028,76 s	2788,00 s	2883,80 s	2763,00 s	3028,40 s	−30,80 %	−28,42 %

TCP auf die Teilmechanismen Portal und Tricept sollten sich beide nicht nur im Rahmen ihrer spezifischen Möglichkeiten so schnell wie möglich bewegen (was bereits durch die Nutzung der maximalen Beschleunigung und Verzögerung sowie der maximalen Verfahrgeschwindigkeit gegeben ist), sie sollten auch ihre jeweilige Endposition näherungsweise zeitgleich erreichen. Hierbei spielt die Vorgabe der Portalposition durch den Optimierer die entscheidende Rolle. In Bild 6-3 sind für zwei exemplarische Sollpositionsfolgen $r10_3$ und $r10_{10}$ (vgl. Abschnitt 6.1) bei $N = 10$ die Verfahrzeiten $\Delta q_{i,k+\nu|k}$ der jeweils langsamsten Achsen von Tricept ($M_1 \ldots M_4$) und Portal (M_5, M_6) dargestellt.

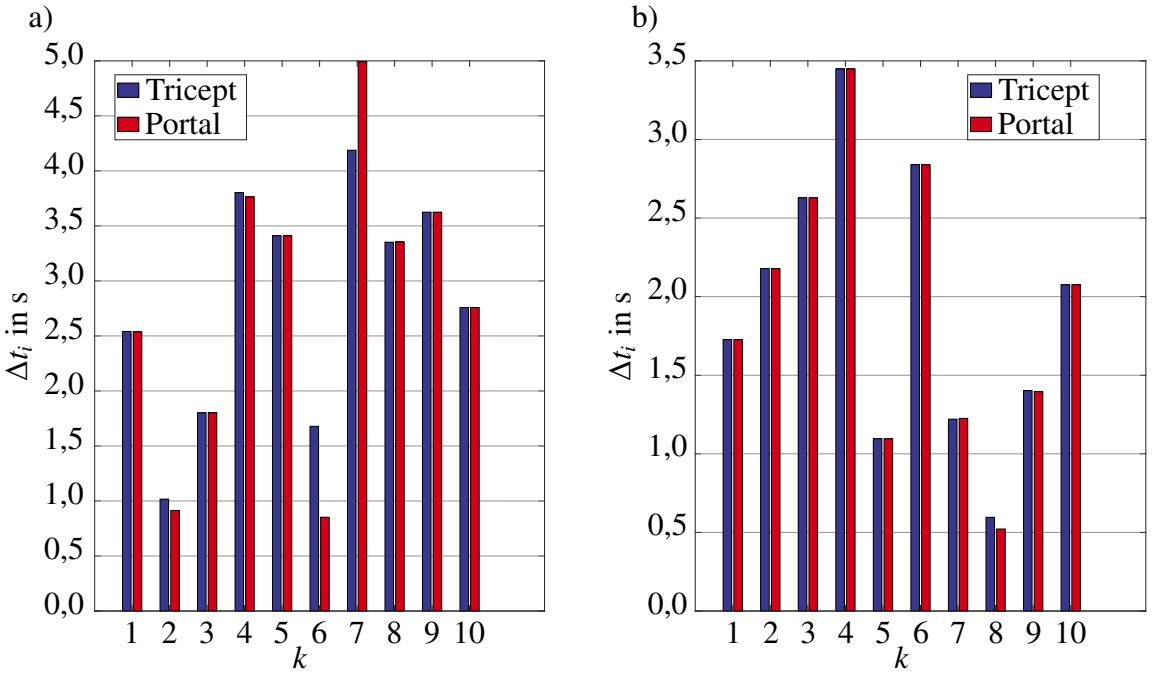


Bild 6-3: *Verfahrzeiten der jeweils langsamsten Achse von Tricept und Portal bei* $N = 10$ *für die Sollpositionfolgen a)* $r10_3$ *und b)* $r10_{10}$

Es ist zu erkennen, dass bei beiden Folgen für einen Großteil der Sollpositionen Portal und Tricept näherungsweise die gleiche Verfahrzeit benötigen. Bei der links dargestellten Folge (Bild 6-3 a)) gibt es insbesondere bei $k = 6$ und $k = 7$ Abweichungen. Im ersten Fall wird die längere Verfahrzeit des Tricept gegenüber dem Portal bestimmt durch den Antrieb M_4, da dieser als einziger die erforderliche Bewegung des TCP in z-Richtung übernehmen kann. Die langsamere Verfahrzeit des Portals im zweiten Fall liegt in der zu verfahrenden Distanz zwischen der Sollposition $\vec{Z}_{\mathrm{TM},6}$ und $\vec{Z}_{\mathrm{TM},7}$. Der Tricept ist nach 4,19 s in der Grenze eines Gelenks angelangt, das Portal verfährt die noch fehlende Strecke bis zum Erreichen der Sollposition. In beiden Fällen existiert somit keine Aufteilung der Gesamtbewegung zwischen den Teilmechanismen, mit der die Verfahrzeit weiter reduziert werden könnte, der modellbasierte Optimierungsansatz erfüllt somit die Anforderungen. Bei der Folge mit den rechts dargestellten Verfahrzeiten (Bild 6-3 b)) verhält es sich vergleichbar.

Nachdem die bisher erzielten Ergebnisse den Schluss zulassen, dass die modellprädiktive Planung zeitoptimierter Trajektorien für mehrachsige redundante Mechanismen gegenüber einer festen Bewegungsaufteilung signifikante Einsparungen der Verfahrzeit ermöglicht, gilt es nun die bisher getroffenen ersten Annahmen insbesondere hinsichtlich des zu

wählenden Prädiktionszeitraums n_p zu überprüfen. Dabei muss auch die für einen Berechnungsschritt erforderliche Rechenzeit $\overline{T}_\mathrm{calc,sm}$ betrachtet werden. Es ist an dieser Stelle zu beachten, dass durch Nutzung von *MATLAB* als Simulationssoftware Echtzeit kein Kriterium sein kann, trotzdem können die Änderungen in der Rechenzeit analysiert und daraus entsprechende Rückschlüsse gezogen werden.

In Bild 6-4 ist die erzielte Verfahrzeit als Mittelwert von 1000 verschiedenen Sollpositionsfolgen der Länge $N = 50$ bei Variation des Prädiktionshorizonts ($n_\mathrm{p} = 1 \ldots 5 \in \mathbb{N}$) dargestellt. Es ist deutlich zu erkennen, dass die Rechenzeit bei Erhöhung von n_p ansteigt, was den Ergebnissen von Pereira et al. in [PLA15] entspricht. Auch hier liegt dies darin begründet, dass mit der Vergrößerung des Prädiktionshorizonts die Anzahl der Optimierungsvariablen ansteigt. Während es bei $n_\mathrm{p} = 1$ nur $2n_\mathrm{p} = 2$ sind ($q_{5,k+1}$, $q_{6,k+1}$), müssen bei $n_\mathrm{p} = 3$ bereits $2n_\mathrm{p} = 6$ und bei $n_\mathrm{p} = 5$ schließlich 10 Variablen berücksichtigt werden. Da die erzielte Verfahrzeit ab $n_\mathrm{p} = 3$ jedoch insbesondere im Vergleich mit den Unterschieden zwischen $n_\mathrm{p} = 1$ und $n_\mathrm{p} = 2$ sowie $n_\mathrm{p} = 2$ und $n_\mathrm{p} = 3$ geringer sinkt, wird an dieser Stelle für alle weiteren Schritte der Prädiktionshorizont auf $n_\mathrm{p} = 3$ festgelegt.

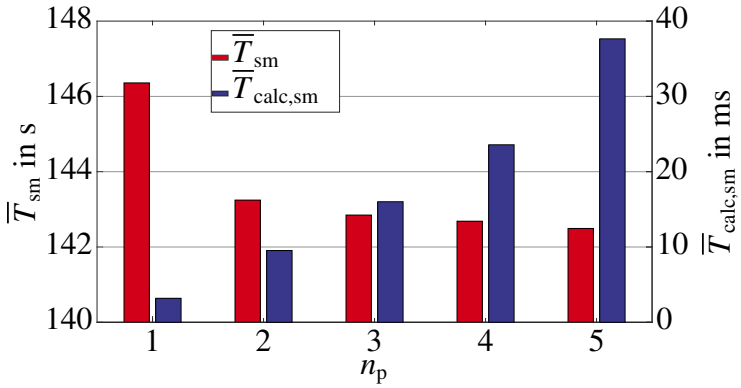


Bild 6-4: *Mittlere Verfahrzeit* $\overline{T}_\mathrm{sm}$ *und Rechenzeit* $\overline{T}_\mathrm{calc,sm}$ *der Umsetzung in MATLAB für* $N = 50$ *bei Variation des Prädiktionshorizonts* n_p

Für den gewählten Prädiktionshorizont muss zudem der Einfluss der maximal erlaubten Iterationen n_it betrachtet werden, da dieser Parameter nach [BKPJ14] eine entscheidende Rolle hinsichtlich der Rechenzeit sowie der erzielten Optimierungsergebnisse hat. Dazu wurde bei gewähltem Prädiktionshorizont der Parameter n_it zwischen zehn und 400 in Schritten von zehn erhöht. Es ergeben sich die in Bild 6-5 dargestellten mittleren Verfahr- und Rechenzeiten.

Deutlich ist in den Ergebnissen der von Binder et al. beschriebene Einfluss der Iterationsanzahl zu sehen. Zunächst steigt der Mittelwerte der Rechenzeit näherungsweise linear mit der Anzahl an Iterationen an. Ab ca. 200 bis 250 Iterationen wächst der Einfluss der Abbruchkriterien für die minimale Änderung der Optimierungsgrößen $\min(\Delta q_5, \Delta q_6)$ (Portalpositionen) durch den Optimierer sowie die damit erzielte minimale Änderung des Wertes der Gütefunktion $\min(\Delta J_\mathrm{rel})$. Aufgrund dessen wird die maximale Anzahl an Iterationen immer seltener erreicht, der Anstieg der mittleren Optimierungszeit wird mit zunehmendem n_it geringer. Auf der anderen Seite sinkt die erforderliche Verfahrzeit bis ungefähr 100 Iterationen stark ab, danach werden die Änderungen deutlich geringer und ab ca. 200 Iterationen reduziert sich die Änderungsrate noch einmal. Insgesamt wird aus den Ergebnissen die Schlussfolgerung gezogen, dass eine Begrenzung der Iterationen

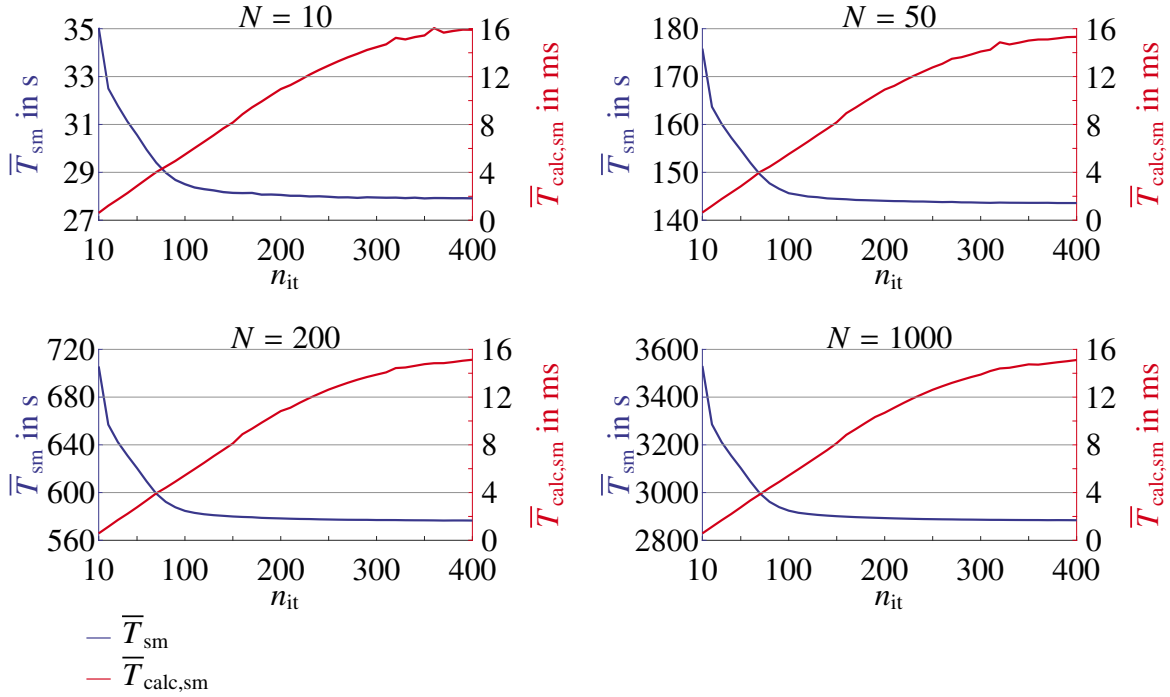


Bild 6-5: *Mittlere Verfahrzeit* $\overline{T}_{sm}$ *sowie mittlere Rechenzeit* $\overline{T}_{calc,sm}$ *bei variierter Iterationbegrenzung und unterschiedlich langen Sollfolgen*

auf $n_{it} = 200$ einen guten Kompromiss aus einer deutlichen Reduzierung der Verfahrzeit und der dafür benötigten Rechenzeit darstellt. In Tabelle 6-9 sind die Ergebnisse des Referenzansatzes sowie der Umsetzung in MATLAB mit *Sbplx* bei $n_{it} = 100000$ und $n_{it} = 200$ gegenüber gestellt. Es ist zu erkennen, dass der Durchschnitt der Rechenzeit über die Folgen der unterschiedlichen Länge N um mindestens 3,9 ms reduziert wird, während die mittlere prozentuale Einsparung der Verfahrzeit gegenüber dem Referenzansatz um maximal 0,47 % sinkt.

Tabelle 6-9: *Ergebnisse des Referenzverfahrens und des modellprädiktiven Ansatzes mit fmincon* ($n_{it} = 100000$) *und Sbplx* ($n_{it} = 100000$ *und* $n_{it} = 200$) *in MATLAB*

N	$\overline{T}_{ref}$	$\overline{T}_{sm}$	$\overline{T}_{sm200}$	$\overline{T}_{calc,sm}$	$\overline{T}_{calc,sm200}$	$\Delta\overline{T}_{ot}$	$\Delta\overline{T}_{sm200}$
10	33,71 s	27,78 s	27,94 s	17,90 ms	12,20 ms	−17,59 %	−17,12 %
50	189,50 s	143,46 s	144,05 s	16,40 ms	12,00 ms	−24,30 %	−23,98 %
100	388,58 s	288,61 s	289,66 s	16,00 ms	12,00 ms	−25,73 %	−25,46 %
200	788,49 s	576,34 s	578,34 s	16,00 ms	12,10 ms	−26,91 %	−26,65 %
1000	4028,76 s	2883,80 s	2893,20 s	15,90 ms	12,00 ms	−28,33 %	−28,19 %

Für das vorliegende Kapitel lässt sich bezogen auf die bisher durchgeführte simulative Untersuchung zusammenfassen, dass die modellprädiktive Planung zeitoptimierter Trajektorien für mehrachsige kinematisch redundante Mechanismen signifikante Einsparungen der Verfahrzeit gegenüber einer festen Bewegungsaufteilung ermöglicht. Dies ist auch

der Fall, wenn auf frei verfügbare Optimierungsbibliotheken und die darin enthaltenen Optimierungsalgorithmen zurückgegriffen wird. Dabei kann durch eine entsprechende Wahl des Prädiktionshorizonts und der maximal zulässigen Anzahl an Iterationen des Optimierers die Rechenzeit reduziert werden, ohne die erzielten Einsparungen der Verfahrzeit entscheidend zu senken. Für den weiteren Verlauf der Arbeit werden diese beiden Parameter basierend auf den durchgeführten simulativen Untersuchungen auf $n_\mathrm{p} = 3$ und $n_\mathrm{it} = 200$ festgelegt.

Eine abschließende Analyse gilt der Online-Fähigkeit des Ansatzes beziehungsweise der Fähigkeit, auf kurzfristige Änderungen der zukünftigen Sollpositionen innerhalb einer Folge zu reagieren. Hierzu wurde eine Folge der Länge $N = 10$ betrachtet, bei der zum Zeitpunkt $k = 4$ die zukünftige Position $\vec{Z}_{\mathrm{TM},5}$ geändert wird ($\vec{Z}_{\mathrm{TM},5} = \vec{Z}_{\mathrm{TM},5'}$). Der Optimierer geht dabei im Schritt $k = 3$ von der ursprünglichen Sollposition $\vec{Z}_{\mathrm{TM},5}$ aus, bei der Verschiebung des betrachteten Bereichs auf $k = 4$ wird mit der neuen Sollposition $\vec{Z}_{\mathrm{TM},5'}$ optimiert. Betrachtet wird die aus der Folge $r10_1$ durch Änderung der fünften Sollposition resultierende Folge $r10_{1'}$. Wird die Änderung „spontan“ durchgeführt, ergibt sich eine resultierende Verfahrzeit $T_{\mathrm{sm200,s}} = 28{,}38\,\mathrm{s}$ (**s**pontan). Ist $\vec{Z}_{\mathrm{TM},5'}$ von Beginn an ein bekannter Bestandteil der Folge, so resultiert die insgesamt erforderliche Verfahrzeit zu $T_{\mathrm{sm200,b}} = 27{,}95\,\mathrm{s}$ (**b**ekannt). Die erhöhte Verfahrzeit für den Fall der spontanen Änderung liegt in der Verwendung eines Prädiktionshorizonts $n_\mathrm{p} > 1$. Bei einer vorab bekannten Sollpositionsfolge sorgt dieser dafür, dass der Mechanismus seine zukünftige Gelenkkonfiguration nicht nur optimiert für die nächste, sondern für die nächsten n_p Sollpositionen berechnet (vergleiche Kapitel 4). Die im aktuellen Schritt nicht genutzten optimierten Achsstellungen für die Schritte $k + 1 \ldots k + n_\mathrm{p}$ dienen im nächsten Schritt als Startwerte der Optimierung. Im Falle einer spontanen Positionsänderung wurden diese allerdings auf Basis nun nicht mehr geltender Werte berechnet. Durch den Einsatz einer exponentiell abfallenden Gewichtung im Rahmen der Gütefunktion wird dieser Einflusses von spontanen Änderungen auf die Verfahrzeit jedoch reduziert. Somit lassen die Ergebnisse den Rückschluss zu, dass der modellprädiktive Ansatz die Anforderung der online-Reaktion auf kurzfristige Änderungen erfüllt.

Aufbauend auf dieser ersten Validierung des Funktionsprinzips wird im folgenden Kapitel nun die Umsetzung des Ansatzes mit dem Optimierer *Sbplx* unter Berücksichtigung der ermittelten Werte für n_p und n_it auf das reale System vorgestellt.

7 Umsetzung am realen System

In diesem Kapitel wird die Umsetzung der vorgestellten modellprädiktiven Planung auf das reale System vorgestellt, wobei zwei unterschiedliche Ansätze realisiert werden. Zunächst wird eine verteilte Ausführung des Ansatzes im Rahmen einer Kombination von SPS und eingebetteter Hardware beschrieben, bevor dann die vollständige Umsetzung auf einer einzelnen SPS inklusive der erzielten Verfahrzeiten und benötigten Rechenzeiten vorgestellt wird.

7.1 Umsetzung der modellprädiktiven Planung auf SPS und eingebetteter Hardware

Vor der vollständigen Implementierung der entwickelten modellprädiktiven Planung auf einer SPS wird zunächst ein verteilter Ansatz umgesetzt. Dieser wurde im Rahmen einer Vorveröffentlichung von Rüting et al. in [RBT17] vorgestellt. Ähnlich wie bei [HWR08] wird die Optimierung von der SPS auf ein zweites System ausgelagert, wodurch die SPS von dieser vergleichsweise rechenintensiven Aufgabe entlastet wird. Vergleichbar zu [MM11] und [MM12] erfolgt die Vernetzung zwischen SPS und Rechensystem über ein Ethernet-basiertes Protokoll. Neben der Entlastung der SPS gestaltet sich zudem die Integration von C-Bibliotheken wie *NLopt* auf eingebetteten Systemen häufig leichter als auf Industriesteuerungen. Konkret wird für die Ausführung der Optimierung sowie des erforderlichen Modells etc. ein Raspberry Pi (RPI) der vierten Generation unter Nutzung von OPC UA an eine SPS, im vorliegenden Fall von der Fa. *Bernecker & Rainer* (B&R), angebunden. Der Einsatz von OPC UA [MLD09] als herstellerübergreifendes Kommunikationsprotokoll ermöglicht neben der im Rahmen der vorliegenden Arbeit genutzten Hardware auch die Vernetzung mit weiteren Steuerungen, wodurch sich dieser Ansatz gut für die Nachrüstung in bestehenden Systemen im Rahmen eines Retrofits eignet. Die Datenblätter von RPI und SPS sind im Anhang der vorliegenden Arbeit (A1-13 und A1-16) zu finden.

Bild 7-1 zeigt den aufgebauten Schaltschrank mit der eingesetzten SPS, dem RPI, Modulen für die vier Motoren des Tricept und zwei Servomotoren des Portals sowie zusätzliche Komponenten wie etwa sicheren Ein- und Ausgängen zur Realisierung eines Not-Halt-Kreises.

In Bild 7-2 ist eine Übersicht des Ansatzes dargestellt. Auf dem eingesetzten RPI wird die in Kapitel 4 vorgestellte modellprädiktive Planung (vgl. Bild 4-5) ausgeführt. Dazu wird *Sbplx* auf Basis der *NLopt*-Bibliothek eingebunden, sowie die IKP der zwei Teilsysteme, das DKP des Portals, das eingesetzte Zeitmodell und die Gütefunktion in *C* implementiert und auf dem RPI kompiliert, sodass ein ausführbares Programm entstand. Zusätzlich wird für die Kommunikation mit der SPS ein OPC-UA-Client (*open62541* [ope19a], [ope19b]) implementiert. Mittels diesem werden zwischen dem RPI und dem OPC-UA-Server als Gegenstelle auf der SPS die erforderlichen Parameter übertragen. Auf der SPS wird zum einen die Verwaltung der Sollposition ausgeführt, die im ersten Schritt aus einer Datenbank stammen. Im Kontext der immer flexibleren Fertigung besteht ebenso die Möglichkeit der

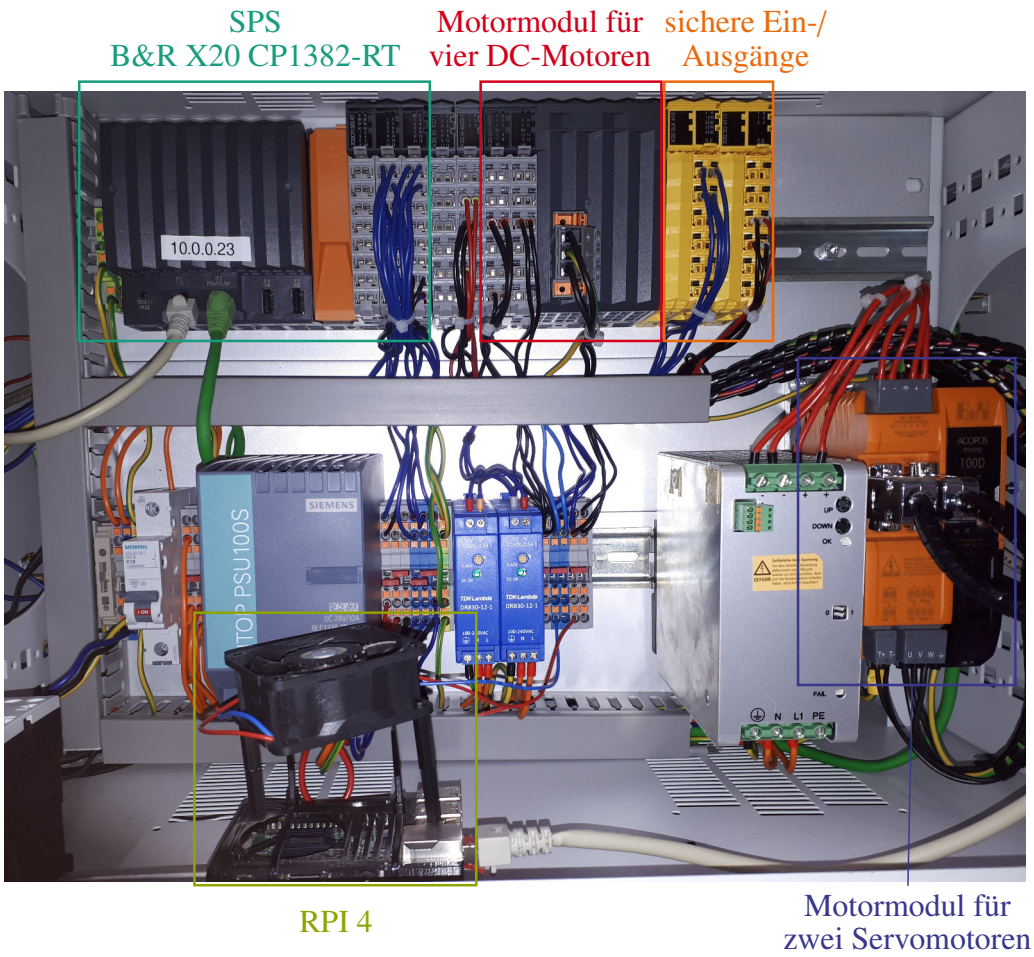


Bild 7-1: *Schaltschrank des Mechanismus mit SPS und RPI*

Einbindung zusätzlicher Komponenten wie eines Radio Frequency Identification (RFID)-Lesegeräts, welches eine gewisse Anzahl an Produkten im Voraus erfasst und abhängig von der vorliegenden Produktintelligenz etwa anhand einer eindeutigen Produktnummer die Bearbeitungspositionen aus einer Datenbank lädt oder diese direkt vom Produkt bezieht [Ber20]. Die für die Optimierung erforderliche aktuelle Position $\vec{Z}_{\mathrm{TM,k}}$ sowie die weiteren Positionen $\vec{Z}_{\mathrm{TM},k+1} \ldots \vec{Z}_{\mathrm{TM},k+\mathrm{n_p}}$ innerhalb des Prädiktionshorizonts werden an den RPI übergeben. Zum anderen erfolgt die Anbindung der Antriebe bzw. der Antriebsregler über die SPS, welche somit die aktuellen Positionen $q_{1,k} \ldots q_{6,k}$ der Antriebe falls erforderlich skaliert und der Optimierung auf dem RPI bereit stellt. Zudem werden aus den mittels der modellprädiktiven Planung ermittelten Sollpositionen $q_{5,k+1}$ und $q_{6,k+1}$ der Portalantriebe und der Sollposition $\vec{Z}_{\mathrm{TM},k+1}$ auf der SPS durch Nutzung der Gleichungen des IKP die Sollpositionen $q_{1,k+1} \ldots q_{4,k+1}$ der Tricept-Antriebe berechnet und an die Positionsregelung der Antriebsregler übergeben. Hier werden die in 4.4 beschriebenen Möglichkeiten moderner Antriebsregler genutzt, um die SPS zusätzlich zu entlasten.

Um auf dem RPI eine möglichst kurze Rechenzeit zu erzielen und zu diesem Zweck den überwiegenden Teil der verfügbaren Rechenleistung für die Optimierung zu nutzen, wurde das Betriebssystem *Raspbian Lite* [Ras20] auf dem RPI installiert, das ohne graphische Benutzeroberfläche auskommt. Um zudem den Nachteil eines Nicht-Echtzeit-Systems hinsichtlich der Schwankungen der Rechenzeit durch den Einfluss anderer Prozesse zu

minimieren, wurde ein Prozessorkern des RPI ausschließlich für die Optimierung reserviert. Sobald am RPI die erforderliche Versorgungsspannung anliegt, werden der OPC-UA-Client sowie das Optimierungsprogramm automatisch initialisiert und gestartet. Von der SPS können nun mittel OPC-UA die erforderlichen Parameter übergeben und über eine zusätzliche Variable der eigentlichen Optimierungsvorgang gestartet werden.

Im Folgenden werden die mit diesem System erzielten Ergebnisse vorgestellt, wobei zum einen die erforderlichen Rechenzeiten und die resultierenden Verfahrzeiten betrachtet werden. Zum anderen gilt es die durch die Kommunikation über OPC UA auftretenden Verzögerungen zu analysieren. Es ist an dieser Stelle festzuhalten, dass die an der SPS von B&R angeschlossenen Antriebe aufgrund der auf die in Abschnitt 5.3 vorgestellten Antriebe angepassten Motoraufnahmen nicht an den realen Mechanismus montiert werden konnten. Aus diesem Grund wurden die Analysen mit Motoren durchgeführt, die nicht an den realen Mechanismus gekoppelt waren. Um möglichst realitätsnahe Verfahrzeiten zu ermitteln, werden die Sollpositionen der einzelnen Antriebe zusätzlich auf der SPS gespeichert und die resultierenden Verfahrzeiten im Anschluss auf Basis des validierten Antriebsmodells (vgl. Abschnitte 4.3 und 5.3) in *MATLAB* berechnet.

In Tabelle 7-1 sind die Verfahrzeiten von Referenzansatz, Optimierung in *MATLAB* und Optimierung auf dem RPI gegenübergestellt (sr: ***S**bplx **R**PI*). Dabei werden auf dem RPI die in Tabelle 7-2 aufgezeigten Parameter für die Optimierung verwendet. Die Initialposition und die Wahl der Startwerte des Optimierers entsprechen denen der simulativen Umsetzung in *MATLAB*. Die Ergebnisse zeigen, dass die mittels des modellprädiktiven Planungsansatzes

Tabelle 7-1: *Ergebnisse des Referenzverfahrens und des modellprädiktiven Ansatzes mit Sbplx in MATLAB und auf dem RPI*

N	$\overline{T}_{\mathrm{ref}}$	$\overline{T}_{\mathrm{sm200}}$	$\overline{T}_{\mathrm{sr}}$	$T_{\mathrm{sr,min}}$	$T_{\mathrm{sr,max}}$	$\Delta\overline{T}_{\mathrm{sm200}}$	$\Delta\overline{T}_{\mathrm{sr}}$
10	33,71 s	27,94 s	27,94 s	14,86 s	46,11 s	−17,12 %	−17,12 %
50	189,50 s	144,05 s	144,03 s	104,03 s	182,33 s	−23,98 %	−24,00 %
100	388,58 s	289,66 s	289,62 s	245,81 s	347,90 s	−25,46 %	−25,47 %
200	788,49 s	578,34 s	578,34 s	515,37 s	657,31 s	−26,65 %	−26,65 %
1000	4028,76 s	2893,20 s	2892,25 s	2712,25 s	3078,28 s	−28,19 %	−28,21 %

und dem Optimierer *Sbplx* auf dem RPI erzielten Einsparungen im gleichen Bereich wie in *MATLAB* liegen. Dass trotz der Nutzung des gleichen Optimierers und der gleichen Sollpositionsfolgen und Algorithmen bei beiden Analysen minimale Unterschiede in den erzielten Gesamtverfahrzeiten (sowohl Durchschnitt über 1000 Sollpositionsfolgen als auch Minimal- und Maximalwerte) auftreten, wird auf die unterschiedlich implementierten Mathematikfunktionen und die Unterschiede in den eingesetzten Datentypen zurückgeführt. So verwendet *MATLAB* standardmäßig Gleitkommazahlen mit 64 Bit, während bei der *C*-Implementierung sogenannte *float* mit 32 Bit eingesetzt werden. Äquivalent zu dem im Rahmen der *MATLAB*-basierten Analyse festgestellten Verhalten ist auch bei dem Ansatz mit SPS und RPI der Trend zu erkennen, dass die prozentuale Einsparung $\Delta\overline{T}_{\mathrm{sr}}$ mit der Länge N der Sollpositionsfolgen ansteigt. Dies lässt sich wie in Abschnitt 6.3 beschrieben begründen.

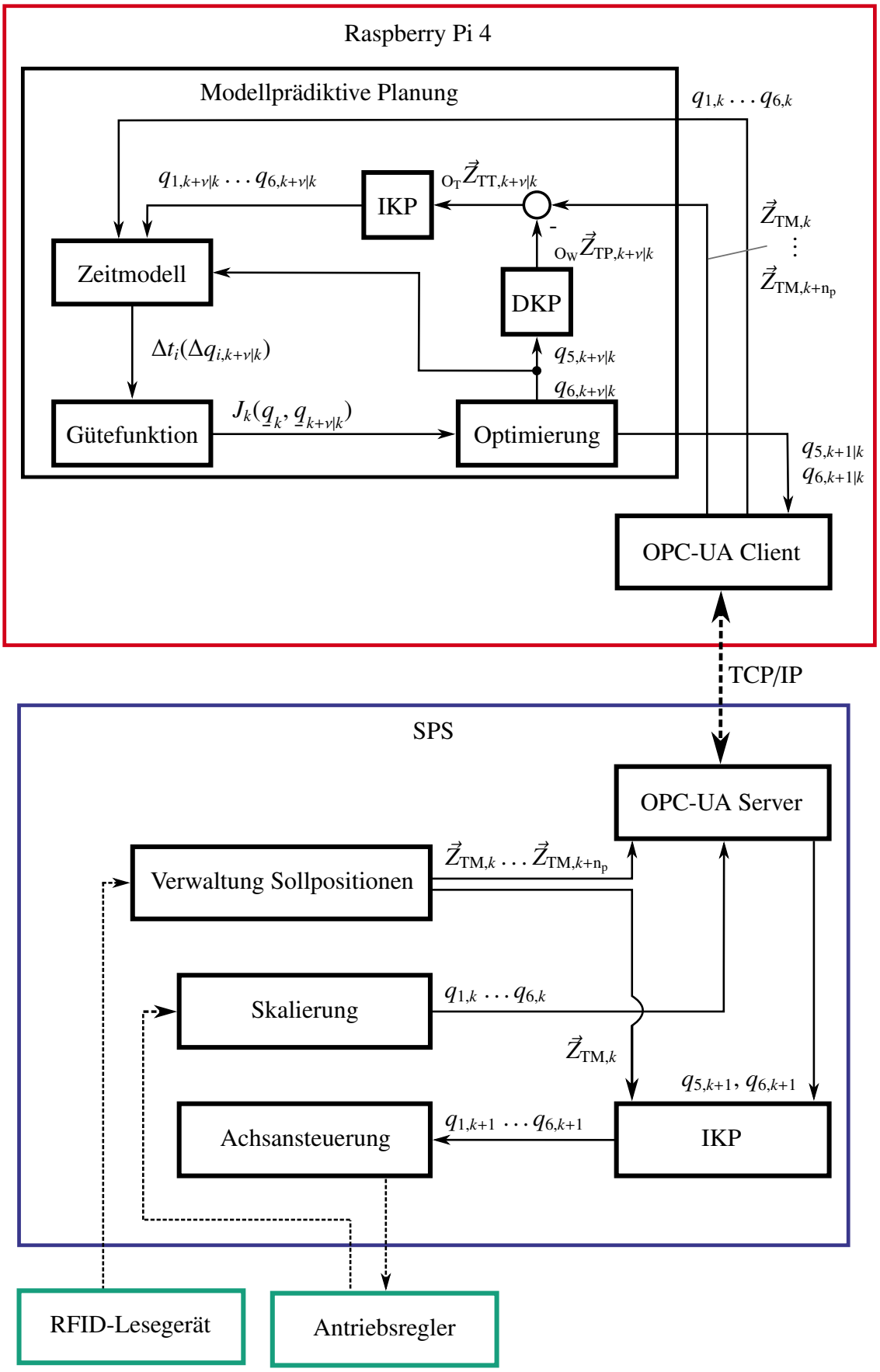


Bild 7-2: Prinzip der modellprädiktiven Planung auf SPS und RPI

Tabelle 7-2: Parameter für Sbplx bei Ausführung auf dem RPI

Parameter	Wert
Optimierer	Sbplx
n_p	3
n_it	200
$\min(\Delta q_5, \Delta q_6)$	1×10^{-5} mm
$\min(\Delta J_\mathrm{rel})$	1×10^{-4} s
w_1	50
w_2	10
w_3	10
$q_{i,\mathrm{os}}$	5 mm

Im Hinblick auf die angestrebte Echtzeitfähigkeit gilt es jedoch nicht nur die Verfahrzeiten, sondern auch die erforderlichen Berechnungszeiten zu analysieren. Hierbei wird unterschieden zwischen der Zeit $\overline{T}_\mathrm{opt,sr}$, die der reine Optimierungsprozess benötigt, und der Zeit $\overline{T}_\mathrm{calc,sr}$. Hierbei handelt es sich um die erforderliche Zeit, um die durch die Sollwertverwaltung bereitgestellten Sollpositionen $\vec{Z}_{\mathrm{TM},k} \ldots \vec{Z}_{\mathrm{TM},k+\mathrm{n_p}}$ mittels OPC-UA von der SPS zu holen, die Optimierung durchzuführen und die Antriebspositionen $q_{5,k+1}$ und $q_{6,k+1}$ als Ergebnisse an die SPS zurückschreiben. Zudem wird hier die Zeit zum Ausführen des IKP auf der SPS sowie die Weitergabe von $q_{1,\mathrm{k+1}} \ldots q_{6,\mathrm{k+1}}$ durch die Achsansteuerung an die Antriebsregler berücksichtigt. Nachfolgend ist der prinzipielle Ablauf als Pseudocode[20] dargestellt, in dem die Messzeitpunkte der Zeiten farblich gekennzeichnet sind:

Initialisierung
for $k = 1 \ldots N \in \mathbb{N}$ **do**
Start der Zeitmessung für $T_{\mathrm{calc,sr},k}$
Lesen der Sollpositionen $\vec{Z}_{\mathrm{TM},k} \ldots \vec{Z}_{\mathrm{TM},k+\mathrm{n_p}}$ auf der SPS
Übertragen der Sollpositionen $\vec{Z}_{\mathrm{TM},k} \ldots \vec{Z}_{\mathrm{TM},k+\mathrm{n_p}}$ und der aktuellen Antriebspositionen $q_{1,k} \ldots q_{6,k}$ mittels OPC-UA an den RPI
Start der Zeitmessung für $T_{\mathrm{opt,sr},k}$
Ausführung der Optimierung
Ende der Zeitmessung für $T_{\mathrm{opt,sr},k}$
Übertragen der optimierten Portalpositionen $q_{5,k+1}$ und $q_{6,k+1}$ mittels OPC-UA
Berechnung der Sollpositionen $q_{1,k+1} \ldots q_{4,k+1}$
Übergabe der Sollpositionen $q_{1,k+1} \ldots q_{6,k+1}$ an Achsansteuerung
Ende der Zeitmessung für $T_{\mathrm{calc,sr},k}$
end for
Abschluss

[20] Pseudocode dient der Beschreibung (komplexer) Algorithmen losgelöst von der konkreten Implementierung in einer Programmiersprache. Zusammen mit der Wahl eines geeigneten Abstraktionsniveaus liegt das Ziel dabei in einer Verbesserung der Lesbarkeit. [SW07], S. 175

Aus den N gemessenen Zeiten für jede der 1000 unterschiedlichen Sollpositionsfolgen rN wird zunächst der Mittelwert

$$\overline{T}_{\mathrm{opt,sr}}(rN_j) = \frac{\sum_{k=1}^{N} T_{\mathrm{opt,sr},k}}{N} \tag{7-1}$$

für die jeweilige Folge rN_j mit $j = 1 \ldots 1000 \in \mathbb{N}$ der Länge N und darauf basierend der Mittelwert

$$\overline{T}_{\mathrm{opt,sr}}(rN) = \frac{\sum_{j=1}^{1000} \overline{T}_{\mathrm{opt,sr}}(rN_j)}{1000} \tag{7-2}$$

über die 1000 unterschiedlichen Folgen mit gleicher Länge berechnet. Die Vorgehensweise für $\overline{T}_{\mathrm{calc,sr}}$ ist äquivalent dazu. Zusätzlich zu den Mittelwerten werden die maximalen Zeiten $T_{\mathrm{optmax,sr}}$ und $T_{\mathrm{calcmax,sr}}$ betrachtet, da hier bereits ein Ausreißer zu einer Verletzung der Echtzeitbedingung führen kann. Für die Ermittlung des Maximalwerts wird die längste Optimierungs- bzw. Rechenzeit für jede der 1000 Folgen gespeichert und hiervon wiederum das Maximum ermittelt. Die Ergebnisse sind in Tabelle 7-3 aufgeführt.

Tabelle 7-3: *Erforderliche Optimierungs- und Rechenzeiten für die modellprädiktive Planung auf dem RPI*

N	$\overline{T}_{\mathrm{opt,sr}}$	$T_{\mathrm{optmax,sr}}$	$\overline{T}_{\mathrm{calc,sr}}$	$T_{\mathrm{calcmax,sr}}$
10	0,785 ms	0,963 ms	13,0 ms	15,1 ms
50	0,775 ms	0,979 ms	13,5 ms	24,0 ms
100	0,774 ms	0,950 ms	13,5 ms	21,2 ms
200	0,773 ms	0,970 ms	13,6 ms	22,2 ms
1000	0,740 ms	0,970 ms	13,5 ms	14,8 ms

Den gemessenen Zeiten ist zu entnehmen, dass das angestrebte Ziel einer Optimierungszeit kleiner 1 ms insbesondere auch bezogen auf die maximal auftretende Optimierungszeit $T_{\mathrm{optmax,sr}}$ erreicht wurde. Die Anzahl an Sollpositionen je Folge hat keinen entscheidenden Einfluss auf die erforderliche Zeit. Anderes sieht es hinsichtlich der Rechenzeit $\overline{T}_{\mathrm{calc,sr}}$ bzw. dem Maximalwert $T_{\mathrm{calcmax,sr}}$ aus. Hier wird unter anderem auch die Zeit für die Kommunikation mit der SPS mittels OPC UA berücksichtigt, welche einen signifikanten Einfluss hat. Es wurde eine gesonderte Analyse durchgeführt, die die Kommunikationszeit der übertragenen Daten betrachtet. Hierbei werden vom RPI 15 zufällig generierte Werte (Datentyp Double) als Array (Vektor) per OPC UA an die SPS übertragen. Auf die gleiche Variable der SPS, auf die der Schreibzugriff erfolgt, wird dann von Seiten des RPI lesend zugegriffen und überprüft, ob die erhaltenen Werte den zuvor generierten Zufallszahlen entsprechen. Ist dies der Fall, wird der Schreib- und Lesevorgang als abgeschlossen und korrekt durchgeführt angesehen. Die Zeitmessung folgt dabei dem im Folgenden als

Pseudocode dargestellten Schema:

```
for i = 1 bis 500 do
    Generierung von 15 Zufallszahlen des Typs Double als Array
    Start der Zeitmessung
    Schreiben der Werte des Arrays auf eine entsprechende Variable der SPS
    Lesen der entsprechenden Variable der SPS
    if Gelesene Werte gleich den vorher geschriebenen Werten then
        Ende der Zeitmessung
    else
        Erneuter Lesezugriff
    end if
end for
```

Die Zeitmessung wurde für 500 unterschiedliche Gruppen von je 15 Zufallszahlen durchgeführt. Dabei waren die SPS und der RPI direkt über ein Ethernet-Kabel verbunden, um eine Beeinflussung der gemessenen Zeit durch die Auslastung der Netzwerks, etwa bei mehreren Teilnehmern an einem Netzwerkverteiler, auszuschließen. Der Mittelwert der gemessenen Zeit ergibt sich zu $\overline{T}_{\mathrm{com}} = 10{,}63\,\mathrm{ms}$ und der Maximalwert zu $\overline{T}_{\mathrm{com,max}} = 15{,}04\,\mathrm{ms}$. Bezogen auf die mittleren Berechnungszeiten $\overline{T}_{\mathrm{calc,sr}}$ wird somit ein großer Anteil davon für die Kommunikation über OPC UA verwendet. Nach Abzug dieser Zeit sowie der mittleren Optimierungszeit $\overline{T}_{\mathrm{opt,sr}}$ benötigt die SPS ca. $1{,}5\,\mathrm{ms} - 2{,}0\,\mathrm{ms}$ für die Sollpositionsverwaltung, die Berechnung des IKP und die Weitergabe der Sollpositionen an die Antriebsregler der Antriebe $M_1 \ldots M_6$. Da die eingesetzte SPS bereits mit ihrer minimal möglichen Zykluszeit von 1,0 ms betrieben wird, ist hier keine weitere Einsparung möglich. Durch den Umstand, dass OPC UA nicht echtzeitfähig und ohne deterministisches Zeitverhalten ist, unterliegt die Kommunikationszeit zudem zusätzlich Schwankungen. Die maximal gemessene Zeit beträgt dadurch das ungefähr anderthalbfache der mittleren Zeit.

Es ist an dieser Stelle somit festzuhalten, dass der eigentliche modellprädiktive Ansatz mit Optimierung echtzeitfähig ausgeführt werden kann, durch die Kommunikation die insgesamt erforderliche Zeit je Zyklus jedoch stark steigt. Als mögliche Lösung kann hier OPC UA over Time Sensitive Networking (OPC UA TSN) genannt werden, ein ähnlich wie OPC UA herstellerübergreifender Kommunikationsstandard, der jedoch explizit für Echtzeitkommunikation entwickelt wurde. Laut der Fa. B&R wurden bei entsprechenden Versuchen Zykluszeiten von 50 µs erreicht [BR19a], insgesamt soll OPC UA TSN im Durchschnitt 18-mal schneller sein als bisher verfügbare industrielle Kommunikationsmöglichkeiten und ca. doppelt so schnell wie das aktuell schnellste Gigabit-Feldbus-Protokoll [BR 19; Bru20]. Die im Rahmen der vorliegenden Dissertation eingesetzten Systeme unterstützen OPC UA TSN jedoch noch nicht, sodass eine Adaption der Kommunikation auf diesen Standard nicht möglich ist und an dieser Stelle nur ein Ausblick gegeben werden kann. Zudem muss die eingesetzte SPS entsprechende Zykluszeiten von $< 1{,}0\,\mathrm{ms}$ ermöglichen. Dies ist mit der aktuell eingesetzten Hardware nicht möglich, durch die Herstellerunabhängigkeit von OPC UA beziehungsweise OPC UA TSN kann hierzu neben leistungsstärkeren Steuerungen des Herstellers B&R aber auch auf andere Steuerungen zurückgegriffen werden.

7.2 Umsetzung der modellprädiktiven Planung auf einer SPS

Nachdem der modellprädiktive Ansatz zunächst simulativ in *MATLAB* und im Anschluss durch Implementierung auf einem eingebetteten System in Kombination mit einer SPS validiert und Einsparungen in der Verfahrzeit von bis zu 28 % erzielt wurden, erfolgt nun die vollständige Umsetzung auf einer SPS der Fa. Beckhoff (Datenblatt siehe Anhang A1-21). Im Rahmen einer Vorveröffentlichung wurde dies von Rüting et al. in [RHT18] und [RHT19] vorgestellt. Während bei Nutzung des RPI als Rechenhardware die angestrebten Zykluszeiten kleiner 1 ms zwar erreicht wurden, die Übertragungszeit zwischen SPS und RPI mittels OPC-UA jedoch einen signifikanten Einfluss hatte und die insgesamt benötigte Berechnungszeit somit größer als die Zielzeit waren, gilt es nun dieses Ziel der Echtzeitfähigkeit unter Betrachtung aller Einflussfaktoren zu erreichen.

In Bild 7-3 ist der eingesetzte Schaltschrank dargestellt.

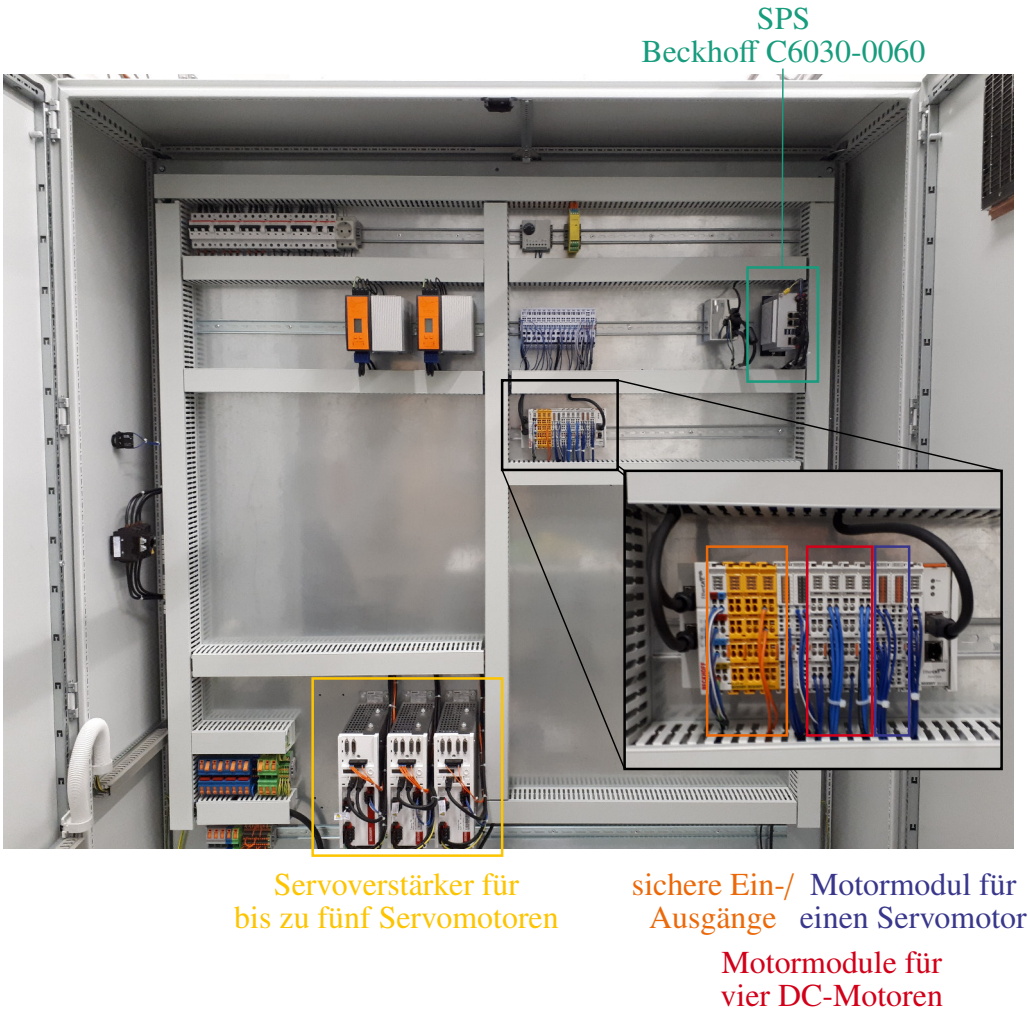


Bild 7-3: *Schaltschrank des Mechanismus mit SPS der Fa. Beckhoff*

Neben der SPS beinhaltet dieser sichere Ein- und Ausgänge zur Realisierung einer Not-Halt-Funktion, zwei Motormodule für die drei DC-Motoren $M_1 \dots M_3$ des Tricept, ein Modul für den DC-Servomotor M_4 und zwei Servoverstärker der AX5000-Baureihe, die unter anderem für die Antriebe M_5 und M_6 des Portals eingesetzt werden.

Die grundlegende Struktur der Implementierung des modellprädiktiven Ansatzes auf der SPS zeigt Bild 7-4.

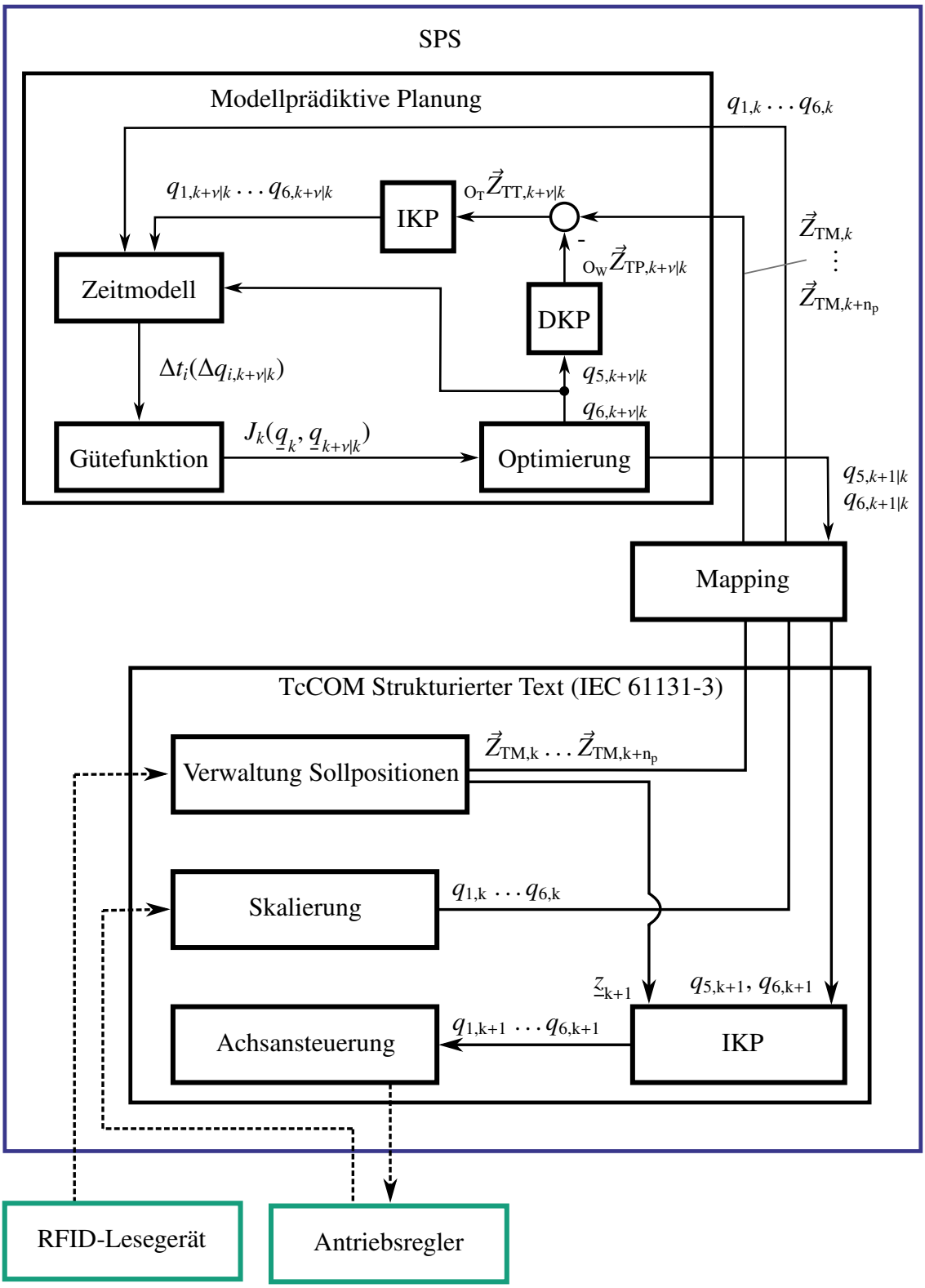


Bild 7-4: *Prinzip der modellprädiktiven Planung auf einer SPS*

Der Optimierer *Sbplx* wurde auf Basis der *NLopt*-Bilbiothek mit der in Abschnitt 4.2 vorgestellten Gütefunktion und dem erforderlichen Zeitmodell sowie den Gleichungen des kinematischen Problems als *C*-Code auf der SPS eingebunden. Trotz der Nutzbarkeit von *C*-Code auf der SPS waren hinsichtlich der *NLopt*-Bibliothek folgende umfangreiche Anpassungen erforderlich, da diese ursprünglich nicht für den echtzeitfähigen Einsatz entwickelt wurde (siehe auch [Abd17]):

- Alle ursprünglich mittels der Header-Datei *math.h* in *C* genutzten mathematischen Funktionen müssen auf die Beckhoff-spezifische Datei *TcMath.h* geändert werden. Hintergrund ist, dass die math.h-Implementierung nicht echtzeitfähig ist. Beispielsweise wird aus dem Befehl `sqrt` für die Quadratwurzel so der Befehl `sqrt_`.
- Alle Befehle für Ausgaben in die Konsole wie etwa `printf` müssen entfernt werden. In *NLopt* dienen sie nur der Ausgabe von zusätzlichen Informationen an den Nutzer, im Rahmen der eigentlichen Optimierung werden sie nicht benötigt.
- Funktionen zur dynamischen Allokation (vom engl. allocate, dt. zuweisen, zuteilen) von Speicher müssen von *C*-eigenen auf Beckhoff-spezifische Befehle geändert werden. So wird etwa aus `malloc` zur Allokation von Speicher der Befehl `TcMemAlloc`.
- Für große Werte werden in der ursprünglichen Implementierung der *NLopt*-Bibliothek die Makros `DBL_MAX`, `HUGE_VAL` und `FLT_MAX` genutzt. Die ersten beiden führen jedoch bei der SPS zu einer Überschreitung der Variablengrenzen, einem sogenannten *overflow*. Um dies zu verhindern, wurden alle Stellen im Quellcode, an denen `DBL_MAX` oder `HUGE_VAL` genutzt werden, auf den Einsatz von `FLT_MAX` umgestellt.
- In der ursprünglichen Implementierung der *NLopt*-Bibliothek werden Funktionen zum Auslesen der aktuellen Systemzeit eingesetzt. Diese stehen bei der SPS nicht zur Verfügung, vielmehr muss anstelle der Funktionsaufrufe auf eine zusätzliche Variable zurückgegriffen werden. Dieser wiederum wird an anderer Stelle von außen in jedem Zyklus die aktuelle Systemzeit übergeben.

Durch die genannten Anpassungen ist es möglich, die *NLopt*-Bibliothek in eine sogenannte statische Bibliothek zu überführen, welche dann auf der SPS verwendet werden kann. Für weitere Informationen etwa hinsichtlich der zu tätigenden Einstellungen auf der SPS wird hier auf das entsprechende Handbuch der Fa. Beckhoff verwiesen [Bec19c]. Die Verwaltung der Sollpositionen, die Skalierung von Messwerten etwa bezüglich der aktuellen Achspositionen und die Achsansteuerung wurde nicht über die Einbindung von *C*-Code sondern als *strukturierter Text* (ST) implementiert.

Zunächst ermittelt der Optimierer auf Basis der aktuellen TCP-Position $\vec{Z}_{\mathrm{TM},k}$, den Achspositionen $q_{1,k} \dots q_{6,k}$ und den von der Sollpositionsverwaltung bereitgestellten Sollpositionen $\vec{Z}_{\mathrm{TM},k+1} \dots \vec{Z}_{\mathrm{TM},k+\mathrm{n_p}}$ innerhalb des Prädiktionshorizonts n_p die optimierten Positionen $q_{5,k+1}$ und $q_{6,k+1}$ der Portalachsen. Mittels des zusätzlich in ST implementierten IKP werden hieraus die Achspositionen $q_{1,k+1} \dots q_{4,k+1}$ berechnet. Zusammen werden $q_{1,k+1} \dots q_{6,k+1}$ für die Ansteuerung der Achsen verwendet, wobei wie in Abschnitt 4.4 beschrieben auf der SPS mittels des NC-Profilgenerators ein Sollpositionsprofil basierend auf vorgegebenen Werten für Ruck, Beschleunigung/Verzögerung und Geschwindigkeit generiert wird. Die Positionsregelung der einzelnen Antriebe $M_1 \dots M_6$ nebst der unterlagerten Regelungen ist auf die Antriebshardware ausgelagert. Die von hier an die SPS zurückgegebenen Ist-Positionen

der einzelnen Antriebe werden vor der Weitergabe an die Optimierung zunächst, soweit erforderlich, skaliert. Zudem können Änderungen der zukünftigen Sollpositionen, die etwa mittels eines RFID- oder Barcode-Lesegeräts direkt vom Produkt gemeldet werden, im Rahmen der Verwaltung der Sollpositionen berücksichtigt werden.

Wie bereits bei der Umsetzung des Ansatzes auf der Kombination aus RPI und SPS erfolgt auch für die vollständige Umsetzung auf der SPS die Analyse der Ergebnisse zum einen hinsichtlich der erzielten Einsparungen der Verfahrzeit und zum anderen bezogen auf die erforderliche Rechenzeit. Analog zum Abschnitt 7.1 wird unterschieden zwischen $\overline{T}_{\mathrm{calc,sb}}$ (sb: ***S**bplx* **B**eckhoff) und $\overline{T}_{\mathrm{opt,sb}}$, zusätzlich werden zudem die maximalen Zeiten betrachtet. Die Messung der Zeiten verdeutlicht der folgende Pseudocode:

Initialisierung
for $k = 1 \ldots N \in \mathbb{N}$ **do**
Start der Zeitmessung für $T_{\mathrm{calc,sb},k}$
Lesen der Sollpositionen $\vec{Z}_{\mathrm{TM},k} \ldots \vec{Z}_{\mathrm{TM},k+\mathrm{n_p}}$
Start der Zeitmessung für $T_{\mathrm{opt,sb},k}$
Ausführung der Optimierung
Ende der Zeitmessung für $T_{\mathrm{opt,sb},k}$
Berechnung der Sollpositionen $q_{1,k+1} \ldots q_{4,k+1}$
Übergabe der Sollpositionen $q_{1,k+1} \ldots q_{6,k+1}$, Ausführung des NC-Profilgenerator
Ende der Zeitmessung für $T_{\mathrm{calc,sb},k}$
end for
Abschluss

Um die vorab im Rahmen der Simulation in *MATLAB* getroffenen Aussagen hinsichtlich des zu nutzenden Prädiktionshorizonts n_{p} und der Anzahl n_{it} an maximal zugelassenen Iterationen zu validieren, werden zunächst die gleichen Analysen wie in *MATLAB* auch auf der SPS ausgeführt. In Bild 7-5 sind die Verfahr- und die Rechenzeit bei Variation des Prädiktionshorizonts von $n_{\mathrm{p}} = 1 \ldots 5$ bei $n_{\mathrm{it}} = 100000$ dargestellt. Es ist zu erkennen, dass auch auf der SPS ein Prädiktionhorizont von $n_{\mathrm{p}} = 3$ einen guten Kompromiss zwischen der Reduzierung der Verfahrzeit und der erforderlichen Rechenzeit darstellt.

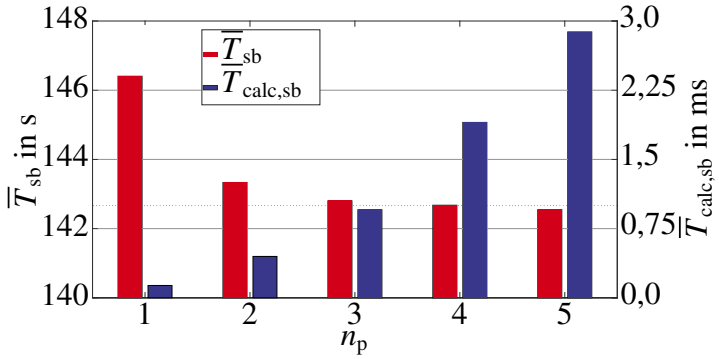


Bild 7-5: *Mittlere Verfahrzeit* $\overline{T}_{\mathrm{sb}}$ *und Rechenzeit* $\overline{T}_{\mathrm{calc,sb}}$ *der Umsetzung auf der SPS für* $N = 50$ *bei unterschiedlichen Prädiktionshorizonten* n_{p} *und* $n_{\mathrm{it}} = 100000$

Die Analyse des Einflusses der maximal zulässigen Anzahl an Iterationen n_{it} bei $n_{\mathrm{p}} = 3$ liefert die in Bild 7-6 dargestellten Ergebnisse. Gegenüber *MATLAB* wird hier die Optimierungszeit $\overline{T}_{\mathrm{opt,sb}}$ anstelle der insgesamt benötigten Rechenzeit betrachtet, da die Rechenzeit

jedoch nur zusätzlich die Zeiten etwa für den Zugriff auf die Sollpositionen beinhaltet, stellt dies nur einen Offset der Zeiten dar. Dafür wird hier zusätzlich der Maximalwert $\overline{T}_{\text{optmax,sb}}$ der Optimierungszeit betrachtet, da aufgrund der harten Echtzeitanforderungen auch kein einziger Rechenschritt länger als die geforderte eine Millisekunde benötigen darf. Wie bereits bei der simulativen Umsetzung in *MATLAB* steigen sowohl der Mittelwert als auch der Maximalwert der Optimierungszeit zunächst näherungsweise linear mit der Anzahl an Iterationen an. Ab ca. 200 Iterationen wächst der Einfluss der Abbruchkriterien für die minimale Änderung der Optimierungsgrößen $\min(\Delta q_5, \Delta q_6)$ durch den Optimierer sowie die damit erzielte minimale Änderung des Wertes der Gütefunktion $\min(\Delta J_{\text{rel}})$. Dadurch wird die maximale Anzahl an Iterationen immer seltener erreicht, der Anstieg der mittleren Optimierungszeit wird mit zunehmendem n_{it} geringer. Da jedoch abhängig von der jeweiligen Sollpositionsfolge die Iterationsgrenze trotzdem erreicht werden kann beziehungsweise erreicht wird, steigt die maximale Zeit weiter näherungsweise linear an. Als Schlussfolgerung aus den Ergebnissen wird eine Begrenzung der Iterationen auf $n_{\text{it}} = 200$ weiterhin als zielführend angesehen, da hier die maximale Zeit deutlich unterhalb der geforderten 1 ms liegt und zudem die Verfahrzeit mit weiter steigender Iterationsanzahl immer weniger stark sinkt. Somit sind die auf Basis der Simulation in *MATLAB* getroffenen Annahmen hinsichtlich der Parameter der Optimierung auch für die Umsetzung am realen System zutreffend. Sie werden somit bei der weiteren Analyse der Optimierungs-, Rechen- und Verfahrzeit genutzt.

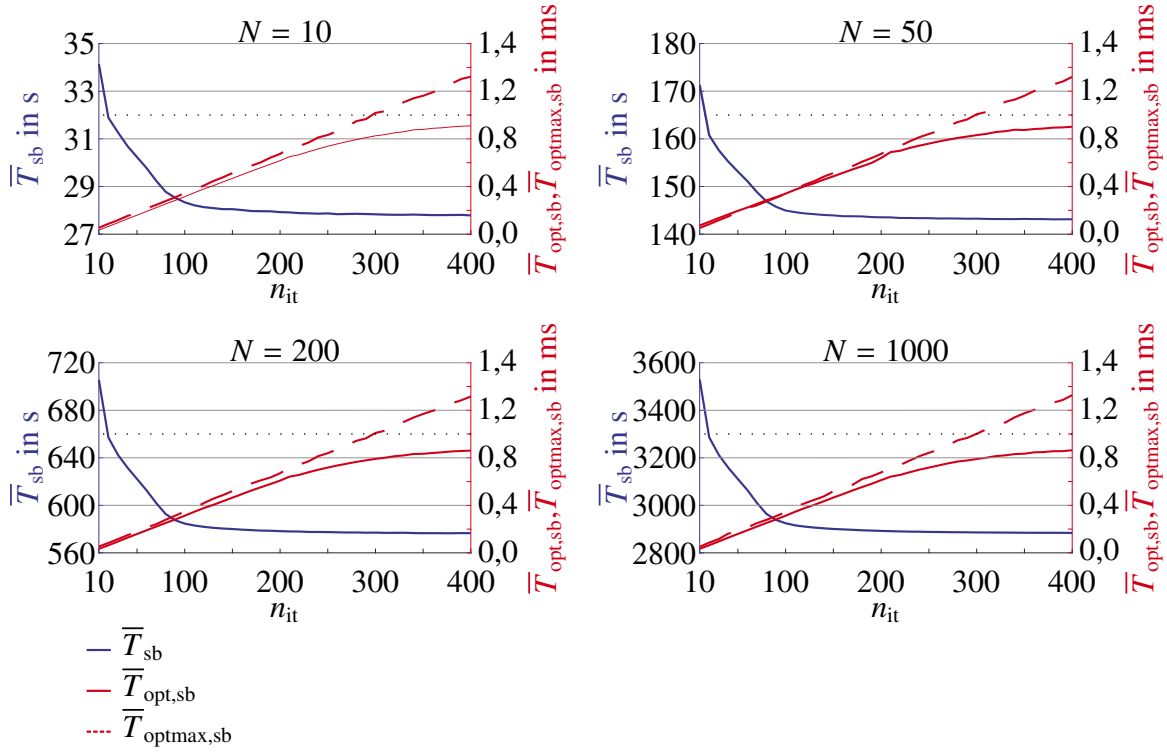


Bild 7-6: *Mittlere Verfahrzeit $\overline{T}_{\text{sb}}$ sowie mittlere und maximale Optimierungszeit $\overline{T}_{\text{opt,sb}}$ und $\overline{T}_{\text{optmax,sb}}$ der SPS bei variierter Iterationbegrenzung und unterschiedlich langen Sollfolgen*

In Tabelle 7-4 sind zunächst die Verfahrzeiten $\overline{T}_{\text{sb}}$ und die dabei erzielten Einsparungen $\Delta\overline{T}_{\text{sb}}$ bei vollständiger Umsetzung auf der SPS den Zeiten des Referenzansatzes sowie

den Ergebnissen aus *MATLAB* bei $n_{\mathrm{it}} = 200$ Iterationen (Index *sm*200) gegenübergestellt, wobei die gleichen Einstellungen für den Optimierer genutzt wurden wie beim RPI (siehe Tabelle 7-2). Die erzielten Ergebnisse liegen dabei sehr nah an den Werten aus der Simu-

Tabelle 7-4: *Ergebnisse des Referenzverfahrens und des modellprädiktiven Ansatzes mit Sbplx auf der SPS*

N	$\overline{T}_{\mathrm{ref}}$	$\overline{T}_{\mathrm{sm200}}$	$\overline{T}_{\mathrm{sb}}$	$T_{\mathrm{sb,min}}$	$T_{\mathrm{sb,max}}$	$\Delta\overline{T}_{\mathrm{sm200}}$	$\Delta\overline{T}_{\mathrm{sb}}$
10	33,71 s	27,94 s	27,93 s	14,86 s	46,12 s	−17,12 %	−17,15 %
50	189,50 s	144,05 s	144,05 s	103,99 s	180,35 s	−23,98 %	−23,98 %
100	388,58 s	289,66 s	289,55 s	249,38 s	351,48 s	−25,46 %	−25,49 %
200	788,49 s	578,34 s	578,26 s	516,61 s	659,81 s	−26,65 %	−26,66 %
1000	4028,76 s	2893,20 s	2892,24 s	2718,27 s	3080,00 s	−28,19 %	−28,21 %

lation in *MATLAB* sowie der Umsetzung auf dem RPI. Auch hier sind die existierenden Abweichungen auf eine abweichende Implementierung der eingesetzten Mathematikfunktionen zurückzuführen. Da die Zeiteinsparung somit weiterhin gegeben ist, gilt es nun die bei der Berechnung der vorgestellten Ergebnisse erforderlichen Rechen- und Optimierungszeiten hinsichtlich des entscheidenden Kriteriums der Echtzeitfähigkeit zu betrachten. Sie sind in Tabelle 7-5 zusammengefasst. Auf Basis der Daten in der Tabelle lässt sich

Tabelle 7-5: *Erforderliche Optimierungs- und Rechenzeiten für die modellprädiktive Planung auf der SPS bei* $n_{\mathrm{it}} = 200$

N	$\overline{T}_{\mathrm{opt,sb}}$	$T_{\mathrm{optmax,sb}}$	$\overline{T}_{\mathrm{calc,sb}}$	$T_{\mathrm{calcmax,sb}}$
10	0,616 ms	0,671 ms	0,641 ms	0,689 ms
50	0,609 ms	0,675 ms	0,635 ms	0,695 ms
100	0,608 ms	0,670 ms	0,636 ms	0,658 ms
200	0,607 ms	0,674 ms	0,632 ms	0,661 ms
1000	0,607 ms	0,660 ms	0,633 ms	0,652 ms

zunächst die entscheidende Aussage treffen, dass unter Berücksichtigung einer maximalen Rechenzeit von $T_{\mathrm{calcmax,sb}} = 0{,}675\,\mathrm{ms}$ das Ziel der Echtzeitfähigkeit mit der Bedingung einer Zykluszeit kleiner 1,0 ms durch den vorgestellten Ansatz auf der SPS vollständig erfüllt wird. Dass sowohl die mittlere Optimierungs- als auch die mittlere Rechenzeit $\overline{T}_{\mathrm{opt,sb}}$ und $\overline{T}_{\mathrm{calc,sb}}$ mit steigender Länge N der Sollpositionsfolgen leicht sinkt, ist wie folgt zu erklären: Für den ersten Durchlauf der Optimierung mit $k = 1$ liegen für den Optimierer fest vorgegebene Startwerte vor (Bewegung in x- und y-Richtung erfolgt initial vollständig durch das Portal), in allen weiteren Schritten ($1 < k \leq N$) werden die Positionen $q_{5,k+\nu}$ und $q_{6,k+\nu}$ mit $\nu = 2 \ldots n_{\mathrm{p}}$ aus dem vorhergegangenen Schritt ($k-1$) als Startwerte eingesetzt. Da diese bereits aus einem Optimierungsdurchlauf stammen, werden sie tendenziell näher an dem optimalen Wert liegen als vorab „willkürlich“ gewählte Startwerte. Somit benötigt

der Optimierer bei $k = 1$ aufgrund der „ungünstigen“ Startwerte teilweise mehr Iterationen als für die darauffolgenden Schritte. Bei kurzen Sollpositionsfolgen, etwa mit $N = 10$, hat dieser Effekt bei der Mittelung (vgl. Gleichung (7-1)) über die Länge der Folge einen größeren Einfluss als bei Folgen mit vielen Sollpositionen. Bei den Maximalwerten sowohl für die Optimierungs- als auch die Rechenzeit sind zwar Schwankungen aber kein eindeutiger Trend zu erkennen.

Es ist an dieser Stelle abschließend festzuhalten, dass das Ziel der Entwicklung einer echtzeitfähigen modellprädiktiven Planung zeitoptimierter Trajektorien für kinematisch redundante Mechanismen und die Umsetzung auf industrielle Rechenhardware mit Zykluszeiten kleiner 1 ms erreicht wurde.

8 Anwendungsszenario 2: HKM bestehend aus Raumportal und Sechsachs-Knickarmroboter

Um die Flexibilität der entwickelten echtzeitfähigen modellprädiktiven Planung zu zeigen, wird in diesem Kapitel die Adaption vom Mechanismus des Fraunhofer IEM auf einen weiteren mehrachsigen kinematisch redundanten Mechanismus vorgestellt. Auch bei diesem handelt es sich nach der in Kapitel 2.2 vorgestellten Definition aus [Sch16] um einen HKM, der durch die serielle Kopplung zweier SKM gebildet wird. Der Mechanismus, der in Bild 8-1 auf Basis eines CAD-Modells dargestellt ist, besteht aus einem dreiachsigen Raumportal, an dessen TCP TP ein sechsachsiger Knickarmroboter *UR10* der Firma *Universal Robots* montiert ist. Der TCP TM des resultierenden Gesamtmechanismus ist deckungsgleich mit dem TCP TUR des UR10.

Mittels des Raumportals können Sollpositionen in den drei kartesischen Raumrichtungen angefahren werden. Der UR10 kann sowohl die Position in x-, y- und z-Richtung als auch die Orientierung des TCP verändern. Somit lässt sich die Position des TCP durch gemeinsame Bewegungen von UR10 und Portal beeinflussen, die Orientierung übernimmt der UR10. Der resultierende Mechanismus ist kinematisch redundant. Es gilt jedoch die Einschränkungen zu beachten, dass der UR10 nicht alle Positionen mit beliebiger Orientierung erreichen kann. Zudem weist er durch seine Gelenkanordnung, die auch als Winkelhand bezeichnet wird (vgl. Abschnitt 2), einen Bereich ober- und unterhalb (positive und negative z-Richtung bezogen auf das Roboter-Basiskoordinatensystem mit dem Ursprung O_{UR}) der Basis auf, der mit dem Roboter-TCP nicht erreicht werden kann. Der Arbeitsraum des UR10 ist in Bild 8-2 dargestellt.

Ein erster Ansatz einer modellbasierten Mehrzieloptimierung zur Reduzierung der Verfahrzeiten des kinematisch redundanten Mechanismus wurde bereits in der studentischen Arbeit [Ker18] vorgestellt und findet hier Berücksichtigung. Im Folgenden wird zunächst die Adaption des Ansatzes auf den Mechanismus vorgestellt, bevor näher auf die Herleitung insbesondere des IKP des UR10 eingegangen wird.

8.1 Adaption des Ansatzes

Entsprechend des modellprädiktiven Ansatzes und wie bereits bei der Adaption auf den IEM-Mechanismus wird der nun vorliegende Mechanismus zunächst in zwei Teilmechanismen unterteilt. Der UR10 stellt den nicht-redundanten und das Raumportal den zur kinematischen Redundanz führenden Teil dar. Analog zu Gleichung (5-66) kann dazu die Pose des Mechanismus

$$ {}_{\mathrm{O_W}}\vec{\xi}_{\mathrm{TM}} = {}_{\mathrm{O_{UR}}}\vec{\xi}_{\mathrm{TUR}} + {}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TP}} = \left[{}_{\mathrm{O_W}}Z_{\mathrm{TM,x}}\ {}_{\mathrm{O_W}}Z_{\mathrm{TM,y}}\ {}_{\mathrm{O_W}}Z_{\mathrm{TM,z}}\ {}_{\mathrm{O_W}}R_{\mathrm{TM,r}}\ {}_{\mathrm{O_W}}R_{\mathrm{TM,p}}\ {}_{\mathrm{O_W}}R_{\mathrm{TM,y}} \right]^T \in \mathbb{R}^6 \tag{8-1} $$

aufgeteilt werden in die Pose ${}_{\mathrm{O_{UR}}}\vec{\xi}_{\mathrm{TUR}}$ des UR10 bezogen auf sein Basiskoordinatensystem mit Ursprung O_{UR} und den Positionsvektor ${}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TP}}$ des Raumportals bezogen auf das

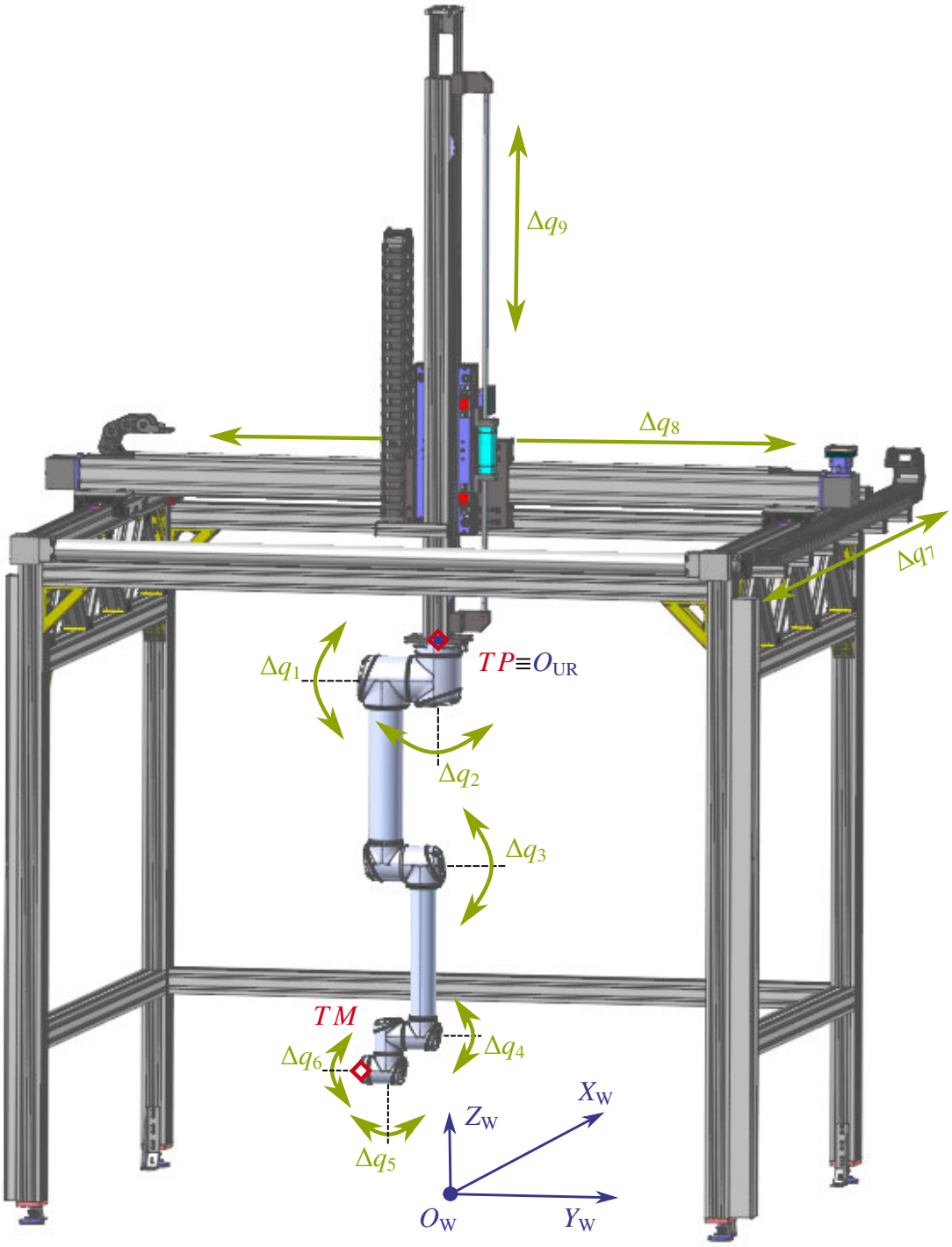


Bild 8-1: *Der hybridkinematische Mechanismus bestehend aus Raumportal und Sechsachs-Knickarmroboter UR10*

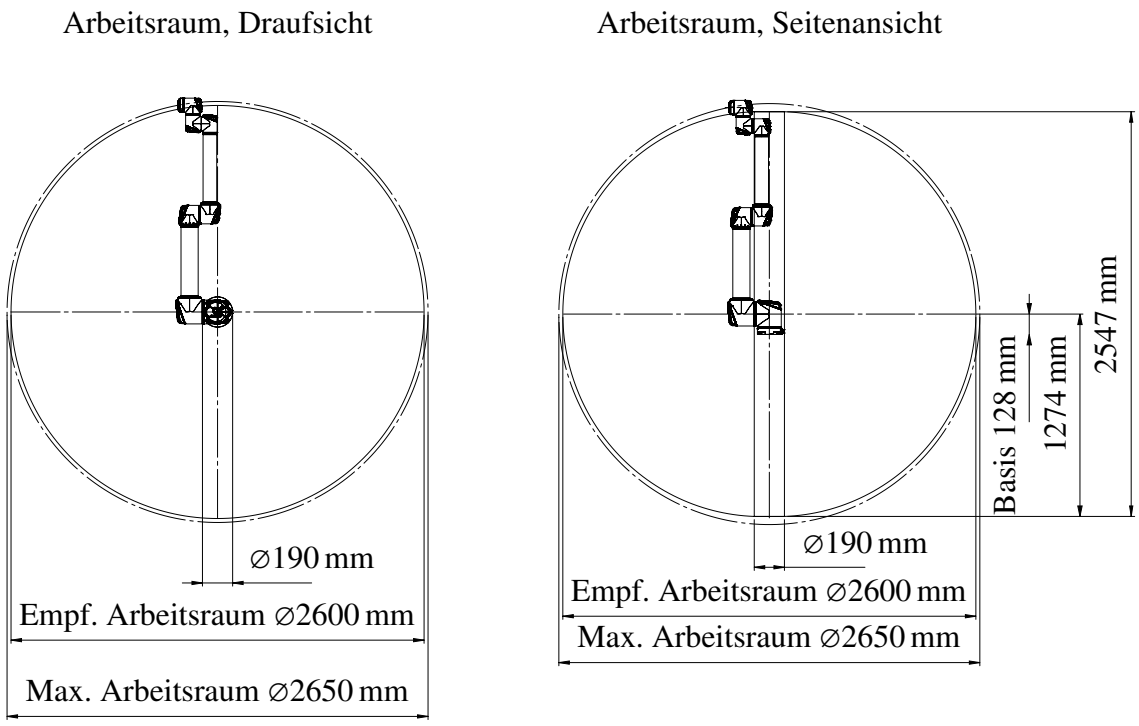


Bild 8-2: *Arbeitsraum des UR10 nach [Uni16]*

Weltkoordinatensystem mit Ursprung O_{W}. Der Ursprung O_{UR} des Basiskoordintensystems des UR10 ist dabei im TCP TP des Portals verortet. Das Portal kann nur die Position des TCP (${}_{\mathrm{O_W}}Z_{\mathrm{TM,x}}$, ${}_{\mathrm{O_W}}Z_{\mathrm{TM,y}}$, ${}_{\mathrm{O_W}}Z_{\mathrm{TM,z}}$) variieren, wohingegen durch den Roboter auch die Orientierung (${}_{\mathrm{O_W}}R_{\mathrm{TM,r}}$, ${}_{\mathrm{O_W}}R_{\mathrm{TM,p}}$, ${}_{\mathrm{O_W}}R_{\mathrm{TM,y}}$) verändert werden kann. Die Positionen der insgesamt neun Achsen werden im Vektor

$$\underline{q} = [q_1\ q_2\ q_3\ q_4\ q_5\ q_6\ q_7\ q_8\ q_9]^T \in \mathbb{R}^9 \tag{8-2}$$

zusammengefasst. Dabei stellen $q_1 \dots q_6$ die Achsen des UR10 und $q_7 \dots q_9$ die Achsen des Raumportals dar.

Im Rahmen der Adaption des Ansatzes auf den dargestellten neun-achsigen Mechanismus wird auf das in Gleichung (4-11) dargestellte Optimierungsproblem zurückgegriffen. Aufgrund der drei Achsen $q_7 \dots q_9$ des Raumportals wird diese zu

$$\min_{q_{7,k+\nu|k}, q_{8,k+\nu|k}, q_{9,k+\nu|k}} J_k(\underline{q}_k, \underline{q}_{k+\nu|k}, {}^{O_{\mathrm{UR}},6}\mathbf{T}_{k+\nu|k}) \tag{8-3}$$

erweitert. Durch die zusätzliche z-Achse des Raumportals steigt gegenüber dem ersten Anwendungsszenario die Anzahl der Optimierungsvariablen je Schritt von zwei auf drei. Die Gütefunktion

$$\begin{aligned} J_k(\underline{q}_k, \underline{q}_{k+\nu|k}, {}^{O_{\mathrm{UR}},6}\mathbf{T}_{k+\nu|k}) = &\sum_{\nu=1}^{n_{\mathrm{p}}} w_1 \exp(-(\nu-1)) \max_{i=1\dots m} \Delta t_i(\Delta q_{i,k+\nu|k}) \\ &+ w_2 p_{\mathrm{qmax}}(\underline{q}_{k+\nu|k}) \\ &+ w_3 p_{\mathrm{qmin}}(\underline{q}_{k+\nu|k}) \\ &+ w_4 p_{\mathrm{UR}}({}^{O_{\mathrm{UR}},6}\mathbf{T}_{k+\nu|k}) \end{aligned} \tag{8-4}$$

orientiert sich an der Gleichung (4-5). Die gewichteten Strafterme $w_2 p_{\mathrm{qmax}}(\underline{q}_{k+\nu|k})$ und $w_3 p_{\mathrm{qmin}}(\underline{q}_{k+\nu|k})$ werden jedoch nur für die drei Achsen des Portals genutzt. Für die Grenzen des Roboters wird der zusätzliche über w_4 gewichtete Strafterm $p_{\mathrm{UR}}({}^{O_{\mathrm{UR}},6}\mathbf{T}_{k+\nu|k})$ eingeführt. Dieser berücksichtigt mittels der Gleichungen des IKP und der Transformationsmatrix ${}^{O_{\mathrm{UR}},6}\mathbf{T}_{k+\nu|k}$ zwischen der Roboterbasis und dem Roboter-TCP (vgl. Abschnitt 8.2.1) allgemein die Erreichbarkeit einer Pose durch den Roboter. Aus Gründen des Umfangs dieser Arbeit wird hierbei für weitere Informationen etwa auf [Haw13] verwiesen.

Die Herleitung der erforderlichen Gleichungen für das IKP und das DKP wird im folgenden Abschnitt vorgestellt.

8.2 Herleitung der Gleichungen der kinematischen Probleme

Zunächst wird in diesem Abschnitt das IKP des UR10 betrachtet, im Anschluss daran werden das IKP und das DKP des Portals vorgestellt.

8.2.1 Inverses kinematisches Problem des Knickarmroboters

Die Pose

$$ {}_{\mathrm{O_{UR}}}\vec{\xi}_{\mathrm{TUR}} = \left[{}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,x}}\ {}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,y}}\ {}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,z}}\ {}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,r}}\ {}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,p}}\ {}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,y}} \right]^T \in \mathbb{R}^6 \quad (8\text{-}5) $$

des UR10 setzt sich zusammen aus dem Ortsvektor

$$ {}_{\mathrm{O_{UR}}}\vec{Z}_{\mathrm{TUR}} = \left[{}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,x}}\ {}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,y}}\ {}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,z}} \right]^T = \left[Z_{\mathrm{TURx}}\ Z_{\mathrm{TURy}}\ Z_{\mathrm{TURz}} \right]^T \in \mathbb{R}^3 \quad (8\text{-}6) $$

als Position sowie dem Vektor

$$ {}_{\mathrm{O_{UR}}}\vec{R}_{\mathrm{TUR}} = \left[{}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,r}}\ {}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,p}}\ {}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,y}} \right]^T = \left[R_{\mathrm{r}}\ R_{\mathrm{p}}\ R_{\mathrm{y}} \right]^T \in \mathbb{R}^3 \quad (8\text{-}7) $$

des TCP als Angabe der Orientierung. Letzterer besteht aus den sogenannten Kardan- oder auch *Roll-Pitch-Yaw*-Winkeln (laut [Cor17] auch als *Roll-Pitch-Yaw-Sequenz* bezeichnet). Für (Knickarm-)Roboter hat sich dabei laut [Cor17] die *xyz*-Konvention durchgesetzt, also die Winkelreihenfolge Roll-Pitch-Yaw[21]. *Roll* (R_{r}) beschreibt die Rotation um die x-Achse, *Pitch* (R_{p}) die Rotation um die resultierende y-Achse und *Yaw* (R_{y}) die Rotation um die wiederum resultierende z-Achse.

Bei der Lösung des IKP des UR10 wird auf die sogenannte DHK (vgl. 2.1) zurückgegriffen. Für den UR10 werden die erforderlichen DHP bezogen auf die Grundstellung des Roboters direkt von *Universal Robots* angegeben [Uni15], sie sind in Tabelle 8-1 aufgeführt. Bild 8-3 zeigt die Koordinatensysteme nach [Haw13] ebenfalls bezogen auf die Nullstellung des UR10 ($q_1 \ldots q_6 = 0°$), welche in Bild 8-4 am realen Roboter aus dem *Robotics Lab* [HR20] des Fraunhofer IEM dargestellt ist. Die DHP werden für die Beschreibung der Lage zweier

[21] Bei Fahrzeugen, Flugzeugen, Schiffen oder mobilen Robotern werden die Winkel in der Regel in der Reihenfolge Yaw-Pitch-Roll angegeben [Cor17].

Tabelle 8-1: *DHP des UR10 nach [Uni15]*

Gelenk	q_i	d_i	a_i	α_i
1	0,00°	0,1273 m	0,00 m	90,00°
2	0,00°	0,00 m	−0,612 m	0,00°
3	0,00°	0,00 m	−0,5723 m	0,00°
4	0,00°	0,163 941 m	0,00 m	90°
5	0,00°	0,1157 m	0,00 m	−90,00°
6	0,00°	0,0922 m	0,00 m	0,00°

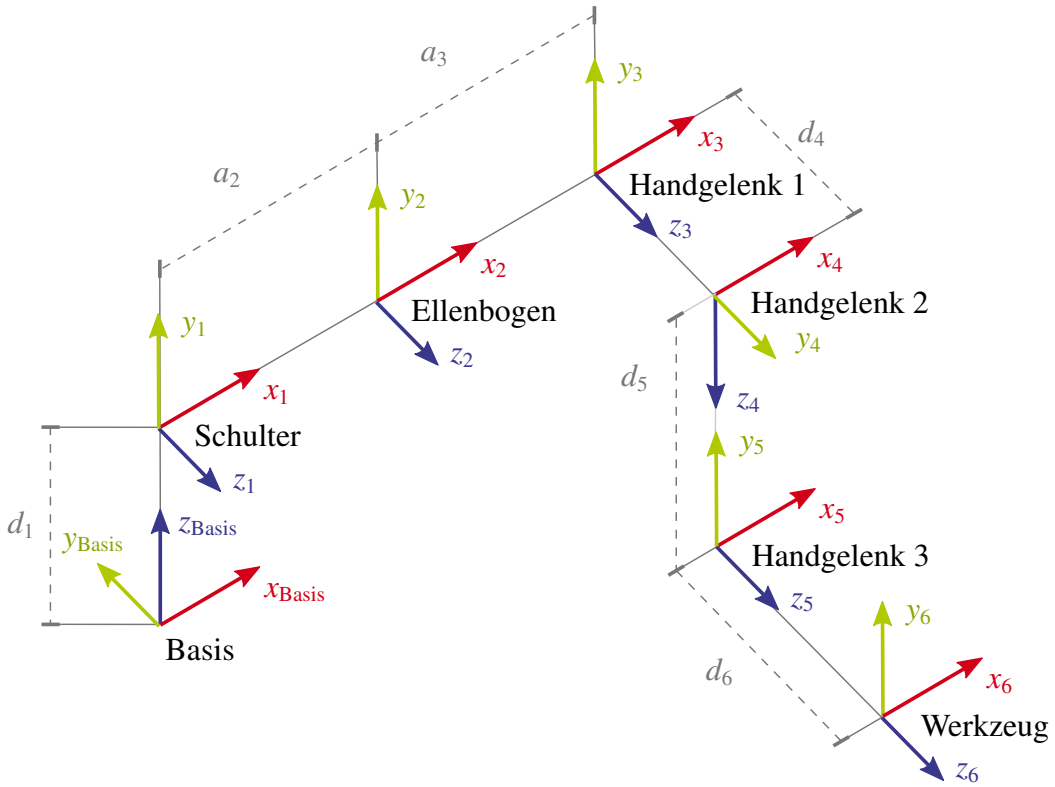


Bild 8-3: *Koordinatensysteme des UR10 in der Grundstellung nach [Haw13]*

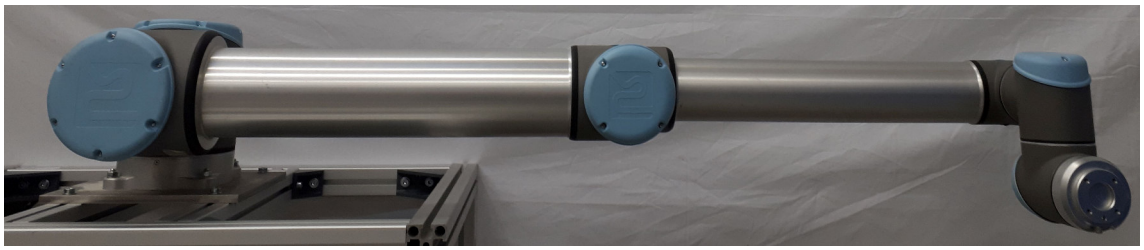


Bild 8-4: *Sechsachs-Knickarmroboter UR10 in seiner Grundstellung im Robotics Lab des Fraunhofer IEM*

benachbarter Koordinatensysteme S_{i-1} und S_i im Rahmen homogener Matrizen genutzt, wobei die Matrix

$$ {}^{i-1,i}\mathbf{T} = \begin{bmatrix} \cos(q_i) & -\sin(q_i)\cos(\alpha_i) & \sin(q_i)\sin(\alpha_i) & a_i\cos(q_i) \\ \sin(q_i) & \cos(q_i)\cos(\alpha_i) & -\cos(q_i)\sin(\alpha_i) & a_i\sin(q_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{8-8} $$

auch als *Denavit-Hartenberg*-Matrix bezeichnet wird [Web09]. Für weitere Informationen zu der Wahl der Gelenkkoordinatensysteme sowie der ausführlichen Herleitung der Gleichungen zur Ermittlung der Roboter-Gelenkpositionen $q_{\mathrm{r},i}$ mit $i = 1 \ldots 6$ für die sechs Antriebe $M_{\mathrm{r},1} \ldots M_{\mathrm{r},6}$ wird etwa auf [Haw13], [Kea17], [Ker18] oder [And18] verwiesen. Im Rahmen der vorliegenden Arbeit werden aus den genannten Quellen die Gleichungen

$$ q_1 = \operatorname{atan2}({}_{\mathrm{O_{UR}}}r_{5,2}, {}_{\mathrm{O_{UR}}}r_{5,1}) \pm \cos^{-1}\left(\frac{d_4}{\sqrt{{}_{\mathrm{O_{UR}}}r_{5,1}^2 + {}_{\mathrm{O_{UR}}}r_{5,12}^2}}\right) + \frac{\pi}{2} \tag{8-9} $$

$$ q_5 = \pm\cos^{-1}\left(\frac{{}^{O_{\mathrm{UR}},6}t_{1,4}\sin(q_1) - {}^{O_{\mathrm{UR}},6}t_{2,4}\cos(q_1) - d_4}{d_6}\right) \tag{8-10} $$

$$ q_6 = \operatorname{atan2}\left(\frac{-{}^{O_{\mathrm{UR}},6}t_{1,2}\sin(q_1) + {}^{O_{\mathrm{UR}},6}t_{2,2}\cos(q_1)}{\sin(q_5)}, \frac{{}^{O_{\mathrm{UR}},6}t_{1,1}\sin(q_1) - {}^{O_{\mathrm{UR}},6}t_{2,1}\cos(q_1)}{\sin(q_5)}\right) \tag{8-11} $$

$$ q_3 = \pm\cos^{-1}\left(\frac{\left|{}_1\vec{r}_3\right|^2 - a_2^2 - a_3^2}{2a_2a_3}\right) \tag{8-12} $$

$$ q_2 = \operatorname{atan2}\left({}^{1,4}t_{2,4}, -{}^{1,4}t_{1,4}\right) + \sin^{-1}\left(\frac{a_3\sin(q_3)}{\sqrt{{}^{1,4}t_{1,4}^2 + {}^{1,4}t_{2,4}^2}}\right) \tag{8-13} $$

$$ q_4 = \operatorname{atan2}\left({}^{3,4}t_{2,1}, {}^{3,4}t_{1,1}\right) + 2\pi \tag{8-14} $$

für die einzelnen Gelenkwinkel entnommen, wobei sich die dargestellte Reihenfolge der Berechnungen aus den gegenseitigen Abhängigkeiten ergibt. Die im Rahmen der Berechnung erforderlichen Parameter werden im Folgenden erläutert.

Die Transformationsmatrix ${}^{O_{\mathrm{UR}},6}\mathbf{T}$ vom Fußpunkt bis zum TCP des UR10 basiert dabei auf dem Ortsvektor ${}_{\mathrm{O_{UR}}}\vec{Z}_{\mathrm{TUR}}$ sowie dem Rotationsvektor ${}_{\mathrm{O_{UR}}}\vec{R}_{\mathrm{TUR}}$ des TCP. Mittels der Gleichung

$$ {}^{O_{\mathrm{UR}},6}\mathbf{A} = \begin{bmatrix} \underline{A}_1 & \underline{A}_2 & \underline{A}_3 \end{bmatrix} \tag{8-15} $$

mit

$$ \underline{A}_1 = \begin{bmatrix} \cos(R_\mathrm{p})\cos(R_\mathrm{y}) \\ \cos(R_\mathrm{r})\sin(R_\mathrm{y}) + \cos(R_\mathrm{y})\sin(R_\mathrm{r})\sin(R_\mathrm{p}) \\ \sin(R_\mathrm{r})\sin(R_\mathrm{y}) - \cos(R_\mathrm{r})\cos(R_\mathrm{y})\sin(R_\mathrm{p}) \end{bmatrix}, \tag{8-16} $$

$$ \underline{A}_2 = \begin{bmatrix} -\cos(R_\mathrm{p})\sin(R_\mathrm{y}) \\ \cos(R_\mathrm{r})\cos(R_\mathrm{y}) - \sin(R_\mathrm{r})\sin(R_\mathrm{p})\sin(R_\mathrm{y}) \\ \cos(R_\mathrm{y}) * \sin(R_\mathrm{r}) + \cos(R_\mathrm{r})\sin(R_\mathrm{p})\sin(R_\mathrm{y}) \end{bmatrix} \tag{8-17} $$

und

$$ \underline{A}_3 = \begin{bmatrix} \sin(R_\mathrm{p}) \\ -\cos(R_\mathrm{p})\sin(R_\mathrm{r}) \\ \cos(R_\mathrm{r})\cos(R_\mathrm{p}) \end{bmatrix} \tag{8-18} $$

wird aus den drei Winkeln R_r, R_p und R_y die Rotationsmatrix ${}^{O_{\mathrm{UR}},6}\mathbf{A}$ berechnet. Aus dieser und der Translation kann die Transformationsmatrix wie in Abschnitt 5.2.1, Gleichung (5-18), zusammengesetzt werden. Mit dieser und dem DHP d_6 wird

$$ {}_{\mathrm{O_{UR}}}\vec{r}_5 = {}^{O_{\mathrm{UR}},6}\mathbf{T} \begin{bmatrix} 0 \\ 0 \\ -d_6 \\ 1 \end{bmatrix} \tag{8-19} $$

ermittelt, sodass nun q_1 unter Nutzung des DHP d_4 berechnet werden kann (siehe Gleichung (8-9)). Mit q_1 und Elementen der Matrix ${}^{O_{\mathrm{UR}},6}\mathbf{T}$ kann nun q_5 (Gleichung (8-10)), sowie hiermit wiederum q_6 (Gleichung (8-11)) ermittelt werden.

Für die Berechnung von q_3 (Gleichung (8-12)) wird die Position

$$ {}_1\vec{r}_3 = {}^{1,4}\mathbf{T} \begin{bmatrix} 0 \\ -d_4 \\ 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{8-20} $$

des dritten Gelenks bezogen auf das erste Gelenk mit

$$ {}^{1,4}\mathbf{T} = {}^{1,6}\mathbf{T} \left({}^{4,5}\mathbf{T}\,{}^{5,6}\mathbf{T} \right)^{-1} \tag{8-21} $$

benötigt, wobei ${}^{1,4}\mathbf{T}$ auch für die Berechnung von q_2 und q_4 (Gleichungen (8-13) und (8-14)) erforderlich ist. Für die abschließende Ermittlung von q_4 wird zudem die Transformation

$${}^{3,4}\mathbf{T} = \left({}^{1,2}\mathbf{T}\,{}^{2,3}\mathbf{T}\right)^{-1}\,{}^{1,4}\mathbf{T} \tag{8-22}$$

zwischen dem dritten und dem vierten Gelenk benötigt. Durch die Lösung des vorgestellten IKP können die Stellungen $q_1 \ldots q_6$ aller sechs Gelenke für eine Sollpose ${}_{\mathrm{O_{UR}}}\vec{\xi}_{\mathrm{TUR}}$ berechnet werden. Bei einem Knickarmroboter können dabei nach [Sta09] vier verschiedene Fälle unterschieden werden:

Keine Lösung
Die vorgegebene Sollpose liegt außerhalb des Arbeitsbereichs des Roboters.

Genau eine Lösung
Die vorgegebene Sollpose liegt am Rand des Arbeitsraumes und kann nur bei vollständig ausgestrecktem Arm erreicht werden, stellt jedoch keine Singularität dar.

Endlich viele Lösungen
Die vorgegebene Sollpose kann durch mehrere unterschiedliche Stellungen der Robotergelenke (Gelenkkonfigurationen) erreicht werden.

Unendlich viele Lösungen
Die vorgegebene Sollpose kann durch unendlich viele Stellungen der Gelenke erreicht werden, es liegt eine Singularität vor. Dies tritt etwa auf, wenn die Drehachsen von zwei oder mehr Gelenken in Flucht liegen und sich somit die Bewegungen der entsprechenden Gelenke gegenseitig ausgleichen können.

In Bild 8-5 ist der Fall mit mehreren Lösungen dargestellt. Diese resultieren direkt aus den Gleichungen des IKP. Für q_1 wird der Kosinus-Term sowohl addiert als auch subtrahiert, was zu zwei Lösungen und somit zwei möglichen Positionen des Antriebs führt. Im Bereich der Robotik wird dies als *Arm links* oder *Arm rechts* bezeichnet [SSVO09]. Die jeweils zwei Lösungen für q_5 führen zu einer Verdrehung des Handgelenks, die zwei Positionen bei q_3 zu den Fällen *Ellenbogen oben* oder *Ellenbogen unten* [SSVO09]. Insgesamt ergeben sich aus den Gleichungen des IKP acht unterschiedliche Gelenkstellungen für eine Pose, diese sind in Bild 8-6 anhand des UR10 beispielhaft dargestellt. Aufgrund etwa des nicht-erreichbaren zylinderförmigen Bereichs innerhalb des Arbeitsraumes sind jedoch nicht immer alle Lösungen möglich, dies gilt es bei der Berechnung im Rahmen des modellprädiktiven Ansatzes auf Basis geeigneter mathematischer Formulierungen der Grenzen zu berücksichtigen, hierzu wird an dieser Stelle auf [Haw13] verwiesen.

Da die Gelenke des UR10 jeweils einen Winkelbereich von ±360° aufweisen (siehe Datenblatt im Anhang A1-12), gibt es für jedes Gelenk zusätzlich zu den mittels des IKP berechneten Lösungen die um +360° oder −360° verschobene Position. Somit ergeben sich theoretisch vier Positionen für q_1, für jede dieser Positionen wieder vier für q_5, dann wieder zwei für q_6 usw., sodass abschließend maximal $4 \cdot 4 \cdot 2 \cdot 4 \cdot 2 \cdot 2 = 512$ Lösungen vorliegen. Aufgrund von Singularitäten oder der Begrenzung des Arbeitsraumes sind jedoch nicht alle Varianten jederzeit nutzbar.

Im folgenden Abschnitt werden das DKP und das IKP des Raumportals vorgestellt.

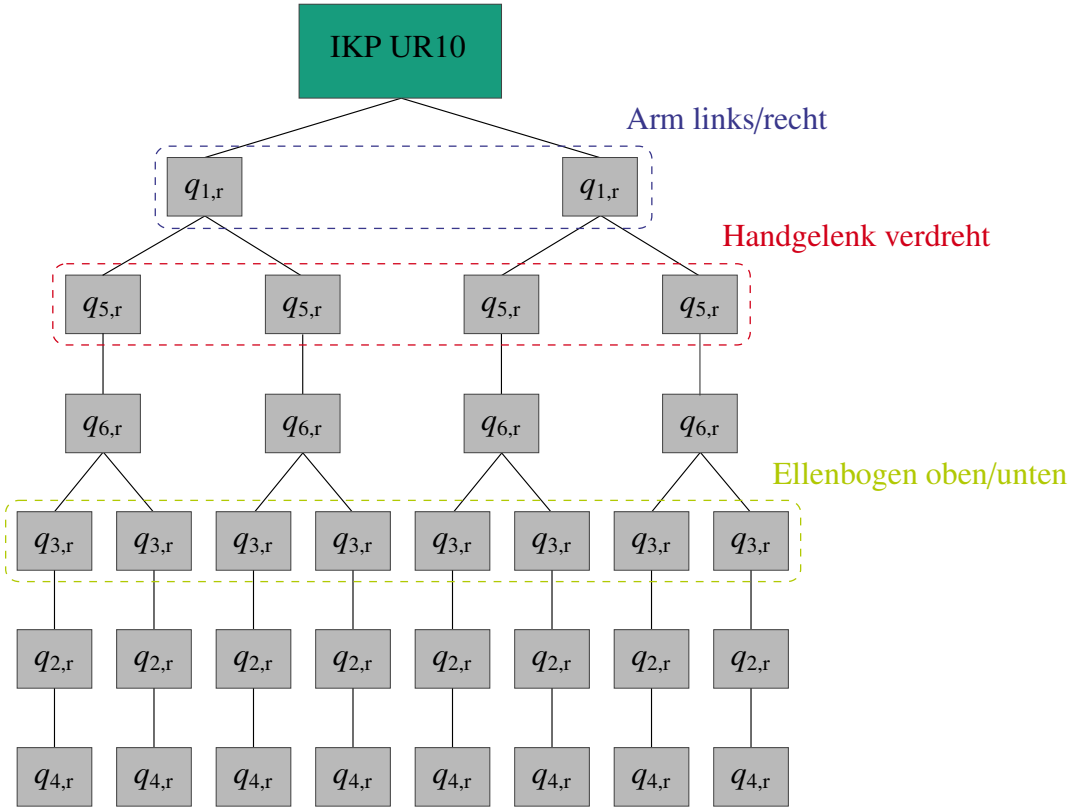


Bild 8-5: *Mögliche Gelenkkonfigurationen des UR10*

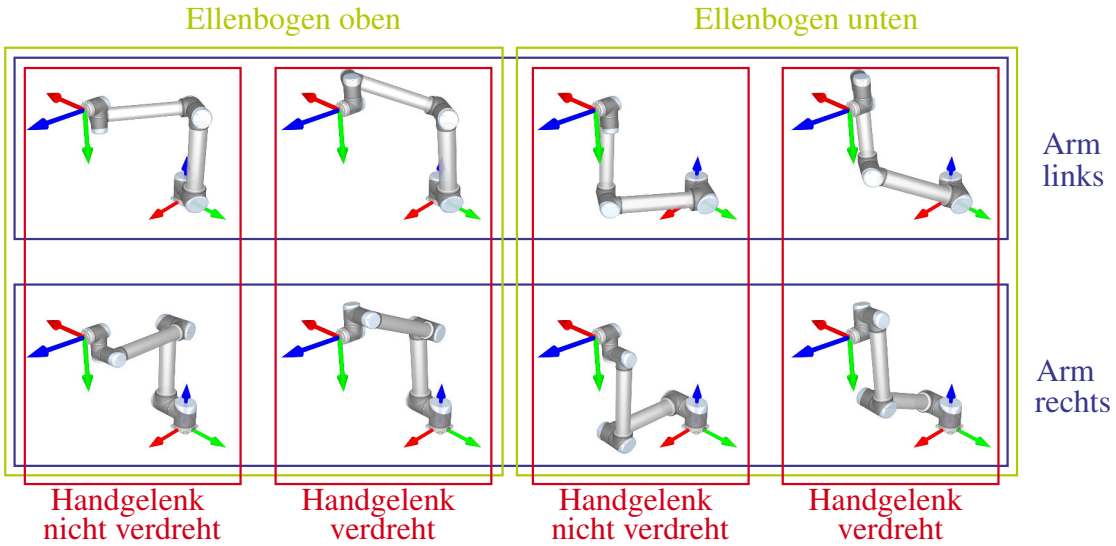


Bild 8-6: *Unterschiedliche Gelenkkonfigurationen des UR10 für eine TCP-Pose*

8.2.2 Inverses und direktes kinematisches Problem des Portals

Der Ortsvektor

$$ {}_{\mathrm{O_W}}\vec{Z}_{\mathrm{TP}} = \left[{}_{\mathrm{O_W}}Z_{\mathrm{TP,x}} \; {}_{\mathrm{O_W}}Z_{\mathrm{TP,y}} \; {}_{\mathrm{O_W}}Z_{\mathrm{TP,z}} \right]^T \in \mathbb{R}^3 \tag{8-23} $$

des Portals ähnelt stark der Gleichung (5-62) des Flächenportals aus dem Abschnitt 5.2.2. Aufgrund der Ausprägung dieses Teilmechanismus als Raumportal ist die z-Komponente ${}_{\mathrm{O_W}}Z_{\mathrm{TP,z}}$ des Vektors entgegen dem bisherigen Flächenportal ebenfalls variabel.

Auch hier gilt, dass die Position des TCP TP bezogen auf das Weltkoordinatensystem genau den Positionen der einzelnen Achsen entspricht. Die Achsstellung $q_7 = 0$, $q_8 = 0$ und $q_9 = 0$ entspricht dem Fall, dass sich der TCP im Ursprung O_{W} des Weltkoordinatensystems befindet. Das DKP ergibt sich ebenfalls wie in Abschnitt 5.2.2 beschrieben.

8.2.3 Simulative Validierung des inversen kinematischen Problems mittels Mehrkörpersimulation

Wie bereits bei der Ermittlung der Gleichungen der inversen kinematischen Probleme im Rahmen des ersten Anwendungsszenarios wird auch für den zweiten Mechanismus auf eine Mehrkörpersimulation in der Software *RecurDyn* zurückgegriffen. Ziel ist es mittels des Modells das IKP des UR10 simulativ zu validieren. Das IKP des Portals wird nicht erneut validiert. Es wurden unterschiedliche Sollposen mit unterschiedlichen Orientierungen untersucht. Tabelle 8-2 zeigt exemplarisch die Sollwerte für die sechs Elemente der Pose ${}_{\mathrm{O_{UR}}}\vec{\xi}_{\mathrm{TUR}}$ sowie jeweils die maximalen und minimalen Werte der resultierende Pose ${}_{\mathrm{O_{UR}}}\vec{\xi}_{\mathrm{TURres}}$ bezogen auf die für die Pose insgesamt möglichen 256 Gelenkkonfiguration. Es

Tabelle 8-2: Ergebnisse der Simulation für die erste Sollpose (maximale und minimale resultierende Werte)

	${}_{\mathrm{O_{UR}}}\vec{\xi}_{\mathrm{TUR}}$	$\min({}_{\mathrm{O_{UR}}}\vec{\xi}_{\mathrm{TURres}})$	$\max({}_{\mathrm{O_{UR}}}\vec{\xi}_{\mathrm{TURres}})$
${}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,x}}$	500 mm	499,8856 mm	500,1008 mm
${}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,y}}$	400 mm	399,9013 mm	400,1064 mm
${}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,z}}$	1000 mm	1001,70 mm	1001,80 mm
${}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,r}}$	90,00°	89,9987°	90,0008°
${}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,p}}$	0,00°	−0,0016°	0,0016°
${}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,y}}$	180,00°	179,6046°	179,6086°

ist an den Ergebnissen zu erkennen, dass die Abweichungen zwischen Soll- und Istpose bezogen auf die absoluten Werte gering sind. Für ${}_{\mathrm{O_{UR}}}Z_{\mathrm{TUR,z}}$ und ${}_{\mathrm{O_{UR}}}R_{\mathrm{TUR,y}}$ liegen nahezu konstante Abweichungen vor, dies kann in einer geringfügigen Abweichung der DHP vom CAD-Modell begründet liegen. Das grundlegende IKP inklusive der Erweiterung auf die Verfahrbereiche von ±360° je Gelenk erfüllt für die weiterführenden simulativen Analysen die Anforderungen, für den Einsatz am realen Roboter sollte jedoch ein Abgleich der DHP

erfolgen. Hierzu können beispielsweise die in jedem UR10 hinterlegten DHP ausgelesen und mit den bisher eingesetzten Werten verglichen werden.

Im folgenden Abschnitt wird die simulative Umsetzung des modellprädiktiven Ansatzes unter Nutzung der vorgestellten Gleichungen erläutert.

8.3 Simulative Umsetzung des modellprädiktiven Ansatzes

Im Rahmen der vorliegenden Arbeit erfolgt im Kontext der modellbasierten Entwicklung lediglich die simulative Analyse des Ansatzes für den im Rahmen des zweiten Anwendungsszenarios genutzten Mechanismus. Eine Umsetzung auf die reale Hardware wird als Ausblick gesehen. Bezüglich der Struktur des Systems hinsichtlich des Informationsaustausches und der synchronisierten Ansteuerung von Portal und Roboter kann hierzu auf der von Schütz et al. in [SRHT19], [SRHT20] und [SRHT21] im Rahmen eines Schweißvorgangs vorgestellten Struktur aufgesetzt werden. Dabei wird eine zentrale SPS eingesetzt, die wie im Rahmen der vorliegenden Arbeit die Ansteuerung des Portal realisiert, zusätzlich aber auch die Ansteuerung des Roboters, die Auswertung zusätzlicher prozessspezifischer Sensoren und die Einbindung zusätzlicher Komponenten übernimmt. Auf dieser zentralen SPS kann zusätzlich der modellprädiktive Optimierungsansatz implementiert werden.

Gegenüber des ersten Anwendungsszenarios werden für die durchzuführenden Analysen Sollposen anstelle von Sollpositionen benötigt. Die Generierung erfolgt erneut mittels eines Zufallsgenerator in *MATLAB* (vergleiche Abschnitt 6.1). In jedem Schritt werden sechs gleichverteilten Zufallszahlen (${}_{\mathrm{O_W}}Z_{\mathrm{TM,x}}$, ${}_{\mathrm{O_W}}Z_{\mathrm{TM,y}}$, ${}_{\mathrm{O_W}}Z_{\mathrm{TM,z}}$, ${}_{\mathrm{O_W}}R_{\mathrm{TM,r}}$, ${}_{\mathrm{O_W}}R_{\mathrm{TM,p}}$, ${}_{\mathrm{O_W}}R_{\mathrm{TM,y}}$) erzeugt. Diese liegen innerhalb der Grenzen

$$\begin{aligned}
-500\,\mathrm{mm} &\leq {}_{\mathrm{O_W}}Z_{\mathrm{TM,x}} \leq 500\,\mathrm{mm}\\
-500\,\mathrm{mm} &\leq {}_{\mathrm{O_W}}Z_{\mathrm{TM,y}} \leq 500\,\mathrm{mm}\\
1000\,\mathrm{mm} &\leq {}_{\mathrm{O_W}}Z_{\mathrm{TM,z}} \leq 2000\,\mathrm{mm}\\
-90^\circ &\leq {}_{\mathrm{O_W}}R_{\mathrm{TM,r}} \leq 90^\circ\\
-90^\circ &\leq {}_{\mathrm{O_W}}R_{\mathrm{TM,p}} \leq 90^\circ\\
-90^\circ &\leq {}_{\mathrm{O_W}}R_{\mathrm{TM,y}} \leq 90^\circ
\end{aligned} \tag{8-24}$$

und bilden in x-, y- und z-Richtung die Möglichkeiten des Gesamt-Mechanismus ab. Hinsichtlich der Orientierung wird der eigentlich mögliche Winkelbereich eingeschränkt. Es wird angenommen, dass nur Bauteile im Bereich zwischen Roboter und der x-y-Ebene des Weltkoordinatensystems bearbeitet werden.

Die Simulation erfolgt wie in Abschnitt 6.3 beschrieben unter Einsatz des Optimierers *Sblpx* mit den in Tabelle 8-3 aufgezeigten Parametern. Aufgrund der höheren Anzahl an Optimierungsvariablen wird eine höhere Abbruchgrenze n_{it} für die Anzahl an zugelassenen Iterationen verwendet.

In Tabelle 8-4 sind die resultierenden Verfahrzeiten sowohl des Referenzansatzes als auch der je Sollpose individuellen optimierten Aufteilung mittels des vorgestellten Ansatzes dargestellt. Wie bei den bisherigen Analysen handelt es sich dabei um die Mittels- sowie die Minimal- und Maximalwerte bezogen auf die 1000 Folgen der jeweiligen Länge N. Es ist auffällig, dass für den Referenzansatz (siehe Abschnitt 4.5) keine Werte eingetragen sind.

Tabelle 8-3: *Parameter für Sbplx bei der simulativen Ausführung im Rahmen des zweiten Anwendungsszenarios*

Parameter	Wert
Optimierer	Sbplx
n_p	3
n_it	10000
$\min(\Delta q_5, \Delta q_6)$	1×10^{-5} mm
$\min(\Delta J_\mathrm{rel})$	1×10^{-4} s
w_1	50
w_2	10
w_3	10
w_4	1
$q_{i,\mathrm{os}}$	5 mm

Tabelle 8-4: *Ergebnisse des Referenzverfahrens und des modellprädiktiven Ansatzes mit Sbplx in MATLAB für das zweite Anwendungsszenario*

N	$\overline{T}_\mathrm{ref}$	$\overline{T}_\mathrm{sm}$	$T_\mathrm{sm,min}$	$T_\mathrm{sm,max}$	$\Delta\overline{T}_\mathrm{sm}$
10	-	12,9 s	10,95 s	15,79 s	-
50	-	65,38 s	61,91 s	69,80 s	-
100	-	1304,27 s	1250,19 s	1377,17 s	-

Der Optimierer *fmincon* findet keine feste Aufteilung $\underline{\xi}_{\text{ref}}$, die für alle Sollposen einer Folge der Länge N alle Nebenbedingungen erfüllt. Dies liegt im kugelförmigen Arbeitsraum des UR10 mit dem innen liegenden nicht-erreichbaren zylinderförmigen Bereich und der Abhängigkeit von Position und erreichbarer Pose begründet. Der UR10 kann nicht in jeder Position jede Orientierung erreichen. Dies führt dazu, dass der Roboter für eine geforderte Orientierung in eine Position verfährt, die durch das Portal kompensiert werden muss. Die große Varianz an vorgegebenen Sollposen kann durch eine feste Aufteilung nicht mehr erreicht werden. Eine weitere Analyse mittels eines *Brute-Force*-Ansatzes[22], bei dem mit einer Schrittweite von $\Delta\xi_{\text{ref}} = 0,05$ alle möglichen Kombinationen für die Elemente von $\underline{\xi}_{\text{ref}}$ zwischen Null und Eins analysiert wurden, liefert die gleiche Aussage. Somit ist für einen derartigen kinematisch redundanten Mechanismus zwingend eine individuelle Bewegungsaufteilung je Sollpose erforderlich, welche durch die vorgestellte echtzeitfähige modellprädiktive Planung zeitoptimierter Trajektorien ermittelt wird.

Da keine Referenzzeiten für die weitere Validierung des Funktionsprinzips der modellprädiktiven Planung vorliegen, werden wie bereits beim ersten Anwendungsszenario zusätzlich die Verfahrzeiten der jeweils langsamsten Achse der beiden Teilmechanismen verglichen. Diese müssen entsprechend der grundlegenden Überlegungen (vergleiche Abschnitt 6.4.2) ihre jeweilige Sollposition für einen überwiegenden Teil der Sollposen nahezu zeitgleich erreichen. In Bild 8-7 sind die Verfahrzeiten für zwei unterschiedliche Folgen der Länge $N = 10$ dargestellt.

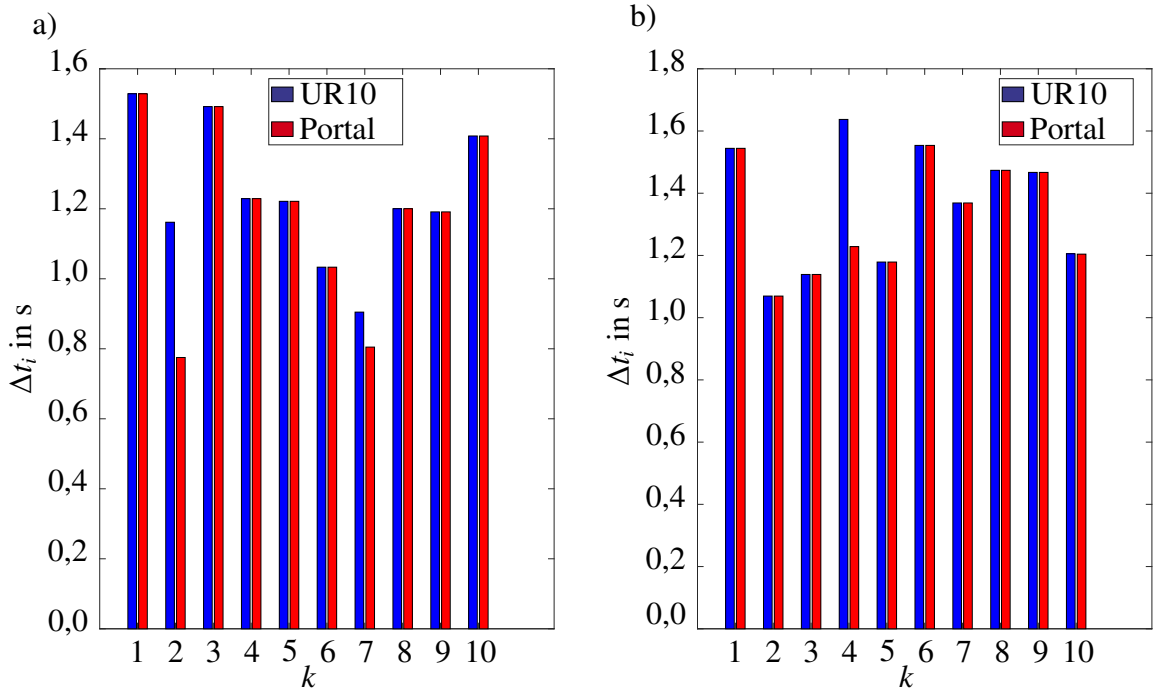


Bild 8-7: *Verfahrzeiten der jeweils langsamsten Achse von UR10 und Portal bei* $N = 10$ *für die Sollpositionsfolgen a)* $r10_1$ *und b)* $r10_2$

Es ist zu erkennen, dass die Verfahrzeiten der langsamsten Achse von Portal und UR10 für einen Großteil der dargestellten Posen nahezu identisch sind und der Ansatz somit die

[22] Mit *Brute-Force* (engl. rohe Gewalt) wird im Bereich der Kryptografie eine Angriffsmethode bezeichnet, bei der alle möglichen Kombination, beispielsweise aus Zahlen, Buchstaben und Sonderzeichen, nacheinander ausprobiert werden, um einen Passwortschutz zu umgehen [LLL+20].

Anforderung erfüllt. Die Ausnahmen bei der ersten sowie bei der zweiten Folge ($k = 2$ und $k = 7$ sowie $k = 4$) liegen in einer Änderung der Orientierung begründet, aus der sich für einzelne Gelenke des UR10 ein größerer Verfahrweg ergibt. Hinzu kommt die Wahl eines Prädiktionshorizonts von $n_\mathrm{p} = 3$, sodass die Verfahrwege der Gelenke nicht nur aus der nächsten Sollpose resultieren, sondern auch aus dem Streben nach einer „guten“ Ausgangskonfiguration für die folgenden Posen.

Insgesamt lässt die durchgeführte Simulation im Rahmen der Adaption der modellprädiktiven Planung auf einen HKM bestehend aus einem Sechsachs-Knickarmroboter und einem dreiachsigen Raumportal den Schluss zu, dass der Ansatz die an ihn gestellten Erwartungen übergreifend erfüllt und sehr gut für die Ermittlung zeitoptimierter Trajektorien für kinematisch redundante mehrachsige Mechanismus eingesetzt werden kann.

9 Zusammenfassung und Ausblick

Nach einer Vorstellung relevanter Grundlagen und Begrifflichkeiten wird in der vorliegenden Arbeit zunächst der aktuelle Stand der Wissenschaft und Technik bezogen auf drei Themenfelder diskutiert. Dabei handelt es sich um die Regelung seriellkinematischer, parallelkinematischer und hybridkinematischer Mechanismen im Allgemeinen, die optimale Regelung dieser Mechanismen mit und ohne Redundanz bezogen auf verschiedene Kriterien und die Ausführung modell- und optimierungsbasierter Ansätze auf industrieller Hardware. Daran anschließend wird als Kern dieser Arbeit die echtzeitfähige modellprädiktive Planung zeitoptimierter Trajektorien für kinematisch redundante Mechanismen erläutert. Diese ermöglicht eine Minimierung der Verfahrzeit zwischen diskreten Sollpositionen im Rahmen von PTP-Bewegungen unter Nutzung der kinematischen Redundanz. Die insgesamt durchzuführenden Bewegungen werden individuell für jede Sollposition derart auf die entsprechenden Achsen des Gesamtmechanismus aufgeteilt (*joint space decomposition*), dass die für die gesamte Bewegung benötigte Zeit minimiert wird. Als erstes Anwendungsszenario wird die Adaption des Ansatzes auf einen kinematisch redundanten HKM bestehend aus einem Tricept und einem zusätzlichen zweiachsigen Flächenportal vorgestellt. Entsprechend des methodischen Vorgehens der modellbasierten Entwicklung wird die Funktionsfähigkeit des Ansatzes zunächst virtuell abgesichert. Dies erfolgt simulativ in der Software *MATLAB*. Die dabei durchgeführten Analysen zeigen, dass mit dem modellbasierten Ansatz Einsparungen der Verfahrzeit zwischen 18 % und 30 % gegenüber einem Vergleichsansatz mit fester Aufteilung der Bewegung erzielt werden können. Zudem wird anhand der Simulation gezeigt, dass der Ansatz auch bei zufällig auftretenden Änderungen zukünftiger Sollpositionen des Mechanismus gute Ergebnisse erzielt, er ist somit im Rahmen flexibler Fertigungen mit kleinen und kleinsten Losgrößen im Kontext Industrie 4.0 einsetzbar. Aufgrund dieses Ergebnisses erfolgt im Anschluss die Umsetzung auf realer Hardware im Rahmen zweier unterschiedlicher Varianten. Im ersten Fall wird die modellbasierte Optimierung auf eingebetteter Hardware ausgeführt, die mittels OPC UA mit einer SPS verbunden ist. Hierbei werden nahezu gleiche Einsparungen wie bei den Analysen in *MATLAB* erzielt. Bezogen auf die reine Optimierung wird zudem das Ziel der Echtzeitfähigkeit, hier definiert als Zykluszeiten ≤ 1 ms, erreicht. Die Kommunikation zwischen den beiden Systemen sorgt jedoch für Verzögerungen, wodurch die insgesamt erforderliche Rechenzeit die zeitliche Grenze überschreiten. Deswegen erfolgt im zweiten Fall die vollständige Umsetzung auf einer Industriesteuerung, wobei keine zusätzliche Kommunikation benötigt wird. Bei nahezu gleichen Einsparungen der Verfahrzeit liegt die erforderliche Rechenzeit bei unter 0,7 ms, somit ist das Ziel der echtzeitfähigen Ausführung mit Steuerungsechtzeit erreicht. Abschließend wird die Leistungsfähigkeit des modellbasierten Ansatzes durch die Adaption auf einen zweiten kinematisch redundanten Mechanismus bestehend aus einem drei-achsigen Raumportal und einem Sechsachs-Knickarmroboter simulativ demonstriert. Die Ergebnisse zeigen, dass für ein derartiges System die bisher als Vergleichsansatz genutzte, für alle Sollpositionen einer Folge feste, Aufteilung der Bewegungen keine plausiblen Ergebnisse liefert. Vielmehr ist zwangsläufig eine für jede Sollpose individuelle Aufteilung der Positionen erforderlich. Der vorgestellte modellprädiktive Optimierungsansatz liefert hier eine Lösungsansatz, der zusätzlich simulativ validiert wurde.

Abschließend gilt es zu analysieren, inwieweit die in Abschnitt 1.2 definierten Anforderungen an den vorgestellten Ansatz erfüllt werden:

Handhabung der Unbestimmtheit des IKP kinematisch redundanter Mechanismen
Die Unbestimmtheit des IKP kinematisch redundanter Mechanismen wird durch den vorgestellten Ansatz genutzt, um die erforderliche Verfahrzeit zwischen diskreten Sollpositionen im Rahmen von PTP-Bewegungen zu minimieren. Dazu wird die insgesamt durchzuführende Bewegung aufgeteilt auf den *nicht-redundanten* Teilmechanismus sowie den *Redundanz-erzeugenden* Teilmechanismus. Letzterer besteht aus den zusätzlichen Achsen, die zu einer kinematischen Redundanz des Gesamt-Mechanismus führen. Mittels des Ansatzes wird das IKP des Mechanismus eindeutig lösbar, die für eine Sollpositionen erforderlichen Achspositionen können eindeutig berechnet werden. Die Anforderung ist damit erfüllt.

Onlinefähigkeit
Der vorgestellte Ansatz nutzt neben einem Prädiktionshorizont von $n_\mathrm{p} = 3$ das Prinzip des gleitenden Horizonts. Es werden immer die nächsten drei Sollpositionen betrachtet, nach erfolgter Optimierung wird jedoch nur der ermittelte Wert für die nächste Position genutzt und der betrachtete Bereich dann um eine Sollposition in die Zukunft verschoben. Sollte sich einer der zukünftigen Positionen zur Laufzeit ändern, so kann im nächsten Zyklus online darauf reagiert werden. Zusätzlich wird über eine exponentiell abfallende Gewichtung zukünftiger Sollwerte der Einfluss kurzfristiger Änderungen auf die Gesamt-Verfahrzeit einer Folge reduziert. Im Abschnitt 6 wird diese simulativ nachgewiesen, der Ansatz ist wie gefordert online-fähig.

Ausführbarkeit auf SPS und industrielle Akzeptanz
Durch die Nutzung einer als *C*-Code verfügbaren Optimierungsbibliothek und Durchführung der erforderlichen Quellcode-Anpassungen sowie die Implementierung des Zeitmodells, des IKP und zusätzlicher Bausteinen etwa für die Verwaltung der Sollpositionen in *C* und *strukturierten Text* kann der modellprädiktive Ansatz auf einer Industriesteuerung ausgeführt werden. Diese Anforderung wird somit erfüllt.

Echtzeitfähigkeit
Bei Ausführung des modellprädiktiven Ansatzes auf einer SPS liegen die benötigten Rechenzeiten unter der geforderten Zeit von 1 ms. Durch die Verwendung eines Echtzeit-Systems mit einer definierten Zykluszeit in Kombination mit der Begrenzung der maximalen Anzahl an Iterationen des Optimierers treten keine Ausreißer in der Rechenzeit auf, die die geforderte Grenze überschreiten. Wird der Ansatz auf dem RPI als eingebettetes System ausgeführt, so liegt die benötigte Rechenzeit ebenfalls immer unter 1 ms. Der modellprädiktive Ansatz selber ist somit auch echtzeitfähig im Sinne der im Rahmen der Arbeit genutzten Definition (vgl. Abschnitt 2.2). Als Zusatz ist festzuhalten, dass die Definition der Zeitschwelle abhängig vom Anwendungsfall und dem Mechanismus anders gewählt werden kann oder muss (vgl. Abschnitt 1.2). Die im Rahmen der vorliegenden Arbeit genutzte Schwelle ermöglicht jedoch eine Einsetzbarkeit für eine große Bandbreite an Anwendungsfällen.

Retrofit
Insbesondere der Ansatz mit der Ausführung der modellprädiktiven Optimierung auf einem eingebetteten System (RPI) eignet sich für die Integration in eine bereits existierende Anlage im Rahmen eines Retrofits. Es gilt jedoch eine geeignete Kom-

munikation zwischen dem RPI und der SPS des ursprünglichen Systems zu wählen, damit die insgesamt erforderliche Zeit als Kombination von Rechen- und Kommunikationszeit nicht über 1 ms bzw. über der für das Gesamtsystem erforderlichen Zykluszeit liegt.

Nutzung der Möglichkeiten moderner Automatisierungssysteme
Um möglichst viel Rechenleistung für die Optimierung nutzen zu können, gilt es die SPS von zusätzlichen Aufgaben zu entlasten. Aus diesem Grund wird in der vorliegenden Arbeit die Positionsregelung der eingesetzten Antriebe inklusive der unterlagerten Geschwindigkeits- und Stromregelung auf die Antriebshardware ausgelagert. Zudem übernimmt diese die Überwachung der Grenzen für Beschleunigung, Ruck etc., sodass diese nicht in der Optimierung berücksichtigt werden müssen.

Kommerzielle Nutzbarkeit
Der Punkt der kommerziellen Nutzbarkeit ist insbesondere hinsichtlich der Lizenzierung des eingesetzten Optimierers beziehungsweise der entsprechenden Optimierungsbibliothek entscheidend. Die *NLopt*-Bibliothek steht unter der *GNU Lesser General Public License*. Dies ermöglicht die freie Nutzung der Bibliothek sowie ihre Veränderung, die Weitergabe und den Verkauf der resultierenden Software. Das Copyleft besagt lediglich, dass dem Endnutzer der aus der ursprünglichen Bibliothek stammende Code offengelegt werden muss, nicht jedoch die gesamte verkaufte Software. Somit stellt die Lizenz der Optimierungsbibliothek kein Hindernis bei der Vermarktung der Software dar.

Adaptierbarkeit
Wie anhand zweier unterschiedlicher mehrachsiger kinematisch redundanter HKM im Rahmen der beiden Anwendungsszenarien gezeigt wurde, ist der modellprädiktive Ansatz auf unterschiedliche Mechanismen adaptierbar. Somit ist diese Anforderung erfüllt.

Zusammenfassend lässt sich somit festhalten, dass die vorgestellte modellprädiktive Planung zeitoptimierter Trajektorien für mehrachsige redundante Mechanismen bezogen auf den kinematisch redundanten HKM aus dem ersten Anwendungsszenario Einsparungen der Verfahrzeit bei PTP-Bewegungen zwischen 18 % und 30 % ermöglicht. Um den Ansatz bei einer Kombination aus eingebetteter Hardware und SPS unter Echtzeitbedingungen einzusetzen, muss über eine Alternative zur Kommunikation mittels OPC UA nachgedacht werden. Hier bietet sich OPC UA TSN als Kommunikationsprotokoll an, welches ebenfalls herstellerübergreifend nutzbar ist, explizit aber für Echtzeitanwendungen entwickelt wurde. Dies muss jedoch durch die eingesetzte Hardware unterstützt werden, was bei den im Rahmen der Arbeit eingesetzten Komponenten nicht der Fall ist. Somit ist dies als Ausblick für zukünftige Arbeiten zu sehen.

Bezogen auf den zweiten Anwendungsfall ist festzuhalten, dass mit der festen Bewegungsaufteilung im Rahmen des Referenzansatzes keine praktikable Nutzung des Mechanismus möglich ist. Dies wird erst durch den modellprädiktiven Optimierungsansatz und die damit verbundene individuelle Bewegungsaufteilung für jede Sollpose erreicht. In zukünftigen Arbeiten gilt es daher die bisher simulativ durchgeführte Adaption auf den zweiten Mechanismus auf die reale Hardware zu übertragen und hinsichtlich der Echtzeitfähigkeit zu untersuchen. Dabei gilt es auch die Wahl der Abbruchgrenzen des Optimierers insbesondere hinsichtlich der maximalen Anzahl an Iterationen n_{it} zu überprüfen. Um die

erzielten Verfahrzeiten bewerten zu können, wird zudem ein entsprechender Referenzansatz benötigt.

In einem weiteren Entwicklungsschritt kann zudem die Gütefunktion um zusätzliche Terme etwa für die Kollisionserkennung ergänzt oder das Zeitmodell von Punkt-zu-Punkt-Bewegungen auf kontinuierliche Bewegungen erweitert werden, um die Durchführung weiterer industrieller Anwendungsszenarien mit den betrachteten kinematisch redundanten Mechanismen zu ermöglichen.

Literaturverzeichnis

[ABMS08] Andreani, R.; Birgin, E. G.; Martínez, J. M.; Schuverdt, M. L.: On Augmented Lagrangian Methods with General Lower-Level Constraints. *SIAM Journal on Optimization* 18 (2008), Nr. 4, S. 1286–1309

[ABO05] Armand, P.; Benoist, J.; Orban, D.: *Interpretation of interior point methods as damped Newton methods*. 2005. `https://pdfs.semanticscholar.org/dda0/91260b743aa670c2c64d128526de10210fa8.pdf` (besucht am 01. 11. 2018)

[AHSZ17] Aslam, S.; Hannan, S.; Sajjad, M. U.; Zafar, W.: PLC based model predictive control for industrial process control. *International Journal of Advanced and Applied Sciences* 4 (2017), Nr. 6, S. 63–71

[AIS11] Ayten, K. K.; Iravani, P.; Sahinkaya, M. N.: Optimum Trajectory Planning for Industrial Robots through Inverse Dynamics. *ICINCO - 8th International Conference on Informatics in Control, Automation and Robotics*. 2011

[AM06] Ata, A. A.; Myo, T. R.: Optimal Point-to-Point Trajectory Tracking of Redundant Manipulators using Generalized Pattern Search. *CoRR* abs/cs/0601063 (2006)

[And18] Andersen, R. S.: *Kinematics of a UR5*. 2018. `http://www.rasmusan.blog.aau.dk/files/ur5_kinematics.pdf` (besucht am 23. 05. 2018)

[APW07] Alba-Gómez, O. G.; Pamanes, J. A.; Wenger, p.: Trajectory planning of a redundant parallel manipulator changing of working mode. *Proceedings of Twelfth World Congress in Mechanism and Machine Science, IFToMM*. 2007

[APW08] Alba-Gómez, O. G.; Pamanes, J. A.; Wenger, P.: Trajectory Planning of Parallel Manipulators for Global Performance Optimization. *Advances in Robot Kinematics: Analysis and Design*. Hrsg. von Lenarčič, J.; Wenger, P. Dordrecht: Springer Netherlands, 2008, S. 253–261

[Ayt12] Ayten, K. K.: *Optimum Trajectory Planning for Redundant Manipulators Through Inverse Dynamics: Dissertation*. 2012.

[BB10] Briot, S.; Bonev, I. A.: Pantopteron-4: A new 3T1R decoupled parallel manipulator for pick-and-place applications. *Mechanism and machine theory* 45 (2010), Nr. 5, S. 707–721

[BC03] Basile, F.; Chiacchio, P.: A contribution to minimum-time task-space path-following problem for redundant manipulators. *Robotica* 21 (2003), Nr. 02

[BDG83] Bobrow, J. E.; Dubowsky, S.; Gibson, J. S.: On the Optimal Control of Robotic Manipulators with Actuator Constraints. *1983 American Control Conference*. 1983, S. 782–787

[BDG85] Bobrow, J. E.; Dubowsky, S.; Gibson, J. S.: Time-Optimal Control of Robotic Manipulators Along Specified Paths. *The International Journal of Robotics Research* 4 (1985), Nr. 3, S. 3–17

[BDV05] Biloti, R.; D Afonesca, L.; Ventura, S.: *Open Optimization Library: Reference Manual Edition 0.2.0, for OOL Version 0.2.0*. 2005. http://ool.sourceforge.net/ool-ref.pdf

[Bec17] Beckhoff Automation GmbH & Co. KG: *AG2250*. 2017. https://download.beckhoff.com/download/document/motion/ag2250_ba_en.pdf (besucht am 11. 11. 2019)

[Bec18a] Beckhoff Automation GmbH & Co. KG: *Servoverstärker AX5000: Systemhandbuch*. Hrsg. von Beckhoff Automation GmbH & Co. KG. 2018.

[Bec18b] Beckhoff Automation GmbH & Co. KG: *Synchron - Servomotoren AM8100: für kompakte Antriebstechnik*. 2018. https://download.beckhoff.com/download/document/motion/am8100_ba_de.pdf (besucht am 11. 11. 2019)

[Bec19a] Beckhoff Automation GmbH & Co. KG: *AM8000 + AM8500 Synchron-Servomotor: Original-Betriebsanleitung DE*. 2019. https://download.beckhoff.com/download/document/motion/am8000_am8500_ba_de.pdf (besucht am 12. 11. 2019)

[Bec19b] Beckhoff Automation GmbH & Co. KG: *C6030-0060*. 2019. https://download.beckhoff.com/download/Document/Catalog/Main_Catalog/german/Einzelseiten/Industrie-PC/c6030_0060.pdf (besucht am 12. 11. 2019)

[Bec19c] Beckhoff Automation GmbH & Co. KG: *TC3 C++: Handbuch*. 2019. http://download.beckhoff.com/download/document/automation/twincat3/TC1300_C_DE.pdf (besucht am 08. 11. 2019)

[Bec20] Beckhoff Automation GmbH & Co. KG: *News*. 2020. https://beckhoff.de/ (besucht am 06. 05. 2020)

[Bec23] Beckhoff Automation GmbH & Co. KG: *EP7342-0002 2-Kanal-DC-Motor-Endstufe 48 V DC, 3,5 A - Kv-Faktoren*. 2023. https://infosys.beckhoff.com/index.php?content=../content/1031/ep7342-0002/8830000907.html&id=4987556148500217984 (besucht am 19. 11. 2023)

[Ber20] Bertelsmeier, F.; Trächtler, A. (BetreuerIn); Dumirescu, R. (BetreuerIn): *Produkttolerante Automation zellenbasierter Fertigungssysteme*. Dissertation. Paderborn: Universität Paderborn, 2020.

[BHS98] Bartenschlager, J.; Hebel, H.; Schmidt, G.: *Handhabungstechnik mit Robotertechnik*. Wiesbaden: Vieweg+Teubner Verlag, 1998

[BK15] Briot, S.; Khalil, W.: *Dynamics of parallel robots: From rigid bodies to flexible elements*. Bd. volume 35. Mechanisms and machine science. Springer International Publishing Switzerland, 2015

[BKPJ14] Binder, B. J. T.; Kufoalor, D. K. M.; Pavlov, A.; Johansen, T. A.: Embedded Model Predictive Control for an Electric Submersible Pump on a Programmable Logic Controller. *IEEE Conference on Control Applications (CCA)* (2014)

[BL04] Buttelmann, M.; Lohmann, B.: Optimierung mit Genetischen Algorithmen und eine Anwendung zur Modellreduktion (Optimization with Genetic Algorithms and an Application for Model Reduction). *at - Automatisierungstechnik* 52 (2004), Nr. 4-2004, S. 151–163

[BLNZ95] Byrd, R. H.; Lu, P.; Nocedal, J.; Zhu, C.: A Limited Memory Algorithm for Bound Constrained Optimization. *SIAM Journal on Scientific Computing* 16 (1995), Nr. 5, S. 1190–1208

[BM02] Birgin, E. G.; Mario Martínez, J.: Large-Scale Active-Set Box-Constrained Optimization Method with Spectral Projected Gradients. *Computational Optimization and Applications* 23 (2002), Nr. 1, S. 101–125

[Bob82] Bobrow, J. E.: *Optimal Control of Robotic Manipulators*. THESIS (Ph.D) - UNIVERSITY OF CALIFORNIA, 1982. University Microfilms, 1982

[BR 19] B&R Industrial Automation GmbH: *Häufig gestellte Fragen zu OPC UA OVER TSN*. Hrsg. von B&R. 2019. https://www.br-automation.com/downloads_br_productcatalogue/assets/1570369249721-de-original-1.2.pdf

[BR19a] B&R: *OPC UA over TSN: Häufig gestellte Fragen zu OPC UA over TSN*. 2019. https://www.br-automation.com/downloads_br_productcatalogue/assets/1570369249721-de-original-1.1.pdf (besucht am 29. 11. 2019)

[BR19b] B&R: *X20CP1381-RT und X20CP1382-RT: Datenblatt V 1.23*. 2019. https://download.br-automation.com/BRP444000000%2000000000621406/X20CP13xx-RT-GER_V1.23.pdf?px-hash=1d20d7e9757100594e211ce9708c31df&px-time=1573585012 (besucht am 12. 11. 2019)

[Bre73] Brent, R. P.: *Algorithms for minimization without derivatives*. (Prentice-Hall series in automatic computation). Englewood Cliffs, N.J.: Prentice-Hall, 1973

[Bru20] Bruckner, D.: *OPC UA over TSN ist 18-mal schneller als bisherige Protokolle. Dietmar Bruckner erklärt, wieso dieser Performancesprung notwendig ist.* 2020. https://www.br-automation.com/de-at/ueber-uns/presse/technology-highlights/opc-ua-over-tsn-muss-fuer-die-performance-anforderungen-der-naechsten-20-jahre-geruestet-sein/ (besucht am 13. 05. 2020)

[BSMM06] Bronstein, I. N.; Semendjajew, K. A.; Musiol, G.; Mühlig, H.: *Taschenbuch der Mathematik*. 6., vollst. überarb. und erg. Aufl., Nachdruck. Frankfurt am Main: Deutsch, 2006

[BST16] BST eltromat: *Produktdatenblatt EMS 18*. 2016

[Bun15] Bundesministerium für Wirtschaft und Energie - Öffentlichkeitsarbeit, Hrsg.: *Industrie 4.0 und Digitale Wirtschaft: Impulse für Wachstum, Beschäftigung und Innovation*. 2015.

[CEB08] Carretero, J. A.; Ebrahimi, I.; Boudreau, R.: A Comparison between Two Motion Planning Strategies for Kinematically Redundant Parallel Manipulators. *Advances in Robot Kinematics*. Hrsg. von Lenarcic, J.; Philippe, W. Berlin: Springer Netherland, 2008, S. 243–252

[CEB12] Carretero, J. A.; Ebrahimi, I.; Boudreau, R.: Overall Motion Planning for Kinematically Redundant Parallel Manipulators. *Journal of Mechanisms and Robotics* 4 (2012), Nr. 2, S. 024502

[CLV06] Cha, S.-H.; Lasky, T. A.; Velinsky, S. A.: Kinematic Redundancy Resolution for Serial-Parallel Manipulators via Local Optimization Including Joint Constraints. *Mechanics Based Design of Structures and Machines* 34 (2006), Nr. 2, S. 213–239

[CLV07a] Cha, S.-H.; Lasky, T. A.; Velinsky, S. A.: Kinematically-Redundant Variations of the 3- R RR Mechanism and Local Optimization-Based Singularity Avoidance. *Mechanics Based Design of Structures and Machines* 35 (2007), Nr. 1, S. 15–38

[CLV07b] Cha, S.-H.; Lasky, T. A.; Velinsky, S. A.: Singularity Avoidance for the 3-RRR Mechanism Using Kinematic Redundancy. *2007 IEEE International Conference on Robotics and Automation*. 2007, S. 1195–1200

[COD19] CODIAN-ROBOTICS: *Zweidimensionale Delta-Roboter*. 2019. (Besucht am 22. 10. 2019)

[Cor17] Corke, P.: *Robotics, Vision and Control*. Bd. 118. Cham: Springer International Publishing, 2017

[Cra94] Craig, J. J.: *Introduction to robotics: Mechanics and control*. 2. ed., [Nachdr.] Addison-Wesley series in electrical and computer engineering: control engineering. Reading, Mass. [u.a.]: Addison-Wesley, 1994

[DEN20] DENSO Robotics Europe: *Technical Specification VS-068 / VS-087*. 2020. `https://www.densorobotics-europe.com/fileadmin/Products/VS_-_068_and_VS_087-Technical_Data_Sheet/VS_-_068_and_VS_087-Technical_Data_Sheet.pdf` (besucht am 11. 05. 2020)

[DFA03] Dawes, B.; Frey, D.; Abrahams, D.: *Boost Software Licence*. 2003. `https://www.boost.org/LICENSE_1_0.txt` (besucht am 27. 10. 2018)

[DGM+08] Denkena, B.; Günther, G.; Mehrmann, V.; H.-C., M.; Steinbrecher, A.: *Kalibrierverfahren für hybride Parallelkinematiken*. 2008

[DH08] Deuflhard, P.; Hohmann, A.: *Numerische Mathematik 1: Eine algorithmisch orientierte Einführung*. 4., überarb. und erw. Aufl. Bd. 1, Ed. 4. Numerische Mathematik. Berlin: de Gruyter, 2008

[Dit04] Dittmar, R.: *Modellbasierte prädiktive Regelung: Eine Einführung für Ingenieure*. 1. Aufl. München: Oldenbourg, 2004

[ECB07a] Ebrahimi, I.; Carretero, J. A.; Boudreau, R.: Actuation Scheme for a 6-DOF Kinematically Redundat Planar Parallel Manipulator. *Proceedings of Twelfth World Congress in Mechanism and Machine Science, IFToMM 2007: June 17 - 21, 2007, Besançon, France*. Hrsg. von Merlet, J. P.; Mechanism, I. F. f. t. P. o.; Science, M. 2007

[ECB07b] Ebrahimi, I.; Carretero, J. A.; Boudreau, R.: The 3-RPRR Kinematically Redundant Planar Parallel Manipulator. *The 2007 CCToMM Symposium on Mechanisms, Machines, and Mechatronics* (2007)

[ECB08] Ebrahimi, I.; Carretero, J. A.; Boudreau, R.: Kinematic analysis and path planning of a new kinematically redundant planar parallel manipulator. *Robotica* 26 (2008), Nr. 03

[EPS11] EPSON-EUROPE: *Epson bringt neue Familie effizienter SCARA-Roboter für Montage und Handhabung auf den Markt: Die neue SCARA LS-Serie bietet hohe Produktivität und Qualität*. 2011. https://neon.epson-europe.com/de/de/robots/?id=1053&content=656 (besucht am 21. 06. 2017)

[FAN17] FANUC: *LR Mate 200iD: Mehrzweckroboter mit hohem Durchsatz*. 2017. http://www.fanuc.eu/at/de/roboter/roboterfilter-seite/lrmate-serie/lrmate-200-id (besucht am 21. 06. 2017)

[Fd15] Fontes, J. V.; da Silva, M. M.: Assessing the Dynamic Performance of Kinematically Redundant Manipulators. *23rd ABCM International Congress of Mechanical Engineering*. Hrsg. von Mattos, H. S. d. C. 23rd ABCM International Congress of Mechanical Engineering. 2015

[Fd16] Fontes, J. V.; da Silva, M. M.: On the dynamic performance of parallel kinematic manipulators with actuation and kinematic redundancies. *Mechanism and machine theory* 103 (2016), S. 148–166

[FES17] FESTO: *2D Linienportale: YXCL*. 2017. https://www.festo.com/cat/de_de/products__43903 (besucht am 21. 06. 2017)

[Fes18] Festo Vertrieb GmbH & Co. KG: *Tripod EXPT: Deltakinematik für High-Speed-Anwendungen*. 2018. https://www.festo.com/rep/de_de/assets/EXPT_9747mk_1_710px.jpg (besucht am 15. 02. 2018)

[FF08] Fetzer, A.; Fränkel, H.: *Mathematik 1: Lehrbuch für ingenieurwissenschaftliche Studiengänge*. 10., bearbeitete Aufl. Springer-Lehrbuch. Berlin und Heidelberg: Springer-Verlag, 2008

[FKP+14] Ferreau, H. J.; Kirches; Potschka, A.; Bock, H. G.; Diehl, M.: qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation* 6 (2014), Nr. 4, S. 327–363

[Föl13] Föllinger, O.: *Regelungstechnik - Einführung in die Methoden und ihre Anwendung*. Hrsg. von Konigorski, U.; Lohmann, B.; Roppenecker, G.; Trächtler, A. 11. VDE Verlag GmbH, 2013

[Fre07a] Free Software Foundation, I.: *GNU GPL 3.0*. 2007

[Fre07b] Free Software Foundation, I.: *GNU LESSER GENERAL PUBLIC LICENSE*. 2007.

[FSd14a] Fontes, J. V.; Santos, J. C.; da Silva, M. M.: Optimization Strategies for Actuators of Kinematically Redundant Manipulators to Achieve High Dynamic Performance. *2014 Joint Conference on Robotics [and Intelligent Systems]: SBR-LARS Robotics Symposium and Robocontrol (SBR LARS Robocontrol)*. Hrsg. von Santos Osório, F. Piscataway, NJ, 2014, S. 31–36

[FSd14b] Fontes, J. V.; Santos, J. C.; da Silva, M. M.: Torque Optimization of Parallel Manipulators by the Application of Kinematic Redundancy (2014)

[Fun20] FunctionBay GmbH: *RecurDyn dynamische und kinematische Mehrkörpersimulation & Mehrkörpersimulationssoftware / multibody dynamics simulation and cae software: About Us*. 2020. http://www.functionbay.org/ (besucht am 06. 05. 2020)

[GA88] Gosselin, C.; Angeles, J.: The Optimum Kinematic Design of a Planar Three-Degree-of-Freedom Parallel Manipulator. *Journal of Mechanisms Transmissions and Automation in Design* 110 (1988), Nr. 1, S. 35

[GA90] Gosselin, C.; Angeles, J.: Singularity analysis of closed-loop kinematic chains. *IEEE Transactions on Robotics and Automation* 6 (1990), Nr. 3, S. 281–290

[Gal16] Gallardo-Alvarado, J.: *Kinematic Analysis of Parallel Manipulators by Algebraic Screw Theory*. Cham: Springer International Publishing, 2016

[Gev06] Gevatter, H.-J.: *Handbuch der Mess- und Automatisierungstechnik in der Produktion*. 2., vollst. bearb. Aufl. Berlin [u.a.]: Springer, 2006

[GK12] Graichen, K.; Kaepernick, B.: A Real-Time Gradient Method for Nonlinear Model Predictive Control. *Frontiers of Model Predictive Control*. Hrsg. von Tao Zheng. Rijeka: IntechOpen, 2012

[GRN13] Gattringer, H.; Riepl, R.; Neubauer, M.: Optimizing Industrial Robots for Accurate High-Speed Applications. *Journal of Industrial Engineering* 2013 (2013), Nr. 4, S. 1–12

[Har98] Harnisch, H.-G.: Arbeiten mit Polylinien. *AutoCAD-Zeichenkurs*. Hrsg. von Harnisch, H.-G. Viewegs Fachbücher der Technik. Wiesbaden: Vieweg+Teubner Verlag, 1998, S. 129–159

[Hau13] Haun, M.: *Handbuch Robotik: Programmieren und Einsatz intelligeter Roboter*. 2. Aufl. Berlin, Heidelberg: Springer, 2013

[Haw13] Hawkins, K. P.: *Analytic Inverse Kinematics for the Universal Robots UR-5 UR-10 Arms*. Hrsg. von Georgia Institute of Technology. 2013. https://smartech.gatech.edu/bitstream/handle/1853/50782/ur_kin_tech_report_1.pdf (besucht am 28. 02. 2018)

[HBM+13] Huyck, B.; Brabanter, J. de; Moor, B. de; van Impe, J.; Logist, F., Hrsg.: *Model predictive control of a pilot-scale distillation column using a programmable automation controller: Control Conference (ECC), 2013 European*. 2013

[HCL+12] Huyck, B.; Callebaut, L.; Logist, F.; Ferreau, H. J.; Brabanter, J. de; Diehl, M.; van Impe, J.; Moor, B. de: Implementation and Experimental Validation of Classical MPC on Programmable Logic Controllers. *Proceedings of the 20th Mediterranean Conference on Control and Automation, Barcelona, Spain*. 2012, S. 679–684

[HE98] Hebsacker, M.; Epfl, A.: Die Auslegung der Kinematik des Hexaglide: Methodik für die Auslegung paralleler Werkzeugmaschinen. *Neue Maschinenkonzepte mit parallelen Strukturen für Handhabung und Produktion, Tagung Braunschweig, 10. und 11. November*. Hrsg. von VDI. VDI-Berichte. Düsseldorf: VDI Verlag, 1998

[HFD+12] Huyck, B.; Ferreau, H. J.; Diehl, M.; Brabanter, J. de; Van Impe, Jan F. M.; Moor, B. de; Logist, F.: Towards Online Model Predictive Control on a Programmable Logic Controller: Practical Considerations. *Mathematical Problems in Engineering* 2012 (2012), Nr. 1, S. 1–20

[Hil57] Hildreth, C.: A quadratic programming procedure. *Naval Research Logistics Quarterly* 4 (1957), Nr. 1, S. 79–85

[HJ61] Hooke, R.; Jeeves, T. A.: " Direct Search" Solution of Numerical and Statistical Problems. *Journal of the ACM* 8 (1961), Nr. 2, S. 212–229

[HK97] Hirakawa, A. R.; Kawamura, A.: Trajectory planning of redundant manipulators for minimum energy consumption without matrix inversion. *1997 IEEE International Conference on Robotics and Automation*. Piscataway, 1997, S. 2415–2420

[Hor11] Hornig, G.: *Professor Roger Fletcher*. Hrsg. von University of Dundee - Department of Mathematics. 2011. http://www.maths.dundee.ac.uk/~fletcher/ (besucht am 04. 11. 2018)

[HR20] Henke, C.; Rüting, A.: *Robotics Lab: Automatisierungstechnik erforschen und einführen*. 2020. https://www.iem.fraunhofer.de/de/forschung/labore-pruefeinrichtungen/robotics-lab.html (besucht am 13. 05. 2020)

[Huf13] Hufnagel, T. E.: *Theoretische und praktische Entwicklung von Regelungskonzepten für redundant angetriebene parallelkinematische Maschinen: Dissertation*. Duisburg-Essen, 2013.

[HWR08] Haschke, R.; Weitnauer, E.; Ritter, H.: On-line planning of time-optimal, jerk-limited trajectories. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Hrsg. von Chatila, R.; Kelly, A.; Merlet, J.-P. Piscataway NJ, 2008, S. 3248–3253

[Ine19] Ines Stotz. Hrsg. von elektrotechnik AUTOMATISIERUNG. 2019. https://www.elektrotechnik.vogel.de/sps-der-urknall-fuer-die-dritte-und-vierte-industrielle-revolution-a-831646/ (besucht am 30. 01. 2020)

[Int23] International Federation Of Robotics IFR: *World Robotics 2023 Report: Asia ahead of Europe and the Americas: Growth in all regions and major markets*. 2023. https://ifr.org/ifr-press-releases/news/world-robotics-2023-report-asia-ahead-of-europe-and-the-americas#downloads (besucht am 06. 12. 2023)

[Ion03a] Ionescu, T. G.: Standardization of Terminology/German. *Mechanism and machine theory* 38 (2003), Nr. 7–10, S. 969–982

[Ion03b] Ionescu, T. G.: Standardization of Terminology/German 5.1. *Mechanism and machine theory* 38 (2003), Nr. 7–10, S. 1029–1036

[ISG19] ISG Industrielle Steuerungstechnik GmbH, Möller Horcher Public Relations GmbH, Hrsg.: *SPS 2019: ISG präsentiert Simulationssystem ISGvirtuos als offene Plattform für digitale Zwillinge: Anlagensimulation mit herstellerübergreifender Komponentenbibliothek*. 2019. https://www.moeller-horcher.de/wp-content/uploads/2019/08/pm_isg_sps_2019_vfinal.pdf (besucht am 30. 01. 2020)

[ISO12] ISO International Organization for Standardization, Hrsg.: *ISO 8373:2012: Robots and robotic devices - Vocabulary*. Bd. 2012. 2012

[Jäk18] Jäker, K.-P.: *HiL-Achsprüfstand: Demonstrator: HiL-Achsprüfung mit hydraulischem Hexapod*. 2018. https://www.hni.uni-paderborn.de/rtm/forschung/hil-simulation-und-robotik/hil-achspruefstand/ (besucht am 15. 02. 2018)

[Joh16] Johnson, S. G.: *The NLopt nonlinear-optimization packag*. 2016. http://ab-initio.mit.edu/nlopt (besucht am 15. 11. 2016)

[Joh19] Johnson, S. G.: *The NLopt nonlinear-optimization package: NLopt Algorithm-Sbplx*. 2019. https://nlopt.readthedocs.io/en/latest/NLopt_Algorithms/#sbplx-based-on-subplex (besucht am 14. 11. 2019)

[JP11] Jansen, P. W.; Perez, R. E.: Constrained structural design optimization via a parallel augmented Lagrangian particle swarm optimization approach. *Computers & Structures* 89 (2011), Nr. 13-14, S. 1352–1366

[JPS93] Jones, D. R.; Perttunen, C. D.; Stuckman, B. E.: Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications* 79 (1993), Nr. 1, S. 157–181

[Jun04] Jungnickel, U.: *Miniaturisierte Positioniersysteme mit mehreren Freiheitsgraden auf der Basis monolithischer Strukturen*. Darmstadt, 2004

[KAH08] Kotlarski, J.; Abdellatif, H.; Heimann, B.: Improving the pose accuracy of a planar 3RRR parallel manipulator using kinematic redundancy and optimized switching patterns. *IEEE International Conference on Robotics and Automation, 2008*. [Pitscataway, N.J.], 2008, S. 3863–3868

[KAJ+15] Kufoalor, D. K. M.; Aaker, V.; Johansen, T. A.; Imsland, L.; Eikrem, G. O.: Automatically generated embedded model predictive control: Moving an industrial PC-based MPC to an embedded platform. *Optimal Control Applications and Methods* 36 (2015), Nr. 5, S. 705–727

[Kal13] Kallrath, J.: *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*. Wiesbaden: Springer Fachmedien Wiesbaden, 2013

[Kea17] Keating, R.: *UR5 Inverse Kinematics: M.E. 530.646*. 2017. https://de.slideshare.net/RyanKeating13/ur5-ik (besucht am 19. 02. 2019)

[KG13] Kapernick, B.; Graichen, K.: Model predictive control of an overhead crane using constraint substitution. *American Control Conference (ACC), 2013*. Piscataway, NJ, 2013, S. 3973–3978

[KG14a] Kaepernick, B.; Graichen, K., Hrsg.: *PLC Implementation of a Nonlinear Model Predictive Controller*. 2014

[KG14b] Kaepernick, B.; Graichen, K.: Nichtlineare modellprädiktive Regelung auf SPS. *atp edition - Automatisierungstechnische Praxis* 56 (2014), Nr. 03, S. 38

[KHO11] Kotlarski, J.; Heimann, B.; Ortmaier, T.: Experimental validation of the influence of kinematic redundancy on the pose accuracy of parallel kinematic machines. *IEEE International Conference on Robotics and Automation (ICRA)*. 2011, S. 1923–1929

[Kin09] King, D. E.: Dlib-ml: A Machine Learning Toolkit. *J. Mach. Learn. Res.* 10 (2009), S. 1755–1758

[KK97] Kuntsevich, A.; Kappel, F.: *SolvOpt The Solver For Local Nonlinear Optimization Problems: Matlab, C and Fortran Source Codes*. 1997.

[Koc98] Kock, S.: Regelungsstrategien für Parallelroboter mit redundanten Antrieben. *Neue Maschinenkonzepte mit parallelen Strukturen für Handhabung und Produktion, Tagung Braunschweig, 10. und 11. November*. Hrsg. von VDI. Bd. VDI Berichte 1427. VDI-Berichte. Düsseldorf: VDI Verlag, 1998, S. 155–164

[KOF+16] Kohlstedt, A.; Olma, S.; Flottmeier, S.; Traphöner, P.; Jäker, K.-P.; Trächtler, A.: Control of a hydraulic hexapod for a Hardware-in-the-Loop axle test rig. *at - Automatisierungstechnik* 64 (2016), Nr. 5

[Kol12] Kolovsky, M. Z.: *Advanced theory of mechanisms and machines*. Berlin: Springer-Verlag, 2012

[Kra88] Kraft, D.: *A software package for sequential quadratic programming*. 1988

[KRI+14] Kufoalor, D. K. M.; Richter, S.; Imsland, L.; Johansen, T. A.; Morari, M.; Eikrem, G. O.: Embedded Model Predictive Control on a PLC using a primal-dual first-order method for a subsea separation process. *22nd Mediterranean Conference on Control and Automation*. 2014, S. 368–373

[KTHO10] Kotlarski, J.; Thanh, T. D.; Heimann, B.; Ortmaier, T.: Optimization strategies for additional actuators of kinematically redundant parallel kinematic machines. *IEEE International Conference on Robotics and Automation (ICRA 2010)*. 2010, S. 656–661

[KTW06] Kröger, T.; Tomiczek, A.; Wahl, F.: Towards On-Line Trajectory Computation. *IROS 2006*. [Piscataway, N.J.], 2006, S. 736–741

[LLL+20] Lipinski, K.; Lackner, H.; Laudé, O. P.; Kafka, G., Niemann, A.; Raasch, E.; Schoon, B.; Radonic, A.: *Brute-Force-Angriff*. Hrsg. von DATACOM Buchverlag GmbH. 2020. `https://www.itwissen.info/Brute-Force-Angriff-brute-force-attack.html` (besucht am 18. 03. 2020)

[LLS10] Liuzzi, G.; Lucidi, S.; Sciandrone, M.: Sequential Penalty Derivative-Free Methods for Nonlinear Constrained Optimization. *SIAM Journal on Optimization* 20 (2010), Nr. 5, S. 2614–2635

[LN89] Liu, D. C.; Nocedal, J.: On the limited memory BFGS method for large scale optimization. *Mathematical Programming* 45 (1989), Nr. 1-3, S. 503–528

[LRDW12] Lara-Molina, F. A.; Rosario, J. M.; Dumur, D.; Wenger, P.: Application of predictive control techniques within parallel robot. *Sba: Controle & Automação Sociedade Brasileira de Automatic* 23 (2012), Nr. 5, S. 530–540

[LT96] Lawrence, C. T.; Tits, A. L.: Nonlinear equality constraints in feasible sequential quadratic programming ∗. *Optimization Methods and Software* 6 (1996), Nr. 4, S. 265–282

[LY09] Li, C.; Yang, S.: An adaptive learning particle swarm optimizer for function optimization. *2009 IEEE Congress on Evolutionary Computation: CEC : Trondheim, Norway, 18-21 May 2009*. Piscataway, N.J., 2009, S. 381–388

[MAM+17] Mohammadi, A.; Asadi, H.; Mohamed, S.; Nelson, K.; Nahavandi, S.: OpenGA, a C++ Genetic Algorithm Library. *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. Piscataway, NJ, 2017, S. 2051–2056

[Mar05] Martínez, J. M.: *TANGO Trustable Algorithms for Nonlinear General Optimization*. 2005. https://www.ime.usp.br/~egbirgin/tango/index.php (besucht am 04. 11. 2018)

[Mat20] MathWorks: *Optimization Toolbox: Solve linear, quadratic, integer, and nonlinear optimization problems*. 2020. https://de.mathworks.com/help/optim/ (besucht am 11. 05. 2020)

[Mer06] Merlet, J.-P.: *Parallel robots*. 2nd ed. Bd. 74. Solid mechanics and its applications. Dordrecht und Boston, MA: Kluwer Academic Publishers, 2006

[Mer07] Merlet, J.-P.: Jacobian, Manipulability, Condition Number and Accuracy of Parallel Robots. *Robotics research*. Hrsg. von Thrun, S.; Brooks, R. A.; Durrant-Whyte, H. F. Bd. 28. Springer Tracts in Advanced Robotics. Berlin: Springer, 2007, S. 175–184

[MH12] Müller, A.; Hufnagel, T.: Model-based control of redundantly actuated parallel manipulators in redundant coordinates. *Robotics and Autonomous Systems* 60 (2012), Nr. 4, S. 563–571

[Mil13] Milutinović, M.: Kinematic modelling of hybrid parallel-serial five-axis machine tool. *FME Transactions* 41 (2013), S. 1–10

[MLD09] Mahnke, W.; Leitner, S.-H.; Damm, M.: *OPC unified architecture*. Berlin und Heidelberg: Springer-Verlag, 2009

[MM11] Mrosko, M.; Miklovičová, E.: Real-time model predictive control. *Proceedings of the 13th WSEAS international conference on Mathematical and computational methods in science and engineering*. 2011, S. 138–143

[MM12] Mrosko, M.; Miklovičová, E.: Real-time implementation of predictive control using programmable logic controllers. *International Journal of Systems Apllication, Engineering & Development*. Bd. Issue 1, Volume 6. 2012

[Moz12] Mozilla Foundation: *Mozilla Public License Version 2.0*. 2012. https://www.mozilla.org/en-US/MPL/2.0/

[Moz18] Mozilla Corporation: *MPL 2.0 FAQ*. 2018. https://www.mozilla.org/en-US/MPL/2.0/FAQ/ (besucht am 30. 10. 2018)

[MSS00] McAvoy, B.; Sangolola, B.; Szabad, Z.: Optimal trajectory generation for redundant planar manipulators. *Cybernetics evolving to systems, humans, organizations, and their complex interactions*. Piscataway, NJ, 2000, S. 3241–3246

[Mül08] Müller, A.: Redundant Actuation of Parallel Manipulators. *Parallel manipulators, towards new applications*. Hrsg. von Wu, H. Vienna, Austria: I-Tech, 2008

[Mül19] Müller, C.: *IFR Press Conference*. 2019. https://ifr.org/downloads/press2018/IFR%20World%20Robotics%20Presentation%20-%2018%20Sept%202019.pdf (besucht am 09. 01. 2020)

[MW04] Ma, S.; Watanabe, M.: Time Optimal Path-Tracking Control of Kinematically Redundant Manipulators. *JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing* 47 (2004), Nr. 2, S. 582–590

[Nes04] Nesterov, Y.: *Introductory Lectures on Convex Optimization: A Basic Course*. Bd. 87. Applied Optimization. Boston, MA und s.l.: Springer US, 2004

[Neu06] Neugebauer, R.: *Parallelkinematische Maschinen: Entwurf, Konstruktion, Anwendung*. VDI-Buch. Berlin: Springer, 2006

[Nik01] Niku, S. B.: *Introduction to robotics analysis, systems, applications*. Upper Saddle River: Prentice Hall, 2001

[NKOM13] Niemann, S.; Kotlarski, J.; Ortmaier, T.; Müller-Schloer, C.: Reducing the optimization problem for the efficient motion planning of kinematically redundant parallel robots. *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. 2013, S. 618–624

[NM65] Nelder, J. A.; Mead, R.: A Simplex Method for Function Minimization. *The Computer Journal* 7 (1965), Nr. 4, S. 308–313

[NSS+02] Neugebauer, R.; Schwaar, M.; St. Ihlenfeldt; Pritschow, G.; Eppler, C.; Garber, T.: New Approaches to Machine Structures to Overcome the Limits of Classical Parallel Structures. *CIRP Annals - Manufacturing Technology* 51 (2002), Nr. 1, S. 293–296

[NW06] Nocedal, J.; Wright, S. J.: *Numerical optimization*. 2nd ed. Springer series in operations research and financial engineering. New York, N.Y.: Springer, 2006

[ope18] openMDAO: *Publications*. 2018. http://openmdao.org/publications/ (besucht am 04. 11. 2018)

[ope19a] open62541: *open62541*. 2019. https://open62541.org/ (besucht am 05. 09. 2019)

[ope19b] open62541: *open62541 Documentation: Release 1.0.0-dev-2-g3ec05f52: The open62541 authors*. 2019. https://open62541.org/doc/open62541-current.pdf (besucht am 05. 09. 2019)

[PBP02] Prautzsch, H.; Boehm, W.; Paluszny, M.: *Bézier and B-spline techniques*. Mathematics and visualization. Berlin und New York: Springer, 2002

[PC11] Pock, T.; Chambolle, A.: Diagonal preconditioning for first order primal-dual algorithms in convex optimization. *IEEE International Conference on Computer Vision (ICCV), 2011*. Piscataway, NJ, 2011, S. 1762–1769

[PD12] Pandilov, Z.; Dukovski, V.: Parallel Kinematics Machine Tools: Overview-From History To The Future. *Annals of the Faculty of Engineering Hunedoara*. Hunedoara: University Politehnica Timisoara Faculty of Engineering Hunedoara, 2012

[PJM12] Perez, R. E.; Jansen, P. W.; Martins, J. R. R. A.: pyOpt: a Python-based object-oriented framework for nonlinear constrained optimization. *Structural and Multidisciplinary Optimization* 45 (2012), Nr. 1, S. 101–118

[PKM18] PKM TRICEPT: *Tricept T606*. 2018. `http://www.pkmtricept.com/productos/index.php?id=en&Nproduct=1240238156`

[PLA15] Pereira, M.; Limon, D.; Alamo, T.: MPC implementation in a PLC based on Nesterov's fast gradient method. *23rd Mediterranean Conference on Control and Automation (MED)* (2015)

[Pow02] Powell, M. J. D.: UOBYQA: unconstrained optimization by quadratic approximation. *Mathematical Programming* 92 (2002), Nr. 3, S. 555–582

[Pow04] Powell, M. J. D.: The NEWUOA software for unconstrained optimization without derivatives 1. 2004

[Pow07] Powel, M. J. D.: *A view of algorithms for optimization without derivatives*. 2007

[Pow09] Powel, M. J. D.: *The BOBYQA algorithm for bound constrained optimization without derivatives*. 2009

[Pow15] Powell, M. J. D.: On fast trust region methods for quadratic models with linear constraints. *Mathematical Programming Computation* 7 (2015), Nr. 3, S. 237–267

[Pow94] Powell, M. J. D.: A Direct Search Optimization Method that Models the Objective and Constraint Functions by Linear Interpolation. *Advances in Optimization and Numerical Analysis*. Hrsg. von Gomez, S.; Hennart, J.-P. Mathematics and Its Applications. Dordrecht: Springer Netherlands, 1994, S. 51–69

[Ras19] Raspberry Pi (Trading) Ltd.: *Raspberry Pi 4 Model B: Release 1*. 2019. `https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2711/rpi_DATA_2711_1p0_preliminary.pdf` (besucht am 12. 11. 2019)

[Ras20] Raspberry Pi Foundation: *Download Raspbian for Raspberry Pi*. 2020. `https://www.raspberrypi.org/downloads/raspbian/` (besucht am 11. 05. 2020)

[RB05] Raghunathan, A. U.; Biegler, L. T.: An Interior Point Method for Mathematical Programs with Complementarity Constraints (MPCCs). *SIAM Journal on Optimization* 15 (2005), Nr. 3, S. 720–750

[RBT16] Rüting, A.; Blumenthal, L.; Trächtler, A.: Model Predictive Feedforward Compensation for Control of Multi Axes Hybrid Kinematics on PLC. *IECON*. 2016, S. 583–588

[RBT17] Rüting, A.; Block, E.; Trächtler, A.: Modellprädiktive Vorsteuerung für einen kinematisch redundanten hybridkinematischen Mechanismus im Industrieumfeld. *Fachtagung Mechatronik 2017*. Hrsg. von Bertram, T.; Corves, B.; Janschek, K. 2017

[Rei12] Reinhardt, H.-J.: *Numerik gewöhnlicher Differentialgleichungen: Anfangs- und Randwertprobleme*. 2. Auflage. Berlin, Boston: De Gruyter, 2012

[Rei13] Reichenbach, A., Hrsg.: *Koordination und Erlernen von komplexen Bewegungen mit redundanten Gliedmassen*. 2013. `https://www.alexreichenbach.de/doktorarbeit_David.html` (besucht am 01. 02. 2020)

[RGM18] Reiter, A.; Gattringer, H.; Müller, A.: On Redundancy Resolution in Minimum-Time Trajectory Planning of Robotic Manipulators Along Predefined End-Effector Paths. *Informatics in control*. Hrsg. von Madani, P. e. a. 2018, S. 190–206

[RHT18] Rüting, A.; Henke, C.; Trächtler, A.: Umsetzung einer echtzeitfähigen Mehrgrößenoptimierung auf einer Industriesteuerung. *Entwurf komplexer Automatisierungssysteme*. Hrsg. von Jumar, U.; Diedrich, C. 2018

[RHT19] Rüting, A.; Henke, C.; Trächtler, A.: Umsetzung einer echtzeitfähigen modellprädiktiven Trajektorienplanung für eine mehrachsige Hybridkinematik auf einer Industriesteuerung. *at - Automatisierungstechnik* 67 (2019), Nr. 4, S. 326–336

[RHWT15] Rüting, A.; Henke, C.; Warkentin, A.; Trächtler, A.: Entwurf eines Tripod-basierten Inspektionssystems - Vom virtuellen Prototypen zum Vorseriensystem. *10. Paderborner Workshop Entwurf Mechatronischer Systeme*. Bd. 343. 2015, S. 113–126

[RIS13] Rymansaib, Z.; Iravani, P.; Sahinkaya, M. N., Hrsg.: *Exponential trajectory generation for point to point motions*. IEEE, 2013

[RK 20] RK Rose+Krieger GmbH: *Lineareinheiten / Linearführungen*. 2020. `https://www.rk-rose-krieger.com/deutsch/produkte/linear-technik/produkt-kenner/lineareinheiten/` (besucht am 11. 05. 2020)

[RMG16] Reiter, A.; Müller, A.; Gattringer, H.: Inverse Kinematics in Minimum-Time Trajectory Planning for Kinematically Redundant Manipulators. *Proceedings of the IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*. 2016

[RMG18] Reiter, A.; Müller, A.; Gattringer, H.: On Higher Order Inverse Kinematics Methods in Time-Optimal Trajectory Planning for Kinematically Redundant Manipulators. *IEEE Transactions on Industrial Informatics* 14 (2018), Nr. 4, S. 1681–1690

[RO09] Richalet, J.; O'Donovan, D.: *Predictive Functional Control: Principles and Industrial Applications*. Advances in industrial control. London: Springer London, 2009

[Rös11] Röse, A.: *Parallelkinematische Mechanismen zum intrakorporalen Einsatz in der laparoskopischen Chirurgie*. Mikro- und Sensortechnik. Inst. für Elektromechanische Konstruktionen, 2011

[Row90] Rowan, T. H.: *Functional Stability Analysis Of Numerical Algorithms*. 1990

[RPW16] Reveles, R. D.; Pamanes, A.; Wenger, p.: Trajectory planning of kinematically redundant parallel manipulators by using multiple working modes. *Mechanism and machine theory* 98 (2016), S. 216–230

[RSGM15] Reiter, A.; Springer, K.; Gattringer, H.; Müller, A.: An explicit approach for time-optimal trajectory planning for kinematically redundant serial robots. *PAMM* 15 (2015), Nr. 1, S. 67–68

[RVKF11] Rauova, I.; Valo, R.; Kvasnica, M.; Fikar, M., Hrsg.: *Real-Time Model Predictive Control of a Fan Heater via PLC*. Bratislava, 2011

[RY00] Runarsson, T. P.; Yao, X.: Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation* 4 (2000), Nr. 3, S. 284–294

[RY05] Runarsson, T. P.; Yao, X.: Search Biases in Constrained Evolutionary Optimization. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)* 35 (2005), Nr. 2, S. 233–243

[SB00] Stoer, J.; Bulirsch, R.: *Numerische Mathematik II - Eine Einführung*. 4. ed. Springer-Lehrbuch. Heidelberg: Springer-Verlag Berlin, 2000

[SB02] Stoer, J.; Bulirsch, R.: *Introduction to numerical analysis*. 3. ed. Bd. 12. Texts in applied mathematics. New York, NY [u.a.]: Springer, 2002

[Sch04] Schreiber, G.: *Steuerung für redundante Robotersysteme: Benutzer- und aufgabenorientierte Verwendung der Redundanz*. 2004

[Sch06] Schaal, S.: Dynamic Movement Primitives -A Framework for Motor Control in Humans and Humanoid Robotics. *Adaptive Motion of Animals and Machines*. Hrsg. von Kimura, H.; Tsuchiya, K.; Ishiguro, A.; Witte, H. Tokyo: Springer Tokyo, 2006, S. 261–280

[Sch16] Schlosser, C.: *Modellierung einer mobilen Hybridkinematik und Umsetzung am Beispiel der Flugzeuginspektion: Dissertation*. 2016

[SCH19a] SCHUNK GmbH & Co. KG: *EGS*. 2019. `https://schunk.com/nl_de/greifsysteme/series/egs/` (besucht am 07. 12. 2019)

[SCH19b] SCHUNK GmbH & Co. KG: *Produktinformation: Greif-Schwenk-Modul mit Parallelgreifer EGS*. 2019. `https://schunk.com/fileadmin/pim/docs/IM0020998.PDF` (besucht am 24. 09. 2019)

[SEB09] Schlüter, M.; Egea, J. A.; Banga, J. R.: Extended ant colony optimization for non-convex mixed integer nonlinear programming. *Computers & Operations Research* 36 (2009), Nr. 7, S. 2217–2229

[SGS14] Springer, K.; Gattringer, H.; Stoger, C.: A real-time nearly time-optimal point-to-point trajectory planning method using dynamic movement primitives. *International Conference on Robotics in Alpe-Adria-Danube Region (RAAD)*. 2014, S. 1–6

[Sha17] Sharma, K.: *Overview of Industrial Process Automation*. second edition. Saint Louis: Elsevier Science, 2017

[She74] Shere, K. D.: Remark on algorithm 454 [E4]: The complex method for constrained optimization. *Communications of the ACM* 17 (1974), Nr. 8, S. 471

[Sic99] Siciliano, B.: The Tricept robot: Inverse kinematics, manipulability analysis and closed-loop direct kinematics algorithm. *Robotica* 17 (1999), Nr. 4, S. 437–445

[SL12] Schättler, H. M.; Ledzewicz, U., Hrsg.: *Geometric optimal control: Theory, methods and examples*. Bd. 38. Interdisciplinary Applied Mathematics. New York, NY: Springer, 2012

[SRD14] Santos, J.; Rocha, M.; Da Silva, M.: Performance Evaluation of kinematically redundant planar parallel Manipulators. *ABCM Symposium Series in Mechatronics*. Bd. 6. 2014, S. 391–401

[SRHT19] Schütz, S.; Rüting, A.; Henke, C.; Trächtler, A.: Regelung kollaborativer Robotersysteme zur benutzerfreundlichen, flexiblen Fertigung kleiner Losgrößen am Beispiel eines halbautomatischen Schweißvorgangs. *Fachtagung Mechatronik*. 2019

[SRHT20] Schütz, S.; Rüting, A.; Henke, C.; Trächtler, A.: Echtzeitfähige Planung optimierter Trajektorien für sensorgeführte, kinematisch redundante Mechanismen auf einer Industriesteuerung. *EKA 2020 - Entwurf komplexer Automatisierungssysteme*. Hrsg. von Jumar, U.; Diedrich, C. 2020

[SRHT21] Schütz, S.; Rüting, A.; Henke, C.; Trächtler, A.: Echtzeitfähige Planung optimierter Trajektorien für sensorgeführte, kinematisch redundante Mechanismen auf einer Industriesteuerung. *at - Automatisierungstechnik* 69 (2021), Nr. 3, S. 231–241

[SS11] Syaichu-Rohman, A.; Sirius, R.: Model predictive control implementation on a programmable logic controller for DC motor speed control. *Proceedings of the 2011 International Conference on Electrical Engineering and Informatics* (2011)

[SSVO09] Siciliano, B.; Scavicco, L.; Villa, L.; Oriolo, G., Hrsg.: *Robotics: Modelling, Planning and Control*. London: Springer London, 2009

[Sta09] Stark, G.: *Robotik mit MATLAB: Mit 40 Beispielen, 55 Aufgaben und 37 Listings (Lehrbücher zur Informatik)*. 1. Aufl. Lehrbücher zur Informatik. München: Fachbuchverl. Leipzig im Carl-Hanser-Verl., 2009

[Sva02] Svanberg, K.: A Class of Globally Convergent Optimization Methods Based on Conservative Convex Separable Approximations. *SIAM Journal on Optimization* 12 (2002), Nr. 2, S. 555–573

[Sva14] Svanberg, K.: *MMA and GCMMA – two methods for nonlinear optimization*. 2014. `https://people.kth.se/~krille/mmagcmma.pdf` (besucht am 30. 10. 2018)

[Sva87] Svanberg, K.: The method of moving asymptotes—a new method for structural optimization. *International Journal for Numerical Methods in Engineering* 24 (1987), Nr. 2, S. 359–373

[Sva95] Svanberg, K.: A Globally Convergent Version of MMA without Linesearch. *Proceedings of the First World Congress of Structural and Multidisciplinary Optimization*. Hrsg. von Olhoff, N.; Rozvany, G. I. N. Oxford: Pergamon, 1995, S. 9–16

[SW07] Schneider, U.; Werner, D., Hrsg.: *Taschenbuch der Informatik: Mit 108 Tabellen*. 6., neu bearb. Aufl. München: Fachbuchverl. Leipzig im Carl-Hanser-Verl., 2007

[SW93] Seereeram, S.; Wen, J. T.: A global approach to path planning for redundant manipulators. *IEEE International Conference on Robotics and Automation 1993*. Los Alamitos, Calif., 1993, S. 283–288

[SW98] Schönherr, J.; Weidermann, F.: Bewertung und optimale Auslegung von Bewegungssystemen mit Parallelkinematik. *Neue Maschinenkonzepte mit parallelen Strukturen für Handhabung und Produktion, Tagung Braunschweig, 10. und 11. November*. Hrsg. von VDI. VDI-Berichte. Düsseldorf: VDI Verlag, 1998, S. 35–50

[TGG98] Tönshoff, H.; Günther, G.; Grendel, H.: Vergleichende Betrachtung paralleler und hybrider Strukturen. *Neue Maschinenkonzepte mit parallelen Strukturen für Handhabung und Produktion, Tagung Braunschweig, 10. und 11. November*. Hrsg. von VDI. VDI-Berichte. Düsseldorf: VDI Verlag, 1998, S. 249–270

[The17] The Eclipse Foundation: *Eclipse Public License - v 2.0*. 2017. https://www.eclipse.org/org/documents/epl-2.0/EPL-2.0.pdf (besucht am 04. 11. 2018)

[Tho88] Thornton, G. S.: The GEC Tetrabot-a new serial-parallel assembly robot. *Proceedings: 1988 IEEE International Conference on Robotics and Automation : April 24-29, 1988, Philadelphia, Pennsylvania*. Washington, DC, 1988, S. 437–439

[TMH98] Treib, T.; Meier, P.; Hebsacker, M.: Wachstumsgesetzmaessigkeiten und Einsatzpotentiale parallel-kinematischer Manipulatore. *Neue Maschinenkonzepte mit parallelen Strukturen für Handhabung und Produktion, Tagung Braunschweig, 10. und 11. November*. Hrsg. von VDI. VDI-Berichte. Düsseldorf: VDI Verlag, 1998

[Uni15] Universal Robots: *Parameters for calculations of kinematics and dynamics - 45257*. 2015. https://www.universal-robots.com/how-tos-and-faqs/faq/ur-faq/parameters-for-calculations-of-kinematics-and-dynamics-45257/ (besucht am 14. 02. 2019)

[Uni16] Universak Robots: *Robot working area PDF - Drawing no. 100400*. Hrsg. von Universal Robots. 2016. https://s3-eu-west-1.amazonaws.com/ur-support-site/16350/100400.PDF (besucht am 17. 12. 2019)

[Uni19] Universal Robots: *Technische Daten UR10*. Hrsg. von Universal Robots. 2019. https://www.universal-robots.com/media/1801313/ger_199917_ur10__tech_spec_web_a4.pdf (besucht am 12. 11. 2019)

[VA07] Voigt, C.; Adamy, J.: *Formelsammlung der Matrizenrechnung*. München [u.a.]: Oldenbourg, 2007

[Van73] Vanderplaats, G. N.: *CONMIN, a FORTRAN program for constrained function minimization : user's manual*. Moffett Field, Calif. und Springfield, Va.: Ames Research Center. and U.S. Army Air Mobility Research and Development Laboratory., 1973

[Van83] Vanderplaats, G. N.: A robust Feasible Directions algorithm for design synthesis. *24th Structures, Structural Dynamics and Materials Conference*. Reston, Virigina, 1983, S. 739

[vDS09] van den Broeck, L.; Diehl, M.; Swevers, J.: Time optimal MPC for mechatronic applications. *48th IEEE Conference on Decision and Control (CDC) and 28th Chinese Control Conference (CCC)*. 2009, S. 8040–8045

[vDS11a] van den Broeck, L.; Diehl, M.; Swevers, J.: A model predictive control approach for time optimal point-to-point motion control. *Mechatronics* 21 (2011), Nr. 7, S. 1203–1212

[vDS11b] van den Broeck, L.; Diehl, M.; Swevers, J.: Experimental validation of Time Optimal MPC on a flexible motion system. *American Control Conference (ACC)*. Piscataway, NJ, 2011, S. 4749–4754

[VHR09] Valencia-Palomo, G.; Hilton, K.; Rossiter, J. A., Hrsg.: *Predictive control implementation in a PLC using the IEC 1131.3 programming standard*. 2009

[VS14] Varalakshmi, K. V.; Srinivas, J.: Optimized Configurations of Kinematically Redundant Planar Parallel Manipulator following a Desired Trajectory. *Procedia Technology* 14 (2014), S. 133–140

[VŠ14] Velagić, J.; Šabić, B.: Design, implementation and experimental validation of explicit MPC in programmable logic controller. *IEEE 23rd International Symposium on Industrial Electronics (ISIE)* (2014)

[VS15] Varalakshmi, K. V.; Srinivas, J.: Optimum Design of Kinematically Redundant Planar Parallel Manipulator Following a Trajectory. *International Journal of Materials, Mechanics and Manufacturing* 3 (2015), Nr. 2, S. 74–79

[Wam87] Wampler, C.: Inverse kinematic functions for redundant manipulators. *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*. 1987, S. 610–617

[WB05] Wörn, H.; Brinkschulte, U.: *Echtzeitsysteme: Grundlagen, Funktionsweisen, Anwendungen ; mit 32 Tabellen*. eXamen.press. Berlin: Springer, 2005

[WB06] Wächter, A.; Biegler, L. T.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming* 106 (2006), Nr. 1, S. 25–57

[WCR93] Wenger, p.; Chedmail, P.; Reynier, F.: A global analysis of following trajectories by redundant manipulators in the presence of obstacles. *IEEE International Conference on Robotics and Automation 1993*. Los Alamitos, Calif., 1993, S. 901–906

[Web07] Weber, R.: *Physik: Teil I: Klassische Physik - Experimentelle und theoretische Grundlagen*. Wiesbaden: B. G. Teubner Verlag / GWV Fachverlage GmbH Wiesbaden, 2007

[Web09] Weber, W.: *Industrieroboter*. München: Carl Hanser Verlag GmbH & Co. KG, 2009

[Wre89] Wrenn, G. A.: *An indirect method for numerical optimization using the Kreisselmeier-Steinhauser function*. NASA contractor report. National Aeronautics and Space Administration, Office of Management, Scientific and Technical Information Division, 1989

[WS15] Wang, X.; Swevers, J.: Offset-free energy-optimal model predictive control for point-to-point motions with high positioning accuracy. *Proceedings, 2015 IEEE International Conference on Mechatronics (ICM)*. Piscataway, NJ, 2015, S. 64–69

[WSSP12] Wang, X.; Swevers, J.; Stoev, J.; Pinte, G.: Energy optimal Point-to-point motion using model predictive control. *5th Annual Dynamic Systems and Control Conference ASME*. 2012

[Zha17] Zhang, Z.: *Software*. 2017. `http://mat.uc.pt/~zhang/software.html` (besucht am 04. 11. 2018)

[ZQC+16] Zhou, H.; Qin, Y.; Chen, H.; Ge, S.; Cao, Y.: Structural synthesis of five-degree-of-freedom hybrid kinematics mechanism. *Journal of Engineering Design* 27 (2016), Nr. 4-6, S. 390–412

[ZRA15] Zöller, D.; Reiter, M.; Abel, D.: Optimization of a vacuum thermal evaporation process through Model-based Predictive Control of the source temperature. *IFAC-PapersOnLine* 48 (2015), Nr. 11, S. 86–91

[ZRRJ11] Zanchettin, A.; Rocco, P.; Robertsson, A.; Johansson, R.: Exploiting task redundancy in industrial manipulators during drilling operations. *Proceedings - IEEE International Conference on Robotics and Automation*. 2011

Literaturverzeichnis der studentischen Arbeiten

Die nachstehend aufgeführten studentischen Arbeiten wurden im Kontext der vorliegenden Dissertation am Lehrstuhl für Regelungstechnik und Mechatronik der Universität Paderborn angefertigt. Die Definition der Zielsetzung, die Bearbeitung sowie die Auswertung, Interpretation und Visualisierung von Ergebnissen erfolgten unter wissenschaftlicher Anleitung der jeweils angegebenen Betreuenden. Die erzielten Ergebnisse sind zum Teil in die Dissertation eingeflossen.

[Abd17] Abdelsabour, H.; Trächtler, A. (BetreuerIn); Rüting, A. (BetreuerIn); Biemelt, P. (BetreuerIn): *Analysis on the Implementability of Real-Time Capable Optimization Algorithms on Real-Time Systems and Industrial Controllers*. unveröffentlichte Masterarbeit. Paderborn: Universität Paderborn, 2017

[Blu16] Blumenthal, L. M.; Trächtler, A. (BetreuerIn); Henke, C. (BetreuerIn); Rüting, A. (BetreuerIn): *Modellierung einer hybriden Mehrachskinematik mittels Mehrkörpersimulation zur Ermittlung der kinematischen Gleichungen*. unveröffentlichte Masterarbeit. Paderborn: Universität Paderborn, 2016

[Cra18] Cramer, D. J.; Trächtler, A. (BetreuerIn); Henke, C. (BetreuerIn); Rüting, A. (BetreuerIn): *Analyse und Bewertung von Ansätzen zur Ermittlung zeitoptimaler Punkt-zu-Punkt-Bewegungen bei hybriden Kinematiken*. unveröffentlichte Bachelorarbeit. Paderborn: Universität Paderborn, 2018

[Ker18] Kersting, N.; Trächtler, A. (BetreuerIn); Henke, C. (BetreuerIn); Rüting, A. (BetreuerIn): *Entwicklung einer modellbasierten Mehrzieloptimierung zur Reduzierung der Verfahrzeiten eines redundant seriellen Mechanismus*. unveröffentlichte Bachelorarbeit. Paderborn: Universität Paderborn, 2018

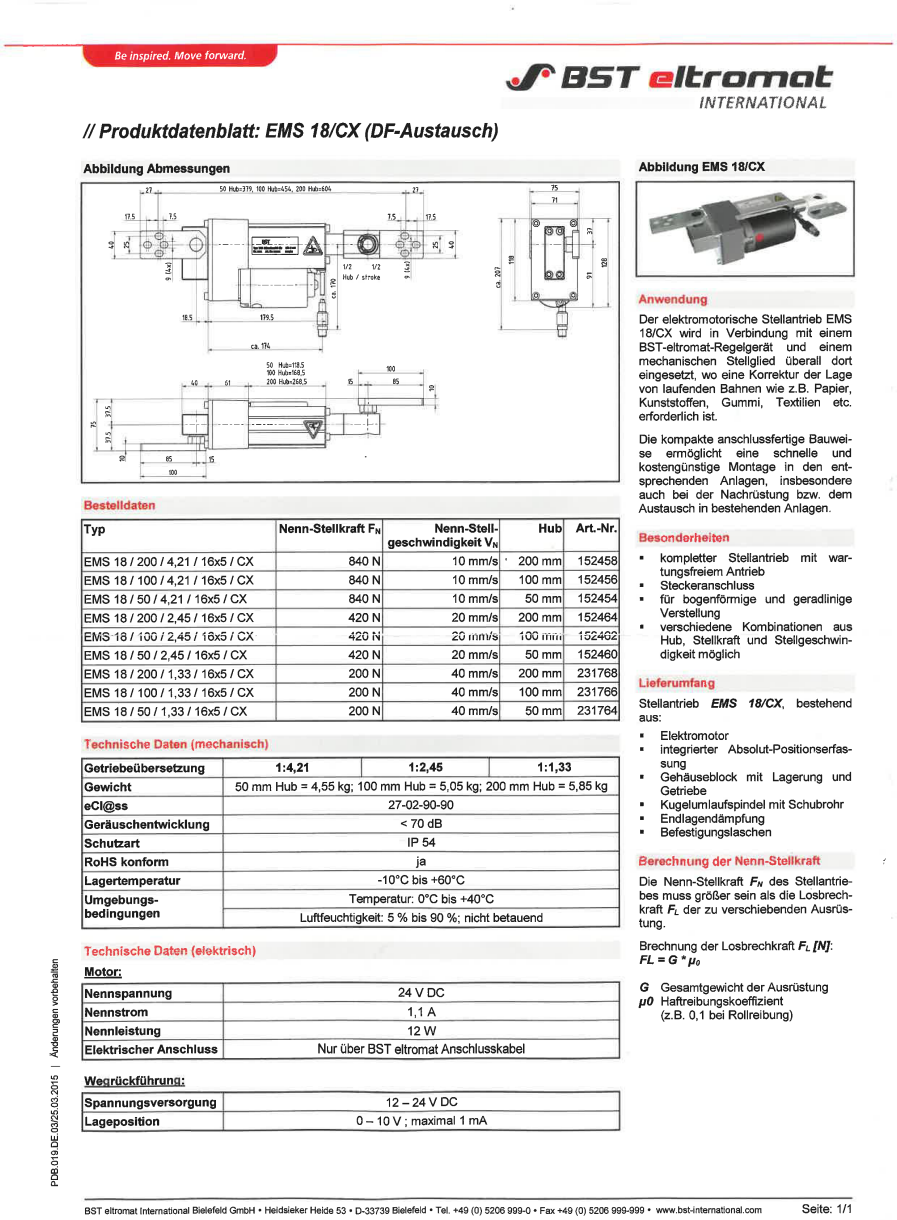
Anhang

Inhaltsverzeichnis

A1 Datenblätter **169**
A1.1 Parallele Antriebe $M_1 \ldots M_3$ des Tricept 169
A1.2 Serieller Antrieb M_4 des Tricept 172
A1.3 Antriebe M_5 und M_6 des Portals 174
A1.4 Getriebe der Antriebe M_4, M_5 und M_6 177
A1.5 Universal Robots UR10 180
A1.6 Raspberry Pi 4B 181
A1.7 SPS B&R X20CP1382-RT 184
A1.8 SPS Beckhoff C6030-0060 189

A1 Datenblätter

A1.1 Parallele Antriebe $M_1 \ldots M_3$ des Tricept



Be inspired. Move forward.

BST eltromat INTERNATIONAL

// Produktdatenblatt: EMS 18/CX (DF-Austausch)

Abbildung Abmessungen

Abbildung EMS 18/CX

Bestelldaten

Typ	Nenn-Stellkraft F_N	Nenn-Stell-geschwindigkeit V_N	Hub	Art.-Nr.
EMS 18 / 200 / 4,21 / 16x5 / CX	840 N	10 mm/s	200 mm	152458
EMS 18 / 100 / 4,21 / 16x5 / CX	840 N	10 mm/s	100 mm	152456
EMS 18 / 50 / 4,21 / 16x5 / CX	840 N	10 mm/s	50 mm	152454
EMS 18 / 200 / 2,45 / 16x5 / CX	420 N	20 mm/s	200 mm	152464
EMS 18 / 100 / 2,45 / 16x5 / CX	420 N	20 mm/s	100 mm	152462
EMS 18 / 50 / 2,45 / 16x5 / CX	420 N	20 mm/s	50 mm	152460
EMS 18 / 200 / 1,33 / 16x5 / CX	200 N	40 mm/s	200 mm	231768
EMS 18 / 100 / 1,33 / 16x5 / CX	200 N	40 mm/s	100 mm	231766
EMS 18 / 50 / 1,33 / 16x5 / CX	200 N	40 mm/s	50 mm	231764

Technische Daten (mechanisch)

Getriebeübersetzung	1:4,21	1:2,45	1:1,33
Gewicht	50 mm Hub = 4,55 kg; 100 mm Hub = 5,05 kg; 200 mm Hub = 5,85 kg		
eCl@ss	27-02-90-90		
Geräuschentwicklung	< 70 dB		
Schutzart	IP 54		
RoHS konform	ja		
Lagertemperatur	-10°C bis +60°C		
Umgebungs-bedingungen	Temperatur: 0°C bis +40°C		
	Luftfeuchtigkeit: 5 % bis 90 %; nicht betauend		

Technische Daten (elektrisch)

Motor:

Nennspannung	24 V DC
Nennstrom	1,1 A
Nennleistung	12 W
Elektrischer Anschluss	Nur über BST eltromat Anschlusskabel

Wegrückführung:

Spannungsversorgung	12 – 24 V DC
Lageposition	0 – 10 V ; maximal 1 mA

Anwendung

Der elektromotorische Stellantrieb EMS 18/CX wird in Verbindung mit einem BST-eltromat-Regelgerät und einem mechanischen Stellglied überall dort eingesetzt, wo eine Korrektur der Lage von laufenden Bahnen wie z.B. Papier, Kunststoffen, Gummi, Textilien etc. erforderlich ist.

Die kompakte anschlussfertige Bauweise ermöglicht eine schnelle und kostengünstige Montage in den entsprechenden Anlagen, insbesondere auch bei der Nachrüstung bzw. dem Austausch in bestehenden Anlagen.

Besonderheiten

- kompletter Stellantrieb mit wartungsfreiem Antrieb
- Steckeranschluss
- für bogenförmige und geradlinige Verstellung
- verschiedene Kombinationen aus Hub, Stellkraft und Stellgeschwindigkeit möglich

Lieferumfang

Stellantrieb ***EMS 18/CX***, bestehend aus:

- Elektromotor
- integrierter Absolut-Positionserfassung
- Gehäuseblock mit Lagerung und Getriebe
- Kugelumlaufspindel mit Schubrohr
- Endlagendämpfung
- Befestigungslaschen

Berechnung der Nenn-Stellkraft

Die Nenn-Stellkraft F_N des Stellantriebes muss größer sein als die Losbrechkraft F_L der zu verschiebenden Ausrüstung.

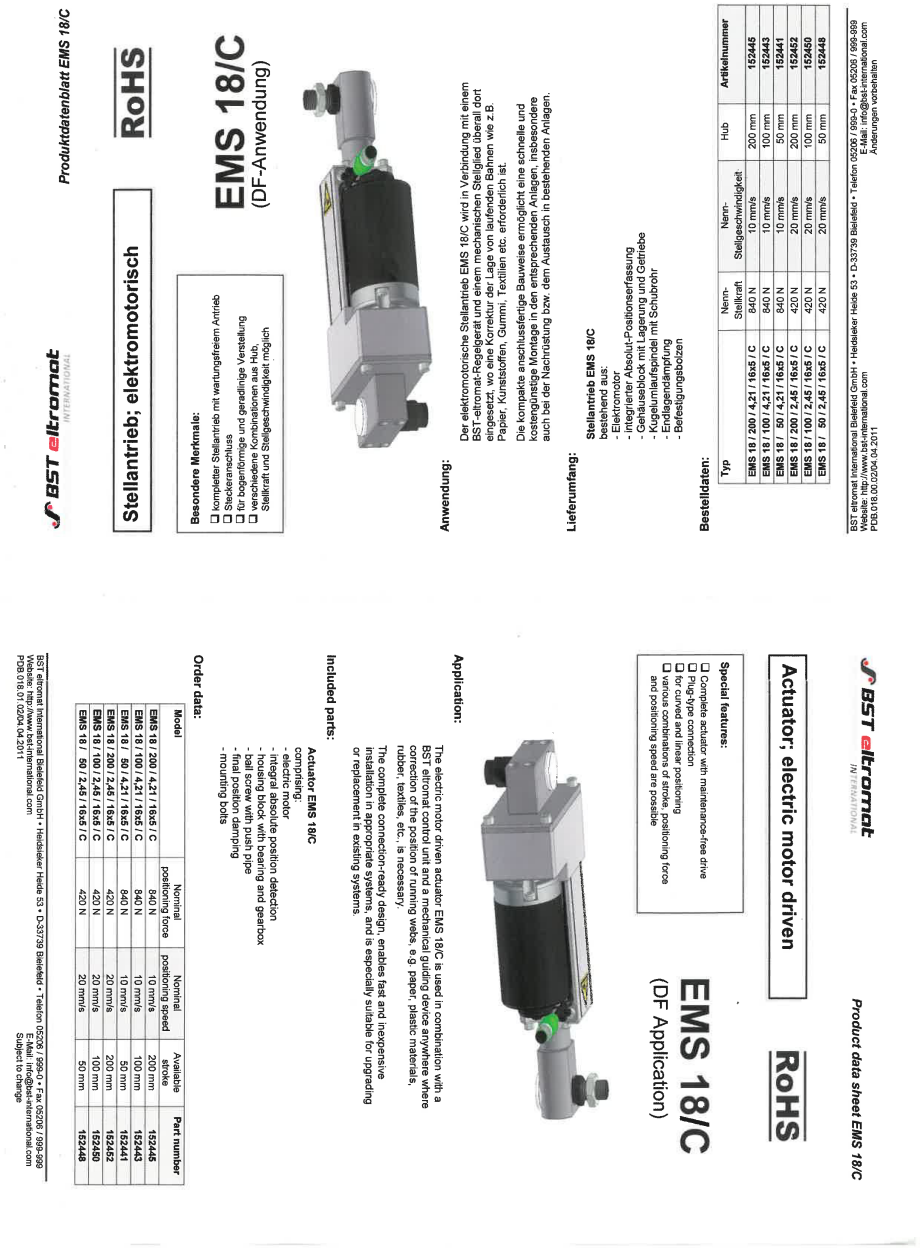
Brechnung der Losbrechkraft F_L *[N]*:
$FL = G * \mu_0$

G Gesamtgewicht der Ausrüstung
$\mu 0$ Haftreibungskoeffizient (z.B. 0,1 bei Rollreibung)

PDB.019.DE.03/25.03.2015 | Änderungen vorbehalten

BST eltromat International Bielefeld GmbH • Heidsieker Heide 53 • D-33739 Bielefeld • Tel. +49 (0) 5206 999-0 • Fax +49 (0) 5206 999-999 • www.bst-international.com Seite: 1/1

Bild A1-1: Datenblatt der Antriebe $M_1 \ldots M_3$ (EMS 18/100/2,45/16-5/CX der Fa. BST eltromat), Seite 1 [BST16]



BST eltromat INTERNATIONAL

Produktdatenblatt EMS 18/C

Stellantrieb; elektromotorisch

RoHS

EMS 18/C
(DF-Anwendung)

Besondere Merkmale:
- ☐ kompletter Stellantrieb mit wartungsfreiem Antrieb
- ☐ Steckeranschluss
- ☐ für bogenförmige und geradlinige Verstellung
- ☐ verschiedene Kombinationen aus Hub, Stellkraft und Stellgeschwindigkeit möglich

Anwendung:

Der elektromotorische Stellantrieb EMS 18/C wird in Verbindung mit einem BST-eltromat-Regelgerät und einem mechanischen Stellglied überall dort eingesetzt, wo eine Korrektur der Lage von laufenden Bahnen wie z.B. Papier, Kunststoffen, Gummi, Textilien etc. erforderlich ist.

Die kompakte anschlussfertige Bauweise ermöglicht eine schnelle und kostengünstige Montage in den entsprechenden Anlagen, insbesondere auch bei der Nachrüstung bzw. dem Austausch in bestehenden Anlagen.

Lieferumfang:

Stellantrieb EMS 18/C
bestehend aus:
- Elektromotor
- integrierter Absolut-Positionserfassung
- Gehäuseblock mit Lagerung und Getriebe
- Kugellaufspindel mit Schubrohr
- Endlagendämpfung
- Befestigungsblöcken

Bestelldaten:

Typ	Nenn-Stellkraft	Nenn-Stellgeschwindigkeit	Hub	Artikelnummer
EMS 18 / 200 / 4,21 / 16x5 / C	840 N	10 mm/s	200 mm	**152445**
EMS 18 / 100 / 4,21 / 16x5 / C	840 N	10 mm/s	100 mm	**152443**
EMS 18 / 50 / 4,21 / 16x5 / C	840 N	10 mm/s	50 mm	**152441**
EMS 18 / 200 / 2,45 / 16x5 / C	420 N	20 mm/s	200 mm	**152452**
EMS 18 / 100 / 2,45 / 16x5 / C	420 N	20 mm/s	100 mm	**152450**
EMS 18 / 50 / 2,45 / 16x5 / C	420 N	20 mm/s	50 mm	**152448**

BST eltromat International Bielefeld GmbH • Heidekiener Heide 53 • D-33739 Bielefeld • Telefon 05206 / 999-0 • Fax 05206 / 999-999
Website: http://www.bst-international.com
E-Mail: info@bst-international.com
PDB.018.00.0204.04.2011

BST eltromat INTERNATIONAL

Product data sheet EMS 18/C

Actuator; electric motor driven

RoHS

EMS 18/C
(DF Application)

Special features:
- ☐ Complete actuator with maintenance-free drive
- ☐ Plug-type connection
- ☐ For curved and straight positioning
- ☐ various combinations of stroke, positioning force and pod position possible

Application:

The electric motor driven actuator EMS 18/C is used in combination with a BST-eltromat control unit and mechanical guiding device anywhere where correction of the position of running webs, e.g. paper, plastic materials, rubber, textiles, etc., is necessary.

The complete connection-ready design, enables fast and inexpensive installation in appropriate systems, and is especially suitable for upgrading or replacement in existing systems.

Included parts:

Actuator EMS 18/C
comprising:
- a.c. motor
- integral absolute position detection
- housing block with bearing and gearbox
- ball screw with push pipe
- final position dampning
- mounting bolts

Order data:

Model	Nominal positioning force	Nominal positioning speed	Available stroke	Part number
EMS 18 / 200 / 4,21 / 16x5 / C	840 N	10 mm/s	200 mm	**152445**
EMS 18 / 100 / 4,21 / 16x5 / C	840 N	10 mm/s	100 mm	**152443**
EMS 18 / 50 / 4,21 / 16x5 / C	840 N	10 mm/s	50 mm	**152441**
EMS 18 / 200 / 2,45 / 16x5 / C	420 N	20 mm/s	200 mm	**152452**
EMS 18 / 100 / 2,45 / 16x5 / C	420 N	20 mm/s	100 mm	**152450**
EMS 18 / 50 / 2,45 / 16x5 / C	420 N	20 mm/s	50 mm	**152448**

BST eltromat International Bielefeld GmbH • Heidekiener Heide 53 • D-33739 Bielefeld • Telefon 05206 / 999-0 • Fax 05206 / 999-999
E-Mail: info@bst-international.com
Website: http://www.bst-international.com
PDB.018.01.0204.04.2011

Bild A1-2: Datenblatt der Antriebe $M_1 \ldots M_3$ (EMS 18/100/2,45/16-5/CX der Fa. BST eltromat), Seite 2 [BST16]

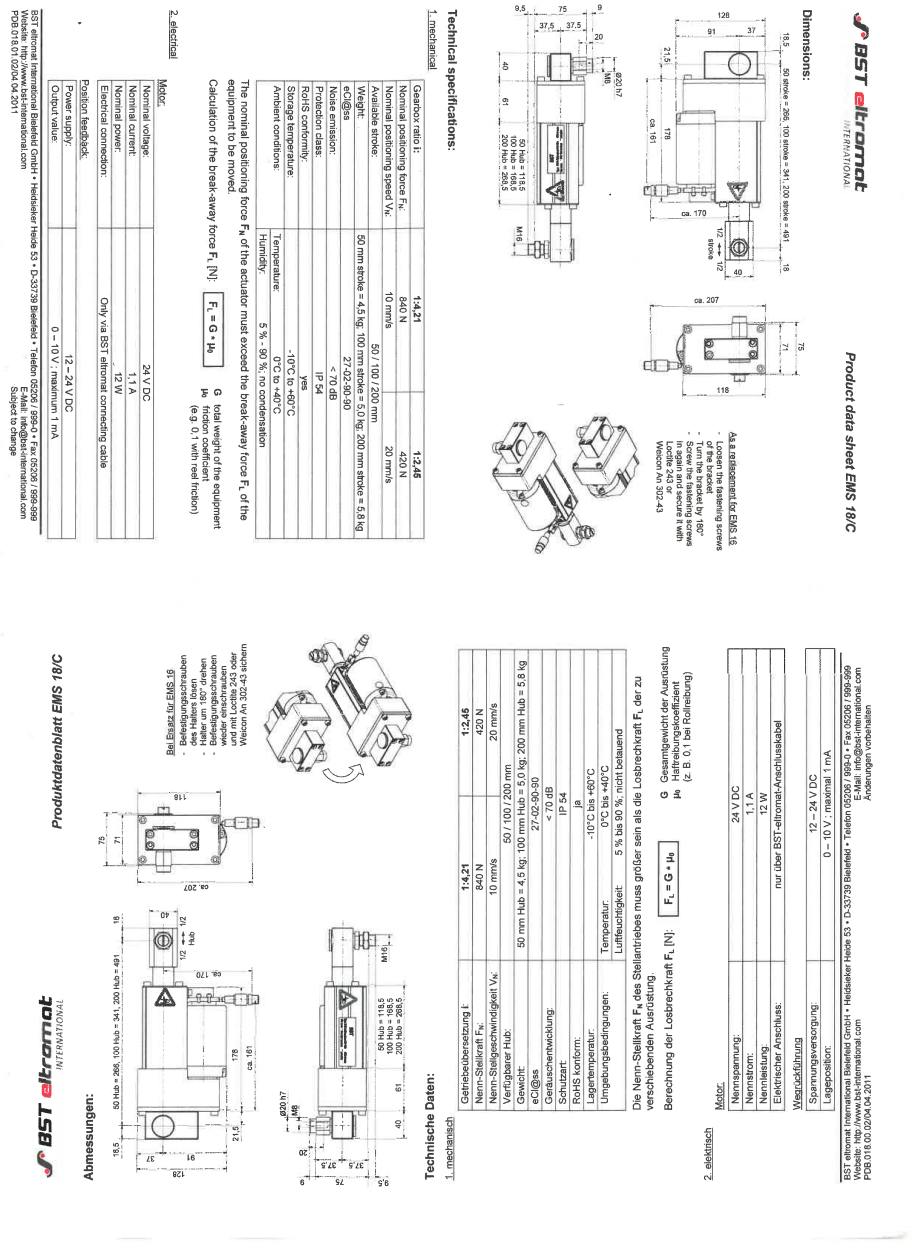


Bild A1-3: Datenblatt der Antriebe $M_1 \ldots M_3$ (EMS 18/100/2,45/16-5/CX der Fa. BST eltromat), Seite 3 [BST16]

A1.2 Serieller Antrieb M_4 des Tricept

BECKHOFF Technische Daten

10.3 AM813x

Elektrische Daten		Symbol [Einheit]	AM8131-F	AM8131-F an EL7201	AM8131-J	AM8132-J
	Stillstandsdrehmoment	M_0 [Nm]	1,35	0,80	1,35	2,37
	Stillstandsstrom	I_{0rms} [A]	5,0	2,8	8,0	8,0
	Max. mech. Drehzahl	N_{max} [min^{-1}]	10000	10000	10000	10000
	Max. Netzspannung	U_N [V_{DC}]	50	50	50	50
U_N = 24 VDC	Nenndrehzahl	N_n [min^{-1}]	300	300	600	300
	Nenndrehmoment	M_n [Nm]	1,35	0,80	1,35	2,36
	Nennleistung	P_n [W]	28	25	94	74
U_N = 48 VDC	Nenndrehzahl	N_n [min^{-1}]	1000	1000	1800	1000
	Nenndrehmoment	M_n [Nm]	1,35	0,80	1,34	2,35
	Nennleistung	P_n [W]	140	84	253	246
	Spitzenstrom	I_{0max} [A]	27,8	5,66	44,7	44,3
	Spitzendrehmoment	M_{0max} [Nm]	6,07	1,76	6,07	11,7
	Drehmomentkonstante	K_{Trms} [Nm/A]	0,27	0,28	0,169	0,296
	Spannungskonstante	K_{Erms} [mVmin]	19	19	11,8	21
	Wicklungswiderstand	R20 [Ω]	1,95	1,95	0,73	0,96
	Wicklungsinduktivität	L [mH]	5,3	5,3	2,05	3,4

* Bemessungsflansch Aluminium 230 mm x 130 mm x 10 mm

Einbau eines Wellendichtrings führt zu einer Reduktion der Nenndaten.

Mechanische Daten	Symbol [Einheit]	AM8131	AM8132
Rotorträgheitsmoment (ohne Bremse)	J [kgcm^2]	0,462	0,842
Rotorträgheitsmoment (mit Bremse)	J [kgcm^2]	0,541	0,921
Polzahl		8	8
Statisches Reibmoment	M_R [Nm]	0,009	0,009
Thermische Zeitkonstante	t_{TH} [min]	24	24
Gewicht (ohne Bremse)	G [kg]	1,80	2,4
Gewicht (mit Bremse)	G [kg]	2,20	2,8
Zulässige Radialkraft am Wellenende	F_R [N]	Siehe 10.3.2	
Zulässige Axialkraft	F_A [N]		

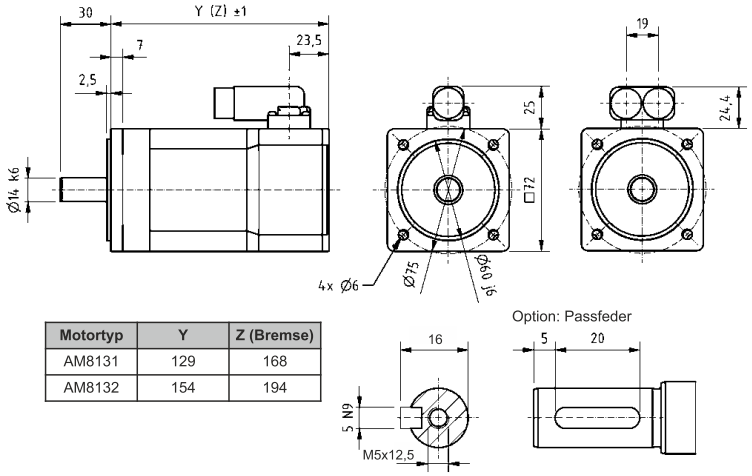
Daten der optionalen Bremse

Daten	Symbol [Einheit]	AM813x
Haltemoment bei 120°C	M_{BR} [Nm]	2,0
Anschlussspannung	U_{BR} [V_{DC}]	24 +6 -10 %
Elektrische Leistung	P_{BR} [W]	11
Strom	I_{on} [A]	0,33
Lüftverzögerungszeit	t_{BRH} [ms]	25
Einfallverzögerungszeit	t_{BRL} [ms]	8

Synchron - Servomotoren AM8100 Version: 2.0 37

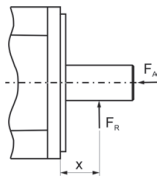
Bild A1-4: Datenblatt des Antriebs M_4 (AM8131-0F21-0000 der Fa. Beckhoff), Seite 1 ([Bec18b], S. 37)

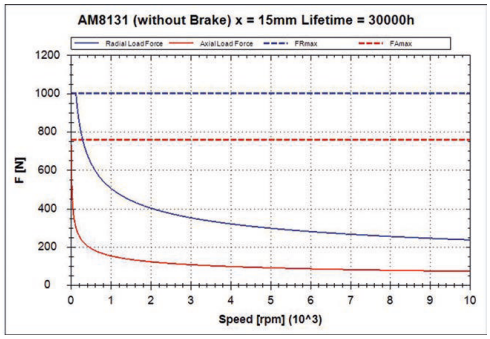
10.3.1 Maßzeichnung AM813x



Motortyp	Y	Z (Bremse)
AM8131	129	168
AM8132	154	194

10.3.2 Radial / Axialkräfte am Wellenende





10.3.3 Drehmoment- / Drehzahlkennlinien

Drehmoment- / Drehzahlkennlinien finden Sie auf der Beckhoff-Homepage unter Motion.

Bild A1-5: Datenblatt des Antriebs M_4 (AM8131-0F21-0000 der Fa. Beckhoff), Seite 2 ([Bec18b], S. 38)

A1.3 Antriebe M_5 und M_6 des Portals

Technische Daten

AM803x & AM853x

Elektrische Daten	AM80xx / AM85xx								
	31C	31D	31F	32D	32E	32H	33E	33F	33J
Stillstandsdrehmoment* M_0 [Nm]	1,37	1,38	1,40	2,38	2,37	2,37	3,20	3,22	3,22
Stillstandsstrom I_{0rms} [A]	1,00	1,95	3,20	1,70	2,95	5,10	2,10	4,10	6,80
Maximale mechanische Drehzahl N_{max} [min^{-1}]	10000								
Maximale Netz-Nennspannung U_N [V_{AC}]	480								
Spitzenstrom I_{0max} [A]	5,5	10,7	17,6	9,6	17,2	29,5	12,9	24,6	39,8
Spitzendrehmoment M_{0max} [Nm]	6,1	6,07	6,07	11,66	11,66	11,65	17,19	17,71	17,22
Drehmomentkonstante $K_{T rms}$ [Nm/A]	1,37	0,71	0,44	1,4	0,8	0,46	1,52	0,78	0,47
Spannungskonstante $K_{E rms}$ [mVmin]	99	50	30	100	56	32	106	57	34
Wicklungswiderstand Ph-Ph R_{20} [Ω]	51,0	12,6	5,0	21,0	6,5	2,2	13,2	3,9	1,35
Wicklungsinduktivität Ph-Ph, gemessen bei 1 kHz L [mH]	134	36	13,3	71,9	22,6	7,7	46,3	14	4,9
Spannungsversorgung U_N= 115 V									
Nenndrehzahl Nn [min-1]	400	1400	2700	600	1400	2700	600	1400	2700
Nenndrehmoment* M_n [Nm]	1,36	1,38	1,37	2,37	2,34	2,29	3,15	3,10	3,05
Nennleistung P_n [kW]	0,06	0,20	0,39	0,15	0,34	0,65	0,20	0,45	0,86
Spannungsversorgung U_N= 230 V									
Nenndrehzahl Nn [min-1]	1400	3300	6000	1500	3000	6000	1500	3000	5900
Nenndrehmoment* M_n [Nm]	1,35	1,36	1,34	2,34	2,30	2,10	3,10	3,00	2,70
Nennleistung P_n [kW]	0,20	0,47	0,84	0,37	0,76	1,32	0,49	1,00	1,67
Spannungsversorgung U_N= 400 V									
Nenndrehzahl Nn [min-1]	3000	6000	9000	3000	6000	9000	3000	6000	9000
Nenndrehmoment* M_n [Nm]	1,34	1,33	1,30	2,30	2,20	1,85	2,98	2,70	2,30
Nennleistung P_n [kW]	0,42	0,84	1,23	0,72	1,38	1,74	0,94	1,70	2,17
Nennstrom I_n [A]	0,95	1,90	3,00	1,60	2,75	4,10	2,00	3,60	5,10
Spannungsversorgung U_N= 480 V									
Nenndrehzahl Nn [min-1]	3400	6800	9000	3400	6800	9000	3400	6800	9000
Nenndrehmoment* M_n [Nm]	1,33	1,32	1,3	2,26	2,1	1,85	2,95	2,6	2,3
Nennleistung P_n [kW]	0,47	0,94	1,23	0,8	1,5	1,74	1,05	1,85	2,17
Anschlusstechnik	iTec								
** Bemessungsflansch Aluminium 230 mm x 130 mm x 10 mm*									

Mechanische Daten	AM80xx / AM85xx					
	AM8031	AM8531	AM8032	AM8532	AM8033	AM8533
Rotorträgheitsmoment ohne Bremse J [kgcm²]	0,467	1,67	0,847	2,05	1,23	2,440
Rotorträgheitsmoment mit Bremse J [kgcm²]	0,546	1,76	0,926	2,15	1,46	---
Polzahl	8	8	8	8	8	8
Statisches Reibmoment M_R [Nm]	0,009	0,009	0,015	0,015	0,020	0,020
Thermische Zeitkonstante t_{TH} [min]	24	24	26	26	28	28
Gewicht ohne Bremse [kg]	1,80	2,40	2,40	3,00	3,00	3,60
Gewicht mit Bremse [kg]	2,20	2,60	2,80	3,30	3,60	---
Flansch	**IEC-Norm / DIN 42955**					
Passung	J6					
Toleranzklasse	N					
Schutzart						
Gehäuse Standardausführung	IP65					
Wellendurchführung Standardausführung	IP54					
Wellendurchführung mit Wellendichtring	IP65					
Lackfarben						
Eigenschaften	Acryl-pulverbeschichtet					
Farbton	dunkelgrau / RAL 7016					

BECKHOFF Version: 4.1 AM8000 & AM8500 25

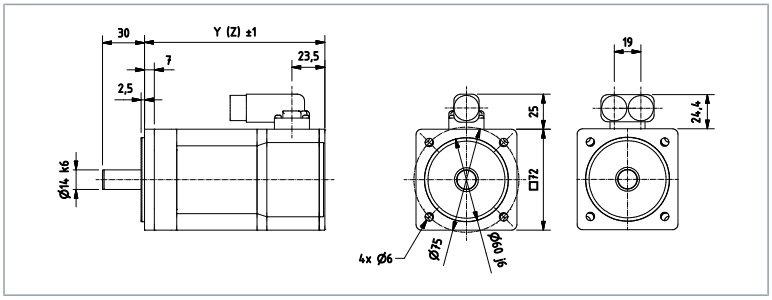
Bild A1-6: Datenblatt der Antriebe M_5 und M_6 (AM8031-0D21-0000 der Fa. Beckhoff), Seite 1 ([Bec19a], S. 25)

Technische Daten

Optionale Haltebremse [+]	AM8031	AM8531	AM8032	AM8532	AM8033
Haltemoment bei 120 °C M_{BR} [Nm]	2,0	2,0	2,0	2,0	3,5
Anschlussspannung U_{BR} [V_{DC}]	24; +6 % bis -10 %				
Elektrische Leistung P_{BR} [W]	11	11	11	11	12
Strom I_{on} [A]	0,33	0,33	0,33	0,33	0,36
Lüftverzögerungszeit t_{BRH} [ms]	25	25	25	25	35
Einfallverzögerungszeit t_{BRL} [ms]	8	8	8	8	15

Maßzeichnung

- Alle Angaben in Millimetern



Motor	Y	Z - Bremse
AM8031	129	168
AM8032	154	194
AM8033	180	229
AM8531	168	194
AM8532	194	229
AM8533	229	--

Passfeder [+]

- Zentrierbohrung gemäß DIN 332-D

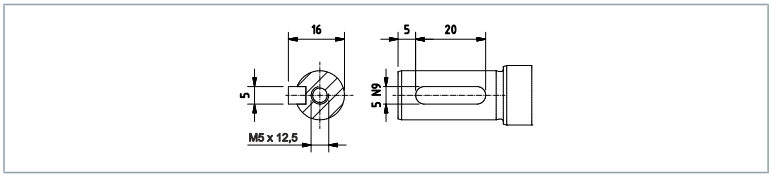


Bild A1-7: Datenblatt der Antriebe M_5 und M_6 (AM8031-0D21-0000 der Fa. Beckhoff), Seite 2 ([Bec19a], S. 26)

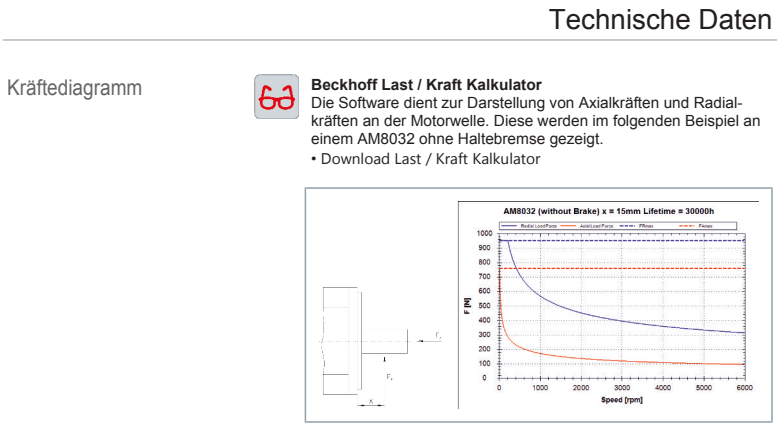
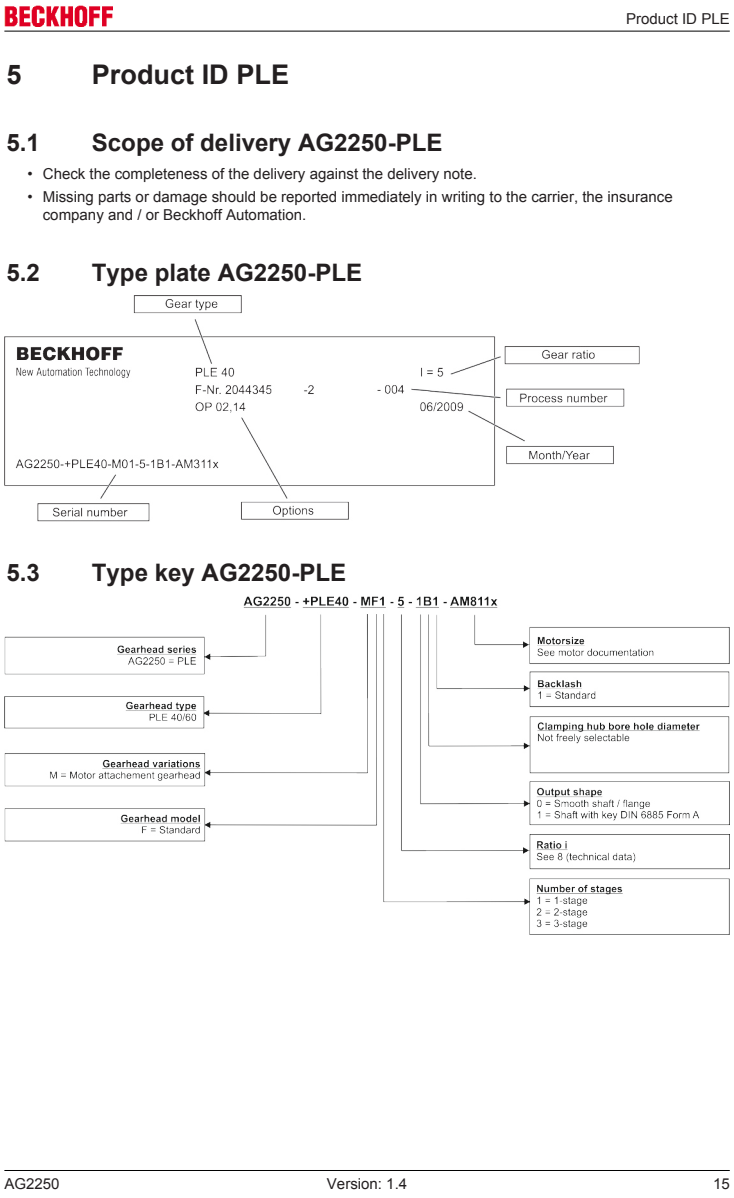


Bild A1-8: Datenblatt der Antriebe M_5 und M_6 (AM8031-0D21-0000 der Fa. Beckhoff), Seite 3 ([Bec19a], S. 27)

A1.4 Getriebe der Antriebe M_4, M_5 und M_6



BECKHOFF Product ID PLE

5 Product ID PLE

5.1 Scope of delivery AG2250-PLE

- Check the completeness of the delivery against the delivery note.
- Missing parts or damage should be reported immediately in writing to the carrier, the insurance company and / or Beckhoff Automation.

5.2 Type plate AG2250-PLE

5.3 Type key AG2250-PLE

AG2250 Version: 1.4 15

Bild A1-9: Datenblatt der Getriebe für die Antriebe M_4 sowie M_5 und M_6 der Fa. Beckhoff, Seite 1 ([Bec17], S. 15)

8.2 PLE60

PLE 60			1-stage						2-stage									3-stage								
Ratio	**i**		**3**	**4**	**5**	**7**	**8**	**10**	**9**	**12**	**15**	**16**	**20**	**25**	**32**	**40**	**64**	**60**	**80**	**100**	**120**	**160**	**200**	**256**	**320**	**512**
Max. output torque (1)(2)(3)(4)	$T2_{Max}$	Nm	45	61	64	40	29	24	70	70	70	70	70	64	70	64	29	70	70	70	70	70	64	70	64	29
Nominal output torque (1)(2)(3)	$T2_N$	Nm	28	38	40	25	18	15	44	44	44	44	44	40	44	40	18	44	44	44	44	44	40	44	40	18
Emergency stop torque allowed 1000 times	$T2_{Not}$	Nm	2 x $T2_n$																							
Max. middle input speed allowed operating temperature must be kept; other input speed on inquiry	n_{1N}	min^{-1}	4.500																							
Max. input speed allowed operating temperature must be kept; other input speed on inquiry	n_{1Max}	min^{-1}	13.000																							
No-load running torque at 20 °C and 3.000 min^{-1}	T0	Nm	0,15	0,00	0,15	0,10	0,10	0,10	0,15	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10
Backlash	j_t	arcmin	< 10						< 12									< 15								
Torsional stiffness		Nm/arcmin	2,3						2,5									2,5								
Max. axial force for 20.000 h / max. axial force for 30.000 h (1)(5)	FA	N	500 / 450																							
Max. radial force for 20.000 h / max. radial force for 30.000 h (1)(5)	FR	N	400 / 340																							
Efficiency with full load depends on ratio, n2 = 100 min^{-1}	η	%	96						94									90								
Lifetime	L_h	h	30.000																							
Weight	m	kg	0,90						1,10									1,30								
Running noise (6)	L_{PA}	dB(A)	58																							
Max. operating temp. refering of the middle of the body surface		°C	+90																							
Min. operating temp. refering of the middle of the body surface		°C	-25																							
Lubrication			Life lubrication																							
Degree of protection			IP 54																							
Inertia (7)		$kgcm^2$	0,135	0,093	0,078	0,072	0,065	0,064	0,131	0,127	0,077	0,088	0,075	0,075	0,064	0,064	0,064	0,076	0,075	0,075	0,064	0,064	0,064	0,064	0,064	0,064

(1) Values refer to a speed of the output shaft n2=100 min-1 and S1-Mode for electrical machines and T = 30 °C

(2) depends on the motor shaft diameter

(3) with key, at tumscent load

(4) allowable for 30000 revolutions at the output shaft

(5) half way along the output shaft

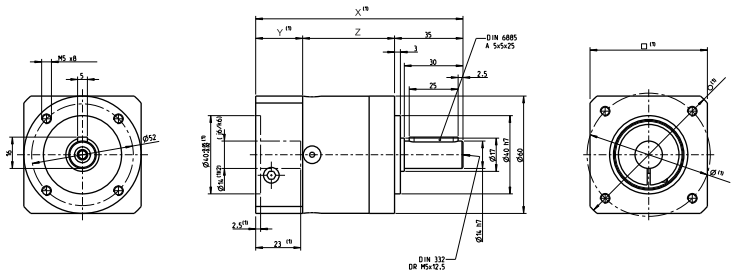
(6) Sound pressure level; distance 1 m; measured on idle running with an input speed n_1 = 3000 min^{-1}

(7) relates to the input shaft and to standard motor shaft diameter from 9 mm

Bild A1-10: *Datenblatt der Getriebe für die Antriebe M_4 (AG2250-+PLE60-M01-3-1C1-AM813x) sowie M_5 und M_6 (AG2250-+PLE60-M02-25-1C1-AM813x) der Fa. Beckhoff, Seite 2 ([Bec17], S. 22)*

8.2.1 Dimensional drawing, PLE60

Power takeoff side Drive side



Size	Unit	PLE60
D30 motor schaft diameter[1)3)]	mm	6/6,35/8/9/9,525/ 10/11/12/14/16/19
L30 min. motor shaft length[1)]	mm	13 (16[4)])

	X[1)]	Y[1)]	Z[1)]
1-stepped	106,00 mm	24,00 mm	47,00 mm
2-stepped	118,50 mm	24,00 mm	59,50 mm
3-stepped	131,00 mm	24,00 mm	72,00 mm

Materials:
Input flange: aluminium - untreated
Housing: steel - black
Output flange: aluminium - untreated
Modification reserved!

(1) dimensions depend on the motor
(2) refered to horizontal and stationary mounting
(3) shaft fit: j6; k6
(4) D30 > 9 mm
(5) D30 > 14 mm
(6) D30 > 19 mm
(7) D30 > 24 mm
(8) D30 > 35 mm

Bild A1-11: Datenblatt der Getriebe für die Antriebe M_4 (AG2250-+PLE60-M01-3-1C1-AM813x) sowie M_5 und M_6 (AG2250-+PLE60-M02-25-1C1-AM813x) der Fa. Beckhoff, Seite 3 ([Bec17], S. 23)

A1.5 Universal Robots UR10

UNIVERSAL ROBOTS — Technische Daten

UR10

Leistung

Wiederholgenauigkeit	±0,1 mm / ±0,0039 in (4 mils)
Umgebungstemperaturbereich	0–50°
Stromverbrauch	Min. 90 W, typisch 250 W, max. 500 W
Kollaborationsbetrieb	15 erweiterte Sicherheitsfunktionen. Vom TüV NORD genehmigte Sicherheitsfunktion Test in Übereinstimmung mit: EN ISO 13849:2008 PL d

Spezifikation

Traglast	10 kg
Reichweite	1300 mm
Freiheitsgrade	6 rotierende Gelenke
Programmierung	Polyscope grafische Benutzerschnittstelle auf 12" Touchscreen mit Halterung

Bewegungen

Achsbewegung, Roboterarm	Arbeitsradius	Max. Geschwindigkeit
Fuß	± 360°	± 120°/Sek.
Schulter	± 360°	± 120°/Sek.
Ellenbogen	± 360°	± 180°/Sek.
Gelenk 1	± 360°	± 180°/Sek.
Gelenk 2	± 360°	± 180°/Sek.
Gelenk 3	± 360°	± 180°/Sek.
Typisches Werkzeug		1 m/Sek.

Eigenschaften

IP-Klassifikation	IP54	
ISO Reinraum Klassifizierung	5	
Lärmbelastung	72dB	
Roboterbefestigung	Jede	
I/O-Anschlüsse	Digital ein	2
	Digital aus	2
	Analog ein	2
	Analog aus	0
I/O-Stromversorgung im Werkzeug	12 V/24 V 600 mA in Werkzeug	

Technische Daten

Grundfläche	Ø 190mm
Material	Aluminium, PP-Kunststoff
Werkzeugverbindung, Typ	M8
Kabellänge, Roboterarm	6 m
Gewicht einschl. Kabel	28,9 kg

SCHALTKASTEN

Eigenschaften

IP-Klassifikation	IP20	
ISO Reinraum Klassifizierung	6	
Lärmbelastung	<65dB(A)	
I/O-Anschlüsse	Digital ein	16
	Digital aus	16
	Analog ein	2
	Analog aus	2
I/O-Stromversorgung	24V 2A	
Kommunikation	TCP/IP 100 Mbit, Modbus TCP, Profinet, EthernetIP	
Stromquelle	100-240 VAC, 50-60 Hz	
Umgebungstemperaturbereich	0–50°	

Technische Daten

Maße Schaltkasten	475 mm x 423 mm x 268 mm
Gewicht	17 kg
Material	Stahl

TEACH PANEL

Eigenschaften

IP-Klassifikation	IP20

Technische Daten

Material	Aluminium, PP
Gewicht	1,5 kg
Kabellänge	4,5 m

Bild A1-12: Datenblatt des Universal Robots UR10 [Uni19]

A1.6 Raspberry Pi 4B

2 Features

2.1 Hardware

- Quad core 64-bit ARM-Cortex A72 running at 1.5GHz
- 1, 2 and 4 Gigabyte LPDDR4 RAM options
- H.265 (HEVC) hardware decode (up to 4Kp60)
- H.264 hardware decode (up to 1080p60)
- VideoCore VI 3D Graphics
- Supports dual HDMI display output up to 4Kp60

2.2 Interfaces

- 802.11 b/g/n/ac Wireless LAN
- Bluetooth 5.0 with BLE
- 1x SD Card
- 2x micro-HDMI ports supporting dual displays up to 4Kp60 resolution
- 2x USB2 ports
- 2x USB3 ports
- 1x Gigabit Ethernet port (supports PoE with add-on PoE HAT)
- 1x Raspberry Pi camera port (2-lane MIPI CSI)
- 1x Raspberry Pi display port (2-lane MIPI DSI)
- 28x user GPIO supporting various interface options:
  - Up to 6x UART
  - Up to 6x I2C
  - Up to 5x SPI
  - 1x SDIO interface
  - 1x DPI (Parallel RGB Display)
  - 1x PCM
  - Up to 2x PWM channels
  - Up to 3x GPCLK outputs

Bild A1-13: *Datenblatt des Raspberry Pi 4B ([Ras19], S.6)*

2.3 Software

- ARMv8 Instruction Set
- Mature Linux software stack
- Actively developed and maintained
  - Recent Linux kernel support
  - Many drivers upstreamed
  - Stable and well supported userland
  - Availability of GPU functions using standard APIs

3 Mechanical Specification

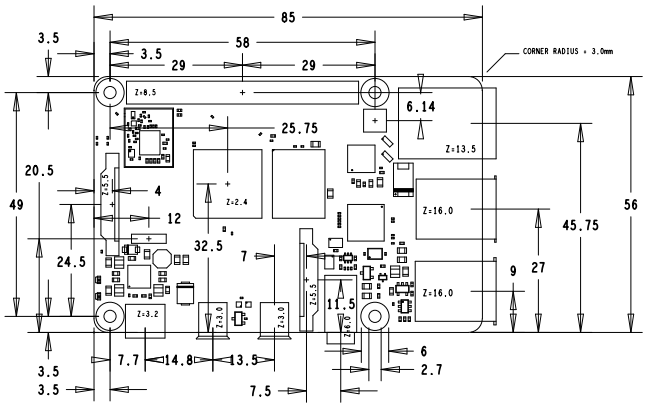


Figure 1: Mechanical Dimensions

4 Electrical Specification

Caution! Stresses above those listed in Table 2 may cause permanent damage to the device. This is a stress rating only; functional operation of the device under these or any other conditions above those listed in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Bild A1-14: Datenblatt des Raspberry Pi 4B ([Ras19], S.7)

Symbol	Parameter	Minimum	Maximum	Unit
VIN	5V Input Voltage	-0.5	6.0	V

Table 2: Absolute Maximum Ratings

Please note that VDD_IO is the GPIO bank voltage which is tied to the on-board 3.3V supply rail.

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{IL}	Input low voltage[a]	VDD_IO = 3.3V	-	-	TBD	V
V_{IH}	Input high voltage[a]	VDD_IO = 3.3V	TBD	-	-	V
I_{IL}	Input leakage current	TA = +85°C	-	-	TBD	μA
C_{IN}	Input capacitance	-	-	TBD	-	pF
V_{OL}	Output low voltage[b]	VDD_IO = 3.3V, IOL = -2mA	-	-	TBD	V
V_{OH}	Output high voltage[b]	VDD_IO = 3.3V, IOH = 2mA	TBD	-	-	V
I_{OL}	Output low current[c]	VDD_IO = 3.3V, VO = 0.4V	TBD	-	-	mA
I_{OH}	Output high current[c]	VDD_IO = 3.3V, VO = 2.3V	TBD	-	-	mA
R_{PU}	Pullup resistor	-	TBD	-	TBD	kΩ
R_{PD}	Pulldown resistor	-	TBD	-	TBD	kΩ

[a] Hysteresis enabled
[b] Default drive strength (8mA)
[c] Maximum drive strength (16mA)

Table 3: DC Characteristics

Pin Name	Symbol	Parameter	Minimum	Typical	Maximum	Unit
Digital outputs	t_{rise}	10-90% rise time[a]	-	TBD	-	ns
Digital outputs	t_{fall}	90-10% fall time[a]	-	TBD	-	ns

[a] Default drive strength, CL = 5pF, VDD_IO = 3.3V

Table 4: Digital I/O Pin AC Characteristics

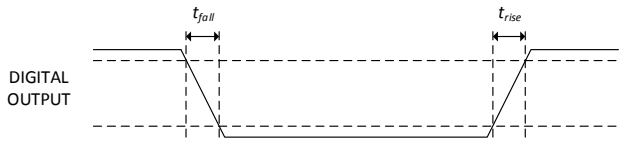


Figure 2: Digital IO Characteristics

Bild A1-15: Datenblatt des Raspberry Pi 4B ([Ras19], S.8)

A1.7 SPS B&R X20CP1382-RT

4 Technische Daten

Bestellnummer	X20CP1381-RT	X20CP1382-RT	X20cCP1382-RT
Kurzbeschreibung			
Schnittstellen	1x RS232, 1x Ethernet, 1x POWERLINK, 2x USB, 1x X2X Link, 1x CAN-Bus		
Systemmodul	Zentraleinheit		
Allgemeines			
Kühlung	Lüfterlos		
B&R ID-Code	0xE35D	0xE35E	0xE707
Statusanzeigen	CPU-Funktion, Ethernet, POWERLINK, RS232, CAN-Bus, CAN-Bus-Abschlusswiderstand, CPU-Versorgung, I/O-Versorgung, I/O-Funktion pro Kanal		
Diagnose			
Ausgänge	Digitalausgänge: Ja, per Status-LED und SW-Status (Ausgangsfehlerstatus)		
CPU-Funktion	Ja, per Status-LED		
Datenübertragung CAN-Bus	Ja, per Status-LED		
Datenübertragung RS232	Ja, per Status-LED		
Eingänge	Analogeingänge: Ja, per Status-LED und SW-Status		
Ethernet	Ja, per Status-LED		
I/O-Versorgung	Ja, per Status-LED		
POWERLINK	Ja, per Status-LED		
Versorgungsspannungsüberwachung	Ja, per Status-LED		
Übertemperatur	Ja, per SW-Status		
Abschlusswiderstand	Ja, per Status-LED		
CPU Redundanz möglich	Nein		
ACOPOS fähig	Ja		
reACTION-fähige I/Os	Ja		
Visual Components fähig	Ja		
Leistungsaufnahme ohne Schnittstellenmodul und ohne USB	5,1 W	5,8 W	
Leistungsaufnahme für X2X Link Versorgung [1]	0,8 W		
Leistungsaufnahme [1]			
I/O-intern	0,8 W		
Zusätzliche Verlustleistung durch Aktoren (ohmsch) [W]	-		
Ausführung der Signalleitungen	Für alle schnellen digitalen Ein-/Ausgänge sind geschirmte Leitungen zu verwenden, Leitungslänge: max. 20 m		
Zulassungen			
CE	Ja		
ATEX	Zone 2, II 3G Ex nA nC IIA T5 Gc IP20, Ta (siehe X20 Anwenderhandbuch) FTZÚ 09 ATEX 0083X		
UL	cULus E115267 Industrial Control Equipment		
HazLoc	cCSAus 244665 Process Control Equipment for Hazardous Locations Class I, Division 2, Groups ABCD, T5		
DNV GL	Temperature: **B** (0 - 55 °C) Humidity: **B** (up to 100%) Vibration: **B** (4 g) EMC: **B** (bridge and open deck)		
KR	Ja		
EAC	Ja		
CPU und X2X Link Versorgung			
Eingangsspannung	24 VDC -15% / +20%		
Eingangsstrom	max. 1 A		
Sicherung	Integriert, nicht tauschbar		
Verpolungsschutz	Ja		
Ausgang X2X Link Versorgung			
Ausgangsnennleistung	2 W		
Parallelschaltung	Ja [2]		
Redundanzbetrieb	Ja [3]		
Eingang I/O-Versorgung			
Eingangsspannung	24 VDC -15% / +20%		
Sicherung	Erforderliche Vorsicherung max. T 10 A		
Ausgang I/O-Versorgung			
Ausgangsnennspannung	24 VDC		
Zulässige Kontaktbelastung	10 A		
Controller			
Echtzeituhr	Pufferung min. 300 Std., typ. 1000 Std. bei 25°C, Auflösung 1 s, -18 bis 28 ppm Genauigkeit bei 25°C		
FPU	Ja		

Tabelle 3: Technische Daten

Bild A1-16: Datenblatt der SPS X20CP1382-RT der Fa. B&R, Seite 1 ([BR19b], S. 3)

X20CP1381-RT und X20(c)CP1382-RT

Bestellnummer	X20CP1381-RT	X20CP1382-RT	X20cCP1382-RT
Prozessor			
Typ		Vx86EX	
Taktfrequenz	200 MHz	400 MHz	
L1 Cache			
Datencode		16 kByte	
Programmcode		16 kByte	
L2 Cache		128 kByte	
Integrierter I/O-Prozessor		Bearbeitet I/O-Datenpunkte im Hintergrund	
Modulare Schnittstellensteckplätze		1	
Remanente Variablen	16 kByte FRAM, Puf-ferung >10 Jahre [4]	32 kByte FRAM, Pufferung >10 Jahre [4]	
Kürzeste Taskklassen-Zykluszeit	2 ms	1 ms	
Typische Befehlszykluszeit	0,0419 µs	0,0199 µs	
Standardspeicher			
Arbeitsspeicher	128 MByte DDR3-SDRAM	256 MByte DDR3-SDRAM	
Anwenderspeicher			
Typ	Flashspeicher 1 GByte eMMC	Flashspeicher 2 GByte eMMC	
Datenerhaltung		10 Jahre	
schreibbare Datenmenge			
garantiert		40 TByte	
ergibt bei 5 Jahren		21,9 GByte/Tag	
garantierte Lösch-/Schreibzyklen		20.000	
Error Correction Coding (ECC)		Ja	
Schnittstellen			
Schnittstelle IF1			
Signal		RS232	
Ausführung		Kontaktierung über 16-polige Feldklemme X20TB1F	
max. Reichweite		900 m	
Übertragungsrate		max. 115,2 kBit/s	
Schnittstelle IF2			
Signal		Ethernet	
Ausführung		1x RJ45 geschirmt	
Leitungslänge		max. 100 m zwischen 2 Stationen (Segmentlänge)	
Übertragungsrate		10/100 MBit/s	
Übertragung			
Physik		10BASE-T/100BASE-TX	
Halbduplex		Ja	
Vollduplex		Ja	
Autonegotiation		Ja	
Auto-MDI/MDIX		Ja	
Schnittstelle IF3			
Feldbus		POWERLINK Managing oder Controlled Node	
Typ		Typ 4 [5]	
Ausführung		1x RJ45 geschirmt	
Leitungslänge		max. 100 m zwischen 2 Stationen (Segmentlänge)	
Übertragungsrate		100 MBit/s	
Übertragung			
Physik		100BASE-TX	
Halbduplex		Ja	
Vollduplex		POWERLINK-Modus: Nein / Ethernet-Modus: Ja	
Autonegotiation		Ja	
Auto-MDI/MDIX		Ja	
Schnittstelle IF4			
Typ		USB 1.1/2.0	
Ausführung		Typ A	
max. Ausgangsstrom		0,5 A	
Schnittstelle IF5			
Typ		USB 1.1/2.0	
Ausführung		Typ A	
max. Ausgangsstrom		0,1 A	
Schnittstelle IF6			
Feldbus		X2X Link Master	
Schnittstelle IF7			
Signal		CAN-Bus	
Ausführung		Kontaktierung über 16-polige Feldklemme X20TB1F	
max. Reichweite		1000 m	
Übertragungsrate		max. 1 MBit/s	
Abschlusswiderstand		Im Modul integriert	
Controller		SJA 1000	
Digitale Eingänge			
Anzahl		14 Standardeingänge, 4 schnelle Eingänge und 4 Mischkanä-le, Konfiguration als Ein- oder Ausgang erfolgt über Software	
Nennspannung		24 VDC	

Tabelle 3: Technische Daten

Bild A1-17: Datenblatt der SPS X20CP1382-RT der Fa. B&R, Seite 2 ([BR19b], S. 4)

X20CP1381-RT und X20(c)CP1382-RT

Bestellnummer	X20CP1381-RT	X20CP1382-RT	X20cCP1382-RT
Eingangsspannung	24 VDC -15% / +20%		
Eingangsstrom bei 24 VDC	X1 - Standardeingänge: Typ. 3,5 mA X2 - Standardeingänge: Typ. 2,68 mA X2 - schnelle Eingänge: Typ. 3,5 mA X3 - Mischkanäle: Typ. 2,68 mA		
Eingangsbeschaltung	Sink		
Eingangsfilter			
Hardware	Standardeingänge und Mischkanäle: ≤200 µs Schnelle Eingänge: ≤2 µs, bei Verwendung als Standardeingänge: ≤200 µs		
Software	Default 1 ms, zwischen 0 und 25 ms in 0,1 ms Schritten einstellbar		
Anschlusstechnik	1-Leitertechnik		
Eingangswiderstand	X1 - Standardeingänge: 6,8 kΩ X2 - Standardeingänge: 8,9 kΩ X2 - schnelle Eingänge: 6,8 kΩ X3 - Mischkanäle: 8,9 kΩ		
Zusatzfunktionen	X2 - schnelle digitale Eingänge: 2x 250 kHz Ereigniszählung, 2x AB-Zähler, ABR-Inkrementalgeber, Richtung/Frequenz, Periodendauermessung, Torzeitmessung, Differenzzeitmessung, Flankenzähler, Flankenzeiten		
Schaltschwellen			
Low	<5 VDC		
High	>15 VDC		
AB-Inkrementalgeber			
Anzahl	2		
Gebereingänge	24 V, asymmetrisch		
ZählTiefe	32 Bit		
Eingangsfrequenz	max. 100 kHz		
Auswertung	4-fach		
Geberversorgung	Modulintern, max. 300 mA		
Überlastverhalten der Geberversorgung	Kurzschlussfest, überlastfest		
ABR-Inkrementalgeber			
Anzahl	1		
Gebereingänge	24 V, asymmetrisch		
ZählTiefe	32 Bit		
Eingangsfrequenz	max. 100 kHz		
Auswertung	4-fach		
Geberversorgung	Modulintern, max. 300 mA		
Überlastverhalten der Geberversorgung	Kurzschlussfest, überlastfest		
Ereigniszähler			
Anzahl	2		
Signalform	Rechteckimpulse		
Auswertung	1-fach		
Eingangsfrequenz	max. 250 kHz		
Zählfrequenz	250 kHz		
ZählTiefe	32 Bit		
Flankenerkennung / Zeitmessung			
Mögliche Messungen	Periodendauermessung, Torzeitmessung, Differenzzeitmessung, Flankenzähler, Flankenzeiten		
Messungen pro Modul	Jede Funktion bis zu 2-mal		
ZählTiefe	32 Bit		
Eingangsfrequenz	max. 10 kHz		
Zeitstempel	1 µs Auflösung		
Signalform	Rechteckimpulse		
Analoge Eingänge			
Eingang	±10 V oder 0 bis 20 mA / 4 bis 20 mA, über unterschiedliche Klemmstellen		
Eingangsart	Differenzeingang		
Digitale Wandlerauflösung			
Spannung	±12 Bit		
Strom	12 Bit		
Wandlungszeit	1 Kanal aktiviert: 100 µs 2 Kanäle aktiviert: 200 µs		
Ausgabeformat			
Datentyp	INT		
Spannung	INT 0x8001 - 0x7FFF / 1 LSB = 0x0008 = 2,441 mV		
Strom	INT 0x0000 - 0x7FFF / 1 LSB = 0x0008 = 4,883 µA		
Eingangsimpedanz im Signalbereich			
Spannung	20 MΩ		
Strom	-		
Bürde			
Spannung	-		
Strom	<300 Ω		
Eingangsschutz	Schutz gegen Beschaltung mit Versorgungsspannung		
Zulässiges Eingangssignal			
Spannung	max. ±30 V		
Strom	max. ±50 mA		

Tabelle 3: Technische Daten

Bild A1-18: Datenblatt der SPS X20CP1382-RT der Fa. B&R, Seite 3 ([BR19b], S. 5)

X20CP1381-RT und X20(c)CP1382-RT

Bestellnummer	X20CP1381-RT	X20CP1382-RT	X20cCP1382-RT
Ausgabe des Digitalwertes unter Überlastbedingungen	Konfigurierbar		
Wandlungsverfahren	SAR		
Eingangsfilter	Tiefpass 3. Ordnung / Eckfrequenz 1 kHz		
max. Fehler bei 25°C			
Spannung			
Gain	0,18% (Rev. <C0: 0,37%) [6]		
Offset	0,04% (Rev. <C0: 0,25%) [7]		
Strom			
Gain	0 bis 20 mA = 0,15% (Rev. <C0: 0,52%) / 4 bis 20 mA = 0,25% [6]		
Offset	0 bis 20 mA = 0,1% (Rev. <C0: 0,4%) / 4 bis 20 mA = 0,15% [8]		
max. Gain-Drift			
Spannung	0,017 %/°C [6]		
Strom	0 bis 20 mA = 0,015 %/°C / 4 bis 20 mA = 0,023 %/°C [6]		
max. Offset-Drift			
Spannung	0,008 %/°C [7]		
Strom	0 bis 20 mA = 0,008 %/°C / 4 bis 20 mA = 0,012 %/°C [8]		
Gleichtaktunterdrückung			
DC	70 dB		
50 Hz	70 dB		
Gleichtaktbereich	±12 V		
Übersprechen zwischen den Kanälen	<-70 dB		
Nichtlinearität			
Spannung	<0,025 % [7]		
Strom	<0,05 % [8]		
Temperatureingänge Widerstandsmessung			
Anzahl	1		
Eingang	Widerstandsmessung mit Konstantstromspeisung für 2-Leitertechnik		
Digitale Wandlerauflösung	13 Bit		
Wandlungszeit	Nur Temperatureingang aktiviert: 200 µs Temperatur- und Analogeingang aktiviert: 400 µs		
Wandlungsverfahren	SAR		
Ausgabeformat	INT bzw. UINT für Widerstandsmessung		
Fühler			
PT1000	-200 bis 850°C		
Widerstandsmessbereich	0,1 bis 4000 Ω		
Auflösung Temperaturfühler	1LSB = 0x0005 = 0,16 °C		
Auflösung bei Widerstandsmessung	1LSB = 0x0005 = 0,49 Ω		
Eingangsfilter	Tiefpass 1. Ordnung / Eckfrequenz 7 Hz		
Fühlernorm	EN 60751		
Gleichtaktbereich	1 V		
Linearisierungsmethode	Intern		
Messstrom	1 mA		
Zulässiges Eingangssignal	Kurzzeitig max. ±30 V		
max. Fehler bei 25°C			
Gain	0,3% (Rev. <C0: 1,93%) [9]		
Offset	0,15% (Rev. <C0: 0,32%) [10]		
max. Gain-Drift	0,023 %/°C [9]		
max. Offset-Drift	0,012 %/°C [10]		
Nichtlinearität	<0,05% [10]		
normierter Wertebereich bei Widerstandsmessung	0,1 bis 4000,0 Ω		
Übersprechen zwischen den Kanälen	<-70 dB		
Gleichtaktunterdrückung			
50 Hz	>60 dB		
DC	-		
Normierung Temperaturfühler			
PT1000	-200 bis 850°C		
Digitale Ausgänge			
Anzahl	4 Standardausgänge, 4 schnelle Ausgänge und 4 Mischkanäle, Konfiguration als Ein- oder Ausgang erfolgt über Software		
Ausführung	Standardausgänge und Mischkanäle: FET Plus-schaltend Schnelle Ausgänge: Push-Pull		
Nennspannung	24 VDC		
Schaltspannung	24 VDC -15% / +20%		
Ausgangsnennstrom	Standardausgänge und Mischkanäle: 0,5 A Schnelle Ausgänge: 0,2 A		
Summennennstrom	Standardausgänge und Mischkanäle: 4 A Schnelle Ausgänge: 0,8 A		
Anschlusstechnik	1-Leitertechnik		
Ausgangsbeschaltung	Standardausgänge und Mischkanäle: Source Schnelle Ausgänge: Sink oder Source		
Ausgangsschutz [11]	Thermische Abschaltung bei Überstrom oder Kurzschluss (siehe Wert "Kurzschlussspitzenstrom") Interne Freilaufdiode zum Schalten induktiver Lasten (siehe Abschnitt "Schalten induktiver Lasten")		

Tabelle 3: Technische Daten

Bild A1-19: Datenblatt der SPS X20CP1382-RT der Fa. B&R, Seite 4 ([BR19b], S. 6)

X20CP1381-RT und X20(c)CP1382-RT

Bestellnummer	X20CP1381-RT	X20CP1382-RT	X20cCP1382-RT
Pulsweitenmodulation [12]			
Periodendauer	5 bis 65535 µs entspricht 200 kHz bis 15 Hz		
Impulsdauer	0 bis 100%, minimal 2,5 µs		
Auflösung für Impulsdauer	0,1% der eingestellten Frequenz		
Diagnosestatus	Standardausgänge und Mischkanäle: Ausgangsüberwachung mit Verzögerung 10 ms Schnelle Ausgänge: Ausgangsüberwachung mit Verzögerung 10 µs		
Leckstrom im ausgeschalteten Zustand	Standardausgänge und Mischkanäle: 5 µA Schnelle Ausgänge: 25 µA		
$R_{DS(on)}$	140 mΩ [13]		
Restspannung	Standardausgänge und Mischkanäle: <0,1 V bei Nennstrom 0,5 A Schnelle Ausgänge: <0,9 V bei Nennstrom 0,1 A		
Kurzschlussspitzenstrom	Standardausgänge und Mischkanäle: <3 A Schnelle Ausgänge: <20 A		
Einschaltung bei Überlastabschaltung bzw. Kurzschlussabschaltung	Standardausgänge und Mischkanäle: ca. 10 ms (abhängig von der Modultemperatur) Schnelle Ausgänge: Keine Einschaltung		
Schaltverzögerung			
0 -> 1	Standardausgänge und Mischkanäle: <300 µs Schnelle Ausgänge: <3 µs		
1 -> 0	Standardausgänge und Mischkanäle: <300 µs Schnelle Ausgänge: <3 µs		
Schaltfrequenz			
ohmsche Last [14]	Standardausgänge und Mischkanäle: max. 500 Hz Schnelle Ausgänge: 50 kHz, max. 200 kHz (siehe Abschnitt "Derating für Schaltfrequenz der schnellen digitalen Ausgänge")		
induktive Last	Siehe Abschnitt "Schalten induktiver Lasten"		
Bremsspannung beim Abschalten induktiver Lasten	Standardausgänge und Mischkanäle: typ. 45 VDC		
Elektrische Eigenschaften			
Potenzialtrennung	Ethernet (IF2), POWERLINK (IF3) und X2X (IF6) zueinander, zu weiteren Schnittstellen und zur SPS getrennt Kanal zu Bus getrennt Kanal zu Kanal und Kanal zu SPS nicht getrennt		
Einsatzbedingungen			
Einbaulage			
waagrecht	Ja		
senkrecht	Ja		
Aufstellungshöhe über NN (Meeresspiegel)			
0 bis 2000 m	Keine Einschränkung		
>2000 m	Reduktion der Umgebungstemperatur um 0,5°C pro 100 m		
Schutzart nach EN 60529	IP20		
Umgebungsbedingungen			
Temperatur			
Betrieb			
waagrechte Einbaulage	-25 bis 60°C		
senkrechte Einbaulage	-25 bis 50°C		
Derating	Siehe Abschnitt "Derating für Schaltfrequenz der schnellen digitalen Ausgänge"		
Lagerung	-40 bis 85°C		
Transport	-40 bis 85°C		
Luftfeuchtigkeit			
Betrieb	5 bis 95%, nicht kondensierend		Bis 100%, kondensierend
Lagerung	5 bis 95%, nicht kondensierend		
Transport	5 bis 95%, nicht kondensierend		
Mechanische Eigenschaften			
Anmerkung	X20 Abschlussplatte rechts im Lieferumfang enthalten 3 Stück X20 Feldklemmen 16-fach im Lieferumfang enthalten Abdeckung für den Schnittstellenmodulsteckplatz im Lieferumfang enthalten		
Abmessungen			
Breite	164 mm		
Höhe	99 mm		
Tiefe	75 mm		
Gewicht	310 g		

Tabelle 3: Technische Daten

1) Die angegebenen Werte sind Maximalangaben. Beispiele für die genaue Berechnung sind im X20 System Anwenderhandbuch im Abschnitt "Mechanische und elektrische Konfiguration" zu finden.
2) Bei Parallelbetrieb darf die Nennleistung von 2 W nicht zur Gesamtleistung addiert werden.
3) Bis zu 2 W Buslast.
4) Die Speichergröße für die remanenten Variablen ist in Automation Studio einstellbar.
5) Siehe Automation Help unter "Kommunikation, POWERLINK, Allgemeines, Hardware - IF/LS" für weitere Informationen.
6) Bezogen auf den aktuellen Messwert.
7) Bezogen auf den Messbereich 20 V.
8) Bezogen auf den Messbereich 20 mA.
9) Bezogen auf den aktuellen Widerstandsmesswert.
10) Bezogen auf den gesamten Widerstandsmessbereich.
11) Bei den schnellen digitalen Ausgängen ist bei einer Schaltfrequenz >50 kHz ein Derating zu beachten (siehe Abschnitt "Derating für Schaltfrequenz der schnellen digitalen Ausgänge"). Es ist kein Übertemperaturschutz vorgesehen.

Bild A1-20: Datenblatt der SPS X20CP1382-RT der Fa. B&R, Seite 5 ([BR19b], S. 7)

A1.8 SPS Beckhoff C6030-0060

C6030-0060

C6030-0060 | Ultra-Kompakt-Industrie-PC

Varianten	Prozessor	verfügbar
C6030-0060	Intel® Celeron® G3900 2.8 GHz, 2 Cores (TC3: 50)	ja

*Die TwinCAT-3-Leistungsklasse bestimmt die genaue Bestellnummer für das jeweilige TwinCAT-3-Produkt. Eine Übersicht der einzelnen TC3-Leistungsklassen finden Sie hier.

C6030-0060	Ultra-Kompakt-Industrie-PC
Gehäuse	Industrie-PC für den raumsparenden Schaltschrankeinbau
	Montageplatte an der Rückwand zur freien Orientierung der Anschlussebene
	alle Anschlüsse auf einer Ebene
	flexible Montagevorrichtung zur freien Orientierung der Anschlussebene
	Aluminium-Zinkdruckguss-Gehäuse
	Status-LEDs
	Lithiumbatterie leicht zugänglich unter der seitlichen Abdeckung
	2 Slots für M.2-SSDs leicht zugänglich unter der seitlichen Abdeckung
	drehzahlüberwachter geregelter Lüfter, doppelt kugelgelagert, wechselbar
	5 cm freier Raum umlaufend um den PC zur Luftzirkulation erforderlich
	Schutzart IP 20
	Betriebstemperaturbereich 0…55 °C
	geringe Abmessungen (B x H x T) 129 x 133 x 76 mm, ohne Montageplatte
Leistungsmerkmale	Prozessor Intel® Celeron® G3900 2,8 GHz, 2 Cores (TC3: 50)
	kompaktes Motherboard für Intel® Celeron®, Pentium®, Core™ i3, Core™ i5 oder Core™ i7 der sechsten und siebten Generation
	4 GB DDR4-RAM, erweiterbar auf 32 GB, mit einem 32-Bit-Betriebssystem sind nur 3 GB adressierbar.
	Grafikadapter im Intel®-Prozessor integriert, 2 DisplayPort-Anschlüsse
	On-Board-Ethernet-Adapter mit 4 x 100/1000BASE-T-Anschluss
	40-GB-M.2-SSD, 3D-Flash, erweiterter Temperaturbereich
	4 x USB 3.0
	24-V-DC-Netzteil

Optionen	C6030-0060
C9900-C611	Prozessor Intel® Pentium® G4400 der sechsten Generation, 3,3 GHz, 2 Cores (TC3: 50), statt Intel® Celeron® G3900 2,8 GHz (TC3: 50)
C9900-C612	Prozessor Intel® Core™ i3-6100 der sechsten Generation, 3,7 GHz, 2 Cores (TC3: 60), statt Intel® Celeron® G3900 2,8 GHz (TC3: 50)
C9900-C615	Prozessor Intel® Core™ i3-7101E der siebten Generation, 3,9 GHz, 2 Cores (TC3: 60), erfordert Windows 10, 64 Bit, statt Intel® Celeron® G3900 2,8 GHz (TC3: 50)
C9900-C613	Prozessor Intel® Core™ i5-6500 der sechsten Generation, 3,2 GHz, 4 Cores (TC3: 70), statt Intel® Celeron® G3900 2,8 GHz (TC3: 50)
C9900-C616	Prozessor Intel® Core™ i5-7500 der siebten Generation, 3,4 GHz, 4 Cores (TC3: 70), erfordert Windows 10, 64 Bit, statt Intel® Celeron® G3900 2,8 GHz (TC3: 50)
C9900-C614	Prozessor Intel® Core™ i7-6700 der sechsten Generation, 3,4 GHz, 4 Cores (TC3: 80), statt Intel® Celeron® G3900 2,8 GHz (TC3: 50)
C9900-C617	Prozessor Intel® Core™ i7-7700 der siebten Generation, 3,6 GHz, 4 Cores (TC3: 80), erfordert Windows 10, 64 Bit, statt Intel® Celeron® G3900 2,8 GHz (TC3: 50)
C9900-R270	Speichererweiterung auf 8 GB DDR4-RAM, statt 4 GB, erfordert ein 64-Bit-Betriebssystem.
C9900-R271	Speichererweiterung auf 16 GB DDR4-RAM, statt 4 GB, erfordert ein 64-Bit-Betriebssystem.
C9900-R272	Speichererweiterung auf 32 GB DDR4-RAM, statt 4 GB, erfordert ein 64-Bit-Betriebssystem.
C9900-Z468	Adapterkabel DisplayPort auf DVI, 40 cm
C9900-B406	Motherboard mit On-Board-SATA-RAID-1-Controller, Intel® Rapid Storage Technology, statt Standard Motherboard
C9900-M668	Montageplatte an der Seitenwand, statt an der Rückwand
C9900-M669	Montageplatte für die seitliche Montage des C603x, Einzelteil, nicht montiert
C9900-H597	80-GB-M.2-SSD, 3D-Flash, erweiterter Temperaturbereich, für C603x, statt 40-GB-M.2-SSD
C9900-H598	160-GB-M.2-SSD, 3D-Flash, erweiterter Temperaturbereich, für C603x, statt 40-GB-M.2-SSD
C9900-H594	40-GB-M.2-SSD, 3D-Flash, erweiterter Temperaturbereich, für C603x
C9900-H595	80-GB-M.2-SSD, 3D-Flash, erweiterter Temperaturbereich, für C603x
C9900-H596	160-GB-M.2-SSD, 3D-Flash, erweiterter Temperaturbereich, für C603x

BECKHOFF New Automation Technology

Technische Änderungen vorbehalten

Bild A1-21: Datenblatt der SPS C6030-0060 der Fa. Beckhoff [Bec19b]

Erklärung zur Zitation von Inhalten aus studentischen Arbeiten

In Ergänzung zu meinem Antrag auf Zulassung zur Promotion in der Fakultät für Maschinenbau der Universität Paderborn erkläre ich gemäß §11 der Promotionsordnung und unter Beachtung der Regelung zur Zitation studentischer Arbeiten:

Die von mir vorgelegte Dissertation habe ich selbstständig verfasst und ich habe keine anderen als die dort angegebenen Quellen und Hilfsmittel benutzt. Es sind Inhalte studentischen Ursprungs (studentische Arbeiten) in dieser Dissertation enthalten.

Ich habe die verwendeten Arbeiten entsprechend der Regelung „Zitation aus studentischen Arbeiten in Dissertationen“ zitiert.

Lübbecke, im Oktober 2024 Arne Thorsten Rüting, M.Sc.