Analyse und Synthese von Werkzeugmaschinen mit paralleler Kinematik

Von der Fakultät für Ingenieurwissenschaften, Abteilung Maschinenbau der
Universität Duisburg-Essen
zur Erlangung des akademischen Grads

DOKTOR-INGENIEUR

genehmigte Dissertation

von

Andreas Pott
aus
Solingen

Referent: Prof. Dr.-Ing. habil. Dr. h.c. M. Hiller
Korreferent: Prof. Dr.-Ing. A. Verl
Tag der Einreichung: 29. September 2006
Tag der mündlichen Prüfung: 29. März 2007

Danksagung

Die vorliegende Dissertation entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter bei Herrn Prof. Dr.-Ing. habil. Dr. h.c. M. Hiller sowie Herrn Prof. Dr.-Ing. D. Schramm am Lehrstuhl für Mechatronik der Universität Duisburg-Essen im Rahmen eines von der Deutschen Forschungsgemeinschaft geförderten Projekts im Schwerpunktprogramm 1099 *Fertigungsmaschinen mit paralleler Kinematik*.

Mein ganz besonderer Dank gilt Herrn Prof. Manfred Hiller für die vielfältige Förderung während meines Studiums, die zahlreichen Ratschläge, die fachlichen Diskussionen, die Unterstützung meiner wissenschaftlichen Arbeit und die Übernahme des Hauptberichts dieser Arbeit.

Herrn Prof. Dr.-Ing. Alexander Verl vom Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW) der Universität Stuttgart danke ich herzlich für die gute Zusammenarbeit im Rahmen des DFG Projekts sowie für die Übernahme des Mitberichts.

Ich danke allen ehemaligen Kollegen am Lehrstuhl für Mechatronik für das angenehme Arbeitsklima, die vielen Diskussionen und die hilfsbereite Zusammenarbeit. Insbesondere gilt mein Dank Herrn Dr.-Ing. Richard Verhoeven und Herrn Dr.-Ing. Daniel Germann für die fachlichen Anregungen und Ratschläge, die wesentliche Impulse für meine Arbeit lieferten. Weiterhin danke ich allen meinen Kollegen, die durch eine aufmerksame Durchsicht der Arbeit zu ihrem Gelingen beigetragen haben. Meinem Kollegen Dipl.-Ing. Tim Boye vom Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW) der Universität Stuttgart danke ich für die Jahre der erfolgreichen Zusammenarbeit am gemeinsamen Projekt. Weiterhin möchte ich mich bei allen Studenten bedanken, die durch ihre Studien- und Diplomarbeiten einen Beitrag zu dieser Arbeit geleistet haben.

Ganz besonders bedanke ich mich bei meinen Eltern, die mich auf meinem Lebensweg stets unterstützt und gefördert haben. Ohne das Verständnis und die Geduld meiner Frau Ariane Pott wäre diese Arbeit sicherlich nicht zustande gekommen.

Sindelfingen, im August 2007 Andreas Pott

für Ariane

Inhaltsverzeichnis

Liste der verwendeten Formelzeichen

Variablen

Symbol	Beschreibung
$\underline{a}_i$	Ortsvektor zu den rahmenseitigen Anlenkpunkten des Beins
$\underline{b}_i$	Ortsvektor zu den plattformseitigen Anlenkpunkten des Beins
Δa	Tangentialer Versatz der rahmenseitigen Anlenkpunkte (*Linapod*)
Δh	Abstand zwischen unterer und oberer Plattformebene (*Linapod*)
Δt	Abstand zwischen $\mathcal{K}_p$ und unterer Plattformebene (*Linapod*)
$\Delta\alpha_p$	Verdrehung der Anlenkpunkte der Plattform
$\Delta\alpha_b$	Verdrehung der Anlenkpunkte der Basis
$\underline{c}$	Vektor der Berechnungsvariablen
$\underline{e}_x, \underline{e}_y, \underline{e}_z$	Einheitsvektor in Richtung der x, y, z-Achse
f	Kostenfunktion für die Optimierung
$\underline{f}$	Schnittkraft
$\underline{g}$	Vektor aller Geometrieparameter eines Roboters
$\underline{g}^{\mathrm{C}}$	Vektor der Zentrifugal- und Corioliskräfte
$\mathbb{I}$	Menge der reellen Intervalle
$\underline{\underline{I}}_n$	n-dimensionale Einheitsmatrix
$\underline{\underline{J}}$	JACOBI-Matrix
$\underline{\underline{J}}_g$	JACOBI-Matrix bezüglich der geometrischen Parameter
$\underline{\underline{J}}_q$	JACOBI-Matrix bezüglich der verallgemeinerten Koordinaten
$\mathcal{K}_i$	Koordinatensystem i
κ	Konditionszahl einer Matrix
$\underline{\underline{K}}$	Steifigkeitsmatrix
$\mathcal{L}$	Liste von Boxen
l_i	Länge des $i$ten Beins
l_l	Länge des unteren Beins (*Linapod*)
l_u	Länge des oberen Beins (*Linapod*)
$\underline{\underline{M}}$	verallgemeinerte Massenmatrix
$\underline{\tau}$	Schnittmoment
n_B	Anzahl der starren Körper
n_f	Anzahl der Freiheitsgrade
n_G	Anzahl der Gelenke
n_{G_i}	Anzahl der Freiheitsgrade des Gelenks G_i
n_L	Anzahl der kinematischen Schleifen
$\underline{\omega}$	Winkelgeschwindigkeit
$\underline{\Phi}$	System von Ungleichungen
$\underline{\varphi}^{\mathrm{DK}}$	Übertragungsfunktion der direkten Kinematik
$\underline{\varphi}^{\mathrm{IK}}$	Übertragungsfunktion der inversen Kinematik
$\underline{q}$	Vektor der verallgemeinerten Koordinaten

Symbol	Beschreibung
$\underline{Q}$	Vektor der verallgemeinerten Kräfte und Momente
$\underline{\Gamma}$	Gleichungssystem der Schließbedingungen der Kinematik
ν_i	Schließbedingung am Bein
$\underline{\underline{R}}$	Transformationsmatrix
$\underline{r}$	Position des TCP
$\mathbb{R}$	Menge der reellen Zahlen
r_b	Radius des Rahmens der Maschine
r_p	Radius der Plattform (SSM)
r_u	Radius der oberen Plattformebene (*Linapod*)
r_l	Radius der unteren Plattformebene (*Linapod*)
σ_i	Standardabweichung eines geometrischen Parameters
$\overline{\sigma}$	Gesamtfehlerverstärkung
ϱ	Metrikkoeffizienten zur Umrechnung zwischen Translation und Rotation
$\underline{\underline{T}}(\underline{u}, \beta)$	Drehtensor um die Achse $\underline{u}$ und den Winkel β
$\underline{\theta}$	Weltkoordinaten (Orientierung)
$\delta\underline{t}$	virtueller Geschwindigkeitswinder, zusammengesetzt aus den virtuellen Verschiebungen $\delta\underline{r}$ und den virtuellen Verdrehungen $\delta\underline{\varphi}$
$\delta\underline{r}$	virtuelle Verschiebungen
$\delta\underline{\varphi}$	virtuelle Verdrehungen
$\underline{v}$	Vektor der Verifikationsvariablen
$\mathcal{W}$	Arbeitsraum eines Roboters
$\mathcal{W}_{\mathrm{T}}$	translatorische Arbeitsraum
$\mathcal{W}_{\mathrm{GO}}$	Gesamtorientierungsarbeitsraum
$\underline{w}$	Kraftwinder, setzt sich zusammen aus dem Moment $\underline{\tau}$ und der Kraft $\underline{f}$
$\mathcal{X}$	Menge beim einem Constraint Satisfaction Problem
$\underline{y}$	Weltkoordinaten der Plattform; umfasst Position und Orientierung

Indizes

Index	Beschreibung
DK	direkte Kinematik
IK	inverse Kinematik
TCP	Tool Center Point
EEF	End-Effektor-Koordinatensystem

Operatoren

Ausdruck	Beschreibung
$\inf \widehat{x}$	Infimum des Intervalls $\widehat{x}$
$\sup \widehat{x}$	Supremum des Intervalls $\widehat{x}$

Ausdruck	Beschreibung
diam $\widehat{x}$	Breite des Intervalls $\widehat{x}$
mid $\widehat{x}$	Mittelpunkt des Intervalls $\widehat{x}$
Sp $\underline{\underline{A}}$	Spur der Matrix $\underline{\underline{A}}$
vect $\underline{\underline{A}}$	Erzeugt aus einer Matrix $\underline{\underline{A}}$ einen Vektor, vgl. Gl. (3.24)

Abkürzungen

Abkürzung	Bedeutung
CAS	Computeralgebra-System
CSP	Constraint Satisfaction Problem
EEF	End-Effector Frame (End-Effektor-Koordinatensystem)
MKS	Mehrkörpersystem
MPE	Multi Processing Environment
MPI	Message Passing Interface
PKM	Parallelkinematikmaschine
SSM	Simplified Symmetric Manipulator (vereinfachter symmetrischer Manipulator)
TCP	Tool Center Point (Werkzeugmittelpunkt)

Kapitel 1

Einleitung

In Abschnitt 1.1 wird die Motivation für die vorliegende Arbeit beschrieben, und Abschnitt 1.2 fasst den Stand der Technik zusammen. Abschnitt 1.3 formuliert die Ziele der Arbeit und gibt einen Überblick über die folgenden Kapitel. In Abschnitt 1.4 werden Konventionen für die Schreibweise zusammengefasst.

1.1 Problemstellung

Eine Parallelkinematikmaschine (PKM) ist ein Manipulator, der eine starre Basis mit einer beweglichen Plattform durch mehrere Beine gelenkig verbindet. Als im Jahr 1994 die ersten Werkzeugmaschinen mit paralleler Kinematik auf der Messe International Machine Tool Show (IMTS) in Chicago dem Fachpublikum vorgestellt wurden, herrschte eine große Euphorie, da die neue Maschinentechnik eine Reihe grundlegender Vorteile gegenüber bewährten Werkzeugmaschinen versprach. Eine drastische Reduzierung der bewegten Massen wird erreicht, indem alle Motoren am ruhenden Maschinenbett angebracht werden. Damit sollte eine erhebliche Verbesserung der dynamischen Eigenschaften und letztlich der Produktivität der neuen Werkzeugmaschinen einhergehen. Der fachwerkartige Aufbau des Manipulators erhöht die strukturelle Steifigkeit und reduziert gleichzeitig die Herstellungskosten durch den symmetrischen Aufbau mit zahlreichen Gleichteilen. Schließlich verringert die geschlossene kinematische Struktur die Auswirkungen einzelner Fertigungsfehler auf die Genauigkeit der Maschine.

Daraufhin wurden diverse Prototypen konstruiert und untersucht, doch konnten diese die hohen Erwartungen nicht erfüllen. Aufbau und Inbetriebnahme der Maschinen erwies sich als schwieriger als vermutet. Obwohl viele Baugruppen mehrfach auftreten, konnten die Kosten aufgrund der insgesamt großen Anzahl der Bauteile nicht reduziert werden. Insgesamt sind die Maschinen dadurch gekennzeichnet, dass sie eine große Empfindlichkeit gegenüber Änderungen der Geometrie aufweisen. Infolgedessen hat sich die Kalibrierung von PKM zu einem wichtigen Forschungsschwerpunkt entwickelt. Neben bisher fehlendem theoretischem und praktischem Wissen werden für PKM besondere Hardware-Komponenten benötigt, die in dieser Form eine geringe Verbreitung aufweisen. Beispielsweise mangelt es an tauglichen Kugel- und Kardangelenken, die sowohl bezüglich ihrer Steifigkeit als auch bezüglich der möglichen Beweglichkeit den Anforderungen von PKM gerecht werden.

Bei der Mehrzahl der räumlichen PKM ist keine geschlossene Lösung der direkten Kinematik bekannt. Daher sind die meisten für serielle Roboter entwickelten Verfahren nicht auf PKM anwendbar. Dies betrifft sowohl die kinematische Analyse und Maßsynthese als auch die Steuerungs- und Regelungstechnik. Weiterhin ändern sich die kinematischen Eigenschaften einer PKM stark über dem Arbeitsraum. Dies wirft vor allem die Frage auf, wie Maschinen entworfen werden können, die vom Anwender gestellte Anforderungen bezüglich der Größe und Qualität des Arbeitsraums erfüllen. Dabei spielt die Wahl einer geeigneten Geometrie des Roboters eine entscheidende Rolle, da der Einfluss der Geometrie sogar den Einfluss der topologischen Struktur übertrifft.

1.2 Literaturübersicht

Bereits in den 1950er Jahren stellte Gough einen Prüfstand für Reifen vor und wendete damit das Funktionsprinzip der Parallelkinematikmaschine erstmals praktisch an [36]. Im Jahr 1965 veröffentlichte Stewart den Entwurf eines Flugsimulators, bei dem das Cockpit durch sechs lineare Antriebe bewegt werden kann [124]. Dabei greifen drei Antriebe direkt an der Plattform an, während die anderen Antriebe die ersten drei abstützen. Die in den folgenden Jahren gebauten Flugsimulatoren [11, 14, 126] basieren auf dem von Gough vorgeschlagenen Manipulator, werden ironischerweise jedoch häufig als Stewart-Plattformen bezeichnet.

Die grundsätzliche Idee, eine bewegliche Plattform über mehreren kinematischen Ketten mit einem ruhenden Rahmen zu verbinden, wurde von vielen Wissenschaftlern untersucht. Neben diversen ebenen und sphärischen PKM spielen bei den räumlichen Robotern mit sechs Freiheitsgraden drei Architekturen eine große Rolle. Die erste ist die klassische Gough-Plattform, bei der sechs Beine mit veränderlicher Länge den Rahmen mit der Plattform verbinden. Bei den beiden anderen Architekturen werden Stäbe konstanter Länge verwendet und deren rahmenseitigen Anlenkpunkte werden variiert. In der Praxis wird dies entweder durch Schienen mit linearen Aktuatoren [139] oder durch rotatorisch angetriebene Hebel wie beim Delta-Roboter [19] realisiert.

Ein fundamentales Problem bei der Analyse von PKM ist die Lösung der direkten Kinematik. Während für serielle Roboter die Position des End-Effektors bei gegebenen Positionen der Aktuatoren eindeutig bestimmt ist und relativ einfach berechnet werden kann [26], ist dieses Problem bei PKM ausgesprochen kompliziert. Anhand von Beispielen wurde schnell klar, dass es bei PKM mehr als eine Lösung für die direkte Kinematik gibt. Erst im Jahr 1992 wurde gezeigt, dass für die allgemeine Gough-Plattform höchstens 40 verschiedene Lösungen der direkten Kinematik existieren, von denen jedoch einige komplex sein können [69, 115]. Später fand Dietmaier eine PKM, die über 40 verschiedene reelle Lösungen verfügt und damit 40 verschiedene Zusammenbaumodi besitzt [28]. Einen Algorithmus zur Lösung der direkten Kinematik stellt Husty [50] vor, der durch die Anwendung von kinematischen Abbildungen (engl. *Kinematic Mapping*) [7] die Gleichungen der direkten Kinematik in einen projektiven Raum überführt und damit algebraisiert. Die Lösung des Systems der Bindungsgleichungen mit Intervallanalyse stellt eine Möglichkeit dar, garantiert alle Lösungen der direkten Kinematik zu finden [80]. Überraschenderweise wird die Anzahl der Lösungen der direkten Kinematik nicht notwendigerweise beschränkt, wenn die Maschine mehr als sechs angetriebene Beine besitzt [52, 53]. Für praktische Zwecke

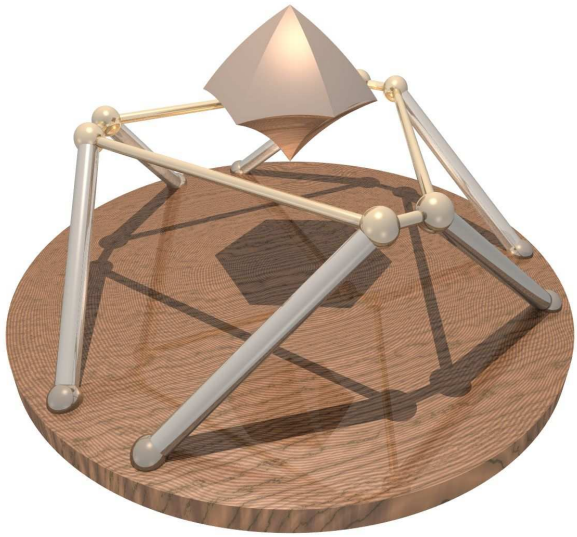


Abbildung 1.1: Arbeitsraum einer Stewart-Gough-Plattform mit Anwendung als Fahrsimulator. Der Arbeitsraum wurde geometrisch als Schnitt von Kugeln bestimmt.

interessiert meist nur die Lösung der direkten Kinematik, die zu einem bestimmten Zusammenbaumodus der Maschine gehört. Bis heute ist kein Verfahren bekannt, das es im allgemeinen Fall erlaubt, die verschiedenen Lösungen bestimmten Zusammenbaumodi zuzuordnen. In der Praxis wird daher eine Pose[1] eines bestimmten Zusammenbaumodus als Referenz vorgegeben und inkrementell verfolgt.

Für die meisten Anwendungen ist es notwendig, die möglichen Bewegungen der PKM zu kennen, so dass in zahlreichen Arbeiten Verfahren zur Berechnung des Arbeitsraums entwickelt wurden. Dabei wird betrachtet, welche Positionen die Plattform bei gegebenen Bewegungsmöglichkeiten der Aktuatoren erreichen kann. Für den translatorischen Arbeitsraum kann dies rein geometrisch mit Constructive Solid Geometry (CSG) Systemen erfolgen, d. h. durch die BOOLE'sche Verknüpfung von geometrischen Grundkörpern (Abb. 1.1) [77]. Der Arbeitsraum von verschiedenen Robotern mit drei Beinen wurde von Alizade et al. [1], Dash et al. [25] und Li [71] untersucht. Der 6D Arbeitsraum einer Gough-Plattform wurde von Merlet mit geometrischen Methoden berechnet [75]. Neben geometrischen Methoden hat sich die Intervallanalyse bewährt, um den Arbeitsraum von PKM zu berechnen. Intervallmethoden wurden für die Berechnung des Arbeitsraums einer Gough-Plattform [76], des Orthoglide [15, 16] und einer PKM mit Beinen konstanter Länge [103] erfolgreich eingesetzt.

In der Literatur wird der Rand des Arbeitsraums häufig mit Singularitäten identifiziert, was jedoch nur teilweise korrekt ist. Singularitäten stellen nur eine von mehreren Begrenzungen des Arbeitsraums dar. Eine Klassifikation von singulären Stellungen für Mechanismen mit kinematischen Schleifen wurde von Gosselin und Angeles [35] eingeführt. Bei Gough-Plattformen lassen sich singuläre Stellungen auch geometrisch beurteilen. Werden die Beine als PLÜCKER-Vektoren interpretiert, können anhand der GRASSMANN-Algebra alle singulären Konfigurationen untersucht werden [73, 74]. Eine Untersuchung der Singularitäten im Gesamtorientierungsarbeitsraum mit Quaternionen wurde von Pernkopf vorgestellt [97, 98]. Mit Intervallanalyse kann verifiziert werden, ob der gesamte Arbeits-

[1] Eine *Pose* bezeichnet die Position und die Orientierung der Plattform des Manipulators.

raum frei von Singularitäten ist [83]. Eine Untersuchung singulärer Stellungen mit Grassmann-Cayley-Algebra findet sich bei Ben-Horin und Shoham [3].

In einer Singularität degeneriert das kinematische Übertragungsverhalten eines Manipulators und stellt damit einen qualitativen Defekt der Übertragung dar. Eine quantitative Bewertung der Übertragung wird anhand eines Manipulierbarkeitsindex (engl. *Dexterity Index*) vorgenommen. In der Literatur wurden diverse Kennzahlen definiert, die sich aus der Jacobi-Matrix ableiten lassen, siehe beispielsweise [34, 111, 142]. Die bekanntesten Indizes sind die Determinante [141], die Konditionszahl [119] und der kleinste Singulärwert [65]. Obwohl diese Indizes für eine Konfiguration die Qualität der Übertragung bewerten, können diese Indizes nicht verwendet werden, um den geometrischen Abstand von einer singulären Stellung zu bestimmen, da es für den Raum der verallgemeinerten Geschwindigkeiten und Kräfte keine natürliche Metrik gibt [123]. In einigen neueren Ansätzen wird daher auf eine Bestimmung des Abstands verzichtet und stattdessen überprüft, ob ein zulässiger Wert für die Manipulierbarkeit im Arbeitsraum nicht über- bzw. unterschritten wird [82, 101].

Eine technische Beschränkung des Arbeitsraums entsteht durch die begrenzten Bewegungsmöglichkeiten der Gelenke und durch Kollisionen der bewegten Bauteile. Bei der klassischen Gough-Plattform wird der Arbeitsraum erheblich durch den maximalen Hub der Beine und den Schwenkbereich der Kugel-/Kardangelenke eingeschränkt. Der aus den Begrenzungen der Gelenke resultierende Rand des Arbeitsraums ist nicht durch eine singuläre Stellung gekennzeichnet. Daneben spielen die Kollisionen zwischen den Beinen eine wichtige Rolle.

Parallele Roboter besitzen eine hohe strukturelle Steifigkeit, d. h. Kräfte und Momente an der Plattform führen zu geringen Verformungen der Maschine. Der Vorteil relativiert sich jedoch aufgrund der meist unbefriedigenden Steifigkeiten derzeit verfügbarer Gelenke. Eine Analyse der Steifigkeit für PKM wurde von Gosselin [33] vorgestellt, und die Steifigkeit von parallelen Werkzeugmaschinen wurde von Rebeck [113] untersucht. Mithilfe der Jacobi-Matrix lässt sich leicht die Steifigkeit bezüglich der Abtriebe bestimmen [77, p. 256]. Eine Erweiterung durch eine vollständige geometrische Linearisierung der PKM erlaubt es, alle Bauteile als lineare Federn zu modellieren [107] und damit die Steifigkeit der PKM zu ermitteln.

Der Positionsfehler des End-Effektors von PKM wurde häufig in Abhängigkeit von Fehlern der Geometrie untersucht [10, 56, 62, 122, 143]. Weiterhin wurde der Einfluss von nicht exakt erfüllten Bindungen auf die kinematische Übertragung von PKM betrachtet [58]. Neben geometrischen Fehlern führt auch das Spiel in den Gelenken zu Fehlern [95]. Eine Modellierung von geometrischen Fehlern und Spiel in den Gelenken mit Intervallanalyse wurde von Wu vorgeschlagen [138], und Merlet berechnete mit Intervallanalyse eine obere Schranke für den größten Fehler im Arbeitsraum [81]. In diesem Zusammenhang wurde die Frage untersucht, wie die Fertigungstoleranzen gewählt werden müssen, um eine gegebene Genauigkeit zu erreichen [18, 57].

Bei der Kalibrierung wird die tatsächliche Geometrie der Maschine durch Vermessung identifiziert. Verfahren zur Kalibrierung von Robotern wurden von Hollerbach [49] klassifiziert. Bei PKM tritt das Problem auf, dass infolge der komplexen Kinematik alle Bewegungen miteinander gekoppelt sind. Daher wird mit externen Messgeräten die Position des Werkzeugmittelpunkts erfasst, um durch Vergleich mit dem internen Messsystem die Geometrie zu identifizieren [118, 128, 134]. Eine andere Möglichkeit besteht in der Integra-

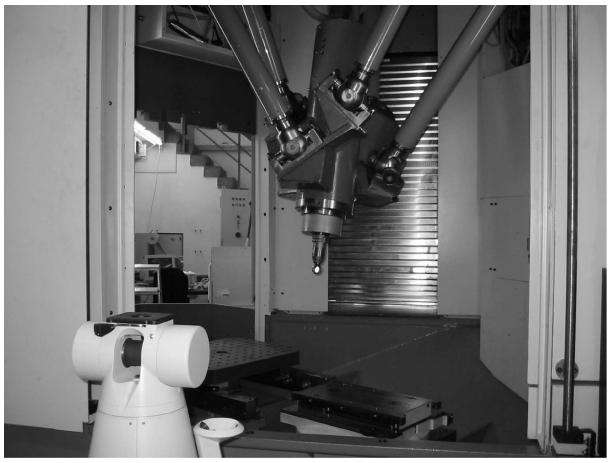


Abbildung 1.2: Kalibrierung des *Linapod* mit dem optischen Messgerät Lasertracker [8]

tion zusätzlicher Sensoren in die passiven Gelenke [55, 61]. Die Qualität der Kalibrierung wird anhand von Beobachtbarkeitsindizes untersucht, die aus der Identifikationsmatrix bestimmt werden [23, 90]. Eine Methode zur Kalibrierung von PKM mit Intervallanalyse wurde von Daney vorgestellt [23, 24]. Die Anzahl und Position der notwendigen Messstellen für die PKM *Linapod* untersuchte Boye [8] mit einem Lasertracker (Abb. 1.2).

Beim Entwurf von PKM muss zunächst die Struktur der Maschine ermittelt werden [110]. Die ausführlichen Analysen von PKM haben jedoch gezeigt, dass die Eigenschaften von PKM stärker durch die Geometrie als durch die Architektur beeinflusst werden. Zudem ändern sich die Eigenschaften signifikant über dem Arbeitsraum. Daher ist es notwendig, PKM speziell für bestimmte Aufgaben zu entwerfen [76]. Der typische Ansatz ist das *optimale Design* von Mechanismen, wie es für Mehrkörpersysteme im Allgemeinen [29] und PKM im Speziellen [41, 68, 96] angewendet wird. Für praktische Probleme müssen in der Regel mehrere antagonistische Kriterien berücksichtigt werden [63], wobei dies beim optimalen Design in Form eines Kompromisses realisiert wird. Um das Optimum der Zielfunktion zu finden, werden neben Gradientenverfahren auch evolutionäre Algorithmen verwendet [9]. Ein besonderer Aspekt der Auslegung besteht im Entwurf für einen gegebenen Arbeitsraum [94] oder für optimierte Manipulierbarkeit im Arbeitsraum [42, 99]. Einen neuen Ansatz für den Entwurf von PKM stellen Hao und Merlet [40] vor, die einige Anforderungen als Zwangsbedingungen umsetzen und mit Intervallanalyse Varianten der Maschine finden, die garantiert über einen vorgegebenen Arbeitsraum verfügen. Dieser Ansatz wird im Wesentlichen auch in dieser Arbeit verfolgt.

Eine verhältnismäßig neue Variante von PKM sind parallele Seilroboter, bei denen die starren Beine durch Seile ersetzt werden (Abb. 1.3 [46]). Diese wurden zunächst als intelligente Kräne für den Schiffbau vorgeschlagen [22]. Ming und Higuchi klassifizieren Seilroboter anhand der Anzahl der Seile und Freiheitsgrade [85, 86]. Um die Plattform vollständig kontrollieren zu können, müssen mehr Seile als Freiheitsgrade verwendet werden. Im Gegensatz zu starren PKM wird der Arbeitsraum von Seilrobotern durch die Forderung nach Zugkräften in den Seilen bestimmt [131, 132]. Bei Seilrobotern können die Kollisionen von Seilen eine große Rolle spielen [78]. Damit ist es für vollständig bestimmte Seilroboter nötig, die Kräfte geeignet unter den Seilen aufzuteilen [12, 13, 121, 133]. Die Regelung von vollständig bestimmten Seilrobotern wurde von Fang betrachtet [31]. Mai-

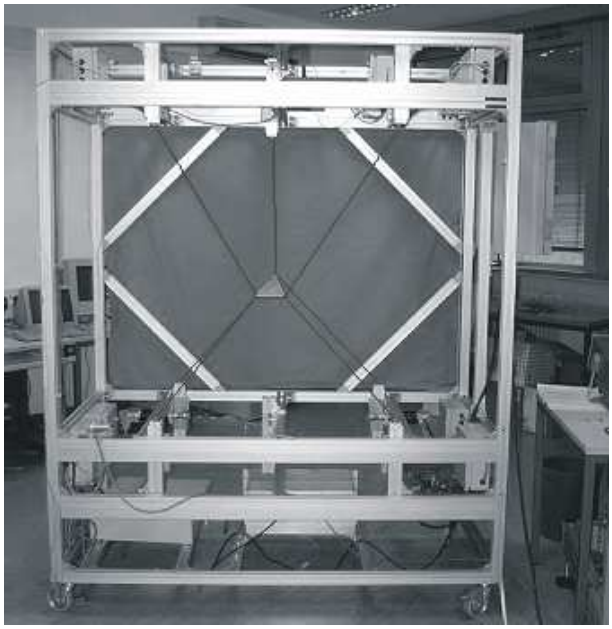


Abbildung 1.3: Prüfstand des Seilroboters *Segesta* am Lehrstuhl für Mechatronik, Universität Duisburg-Essen [46]

er [72] und Heyden [45] untersuchten unterbestimmte Handhabungssysteme mit weniger Seilen als Freiheitsgraden.

Die Monographie von Merlet [77] behandelt viele theoretische Aspekte paralleler Roboter und gibt einen umfassenden Überblick über die Literatur. Dagegen widmet sich Neugebauer schwerpunktmäßig den praktischen Aspekten des Entwurfs und der Nutzung von Parallelkinematik-Werkzeugmaschinen [91]. Grundlagen der Kinematik sind z. B. bei Bottema und Roth [7] sowie bei Husty et al. [51] zu finden. Diverse Forschungsarbeiten sind von Boër et al. zusammengestellt [6], während Heisel einen Überblick über das Schwerpunktprogramm 1099 'Fertigungsmaschinen mit Parallelkinematiken' der Deutschen Forschungsgemeinschaft gibt [44].

Die Erweiterung der Algebra reeller Zahlen zu Intervallen wurde von Moore vorgeschlagen [88] und ermöglicht die Konstruktion von numerischen Algorithmen mit erstaunlichen Eigenschaften. Eine Einführung in die numerische Mathematik mit Intervallen gibt Neumaier [93] und Ratschek [112]. Ursprünglich wurde die Intervallanalyse für den korrekten und robusten Umgang mit Rundungsfehlern verwendet. Allerdings ermöglicht das Intervallkalkül neuartige Algorithmen zur Lösung linearer und nichtlinearer Gleichungssysteme [92] und zur globalen Optimierung [5, 39, 135], denn es können garantierte Aussagen über kontinuierliche Mengen gemacht werden. In der jüngeren Vergangenheit entstanden eine Reihe von Arbeiten zur Analyse und Synthese von PKM, die diese spezielle Eigenschaft der Intervallanalyse ausnutzen [15, 16, 23, 24, 40, 42, 76, 79, 80, 83, 102, 103, 106, 138].

1.3 Ziel und Inhalt der Arbeit

In dieser Arbeit werden Methoden zur Maßsynthese von Werkzeugmaschinen mit paralleler Kinematik für gegebene Prozessanforderungen untersucht. Dazu wird zunächst ein

generisches Modell einer Maschine vorgeschlagen, das aus einem Baukasten von Komponenten zusammengestellt wird und das eine Analyse der kinematischen Eigenschaften anhand der kinetostatischen Methode erlaubt. Durch Kombination der Module kann leicht eine ganze Familie verschiedener PKM modelliert werden. Für diese Manipulatoren wird die Berechnung der inversen und direkten Kinematik automatisiert. Es wird ein allgemeines Verfahren vorgestellt, das eine vollständige Sensitivitätsanalyse und darauf aufbauend eine Untersuchung der Genauigkeit und Steifigkeit der Maschine erlaubt. Ferner wird skizziert, wie sich das generische Modell für die Kalibrierung von PKM verwenden lässt.

Die Berechnung und Verifizierung des Arbeitsraums einer PKM wird als *Constraint Satisfaction Problem* (CSP) formuliert und ausführlich anhand von Intervallanalyse untersucht. Besonderes Augenmerk wird dabei einer Methode gewidmet, die alle Arbeitsschritte innerhalb eines Programmgerüsts (Framework) zusammenfasst. Damit kann ein Anwender von einer vorläufigen Analyse bis hin zur Maßsynthese und Optimierung alle Berechnungen mit dem gleichen Modell durchführen. Durch den modularen Aufbau kann frei ausgewählt werden, welche Prozessanforderungen für die jeweilige Berechnung berücksichtigt werden. Bei der Maßsynthese wird die Menge aller Werte für die geometrischen Parameter einer Klasse von Maschinen bestimmt, so dass jede Maschine in dieser Menge über die vorgegebenen Eigenschaften verfügt. Anhand dieser Menge gültiger Varianten werden durch eine globale Optimierung Maschinen ermittelt, die eine ausgewählte Zielfunktion minimieren. Aufgrund der speziellen Eigenschaften der verwendeten Intervallanalyse wird diese Optimierung so gestaltet, dass das globale Minimum garantiert gefunden wird. Da die Algorithmen sehr rechenintensiv sind, wird eine Implementierung für parallele Computer vorgeschlagen.

In Kapitel 2 werden bekannte Methoden zusammengefasst, die für die Modellierung, Analyse und Synthese von Werkzeugmaschinen mit paralleler Kinematik verwendet werden. Zunächst wird die *kinetostatische Methode* zur Untersuchung der Kinematik und Dynamik allgemeiner Mehrkörpersysteme beschrieben. Im Anschluss daran folgt eine Einführung von Begriffen und Eigenschaften paralleler Roboter sowie eine Beschreibung der Geometrie der in dieser Arbeit untersuchten PKM. Im letzten Abschnitt des Kapitels werden die Grundlagen der Intervallanalyse dargestellt, die ein wesentliches Werkzeug zum verifizierten und kontinuierlichen Rechnen in dieser Arbeit bildet.

In Kapitel 3 werden Methoden zur kinematischen Untersuchung von PKM an diskreten Posen innerhalb des Arbeitsraums vorgestellt. Dazu wird zunächst die *kinetostatische Methode* verwendet, um ein modulares Modell für PKM zu erstellen, das die Untersuchung einer großen Klasse von PKM mit generischen Algorithmen gestattet. Für dieses generische Modell wird ein neuer Algorithmus zur vollständigen Linearisierung von PKM beschrieben. Anhand des linearisierten Modells wird der Einfluss von Fertigungsfehlern, die Steifigkeit und die Kalibrierung einer PKM analysiert. Beispiele für das modulare kinetostatische Modell und dessen Linearisierung werden am Ende des Kapitels vorgestellt. Weiterhin wird für die Maschine *Linapod* die Empfindlichkeit gegenüber geometrischen Fehlern und die Steifigkeit untersucht.

In Kapitel 4 wird eine kontinuierliche Methode verwendet, um den Arbeitsraum von PKM zu berechnen. Dazu wird die Berechnung des Arbeitsraums von PKM als CSP formuliert. Zur Lösung des CSP werden Algorithmen vorgestellt, die auf Intervallanalyse basieren. Die prozessbedingten Anforderungen werden als Bindungen für das CSP aufgestellt. Diese Bindungen berücksichtigen den kinematisch erreichbaren Arbeitsraum, Singularitäten,

Begrenzungen der aktiven und passiven Gelenke, die Übersetzungsverhältnisse zwischen Antrieben und Plattform sowie Kollisionen zwischen den Beinen. Weiterhin wird gezeigt, wie sich der CSP-Löser parallelisieren lässt. Beispiele für die verschiedenen Arten der Arbeitsraumberechnung von PKM werden am Ende des Kapitels gezeigt.

In Kapitel 5 wird die Methode zum Berechnen des Arbeitsraums so erweitert, dass sie für den Entwurf von PKM verwendet werden kann. Dazu wird die Maßsynthese für gegebene Prozessanforderungen als CSP formuliert. Im Anschluss wird dieses CSP mit einem Algorithmus zur globalen Optimierung kombiniert. Die ähnliche Struktur von Analyse und Synthese wird verwendet, um ein Programmgerüst zur vereinheitlichten Analyse, Synthese und Optimierung von PKM zu entwerfen. Beispiele für Synthese und Optimierung von PKM werden anhand dieses Programmgerüsts vorgestellt.

Die wesentlichen Ergebnisse dieser Arbeit werden in Kapitel 6 zusammengefasst, und es wird ein Ausblick auf mögliche Themen und Anwendungen weiterer Forschungsarbeiten gegeben.

1.4 Anmerkungen zur Schreibweise

Alle Formelzeichen werden bei ihrer ersten Verwendung im Text erläutert und sind zur Übersicht auf Seite VIII zusammengestellt. Allgemeine Konventionen werden im Folgenden ausgeführt.

Wie in der Literatur üblich werden Formeln in Matrizenschreibweise notiert. Skalare a werden in Normaldruck dargestellt, für physikalische Vektoren $\boldsymbol{a}$ und Tensoren 2. Stufe $\boldsymbol{A}$ wird der Fettdruck verwendet. Mathematische Vektoren[2] $\underline{a}$ werden mit einfachem Unterstrich, Matrizen $\underline{\underline{A}}$ mit den Komponenten von Tensoren 2. Stufe mit doppeltem Unterstrich gekennzeichnet. Sofern nicht anders angegeben werden Vektoren als einspaltige Matrizen interpretiert. Transponierte Größen $\underline{a}^{\mathrm{T}}$ werden mit $^{\mathrm{T}}$ gekennzeichnet. Vektoren und Matrizen werden entweder komponentenweise oder in Blockschreibweise in eckigen Klammern angegeben, z. B. $\underline{a} = [1, 3, 5]^{\mathrm{T}}$. Nachgestellte Indizes an der Vektorklammer wählen gegebenenfalls die Komponenten $[\underline{a}]_2 = 3$ aus. Multiplikationen zwischen Vektoren und Matrizen sind als Vektorprodukt zu interpretieren ($\underline{b} = \underline{\underline{A}}\,\underline{a}$), während mit $\underline{a} \cdot \underline{b}$ das Skalarprodukt notiert wird, für welches in Matrizenschreibweise $\underline{a} \cdot \underline{b} = \underline{a}^{\mathrm{T}} \underline{b}$ gilt. Das Kreuzprodukt $\underline{a} \times \underline{b}$ ist nur für $\underline{a}, \underline{b} \in \mathbb{R}^3$ definiert. Der Tilde-Operator $\tilde{}$ generiert aus einem Vektor $\underline{a} \in \mathbb{R}^3$ eine Matrix $\underline{\tilde{a}} \in \mathbb{R}^{3\times3}$, so dass $\underline{a} \times \underline{b} = \underline{\tilde{a}}\,\underline{b}$ gilt.

Die Menge der reellen Intervalle wird mit $\mathbb{I}$ bezeichnet und Intervallvariablen $\hat{a}$ werden mit einem Dach gekennzeichnet. Ein Vektor von Intervallen wird als Box bezeichnet und mit $\underline{\hat{a}}$ notiert.

Mengen werden mit kalligraphischen Großbuchstaben bezeichnet, z. B. für den Arbeitsraum $\mathcal{W}$, den Suchraum $\mathcal{X}$ oder für Listen von Boxen $\mathcal{L}$. Eine Ausnahme bildet $\mathcal{K}_i$, welches Koordinatensysteme bezeichnet, wobei der Index i das System auswählt.

[2] Ein mathematischer Vektor ist ein schlichtes n-Tupel von Elementen. Beispielsweise bilden in dieser Arbeit die Komponenten eines Ortsvektors ein Tripel und es wird implizit vorausgesetzt, dass ein orthonormiertes Koordinatensystem zur Darstellung verwendet wird.

Kapitel 2

Grundlagen

In diesem Kapitel werden bekannte Methoden zusammengefasst, die in dieser Arbeit für die Modellierung, Analyse und Synthese von Werkzeugmaschinen mit paralleler Kinematik verwendet werden. Abschnitt 2.1 beschreibt die kinetostatische Methode *zur Untersuchung der Kinematik allgemeiner Mehrkörpersysteme. In Abschnitt 2.2 folgt eine Einführung von Begriffen und Eigenschaften paralleler Roboter. Abschnitt 2.3 skizziert die Grundlagen der* Intervallanalyse *zum kontinuierlichen verifizierten Rechnen.*

2.1 Kinetostatische Methode

Parallelkinematikmaschinen (PKM) sind komplexe Mehrkörpersysteme (MKS), die durch das Auftreten kinematischer Schleifen gekennzeichnet sind. Für die Untersuchung der Kinematik von MKS wird in dieser Arbeit die *kinetostatische Methode* nach Kecskeméthy [59] verwendet. Der Grundgedanke bei der kinetostatischen Methode besteht in der Modellierung der Komponenten von MKS durch *Zustandsobjekte* und *kinetostatische Übertragungselemente*. Der Zustand eines mechanischen Systems setzt sich dabei aus der verallgemeinerten Position, Geschwindigkeit, Beschleunigung und Kraft zusammen. Unter einer verallgemeinerten Position wird hier sowohl ein Ortsvektor als auch eine Orientierung verstanden. Entsprechend umfassen verallgemeinerte Geschwindigkeiten und Beschleunigungen sowohl translatorische als auch rotatorische Ausdrücke. Die verallgemeinerten Kräfte beschreiben die Schnittkräfte und -momente, die zwischen den jeweiligen Bauteilen auftreten. Die Gelenke und Körper werden als elementare Bestandteile des Systems durch Übertragungselemente dargestellt. Diese bilden den kinetostatischen Zustand bestehend aus verallgemeinerter Position, Geschwindigkeit, Beschleunigung und Kraft an ihrem Eingang auf den Zustand am Ausgang ab. Die Übertragungselemente werden durch Übertragungsfunktionen für Position, Geschwindigkeit, Beschleunigung und Kraft charakterisiert, welche die jeweiligen kinematischen Größen vom Eingangskoordinatensystem $\mathcal{K}$ des Übertragungselements auf die Größen des Koordinatensystems am Ausgang $\mathcal{K}'$ abbilden (Abb. 2.1). Übertragungselemente besitzen daher eine ausgeprägte Übertragungsrichtung. Eine differentialgeometrische Interpretation der Kinetostatik wurde von Kecskeméthy gegeben [59].

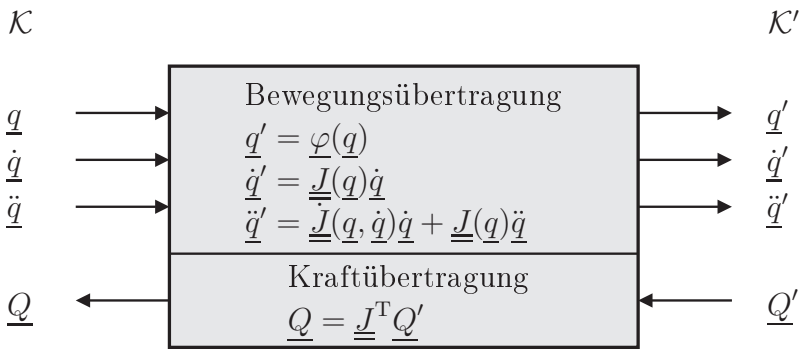


Abbildung 2.1: Allgemeines kinetostatisches Übertragungselement

2.1.1 Übertragungsfunktionen

Ein mechanisches Übertragungselement wird durch vier Funktionen für Positions-, Geschwindigkeits-, Beschleunigungs- und Kraftübertragung charakterisiert, die in einem engen Zusammenhang stehen. Für ein allgemeines Übertragungselement wird zunächst die Positionsübertragung betrachtet, aus der sich die übrigen Übertragungsfunktionen ableiten lassen. Bei der Positionsübertragung werden die Zustände am Eingang des Elements auf die Zustände am Ausgang abgebildet (Abb. 2.1). Diese Abbildung ist eine nichtlineare Funktion

$$\underline{q}' = \underline{\varphi}(\underline{q}), \tag{2.1}$$

welche die Ausgänge $\underline{q}'$ aus den Eingängen $\underline{q}$ ermittelt. Diese Funktion besitzt nicht notwendigerweise eine geschlossene analytische Darstellung. Durch Differentiation nach der Zeit ergibt sich daraus die lineare Geschwindigkeitsübertragung

$$\dot{\underline{q}}' = \frac{\partial \underline{\varphi}(\underline{q})}{\partial \underline{q}}\dot{\underline{q}} = \underline{\underline{J}}(\underline{q})\dot{\underline{q}} \quad \text{mit} \quad \left[\underline{\underline{J}}\right]_{ij} = \frac{\partial \varphi_i}{\partial q_j}, \tag{2.2}$$

wobei die Matrix der partiellen Ableitungen $\underline{\underline{J}}$ als JACOBI-Matrix bezeichnet wird. Diese JACOBI-Matrix stellt auch den wichtigen Zusammenhang $\delta\underline{q}' = \underline{\underline{J}}(\underline{q})\delta\underline{q}$ zwischen den virtuellen Verrückungen am Eingang und Ausgang her. Durch eine zweite Differentiation nach der Zeit ergibt sich die Beschleunigungsübertragung

$$\ddot{\underline{q}}' = \dot{\underline{\underline{J}}}(\underline{q},\dot{\underline{q}})\dot{\underline{q}} + \underline{\underline{J}}(\underline{q})\ddot{\underline{q}}. \tag{2.3}$$

Werden ideale Gelenke und damit eine verlustfreie Übertragung vorausgesetzt, lässt sich aus der Geschwindigkeitsübertragung die Kraftübertragung herleiten. Bei einer verlustfreien Übertragung muss die virtuelle Arbeit am Eingang und am Ausgang des Übertragungselements gleich sein und es ergibt sich

$$\delta\underline{q}^{\mathrm{T}}\underline{Q} = \delta\underline{q}'^{\mathrm{T}}\underline{Q}', \tag{2.4}$$

wobei für die virtuellen Verrückungen $\delta\underline{q}' = \underline{\underline{J}}\delta\underline{q}$ gilt. Daraus folgt $\delta\underline{q}^{\mathrm{T}}\underline{Q} = \delta\underline{q}^{\mathrm{T}}\underline{\underline{J}}^{\mathrm{T}}\underline{Q}'$, was für jedes beliebige $\delta\underline{q}$ erfüllt sein muss, so dass für die Kraftübertragung

$$\underline{Q} = \underline{\underline{J}}^{\mathrm{T}}\underline{Q}' \tag{2.5}$$

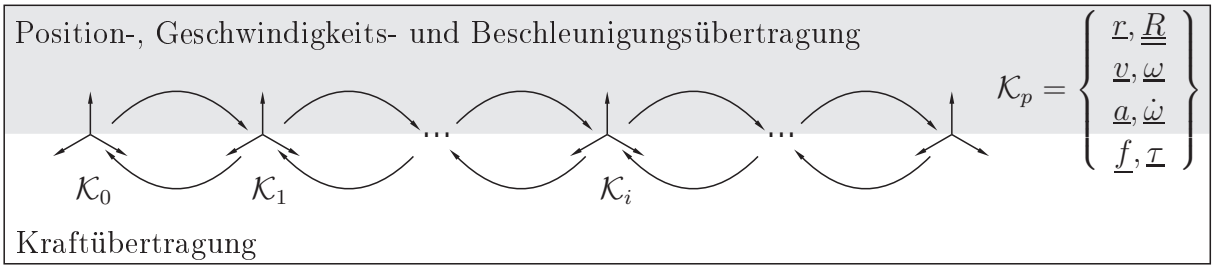


Abbildung 2.2: Übertragung kinematischer Zustände in einer Kette von Übertragungselementen

gilt. Diese Analogie zwischen der Geschwindigkeitsübertragung (2.2) und der Kraftübertragung (2.5) wird als *kinetostatischer Dualismus* bezeichnet. Da die JACOBI-Matrix $\underline{\underline{J}}$ nicht notwendigerweise quadratisch ist, kann $\underline{\underline{J}}$ im Allgemeinen auch nicht invertiert werden. Daher ist die natürliche Übertragungsrichtung der Kräfte entgegengesetzt zur Bewegungsübertragung.

Bei einem Übertragungselement werden die Zustände von einem Koordinatensystem $\mathcal{K}$ auf ein anderes $\mathcal{K}'$ abgebildet. Dieses Koordinatensystem kann wiederum als Eingang für ein weiteres Übertragungselement verwendet werden, so dass eine *kinematische Kette* entsteht (Abb. 2.2), welche die Beschreibung komplexer Mechanismen erlaubt.

2.1.2 Allgemeines Gelenk mit sechs Freiheitsgraden

Im Folgenden wird der Aufbau eines kinetostatischen Übertragungselements anhand eines Gelenks mit sechs Freiheitsgraden[1] veranschaulicht (Abb. 2.3). Für die vollständige Beschreibung des Übertragungselements müssen die Übertragungsfunktionen (2.1)–(2.3) sowie (2.5) definiert werden. Das hier betrachtete Übertragungselement beinhaltet drei translatorische Freiheitsgrade $\underline{x}$ und drei rotatorischen Freiheitsgraden $\underline{\theta}$, wobei die Rotationsmatrix $\underline{\underline{R}}(\underline{\theta})$ beispielhaft durch die KARDAN-Winkel $\underline{\theta}$ parametrisiert wird. Die Zustände am Koordinatensystem $\mathcal{K}_i$ werden durch die Position $\underline{r}_i$ sowie durch die Orientierung $\underline{\underline{R}}_i$ dargestellt. Die Positionsübertragung ergibt sich dann zu

$$\underline{\underline{R}}_{i+1} = \underline{\underline{R}}_i \underline{\underline{R}}(\underline{\theta}) \,, \tag{2.6}$$

$$\underline{r}_{i+1} = \underline{\underline{R}}(\underline{\theta}) \underline{r}_i + \underline{x}. \tag{2.7}$$

Die Übertragung der Geschwindigkeiten $\underline{\omega}$ und $\underline{v}$ ist stets linear und es folgt

$$\begin{bmatrix} \underline{\omega}_{i+1} \\ \underline{v}_{i+1} \end{bmatrix} = \underbrace{\begin{bmatrix} \underline{\underline{R}}(\underline{\theta})^{\mathrm{T}} & \underline{\underline{0}} & \underline{\underline{I}}_3 & \underline{\underline{0}} \\ -\underline{\widetilde{x}}\, \underline{\underline{R}}(\underline{\theta})^{\mathrm{T}} & \underline{\underline{R}}(\underline{\theta})^{\mathrm{T}} & \underline{\underline{0}} & \underline{\underline{I}}_3 \end{bmatrix}}_{\underline{\underline{J}}_{\mathrm{lokal}}} \begin{bmatrix} \underline{\omega}_i \\ \underline{v}_i \\ \dot{\underline{\theta}} \\ \dot{\underline{x}} \end{bmatrix}. \tag{2.8}$$

[1] Rein technisch ist ein Gelenk mit sechs Freiheitsgraden sinnlos, denn es impliziert keinerlei Bindungen für das System. Formal lässt sich diese Gelenk zum einen für die Modellierung der Bewegung der Plattform bei der inversen Kinematik verwendet. Zum anderen bildet es einen wesentlichen Baustein zur Beschreibung der Linearisierung in Kapitel 3.3. Weiterhin lassen sich alle elementaren Gelenke durch Spezialisierung aus diesem allgemeinen Fall ableiten.

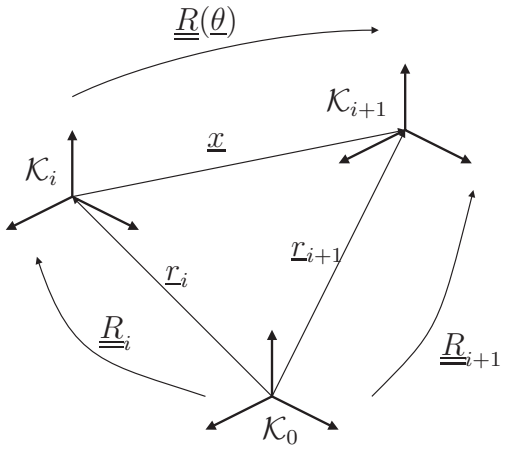


Abbildung 2.3: Relativkinematik für ein Gelenk mit sechs Freiheitsgraden

Die Beschleunigungen $\dot{\underline{\omega}}$ und $\underline{a}$ ergeben sich schließlich zu

$$\begin{bmatrix} \dot{\underline{\omega}}_{i+1} \\ \underline{a}_{i+1} \end{bmatrix} = \begin{bmatrix} \underline{\omega}_i \times \dot{\underline{\theta}} \\ 2\underline{\omega}_i \times \dot{\underline{x}} + \underline{\omega}_i \times (\underline{\omega}_i \times \underline{x}) \end{bmatrix} + \underbrace{\begin{bmatrix} \underline{\underline{R}}(\underline{\theta})^{\mathrm{T}} & \underline{\underline{0}} & \underline{\underline{I}}_3 & \underline{\underline{0}} \\ -\underline{\tilde{x}}\,\underline{\underline{R}}(\underline{\theta})^{\mathrm{T}} & \underline{\underline{R}}(\underline{\theta})^{\mathrm{T}} & \underline{\underline{0}} & \underline{\underline{I}}_3 \end{bmatrix}}_{\underline{\underline{J}}_{\text{lokal}}} \begin{bmatrix} \dot{\underline{\omega}}_i \\ \underline{a}_i \\ \ddot{\underline{\theta}} \\ \ddot{\underline{x}} \end{bmatrix}. \quad (2.9)$$

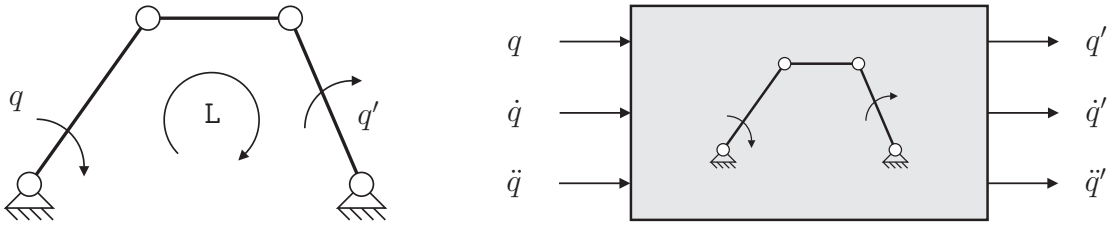
Unter Ausnutzung des kinetostatischen Dualismus wird die Kraftübertragung unmittelbar aus der Geschwindigkeitsübertragung abgeleitet. Dabei werden die Zustandsgrößen der Geschwindigkeit $\underline{\omega}, \underline{v}$ gegen die entsprechenden Momente/Kräfte $\underline{\tau}, \underline{f}$ ausgetauscht und anstelle der lokalen JACOBI-Matrix $\underline{\underline{J}}_{\text{lokal}}$ wird deren Transponierte eingesetzt. Damit bestimmen sich die Kräfte aus

$$\begin{bmatrix} \underline{\tau}_i \\ \underline{f}_i \\ \underline{Q}^{(\theta)} \\ \underline{Q}^{(x)} \end{bmatrix} = \underbrace{\begin{bmatrix} \underline{\underline{R}}(\underline{\theta}) & \underline{\underline{R}}(\underline{\theta})\underline{\tilde{x}} \\ \underline{\underline{0}} & \underline{\underline{R}}(\underline{\theta}) \\ \underline{\underline{I}}_3 & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{I}}_3 \end{bmatrix}}_{\underline{\underline{J}}^{\mathrm{T}}_{\text{lokal}}} \begin{bmatrix} \underline{\tau}_{i+1} \\ \underline{f}_{i+1} \end{bmatrix}, \quad (2.10)$$

wobei $\underline{Q}^{(\theta)}$ und $\underline{Q}^{(x)}$ die verallgemeinerten Momente/Kräfte des Gelenks darstellen. Die elementaren Bausteine, wie starre Körper sowie Schub- und Drehgelenke, können leicht spezialisiert werden, indem für den Vektor $\underline{x}$ und die Rotationsmatrix $\underline{\underline{R}}(\underline{\theta})$ entsprechend vereinfachte Terme eingesetzt werden. Zur übersichtlichen Notation der kinetostatischen Zustände werden diese kompakt als Koordinatensystem $\mathcal{K}_i = \left\{\underline{\underline{R}}_i, \underline{r}_i, \underline{\omega}_i, \underline{v}_i, \dot{\underline{\omega}}_i, \underline{a}_i, \underline{\tau}_i, \underline{f}_i\right\}$ bezeichnet.

2.1.3 Lösung kinematischer Schleifen

Bisher wurde die Übertragung von Bewegungen und Kräften für kinematische Ketten betrachtet. Für komplexe MKS wie PKM ist das Auftreten von *kinematischen Schleifen* charakteristisch, d.h. es gibt Körper im System, die über verschiedene Wege mit dem Inertialsystem verbunden sind (Abb. 2.4a). Die Bewegungen der Gelenke innerhalb einer



a) Kinematische Schleife L eines ebenen Gelenkvierecks

b) Modellierung der Kinematik der Schleife L als Übertragungselement

Abbildung 2.4: Kinematische Schleife als Übertragungselement

Schleife sind nicht mehr unabhängig voneinander, sondern unterliegen Zwangsbedingungen, die sich aus dem Schließen der kinematischen Schleifen ergeben. Diese Abhängigkeiten können durch geeignete Bindungsgleichungen herausgearbeitet und im Rahmen der kinetostatischen Methode als Übertragungsfunktionen modelliert werden. Folglich werden kinematische Schleifen als kinetostatische Übertragungselemente dargestellt, deren Übertragungsfunktionen und innere Zustände sich berechnen lassen (Abb. 2.4b). Für einige Typen einfacher kinematischer Schleifen mit einem Freiheitsgrad kann dies sogar in einer expliziten Form erfolgen. Bei allgemeinen Schleifen wird ein NEWTON-RAPHSON-Verfahren verwendet, um die nichtlinearen Übertragungsfunktionen iterativ zu approximieren, so dass der vollständige kinetostatische Zustand am Ausgang bestimmt werden kann. Aus der Sicht des Gesamtsystems werden kinematische Schleifen als *kinematischer Transformator* modelliert [48], der als Übertragungsblock mit einem definierten Ein-/Ausgangsverhalten behandelt wird und die komplexen Details der Schleife kapselt.

2.1.4 Modularität

Das Konzept des kinetostatischen Übertragungselements kann dahingehend verallgemeinert werden, mehrere Übertragungselemente zu einem sogenannten *Superelement* zusammenzufassen (Abb. 2.5). Dabei erweist es sich als vorteilhaft, dass eine Kette von Übertragungselementen wiederum wie ein einzelnes Element behandelt werden kann. Als einfaches Beispiel dafür können die Gelenke mit mehreren Freiheitsgraden betrachtet werden. Dabei werden elementare Drehgelenke zusammengefügt, um beispielsweise Kardan- und Kugelgelenke nachzubilden. Die Bildung solcher Subsysteme ist jedoch nicht auf einzelne Gelenke beschränkt, sondern kann auch zur Abbildung ganzer Baugruppen verwendet werden. In diesem Zusammenhang muss jedoch beachtet werden, dass das Zusammenfügen von Subsystemen nur für MKS mit Baumstruktur ohne zusätzlichen Aufwand durchgeführt werden kann. Bei Mechanismen mit kinematischen Schleifen gibt es im Allgemeinen keine geschlossene Lösung der Kinematik und der implizite Kern kann nicht trivial ermittelt werden. Eine spezielle Möglichkeit zur Ermittelung einer Lösung für PKM wird in Abschnitt 3.2 vorgestellt.

2.1.5 Kinematische Differentiale

Durch die Verkettung der Übertragungselemente entsteht ein Superelement für das Gesamtsystem. In Folge des modularen Aufbaus dieses Superelements liegen nur die lokalen

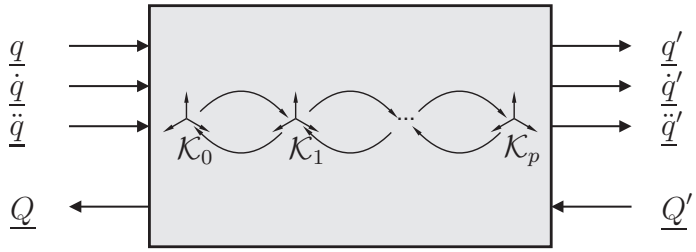


Abbildung 2.5: Eine Kette von Übertragungselementen kann als Superelement abgebildet werden.

JACOBI-Matrizen $\underline{\underline{J}}_i$ vor, während die JACOBI-Matrix des gesamten Manipulators $\underline{\underline{J}}$ nicht explizit bekannt ist. Beinhaltet das Superelement kinematische Schleifen mit impliziten Schließbedingungen, ist es nicht möglich, die Elemente von $\underline{\underline{J}}$ analytisch anzugeben. Viele praktische Anwendungen erfordern aber die Kenntnis der JACOBI-Matrix. Mit dem Verfahren der *kinematischen Differentiale* kann durch Auswertung der Geschwindigkeitsübertragungsfunktion (2.2) die JACOBI-Matrix numerisch rekonstruiert werden. Das Übertragungselement wird dabei als Black-Box betrachtet, das für gegebene Eingangsgrößen $\underline{q}, \underline{\dot{q}}$ die Ausgangsgrößen $\underline{\dot{q}}' = \underline{\dot{\varphi}}(\underline{q}, \underline{\dot{q}})$ berechnet. Die globale Geschwindigkeitsübertragungsfunktion $\underline{\dot{\varphi}}(\underline{q}, \underline{\dot{q}})$ hat die bekannte lineare Form $\underline{\underline{J}}(\underline{q})\underline{\dot{q}}$ nach Gleichung (2.2), so dass sich die Matrix $\underline{\underline{J}}$ folgendermaßen rekonstruieren lässt. Wird die Funktion $\underline{\dot{\varphi}}(\underline{q}, \underline{\dot{q}})$ für einen gegebenen Vektor verallgemeinerter Koordinaten $\underline{q}_0$ und die *Pseudogeschwindigkeit* $\check{\underline{\dot{q}}}^{(j)}$ mit

$$\check{\dot{q}}_i^{(j)} = \begin{cases} 1 & \text{für } i = j \\ 0 & \text{sonst} \end{cases} \tag{2.11}$$

ausgewertet, ergibt sich genau die j-te Spalte $[\underline{\underline{J}}]_j$ der JACOBI-Matrix zu

$$[\underline{\underline{J}}]_j = \underline{\dot{\varphi}}(\underline{q}_0, \check{\underline{\dot{q}}}^{(j)}). \tag{2.12}$$

Wird diese Auswertung für jeden Einheitsvektor $\check{\underline{\dot{q}}}^{(j)}, j = 1, \ldots, n$ wiederholt, lässt sich sukzessiv die Matrix $\underline{\underline{J}}$ spaltenweise berechnen. Dabei bezeichnet n die Anzahl der betrachteten verallgemeinerten Koordinaten.

Die JACOBI-Matrix $\underline{\underline{J}}$ kann aufgrund des kinetostatischen Dualismus auch aus der Kraftübertragung bestimmt werden. Diese weist nach Gleichung (2.5) die gleiche Struktur wie die Geschwindigkeitsübertragung auf. Das Konzept der kinematischen Differentiale lässt sich anwenden, indem zu den Pseudogeschwindigkeiten die Pseudokräfte $\check{\underline{Q}}'^{(j)}$ eingeführt werden, so dass sich anhand von

$$[\underline{\underline{J}}^{\mathrm{T}}]_j = \underline{Q}(\underline{q}_0, \check{\underline{Q}}^{(j)}) \tag{2.13}$$

die JACOBI-Matrix zeilenweise berechnen lässt. Der Aufwand für die Berechnung der Geschwindigkeitsübertragung und der Kraftübertragung ist etwa gleich groß. Im allgemeinen ist jedoch die JACOBI-Matrix $\underline{\underline{J}}$ nicht quadratisch, so dass die beiden Methoden unterschiedlich viele Auswertungen der Übertragungsfunktionen erfordern und die Berechnung dahingehend optimiert werden kann. Wie in Abschnitt 3.3 gezeigt wird, lässt sich dieser Algorithmus zur vollständigen Linearisierung von MKS für PKM einsetzen. Es zeigt sich, dass das linearisierte Modell mit der kraftgestützten Methode bei PKM um ein vielfaches schneller berechnet werden kann als mit der geschwindigkeitsgestützten Methode.

2.1.6 Dynamik

Neben der Kinematik spielt die Dynamik für die Simulation und Auslegung von MKS eine wichtige Rolle. Die kinetostatische Methode bietet eine interessante Möglichkeit, die Bewegungsgleichungen von allgemeinen MKS durch mehrfache Auswertung der Kinematik aufzustellen und damit den engen Zusammenhang zwischen Kinematik und Dynamik auszunutzen [47]. Das Konzept der kinematischen Differentiale lässt sich auch hier anwenden. Dabei werden die Übertragungsfunktionen des Mechanismus für spezielle Pseudogeschwindigkeiten und -beschleunigungen ausgewertet, um mit den so ermittelten verallgemeinerten Kräften die Koeffizienten der Bewegungsgleichungen zusammenzustellen. Dies entspricht der Methode der kinematischen Differentiale, wie sie in Abschnitt 2.1.5 beschrieben wurde. Da die Bewegungsgleichungen numerisch berechnet werden, für die Berechnung zum Teil aber explizite Methoden verwendet werden, wird dieses Verfahren auch als *symbolisch-numerisch* bezeichnet. Die Bewegungsgleichungen eines MKS in Minimalkoordinaten sind ein System gewöhnlicher Differentialgleichungen 2. Ordnung

$$\underline{\underline{M}}(\underline{q})\ddot{\underline{q}} + \underline{g}^C(\underline{q}, \dot{\underline{q}}) = \underline{Q}, \tag{2.14}$$

wobei mit $\underline{\underline{M}}(\underline{q}) \in \mathbb{R}^{n_f \times n_f}$ die verallgemeinerte Massenmatrix bezeichnet wird, $\underline{g}^C(\underline{q}, \dot{\underline{q}}) \in \mathbb{R}^{n_f}$ den Vektor der verallgemeinerten Coriolis-, Zentrifugal- und Kreiselkräfte darstellt und $\underline{Q} \in \mathbb{R}^{n_f}$ die verallgemeinerten eingeprägten Kräfte repräsentiert. Die Zahl der Freiheitsgrade ist n_f. Der Zusatz *verallgemeinert* beschreibt die Tatsache, dass der Vektor $\underline{q}$ sowohl translatorische als auch rotatorische Komponenten enthalten kann und dass die Kräfte $\underline{g}^C$ und $\underline{Q}$ von Weltkoordinaten auf Minimalkoordinaten transformiert wurden. Da die Bewegungsgleichungen für die Ziele dieser Arbeit keine Rolle spielen, wird für die Berechnung auf die Literatur verwiesen [59]. Das in Abschnitt 3.2 vorgestellte modulare Modell bietet jedoch alle benötigten Informationen, um die Bewegungsgleichungen aufzustellen und zu lösen. Somit lässt sich für weiterführende Untersuchungen ein Modell der Dynamik bereitstellen.

2.1.7 Implementierung

Der kinetostatische Ansatz zur Modellierung von MKS wurde in der objektorientierten C++-Bibliothek MOBILE implementiert [60], welche bereits zur Lösung von einer Reihe von kinematischen und dynamischen Problemen eingesetzt wurde [70, 87, 120]. Die offene Schnittstelle der C++-Bibliothek erlaubt eine Erweiterung um neue Klassen. Daher wird in Kapitel 3 ein Baukasten vorgeschlagen, der die Modellierung diverser PKM mit automatisiertem Aufstellen der Kinematik und Dynamik ermöglicht. Sofern nicht anders angegeben, wurden die kinematischen Berechnungen in der vorliegenden Arbeit anhand dieser Erweiterung von MOBILE durchgeführt.

2.2 Parallele Roboter

Klassische Industrieroboter (Abb. 2.6a) besitzen einen seriellen Aufbau, d. h. die Gelenke und Körper sind in einer unverzweigten Kette miteinander verbunden. Der *Tool Center Point* (TCP) wird durch genau eine Kette von Körpern und Gelenken bewegt, und es ist für die vollständige Bestimmtheit des Systems erforderlich, jedes Gelenk durch einen

a) Serieller Industrieroboter

b) Paralleler Roboter

Abbildung 2.6: Verschiedene Manipulatoren

Aktuator anzutreiben. Serielle Roboter besitzen daher keine passiven Gelenke. Da jedes Gelenk und damit jeder Aktuator alle folgenden Bauteile trägt und antreibt, werden die benötigten Antriebe beginnend vom TCP aus größer und damit schwerer. Dieser Aufbau begrenzt in natürlicher Weise die erreichbare Dynamik, d. h. er limitiert die maximal erreichbaren Geschwindigkeiten und Beschleunigungen.

2.2.1 Klassifikation

Besitzt ein Manipulator mehr als eine *kinematische Kette*, welche die Basis mit der beweglichen Plattform (End-Effektor) verbindet, so wird ein solcher Mechanismus *paralleler Roboter* genannt (Abb. 2.6b). Als *Rahmen* wird das Maschinenbett bezeichnet, das dem Manipulator als Basis dient. Die *bewegliche Plattform* trägt den zu manipulierenden Körper bzw. bei einer Werkzeugmaschine das Werkzeug, wobei der TCP einen speziellen Punkt als Referenz definiert. Für räumliche Manipulatoren muss auch die Orientierung der Plattform berücksichtigt werden. Die verbindenden kinematischen Ketten heißen *Beine*. Im Folgenden wird von der Definition von Merlet [77] ausgegangen:

Ein paralleler Roboter ist ein Mechanismus mit geschlossenen kinematischen Schleifen, der eine bewegliche Plattform (End-Effektor) über mehrere unabhängige kinematische Ketten mit einer Basis verbindet. Dabei besitzt die Plattform genau so viele Freiheitsgrade wie Aktuatoren und bei festgehaltenen Aktuatoren ist die Lage der Plattform vollständig bestimmt.

Bei parallelen Robotern treten also stets kinematische Schleifen auf, welche Abhängigkeiten zwischen den Gelenken entstehen lassen. Diese Definition umfasst eine sehr große Anzahl möglicher Architekturen, die im Rahmen einer Arbeit nicht umfassend betrachtet werden können. Daher wird der Begriff des vollständig parallelen Roboters eingeführt [77]:

Ein vollständig paralleler Roboter besitzt genau so viele kinematische Ketten, welche die Plattform mit der Basis verbinden, wie der Roboter Freiheitsgrade hat.

Damit gehören PKM per Definition der Klasse der komplexen MKS an und in dieser Arbeit werden nur solche PKM betrachtet, die der Definition eines vollständig parallelen Roboters genügen. Der spezielle Aufbau von PKM bringt im Vergleich zu seriellen Robotern eine Reihe von systeminhärenten Vor- und Nachteilen mit sich. Der Arbeitsraum einer PKM ist, verglichen mit den äußeren Abmessungen des Roboters, eher klein. Weiterhin wird der Arbeitsraum in höherem Maße durch Autokollisionen eingeschränkt, d. h. Kontakte zwischen den beweglichen Teilen wie den Beinen müssen vermieden werden. Diese Kollisionen limitieren insbesondere die möglichen Rotationen der Plattform. Obwohl parallele Roboter eine Reihe gleicher Baugruppen aufweisen, ist die Gesamtzahl der Einzelteile vor allem aufgrund der zahlreichen passiven Gelenke deutlich größer. Für die Modellierung der Kinematik und Dynamik paralleler Roboter sind deutlich aufwendigere Verfahren notwendig, da für die direkte Kinematik in vielen Fällen keine geschlossene Lösung gefunden werden kann. Schließlich verändern sich viele Eigenschaften wie Steifigkeit, Genauigkeit und Manipulierbarkeit stark über dem Arbeitsraum.

Auf der anderen Seite hat der parallele Aufbau erhebliche Vorteile bezüglich der strukturellen Steifigkeit, des Verhältnisses zwischen bewegter Masse und Nutzlast sowie der möglichen Dynamik des Systems. Durch die parallele Anordnung der Aktuatoren wird die Last unter den Antrieben verteilt, so dass insgesamt größere Massen bewegt werden können. Die Beine sind nur auf Zug/Druck belastet und thermische Fehler wirken sich hauptsächlich als Längenänderungen aus. Dies erlaubt eine vergleichsweise einfache Kompensation. Da hauptsächlich translatorische Antriebe zum Einsatz kommen, kann gegenüber rotatorischen Antrieben auf genauere Sensoren zurückgegriffen werden. Die Genauigkeit von PKM ist bezüglich der Fehler in den Antrieben sehr gut, d. h. Regelfehler wirken sich weniger stark auf den TCP aus als bei seriellen Robotern. Dieser Vorteil gilt jedoch nicht für die Maschine als Gesamtsystem. Wie in Abschnitt 3.7.3 ausführlich gezeigt wird, sind PKM als Gesamtsystem dagegen eher ungenau, wenn es nicht gelingt, die Fehler in den systembedingt zahlreichen Komponenten zu berücksichtigen.

Wenn nicht anders angegeben, werden in dieser Arbeit nur parallele Roboter betrachtet, die über Beine gleicher Topologie[2] verfügen. Diese Art von topologisch symmetrischen Maschinen tritt in der Praxis am häufigsten auf. Dennoch lassen sich viele Betrachtungen auch leicht auf allgemeinere Maschinen anwenden.

2.2.2 Topologische Analyse

Es gibt zahlreiche Architekturen für vollständig parallele Roboter. Für die Praxis sind vor allem PKM interessant, die sich aus Dreh-, Kugel-, Kardan- und Schubgelenken zusammenstellen lassen. Die Gelenke mit nur einem Freiheitsgrad werden dabei durch Aktuatoren angetrieben. Die Schubgelenke werden meist durch Linearantriebe, Kugelrollspindeln oder Hydraulik-/Pneumatikzylinder angetrieben. Für parallele Mikromanipulatoren werden häufig Piezokeramiken als Aktuator verwendet [32]. Aktive Drehgelenke werden in der Regel direkt durch Elektromotoren betrieben. Aus der Kombination dieser Bauteile entsteht eine begrenzte Anzahl an Modulen, die im Folgenden zu einem Baukasten

[2] Die *Topologie* eines Mechanismus beschreibt die Abfolge von Gelenken und Körpern und verwendet daher keine Metrik zur Beschreibung. Im Gegensatz dazu beschreibt die *Geometrie* die Abmessungen aller beteiligten Bauteile.

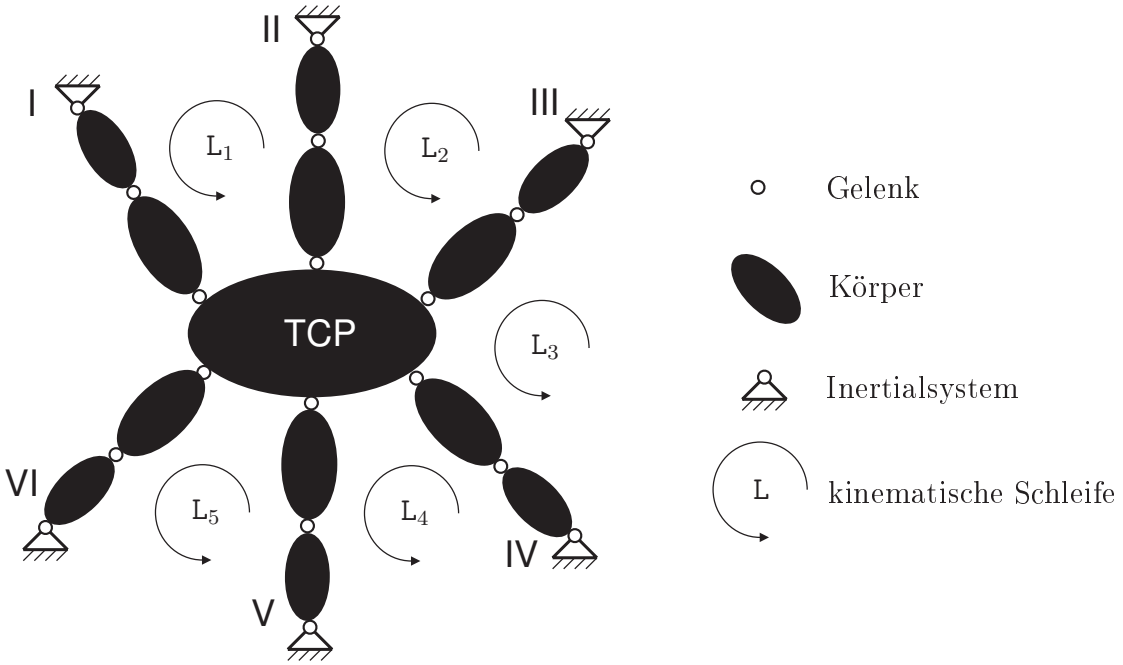


Abbildung 2.7: Topologische Struktur eines räumlichen vollständig parallelen Manipulators

zusammengefasst werden. Weiterhin wird davon ausgegangen, dass aus Fertigungs- und Kostengründen eine Maschine aus topologisch gleichen Modulen aufgebaut wird.

Die drei wichtigsten Module (siehe Abb. 2.9, Seite 21) bestehen aus den Gelenkfolgen `PUS` (prismatic, universal, spherical), `UPS` (universal, prismatic, spherical) und `RUS` (revolute, universal, spherical). Die Varianten `PSU`, `SPU` und `RSU` besitzen das gleiche kinematische Verhalten wie die oben genannten Module und müssen daher nicht gesondert betrachtet werden. Stattdessen wird per Konvention unterstellt, dass das Kugelgelenk des Beins an der Plattform angebracht wird. Die klassische Gough-Plattform besteht aus sechs Modulen vom `UPS`-Typ und ist vor allem durch die Anwendung bei Flug- und Fahrsimulatoren bekannt [11, 126]. Zuletzt wurden aber auch eine Reihe von Werkzeugmaschinen dieses Typs gebaut und untersucht [4, 43, 129, 130]. Mögliche Designs mit `PUS`-Beinen sind der *Linapod* [108, 139, 140] oder der Roboter von INRIA [77, p. 41]. `RUS`-Beine bilden schließlich den Grundbaustein für den Delta Roboter [19], der bei Pick & Place-Anwendungen kommerziell erfolgreich eingesetzt wird.

Alle drei oben genannten Module für die Beine einer PKM besitzen jeweils drei Gelenke, die mit zwei starren Körpern verbunden sind und besitzen insgesamt sechs Gelenkfreiheitsgrade pro Bein (Tabelle 2.1). Weiterhin werden sechs solcher Beinmodule zusammen mit einer Plattform, die als starrer Körper gewählt wird zusammen mit einem Rahmen[3] zu einer Maschine zusammengestellt (Abb. 2.7). Für das Grübler-Kurzbach-Kriterium ergeben sich somit die Anzahl der Körper zu $n_B = 6 \times 2 + 1 = 13$ und die Anzahl der Gelenke zu $n_G = 6 \times 3 = 18$. Insgesamt besitzen die Gelenke $\sum n_{G_i} = 36$ Gelenkfreiheitsgrade. Die Anzahl der unabhängigen kinematischen Schleifen folgt somit zu $n_L = n_G - n_B = 5$. Damit verfügen alle räumliche Maschinen dieses Baukastens über

[3]Der Rahmen stellt das Bezugssystem dar und wird nicht als starrer Körper mitgezählt.

Tabelle 2.1: Gelenkfreiheitsgrade n_{G_i} und Anzahl der Körper n_B für die Beinmodule

Beinmodul	starre Körper n_B	Schub-gelenke P	Dreh-gelenke R	Kardan-gelenke U	Kugel-gelenke S	Freiheits-grade $\sum n_{G_i}$
PUS	2	1	0	1	1	6
UPS	2	1	0	1	1	6
RUS	2	0	1	1	1	6

$n_f = \sum n_{G_i} - 6n_L = 36 - 6 \times 5 = 6$ Freiheitsgrade der Plattform. Diese werden unabhängig von der Kombination der drei Grundmodule erreicht.[4] Dies entspricht der vollständigen Beweglichkeit der Plattform im dreidimensionalen Raum.

Diese Betrachtung gilt nur für die direkte Kinematik. Dagegen liefert eine Analyse der inversen Kinematik ein leicht abweichendes Bild. In diesem Fall ist die Position und Orientierung der Plattform bekannt, so dass die Plattform im Sinne des GRÜBLER-KURZBACH-Kriteriums nicht mehr als Körper gezählt werden darf. Damit erhöht sich die Zahl der Schleifen auf $n_L = 6$, so dass gleichzeitig die Zahl der Freiheitsgrade auf $n_f = 0$ sinkt. Dies entspricht gerade der Tatsache, dass der Manipulator bei gegebener Position und Orientierung der Plattform vollständig bestimmt ist. In diesem Fall sind für jede Schleife alle notwendigen Informationen gegeben, so dass sich diese Schleifen unabhängig voneinander lösen lassen.

2.2.3 Kinematik und Geometrie der Beine

Die hier betrachteten Beinmodule verfügen alle über eine Kugel-/Kardangelenkpaarung. Für die Berechnung der Kinematik ist es zweckmäßig, das MKS an diesen Stellen aufzutrennen und den Abstand zwischen den beiden Gelenken als Bindungsgleichung einzuführen. Dies entspricht der Anwendung der Methode des *charakteristischen Gelenkpaars* nach Woernle [137] auf PKM. Für die hier behandelten Beinmodule PUS, UPS und RUS wird ein generisches kinematisches Modell der Maschine erstellt. Durch das Auftrennen des Mechanismus an den Kugel-/Kardangelenkpaaren wird die komplexe Kinematik des MKS durch ein System von sechs nichtlinearen Bindungsgleichungen beschrieben. Die kinematischen Beziehungen werden im Folgenden in physikalischen Vektoren formuliert, so dass die Wahl geeigneter Bezugssysteme und Parametrisierungen zunächst nicht zu erfolgen braucht. Für jedes Bein ergibt sich der geschlossene Vektorzug (Abb. 2.8)

$$\boldsymbol{a}_i + \boldsymbol{l}_i = \boldsymbol{r} + \boldsymbol{R}\boldsymbol{b}_i, \quad i = 1, \ldots, 6, \tag{2.15}$$

wobei $\boldsymbol{a}_i$ der Ortsvektor des Punkts A_i ist, $\boldsymbol{b}_i$ die relative Position des Punkts B_i bezüglich des TCP im Koordinatensystem $\mathcal{K}_p$ bezeichnet, $\boldsymbol{r}$ den Ortsvektor des TCP $\mathcal{K}_p$ und $\boldsymbol{R}$ die Orientierung der Plattform $\mathcal{K}_p$ bezüglich $\mathcal{K}_0$ beschreibt. Diese Gleichung wird nach dem Abstand $l_i^2 = \boldsymbol{l}_i^2$ zwischen den Punkten A_i und B_i aufgelöst, so dass sich ein System nichtlinearer Bindungsgleichungen ergibt

$$(\boldsymbol{r} + \boldsymbol{R}\boldsymbol{b}_i - \boldsymbol{a}_i)^2 - l_i^2 = 0, \quad i = 1, \ldots, 6. \tag{2.16}$$

[4] Dabei wird vorausgesetzt, dass die Maschine eine geeignete Geometrie besitzt, also nicht bereits architektonisch singulär ist.

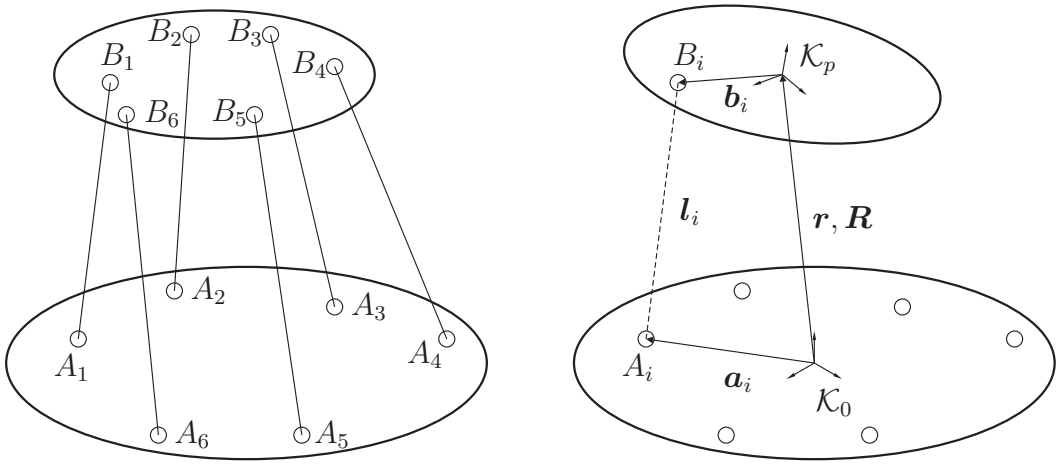


Abbildung 2.8: Generisches Modell einer räumlichen PKM mit sechs Freiheitsgraden

Die Weltkoordinaten $\boldsymbol{y}$ bestehen aus einer Parametrisierung der Position und Orientierung der Plattform und treten dabei in Form des Ortsvektors $\boldsymbol{r}$ und des Drehtensors $\boldsymbol{R}$ in Erscheinung. Die Geometrie der Plattform wird durch die Vektoren $\boldsymbol{b}_i$ beschrieben, welche die Position der Punkte B_i relativ zum plattformfesten Koordinatensystem $\mathcal{K}_p$ in einer beliebigen Anfangslage beschreiben. Im Folgenden wird für jedes Beinmodul der Zusammenhang zwischen den Größen $l_i, \boldsymbol{a}_i$ und den verallgemeinerten Koordinaten $\underline{q}$ beschrieben.

Bei einem UPS-Bein ist ein linearer Antrieb zwischen einem Kugel-/Kardangelenkpaar eingespannt. Das UPS-Bein entspricht dem ursprünglichen Entwurf von Gough [36] und wurde vor allem bei Plattformen für Flug- und Fahrsimulatoren [11] verwendet. Bei neueren Designs für Werkzeugmaschinen wurde der Hydraulikzylinder durch eine Kugelrollspindeln oder einen Lineardirektantrieb [129] ersetzt. Maschinen dieses Typs werden auch als *Hexapod* bezeichnet. Beine vom UPS-Typ werden teilweise auch zur Modellierung von Seilplattformen eingesetzt, wobei unterstellt wird, dass die Seile stets unter hinreichender Spannung stehen und die Seilführungen als Kugelgelenke modelliert werden können [31]. Rein kinematisch gesehen wird bei einem UPS-Modul der Abstand zwischen den beiden Endpunkten durch den Aktuator bestimmt. Die generische Modellierung der PKM nach Abbildung 2.9a ist für dieses Beinmodul sehr einfach. Der für die Bindungsgleichungen (2.16) benötigte rahmenseitige Anlenkpunkt des Beins $\boldsymbol{a}_i$ ist mit

$$\boldsymbol{a}_i = \boldsymbol{c}_i \tag{2.17}$$

fest vorgegeben, wobei $\boldsymbol{c}_i$ ein maschinenabhängiger konstanter Ortsvektor ist und daher als gegeben betrachtet werden kann. Die Länge

$$l_i = q_i + q_0 \tag{2.18}$$

des Beins hängt nur von der verallgemeinerten Koordinate q_i des Schubgelenks ab. Hier muss nur ein konstanter Offset q_0 berücksichtigt werden, der z. B. bei Hydraulikzylindern auftreten kann. Durch diese simple Struktur besitzt das UPS-Modul die wenigsten geometrischen Parameter.

Durch eine Variation der Reihenfolge der Gelenke entsteht aus dem UPS-Modul ein Bein vom PUS-Typ (Abb. 2.9b). Bei dieser Konstruktion wird das Kugel-/Kardangelenkpaar

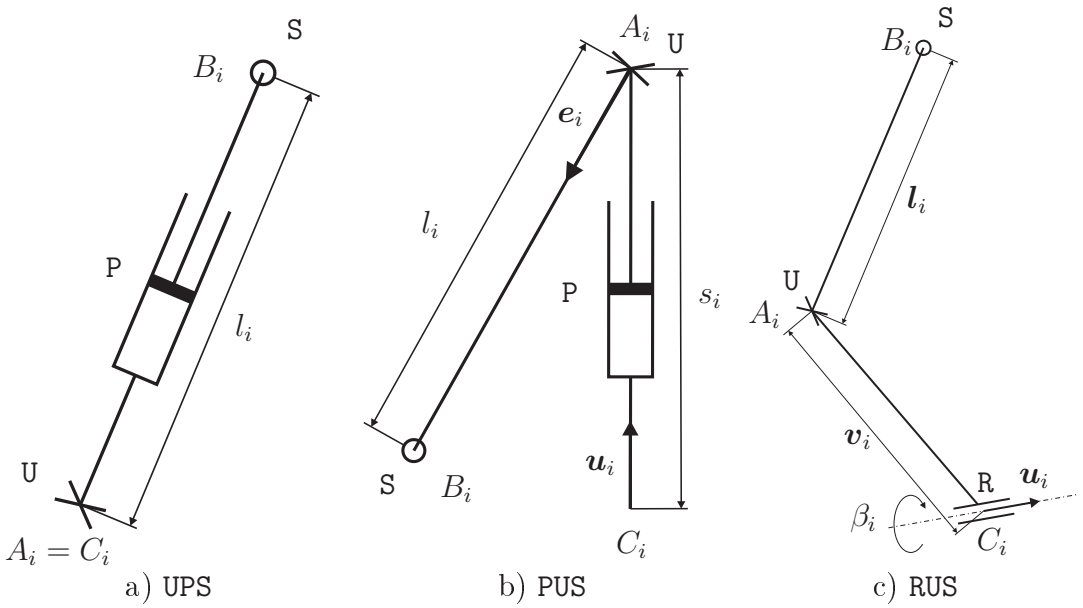


Abbildung 2.9: Geometrische Parameter der betrachteten Beinmodule

durch einen Stab konstanter Länge getrennt. Die Bewegungseinleitung erfolgt durch Veränderung des rahmenseitigen Fußpunkts in einer linearen Führung. Damit gelingt es, die Aktuatoren am ruhenden Maschinenrahmen anzubringen, während alle bewegungsübertragenden Komponenten passiv sind. Maschinen dieses Typs werden als *Hexaglide*, *Linaglide* oder *Linapod* bezeichnet. Die Geometrie bestimmt sich durch den konstanten Ortsvektor $\boldsymbol{c}_i$ und den konstanten Einheitsvektor der Führung $\boldsymbol{u}_i$ des Schubgelenks. Für die Bindungsgleichung (2.16) ergibt sich

$$\boldsymbol{a}_i = \boldsymbol{c}_i + s_i \boldsymbol{u}_i, \tag{2.19}$$

wobei die verallgemeinerte Koordinate s_i den Weg des Antriebs P misst. Jeder Stab besitzt eine konstante Länge l_i.

Eine PKM nach dem Baumuster des *Delta-Roboters* [19] verwendet Beine vom RUS-Typ (Abb. 2.9c). Dabei werden Stäbe konstanter Länge über einen drehbaren ersten Hebel $\boldsymbol{v}_i$ angetrieben. Der Aufbau ähnelt dem PUS-Typ, jedoch bewegen sich die Punkte A_i hier auf Kreisen anstatt auf Geraden. Auch bei diesem Konzept müssen die Aktuatoren selber nicht bewegt werden, so dass sich eine große Dynamik realisieren lässt. Die Position der rahmenseitigen Stabenden A_i für die Bindungsgleichung (2.16) lässt sich mithilfe des Drehtensors zu

$$\boldsymbol{a}_i = \boldsymbol{c}_i + \boldsymbol{T}(\boldsymbol{u}_i, \beta_i)\boldsymbol{v}_i \tag{2.20}$$

bestimmten, wobei $\boldsymbol{c}_i$ der konstante Ortsvektor zum Fußpunkt des Beins ist, $\boldsymbol{u}_i$ die Achse des Drehgelenks und β_i der Drehwinkel bezüglich der Anfangslage von $\boldsymbol{v}_i$. Für das kinematische Modell kann ohne Beschränkung der Allgemeinheit angenommen werden, dass $\boldsymbol{c}_i$ und $\boldsymbol{v}_i$ so gewählt wurden, dass $\boldsymbol{v}_i$ und $\boldsymbol{u}_i$ orthogonal sind. Die Länge der jeweils zweiten Stäbe wird wiederum als Konstante l_i betrachtet.

Anhand der topologischen Analyse wird die PKM also in die Subsysteme Plattform, Rahmen und Bein aufgeteilt. Diese Gliederung bildet die Grundlage für das modulare kinetostatische Modell, das in Kapitel 3 betrachtet wird. Die kinematischen Gleichungen lassen

sich komponentenweise auswerten, um die benötigten Bindungen für das in den Kapiteln 4 und 5 beschriebene Verfahren zur Arbeitsraumanalyse und Maßsynthese zu erhalten.

2.2.4 Inverse Kinematik

Die Aufgabe bei der inversen Kinematik besteht darin, die Bewegung der Antriebe aus einer gegebenen Bewegung des End-Effektor zu bestimmen. Die kinematischen Größen der Plattform sind somit bekannt und die Bindungsgleichungen müssen nach den Gelenkgrößen aufgelöst werden. Eine vollständig parallele Maschine besitzt genau so viele Beine wie Freiheitsgrade und die Kinematik kann für jedes dieser Beine einzeln betrachtet werden. Insofern ist die inverse Kinematik bei parallelen Robotern auf natürliche Weise entkoppelt. Jede Bindungsgleichung (2.16) wird zunächst nach den verallgemeinerten Koordinaten q_i aufgelöst. Bei vielen PKM kann eine geschlossene Lösung der inversen Kinematik gefunden werden. Dies ist insbesondere für die oben genannten Beintypen `PUS`, `UPS` und `RUS` möglich. Da die direkte Kinematik bei parallelen Robotern sehr kompliziert ist, bildet die Lösung der inversen Kinematik einen wesentlichen Grundbaustein für die Berechnung des Arbeitsraums sowie für die Maßsynthese und Optimierung der Maschine.

Von den betrachteten Beinmodulen besitzen die Beine vom `UPS`-Typ die einfachste Lösung, denn in diesem Fall liegt die Gleichung bereits in einer entkoppelten Form vor und kann direkt nach der Koordinate

$$q_i = (\boldsymbol{r} + \boldsymbol{R}\boldsymbol{b}_i - \boldsymbol{a}_i)^2 \quad i = 1, \ldots, 6 \tag{2.21}$$

aufgelöst werden.

Für das `PUS`-Modul werden die geometrischen Beziehungen eingesetzt und es ergibt sich die quadratische Gleichung

$$(\boldsymbol{r} + \boldsymbol{R}\boldsymbol{b}_i - \boldsymbol{c}_i - s_i\boldsymbol{u}_i)^2 - l_i^2 = 0, \quad i = 1, \ldots, 6, \tag{2.22}$$

die sich zu

$$q_i = \boldsymbol{u}_i \cdot \boldsymbol{d}_i \pm \sqrt{(\boldsymbol{u}_i \cdot \boldsymbol{d}_i)^2 - \boldsymbol{d}_i^2 + l_i^2} \quad \text{mit} \quad \boldsymbol{d}_i = \boldsymbol{r} + \boldsymbol{R}\boldsymbol{b}_i - \boldsymbol{c}_i \tag{2.23}$$

auflösen lässt. Jedes `PUS`-Bein hat also zwei Lösungen für die inverse Kinematik, die den beiden Möglichkeiten für den Zusammenbau entsprechen.

Für das `RUS`-Modul ergibt sich durch Einsetzten der Relativterme

$$(\boldsymbol{r} + \boldsymbol{R}\boldsymbol{b}_i - \boldsymbol{c}_i - \boldsymbol{T}(\boldsymbol{u}_i, \beta_i)\boldsymbol{v}_i)^2 - l_i^2 = 0, \quad i = 1, \ldots, 6. \tag{2.24}$$

Auch hier ist es nützlich, die Abkürzung $\boldsymbol{d}_i = \boldsymbol{r} + \boldsymbol{R}\boldsymbol{b}_i - \boldsymbol{c}_i$ einzuführen. Für den Drehtensor $\boldsymbol{T}(\boldsymbol{u}, \beta)$ wird die RODRIGUES-Formel

$$\boldsymbol{T}(\boldsymbol{u}, \beta) = \cos\beta \boldsymbol{I}_3 + (1 - \cos\beta)\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}} + \sin\beta\tilde{\boldsymbol{u}} \tag{2.25}$$

verwendet, wobei der Tilde-Operator aus einem Vektor $\boldsymbol{u}$ einen Tensor generiert, so dass $\tilde{\boldsymbol{u}}\boldsymbol{x} = \boldsymbol{u} \times \boldsymbol{x}$ gilt. Durch Einsetzten oben genannter Terme lässt sich die Gleichung (2.24) in die Form

$$A \sin\beta_i + B \cos\beta_i + C = 0 \tag{2.26}$$

umschreiben, wobei sich für die skalaren Konstanten die Werte

$$A = 2\boldsymbol{d}_i \cdot \boldsymbol{v}_i - 2\boldsymbol{d}_i \cdot (\boldsymbol{u}_i \boldsymbol{u}_i^{\mathrm{T}})\boldsymbol{v}_i, \tag{2.27}$$

$$B = 2\boldsymbol{d}_i \cdot \tilde{\boldsymbol{u}}_i \boldsymbol{v}_i, \tag{2.28}$$

$$C = \boldsymbol{d}_i^2 + \boldsymbol{v}_i^2 - l_i^2 + 2\boldsymbol{d}_i \cdot (\boldsymbol{u}_i \boldsymbol{u}_i^{\mathrm{T}})\boldsymbol{v}_i \tag{2.29}$$

ergeben. Damit lässt sich die inverse Kinematik lösen und es ergeben sich die beiden Lösungen

$$\tan \beta_i = \frac{AB \pm \sqrt{C^2 \left(B^2 + A^2 - C^2\right)}}{C^2 - A^2}. \tag{2.30}$$

2.2.5 Direkte Kinematik

Das Problem der direkten Kinematik lässt sich für räumliche PKM folgendermaßen formulieren: Gegeben sind die sechs festen Punkte im Raum $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_6$, die relativen Positionen von sechs Punkten $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_6$ auf einem starren Körper im Raum sowie die paarweisen Abstände $l_i = \overline{A_i B_i}$. Gesucht ist die Position und Orientierung des starren Körpers (Abb. 2.8). Das Problem der Vorwärtskinematik ist verglichen mit der inversen Kinematik bei parallelen Robotern erheblich schwieriger zu lösen. Bisher konnte für eine allgemeine Geometrie keine geschlossene Lösung für die Berechnung von Position und Orientierung der Plattform aus gegebenen Längen l_i angegeben werden. Spezielle Lösungen existieren jedoch für Fälle mit Konfigurationen, in denen z. B. einige Punkte A_i, A_j bzw. B_i, B_j identisch sind [54]. Für den allgemeinen Fall wurde jedoch gezeigt, dass höchstens 40 Lösungen existieren [69], die Bindungsgleichungen auf ein Polynom 40. Grads transformiert werden können [50] und bei geeigneter Geometrie bis zu 40 verschiedene reelle Lösungen für das direkte kinematische Problem existieren [28]. Diese Untersuchungen sind von hohem Wert für die theoretische Analyse, da viele Rückschlüsse über die Eigenschaften der Kinematik gezogen werden können. Jedoch ist das Polynom für die praktische Berechnung eher ungeeignet, da numerische Algorithmen zur Berechnung der Nullstellen solcher Polynome problematisch sind. Algorithmen auf Basis der Intervallanalyse sind dagegen in der Lage, sämtliche Lösungen der direkten Kinematik zu finden [80]. In der Praxis lässt sich das System der nichtlinearen Bindungsgleichungen mit numerischen Algorithmen hinreichend schnell und genau lösen. Konvergiert der numerische Algorithmus nur unzureichend gegen die gesuchte Lösung, weist dies häufig auf eine PKM mit ungünstiger Geometrie hin. Für die Transformation innerhalb der Steuerungen haben sich Newton-Raphson-Verfahren bewährt, die eine Lösung verfolgen und aufgrund der inkrementellen Veränderung gegenüber der letzten Auswertung der Kinematik in wenigen Schritten eine hinreichend genaue Approximation der Lösung finden.

Für die numerische Behandlung der direkten Kinematik müssen die relativen Terme für die verschiedenen Beinkinematik-Module nur in Gleichung (2.16) eingesetzt werden. Da es nicht möglich ist, dieses System geschlossen zu lösen, bietet diese Form einen hinreichend allgemeinen Weg, um die direkte Kinematik auf numerische Weise zu lösen. In Abschnitt 3.2 wird eine Methode vorgestellt, mit der sich ein vollständiges kinetostatisches Modell der Maschine erstellen lässt. Dieses schließt neben der Betrachtung der Positionen auch die Geschwindigkeits-, Beschleunigungs- und Kraftübertragung ein und erlaubt somit das Aufstellen der Bewegungsgleichungen.

2.2.6 Differentielle Kinematik und JACOBI-Matrix

Für eine Reihe von Untersuchungen zur Kinematik von Robotern werden neben der Positionsübertragung auch deren Ableitungen benötigt. Diese Ableitungen werden häufig als differentielle Kinematik bezeichnet und können als Übertragung von Geschwindigkeiten gedeutet werden. Im Allgemeinen ist die Funktion der direkten oder inversen Kinematik nichtlinear in den verallgemeinerten Koordinaten/Weltkoordinaten. Dagegen ergibt sich für den differentiellen Zusammenhang stets eine konfigurationsabhängige lineare Übertragung, die sich formal leicht als JACOBI-Matrix herleiten lässt. Aus der Beziehung der direkten Kinematik $\underline{y} = \underline{\varphi}^{\mathrm{DK}}(\underline{q})$ ergibt sich durch Ableitung nach den verallgemeinerten Koordinaten

$$\delta \underline{y} = \frac{\partial \underline{\varphi}^{\mathrm{DK}}(\underline{q})}{\partial \underline{q}} \delta \underline{q} = \underline{\underline{J}}^{\mathrm{DK}} \delta \underline{q}, \tag{2.31}$$

wobei die Matrix

$$\underline{\underline{J}}^{\mathrm{DK}} = \begin{bmatrix} \frac{\partial \varphi_1^{\mathrm{DK}}}{\partial q_1} & \cdots & \frac{\partial \varphi_1^{\mathrm{DK}}}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \varphi_6^{\mathrm{DK}}}{\partial q_1} & \cdots & \frac{\partial \varphi_6^{\mathrm{DK}}}{\partial q_n} \end{bmatrix} \tag{2.32}$$

als JACOBI-Matrix bezeichnet wird. Entsprechend wird für die inverse Kinematik die JACOBI-Matrix als

$$\delta \underline{q} = \frac{\partial \underline{\varphi}^{\mathrm{IK}}(\underline{y})}{\partial \underline{y}} \delta \underline{y} = \underline{\underline{J}}^{\mathrm{IK}} \delta \underline{y} \tag{2.33}$$

definiert, wobei in nicht singulären Stellungen $\underline{\underline{J}}^{\mathrm{DK}} = (\underline{\underline{J}}^{\mathrm{IK}})^{-1}$ gilt. Da für viele PKM eine geschlossene Lösung der inversen Kinematik existiert, kann auch die inverse JACOBI-Matrix in geschlossener Form angegeben werden. Diese geschlossene Darstellung der JACOBI-Matrix bildet den Ausgangspunkt für die Untersuchungen der Singularitäten (Abschnitt 4.4.3) und der Manipulierbarkeit (Abschnitt 4.4.4) für PKM.

2.2.7 Arbeitsraum

Der Arbeitsraum bildet eine zentrale Eigenschaft bei Robotern und ist daher in den folgenden Kapiteln Gegenstand der Untersuchungen. Er wird dabei nach zwei unabhängigen Merkmalen klassifiziert. Einerseits erfolgt die Einteilung über die geforderten Eigenschaften, die an allen Punkten erfüllt sein müssen. Andererseits werden Unterräume des im Allgemeinen sechs-dimensionalen Arbeitsraums betrachtet, bei denen an jedem Punkt des Unterraums gewisse Positionen und Orientierungen erreichbar sind. Der Begriff des Arbeitsraums wird folgendermaßen definiert:

Als Arbeitsraum eines Roboters wird die Menge $\mathcal{W} \subset \mathbb{R}^6$ aller Posen bestehend aus Position und Orientierung eines End-Effektor-Koordinatensystems bezeichnet, die mit dem Roboter unter gegebenen Anforderungen erreicht werden können.

Der Begriff des Arbeitsraums bezieht sich explizit auf den durch die *gegebenen Anforderungen* eingeschränkten Teil. In der Praxis werden diese Anforderungen so definiert, dass alle mechanischen und technischen Notwendigkeiten berücksichtigt werden. In Kapitel 4 werden mathematische Bedingungen eingeführt, die eine Überprüfung der folgenden Kriterien ermöglichen: kinematische Erreichbarkeit, Begrenzung der Übersetzung sowie

Freiheit von Singularitäten, Kollisionen und Gelenkbegrenzungen. Der verwendbare Arbeitsraum ist folglich die Schnittmenge der Arbeitsräume, die sich für die jeweils einzelnen Kriterien ergeben. Der Arbeitsraum ist also per Definition frei von singulären Stellungen, Kollisionen usw. Weist der Roboter solche Einschränkungen auf, ist der Arbeitsraum dieses Roboters entsprechend kleiner. Zum besseren Verständnis der jeweiligen Einflüsse ist es sinnvoll, den Arbeitsraum für jedes einzelne Kriterium ermitteln zu können.

Die geometrische Gestalt des Arbeitsraums ist für den Anwender von entscheidender Bedeutung. Im Allgemeinen ist der Arbeitsraum eine sechs-dimensionale Menge, die sich nicht anschaulich darstellen lässt. Als zweites Merkmal zur Klassifikation werden daher Untermengen eingeführt, die es erlauben, sich schnell ein Bild von der Gestalt des Arbeitsraums zu machen. Die folgenden Untermengen sind praktisch nützlich:

- Der *translatorische Arbeitsraum* $\mathcal{W}_{\mathrm{T}} \subset \mathbb{R}^3$ umfasst die Menge aller Positionen des End-Effektor-Koordinatensystems, die mit einer gegebenen festen Orientierung $\boldsymbol{R}$ erreicht werden können.
- Der *Orientierungsarbeitsraum* $\mathcal{W}_{\mathrm{O}} \subset \mathbb{R}^3$ ist die Menge aller Orientierungen, die mit dem End-Effektor-Koordinatensystem für eine gegebene Position $\boldsymbol{r}$ erreicht werden können.
- Der *Gesamtorientierungsarbeitsraum* $\mathcal{W}_{\mathrm{GO}} \subset \mathbb{R}^3$ umfasst alle Positionen des End-Effektor-Koordinatensystems, die mit einer gegebenen Menge an Orientierungen erreicht werden können.

Der Arbeitsraum besitzt bei den meisten parallelen Robotern eine sehr komplexe Gestalt. Vor allem bei Werkzeugmaschinen ist für Anwender der Vergleich mit kartesischen seriellen Maschinen wichtig, die einen intuitiv nachvollziehbaren quaderförmigen Arbeitsraum haben. Daher stellt sich bei der Berechnung des Arbeitsraums meist nicht die Frage nach dem gesamten Arbeitsraum, sondern nach einem größten Quader innerhalb des Arbeitsraums. Insofern ist beim Arbeitsraum neben dem Volumen die Form für eine technische Nutzung wichtig.

2.2.8 Parametrisierung der Rotation

Die Formeln der Kinematik wurden unabhängig von Koordinatensystemen mit physikalischen Vektoren hergeleitet. Für eine numerische Auswertung müssen die physikalischen Größen jedoch durch geeignete Parameter in Komponenten abgebildet werden. Daher werden in den folgenden Abschnitten einige Parametrisierungen der Rotationsmatrix betrachtet sowie Parametrisierungen der Geometrie für zwei Klassen von PKM vorgestellt.

Die räumliche Drehung ist nichttrivial und in der Literatur finden sich verschiedene Ansätze, die räumliche Drehung zwischen zwei Koordinatensystemen mit Parametern zu beschreiben. Es scheint keine Parametrisierung der Drehung zu existieren, die gleichermaßen für alle Probleme geeignet ist. An dieser Stelle soll keineswegs ein vollständiger Überblick über dieses Themas gegeben werden. Stattdessen werden ausgewählte Parametrisierungen eingeführt, die in dieser Arbeit verwendet werden. Die Transformation zwischen zwei Koordinatensystemen $\mathcal{K}_0$ und $\mathcal{K}_p$ lässt sich als Matrix $\underline{\underline{R}}_p$ beschreiben, die eine lineare Abbildung zwischen Vektoren im Koordinatensystem $\mathcal{K}_p$ und Vektoren im $\mathcal{K}_0$ herstellt. Bei der Auswahl geeigneter Parametrisierungen wird insbesondere in Betracht

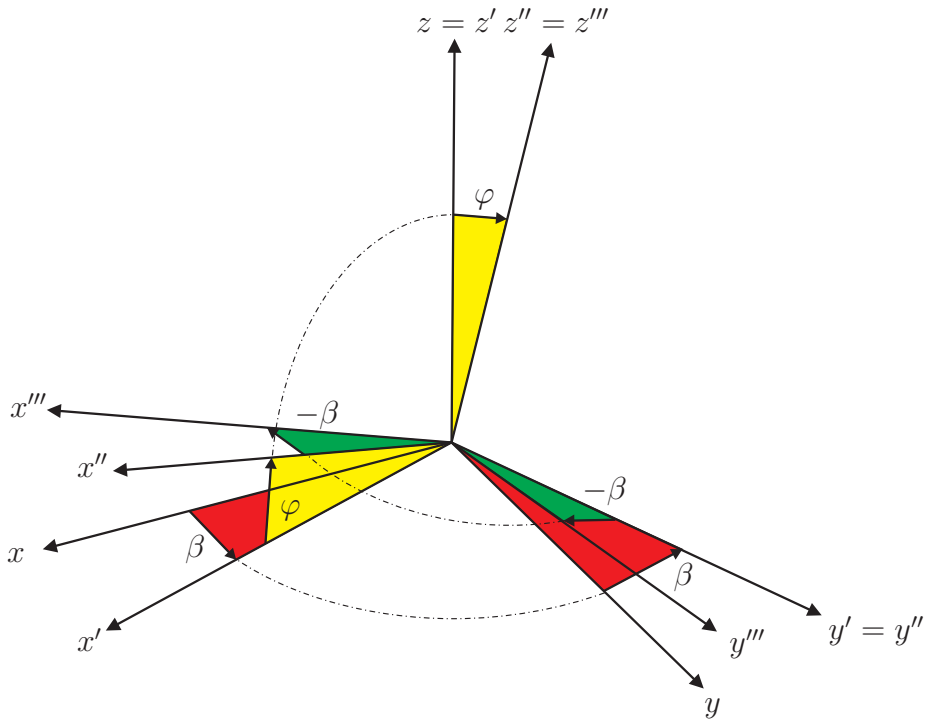


Abbildung 2.10: Rotation $\underline{\underline{R}}_{\mathrm{WZM}}$ zur Werkzeuganstellung mit β und φ

gezogen, dass PKM nur relativ kleine Rotationen der Plattform ermöglichen, so dass eine Darstellung in KARDAN-Winkeln (θ, ψ, φ) möglich ist. Diese setzt sich aus einer Sequenz elementarer Drehungen um die jeweiligen Koordinatenachsen $(\underline{e}_x, \underline{e}_y, \underline{e}_z)$ zusammen

$$\underline{\underline{R}}_{\mathrm{Kardan}}(\theta, \psi, \varphi) = \underline{\underline{T}}(\underline{e}_z, \theta)\underline{\underline{T}}(\underline{e}_y, \psi)\underline{\underline{T}}(\underline{e}_x, \varphi) \tag{2.34}$$

$$= \begin{bmatrix} \cos\theta\,\cos\psi\,\cos\varphi - \sin\theta\,\sin\varphi & -\cos\theta\,\cos\psi\,\sin\varphi - \sin\theta\,\cos\varphi & \cos\theta\,\sin\psi \\ \sin\theta\,\cos\psi\,\cos\varphi + \cos\theta\,\sin\varphi & -\sin\theta\,\cos\psi\,\sin\varphi + \cos\theta\,\cos\varphi & \sin\theta\,\sin\psi \\ -\sin\psi\,\cos\varphi & \sin\psi\,\sin\varphi & \cos\psi \end{bmatrix}.$$

Für die Betrachtung von Werkzeugmaschinen ist es sinnvoll, eine spezielle Parametrisierung der räumlichen Drehung einzuführen, die berücksichtigt, dass die Plattform mit dem Werkzeug in jede Richtung gekippt werden kann, ohne dass eine Rotation des Werkzeugs um seine Längsachse erfolgt (Abb. 2.10). Im Folgenden wird davon ausgegangen, dass die z-Achse des Plattformkoordinatensystems $\mathcal{K}_p$ der Längsachse des Werkzeugs entspricht. Damit ergibt sich die Rotationsmatrix zu

$$\underline{\underline{R}}_{\mathrm{WZM}}(\beta, \varphi) = \underline{\underline{T}}(\underline{e}_z, \beta)\underline{\underline{T}}(\underline{e}_y, \varphi)\underline{\underline{T}}(\underline{e}_z, -\beta) \tag{2.35}$$

$$= \begin{bmatrix} \cos^2\beta\cos\varphi + \sin^2\beta & \cos\beta\sin\beta(\cos\varphi - 1) & \cos\beta\sin\varphi \\ \cos\beta\sin\beta(\cos\varphi - 1) & \sin^2\beta\cos\varphi + \cos^2\beta & \sin\beta\sin\varphi \\ -\cos\beta\sin\varphi & -\sin\beta\sin\varphi & \cos\varphi \end{bmatrix}. \tag{2.36}$$

Für die Beschreibung des Gesamtorientierungsarbeitsraums $\mathcal{W}_{\mathrm{GO}}$ für Werkzeugmaschinen ist die Matrix $\underline{\underline{R}}_{\mathrm{WZM}}$ hilfreich, denn das gewünschte Schwenken der Plattform um einen Winkel $\varphi_{\max}$ kann durch die Parameter β und φ kompakt beschrieben werden. Dazu wird gefordert, dass jede Orientierung der Plattform mit $\beta \in [0, 2\pi]$ und $\varphi \in [0, \varphi_{\max}]$ möglich ist.

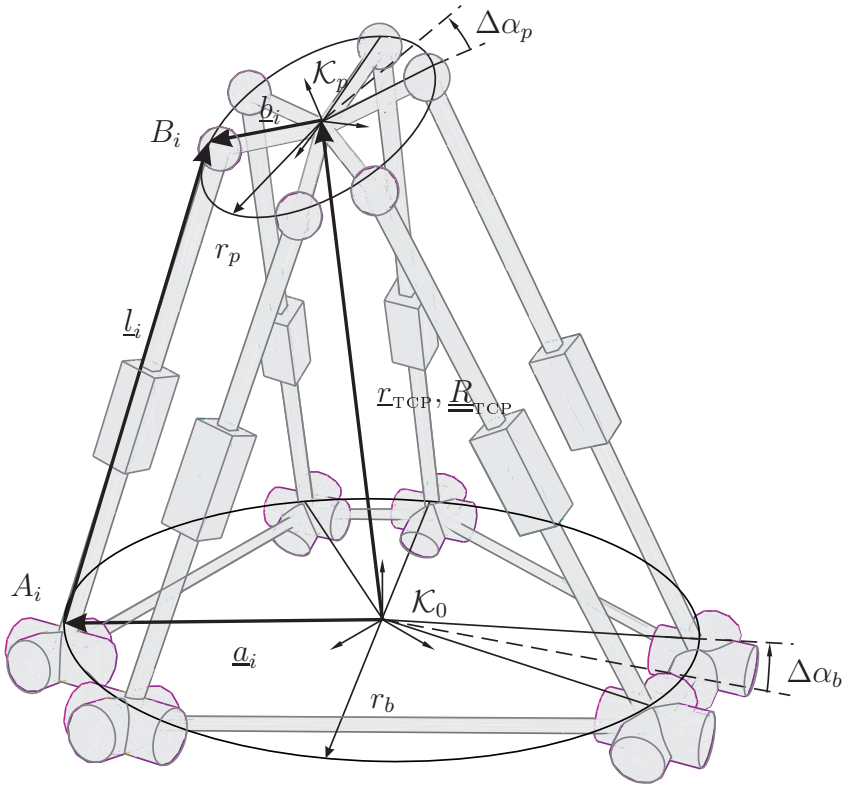


Abbildung 2.11: Parametrisierung und Kinematik eines Simplified Symmetric Mechanism (SSM)

2.2.9 Vereinfachte symmetrische Gough-Plattform

In der Literatur wird für die Beschreibung der klassischen Gough-Plattform häufig ein geometrisch vereinfachtes Modell herangezogen, das als *Simplified Symmetric Mechanism* (SSM) bezeichnet wird. Dabei wird davon ausgegangen, dass die Anlenkpunkte A_i auf einem Kreis mit Radius r_b in der xy-Ebene von $\mathcal{K}_0$ liegen (Abb. 2.11). Die Anlenkpunkte der Plattform B_i liegen ebenfalls auf einem Kreis mit Radius r_p, der mit der xy-Ebene des Koordinatensystems $\mathcal{K}_p$ verbunden ist. Weiterhin wird davon ausgegangen, dass die Ortsvektoren zu den Punkte A_i und B_i – wie in Abbildung 2.11 gezeigt – um die Winkel $\Delta\alpha_b$ und $\Delta\alpha_p$ verdreht sind. Damit ergeben sich für die Anlenkpunkte die in Tabelle 2.2 angegebenen Koordinaten.

Tabelle 2.2: Parametrisierung der Position der Anlenkpunkte für SSM

i	$\boldsymbol{a}_i$	$\boldsymbol{b}_i$
1	$r_b\boldsymbol{T}(\boldsymbol{e}_z, \Delta\alpha_b)\boldsymbol{e}_x$	$r_p\boldsymbol{T}(\boldsymbol{e}_z, \Delta\alpha_p)\boldsymbol{e}_x$
2	$r_b\boldsymbol{T}(\boldsymbol{e}_z, 120° - \Delta\alpha_b)\boldsymbol{e}_x$	$r_p\boldsymbol{T}(\boldsymbol{e}_z, 120° - \Delta\alpha_p)\boldsymbol{e}_x$
3	$r_b\boldsymbol{T}(\boldsymbol{e}_z, 120° + \Delta\alpha_b)\boldsymbol{e}_x$	$r_p\boldsymbol{T}(\boldsymbol{e}_z, 120° + \Delta\alpha_p)\boldsymbol{e}_x$
4	$r_b\boldsymbol{T}(\boldsymbol{e}_z, 240° - \Delta\alpha_b)\boldsymbol{e}_x$	$r_p\boldsymbol{T}(\boldsymbol{e}_z, 240° - \Delta\alpha_p)\boldsymbol{e}_x$
5	$r_b\boldsymbol{T}(\boldsymbol{e}_z, 240° + \Delta\alpha_b)\boldsymbol{e}_x$	$r_p\boldsymbol{T}(\boldsymbol{e}_z, 240° + \Delta\alpha_p)\boldsymbol{e}_x$
6	$r_b\boldsymbol{T}(\boldsymbol{e}_z, -\Delta\alpha_b)\boldsymbol{e}_x$	$r_p\boldsymbol{T}(\boldsymbol{e}_z, -\Delta\alpha_p)\boldsymbol{e}_x$

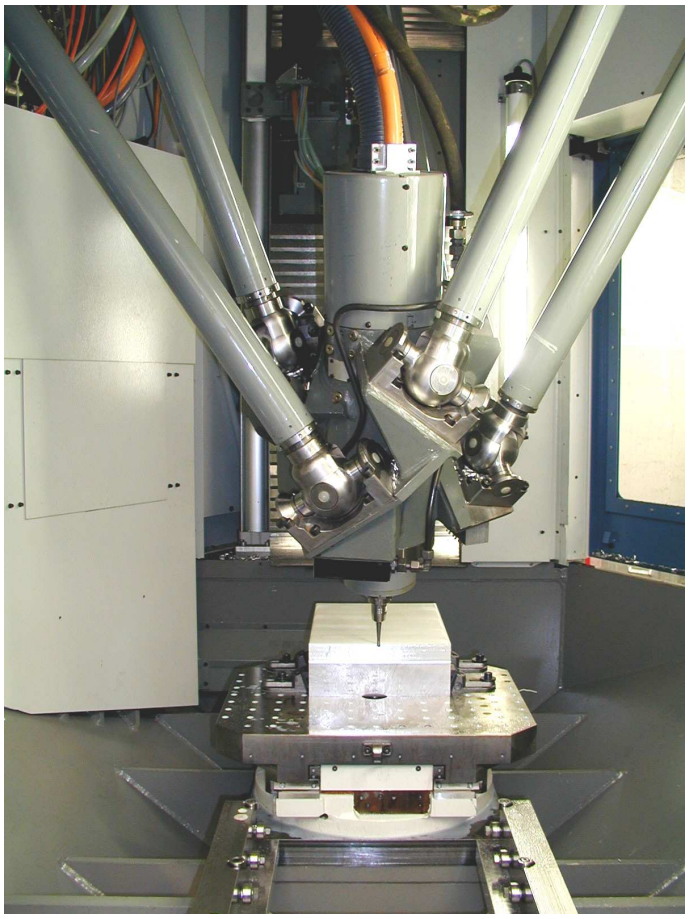


Abbildung 2.12: Versuchsträger *Linapod* am ISW, Stuttgart

2.2.10 Stabkinematik Linapod

Die PKM *Linapod* (Abb. 2.12) ist der Prototyp einer räumlichen Werkzeugmaschine mit sechs Freiheitsgraden, der am Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW) der Universität Stuttgart aufgebaut wurde [109]. Die Kinematik der Maschine ist aus PUS-Elementen aufgebaut. Im Gegensatz zum SSM haben die Stäbe beim *Linapod* eine konstante Länge, und die Bewegung wird durch eine Veränderung der rahmenseitigen Anlenkpunkte erzeugt. Die Schubgelenke sind dabei für alle Beine parallel zur z-Achse von $\mathcal{K}_0$ angeordnet, so dass alle Schlitten am Versuchsträger parallel verschoben werden. Weiterhin wurden am Versuchsträger jeweils zwei Beinmodule auf einer gemeinsamen Führungsbahn montiert.

In Abbildung 2.13 ist die Parametrisierung für die vorliegende Maschine *Linapod* zusammengestellt. Die Schubgelenke der PUS-Module sind auf einem Basiskreis in der xy-Ebene des Koordinatensystems $\mathcal{K}_0$ mit Radius r_b angeordnet und entlang der z-Achse von $\mathcal{K}_0$ ausgerichtet. Ausgehend von einer 120°-Teilung sind die Fußpunkte mit den PUS-Beinen um Δa tangential verschoben. Die Beinmodule 1–3 mit einer Länge l_l werden als *untere* Beine bezeichnet und sind mit der unteren Ebene der Plattform verbunden. Entsprechend haben die *oberen* Beine 4–6 eine Länge l_u und sind an der oberen Plattformebene befestigt. Die plattformseitigen Anlenkpunkte sind auf Kreisen mit Radius r_l (unten) und r_u (oben) angeordnet. Die Kreise liegen jeweils auf den Ebenen, die parallel zur xy-Ebene

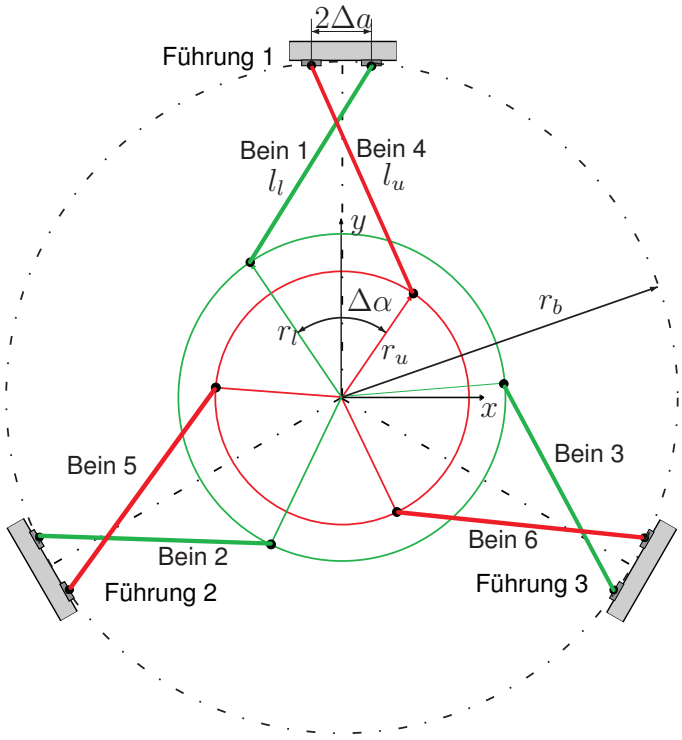


Abbildung 2.13: Parameterisierung der *Linapod*-Geometrie

Tabelle 2.3: Plattform- und rahmenseitige Anlenkpunkte beim *Linapod*

i	$\boldsymbol{c}_i$	$\boldsymbol{b}_i$
1	$\boldsymbol{T}(\boldsymbol{e}_z, 0°)(r_b\boldsymbol{e}_x + \Delta a\boldsymbol{e}_y)$	$\boldsymbol{T}(\boldsymbol{e}_z, \Delta\alpha)r_l\boldsymbol{e}_x + \Delta t\boldsymbol{e}_z$
2	$\boldsymbol{T}(\boldsymbol{e}_z, 120°)(r_b\boldsymbol{e}_x + \Delta a\boldsymbol{e}_y)$	$\boldsymbol{T}(\boldsymbol{e}_z, 120° + \Delta\alpha)r_l\boldsymbol{e}_x + \Delta t\boldsymbol{e}_z$
3	$\boldsymbol{T}(\boldsymbol{e}_z, 240°)(r_b\boldsymbol{e}_x + \Delta a\boldsymbol{e}_y)$	$\boldsymbol{T}(\boldsymbol{e}_z, 240° + \Delta\alpha)r_l\boldsymbol{e}_x + \Delta t\boldsymbol{e}_z$
4	$\boldsymbol{T}(\boldsymbol{e}_z, 0°)(r_b\boldsymbol{e}_x - \Delta a\boldsymbol{e}_y)$	$\boldsymbol{T}(\boldsymbol{e}_z, -\Delta\alpha)r_u\boldsymbol{e}_x + (\Delta t + \Delta h)\boldsymbol{e}_z$
5	$\boldsymbol{T}(\boldsymbol{e}_z, 120°)(r_b\boldsymbol{e}_x - \Delta a\boldsymbol{e}_y)$	$\boldsymbol{T}(\boldsymbol{e}_z, 120° - \Delta\alpha)r_u\boldsymbol{e}_x + (\Delta t + \Delta h)\boldsymbol{e}_z$
6	$\boldsymbol{T}(\boldsymbol{e}_z, 240°)(r_b\boldsymbol{e}_x - \Delta a\boldsymbol{e}_y)$	$\boldsymbol{T}(\boldsymbol{e}_z, 240° - \Delta\alpha)r_u\boldsymbol{e}_x + (\Delta t + \Delta h)\boldsymbol{e}_z$

Tabelle 2.4: Geometrische Parameter der Maschine *Linapod* und ihre nominellen Werte

Parameter	Wert	Beschreibung
l_l	1.25 m	Länge des unteren Beins
l_u	1.7 m	Länge des oberen Beins
r_b	0.886 m	Radius des Rahmens der Maschine
r_u	0.2 m	Radius der oberen Plattformebene
r_l	0.22 m	Radius der unteren Plattformebene
Δa	−0.05 m	Tangentialer Versatz der rahmenseitigen Anlenkpunkte
Δh	0.2 m	Abstand zwischen unterer und oberer Plattformebene
Δt	0.2 m	Abstand zwischen $\mathcal{K}_p$ und unterer Plattformebene
$\Delta\alpha$	35°	Verdrehung von unteren und oberen Anlenkpunkte gegenüber $\mathcal{K}_p$

von $\mathcal{K}_p$ sind. Der Abstand Δt bezeichnet die Entfernung zwischen der unteren Ebene und $\mathcal{K}_p$, und Δh ist der Abstand zwischen der unteren und der oberen Plattformebene. Damit ergeben sich die in Tabelle 2.3 zusammengestellten Ortsvektoren für die Anlenkpunkte $\boldsymbol{c}_i$ und $\boldsymbol{b}_i$. Die nominellen Werte der geometrischen Parameter des Versuchsträgers sind in Tabelle 2.4 zusammengestellt.

2.3 Intervallanalyse

Die *Intervallarithmetik* wurde von Ramon E. Moore [89] vorgeschlagen und ursprünglich dazu verwendet, die Auswirkungen von Rechen- und Rundungsfehlern zu berücksichtigen. Daneben bietet die Intervallarithmetik eine Möglichkeit, den maximalen Wertebereich einer Funktion über einem Intervall abzuschätzen. Dies ist auch dann möglich, wenn einige der Eingangsdaten nicht exakt sondern innerhalb eines Intervalls bekannt sind. Auf der Basis der Intervallarithmetik wurde die *Intervallanalyse* entwickelt, die unter Ausnutzung der speziellen Eigenschaften der Intervallarithmetik auf einen weiten Bereich numerischer Probleme angewendet werden kann. Insbesondere bei der Lösung nichtlinearer Gleichungen, bei *Constraint Satisfaction Problems* (CSP) und bei der globalen Optimierung mit Nebenbedingungen wurden in den letzten Jahrzehnten leistungsfähige Algorithmen entwickelt [5, 39, 92]. Diese neuen Methoden wurden für die Lösungen von Problemen verwendet, die sich mit klassischen Methoden kaum behandeln lassen. Insbesondere der robuste Umgang mit Verfahrens- und Rundungsfehlern ermöglicht auf einfache Weise, strikte Schranken für den numerischen Fehler einer arithmetischen Berechnung anzugeben.

Ein *Intervall* $\widehat{x}$ ist ein geordnetes Paar $[a, b]$ zweier reeller Zahlen

$$\widehat{x} = [a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}, \tag{2.37}$$

wobei a als *Infimum* und b als *Supremum* von $\widehat{x}$ bezeichnet wird. Die Differenz zwischen Infimum und Supremum wird *Intervallbreite* genannt und der Mittelwert wird als Mittelpunkt bezeichnet. Dafür werden die Funktionen

$$\begin{aligned} \inf \widehat{x} &= a, && (2.38)\\ \sup \widehat{x} &= b, && (2.39)\\ \operatorname{diam} \widehat{x} &= b - a, && (2.40)\\ \operatorname{mid} \widehat{x} &= \frac{1}{2}(a + b) && (2.41) \end{aligned}$$

definiert. Die Menge aller reellen Intervalle wird mit dem Symbol $\mathbb{I}$ bezeichnet. Ein Vektor von Intervallen wird als *Box* bezeichnet. Analog zur Arithmetik der reellen Zahlen werden für die Intervallarithmetik die grundlegenden Operatoren $+, -, *, /$ folgendermaßen auf der Menge der Intervalle $\mathbb{I}$ erklärt:

$$\widehat{x} \circ \widehat{y} = [a, b] \circ [c, d] = \{x \circ y \mid a \leq x \leq b, c \leq y \leq d\}, \tag{2.42}$$

wobei durch $\circ$ die elementaren Operatoren $+, -, *, /$ bezeichnet sind. Für den Fall der Division sei der Ausdruck $\widehat{x}/\widehat{y}$ für $0 \in \widehat{y}$ nicht definiert. Für die grundlegenden Operatoren ergeben sich die folgenden Rechenregeln

$$[a, b] + [c, d] = [a + c, b + d], \tag{2.43}$$

$$[a,b]-[c,d] = [a-d,b-c], \tag{2.44}$$
$$[a,b]*[c,d] = [\min(ac,ad,bc,bd),\max(ac,ad,bc,bd)], \tag{2.45}$$
$$[a,b]\ /\ [c,d] = [a,b]*[1/d,1/c] \quad \text{falls} \quad 0\notin[c,d]. \tag{2.46}$$

Das Ergebnis solch einer Verknüpfung ist wiederum ein Intervall, d. h. die Menge der Intervalle ist bezüglich der arithmetischen Operatoren $+,-,*$ abgeschlossen.[5] Die entarteten Intervalle der Form $[a,a]$ werden mit den reellen Zahlen identifiziert. Die Intervallauswertung einer Operation konvergiert gegen die Arithmetik reeller Zahlen, wenn die Intervallbreite aller Variablen gegen 0 strebt. Daher kann die Intervallarithmetik als eine Verallgemeinerung der reellen Arithmetik betrachtet werden [89].

Ein Intervall $\widehat{x}$ heißt *positiv* (*negativ*) falls $\inf\widehat{x}\geq 0$ ($\sup\widehat{x}\leq 0$) und *streng positiv* (*streng negativ*) falls $\inf\widehat{x}>0$ ($\sup\widehat{x}<0$). Zwei Intervalle $\widehat{x},\widehat{y}$ sind genau dann *gleich*, wenn $\inf\widehat{x}=\inf\widehat{y}$ und $\sup\widehat{x}=\sup\widehat{y}$. Intervalle sind teilweise geordnet, und es gilt $[a,b]<[c,d]$ genau für $b<c$.

2.3.1 Intervallauswertung einer Funktion

Die Intervallanalyse lässt sich für gewöhnliche stetige[6] Funktionen durchführen, die durch Kombinationen der elementaren Operationen zusammengesetzt sind. Dabei werden die reellen Variablen $(x_1,\ldots,x_n)$ der Funktion $f:\mathbb{R}^n\to\mathbb{R}$ durch reelle Intervalle $(\widehat{x}_1,\ldots,\widehat{x}_n)$ ersetzt. Damit ergibt sich eine Funktion $f^{\mathrm{I}}:\mathbb{I}^n\to\mathbb{I}$, die entsprechend der reellen Funktion den Intervallvektor $\underline{\widehat{x}}$ auf ein Intervall $\widehat{y}$ abbildet. Da die Funktion f^{I} bis auf den Typ ihrer Argumente mit ihrem reellen Gegenstück f übereinstimmt, wird in dieser Arbeit darauf verzichtet, die Funktionen explizit zu unterscheiden. Stattdessen lässt sich dem Typ ihrer Argumente entnehmen, mit welcher Arithmetik die Funktion auszuwerten ist. Die Berechnung von f mit Intervallen wird als *Intervallauswertung* von f bezeichnet und immer dann verwendet, wenn mindestens ein Argument der Funktion ein Intervall ist. Aus der Definition der Intervalloperationen folgt unmittelbar

$$\widehat{z}=f(\underline{\widehat{x}}) \quad\Leftrightarrow\quad \inf\widehat{z}\leq f(\underline{x})\leq\sup\widehat{z} \quad \forall\ \underline{x}\in\underline{\widehat{x}}, \tag{2.47}$$

d. h. die Intervallauswertung einer Funktion liefert garantierte Schranken $\widehat{z}$ für den Wertebereich der Funktion f über der Box $\underline{\widehat{x}}$.

2.3.2 Überschätzung

Obwohl die Intervallanalyse der Arithmetik mit reellen Zahlen ähnelt, gibt es einige signifikante Unterschiede. Das Assoziativ- und Kommutativgesetz gilt sowohl für die Addition als auch für die Multiplikation von Intervallen. Dagegen gilt das Distributivgesetz nicht in der gewohnten Form:

$$\widehat{a}(\widehat{b}+\widehat{c})\neq\widehat{a}\widehat{b}+\widehat{a}\widehat{c}, \quad \widehat{a},\widehat{b},\widehat{c}\in\mathbb{I}. \tag{2.48}$$

[5] Bei der hier nicht näher betrachteten *erweiterten Intervallarithmetik* wird die Division durch 0 einbezogen, indem auch $\pm\infty$ als Grenze eines Intervalls zugelassen wird. Die erweiterte Intervallarithmetik ist damit auch über der Division abgeschlossen; siehe dazu [39].

[6] Die folgenden Ausführungen beschränken sich auf stetige Funktionen, da dies für die in dieser Arbeit behandelten Funktionen ausreichend ist. In der Literatur existieren Erweiterungen der Intervallanalyse, die auch für nicht stetigen Funktionen erklärt sind; siehe dazu z. B. [39].

Die *Subdistributivität* ist eine schwächere Version des Distributivgesetzes, und es gilt für jedes Intervall $\widehat{a}, \widehat{b}, \widehat{c} \in \mathbb{I}$

$$\widehat{a}(\widehat{b} + \widehat{c}) \subset \widehat{a}\widehat{b} + \widehat{a}\widehat{c}. \tag{2.49}$$

Für eine Auswertung von Intervallfunktionen mit möglichst engen Schranken ist es somit günstiger, die linke Seite von Gleichung (2.49) auszuwerten, da durch diesen Term die schärfsten Schranken berechnet werden. Generell lässt sich feststellen, dass die Auswertung eines Ausdrucks mit Intervallvariablen nur dann den exakten Wertebereich des Ausdrucks liefert, wenn jede Intervallvariable genau einmal in dem Ausdruck vorkommt. Tritt eine Variable mehr als einmal in einem Ausdruck auf, geht die *Intervallidentität* verloren, d. h. es wird nicht berücksichtigt, dass eine Variable x bei jeder Verknüpfung den identischen Wert $x \in \widehat{x}$ hat. Daher wird die Intervallauswertung unschärfer, was zu einer Überschätzung des Wertebereichs führt, die bereits für die einfachen Ausdrücke $\widehat{a}^2 \subset \widehat{a} * \widehat{a}$ und $0 \subset \widehat{a} - \widehat{a} \neq 0$ auftritt. Beispielsweise ergibt für $\widehat{x} = [-1, 1]$ der Ausdruck $\widehat{x} * \widehat{x} = [-1, 1]$ anstelle des genauen Ergebnisses $\widehat{x}^2 = [0, 1]$, denn $\widehat{x} * \widehat{x}$ wird behandelt wie $\widehat{x} * \widehat{y}$, wobei zufällig $\widehat{y} = \widehat{x}$ ist. Da sich komplizierte Ausdrücke nicht so zusammenfassen lassen, dass jede Variable nur genau einmal vorkommt, ist eine Überschätzung unvermeidlich. Es kann jedoch häufig durch eine geeignete Umformung der Terme eine Form gefunden werden, die zu einer moderateren Überschätzung führt. Ist dagegen die Intervallidentität gegeben, kann durch die Intervallauswertung der Funktion

$$\widehat{y} = f(\underline{\widehat{x}}) \tag{2.50}$$

gefolgert werden, dass f auf $\widehat{y}$ surjektiv ist, d. h. für jedes $y \in \widehat{y}$ mindestens ein $\underline{x} \in \underline{\widehat{x}}$ existiert.

2.3.3 Software und Implementierungen

Die Eigenschaften zur Einschließung des Wertebereichs einer Funktion gelten für das mathematische Kalkül der exakten Intervallanalyse, d. h. es wird davon ausgegangen, dass jede reelle Zahl ohne Fehler abgebildet werden kann. In der Praxis werden Berechnungen jedoch mit einer endlichen Genauigkeit auf Computern ausgeführt und bei der Abbildung von reellen Zahlen im Computer entstehen Zahlenabbruchfehler. Durch die Kontrolle der unvermeidlichen Rundung kann erreicht werden, dass bei einer Berechnung die Rundung für das Infimum und das Supremum eines Intervalls stets nach außen ausgeführt wird, so dass das gerundete Intervall das tatsächliche Ergebnis der exakten Intervallanalyse einschließt. Eine wichtige Anwendung der Intervallanalyse ist daher die Möglichkeit, numerische Fehler automatisch zu berücksichtigen. Diese Fehler können als Ungenauigkeit der Eingangsdaten auftreten oder durch numerische Fehler bei der Verarbeitung entstehen. Bei der Gleitkommadarstellung reeller Zahlen im Computer sind Zahlenabbruchfehler unvermeidlich. Auch durch die Intervallanalyse können diese Fehler nicht vermieden werden, jedoch wird mit dem Ergebnis eine rigorose Abschätzung für diese Fehler mitgeliefert. Daher sind Algorithmen auf der Basis von Intervallanalyse *robust* gegenüber Rundungsfehlern.

Es existiert eine Reihe von Programmbibliotheken und Entwicklungsumgebungen für die Intervallanalyse. In dieser Arbeit wird BIAS/Profil von Knüppel [66, 67] verwendet, da für diese Bibliothek Versionen vorliegen, die sich in C++-Programme unter Windows und Linux einbetten lassen. Daneben existieren weitere Implementierungen, wie z. B. PASCAL-

XSC [37], C-XSC [64] und Sun Forte [127], die vergleichbare Funktionen anbieten. Von Rump wird eine Erweiterung für MATLAB (MathWorks Inc.) zur Intervallanalyse beschrieben [116, 117].

Aufbauend auf derartige Implementierungen der grundlegenden Funktionen für die Intervallanalyse wurden eine Reihe von Werkzeugen geschaffen, die speziell für die Intervallanalyse entwickelte Algorithmen enthalten. Ein Beispiel mit zahlreichen Anwendungen für Mechanismen ist ALIAS [20] sowie die Erweiterung ALIAS/Maple [21] für das Computeralgebra-System Maple (Waterloo Maple Inc.). Die Grundlagen für die dort verwendeten Algorithmen finden sich bei Moore [89], Neumaier [92] und Hansen [38, 39].

Eine Bibliothek für die Intervallanalyse stellt neben den arithmetischen Operatoren $+, -, *, /$ auch eine Auswahl der elementaren Funktionen, wie sin, cos, $\sqrt{\cdot}$, usw., zur Verfügung. Diese Funktionen können in der Regel effizient anhand der entsprechenden reellen Funktionen dargestellt werden, indem zusätzlich Informationen über Monotonie eingearbeitet werden.

Kapitel 3

Methoden zur diskreten Arbeitsraumanalyse

In diesem Kapitel werden Methoden zur kinematischen Untersuchung von Parallelkinematikmaschinen an diskreten Punkten innerhalb des Arbeitsraums vorgestellt. Im Abschnitt 3.2 wird die kinetostatische Methode verwendet, um ein modulares Modell für PKM zu erstellen. Abschnitt 3.3 beschreibt einen generischen Algorithmus zur vollständigen Linearisierung von Manipulatoren. Anhand des linearisierten Modells werden der Einfluss von Fertigungsfehlern (Abschnitt 3.4), die Steifigkeit (Abschnitt 3.5) und die Kalibrierung (Abschnitt 3.6) einer PKM untersucht. Beispiele werden im Abschnitt 3.7 diskutiert.

3.1 Einführung

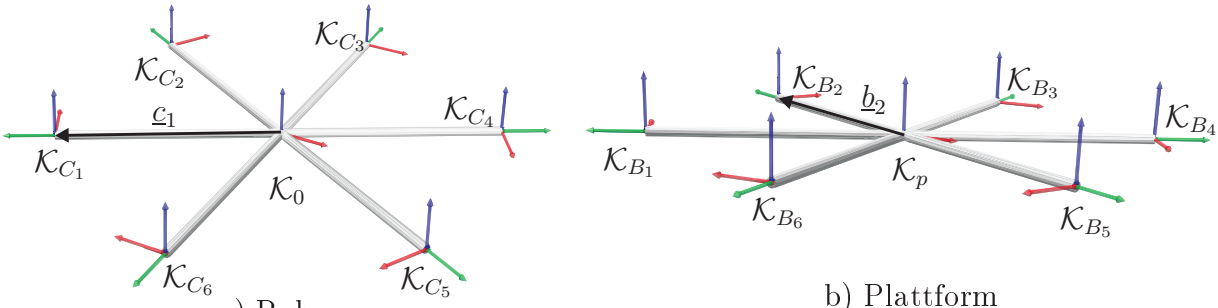
Die Modellbildung für PKM ist verglichen mit seriellen Robotern erheblich aufwendiger. Bei einem seriellen Roboter ist die Berechnung der direkten Kinematik stets eindeutig und kann z. B. mithilfe der DENAVIT-HARTENBERG-Parameter [26] automatisiert werden. Dagegen ist die inverse Kinematik, bei der alle Gelenkgrößen des Roboters aus einer gegebenen Position und Orientierung des End-Effektors bestimmt werden, mit erheblich mehr Aufwand verbunden. Im Prinzip ist die inversen Kinematik bei seriellen und parallelen Robotern identisch: aufgrund der kinematischen Schleifen kann bei einer PKM jedes Bein als serieller Manipulator betrachtet werden, dessen End-Effektor-Position und -Orientierung durch die Plattform der PKM vorgegeben ist. Der Unterschied besteht nur darin, dass die kinematische Struktur der Beine bei vielen PKM eine relativ einfache Lösung der inversen Kinematik ermöglicht. Dagegen ist die direkte Kinematik von PKM sehr aufwendig, denn dazu muss die gekoppelte Bewegung von allen sechs kinematischen Ketten berücksichtigt werden. Wie in Abschnitt 2.2.5 bereits erwähnt wurde, ist für das daraus resultierende Gleichungssystem keine geschlossene Lösung bekannt. Folglich lassen sich viele für serielle Roboter entwickelte Analyseverfahren nicht anwenden, da diese Verfahren eine geschlossene Lösung der direkten Kinematik erfordern. Die Analyse der kinematischen Eigenschaften ist für PKM aber besonders wichtig, denn die Eigenschaften hängen stark von der betrachteten Pose ab und variieren über dem Arbeitsraum erheblich.

Die Modellierung der Kinematik räumlicher paralleler Roboter erfordert für jeden Roboter ein speziell angepasstes komplexes Modell. Um die Analyse zu vereinfachen, wird in diesem Kapitel eine Methode vorgestellt, die es erlaubt, PKM aus einem Baukasten von Modulen zusammenzustellen, um relativ schnell ein vollständiges Modell zur Untersuchung der direkten und inversen Kinematik zu erhalten. Ein besonderer Fokus liegt dabei auf einer Methode, mit der neue Varianten automatisch zu einem vollständigen kinetostatischen Modell zusammengefügt werden können, so dass die Zeit für die Erstellung des Modells drastisch verkürzt wird. Dazu wird das Modell des Gesamtsystems aus funktionalen Modulen aufgebaut, welche die kinetostatischen Übertragungselemente (Abschnitt 2.1) enthalten und sich intelligent verknüpfen lassen. Die Module werden daher so gestaltet, dass ein generischer Algorithmus verwendet werden kann, um die Schließbedingungen automatisch zusammenzustellen und zu lösen. Die so erstellten Modelle besitzen die Gestalt eines kinetostatischen Übertragungselements und können daher mit generischen Algorithmen[1] untersucht werden. Die Analyse basiert auf einem neuen Algorithmus zur vollständigen Linearisierung von Mehrkörpersystemen (MKS). Ein besonderes Augenmerk liegt dabei auf dem Einfluss von geometrischen Parametern, die in den idealisierten Bindungsgleichungen entfallen. Ein anschauliches Beispiel für solche Idealisierungen ist der Abstand der Achsen in einem Kardangelenk, der bei einem idealen Gelenk mit Null angenommen wird und daher als Parameter nicht explizit eingeführt wird. Aufgrund von Fertigungstoleranzen lässt sich der tatsächliche Abstand aber nicht auf Null reduzieren. Die vorgeschlagene Linearisierung ermöglicht es, die partielle Ableitung der Kinematik auch bezüglich dieses Parameters zu berechnen, selbst wenn die interessierenden Parameter in das Modell nicht eingearbeitet werden. Die linearisierte Form wird verwendet, um eine Sensitivitätsanalyse durchzuführen und damit den Einfluss von Fertigungsfehlern auf die Genauigkeit der Maschine zu untersuchen. Weiterhin wird gezeigt, dass das lineare Modell direkt die Untersuchung der Steifigkeit zulässt, sofern die jeweiligen Steifigkeiten aller Bauteile bekannt sind.

3.2 Modulares kinetostatisches Modell

Die kinetostatische Methode stellt einen Baukasten von elementaren Komponenten zur Verfügung, aus denen sich komplexe Mehrkörpersysteme (MKS) aufbauen lassen. Topologisch gesehen lassen sich PKM aus einem kleinen Vorrat an Komponenten zusammenstellen. Es liegt daher nahe, die wiederkehrenden Bausteine (Beine, Plattform und Rahmen) in einer Programmbibliothek zusammenzustellen. Bei der Implementierung stellt sich jedoch heraus, dass das automatische Zusammenfügen dieser einfachen Bausteine zu komplexen Mechanismen erhebliche Probleme aufwirft, die von Hand gelöst werden müssen. Bei seriellen Robotern lassen sich die Komponenten in Form von Übertragungselementen in Reihe schalten, so dass unmittelbar das kinetostatische Modell des MKS entsteht. Bei parallelen Robotern müssen dagegen zunächst die kinematischen Schleifen gelöst werden, was jedoch eine dem speziellen Problem angepasste Vorgehensweise erfordert.

[1] Ein generischer Algorithmus ist ein Verfahren, das nicht für jeden Mechanismus spezialisiert werden muss. Stattdessen lassen sich generische Algorithmen auf eine ganze Klasse von Modellen anwenden. Bei der kinetostatischen Methode sind die wichtigsten generischen Algorithmen die kinematischen Differentiale und der Formalismus zum Aufstellen der Bewegungsgleichungen.



a) Rahmen b) Plattform

Abbildung 3.1: Die Module *Plattform* und *Rahmen* mit den Schnittstellen $\mathcal{K}_{B_i}$ und $\mathcal{K}_{C_i}$ für die Beinmodule

Der hier vorgestellte Ansatz besteht im Wesentlichen darin, Schnittstellen zu definieren, um die erforderlichen Informationen für den Zusammenbau von den Modulen abzurufen. Diese Informationen werden dabei zusammen mit dem Modul gespeichert und beim Zusammenbau generisch abgerufen. Anhand solcher Module wird das kinetostatische Modell des gesamten Manipulators zusammengesetzt, welches wiederum die Form eines kinetostatischen Übertragungselements besitzt. Daher lassen sich generische Algorithmen auf alle Maschinen anwenden, die aus dem Baukasten hervorgehen.

3.2.1 Module und Schnittstellen

Um möglichst viele Maschinen beschreiben zu können, wird ein *modulares Maschinenmodell* betrachtet, dessen innere Übertragungsfunktionen automatisch aus den Funktionen der Komponenten zusammengestellt werden. Dazu wird die Maschine in funktionale Komponenten unterteilt, welche die Module für ein Baukastensystem bilden. So ergeben sich die austauschbaren Module *Rahmen* (Abb. 3.1a), *Plattform* (Abb. 3.1b) und *Bein* (Abb. 3.2).

Das Modul *Rahmen* definiert die Geometrie des Maschinenständers durch die Position und Orientierung der Koordinatensysteme $\mathcal{K}_{C_i},\ i = 1, \ldots, 6$ (Abb. 3.1a). Diese Koordinatensysteme $\mathcal{K}_{C_i}$ werden im MKS durch starre Körper mit dem Weltkoordinatensystem $\mathcal{K}_0$ verbunden und fungieren als Schnittstelle, an welche die Beine angeschlossen werden. Die Modellierung der starren Körper erfolgt durch kinetostatische Übertragungselemente. Für die in dieser Arbeit betrachteten Maschinen SSM und *Linapod* werden Module verwendet, welche die Geometrie des Rahmens nach Tabelle 2.2 (Seite 27) und Tabelle 2.3 (Seite 29) umsetzen.

Das Modul *Plattform* beschreibt bezüglich des Koordinatensystems $\mathcal{K}_p$ die relativen Positionen der Koordinatensysteme $\mathcal{K}_{B_i},\ i = 1, \ldots, 6$, welche die Schnittstellen für die plattformseitigen Anlenkpunkte der Beine bilden (Abb. 3.1b). Dabei werden die Koordinatensysteme $\mathcal{K}_{B_i}$ wiederum durch starre Körper mit dem End-Effektor-Koordinatensystem $\mathcal{K}_p$ verbunden. Weiterhin bilden die Weltkoordinaten $\underline{y}$ der Plattform sowie deren Parametrisierung einen wesentlichen Teil des Moduls *Plattform*. Die Weltkoordinaten werden in dem Modul *Plattform* als *virtueller Manipulator* implementiert, d. h. es wird eine Kette von Übertragungselementen verwendet, die mit sechs Gelenken die Plattform mit der Basis verbindet. Die Übertragungsfunktion dieser Kette und der starren Verbindungen

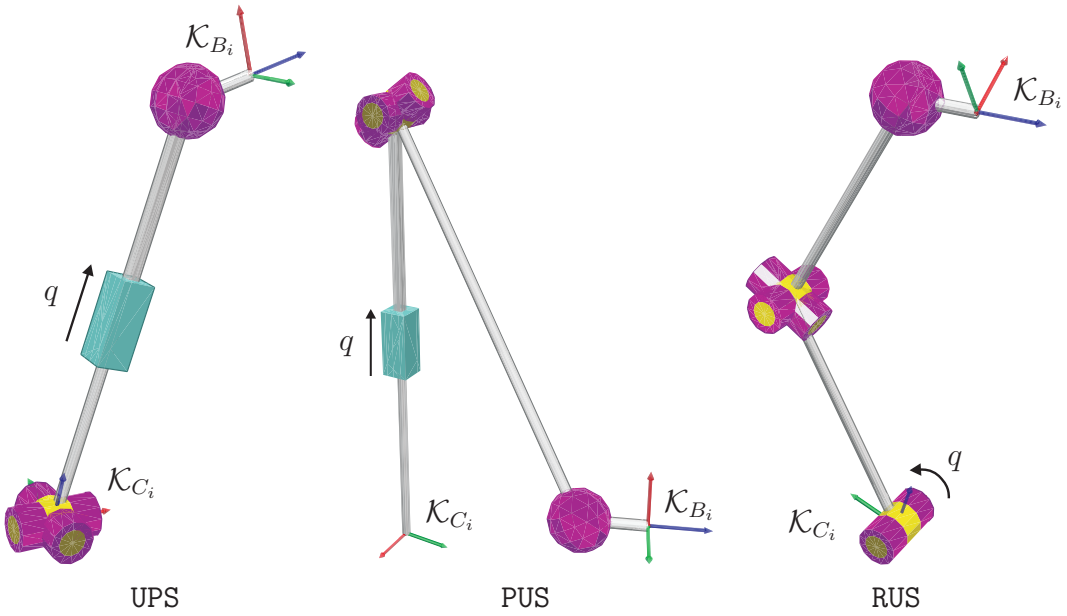


Abbildung 3.2: Die verschiedenen Beinmodule in MOBILE

der Plattform wird mit $\mu(\underline{y})$ bezeichnet[2] und berechnet die kinetostatische Übertragung von $\mathcal{K}_0$ auf die Koordinatensysteme $\mathcal{K}_p$ und $\mathcal{K}_{B_i}$. Die Gelenkkoordinaten der virtuellen Kette entsprechen dabei gerade den Weltkoordinaten $\underline{y}$. Dies kann z. B. in Form von drei orthogonalen Schubgelenken für die kartesische Position $\underline{r}_p = [x, y, z]^{\mathrm{T}}$ und einem Kugelgelenk in KARDAN-Winkeln $(\varphi, \psi, \theta)^{\mathrm{T}}$ für die Orientierung[3] $\underline{\underline{R}}_p$ geschehen. Damit ergibt sich dann $\underline{y} = [x, y, z, \varphi, \psi, \theta]^{\mathrm{T}}$. Grundsätzlich sind die Weltkoordinaten jedoch ein Detail des Moduls *Plattform*, so dass verschiedene Plattformen über unterschiedliche Parametrisierungen verfügen können. Hier werden beispielhaft zwei verschiedene *Plattform*-Module für die Maschinen SSM (Tabelle 2.2, S. 27) und *Linapod* (Tabelle 2.3, S. 29) eingeführt.

Das kinematische Übertragungsverhalten der Maschine wird wesentlich durch das Modul *Bein* bestimmt, das die topologische Struktur der Gelenke und die Geometrie der starren Körper beschreibt. Die verschiedenen Module unterscheiden sich zunächst durch die Art und Anordnung der Gelenke und starren Körper, d. h. es wird für jede der drei Varianten `PUS`, `UPS` und `RUS` eine spezialisierte Version des Moduls benötigt. Das Modul *Bein* beinhaltet daher die Schließbedingungen $\nu_i(B_i, C_i, q_i)$, wobei für das jeweilige Bein in Abhängigkeit der verallgemeinerten Koordinate q_i der Punkt A_i ermittelt wird. Damit kann der Abstand zwischen den beiden Punkten A_i und B_i (vgl. Abb. 2.8) als elementare Messung dargestellt werden, welche in der kinetostatischen Methode als Bindungsgleichung ν_i fungiert. Diese Bindungsgleichung wird zum Aufstellen der direkten und inversen Kinematik benötigt. Für die einfachere inverse Kinematik wird die Berechnung der verallgemeinerten Koordinate $q_i = q_i(B_i, C_i)$ dezentral durch eine Funktion des Moduls *Bein* realisiert. Für die hier betrachteten Beine kann dies stets in geschlossener Form erfolgen,

[2] Die Funktion μ stellt eine Verkettung von kinetostatischen Übertragungsfunktionen dar; folglich bildet die Funktion den kinematischen Zustand des Inertialsystems $\mathcal{K}_0$ und den kinematischen Zustand $\underline{y}$ auf die Koordinatensysteme $\mathcal{K}_p$ und $\mathcal{K}_{B_i}$ ab. Umgekehrt existiert damit auch die Funktion der Kraftübertragung, die in der entgegengesetzten Richtung wirkt. Für eine kompaktere Notation wird jedoch darauf verzichtet, die Abhängigkeit vom Inertialsystem $\mathcal{K}_0$ anzugeben.

[3] Für die Parametrisierung der Drehung siehe auch Abschnitt 2.2.8.

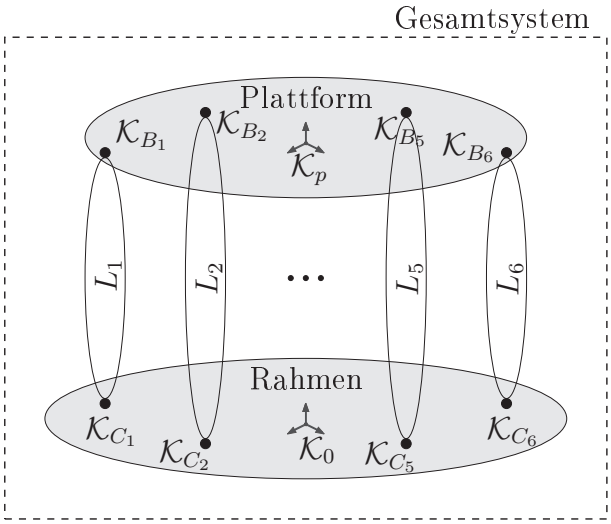


Abbildung 3.3: Aufbau einer generischen modularen Parallelkinematikmaschine

indem die Übertragungselemente des Beins ausgewertet werden. Damit wird im Prinzip die Lösung der inversen Kinematik nach Abschnitt 2.2.4 implementiert. Dagegen wird die direkte Kinematik durch das Zusammenspiel der sechs unabhängigen Beine bestimmt. Zur Lösung der direkten Kinematik besitzt jedes Modul *Bein* daher eine Bindungsgleichung, die an einen zentralen Gleichungslöser übergeben wird. Dazu wird die für jedes Bein spezifische Bindungsgleichung als Übertragungselement dargestellt, so dass eine definierte Schnittstelle entsteht, um die Gleichung für einen generischen Gleichungslöser zu einem Gleichungssystem zusammenzustellen. Die Bindungsgleichung (2.16) ist spezifisch für die jeweiligen Module und ergibt sich durch Auswertung der Übertragungsfunktion des Beins. Für jedes Bein werden im Prinzip die relativen Terme nach Gleichungen (2.21), (2.23) oder (2.30) eingesetzt. Im Rahmen der kinetostatischen Methode erfolgt dieses Einsetzen nicht symbolisch, sondern durch eine Auswertung der Relativkinematik des Beins.

Die Berechnung der Winkel des passiven Kugel-/Kardangelenks kann für die direkte und inverse Kinematik auf die gleiche Weise erfolgen. Durch Projektionen können die Winkel der Kugel-/Kardangelenke stets explizit durch spezialisierte Funktionen des Moduls *Bein* ermittelt werden. Dies entspricht der Vorgehensweise, wie sie bei der Bestimmung der komplementären Winkel für räumliche Gelenkvierecke bekannt ist. Nachdem sowohl die Weltkoordinaten der Plattform als auch die verallgemeinerten Koordinaten der Beine ermittelt wurden, stellt jedes Bein eine entkoppelte kinematische Schleife dar.

Ausgehend von dieser Aufteilung der Funktionalität nach Tabelle 3.1 unter den Modulen *Plattform*, *Rahmen* und *Bein* wird ein generisches Modul für den automatisierten Zusammenbau des Gesamtsystems erstellt. An das Modul *Gesamtsystem* wird je eine *Plattform*

Tabelle 3.1: Schnittstellen und bereitgestellten Funktionen der Module

Modul	Daten	Funktion
Rahmen	$\mathcal{K}_0, \mathcal{K}_{C_i}$	
Plattform	$\mathcal{K}_p, \mathcal{K}_{B_i}, \underline{y}$	$\mu(\underline{y})$
Bein	$\mathcal{K}_{B_i}, \mathcal{K}_{A_i}, \mathcal{K}_{C_i}, q_i$	$\nu_i(B_i, C_i, q_i), q_i(B_i, C_i)$

und ein *Rahmen* übergeben (Abb. 3.3). Danach werden dem System sechs *Beine* hinzugefügt, so dass alle Bausteine einer Maschine vorhanden sind. Anhand von generischen Algorithmen lassen sich diese Bausteine systematisch dazu verwenden, aus den in Modulen vorliegenden Übertragungselementen das kinetostatische Modell für die direkte und inverse Kinematik zusammenzustellen. Durch die oben festgelegten Schnittstellen wird der interne Aufbau der Beine, sowie die Geometrie aller Bauteile von der Struktur des Systems abstrahiert. Dies ermöglicht, eine Bibliothek verschiedener Module für *Plattform*, *Rahmen* und *Beine* anzulegen, die innerhalb einer Bearbeitungszeit von Minuten beliebig miteinander kombiniert werden können.

3.2.2 Inverse Kinematik

Zur Berechnung der inversen Kinematik werden die Weltkoordinaten der Plattform $\underline{y}$ vorgegeben. Gesucht sind die verallgemeinerten Koordinaten $\underline{q}$ der Antriebe sowie die Gelenkwinkel $\underline{\beta}$ der passiven Gelenke. Für den Aufbau eines Modells der inversen Kinematik wird der folgende Algorithmus verwendet.

Algorithmus 1: `Generische inverse Kinematik`

1. Das Modul *Rahmen* berechnet alle $\mathcal{K}_{C_i}$.
2. Das Modul *Plattform* berechnet mit $\mu(\underline{y})$ aus gegebenen Weltkoordinaten $\underline{y}$ Position und Orientierung der plattformseitigen Anlenkpunkte $\mathcal{K}_{B_i}$.
3. Jedes *Bein* berechnet die verallgemeinerte Koordinate $q_i = q_i(B_i, C_i)$.
4. Jedes *Bein* berechnet aus (B_i, C_i, q_i) die abhängigen Koordinaten $\underline{\beta}_i$ der passiven Gelenke.

Alle kinematischen Berechnungen des Algorithmus werden durch Auswertungen von kinetostatischen Übertragungselementen umgesetzt. Durch die Verkettung entsteht ein *Superelement*, das die vollständige Übertragung der kinematischen Zustände erlaubt. Die Weltkoordinaten $\underline{y}$ sowie deren Ableitungen $\dot{\underline{y}}, \ddot{\underline{y}}$ werden damit auf die verallgemeinerten Koordinaten $\underline{q}, \dot{\underline{q}}, \ddot{\underline{q}}$ sowie die abhängigen Koordinaten $\underline{\beta}, \dot{\underline{\beta}}, \ddot{\underline{\beta}}$ abgebildet. Es steht also ein Algorithmus zur Verfügung, der die Funktion $\underline{q} = \underline{\varphi}^{\mathrm{DK}}(\underline{y})$ und ihre Ableitungen berechnen kann. Das Problem der inversen Kinematik ist damit für alle Maschinen einheitlich gelöst.

3.2.3 Direkte Kinematik

Bei der direkten Kinematik werden aus gegebenen verallgemeinerten Koordinaten $\underline{q}$ die Weltkoordinaten $\underline{y}$ der Plattform ermittelt. Auch hier soll ein Superelement für die ganze Maschine ermittelt werden. Für die Berechnung der direkten Kinematik ergibt sich folgender Algorithmus:

Algorithmus 2: `Generische direkte Kinematik`

1. Das Modul *Rahmen* berechnet alle $\mathcal{K}_{C_i}$.
2. Jedes Modul *Bein* übergibt seine Bindungsgleichung $\nu_i(B_i, C_i, q_i)$ an den Gleichungslöser. Die *Plattform* übergibt $\mu(\underline{y})$ an den Gleichungslöser. Dieser stellt daraus ein implizites nichtlineares Gleichungssystem $\underline{\Gamma}(\underline{q}, \underline{y}) = 0$ zusammen.

3. Der Gleichungslöser berechnet eine spezielle Lösung $\underline{y}^*$ für das Gleichungssystem $\underline{\Gamma}(\underline{q}, \underline{y})$ mit einem NEWTON-RAPHSON-Verfahren.
4. Die *Plattform* berechnet alle $\mathcal{K}_{B_i}$ mit $\mu(\underline{y}^*)$.
5. Jedes *Bein* berechnet aus (B_i, C_i, q_i) die abhängigen Koordinaten $\underline{\beta}_i$ der passiven Gelenke.

Auch bei der direkten Kinematik lassen sich alle Schritte durch kinetostatische Übertragungsfunktionen abbilden. Das nichtlineare Gleichungssystem Γ repräsentiert dabei gerade den impliziten Kern der zu lösenden kinematischen Schleifen. Nach Absschnitt 2.1.3 lässt sich aber das Lösen der Bindungsgleichungen als kinetostatisches Übertragungselement abbilden. Dieses Element ermittelt neben einer Lösung $\underline{y}^*$ auch die zeitlichen Ableitungen $\dot{\underline{y}}^*, \ddot{\underline{y}}^*$ sowie die Kraftübertragung in umgekehrter Richtung. Die komplexen Details eines solchen Gleichungslösers sind hinlänglich bekannt, und eine Diskussion würde den Rahmen dieser Arbeit sprengen; eine ausführliche Beschreibung der Theorie [59] und der Implementierung [60] ist in der Literatur zu finden. Zusammenfassend entsteht eine Methode, die punktweise die direkte Kinematik $\underline{y} = \underline{\varphi}^{\mathrm{DK}}(\underline{q})$ und ihre Ableitungen berechnen kann.

3.3 Linearisierung

Das kinetostatische Modell liefert für die in dieser Arbeit betrachteten PKM eine Möglichkeit, die kinematischen Übertragungsfunktionen für Position, Geschwindigkeit und Beschleunigung sowie für die Kräfte zu bestimmen. Ausgehend von diesen elementaren Funktionen werden nun generische Algorithmen vorgestellt, die es erlauben, verschiedene Analysen für PKM durchzuführen. Eine besondere Rolle spielt dabei die geometrische Linearisierung von PKM, die in späteren Abschnitten für die Analyse der Fertigungsfehler (Abschnitt 3.4), zur Berechnung der Steifigkeit (Abschnitt 3.5) und zur Kalibrierung (Abschnitt 3.6) verwendet wird.

Als *nominelle Parameter* $\underline{g}_0$ werden die Geometrieparameter bezeichnet, die als Sollwerte der Geometrie aus dem Entwurf der Maschine vorgegebenen sind. Davon weicht die tatsächliche Geometrie einer Maschine aufgrund von unvermeidlichen Fehlern bei der Herstellung ab. Formal wird die Linearisierung durch partielle Differentiation der Funktion der direkten Kinematik $\underline{\varphi}^{\mathrm{DK}}(\underline{q}, \underline{g})$ nach den verallgemeinerten Koordinaten $\underline{q}$ und den geometrischen Parametern $\underline{g}$ eingeführt und es ergibt sich

$$\delta\underline{y} = \frac{\partial \underline{\varphi}^{\mathrm{DK}}(\underline{q}, \underline{g})}{\partial(\underline{q}, \underline{g})}\delta(\underline{q}, \underline{g}) = \underbrace{\frac{\partial \underline{\varphi}^{\mathrm{DK}}(\underline{q}, \underline{g})}{\partial \underline{q}}}_{\underline{\underline{J}}_q}\delta\underline{q} + \underbrace{\frac{\partial \underline{\varphi}^{\mathrm{DK}}(\underline{q}, \underline{g})}{\partial \underline{g}}}_{\underline{\underline{J}}_g}\delta\underline{g}. \tag{3.1}$$

Dabei sind $\delta\underline{y}, \delta\underline{q}, \delta\underline{g}$ infinitesimale Variationen der Parameter. Mit $\underline{\underline{J}}_q$ wird die JACOBI-Matrix bezeichnet, welche die Manipulierbarkeit des Roboters charakterisiert. Die Matrix $\underline{\underline{J}}_g$ ist ebenfalls eine JACOBI-Matrix, welche die Sensitivität der Position und Orientierung der Plattform bezüglich Änderungen der geometrischen Parameter beschreibt. Für eine gegebene Lage des Manipulators wird bei festgehaltenen verallgemeinerten Koordinaten

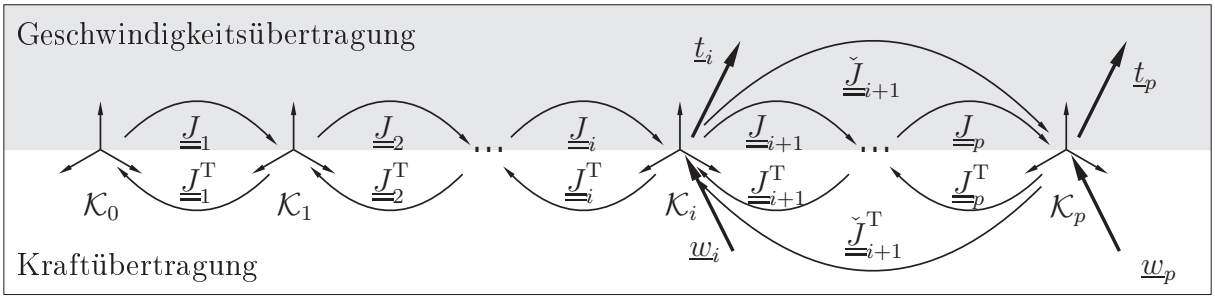


Abbildung 3.4: Geschwindigkeits- und Kraftübertragung in einer Kette von Übertragungselementen

$\delta\underline{q} = \underline{0}$ der Einfluss von Änderungen der geometrischen Parameter $\underline{g}$ auf Position und Orientierung der Plattform durch

$$\delta\underline{y} = \underline{\underline{J}}_g \delta\underline{g} \tag{3.2}$$

beschrieben, wobei der Vektor $\delta\underline{g}$ alle praktisch auftretenden Abweichungen von der nominellen Geometrie beinhaltet. Für serielle und für einige parallele Roboter lässt sich eine geschlossene Lösung für die direkte Kinematik angeben, so dass sich Gleichung (3.1) direkt auswerten lässt, um $\underline{\underline{J}}_q$ und $\underline{\underline{J}}_g$ durch symbolische Differentiation zu berechnen. Für räumliche PKM kann die Funktion der direkten Kinematik $\underline{\varphi}^{\mathrm{DK}}$ im Allgemeinen nicht geschlossen angegeben werden, da sich $\underline{\varphi}^{\mathrm{DK}}$ nur punktweise durch die iterative Lösung eines nichtlinearen Gleichungssystems bestimmen lässt. Dies gilt insbesondere für die PKM, für die im vorangegangenen Abschnitt die direkte Kinematik angegeben wurde. Anstelle der Funktion der direkten Kinematik lassen sich die Schließbedingungen untersuchen, die sich als implizites Gleichungssystem zu

$$\underline{\Gamma}(\underline{y}, \underline{g}) = \underline{0} \tag{3.3}$$

ergeben. Durch Differenzieren folgt dann

$$\underbrace{\frac{\partial \underline{\Gamma}(\underline{y}, \underline{g})}{\partial \underline{y}}}_{\underline{\underline{A}}} \delta\underline{y} + \underbrace{\frac{\partial \underline{\Gamma}(\underline{y}, \underline{g})}{\partial \underline{g}}}_{\underline{\underline{B}}} \delta\underline{g} = \underline{0}. \tag{3.4}$$

Daraus ergibt sich unmittelbar eine differentielle Abbildung $\delta\underline{y} = -\underline{\underline{A}}^{-1}\underline{\underline{B}}\delta\underline{g}$ zwischen den Weltkoordinaten $\underline{y}$ und den geometrischen Parametern $\underline{g}$, falls $\underline{\underline{A}}$ regulär ist. Dieser Ansatz hat den Nachteil, dass für räumliche Mechanismen die Matrix $\underline{\underline{A}}$ analytisch praktisch nicht invertiert werden kann, da die dabei entstehenden Ausdrücke auch mit Computeralgebra kaum behandelt werden können. Weiterhin ist es nicht möglich, mit diesem Ansatz die geometrischen Parameter zu berücksichtigen, die durch eine Idealisierung bei der Formulierung der Schließbedingungen $\underline{\Gamma}$ eliminiert wurden. Zum Beispiel wird der Abstand der Achsen in Kardangelenken normalerweise vernachlässigt, da sich andernfalls die Anzahl und Komplexität der Schließbedingungen relevant erhöhen würde. Im Hinblick auf eine umfassende Sensitivitätsanalyse müssen aber gerade die Einflüsse dieser Fehler betrachtet werden.

Die kinetostatische Methode erlaubt die Formulierung eines generischen Algorithmus zur Untersuchung der Fehlerempfindlichkeit allgemeiner räumlicher Manipulatoren [104, 107].

Dabei wird davon ausgegangen, dass mithilfe des modularen Ansatzes in Abschnitt 3.2 ein vollständiges kinetostatisches Modell erstellt wurde. Damit liegt eine allgemeine Formulierung zur Berechnung der Positions-, Geschwindigkeits-, Beschleunigungs- und Kraftübertragung vor, welche die Form einer Kette von Übertragungselementen hat (Abb 3.4). Die Übertragungselemente der Kette können dabei auch Übertragungselemente zur Lösung kinematischer Schleifen enthalten. Für eine Pose $\underline{q}$ wird ein virtueller Geschwindigkeitswinder $\delta\underline{t}_i^{\mathrm{T}} = [\delta\underline{\varphi}_i^{\mathrm{T}}, \delta\underline{r}_i^{\mathrm{T}}]$ am Koordinatensystem $\mathcal{K}_i$ eingeführt, wobei $\delta\underline{r}_i$ eine virtuelle Verschiebung darstellt und $\delta\underline{\varphi}_i$ einer virtuellen Verdrehung aus dem Raum der Starrkörperdrehungen entspricht. Für den virtuellen Geschwindigkeitswinder $\delta\underline{t}_p$ am End-Effektor-Koordinatensystem $\mathcal{K}_p$ ergibt sich die lineare Beziehung

$$\delta\underline{t}_p = \left(\underline{\underline{J}}_{i+1}\underline{\underline{J}}_{i+2}\cdots\ \underline{\underline{J}}_p\right)\delta\underline{t}_i = \check{\underline{\underline{J}}}_{i+1}\delta\underline{t}_i, \tag{3.5}$$

aus dem Produkt der JACOBI-Matrizen $\underline{\underline{J}}_i$ der Relativbewegungen, wobei die Geschwindigkeiten nach Gleichung (2.2) vom Koordinatensystem $\mathcal{K}_{i-1}$ auf das Koordinatensystem $\mathcal{K}_i$ übertragen werden. Mithilfe der kinematischen Differentiale (Abschnitt 2.1.5) lässt sich die Matrix $\check{\underline{\underline{J}}}_{i+1}$ berechnen, welche die Sensitivität des Koordinatensystem $\mathcal{K}_p$ bezüglich virtueller Verrückungen $\delta\underline{t}_i$ am Koordinatensystem $\mathcal{K}_i$ beschreibt. Das Zusammenfügen aller Matrizen $\underline{\underline{J}}_i$ ergibt die gesuchte Matrix

$$\underline{\underline{J}}_g = \left[\check{\underline{\underline{J}}}_1, \check{\underline{\underline{J}}}_2, \ldots, \check{\underline{\underline{J}}}_p\right]. \tag{3.6}$$

Die vollständige Linearisierung kann also mit $6p$ Auswertungen der Geschwindigkeitsübertragungsfunktion für einen beliebigen Roboter berechnet werden.

Diese Methode liefert die exakten Koeffizienten der JACOBI-Matrix $\underline{\underline{J}}_g$. Für komplexe Mechanismen wie PKM ist dieser Algorithmus rechenintensiv, denn eine PKM besteht aus zahlreichen Gelenken und starren Körpern. Beispielsweise enthält das Modell der Maschine *Linapod* $p = 42$ Koordinatensysteme, die alle mit virtuellen Verrückungen $\delta\underline{t}_i$ beaufschlagt werden müssen. Anstelle der Geschwindigkeiten wird daher die Kraftübertragung betrachtet. Ein Kraftwinder $\underline{w}_i^{\mathrm{T}} = [\underline{\tau}_i^{\mathrm{T}}, \underline{f}_i^{\mathrm{T}}]$ am Koordinatensystem $\mathcal{K}_i$ besteht aus dem Moment $\underline{\tau}_i$, das vom Koordinatensystem $\mathcal{K}_{i+1}$ auf das System $\mathcal{K}_i$ einwirkt, sowie aus einer entsprechenden Kraft $\underline{f}_i$. Gemäß der kinetostatischen Dualität wird dieser Kraftwinder von der Plattform $\mathcal{K}_p$ in Richtung des Inertialsystems $\mathcal{K}_0$ übertragen, und es ergibt sich die Übertragungsfunktion für das Koordinatensystem $\mathcal{K}_i$ zu

$$\underline{w}_i = \left(\underline{\underline{J}}_{i+1}\underline{\underline{J}}_{i+2}\cdots\ \underline{\underline{J}}_p\right)^{\mathrm{T}}\delta\underline{w}_p = \check{\underline{\underline{J}}}_{i+1}^{\mathrm{T}}\delta\underline{w}_p. \tag{3.7}$$

Wird nun die Kraftübertragung für die Schnittkräfte an allen Koordinatensystemen zusammengefasst, ergibt sich

$$\underline{w}_g = \underline{\underline{J}}_g^{\mathrm{T}}\underline{w}_p, \tag{3.8}$$

wobei der Vektor $\underline{w}_g$ aller Schnittkräfte sich aus

$$\underline{w}_g = \begin{bmatrix} \underline{w}_0 \\ \underline{w}_1 \\ \vdots \\ \underline{w}_{p-1} \end{bmatrix} = \begin{bmatrix} w_0^{(1)} \\ w_0^{(2)} \\ \vdots \\ w_{p-1}^{(6)} \end{bmatrix} \tag{3.9}$$

zusammensetzt und die JACOBI-Matrix nach Gleichung (3.6) die Form

$$\underline{\underline{J}}_g^{\mathrm{T}} = \begin{bmatrix} \check{\underline{\underline{J}}}_1^{\mathrm{T}} \\ \vdots \\ \check{\underline{\underline{J}}}_{p-1}^{\mathrm{T}} \\ \check{\underline{\underline{J}}}_p^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} (\underline{\underline{J}}_1 \underline{\underline{J}}_2 \cdots \underline{\underline{J}}_p)^{\mathrm{T}} \\ \vdots \\ (\underline{\underline{J}}_{p-1} \underline{\underline{J}}_p)^{\mathrm{T}} \\ \underline{\underline{J}}_p^{\mathrm{T}} \end{bmatrix} \tag{3.10}$$

hat. Diese Form der Übertragung weist den wesentlichen Vorteil auf, dass die Schnittkraft $\underline{w}_{i+1}$ rekursiv benutzt werden kann, um $\underline{w}_i$ zu berechnen. Entsprechend des kinetostatischen Übertragungsprinzips werden, beginnend von $\mathcal{K}_p$, sukzessive die Schnittkräfte aus

$$\underline{w}_i = \underline{\underline{J}}_{i+1}^{\mathrm{T}} \underline{w}_{i+1} \tag{3.11}$$

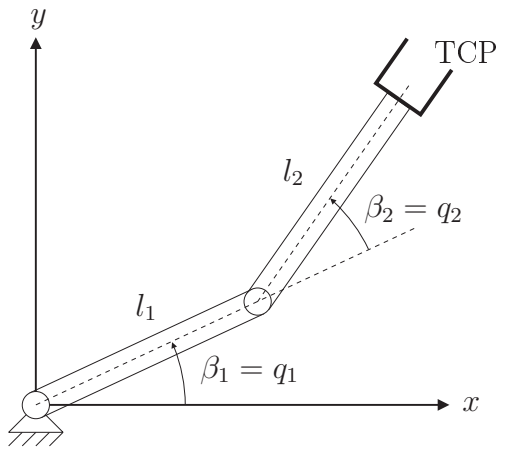
ermittelt. Daher werden mit *einer* Auswertung der Kraftübertragung alle $\underline{w}_i, i = 1, \ldots, p$, für eine gegebene Pseudokraft $\check{\underline{w}}_p^{(j)}$ bestimmt. Insgesamt werden daher nur *sechs* Auswertungen der Kraftübertragung benötigt, um die vollständige Matrix $\underline{\underline{J}}_g$ zu berechnen. Die Anzahl der Funktionsauswertungen hängt weder von der Anzahl der betrachteten Parameter noch von der Komplexität des Manipulators ab. Es ergibt sich damit der folgende Algorithmus zur Aufstellung der Sensitivitätsmatrix $\underline{\underline{J}}_g$:

Algorithmus 3: `Kraftbasierte Linearisierung`

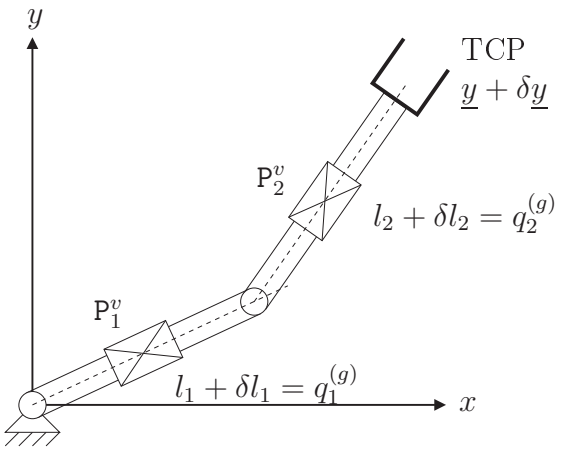
1. Berechne die direkte Kinematik $\underline{\varphi}^{\mathrm{DK}}(\underline{q}, \underline{g}_0)$ des Manipulators für die betrachteten verallgemeinerten Koordinaten $\underline{q}$ und die nominellen Parameter $\underline{g}_0$.
2. Wähle die Einheitskräfte $\check{\underline{f}}_x, \check{\underline{f}}_y, \check{\underline{f}}_z$ und die Einheitsmomente $\check{\underline{\tau}}_x, \check{\underline{\tau}}_y, \check{\underline{\tau}}_z$ mit $|\check{\underline{f}}_x| = |\check{\underline{f}}_y| = \ldots = |\check{\underline{\tau}}_z| = 1$ entlang der x-, y-, z-Achsen des Koordinatensystems $\mathcal{K}_p$.
3. Führe für jede dieser sechs Einheitslasten $\check{\underline{w}}_i$ die folgenden Schritte aus:
   (a) Präge die Last $\check{\underline{w}}_i$ am Koordinatensystem $\mathcal{K}_p$ ein.
   (b) Berechne alle Schnittkräfte und -momente $\underline{w}_i$ an jedem Koordinatensystem $\mathcal{K}_i$.
   (c) Speichere die so berechneten Kräfte/Moment $\underline{w}_i$ in der entsprechenden Zeile der JACOBI-Matrix $\underline{\underline{J}}_g$.

Der Aufwand für eine Auswertung der Kraftübertragung hängt nur von der Anzahl n_P der Elemente in der kinematischen Kette ab, und der Gesamtaufwand des Algorithmus ist damit von der Ordnung $\mathcal{O}(n_P)$. Bei der geschwindigkeitsbasierten Form müssen $6n_P$ Auswertungen der Geschwindigkeitsübertragungsfunktion bestimmt werden, die jeweils eine Komplexität von $\mathcal{O}(n_P)$ haben, so dass der Algorithmus insgesamt eine Ordnung von $\mathcal{O}(n_P^2)$ besitzt.

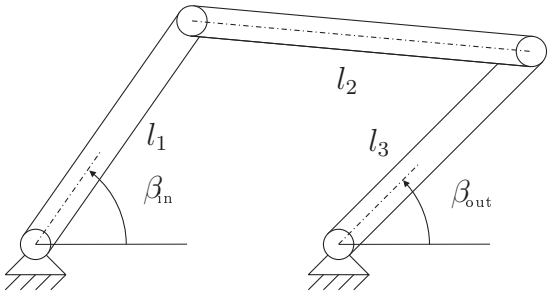
Damit ergibt sich mit dem kraftgestützten Algorithmus ein einfaches, aber effektives Verfahren zur Sensitivitätsanalyse für allgemeine Manipulatoren. Dabei wirkt sich die Tatsache, dass die Kraftübertragung auch intuitiv nachvollziehbar ist, positiv bei der Anwendung des Verfahrens aus. Ein Anwender kann auch ohne Computer eine überschlägige Sensitivitätsanalyse für einen gegebenen Manipulator durchführen. Dazu stelle er sich vor, der End-Effektor würde mit einer Last beaufschlagt. An jeder Stelle, an der unter Last eine Schnittkraft auftritt, würden Fehler proportional zum Betrag der Schnittkraft eine Lageänderung des End-Effektors in Richtung der angenommenen Last generieren.



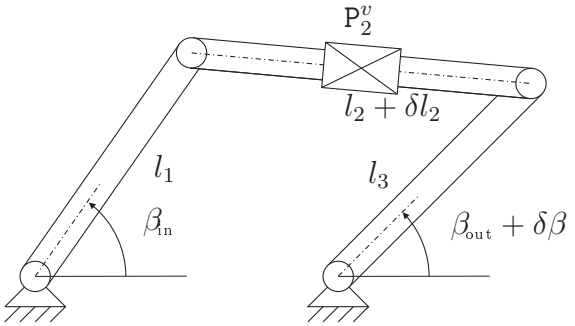
a) Einfacher ebener Manipulator mit zwei Freiheitsgraden



b) Ebener Manipulator mit zwei virtuellen Fehlergelenken $\mathtt{P}_i^v$ für Fehler in der Länge der Arme



c) Ebenes Gelenkviereck mit nomineller Geometrie



d) Ebenes Gelenkviereck mit virtuellem Fehlergelenk $\mathtt{P}_2^v$ in der Koppelstange

Abbildung 3.5: Prinzip der virtuellen Fehlergelenke

3.3.1 Interpretation der Linearisierung durch virtuelle Gelenke

Das Verfahren zur vollständigen Linearisierung der Kinematik von Robotern kann anhand des Konzepts der *virtuellen Gelenke* veranschaulicht werden, welches von Woernle vorgeschlagen wurde [137]. Dabei werden die geometrischen Parameter $\underline{g}$, die als Längen und Winkel bei der Beschreibung der starren Körper auftreten, durch Gelenke ersetzt, deren verallgemeinerte Koordinaten gerade diesen geometrischen Parametern entsprechen. Abbildung 3.5 illustriert, wie sich die Länge eines Roboterarms (Abb. 3.5a,b) oder die Länge der Koppelstange eines Gelenkvierecks (Abb. 3.5c,d) durch virtuelle Gelenke darstellen lassen. Durch diese Erweiterung des Modells werden die geometrischen Parameter $\underline{g}$ formal in zusätzliche verallgemeinerte Koordinaten $\underline{q}^{(g)}$ transformiert, so dass die Übertragungsfunktion $\underline{\varphi}^{\mathrm{DK}}(\underline{q}, \underline{q}^{(g)})$ eine Variation von $\underline{q}$ und $\underline{q}^{(g)}$ erlaubt. Damit steht unmittelbar ein geometrisch nichtlineares Modell zur Verfügung, das sowohl die Untersuchung von geometrischen Varianten des Mechanismus als auch eine Analyse der Empfindlichkeit bezüglich Schwankungen der Parameter $\underline{q}^{(g)}$ ermöglicht. Dazu wird bei festgehaltenen verallgemeinerten Koordinaten $\underline{q}$ das Übertragungsverhalten der virtuellen Gelenke untersucht, indem

die virtuellen Verrückungen $\delta\underline{t}_p$ des End-Effektor-Koordinatensystems[4] (EEF) bezüglich virtueller Verrückungen $\delta\underline{q}^{(g)}$ in den Gelenken betrachtet werden. Dieser Zusammenhang ist stets linear, so dass sich die JACOBI-Matrix mithilfe der kinematischen Differentiale bestimmen lässt (Abschnitt 2.1.5). Insbesondere spielt das *kraftgestützte* Verfahren zur Berechnung der JACOBI-Matrix eine große Rolle, denn bei der Sensitivitätsanalyse werden in der Regel mehr geometrische Parameter betrachtet, als der Manipulator Freiheitsgrade besitzt.

Die Idee, virtuelle Gelenke zur Beschreibung von möglichen Variationen der Geometrie heranzuziehen, wird dahingehend verallgemeinert, dass beliebige Arten geometrischer Abweichungen untersucht werden können. Dabei wird davon ausgegangen, dass die starren Körper wie im ursprünglichen Manipulator behandelt werden. Zusätzlich werden virtuelle Gelenke mit sechs Freiheitsgraden zwischen allen interessierenden Komponenten des Mechanismus eingefügt.

Zur Modellierung des allgemeinen virtuellen Gelenks wird das Sechs-Freiheitsgrade-Gelenk herangezogen, das in Abschnitt 2.1.2 beschrieben wurde. Da die nominelle Geometrie des Manipulators bereits durch die Körper und Gelenke des unmodifizierten Mechanismus beschrieben wurde, ist es hinreichend, die virtuellen Gelenke auf eine Variation ihrer verallgemeinerten Koordinaten in der Umgebung der nominalen Stellung zu beschränken. Damit vereinfacht sich die lokale JACOBI-Matrix aus Gleichung (2.10), da $\underline{x} = \underline{\theta} = \underline{0}$ gesetzt werden, und es ergibt sich für die Kraftübertragung der triviale Zusammenhang

$$\begin{bmatrix} \underline{\tau}_i \\ \underline{f}_i \\ \underline{Q}^{(\theta)} \\ \underline{Q}^{(x)} \end{bmatrix} = \underbrace{\begin{bmatrix} \underline{\underline{I}}_3 & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{I}}_3 \\ \underline{\underline{I}}_3 & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{I}}_3 \end{bmatrix}}_{\underline{\underline{J}}^{\mathrm{T}}_{\mathrm{lokal}}} \begin{bmatrix} \underline{\tau}_{i+1} \\ \underline{f}_{i+1} \end{bmatrix}. \tag{3.12}$$

Damit wird deutlich, dass die verallgemeinerten Kräfte $\underline{Q}^{(\theta)}, \underline{Q}^{(x)}$ eines solchen Gelenks identisch mit den Schnittkräften $\underline{\tau}_i, \underline{f}_i$ am Koordinatensystem $\mathcal{K}_i$ sind. Um nun die JACOBI-Matrix $\underline{\underline{J}}_g$ für den gesamten Manipulators zu berechnen, kann auf die Erweiterung des Modells um *virtuelle Gelenke* verzichtet werden, wenn die kraftgestützte Methode für die Berechnung der JACOBI-Matrix verwendet wird, und anstelle der verallgemeinerten Kräfte $\underline{Q}^{(\theta)}, \underline{Q}^{(x)}$ die Schnittkräfte $\underline{\tau}_i, \underline{f}_i$ betrachtet werden. Diese Überlegung führt dann unmittelbar auf den in Abschnitt 3.3 vorgestellten Algorithmus.

3.3.2 Optimaler Punkt des End-Effektors

Gleichung (3.2) beschreibt den Einfluss von Änderungen der geometrischen Parameter auf die Position und Orientierung des End-Effektor-Koordinatensystem (EEF). Da die Plattform ein starrer Körper im Raum ist, lässt sich kein eindeutiger Referenzpunkt festlegen. Stattdessen können unendlich viele verschiedene Punkte als End-Effektor angenommen werden. Daher stellt sich die Frage, ob es einen Punkt auf der Plattform gibt, der für eine gegebene Lage eine minimale Empfindlichkeit bezüglich Änderungen der geometrischen Parameter aufweist. Dieses Problem ist eng mit der Frage der kinematischen Manipulierbarkeit verwandt, wie es beispielsweise von Ranjbaran et al. [111] beschrieben wird. Dabei

[4] engl. *End-Effector Frame* (EEF)

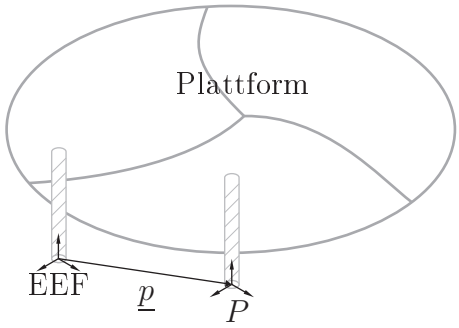


Abbildung 3.6: Verschiebung des Bezugspunkts vom ursprünglichen EEF zum Punkt P

wird ein optimaler End-Effektor-Punkt (engl. Optimal Operation Point) bestimmt, der die Manipulierbarkeit eines Roboters für eine gegebene Konfiguration optimiert. Die Grundidee besteht darin, die Konditionszahl κ der JACOBI-Matrix $\underline{\underline{J}}_g$ zu minimieren. Wie von Ranjbaran et al. [111] vorgeschlagen wurde, kann ein Schätzwert erster Ordnung $\widetilde{\kappa}_F$ für die Konditionszahl verwendet werden, wobei dieser Schätzwert von der FROBENIUS-Norm abgeleitet ist:

$$\widetilde{\kappa}_F^2(\underline{\underline{J}}_g) = \frac{\mathrm{Sp}^m(\underline{\underline{J}}_g \underline{\underline{J}}_g^{\mathrm{T}})}{m^m \det(\underline{\underline{J}}_g \underline{\underline{J}}_g^{\mathrm{T}})}. \tag{3.13}$$

Dieser Schätzwert kann verwendet werden, um die Sensitivität eines bestimmten Punkts bezüglich Änderungen der Geometrie zu bewerten, wobei m die Anzahl der Freiheitsgrade des Manipulators bezeichnet. Die Minimierung der Konditionszahl der Sensitivitätsmatrix $\underline{\underline{J}}_g$ entspricht dem Verfahren, wie es von Ranjbaran et al. [111] für die Manipulierbarkeit $\underline{\underline{J}}_q$ beschrieben wurde. Dabei wird hier die Betrachtung auf die allgemeinere JACOBI-Matrix $\underline{\underline{J}}_g$ erweitert. Die Matrix $\underline{\underline{J}}_g$ wird dazu in Blöcke aufgeteilt

$$\underline{\underline{J}}_g = \begin{bmatrix} \underline{\underline{R}} \\ \underline{\underline{T}} \end{bmatrix}, \tag{3.14}$$

wobei mit $\underline{\underline{R}}, \underline{\underline{T}}$ Blockmatrizen der Dimension $\mathbb{R}^{3\times 6p}$ bezeichnet werden, die den rotatorischen und den translatorischen Teil von $\underline{\underline{J}}_g$ darstellen. Weiterhin kann mit der Übergangsmatrix

$$\underline{\underline{J}}_P = \underline{\underline{U}}(\underline{p})\underline{\underline{J}}_g \quad \text{mit} \qquad \underline{\underline{U}}(\underline{p}) = \begin{bmatrix} \underline{\underline{I}}_3 & \underline{\underline{0}} \\ \widetilde{\underline{p}} & \underline{\underline{I}}_3 \end{bmatrix}, \tag{3.15}$$

die JACOBI-Matrix bezüglich des EEF einem anderen Referenzpunkt P zugeordnet werden [59], wobei mit $\underline{p}$ die relative Verschiebung zwischen EEF und P gegeben ist (Abb. 3.6). Dies entspricht der Übertragungsfunktion eines starren Körpers mit der Verschiebung $\underline{p}$. Da $\det(\underline{\underline{U}}(\underline{p})) = 1$ gilt, hängt die Determinante von $\underline{\underline{J}}_g$ nicht vom Referenzpunkt ab, denn es gilt

$$\det(\underline{\underline{U}}(\underline{p})\underline{\underline{J}}_g) = \det(\underline{\underline{U}}(\underline{p}))\det(\underline{\underline{J}}_g) = \det(\underline{\underline{J}}_g). \tag{3.16}$$

Für den optimalen End-Effektor-Punkt $\underline{p}^*$ minimiert der Vektor $\underline{p}^*$ die Konditionszahl κ_F der JACOBI-Matrix $\underline{\underline{J}}_g$, d. h. es gilt

$$\widetilde{\kappa}_F(\underline{\underline{J}}_g, \underline{p}^*) = \min_{\underline{p}} \widetilde{\kappa}_F(\underline{\underline{J}}_g, \underline{p}). \tag{3.17}$$

Eine notwendige Bedingung für ein Minimum von $\tilde{\kappa}_F$ [111] ist

$$\frac{\partial}{\partial \underline{p}} \frac{\mathrm{Sp}^m(\underline{\underline{J}}_P \underline{\underline{J}}_P^{\mathrm{T}})}{m^m \det(\underline{\underline{J}}_P \underline{\underline{J}}_P^{\mathrm{T}})}\bigg|_{\underline{p}=\underline{p}^*} = \underline{0}. \tag{3.18}$$

Da die Determinante $\det(\underline{\underline{J}}_P \underline{\underline{J}}_P^{\mathrm{T}})$ nicht von $\underline{p}$ abhängt, ergibt sich

$$\frac{\partial}{\partial \underline{p}} \mathrm{Sp}(\underline{\underline{J}}_P \underline{\underline{J}}_P^{\mathrm{T}}) = \underline{0}. \tag{3.19}$$

Nun wird für $\underline{\underline{J}}_P$ die Übergangsmatrix nach Gleichung (3.15) eingesetzt und es folgt

$$\underline{\underline{J}}_P \underline{\underline{J}}_P^{\mathrm{T}} = \underline{\underline{U}}(\underline{p}) \underline{\underline{J}}_g \underline{\underline{J}}_g^{\mathrm{T}} \underline{\underline{U}}(\underline{p})^{\mathrm{T}} \tag{3.20}$$

$$= \begin{bmatrix} \underline{\underline{R}}\underline{\underline{R}}^{\mathrm{T}} & \underline{\underline{R}}\underline{\underline{T}}^{\mathrm{T}}\tilde{\underline{p}} + \underline{\underline{R}}\underline{\underline{R}}^{\mathrm{T}} \\ \underline{\underline{T}}\underline{\underline{R}}^{\mathrm{T}} + \underline{\underline{R}}\underline{\underline{R}}^{\mathrm{T}}\tilde{\underline{p}} & \underline{\underline{T}}\underline{\underline{T}}^{\mathrm{T}} - \underline{\underline{T}}\underline{\underline{R}}^{\mathrm{T}}\tilde{\underline{p}} - \tilde{\underline{p}}\underline{\underline{R}}\underline{\underline{R}}^{\mathrm{T}}\tilde{\underline{p}} + \tilde{\underline{p}}\underline{\underline{R}}\underline{\underline{T}}^{\mathrm{T}} \end{bmatrix}, \tag{3.21}$$

was zu

$$\frac{\partial}{\partial \underline{p}} \left(\mathrm{Sp}(\underline{\underline{T}}\underline{\underline{T}}^{\mathrm{T}} - \underline{\underline{T}}\underline{\underline{R}}^{\mathrm{T}}\tilde{\underline{p}} + \tilde{\underline{p}}\underline{\underline{R}}\underline{\underline{T}}^{\mathrm{T}} - \tilde{\underline{p}}\underline{\underline{R}}\underline{\underline{R}}^{\mathrm{T}}\tilde{\underline{p}}) + \mathrm{Sp}(\underline{\underline{R}}\underline{\underline{R}}^{\mathrm{T}}) \right)\bigg|_{\underline{p}=\underline{p}^*} = \underline{0} \tag{3.22}$$

führt. Unter Ausnutzung von $\mathrm{Sp}(-\underline{\underline{T}}\underline{\underline{R}}^{\mathrm{T}}\tilde{\underline{p}}) = \mathrm{Sp}(\tilde{\underline{p}}\underline{\underline{R}}\underline{\underline{T}}^{\mathrm{T}})$ vereinfacht sich die Gleichung zu

$$\frac{\partial}{\partial \underline{p}} \mathrm{Sp}(-2\underline{\underline{T}}\underline{\underline{R}}^{\mathrm{T}}\tilde{\underline{p}} - \tilde{\underline{p}}\underline{\underline{R}}\underline{\underline{R}}^{\mathrm{T}}\tilde{\underline{p}})\bigg|_{\underline{p}=\underline{p}^*} = \underline{0}, \tag{3.23}$$

wobei die partiellen Ableitungen ohne $\tilde{\underline{p}}$ entfallen. Andererseits ergibt sich für die partiellen Ableitungen bezüglich $\tilde{\underline{p}}$

$$\frac{\partial}{\partial \underline{p}} \mathrm{Sp}(\underline{\underline{A}}\tilde{\underline{p}}) = \mathrm{vect}(\underline{\underline{A}}) = \begin{bmatrix} a_{32} - a_{23} \\ a_{13} - a_{31} \\ a_{21} - a_{12} \end{bmatrix} \quad \text{mit} \quad \underline{\underline{A}} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}. \tag{3.24}$$

Weiterhin gilt

$$\frac{\partial}{\partial \underline{p}} \mathrm{Sp}(\tilde{\underline{p}}\underline{\underline{A}}\underline{\underline{A}}^{\mathrm{T}}\tilde{\underline{p}}^{\mathrm{T}}) = \left(\underline{\underline{A}}\underline{\underline{A}}^{\mathrm{T}} - 2\,\mathrm{Sp}(\underline{\underline{A}})\underline{\underline{I}}_3 \right) \underline{p}, \tag{3.25}$$

was zu

$$2\left(\mathrm{Sp}(\underline{\underline{R}}\underline{\underline{R}}^{\mathrm{T}})\underline{\underline{I}}_3 - \underline{\underline{R}}\underline{\underline{R}}^{\mathrm{T}} \right) \underline{p}^* + 4\,\mathrm{vect}(\underline{\underline{T}}\underline{\underline{R}}^{\mathrm{T}}) = \underline{0} \tag{3.26}$$

führt. Schließlich wird die Gleichung nach $\underline{p}^*$ aufgelöst und es folgt

$$\underline{p}^* = 2\left(\underline{\underline{R}}\underline{\underline{R}}^{\mathrm{T}} - \mathrm{Sp}(\underline{\underline{R}}\underline{\underline{R}}^{\mathrm{T}})\underline{\underline{I}}_3 \right)^{-1} \mathrm{vect}(\underline{\underline{T}}\underline{\underline{R}}^{\mathrm{T}}). \tag{3.27}$$

Es ergibt sich also eine geschlossene Darstellung für den optimalen End-Effektor-Punkt bezüglich Änderungen der geometrischen Parameter. Dieser Punkt hängt offensichtlich von der Position des Manipulators ab und ist daher immateriell. Er kann jedoch verwendet werden, um unterschiedliche Varianten des Manipulators zu vergleichen, die Bewegungen um einen Referenzpunkt erzeugen sollen. In diesem Zusammenhang muss auch festgestellt werden, dass die Konditionszahl willkürlich beeinflusst werden kann, indem das Übersetzungsverhältnis der Antriebe beispielsweise durch Getriebe verändert wird. Insofern stellt

das Konzept der Manipulierbarkeit nur eine Eigenschaft der reinen Mechanik des Roboters dar, während die Eigenschaften des gesamten Roboters durch weitere technische Maßnahmen verbessert werden können.

Aufbauend auf dem oben vorgestellten Verfahren zur vollständigen Linearisierung von MKS, wird die JACOBI-Matrix $\underline{\underline{J}}_g$ in den folgenden Abschnitten verwendet, um den Roboter bezüglich seiner Genauigkeit und Steifigkeit zu untersuchen. Weiterhin wird aufgezeigt, welche Rolle $\underline{\underline{J}}_g$ bei der Kalibrierung spielt.

3.4 Fehler- und Sensitivitätsanalyse

Die offensichtliche Anwendung der linearisierten Kinematik besteht darin, die Sensitivität der Maschine bezüglich geometrischer Fehler zu untersuchen. Reale Maschinen weichen stets von der idealen Geometrie ab, die durch die nominellen Parameter beschrieben wird. Die Ursache dafür liegt in Fertigungstoleranzen und Fehlern bei der Produktion der Bauteile sowie Fehlern während der Montage. Eine weitere Quelle von Fehlern ist das Spiel in Gelenken und an Verbindungen zwischen Bauteilen. Schließlich können thermische und elastische Effekte einen Einfluss auf die Bewegung der Plattform haben. In der Praxis kann jedoch davon ausgegangen werden, dass verglichen mit den Abmessungen der Maschine, alle diese möglichen Fehler klein sind. Daher ist eine Approximation durch ein linearisiertes Modell gerechtfertigt. Mithilfe des in Abschnitt 3.3 vorgestellten Verfahrens zur vollständigen Linearisierung der Kinematik kann der Einfluss beliebiger geometrischer Fehler auf die Genauigkeit der Plattform abgebildet werden. Dazu werden die virtuellen Verrückungen $\delta\underline{t}_i$ mit den geometrischen Fehlern $\Delta\underline{g}$ identifiziert, so dass unmittelbar die Berechnungsvorschrift für den Fehler der Plattform $\Delta\underline{y}$ zu

$$\Delta\underline{y} = \underline{\underline{J}}_g \Delta\underline{g} \tag{3.28}$$

folgt. Neben der Bestimmung des Fehlers der Pose für bekannte Fehler $\Delta\underline{g}$ der Geometrie, soll untersucht werden, wie sich Fehler der Geometrie bei einem Manipular grundsätzlich auswirken. Für verschiedene Roboter des gleichen Typs variieren die tatsächlichen Werte der Längen und Winkel entsprechend der für die Fertigung vorgegebenen Toleranzen. Dabei wird davon ausgegangen, dass für die geometrischen Parameter die Fertigungstoleranzen bekannt sind und dass die tatsächlich auftretenden Fehler normalverteilt sind. Weiterhin wird angenommen, dass die Fehler klein und voneinander unabhängig sind. Der Betrag des Gesamtfehlers ist daher die Summe der Einzelfehler, und es wird eine statistische Fehlerfortpflanzung unterstellt. Bei der Addition zweier unabhängiger normalverteilter Zufallsvariablen gilt für die Standardabweichung σ der Summe

$$\sigma = \sqrt{\sigma_1^2 + \sigma_2^2}, \tag{3.29}$$

wobei σ_1 und σ_2 die Standardabweichungen der Summanden sind. Weiterhin gilt für die Länge e eines Ortsvektors $\underline{e} = [e_1, e_2, e_3]^{\mathrm{T}} \in \mathbb{R}^3$

$$e = |\underline{e}| = \sqrt{e_1^2 + e_2^2 + e_3^2}. \tag{3.30}$$

Bei der Betrachtung der Spalten der JACOBI-Matrix $\underline{\underline{J}}_g$ werden translatorische und rotatorische Anteile miteinander vermischt werden, was durch die Einführung von Metrikkoeffizieten ϱ_j berücksichtigt werden muss. Diese Metrikkoeffizienten können als virtuelle Hebel

betrachtet werden, welche die Rotationen auf Translationen abbilden. Als Abschätzung wird angenommen, dass die Standardabweichung σ_i eines Parameters g_i der bei der Fertigung verwendeten Toleranz entspricht.[5] Unter Berücksichtigung der Fehlerfortpflanzung lässt sich die Standardabweichung σ_p des EEF zu

$$\sigma_p = \sqrt{\sum_{i=1}^{6}\sum_{j=1}^{N}\left(\varrho_j[\underline{\underline{J}}_g]_{ij}\ \sigma_i\right)^2} \tag{3.31}$$

berechnen. Wird weiterhin unterstellt, dass die Toleranzen σ_i alle den gleichen Wert σ haben, vereinfacht sich die Beziehung zu

$$\sigma_p = \sigma \underbrace{\sqrt{\sum_{i=1}^{6}\sum_{j=1}^{N}\left(\varrho_j[\underline{\underline{J}}_g]_{ij}\right)^2}}_{\overline{\sigma}} \tag{3.32}$$

und damit folgt die Definition der *Gesamtfehlerverstärkung*

$$\overline{\sigma} = \frac{\sigma_p}{\sigma} = \sqrt{\sum_{i=1}^{6}\sum_{j=1}^{N}\left(\varrho_j[\underline{\underline{J}}_g]_{ij}\right)^2}, \tag{3.33}$$

welche die Empfindlichkeit des gesamten Manipulators bezüglich geometrischer Fehler beschreibt. Diese Kenngröße ähnelt dem statistischen Ansatz, der von Wittwer et al. [136] beschrieben wird. Ist für einen Prozess die erforderliche Genauigkeit σ_p der Maschine bekannt, lässt sich die Gesamtfehlerverstärkung $\overline{\sigma}$ dazu verwenden, um die erforderliche mittlere Toleranz σ für die Geometrieparameter abzuschätzen. Allgemeiner ausgedrückt, erreicht eine Maschine am EEF eine um den Faktor $\overline{\sigma}$ geringere Genauigkeit als in ihren Komponenten. Dabei spielt es keine Rolle, ob die nominellen Werte für die Geometrieparameter $\underline{g}$ durch Tolerierung, Vermessung oder Kalibrierung beeinflusst werden.

In Anlehnung an die Argumentation in Abschnitt 3.3.2 stellt sich die Frage, welcher Punkt auf der Plattform die Gesamtfehlerverstärkung $\overline{\sigma}$ minimiert. An dieser Stelle wird wieder die Darstellung mit Blockmatrizen nach Gleichung (3.14) aufgegriffen, um nach dem Vektor $\underline{p}^*$ zu suchen, für den $\overline{\sigma}(\underline{\underline{J}}_g)$ minimal ist. Das entsprechende Optimierungsproblem lautet

$$\overline{\sigma}(\underline{\underline{J}}_g, \underline{p}^*) = \min_{\underline{p}} \sum_{i=1}^{6}\sum_{j=1}^{N}[\underline{\underline{U}}(\underline{p})\underline{\underline{J}}_g]_{ij}^2, \tag{3.34}$$

wobei $\underline{\underline{U}}(\underline{p})$ die Übergangsmatrix nach Gleichung (3.15) ist. Die notwendige Bedingung für die Lösung $\underline{p}^*$ ergibt sich dann zu

$$\left.\frac{\partial\overline{\sigma}(\underline{\underline{J}}_g, \underline{p})}{\partial \underline{p}}\right|_{\underline{p}=\underline{p}^*} = \frac{\partial}{\partial \underline{p}}\sum_{i=1}^{6}\sum_{j=1}^{N}\left[\begin{array}{c}\widetilde{\underline{p}}^*\underline{\underline{R}} + \underline{\underline{T}} \\ \underline{\underline{R}}\end{array}\right]_{ij}^2\Bigg|_{\underline{p}=\underline{p}^*} = \underline{0}. \tag{3.35}$$

[5] In der Praxis ist je nach Anwendung eine Toleranz $t = \pm 2\sigma$ oder $t = \pm 3\sigma$ üblich; dieser Faktor ist jedoch für die nachfolgend beschriebenen Überlegungen unerheblich.

Da die Matrizen $\underline{\underline{R}}, \underline{\underline{T}}$ nicht von $\underline{p}$ abhängen, muss nur das obere Teilsystem betrachtet werden. Mit den Abkürzungen $\underline{r}_j, \underline{t}_j$ für die j-te Spalte der Matrix $\underline{\underline{R}}$ bzw. $\underline{\underline{T}}$ folgt

$$\frac{\partial}{\partial \underline{p}} \sum_{i=1}^{3} \sum_{j=1}^{N} (\underline{p} \times \underline{r}_j + \underline{t}_j)_i^2 \Big|_{\underline{p}=\underline{p}^*} = \frac{\partial}{\partial \underline{p}} \sum_{j=1}^{N} ||\underline{p} \times \underline{r}_j + \underline{t}_j||_2^2 \Big|_{\underline{p}=\underline{p}^*} = \underline{0}, \tag{3.36}$$

wobei mit $||\cdot||_2$ die euklidische Norm bezeichnet wird und die Indizes $i = 1, 2, 3$ für die drei Komponenten der kartesischen Vektoren stehen. Ausmultiplizieren der Komponenten liefert

$$\begin{aligned} \sum_{j=1}^{N} \frac{\partial}{\partial \underline{p}} ((p_y r_{3j} - p_z r_{2j} + t_{1j})^2 &+ (p_z r_{1j} - p_x r_{3j} + t_{2j})^2 \\ &+ (p_x r_{2j} - p_y r_{1j} + t_{3j}))^2 \Big|_{\underline{p}=\underline{p}^*} = \underline{0}. \end{aligned} \tag{3.37}$$

Damit ergibt sich die Bedingung für den optimalen End-Effektor-Punkt zu

$$\sum_{j=1}^{N} \begin{bmatrix} -2\left(p_z^* r_{1j} - p_x^* r_{3j} + t_{2j}\right) r_{3j} + 2\left(p_x^* r_{2j} - p_y^* r_{1j} + t_{3j}\right) r_{2j} \\ 2\left(p_y^* r_{3j} - p_z^* r_{2j} + t_{1j}\right) r_{3j} - 2\left(p_x^* r_{2j} - p_y^* r_{1j} + t_{3j}\right) r_{1j} \\ -2\left(p_y^* r_{3j} - p_z^* r_2 + t_{1j}\right) r_{2j} + 2\left(p_z^* r_{1j} - p_x^* r_{3j} + t_{2j}\right) r_{1j} \end{bmatrix} = \underline{0}. \tag{3.38}$$

Umschreiben der Gleichungen in ein lineares Gleichungssystem in den Koeffizienten von $\underline{p}^*$ liefert

$$\sum_{j=1}^{N} \begin{bmatrix} r_{3j}^2 + r_{2j}^2 & -r_{2j} r_{1j} & -r_{3j} r_{1j} \\ -r_{2j} r_{1j} & r_{3j}^2 + r_{1j}^2 & -r_{3j} r_{2j} \\ -r_{3j} r_{1j} & -r_{3j} r_{2j} & r_{2j}^2 + r_{1j}^2 \end{bmatrix} \underline{p}^* = \sum_{j=1}^{N} \begin{bmatrix} r_{3j} t_{2j} - r_{2j} t_{3j} \\ -r_{3j} t_{1j} + r_{1j} t_{3j} \\ r_{2j} t_{1j} - r_{1j} t_{2j} \end{bmatrix}, \tag{3.39}$$

welches zu der kompakteren Vektorform

$$\left[\sum_{j=1}^{N} (\underline{r}_j^{\mathrm{T}} \underline{r}_j \underline{\underline{I}}_3 - \underline{r}_j \underline{r}_j^{\mathrm{T}}) \right] \underline{p}^* = \sum_{j=1}^{N} (\underline{t}_j \times \underline{r}_j) \tag{3.40}$$

führt. Für den Punkt $\underline{p}^*$ mit der geringsten Empfindlichkeit bezüglich Fehlern in den geometrischen Parametern ergibt sich damit die Lösung

$$\underline{p}^* = \left[\sum_{j=1}^{N} \underline{r}_j^{\mathrm{T}} \underline{r}_j \underline{\underline{I}}_3 - \sum_{j=1}^{N} \underline{r}_j \underline{r}_j^{\mathrm{T}} \right]^{-1} \sum_{j=1}^{N} (\underline{t}_j \times \underline{r}_j). \tag{3.41}$$

Dies lässt sich in Matrixnotation zu

$$\underline{p}^* = \left[\mathrm{Sp}(\underline{\underline{R}} \underline{\underline{R}}^{\mathrm{T}}) \underline{\underline{I}}_3 - \underline{\underline{R}} \underline{\underline{R}}^{\mathrm{T}} \right]^{-1} 2 \,\mathrm{vect}(\underline{\underline{T}} \underline{\underline{R}}^{\mathrm{T}}). \tag{3.42}$$

zusammenfassen. Der optimale End-Effektor-Punkt bezüglich der Gesamtfehlerverstärkung ist also mit dem optimalen End-Effektor-Punkt bezüglich der Minimierung der Konditionszahl der Matrix $\underline{\underline{J}}_g$ identisch.

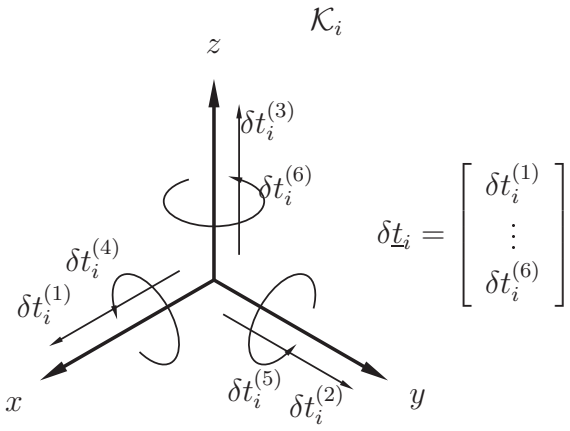


Abbildung 3.7: Zerlegung des Geschwindigkeitswinders $\underline{t}_i$ am Koordinatensystem $\mathcal{K}_i$

3.5 Steifigkeit

Die Linearisierung eines Manipulators bezüglich seiner geometrischen Parameter stellt eine lineare Abbildung zwischen infinitesimalen Änderungen der Geometrie und infinitesimalen Verrückungen des EEF dar. Die JACOBI-Matrix $\underline{\underline{J}}_g$ kann daher auch dazu verwendet werden, die Steifigkeitsmatrix $\underline{\underline{K}}$ des Roboters für eine Pose $\underline{q}$ zu bestimmen. Wie in Abschnitt 2.1.1 beschrieben wurde, gilt für ideale Übertragungselemente

$$\underline{w}_g^{\mathrm{T}} \delta \underline{t}_g = \underline{w}_{\mathrm{EEF}}^{\mathrm{T}} \delta \underline{t}_{\mathrm{EEF}}, \tag{3.43}$$

wobei in $\delta \underline{t}_g = [\delta t_1^{(1)}, \delta t_1^{(2)}, \ldots, \delta t_p^{(6)}]^{\mathrm{T}}$ alle virtuellen Verrückungen der geometrischen Parameter zusammengefasst sind (Abb. 3.7) und $\underline{w}_g = [w_1^{(1)}, w_1^{(2)}, \ldots, w_p^{(6)}]^{\mathrm{T}}$ die entsprechenden Schnittkräfte darstellt. Für die Betrachtung der Steifigkeit wird jedes Übertragungselement des Manipulators als lineare Feder modelliert. Damit ergibt sich

$$\underline{\underline{K}}_g^{-1} \underline{w}_g = \delta \underline{t}_g \quad \text{mit} \quad \underline{\underline{K}}_g^{-1} = \operatorname{diag} \left\{ \frac{1}{c_1^{(1)}}, \ldots, \frac{1}{c_p^{(6)}} \right\} \tag{3.44}$$

mit der lokalen Steifigkeitsmatrix $\underline{\underline{K}}_g$, die sich aus den Federkonstanten c_i für jedes einzelne Element zusammensetzt. Für die Steifigkeit am EEF gilt

$$\underline{\underline{K}}_{\mathrm{EEF}}^{-1} \underline{w}_{\mathrm{EEF}} = \delta \underline{t}_{\mathrm{EEF}}, \tag{3.45}$$

wobei die Matrix $\underline{\underline{K}}_{\mathrm{EEF}}$ die gesuchte Steifigkeitsmatrix für den EEF ist. Dies ist eine Vereinfachung der Eigenschaften mechanischer Bauteile, die jedoch für viele technische Anwendungen, wie beispielsweise schlanke Balken und Gelenke, ausreicht. Eine Verallgemeinerung zu einer voll besetzten Steifigkeitsmatrix kann durch eine Analyse mit Finiten Elementen erreicht werden. Gleichung (3.44) und (3.45) werden nun in Gleichung (3.43) eingesetzt und unter Berücksichtigung der Kraftübertragung $\underline{w}_g = \underline{\underline{J}}_g^{\mathrm{T}} \underline{w}_{\mathrm{EEF}}$ nach Gleichung (3.8) ergibt sich

$$\underline{w}_{\mathrm{EEF}}^{\mathrm{T}} \underline{\underline{J}}_g \underline{\underline{K}}_g^{-1} \underline{\underline{J}}_g^{\mathrm{T}} \underline{w}_{\mathrm{EEF}} = \underline{w}_{\mathrm{EEF}}^{\mathrm{T}} \underline{\underline{K}}_{\mathrm{EEF}}^{-1} \underline{w}_{\mathrm{EEF}}. \tag{3.46}$$

Da diese Beziehung für jedes beliebige $\underline{w}_{\mathrm{EEF}}$ gelten muss, folgt

$$\underline{\underline{K}}_{\mathrm{EEF}}^{-1} = \underline{\underline{J}}_g \underline{\underline{K}}_g^{-1} \underline{\underline{J}}_g^{\mathrm{T}}. \tag{3.47}$$

Die JACOBI-Matrix $\underline{\underline{J}}_g$ kann also verwendet werden, um die elementweise bekannte Steifigkeitsmatrix $\underline{\underline{K}}_g$ auf Weltkoordinaten zu transformieren, so dass sich die gesuchte Steifigkeitsmatrix $\underline{\underline{K}}_{\mathrm{EEF}}$ am End-Effektor ergibt.

3.6 Kalibrierung

Verglichen mit gewöhnlichen Industrierobotern werden für Werkzeugmaschinen hohe Anforderungen bezüglich der Genauigkeit gestellt. Neben einer hochwertigen Fertigung werden diese Maschinen daher nach der Montage kalibriert. Dies kann entweder mechanisch, z. B. durch Stellschrauben an den linearen Führungen der Maschine, geschehen oder durch eine Kompensation in der NC-Steuerung. Bei seriellen Robotern kann die Kalibrierung für jeden Antrieb unabhängig durchgeführt werden, denn die Bewegungen sind entkoppelt und bestehen entweder aus einem Kreisbogen oder einer Geraden. Abweichungen von der vorgegebenen Geometrie können daher messtechnisch leicht festgestellt und kompensiert werden. Bei PKM sind die Bewegungen aller Bauteile miteinander gekoppelt, so dass die Bewegung eines einzelnen Antriebs an der Plattform eine komplexe Trajektorie im Raum erzeugt, die Translationen und Rotationen miteinander vermischt. Es ergeben sich bei der Kalibrierung daher zwei schwerwiegende Probleme. Verglichen mit dem Vermessen von Linearität und Rundheit ist die gleichzeitige Vermessung von Position und Orientierung im Raum gerätetechnisch aufwendig und eher ungenau. Das zweite Problem besteht darin, dass die zu identifizierenden Parameter in der Transformation der NC-Steuerung ebenfalls miteinander gekoppelt sind. Daher ist es nur möglich, alle Parameter gleichzeitig zu ermitteln.

Die Linearisierung der Kinematik bezüglich der geometrischen Parameter spielt daher eine wichtige Rolle bei der Kalibrierung von Robotern. Die Untersuchung von verschiedenen Verfahren zur Kalibrierung würde den Rahmen dieser Arbeit sprengen, aber an dieser Stelle soll skizziert werden, welche Rolle die linearisierte Form der Kinematik bei der Kalibrierung spielt. Hier wird davon ausgegangen, dass für die Kalibrierung N Paare von Messwerten, bestehend aus den Weltkoordinaten des End-Effektors $\underline{y}_i$ und den gemessenen verallgemeinerten Koordinaten $\underline{q}_i$, aufgenommen wurden. Gesucht sind die geometrischen Parameter $\underline{g}^*$, die den Fehler zwischen dem mathematischen Modell $\underline{y} = \underline{\varphi}^{\mathrm{DK}}(\underline{g}, \underline{q})$ und den Messwerten $\{\underline{y}_i, \underline{q}_i\}$ minimieren. Dazu wird in der Praxis die *Methode der kleinsten Fehlerquadrate* von GAUSS verwendet, die auf das nichtlineare Optimierungsproblem

$$\Xi = \sum_{i=1}^{N} \left(\underline{\varphi}^{\mathrm{DK}}(\underline{q}_i, \underline{g}) - \underline{y}_i \right)^2 = \min! \tag{3.48}$$

führt. Eine notwendige Bedingung erster Ordnung für ein Minimum des gesamten Fehlers Ξ ist

$$\sum_{i=1}^{N} \left(\frac{\partial \underline{\varphi}^{\mathrm{DK}}(\underline{q}_i, \underline{g})}{\partial \underline{g}} \right)^{\mathrm{T}} \left(\underline{\varphi}^{\mathrm{DK}}(\underline{q}_i, \underline{g}) - \underline{y}_i \right) \Bigg|_{\underline{g}=\underline{g}^*} = \underline{0}. \tag{3.49}$$

Daraus folgt

$$\sum_{i=1}^{N} \underline{\underline{J}}_g^{\mathrm{T}}(\underline{q}_i) \underline{\varphi}^{\mathrm{DK}}(\underline{q}_i, \underline{g}^*) = \sum_{i=1}^{N} \underline{\underline{J}}_g^{\mathrm{T}}(\underline{q}_i) \underline{y}_i \quad . \tag{3.50}$$

Diese und ähnliche Methoden werden für die Kalibrierung von PKM verwendet, bei denen die Matrix $\underline{\underline{J}}_g$ bei den meisten Ansätzen durch numerische Differentiation bestimmt wird

[2, 27]. Aus der Konstruktion der Maschine sind die nominellen Parameter bekannt, die als erster Schätzwert $\underline{g}_0$ für die Geometrieparameter verwendet werden. Die tatsächlichen Parameter werden durch $\underline{g} = \underline{g}_0 + \Delta\underline{g}$ dargestellt, wobei $\Delta\underline{g}$ eine kleine Korrektur gegenüber den nominellen Parametern $\underline{g}$ ist. Daraus ergibt sich ein lineares Gleichungssystem für die Korrekturterme $\Delta\underline{g}$

$$\sum_{i=1}^{N} \underline{\underline{J}}_g^{\mathrm{T}}(\underline{q}_i)\ \underline{\underline{J}}_g(\underline{q}_i)\Delta\underline{g} = \sum_{i=1}^{N} \underline{\underline{J}}_g^{\mathrm{T}}(\underline{q}_i)\ (\underline{y}_i - \underline{\varphi}_{0i}^{\mathrm{DK}}), \tag{3.51}$$

wobei $\underline{\varphi}_{0i}^{\mathrm{DK}} = \underline{\varphi}^{\mathrm{DK}}(\underline{y}_i, \underline{g}_0)$ die Auswertung der direkten Kinematik für den Schätzwert $\underline{g}_0$ der geometrischen Parameter ist. Die gesuchten Größen $\Delta\underline{g}$ können also mit der JACOBI-Matrix $\underline{\underline{J}}_g$ berechnet werden. In der Praxis werden für die Lösung von Gleichung (3.48) meist NEWTON-Verfahren oder LEVENBERG-MARQUARDT-Methoden verwendet. Weiterhin wurde vorgeschlagen, spezielle Messstellen für die Aufnahme der Messwerte $\underline{y}_i, \underline{q}_i$ zu verwenden. Insbesondere im Hinblick auf die Optimierung der Messpositionen bildet die exakte und schnelle Bestimmung von $\underline{\underline{J}}_g$ eine wichtige Grundlage [8].

3.7 Ergebnisse

3.7.1 Generische Kinematik

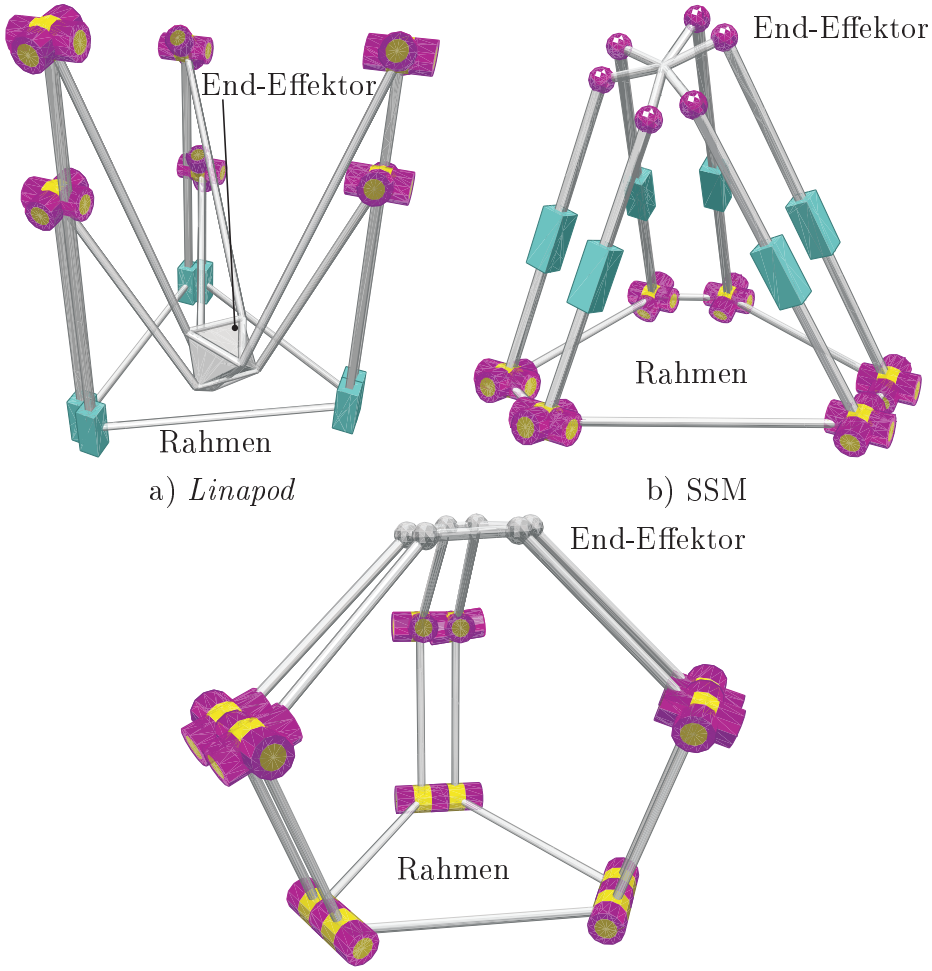
Die in Abschnitt 3.2 vorgestellte Vorgehensweise soll im folgenden Beispiel veranschaulicht werden. Dazu wird das Aufstellen des kinetostatischen Modells anhand des Algorithmus `generische inverse Kinematik` für den im nächsten Abschnitt betrachteten *Linapod* betrachtet.

Zunächst wird davon ausgegangen, dass alle geometrischen Parameter entsprechend Tabelle 2.4 (Seite 29) bekannt sind. Das konkrete Modul `Linapod Rahmen` kapselt für die Maschine *Linapod* die in Tabelle 2.3 (Seite 29) angegebenen Formeln. Damit lassen sich die im Schritt 1 des Algorithmus gesuchten Koordinatensysteme $\mathcal{K}_{C_i}$ zu

$$\begin{aligned}
\underline{c}_1 &= \underline{\underline{T}}(\underline{e}_z, 0°)(r_b\underline{e}_x + \Delta a\underline{e}_y) && (3.52)\\
\underline{c}_2 &= \underline{\underline{T}}(\underline{e}_z, 120°)(r_b\underline{e}_x + \Delta a\underline{e}_y) && (3.53)\\
\underline{c}_3 &= \underline{\underline{T}}(\underline{e}_z, 240°)(r_b\underline{e}_x + \Delta a\underline{e}_y) && (3.54)\\
\underline{c}_4 &= \underline{\underline{T}}(\underline{e}_z, 0°)(r_b\underline{e}_x - \Delta a\underline{e}_y) && (3.55)\\
\underline{c}_5 &= \underline{\underline{T}}(\underline{e}_z, 120°)(r_b\underline{e}_x - \Delta a\underline{e}_y) && (3.56)\\
\underline{c}_6 &= \underline{\underline{T}}(\underline{e}_z, 240°)(r_b\underline{e}_x - \Delta a\underline{e}_y) && (3.57)
\end{aligned}$$

berechnen, wobei hier auf die Komponenten der Vektoren und Matrizen in Weltkoordinaten übergegangen wird. Im Schritt 2 des Algorithmus wird das konkrete Modul `Linapod Plattform` verwendet, um die Koordinatensysteme aller plattformseitigen Anlenkpunkte $\mathcal{K}_{B_i}$ bezüglich des End-Effektors mit

$$\begin{aligned}
\underline{b}_1 &= \underline{\underline{T}}(\underline{e}_z, \Delta\alpha)r_l\underline{e}_x + \Delta t\underline{e}_z && (3.58)\\
\underline{b}_2 &= \underline{\underline{T}}(\underline{e}_z, 120° + \Delta\alpha)r_l\underline{e}_x + \Delta t\underline{e}_z && (3.59)\\
\underline{b}_3 &= \underline{\underline{T}}(\underline{e}_z, 240° + \Delta\alpha)r_l\underline{e}_x + \Delta t\underline{e}_z && (3.60)
\end{aligned}$$



a) *Linapod* b) SSM

c) Delta Roboter

Abbildung 3.8: Modulare Modelle für unterschiedliche PKM

$$\underline{b}_4 = \underline{\underline{T}}(\underline{e}_z, -\Delta\alpha) r_u \underline{e}_x + (\Delta t + \Delta h)\underline{e}_z \quad (3.61)$$

$$\underline{b}_5 = \underline{\underline{T}}(\underline{e}_z, 120° - \Delta\alpha) r_u \underline{e}_x + (\Delta t + \Delta h)\underline{e}_z \quad (3.62)$$

$$\underline{b}_6 = \underline{\underline{T}}(\underline{e}_z, 240° - \Delta\alpha) r_u \underline{e}_x + (\Delta t + \Delta h)\underline{e}_z \quad (3.63)$$

zu bestimmen. Nachdem für jedes Bein die jeweiligen Anlenkpunkte ermittelt wurden, lässt sich im Schritt 3 die geschlossene Lösung (2.23) für das `PUS-Modul` verwenden, um die gesuchten verallgemeinerten Koordinaten

$$q_i = \underline{e}_z \cdot \underline{d}_i \pm \sqrt{(\underline{e}_z \cdot \underline{d}_i)^2 - \underline{d}_i^2 + l_i^2} \quad \text{mit} \quad \underline{d}_i = \underline{\underline{R}}_{\text{Kardan}} \underline{b}_i + \underline{r} - \underline{c}_i \quad \text{für} \quad i = 1, \ldots, 6 \quad (3.64)$$

zu berechnen. Dabei treten die Weltkoordinaten $\underline{y}$ als Ortsvektor $\underline{r}$ und Rotationsmatrix $\underline{\underline{R}}_{\text{Kardan}}$ in Erscheinung. In der Implementierung werden die Berechnungen von den entsprechenden kinetostatischen Übertragungselementen vorgenommen, die den jeweilige konkreten Programmmodulen für Plattform, Rahmen und Bein entstammen.

Tabelle 3.2: Laufzeit des modularen kinetostatischen Modells (Windows, MS Visual C++ 6.0, MOBILE, Pentium IV, 3.2 GHz)

Maschine	Kinematik	Übertragungsfunktion	Auswertungen/Sekunde
Linapod	Invers	Position	16842
		+ Geschwindigkeit	11791
		+ Beschleunigung	8684
		+ Kraft	6784
		nur Geschwindigkeit	39315
		nur Beschleunigung	32955
		nur Kraft	31006
Linapod	Direkt	Position	8132
		+ Geschwindigkeit	6730
		+ Beschleunigung	5629
		+ Kraft	4454
SSM	Invers	Position	17258
		+ Geschwindigkeit	12751
		+ Beschleunigung	9927
		+ Kraft	7284
Delta Roboter	Invers	Position	13821
		+ Geschwindigkeit	10263
		+ Beschleunigung	8273
		+ Kraft	6131

3.7.2 Modulares Modell

Mithilfe des in Abschnitt 3.2 vorgestellten Baukastens lassen sich leicht die vollständigen kinetostatischen Modelle verschiedener PKM erzeugen und untersuchen. Hier werden stellvertretend für die zahlreichen möglichen Kombinationen beispielhaft die Maschinen *Linapod* (Abb. 3.8a), ein SSM (Abb. 3.8b) und der Delta Roboter (Abb. 3.8c) vorgestellt.

In Tabelle 3.2 sind die Rechenzeiten für die Auswertung der Kinematik verschiedener Maschinen angegeben, die anhand des modularen Baukastens zusammengestellt wurden. Es wurde jeweils die Anzahl der Auswertungen der kinetostatischen Übertragungsfunktionen pro Sekunde bestimmt. Ein Vergleich der Maschinen zeigt, dass die jeweilige Kinematik über eine ähnliche Komplexität verfügt.

Das modulare Modell wurde beispielhaft zur Berechnung des kinematisch erreichbaren Arbeitsraums für die PKM *Linapod* verwendet, indem die Positionsübertragungsfunktion an diskreten Punkten berechnet wurde. Diese Punkte wurden auf einem Gitter vorgegeben, um auf diesem Wege eine grobe Abschätzung des Arbeitsraums in der xy-Ebene mit $z = 0$ zu erhalten (Abb. 3.9). Die Rechenzeit für die Bestimmung des Arbeitsraums beträgt 3.6 s, wobei für das Gitter ein Abstand von 0.01 m zur Diskretisierung verwendet wurde. Diese Methode hat jedoch den wesentlichen Nachteil, dass keinerlei Informationen darüber vorliegen, ob sich der Arbeitsraum zwischen den Gitterpunkten fortsetzt. Weiterhin kann auf diese Weise auch nicht ermittelt werden, wie sich der Arbeitsraum in

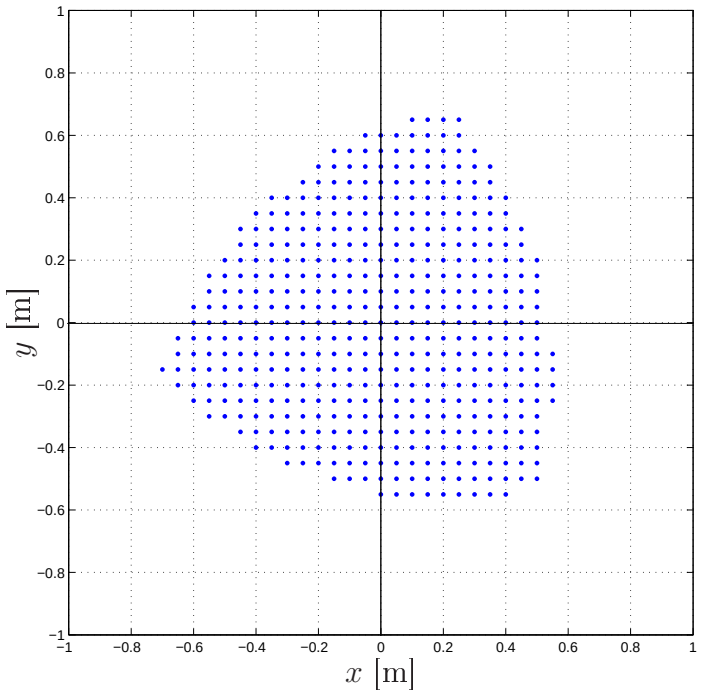


Abbildung 3.9: Kinematisch erreichbarer Arbeitsraum des *Linapod* durch Diskretisierung mit modularem kinetostatischem Modell.

Abhängigkeit von den geometrischen Parametern verändert. Eine Methode zur verifizierten Berechnung des Arbeitsraums wird daher in Kapitel 4 präsentiert und in Kapitel 5 zur Maßsynthese erweitert.

3.7.3 Sensitivitätsanalyse

In den folgenden Abschnitten wird das Verfahren zur Linearisierung und Sensitivitätsanalyse auf verschiedene Manipulatoren angewendet. Zur Veranschaulichung wird zunächst ein ebener serieller Roboter betrachtet. Im Anschluss daran wird die vollständige Linearisierung für das kinetostatische Modell der räumlichen PKM *Linapod* berechnet.

Sensitivitätsanalyse für serielle Roboter

Zunächst wird ein ebener serieller Roboter (Abb. 3.10) betrachtet, um das Verfahren zur Linearisierung anhand eines analytisch nachvollziehbaren Beispiels zu illustrieren. Der betrachtete Roboter besteht aus zwei Drehgelenken mit den Winkeln $\underline{\beta} = [\beta_1, \beta_2]^{\mathrm{T}}$ und zwei Armen mit den Längen l_1, l_2. Die Berechnung der direkten Kinematik ist trivial und es ergibt sich

$$\underline{\varphi}^{\mathrm{DK}}(\underline{\beta}) = \begin{bmatrix} l_1 \cos\beta_1 + l_2 \cos\beta_{12} \\ l_1 \sin\beta_1 + l_2 \sin\beta_{12} \\ \beta_{12} \end{bmatrix}, \tag{3.65}$$

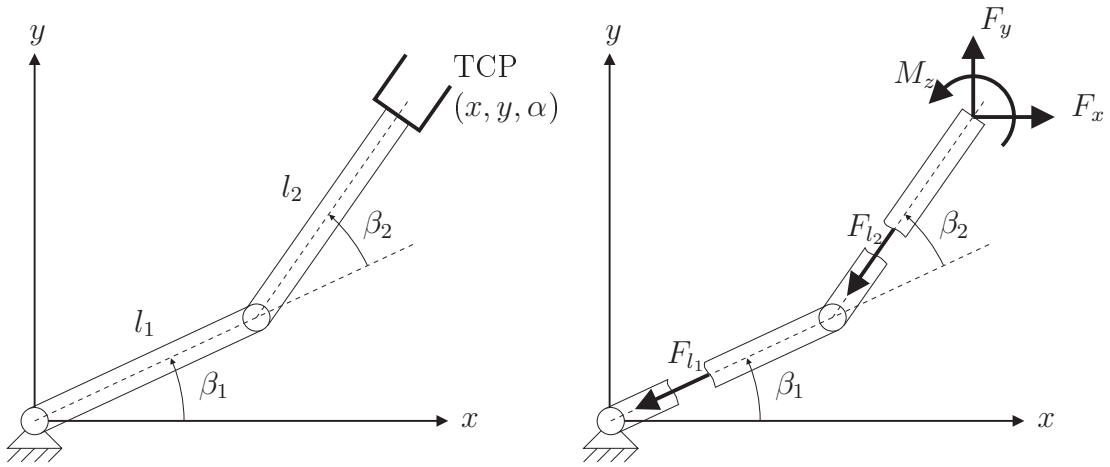


Abbildung 3.10: Serieller ebener Roboter mit zwei Freiheitsgraden a) Kinematik b) eingeprägte Kräfte und Schnittkräfte

wobei die Abkürzung $\beta_{12} = \beta_1 + \beta_2$ verwendet wird. Die Linearisierung kann durch Ableitung von $\underline{\varphi}^{\mathrm{DK}}(\underline{\beta})$ nach den geometrischen Parametern $\underline{l} = [l_1, l_2]^{\mathrm{T}}$ bestimmt werden und für dieses einfache Beispiel lässt sich die analytische Lösung für die gesuchte JACOBI-Matrix angeben mit

$$\delta\underline{e} = \frac{\partial\underline{\varphi}}{\partial\underline{l}}\delta\underline{l} = \underbrace{\begin{bmatrix} \cos\beta_1 & \cos\beta_{12} \\ \sin\beta_1 & \sin\beta_{12} \\ 0 & 0 \end{bmatrix}}_{\underline{\underline{J}}_g} \delta\underline{l}. \tag{3.66}$$

Nun wird der kraftgestützte Algorithmus (Abschnitt 3.3) verwendet, um die Sensitivitätsanalyse durchzuführen. Für eine gegebene Konfiguration des Roboters werden die Einheitskräfte und -momente F_x, F_y, M_z dem End-Effektor eingeprägt und die dazu gehörigen Schnittkräfte F_{l_1}, F_{l_2} berechnet. Dann folgt

$$F_{l_1} = \cos\beta_1 F_x + \sin\beta_1 F_y, \tag{3.67}$$
$$F_{l_2} = \cos\beta_{12} F_x + \sin\beta_{12} F_y \tag{3.68}$$

und eine Faktorisierung in Vektorschreibweise liefert

$$\begin{bmatrix} F_{l_1} \\ F_{l_2} \end{bmatrix} = \underbrace{\begin{bmatrix} \cos\beta_1 & \sin\beta_1 & 0 \\ \cos\beta_{12} & \sin\beta_{12} & 0 \end{bmatrix}}_{\underline{\underline{J}}_g^{\mathrm{T}}} \begin{bmatrix} F_x \\ F_y \\ M_z \end{bmatrix}. \tag{3.69}$$

Offensichtlich stimmen die Matrizen $\underline{\underline{J}}_g$ bei beiden Berechnungsverfahren nach Gleichung (3.69) und Gleichung (3.66) überein. Bemerkenswerterweise werden für die Kraftberechnung nur die Richtungen der Schnittkräfte F_{l_1}, F_{l_2} benötigt, die sich auch aus der numerischen Lösung der direkten Kinematik ermitteln lassen.

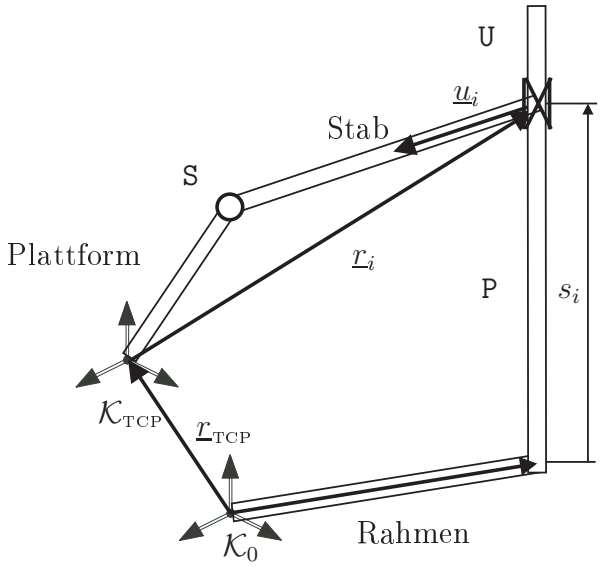


Abbildung 3.11: Geometrie des *Linapod* für das betrachtete PUS-Bein

Sensitivitätsanalyse für die PKM *Linapod*

Im Folgenden wird eine Sensitivitätsanalyse für die PKM *Linapod* durchgeführt, indem der Algorithmus aus Abschnitt 3.3 angewendet wird. Exemplarisch wird zunächst der Einfluss von Änderungen in den unteren und oberen Beinlängen l_l, l_u untersucht. Das Kräftegleichgewicht in der Mitte des Arbeitsraums ($\underline{r}_p = \underline{0}, \underline{\underline{R}}_p = \underline{\underline{I}}$) wird dazu anhand eines einfachen Freischnitts ermittelt. Dabei werden sukzessive die Einheitskräfte $\check{\underline{f}}_x, \check{\underline{f}}_y, \check{\underline{f}}_z$ sowie die Einheitsmomente $\check{\underline{m}}_x, \check{\underline{m}}_y, \check{\underline{m}}_z$ auf die Plattform eingeprägt. Damit ergibt sich die Gleichung

$$\underbrace{\begin{bmatrix} \underline{u}_1 & \underline{u}_2 & \underline{u}_3 & \underline{u}_4 & \underline{u}_5 & \underline{u}_6 \\ \underline{p}_1 & \underline{p}_2 & \underline{p}_3 & \underline{p}_4 & \underline{p}_5 & \underline{p}_6 \end{bmatrix}}_{\underline{\underline{A}}} \underline{f}_l^{(i)} = \check{\underline{w}}_i, \quad i = \{f_x, f_y, f_z, m_x, m_y, m_z\} \tag{3.70}$$

mit $\underline{p}_i = \underline{u}_i \times \underline{r}_i$, wobei die Vektoren $\underline{f}_l^{(i)}$ für den Kraftwinder i die sechs Schnittkräfte in Längsrichtung der Beine beinhalten (Abb. 3.11). Die sechs Einheitskraftwinder werden mit $\check{\underline{w}}_i$ bezeichnet und zu

$$[\check{\underline{w}}_1 \quad \check{\underline{w}}_2 \ \dots \ \check{\underline{w}}_6] = [\check{\underline{f}}_x \quad \check{\underline{f}}_y \ \dots \ \check{\underline{m}}_z] = \underline{\underline{I}}_6 \tag{3.71}$$

zusammengefasst. Die aus den eingeprägten Kraftwindern resultierenden Schnittkräfte

$$\underline{\underline{F}} = [f_l^{(f_x)} \quad f_l^{(f_y)} \ \dots \ f_l^{(m_z)}] \tag{3.72}$$

werden analog als Matrix $\underline{\underline{F}}$ notiert. Nach dem Prinzip der kinematischen Differentiale werden die Schnittkräfte nach Gleichung (3.70) mit den Koeffizienten der JACOBI-Matrix identifiziert und es ergibt sich

$$\underline{\underline{J}}_l^{\mathrm{T}} = \underline{\underline{F}} = \underline{\underline{A}}^{-1}. \tag{3.73}$$

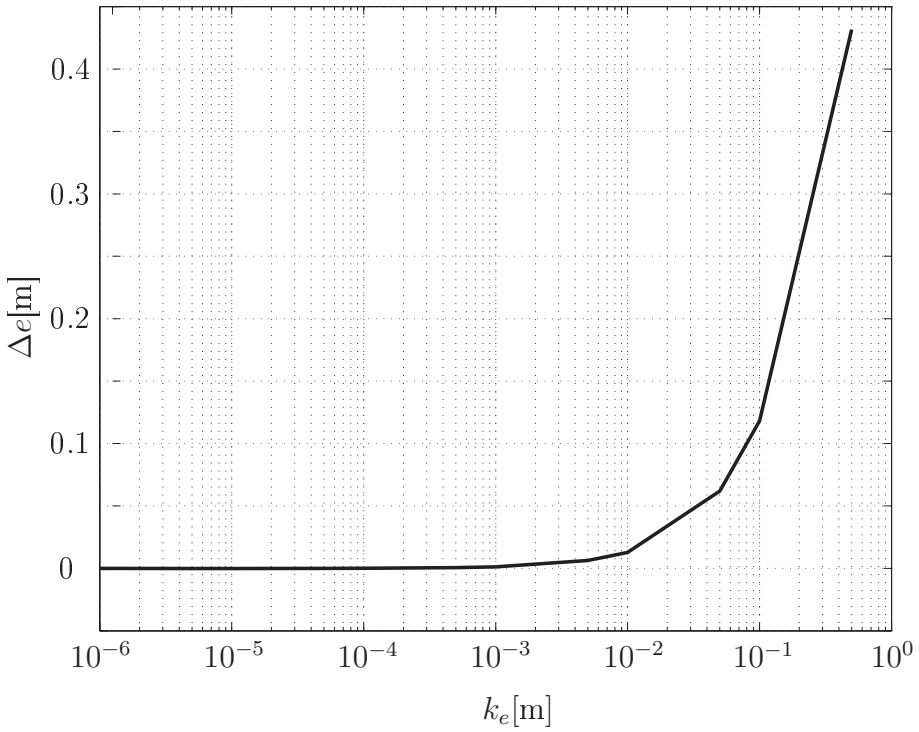


Abbildung 3.12: Differenz zwischen linearisiertem Modell und nichtlinearem Modell

Durch Einsetzten der nominellen geometrischen Parameter (vgl. Tab 2.4, S. 29) ergibt sich die JACOBI-Matrix $\underline{\underline{J}}_l$ für die angegebene Position zu

$$\underline{\underline{J}}_l^{\mathrm{T}} = \begin{bmatrix} 0.058 & 0.558 & -0.616 & -0.009 & 0.567 & -0.557 \\ -0.678 & 0.389 & 0.288 & -0.649 & 0.316 & 0.333 \\ -0.153 & -0.153 & -0.153 & -0.230 & -0.230 & -0.230 \\ -0.905 & -1.219 & 2.125 & 0.103 & 2.411 & -2.515 \\ 1.931 & -1.749 & -0.181 & -2.844 & 1.511 & 1.332 \\ -2.231 & -2.231 & -2.231 & 2.020 & 2.020 & 2.020 \end{bmatrix}. \tag{3.74}$$

Es wird nun angenommen, dass alle Beine einen Längenfehler von $\Delta\underline{e} = k_e[1,1,1,1,1,1]^{\mathrm{T}}$ aufweisen. Dann resultiert aus dem linearisierten Modell für $k_e = 10\,\mu$m ein Schätzwert für den Positionsfehler von $|\Delta r_{\mathrm{TCP}}| = 11.528\,\mu$m. Der exakte Wert aus der nichtlinearen Kinematik unter Berücksichtigung der veränderten Beinlängen liefert ebenfalls den Wert $|\Delta r_{\mathrm{TCP}}| = 11.528\,\mu$m. In Abbildung 3.12 ist der relative Fehler zwischen dem linearisierten Modell und der exakten Lösung mit dem nichtlinearen Modell über den Betrag des Fehlers k_e der Beine aufgetragen. Die Abbildung zeigt, dass die Näherung durch Linearisierung bis zu einem Fehler von ca. $k_e = 1\,$mm genau ist. Für einen Fehler in der Beinlänge von $k_e = 10\,$mm liegt der Fehler bei 1%. Das linearisierte Modell liefert also selbst bei relativ großen Fertigungstoleranzen eine gute Abschätzung des Fehlers am TCP.

Genauigkeit des *Linapod*

Im Gegensatz zum vorangegangen Abschnitt wird nun angenommen, dass die Maschine *Linapod* geometrische Fehler in allen Bauteilen aufweisen kann. Insofern wird die generelle

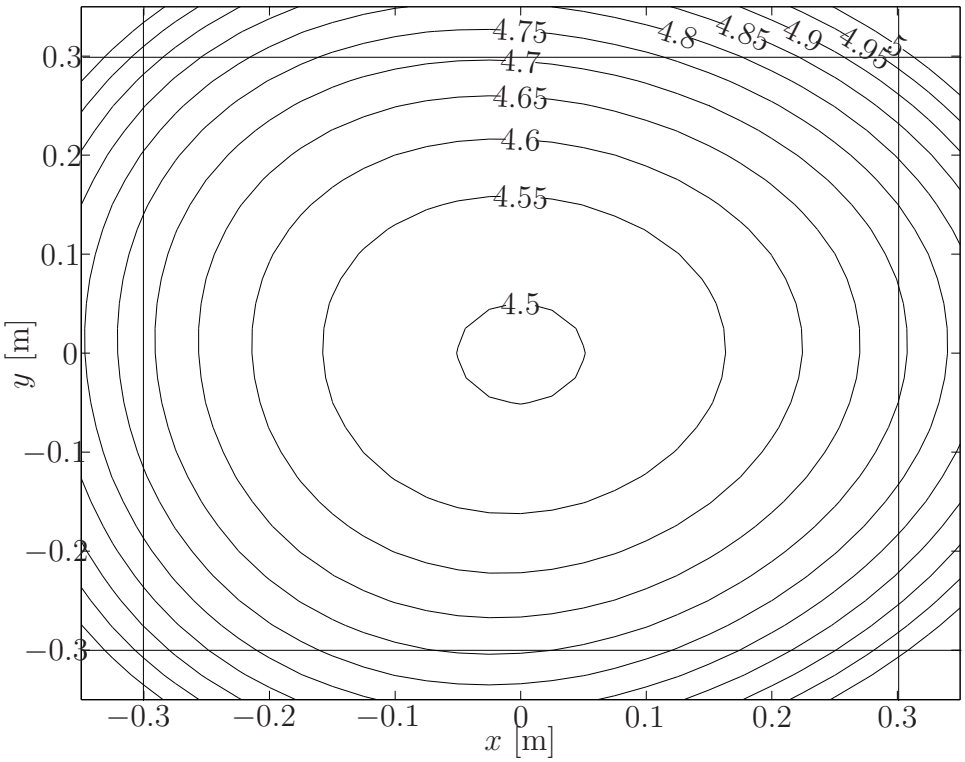


Abbildung 3.13: Gesamtfehlerverstärkung $\overline{\sigma}$ für gleichmäßig verteilte Fehler in allen Komponenten

Empfindlichkeit bezüglich Änderungen der Geometrie anhand der Matrix $\underline{\underline{J}}_g$ untersucht. Insgesamt werden hier 126 geometrische Parameter betrachtet, welche die Position des End-Effektors beeinflussen. Fehler in den Orientierungen der Bauteile werden für diese Betrachtung ignoriert, da ihr Einfluss auf den End-Effektor von der jeweiligen Größe der Bauteile abhängt und es daher nicht plausibel ist, diese Fehler gleichzusetzen. Zur Berechnung der Fehler wird das modulare kinetostatische Modell des *Linapod* verwendet (Abb. 3.8), das eine Berechnung der Schnittkräfte an allen Koordinatensystemen erlaubt. In Abbildung 3.13 ist die Gesamtfehlerverstärkung $\overline{\sigma}$ über dem translatorischen Arbeitsraum des *Linapod* in der $xy-$ Ebene für $z = 0$ aufgetragen. Es ist zu erkennen, dass die Empfindlichkeit der Maschine in der Mitte des Arbeitsraums den geringsten Wert von $\overline{\sigma} = 4.485$ annimmt und näherungsweise radialsymmetrisch zum Rand des Arbeitsraums hin zunimmt. Auf dem Rand des genutzten Arbeitsraums erreicht die Gesamtfehlerverstärkung einen maximalen Wert von $\overline{\sigma} = 5$. Bemerkenswerterweise ändert sich $\overline{\sigma}$ über dem Arbeitsraum nur um ca. 10%. Die Gesamtfehlerverstärkung kann genutzt werden, um abzuschätzen, wie genau die Parameter eines Modells bekannt sein müssen, um eine gegebene Genauigkeit zu erreichen. Wird beispielsweise unterstellt, dass eine Positioniergenauigkeit $\Delta e_{\max} = 10\,\mu$m erreicht werden soll, müssen die geometrischen Parameter des Modells folglich um den Faktor $\overline{\sigma} = 5$ genauer identifiziert werden, um die Vorgabe für die Genauigkeit zu erreichen. Weiterhin muss berücksichtigt werden, dass diese Modellrechnung den Einfluss von Winkelfehlern vernachlässigt, so dass die tatsächliche Gesamtfehlerverstärkung größer als 5 ist. Um hinreichend genaue Parameter für das Modell sicherzustellen, müssen entweder extrem enge Fertigungstoleranzen für die Bauteile angesetzt werden oder die Parameter müssen für die Bauteile entsprechend genau identifiziert

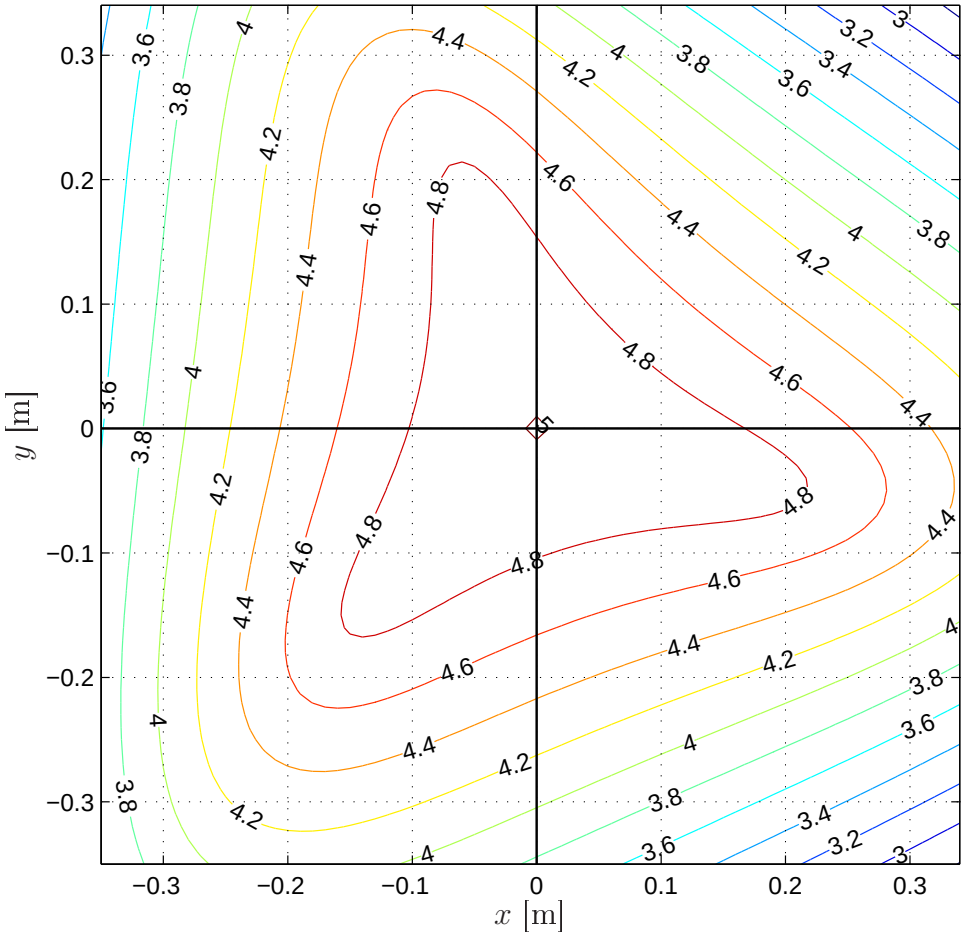


Abbildung 3.14: Kleinster Eigenwert $\lambda_{\min}$ [10^7 Nm^{-1}] der Steifigkeitsmatrix des *Linapod* über dem Arbeitsraum

werden. Dazu können die Bauteile entweder einzeln vermessen werden oder die Maschine wird einer Kalibrierung unterzogen. In diesem Zusammenhang kann die Gesamtfehlerverstärkung der Maschine als Anhaltspunkt verwendet werden um abzuschätzen, welche Genauigkeit für die Messmittel benötigt wird bzw. welche Toleranzen verwendet werden müssen, um eine vorgegebene Positioniergenauigkeit zu erreichen.

3.7.4 Steifigkeitsanalyse

Anhand des in Abschnitt 3.5 diskutierten Verfahrens lässt sich mithilfe der JACOBI-Matrix $\underline{\underline{J}}_g$ die Steifigkeitsmatrix $\underline{\underline{K}}_{\mathrm{EEF}}$ bestimmen. Die Berechnung der JACOBI-Matrix erfolgt, wie bereits bei der Sensitivitätsanalyse, mit dem modularen Modell der Maschine *Linapod*. Hier werden beispielhaft die Elastizitäten der Beine und der Linearantriebe betrachtet. Damit ergibt sich der Vektor der Geometrieparameter zu $\underline{g} = [l_1, l_2, l_3, l_4, l_5, l_6]^{\mathrm{T}}$ und es folgt für die JACOBI-Matrizen

$$\underline{\underline{J}}_g(\underline{q}) = \frac{\partial \underline{\varphi}^{\mathrm{DK}}(\underline{q}, \underline{g})}{\partial \underline{g}} \quad \text{und} \quad \underline{\underline{J}}_q(\underline{q}) = \frac{\partial \underline{\varphi}^{\mathrm{DK}}(\underline{q}, \underline{g})}{\partial \underline{q}}. \tag{3.75}$$

Für die Federkonstanten werden die Werte nach Tabelle 3.3 angenommen und zu lokalen Steifigkeitsmatrizen zusammengesetzt

$$\underline{\underline{K}}_g = \mathrm{diag}(c_l, c_l, c_l, c_u, c_u, c_u), \tag{3.76}$$

$$\underline{\underline{K}}_q = c_d \, \underline{\underline{I}}_6. \tag{3.77}$$

Damit ergibt sich die Steifigkeitsmatrix am End-Effektor zu

$$\underline{\underline{K}}_{\mathrm{EEF}}^{-1}(\underline{q}) = \underline{\underline{J}}_g(\underline{q})\underline{\underline{K}}_g^{-1}\underline{\underline{J}}_g^{\mathrm{T}}(\underline{q}) + \underline{\underline{J}}_q(\underline{q})\underline{\underline{K}}_q^{-1}\underline{\underline{J}}_q^{\mathrm{T}}(\underline{q}). \tag{3.78}$$

In der vorliegenden Berechnung der Steifigkeit wird nur der translatorische Anteil betrachtet, um die Vermischung von Translation und Rotation zu vermeiden. Als Maß für die minimale Steifigkeit wurde in Abbildung 3.14 der Betrag des kleinsten Eigenwerts [30] der Steifigkeitsmatrix über einem Querschnitt des Arbeitsraums in der xy-Ebene für $z = 0$ des *Linapod* aufgetragen. Die maximalen Werte werden in der Mitte des Arbeitsraums erreicht, während die Steifigkeit zum Rand hin abnimmt.

Tabelle 3.3: Federkonstanten für die Maschine *Linapod*

Symbol	Wert	Beschreibung
c_l	$8.8e7\,\mathrm{Nm}^{-1}$	Steifigkeit der unteren Beine
c_u	$6.0e7\,\mathrm{Nm}^{-1}$	Steifigkeit der oberen Beine
c_d	$8.13e8\,\mathrm{Nm}^{-1}$	Steifigkeit der Linearantriebe

3.8 Zusammenfassung

In diesem Kapitel wird ein Verfahren zur Modellierung der Kinematik von PKM durch das Zusammenfügen von Modulen vorgestellt. Damit gelingt eine vereinheitlichte Modellbildung für eine große Anzahl in der Praxis auftretender Maschinenkonzepte. Dazu gehören die Maschine *Linapod*, die für Fahrsimulatoren verwendeten Hexapoden (SSM) und der Delta-Roboter. Weiterhin lassen sich beliebige Mischformen dieser Maschinen sowie geometrische Varianten von Plattform und Rahmen abbilden und einheitlich behandeln. Die praktische Berechnung erfolgt anhand der kinetostatischen Methode, so dass sich alle für diese Methode hergeleiteten Algorithmen anwenden lassen, um die kinematischen Eigenschaften zu untersuchen.[6] Dazu gehören zunächst die Auswertung der direkten und inversen Kinematikfunktion sowie deren Ableitungen zur Berechnung von Geschwindigkeiten und Beschleunigungen. Mit den kinetostatischen Übertragungselementen lässt sich auch die Statikfunktion auswerten, um die Schnittkräfte für gegebene Belastungszustände zu ermitteln. Letztere Möglichkeit wird von einem neuen Algorithmus genutzt, der die exakte Linearisierung der Kinematik bezüglich aller geometrischer Parameter ermöglicht. Die Komplexität des Algorithmus hängt linear von der Anzahl der Körper und Gelenke ab, so dass für komplexe Mehrkörpersysteme wie PKM eine sehr kurze Rechenzeit erforderlich

[6] Mit dem Verfahrens der kinematischen Differentiale lassen sich auch ohne weiteren Aufwand die Bewegungsgleichungen aufstellen [47]. Dies wird in dieser Arbeit nicht weiter verfolgt, da es keine Relevanz für die hier entwickelte Maßsynthese hat.

ist. Bei der Anwendung auf PKM ist weiterhin hervorzuheben, dass das Verfahren auch dann die exakte Linearisierung bezüglich aller geometrischer Parameter ermöglicht, wenn keine geschlossene Lösung der Kinematik bekannt ist. Die linearisierte Kinematik wird eingesetzt, um eine Sensitivitätsanalyse durchzuführen und die Steifigkeit zu berechnen. Die für die Kalibrierung der Maschine notwendigen Ableitungen lassen sich ebenfalls auf diese Weise bestimmen.

Alle bisher betrachteten Verfahren sind auf die Berechnung der Eigenschaften an diskreten Punkten innerhalb des Arbeitsraums von PKM beschränkt. Dies ist für viele praktische Berechnungen wie die Kalibrierung hinreichend, jedoch für die Berechnung des Arbeitsraums und die Maßsynthese unzureichend. Daher wird in den folgenden Kapiteln ein Formalismus entwickelt, der garantierte Aussagen über eine kontinuierliche Menge ermöglicht.

Kapitel 4

Methoden zur kontinuierlichen Arbeitsraumanalyse

In Abschnitt 4.2 wird die Berechnung des Arbeitsraums von PKM als Constraint Satisfaction Problem *(CSP) formuliert und es werden Algorithmen vorgestellt, die auf der Intervallanalyse basieren. Techniken zur Verringerung der Rechenzeit werden in Abschnitt 4.3 behandelt. Im Abschnitt 4.4 werden prozessbedingte Anforderungen als Bindungen für das CSP aufgestellt. Beispiele für die Arbeitsraumberechnung von PKM finden sich in Abschnitt 4.5.*

4.1 Einleitung

Im vorangegangenen Kapitel 3 wurden mithilfe der kinetostatischen Methoden die Eigenschaften von parallelen Robotern an diskreten Punkten im Arbeitsraum betrachtet. Die Analyse solcher Punkte gibt präzise Informationen über die lokalen Eigenschaften der Maschine, beantwortet aber nur bedingt die Frage nach Größe und Gestalt des Arbeitsraums. Das Beispiel zur Arbeitsraumberechnung (Abb. 3.9) bietet zwar eine erste Abschätzung für den Arbeitsraum einer PKM, jedoch stellt sich die Frage, wie fein das Gitter gewählt werden muss und ob nicht selbst für ein sehr feines Gitter spezielle Punkte zwischen den Gitterpunkten existieren, die abweichende Eigenschaften aufweisen. Dieses Problem tritt in besonderem Maße bei PKM auf, da sich deren Eigenschaften über dem Arbeitsraum stark verändern können. Im Gegensatz zu seriellen Robotern können Singularitäten bei PKM mitten im Arbeitsraums auftreten. Im Allgemeinen lassen sich daher die Resultate der diskretisierten Punkte nicht auf den ganzen Arbeitsraum erweitern, da sich nur endlich viele Auswertungen durchführen lassen und keine gesicherten Informationen über die unendlich vielen übrigen Punkte vorliegen.

Daher werden in diesem Kapitel Methoden vorgestellt, die durch die Anwendung der Intervallanalyse verifizierte Informationen über ein Gebiet liefern, so dass sich der Arbeitsraum *kontinuierlich* betrachten lässt. Die Intervallanalyse erlaubt, mit endlich vielen Rechenschritten für alle Punkte im Arbeitsraum eine garantierte Entscheidung zu treffen, ob die geforderten Eigenschaften erfüllt sind oder nicht. Weiterhin kann mit der Intervallanalyse auch eine Abschätzung des ungünstigsten Verhaltens durchgeführt werden. Durch die Anwendung der Intervallanalyse weisen die hier vorgestellten Algorithmen den Vorteil auf, dass die Berechnungen numerisch robust durchgeführt werden, d. h. dass Rundungsfehler

während der Berechnung berücksichtigt werden und somit die Gültigkeit der Ergebnisse garantiert werden kann. Einige Kriterien verlieren bei der Betrachtung durch die Intervallanalyse jedoch etwas an Schärfe. Die mit der Intervallanalyse ermittelten Ergebnisse schließen stets die exakte Lösung ein, liefern jedoch nicht immer die engste Einschließung. In diesem Zusammenhang muss jedoch beachtet werden, dass bei vielen anderen Verfahren eine höhere Genauigkeit nur suggeriert wird, da diese Methoden ihre eigenen Verfahrensfehler weniger rigoros abschätzen, als dies bei der Intervallanalyse der Fall ist.

Im nächsten Abschnitt wird der Begriff des *Constraint Satisfaction Problem* (CSP) eingeführt, der in den folgenden Abschnitten die Grundlage für die Berechnung und Verifikation des Arbeitsraums darstellt. Danach werden mit der Intervallanalyse Algorithmen zur Lösung dieser Klasse von Problemen beschrieben. Im Anschluss daran werden die *Bindungen*[1] (engl. Constraints) eingeführt, die das CSP für PKM definieren. Die Kriterien für den kinematischen Arbeitsraum sowie für technologische Beschränkungen werden dann in Form solcher Bindungen formuliert. Dazu werden die Komponenten der Bindungsgleichungen mit einem Computeralgebra-System ausgewertet und als Programmcode für die Intervallanalyse exportiert. Am Ende des Kapitels werden Beispiele und existierende Maschinen präsentiert, für welche die Berechnung des Arbeitsraums unter verschiedenen Anforderungen durchgeführt wurde.

4.2 Algorithmen zur Lösung des Constraint Satisfaction Problems

Das Ziel der hier vorgestellten Verfahrens ist es, eine einheitliche Methodik für die Analyse und Synthese von PKM zur Verfügung zu stellen. Die Methode erlaubt es, beginnend mit einer vorläufigen Analyse bis hin zur Optimierung der Maschinen, alle erforderlichen Schritte durchzuführen, ohne dass die Bindungsgleichungen des Modells neu erstellt werden müssen. Dazu wird zunächst das folgende Problem

$$\underline{\Phi}(\underline{c}, \underline{v}) > \underline{0} \quad \forall\, \underline{v} \in \mathcal{X}_v \tag{4.1}$$

betrachtet. Dabei werden die *Berechnungsvariablen* im Vektor $\underline{c}$ zusammengefasst und die *Verifikationsvariablen* werden mit dem Vektor $\underline{v}$ bezeichnet. Die Menge $\mathcal{X}_v$ wird *Verifikationsmenge* genannt. Die Aufgabe, alle Lösungen eines solchen Ungleichungssystems $\underline{\Phi}$ zu finden, wird in der Literatur als *Constraint Satisfaction Problem* (CSP) bezeichnet. Folglich wird ein Algorithmus zur Lösung solcher Probleme *CSP-Löser* genannt. In diesem Kapitel werden in der Regel die Weltkoordinaten $\underline{y}$ der Plattform mit dem Vektor $\underline{c}$ identifiziert, so dass das CSP die Berechnung des Arbeitsraums ermöglicht. Die Aufgabe besteht also darin, alle Positionen zu bestimmen, an denen bestimmte Bedingungen erfüllt sind. Im folgenden Kapitel 5 wird die Aufgabe so umformuliert, dass durch ein CSP eine Maßsynthese durchgeführt werden kann.

Die Intervallanalyse hat sich als mächtiges Werkzeug zur Behandlung von CSP bewährt und wird im Folgenden auf die Arbeitsraumberechnung von PKM angewandt. Die grundlegende Idee aller folgenden Algorithmen besteht darin, eine Intervallauswertung der Nebenbedingungen $\underline{\Phi}$ für eine Box $\underline{\widehat{c}}$ dazu zu verwenden, um eine garantierte Aussage für

[1] Der Begriff der Bindung wird hier in Übereinstimmung mit der Literatur zum *constraint programming* verwendet und ist nicht mit der Bedeutung in der Kinematik gleichzusetzen.

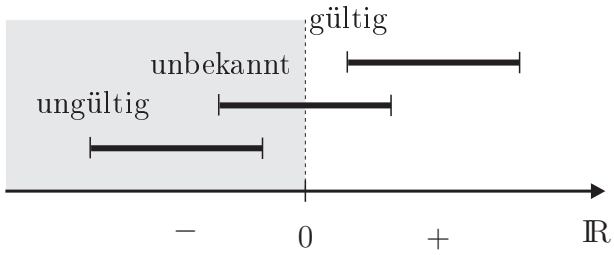


Abbildung 4.1: Bewertung einer Ungleichung $\Phi > 0$ durch den CSP-Löser

alle Punkte $\underline{c} \in \widehat{\underline{c}}$ zu machen. Dazu wird die Intervallauswertung $\widehat{\underline{h}} = \underline{\Phi}(\widehat{\underline{c}})$ berechnet[2] und das Ergebnis $\widehat{\underline{h}}$ wird folgendermaßen unterschieden (Abb. 4.1):

- Falls $\inf \widehat{\underline{h}} > \underline{0}$, ist die Ungleichung $\underline{\Phi}(\widehat{\underline{c}}) > \underline{0}$ in *jedem* Punkt $\underline{c} \in \widehat{\underline{c}}$ erfüllt. Eine solche Box $\widehat{\underline{c}}$ wird als *gültig* bezeichnet.
- Falls $\sup \widehat{\underline{h}}_i < \underline{0}$, ist die Ungleichung $\underline{\Phi}(\widehat{\underline{c}}) > \underline{0}$ in *keinem* Punkt $\underline{c} \in \widehat{\underline{c}}$ erfüllt. Eine solche Box $\widehat{\underline{c}}$ wird als *ungültig* bezeichnet.
- In allen anderen Fällen kann keine verlässliche Aussage gemacht werden. Aufgrund der möglichen Überschätzung durch die Intervallarithmetik kann die Existenz weder von gültigen noch von ungültigen Punkten festgestellt werden. Solche Boxen werden als *unsicher* bezeichnet.

Auf der Grundlage dieser fundamentalen Eigenschaft der Intervallanalyse werden im Folgenden Algorithmen vorgestellt, die es erlauben, die Lösungsmenge zu approximieren.

4.2.1 Generischer Algorithmus

Aus oben genannten Eigenschaften der Intervallanalyse wird nun ein *Branch-and-Bound*-Algorithmus konstruiert, der sich zur Lösung des CSP eignet. Dabei wird das Problem sukzessiv in kleinere Teilprobleme zerlegt (*branching*), für die gesicherte Schranken von $\underline{\Phi}$ bestimmt werden (*bounding*). Im Hinblick auf PKM wird dieser Algorithmus später für die Berechnung und Verifikation des Arbeitsraums spezialisiert. Hierzu werden die Bedingungen für den Arbeitsraum als Ungleichungen formuliert, so dass sie zu einem Ungleichungssystem $\underline{\Phi} > \underline{0}$ zusammengefasst werden können, welches dann systematisch ausgewertet werden kann. Die zentrale Aufgabe besteht darin, die Lösungsmenge

$$\mathcal{X}_c = \{\, \underline{c} \in \mathcal{X}_s \subset \mathbb{R}^n \,|\, \underline{\Phi}(\underline{c}) > \underline{0} \,\} \tag{4.2}$$

zu berechnen, die alle gestellten Bedingungen erfüllt. Dabei wird die Menge $\mathcal{X}_s \subset \mathbb{R}^n$ als Suchraum bezeichnet. Der prinzipielle Aufbau des Verfahrens entspricht dem folgenden Algorithmus:

Algorithmus 4: `Generischer Intervall-CSP-Löser`

1. Speichere eine Approximation $\{\widehat{\underline{c}}_1, \ldots, \widehat{\underline{c}}_n\}$ des Suchraums $\mathcal{X}_s$ in der Liste $\mathcal{L}_\mathrm{T}$.
2. Erstelle leere Listen $\mathcal{L}_\mathrm{S}$,$\mathcal{L}_\mathrm{I}$,$\mathcal{L}_\mathrm{F}$ für die Lösungsmenge ($\mathcal{L}_\mathrm{S}$), für die ungültigen Boxen ($\mathcal{L}_\mathrm{I}$) und für die Boxen, die zu klein sind ($\mathcal{L}_\mathrm{F}$).

[2] $\underline{\Phi}(\widehat{\underline{c}})$ ist eine Abkürzung für $\underline{\Phi}(\widehat{\underline{c}}, \widehat{\underline{v}})$ falls der Vektor $\widehat{\underline{v}}$ keine Verifikationsvariablen enthält.

3. Falls die Liste $\mathcal{L}_\mathrm{T}$ leer ist, beende den Algorithmus.
4. Entnimm die nächste Box $\underline{\widehat{c}}$ der Liste $\mathcal{L}_\mathrm{T}$.
5. Falls diam $\underline{\widehat{c}} < \varepsilon$, d. h. die Breite einer Komponente des Intervallvektors $\underline{\widehat{c}}$ ist kleiner als eine gegebene Genauigkeit ε, bewerte die Box als zu klein und lege sie in der Liste $\mathcal{L}_\mathrm{F}$ ab; gehe zu Schritt (3).
6. Sofern verfügbar, wende Operationen zum *Schrumpfen* von $\underline{\widehat{c}}$ an.
7. Werte die Bedingungen $\underline{\widehat{h}} = \underline{\Phi}(\underline{\widehat{c}})$ aus.
8. Falls $\inf \underline{\widehat{h}} > \underline{0}$, d. h. das Infimum von $\underline{\widehat{h}}$ ist für alle Bedingungen positiv und die Bedingung ist somit für alle $\underline{c} \in \underline{\widehat{c}}$ erfüllt, speichere $\underline{\widehat{c}}$ als Lösung in der Liste $\mathcal{L}_\mathrm{S}$ ab; gehe zu Schritt (3).
9. Falls ein $\sup \widehat{h}_i < 0$ existiert, d. h. mindestens ein Supremum ist negativ, dann existieren keine Lösungen $\underline{c} \in \underline{\widehat{c}}$ und die Box $\underline{\widehat{c}}$ wird in der Liste $\mathcal{L}_\mathrm{I}$ gespeichert; gehe zu Schritt (3).
10. Teile die Box $\underline{\widehat{c}}$ in m Teilboxen $\{\underline{\widehat{c}}_1, \ldots, \underline{\widehat{c}}_m\}$ auf und füge diese neuen Boxen in $\mathcal{L}_\mathrm{T}$ ein; gehe zu Schritt (3).

Die Beschreibung von Operationen zum *Schrumpfen* erfolgt im Abschnitt 4.3.3. Dieser Algorithmus bildet die Grundlage für die Lösung verschiedener Typen von CSP. Das Grundprinzip kann dem oben stehenden Algorithmus entnommen werden. Zu Beginn des Algorithmus werden die zu prüfenden Boxen in einer Liste platziert. Durch die Intervallanalyse werden für eine Box garantierte Schranken für den Wertebereich bestimmt. Ist dieser Wertebereich streng positiv, dann ist die Bedingung stets erfüllt. Ist der Wertebereich streng negativ, wird die Bedingung nirgendwo innerhalb der Box erfüllt. Entsprechend wird die Box entweder als Lösung betrachtet oder als ungültig gespeichert. Schließt der Wertebereich den Wert 0 ein, kann keine gesicherte Aussage getroffen werden. Allerdings kann die Box in zwei oder mehr kleinere Boxen aufgeteilt werden und dem Algorithmus erneut zugeführt werden. Dieses Verfahren lässt sich offensichtlich als rekursiver Algorithmus formulieren. Hier wird jedoch eine sequentielle Umsetzung verwendet, da sich einige Optimierungen so leichter durchführen lassen. Ferner kann der sequentielle Algorithmus leichter parallelisiert werden (Abschnitt 4.3.6). Die Verwaltung der Liste $\mathcal{L}_\mathrm{T}$ spielt bezüglich des benötigten Speicherplatzes eine relevante Rolle. Insbesondere die Operationen Einfügen und Entnehmen sollten so umgesetzt werden, dass sich die Liste $\mathcal{L}_\mathrm{T}$ wie ein *Stapel* (engl. Stack) verhält.

Zunächst werden aus dem oben genannten generischen Algorithmus des CSP-Lösers zwei wichtige Spezialisierungen abgeleitet. Dies führt in den folgenden Abschnitten zu den Algorithmen `Verifizieren` und `Berechnen`.

4.2.2 Verifikationsverfahren

Die Verifikation stellt die einfachste Form der Problemstellung dar. Gegeben ist eine Verifikationsmenge $\mathcal{X}_v$ und es soll überprüft werden, ob für ein gegebenes $\underline{c}$

$$\underline{\Phi}(\underline{c}, \underline{v}) > \underline{0} \quad \forall\, \underline{v} \in \mathcal{X}_v \tag{4.3}$$

erfüllt ist. Bezogen auf den Algorithmus soll also untersucht werden, ob *alle* in der Liste $\mathcal{L}_\mathrm{T}$ enthaltenen Boxen bezüglich des Systems $\underline{\Phi} > 0$ gültig sind. Daher sind als Resultat

der Verifikation zunächst die beiden logischen Ergebnisse *gültig* und *ungültig* zu erwarten. Darüber hinaus tritt auch der Fall auf, dass eine Verifikation nicht abgeschlossen werden kann, weil die zu untersuchenden Boxen die Mindestgröße ε_v unterschreiten. In diesem Fall wird das Ergebnis als *finit* bezeichnet. Für alle diese Ergebnisse ist es nicht notwendig, die Menge $\mathcal{L}_\mathrm{S}$ und $\mathcal{L}_\mathrm{I}$ zu speichern. Stattdessen wird der Algorithmus unter den folgenden Umständen abgebrochen: sobald eine Box als ungültig identifiziert wird, gibt der Algorithmus den Wert *ungültig* zurück. Unterschreitet eine Box die Mindestgröße ε_v, wird *finit* zurückgeliefert. Wenn alle Boxen aus der Liste $\mathcal{L}_\mathrm{T}$ erfolgreich geprüft wurden, gibt der Algorithmus den Wert *gültig* zurück. Daraus ergibt sich die folgende Variante des generischen Algorithmus:

Algorithmus 5: `Verifizieren`

1. Speichere die zu verifizierenden Boxen $\{\underline{\widehat{v}}_1, \ldots, \underline{\widehat{v}}_n\}$ in der Liste $\mathcal{L}_\mathrm{T}$.
2. Falls die Liste $\mathcal{L}_\mathrm{T}$ leer ist, beende den Algorithmus mit dem Ergebnis *gültig*.
3. Entnimm die nächste Box $\underline{\widehat{v}}$ der Liste $\mathcal{L}_\mathrm{T}$.
4. Falls diam $\underline{\widehat{v}} < \varepsilon_v$, d. h. die Breite einer Komponente des Intervallvektors $\underline{\widehat{v}}$ ist kleiner als eine gegebene Genauigkeit ε_v, beende den Algorithmus mit dem Ergebnis *finit*.
5. Falls Operationen zum *Schrumpfen* die Größe von $\underline{\widehat{v}}$ reduzieren, beende den Algorithmus mit dem Ergebnis *ungültig*.
6. Werte die Bedingungen $\underline{\widehat{h}} = \underline{\Phi}(\underline{\widehat{v}})$ aus.
7. Falls $\inf \underline{\widehat{h}} > \underline{0}$, d. h. das Infimum von $\underline{\widehat{h}}$ ist für alle Bedingungen positiv und die Bedingung ist somit für alle $\underline{v} \in \underline{\widehat{v}}$ erfüllt, verwirf $\underline{\widehat{v}}$; gehe zu Schritt (2).
8. Falls ein $\sup \widehat{h}_i < 0$ existiert, d. h. mindestens ein Supremum ist negativ, dann existieren keine Lösungen $\underline{v} \in \underline{\widehat{v}}$, beende den Algorithmus mit dem Ergebnis *ungültig*.
9. Teile die Box $\underline{\widehat{v}}$ in m Teilboxen $\{\underline{\widehat{v}}_1, \ldots, \underline{\widehat{v}}_m\}$ auf und füge diese neuen Boxen in $\mathcal{L}_\mathrm{T}$ ein; gehe zu Schritt (2).

4.2.3 Berechnungsverfahren

Bei der Berechnung wird im Gegensatz zur Verifikation verlangt, die Menge

$$\mathcal{X}_c = \{\, \underline{c} \in \mathcal{X}_s \subset \mathbb{R}^n \,|\, \underline{\Phi}(\underline{c}, \underline{v}) > \underline{0} \,\} \tag{4.4}$$

für ein gegebenes $\underline{v}$ zu ermitteln. Bei der Berechnung wird aus dem Suchraum $\mathcal{X}_s$, der durch die Liste $\mathcal{L}_\mathrm{T}$ dargestellt wird, die Menge $\mathcal{X}_c$ bestimmt, die durch die Boxen der Liste $\mathcal{L}_\mathrm{S}$ approximiert wird. Der Algorithmus liefert eine Liste von Boxen $\mathcal{L}_\mathrm{S}$, die eine Approximation der Menge $\mathcal{X}_c$ von unten darstellt, d. h. die Menge $\mathcal{L}_\mathrm{S}$ ist in $\mathcal{X}_c$ enthalten und konvergiert für $\varepsilon_c \to 0$ gegen $\mathcal{X}_c$. Die Teilung der Boxen wird auch bei der Berechnung nur bis zu einer Mindestgröße ε_c durchgeführt, denn ohne Abbruchkriterium würde die Berechnung in den meisten Fällen endlos fortgesetzt. Ob die Mengen der ungültigen Boxen $\mathcal{L}_\mathrm{I}$ und der finiten Boxen $\mathcal{L}_\mathrm{F}$ benötigt werden, hängt von der speziellen Anwendung ab. Zum Speichern der Mengen $\mathcal{L}_\mathrm{I}$ und $\mathcal{L}_\mathrm{F}$ wird jedoch eine erhebliche Menge Arbeitsspeicher benötigt. Daher wird $\mathcal{L}_\mathrm{F}$ nur dann gespeichert, wenn in einem weiteren Rechengang mit einer kleineren Genauigkeit ε_c die Güte der Approximation verbessert werden soll.

Bei größeren Problemen kann der Bedarf an Speicherplatz für $\mathcal{L}_\mathrm{I}$ und $\mathcal{L}_\mathrm{F}$ nicht vernachlässigt werden, so dass diese Boxen häufig verworfen werden. Es ergibt sich folgender Algorithmus:

Algorithmus 6: `Berechnen`

1. Speichere die zu untersuchenden Boxen $\{\underline{\widehat{c}}_1, \ldots, \underline{\widehat{c}}_n\}$ in der Liste $\mathcal{L}_\mathrm{T}$.
2. Erstelle leere Listen $\mathcal{L}_\mathrm{S}, \mathcal{L}_\mathrm{I}, \mathcal{L}_\mathrm{F}$ für die Lösungsmenge ($\mathcal{L}_\mathrm{S}$), für die ungültigen Boxen ($\mathcal{L}_\mathrm{I}$) und für die Boxen, die zu klein sind ($\mathcal{L}_\mathrm{F}$).
3. Falls die Liste $\mathcal{L}_\mathrm{T}$ leer ist, beende den Algorithmus.
4. Entnimm die nächste Box $\underline{\widehat{c}}$ der Liste $\mathcal{L}_\mathrm{T}$.
5. Falls diam $\underline{\widehat{c}} < \varepsilon_c$, d. h. die Breite einer Komponente des Intervallvektors $\underline{\widehat{c}}$ ist kleiner als eine gegebene Genauigkeit ε_c, bewerte die Box als *finit* und lege sie in der Liste $\mathcal{L}_\mathrm{F}$ ab; gehe zu Schritt (3).
6. Sofern verfügbar, wende Operationen zum *Schrumpfen* von $\underline{\widehat{c}}$ an.
7. Werte die Bedingungen $\underline{\widehat{h}} = \underline{\Phi}(\underline{\widehat{c}})$ aus.
8. Falls $\inf \underline{\widehat{h}} > \underline{0}$, d. h. das Infimum von $\underline{\widehat{h}}$ ist für alle Bedingungen positiv und die Bedingung ist somit für alle $\underline{c} \in \underline{\widehat{c}}$ erfüllt, speichere $\underline{\widehat{c}}$ als Lösung in der Liste $\mathcal{L}_\mathrm{S}$ ab; gehe zu Schritt (3).
9. Falls ein $\sup \widehat{h}_i < 0$ existiert, d. h. mindestens ein Supremum ist negativ, dann existieren keine Lösungen $\underline{c} \in \underline{\widehat{c}}$ und die Box $\underline{\widehat{c}}$ wird in der Liste $\mathcal{L}_\mathrm{I}$ gespeichert; gehe zu Schritt (3).
10. Teile die Box $\underline{\widehat{c}}$ in m Teilboxen $\{\underline{\widehat{c}}_1, \ldots, \underline{\widehat{c}}_m\}$ auf und speichere diese neuen Boxen in $\mathcal{L}_\mathrm{T}$; gehe zu Schritt (3).

4.2.4 Hybrides Verfahren

Für eine Reihe der in der Praxis auftretenden CSP wird eine Kombination von Berechnung und Verifikation benötigt. Ein Beispiel für solch eine Kombination ist die Bestimmung des *Gesamtorientierungsarbeitsraums*. Hierfür müssen alle Positionen der Plattform bestimmt werden, an denen jeweils alle Orientierungen aus einer vorgegebenen Menge möglich sind. Es sollen also alle Lösungen bestimmt werden, für die gilt

$$\mathcal{X}_c = \{\, \underline{c} \in \mathcal{X}_s \subset \mathbb{R}^n \,|\, \underline{\Phi}(\underline{c}, \underline{v}) > \underline{0} \quad \forall\, \underline{v} \in \mathcal{X}_v \subset \mathbb{R}^m \}\,. \tag{4.5}$$

Dies kann erreicht werden, wenn die Konzepte für Berechnung und Verifikation verschachtelt angewendet werden. Dazu werden im gleichen Rechengang sowohl Verifikationsvariablen $\underline{v}$ als auch Berechnungsvariablen $\underline{c}$ zugelassen. Ausgehend vom Aufbau des Algorithmus `Berechnen` wird anstelle der Auswertung des Systems $\underline{\Phi}$ eine Verifikation durchgeführt, wobei für die Berechnungsvariablen die aktuelle Box $\underline{\widehat{c}}$ verwendet wird. Als Algorithmus stellt sich das Verfahren folgendermaßen dar:

Algorithmus 7: `Hybrider CSP-Löser`

1. Speichere die zu untersuchenden Boxen $\{\underline{\widehat{c}}_1, \ldots, \underline{\widehat{c}}_n\}$ in der Liste $\mathcal{L}_\mathrm{T}$.
2. Erstelle leere Listen $\mathcal{L}_\mathrm{S}, \mathcal{L}_\mathrm{I}, \mathcal{L}_\mathrm{F}$ für die Lösungsmenge ($\mathcal{L}_\mathrm{S}$), für die ungültigen Boxen ($\mathcal{L}_\mathrm{I}$) und für die Boxen, die zu klein sind ($\mathcal{L}_\mathrm{F}$).

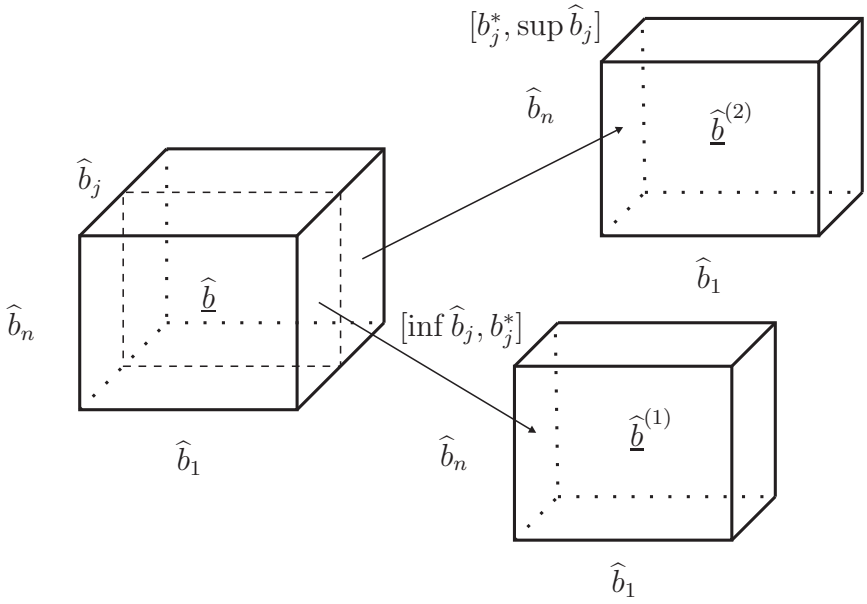


Abbildung 4.2: Ein Bisektionsschritt für die Box $\underline{\widehat{b}}$ an der j-ten Komponente

3. Falls die Liste $\mathcal{L}_\mathrm{T}$ leer ist, beende den Algorithmus.
4. Entnimm die nächste Box $\underline{\widehat{c}}$ der Liste $\mathcal{L}_\mathrm{T}$.
5. Falls diam $\underline{\widehat{c}} < \varepsilon_c$, d. h. die Breite einer Komponente des Intervallvektors $\underline{\widehat{c}}$ ist kleiner als eine gegebene Genauigkeit ε_c, bewerte die Box als *finit* und lege sie in der Liste $\mathcal{L}_\mathrm{F}$ ab; gehe zu Schritt (3).
6. Sofern verfügbar, wende Operationen zum *Schrumpfen* von $\underline{\widehat{c}}$ an.
7. Rufe den Algorithmus `Verifizieren` für die Box $\underline{\widehat{c}}$ und die Menge $\mathcal{X}_v$ auf.
8. Falls `Verifizieren` den Wert *gültig* zurückgibt, sind die Bedingungen $\underline{\Phi}(\underline{c}, \underline{v}) > 0$ für alle $\underline{c} \in \underline{\widehat{c}}$ und alle $\underline{v} \in \mathcal{X}_v$ erfüllt, speichere $\underline{\widehat{c}}$ als Lösung in der Liste $\mathcal{L}_\mathrm{S}$ ab; gehe zu Schritt (3).
9. Falls `Verifizieren` den Wert *ungültig* zurückgibt, erfüllt kein $\underline{c} \in \underline{\widehat{c}}$ die Bedingungen $\underline{\Phi}(\underline{c}, \underline{v}) > 0$ für alle $\underline{v} \in \mathcal{X}_v$, speichere $\underline{\widehat{c}}$ in der Liste $\mathcal{L}_\mathrm{I}$; gehe zu Schritt (3).
10. Falls `Verifizieren` den Wert *finit* zurückgibt, ist die Breite der Box $\underline{\widehat{c}}$ zu groß; teile die Box $\underline{\widehat{c}}$ in m Teilboxen $\{\underline{\widehat{c}}_1, \ldots, \underline{\widehat{c}}_m\}$ auf und speichere diese neuen Boxen in $\mathcal{L}_\mathrm{T}$; gehe zu Schritt (3).

4.2.5 Teilen von Boxen

Ein elementarer Schritt intervallbasierter Algorithmen besteht in der Aufteilung einer Box $\underline{\widehat{b}} \in \mathbb{I}^n$ in m kleinere Boxen $\{\underline{\widehat{b}}^{(1)}, \ldots, \underline{\widehat{b}}^{(m)}\}$. Diese Aufteilung kann nach verschiedenen Kriterien erfolgen. In dieser Arbeit wird davon ausgegangen, dass eine Box stets in zwei kleinere Boxen zerteilt wird, so dass dieser Vorgang als *Bisektion* bezeichnet wird (Abb. 4.2). Im Allgemeinen ist das Teilen jedoch nicht auf genau zwei Teilboxen beschränkt. Bei der Bisektion der Box werden alle Komponenten bis auf die j-te Komponente beibehalten. Diese wird an einer Stelle $b_j^* \in \widehat{b}_j$ zerteilt, so dass die Box in die Hälften

$$\underline{\widehat{b}} \;=\; \underline{\widehat{b}}^{(1)} \;\cup\; \underline{\widehat{b}}^{(2)} \quad \text{mit} \tag{4.6}$$

$$\widehat{\underline{b}} = [\widehat{b}_1, \ldots, \widehat{b}_j, \ldots, \widehat{b}_n], \tag{4.7}$$

$$\widehat{\underline{b}}^{(1)} = [\widehat{b}_1, \ldots, [\inf \widehat{b}_j, b_j^*], \ldots, \widehat{b}_n], \tag{4.8}$$

$$\widehat{\underline{b}}^{(2)} = [\widehat{b}_1, \ldots, [b_j^*, \sup \widehat{b}_j], \ldots, \widehat{b}_n] \tag{4.9}$$

zerfällt. Für die Teilung der Box wird normalerweise der Mittelpunkt des Intervalls

$$b_j^* = \operatorname{mid} \widehat{b}_j = \frac{1}{2}(\inf \widehat{b}_j + \sup \widehat{b}_j) \tag{4.10}$$

verwendet. Beim Teilen der Box stellt sich weiterhin die Frage, welche Komponente geteilt wird. Eine einfache und effektive Methode besteht darin, die Komponente j mit der größten Breite diam b_j zu teilen. Wenn die verschiedenen Komponenten inhomogen in ihrer Größenordnung sind, z. B. wenn Winkel und Längen betrachtet werden, kann die Auswahl durch individuelle Wichtungsfaktoren g_j homogenisiert werden. In diesem Fall wird j so ausgewählt, dass diam $(g_j \widehat{b}_j)$ maximal ist, wobei g_j die jeweiligen Wichtungsfaktoren sind. Neben der Auswahl der größten Komponente kann j zyklisch verändert oder zufällig ausgewählt werden.

4.3 Effizienzsteigerung der Algorithmen

Die Grundstruktur der vorgestellten Algorithmen zur Lösung von CSP ist sehr einfach, führt aber häufig zu inakzeptabel langen Laufzeiten und einem hohen Speicherverbrauch. Daher haben sich Verfahren und Heuristiken etabliert, welche die Menge der benötigten Ressourcen stark reduzieren. Einige dieser Verfahren mit Anwendung auf PKM werden in den folgenden Abschnitten beschrieben.

4.3.1 Separation der Verifikationsmenge

Bei einigen Anforderungen lassen sich die Bindungen des Systems $\underline{\Phi}$ und die damit verbundene Verifikationsmenge $\mathcal{X}_v$ separieren, d. h. in diesen Fällen kann das CSP nach Gleichung (4.1) in die Teilprobleme

$$\underline{\Phi}_1(\underline{c}, \underline{v}_1) > \underline{0} \quad \forall\, \underline{v}_1 \in \mathcal{X}_v^{(1)}, \tag{4.11}$$

$$\vdots$$

$$\underline{\Phi}_n(\underline{c}, \underline{v}_n) > \underline{0} \quad \forall\, \underline{v}_n \in \mathcal{X}_v^{(n)}, \tag{4.12}$$

$$\underline{v}^{\mathrm{T}} = [\underline{v}_1^{\mathrm{T}}, \ldots, \underline{v}_n^{\mathrm{T}}] \tag{4.13}$$

aufgeteilt werden. Dies ist beispielsweise möglich, wenn für bestimmte Bindungen Hilfsvariablen[3] eingeführt werden. Die entkoppelten Subsysteme $\underline{\Phi}_i$ werden durch gemeinsame Berechnungsvariablen $\underline{c}$ bestimmt, besitzen jedoch für jede Bindung $\underline{\Phi}_i$ separate Verifikationsvariablen $\underline{v}_i$. Bei dem hybriden Verfahren (Abschnitt 4.2.4) wird dazu, anstelle einer Auswertung des Algorithmus `Verifizieren` für das Gesamtsystem $\underline{\Phi}$, für jedes Teilsystem eine individuelle Auswertung von `Verifizieren` durchgeführt. Durch die getrennte Verifikation wird vermieden, Bisektionsschritte auf Verifikationsvariablen v_i anzuwenden, die in diesem Teil des Systems nicht vorkommen.

[3] Beispielsweise werden die Parameter λ_i als Hilfsvariablen zur Beschreibung der Stabkollision in Abschnitt 4.4.5 eingeführt.

4.3.2 Abhängigkeiten zwischen Bindungen

Das System der Bindungsungleichungen $\underline{\Phi}$ beinhaltet eine Anzahl verschiedener Kriterien, anhand derer entschieden wird, ob eine Konfiguration als *gültig* oder *ungültig* bewertet wird. Bei praktischen Problemen besteht aber häufig eine kausale Abhängigkeit zwischen den Bindungen, d. h. eine Bindung Φ_1 kann eine notwendige Bedingung dafür sein, dass Bindung Φ_2 erfüllt sein kann. Daher wird Bindung Φ_2 erst dann berechnet und überprüft, wenn das System bezüglich Bindung Φ_1 als *gültig* bewertet wurde. Zur Steigerung der Effizienz bietet es sich daher an, Beziehungen zwischen den Bindungen auszunutzen, um unnötige Rechenoperationen zu vermeiden. Dazu können vom Anwender leicht Regeln vorgegeben werden, die diese Abhängigkeiten beschreiben. Als Beispiel wird die Bindung für die *kinematische Erreichbarkeit* und die Bindung für die *Grenzen der passiven Gelenke* betrachtet. Da es nur sinnvoll ist, eine Konfiguration auf Einhaltung der Gelenkbegrenzungen zu überprüfen, wenn sie innerhalb des erreichbaren Arbeitsraums liegt, wird die Prüfung der Gelenkbegrenzungen nur ausgeführt, wenn die *kinematische Erreichbarkeit* verifiziert wurde. Ein weiterer Grund für die Ausnutzung von Abhängigkeiten ergibt sich daraus, dass die Prüfung der *kinematischen Erreichbarkeit* numerisch schneller ausgeführt werden kann. Weiterhin können bestimmte Bindungen gar nicht berechnet werden, wenn nicht zuvor einige Voraussetzungen verifiziert wurden. Hier wäre z. B. der Zusammenhang zwischen *kinematischer Erreichbarkeit* und *Autokollisionen* zu nennen. Solange durch die kinematische Erreichbarkeit nicht verifiziert wurde, ob es überhaupt Lösungen für die inverse Kinematik gibt, können die Lagen der Bauteile nicht ermittelt werden, die für die Autokollision aber erforderlich sind. Durch Abhängigkeiten lassen sich daher sehr effizient Ausdrücke behandeln, die für bestimmte Werte nicht definiert sind. Als Beispiel wird die vereinfachte Bindung betrachtet

$$\Phi_1 = \sqrt{x} - y > 0. \tag{4.14}$$

Damit der Radikand in der Bindung Φ_1 ausgewertet werden kann, wird die zusätzliche Bindung

$$\Phi_2 = x > 0. \tag{4.15}$$

eingeführt und es muss zunächst anhand von Φ_2 geprüft werden, ob x positiv ist. Diese Abhängigkeit wird in den Algorithmus eingearbeitet, indem schlichte wenn-dann-Bedingungen formuliert werden. Hängen die Bindungen Φ_1, wie im Beispiel, von Φ_2 ab, wird die Auswertung von Φ_1 solange aufgeschoben, bis für Φ_2 der Status *gültig* erreicht wird. Praktisch werden dadurch zahlreiche Auswertungen von Φ_1 eingespart. Gleichzeitig kann die Zuverlässigkeit der Programme erhöht werden, da ungültige Operationen vermieden werden können.

4.3.3 Heuristiken zur Verbesserung der Laufzeit

Die oben vorgestellten Methoden verringern die Laufzeit der Algorithmen, indem zusätzlich Informationen über das CSP $\underline{\Phi} > \underline{0}$ verwendet werden. Daher wird die Anzahl der Funktionsauswertungen reduziert oder bleibt im ungünstigsten Fall gleich. In diesem Abschnitt werden Heuristiken vorgestellt, die in sorgfältig ausgewählter Kombination zu einer erheblichen Reduktion der Rechenzeit führen können. Bei einer ungünstigen Wahl der Parameter verlängert sich die Rechenzeit jedoch signifikant. Für eine sinnvolle Anwendung der Heuristiken ist daher ein erhebliches Maß an Erfahrung erforderlich, um die

Methoden für ein spezielles Problem auszuwählen. Eine automatische Auswahl und Parametrisierung ist noch Gegenstand der aktuellen Forschung. Daher müssen die in diesem Abschnitt vorgestellten Methoden vorsichtig angewendet werden.

Die Auswertung an diskreten Punkten liefert eine notwendige Bedingung für die Verifikation. Da die Verifikation das Ziel hat, die Gültigkeit der Bedingungen für jeden Punkt $\underline{c} \in \widehat{\underline{c}}$ zu prüfen, kann die Box $\widehat{\underline{c}}$ bereits anhand eines Punkts verworfen werden, welcher das System $\underline{\Phi} > 0$ nicht erfüllt. Als spezielle Punkte bieten sich der Mittelpunkt $\mathrm{mid}\,\widehat{\underline{c}}$ sowie die Eckpunkte der Box an. Beim Test der Eckpunkte müssen 2^n Funktionsauswertungen berechnet werden, wobei n die Dimension der Box $\widehat{\underline{c}}$ ist. Von diesen Eckpunkten teilt eine Box jeweils bis zu 2^{n-1} Eckpunkte mit ihren 2^n Nachbarn. Weiterhin sind zahlreiche Punkte zwischen geteilten und ursprünglichen Boxen identisch. Damit der Test effizient funktioniert, muss ein aufwendiges Verfahren zur Auswahl verwendet werden, um mehrfache Funktionsauswertungen für identische Punkte zu vermeiden.

Die Hauptaufgabe des CSP-Lösers besteht darin zu untersuchen, ob Ungleichungen der Form $\Phi(\widehat{\underline{c}}) > 0$ in einer Box $\widehat{\underline{c}}$ Lösungen besitzen. Durch *Konsistenztests* kann geprüft werden, ob eine Funktion Φ in einem Teil oder dem gesamten Bereich einer Variablen c_i Lösungen besitzt [39]. Dazu wird $\Phi(\underline{c})$ zunächst als Gleichung $\Phi(\underline{c}) = g(c_i) - h(\underline{c}) = 0$ betrachtet, wobei die Funktion $g(\widehat{c}_i)$ so gewählt wird, dass sich deren Umkehrfunktion g^{-1} leicht und eindeutig bestimmen lässt. Die Funktion $h(\widehat{\underline{c}})$ kann dabei durchaus von c_i abhängen.[4] Eine Intervallauswertung $\widehat{h}_0 = h(\widehat{\underline{c}})$ liefert garantierte Schranken $\widehat{h}_0$ für den Wertebereich der Funktion $h(\widehat{\underline{c}})$. Diese Abschätzung des Wertebereichs von $h(\widehat{\underline{c}})$ wird verwendet, um den Wertebereich von $g(c_i)$ auf Konsistenz zu untersuchen. Da die Umkehrfunktion g^{-1} bekannt ist, lässt sich auch ein zulässiger Bereich für $\widehat{c}_i$ zu $\widehat{c}_i^* = g^{-1}(h(\widehat{\underline{c}}))$ bestimmen. Lösungen existieren folglich nur in der Schnittmenge der Intervalle $\widehat{c}_i' = \widehat{c}_i^* \cap \widehat{c}_i$. Damit ergibt sich der Iterationsschritt für den Konsistenztest zu

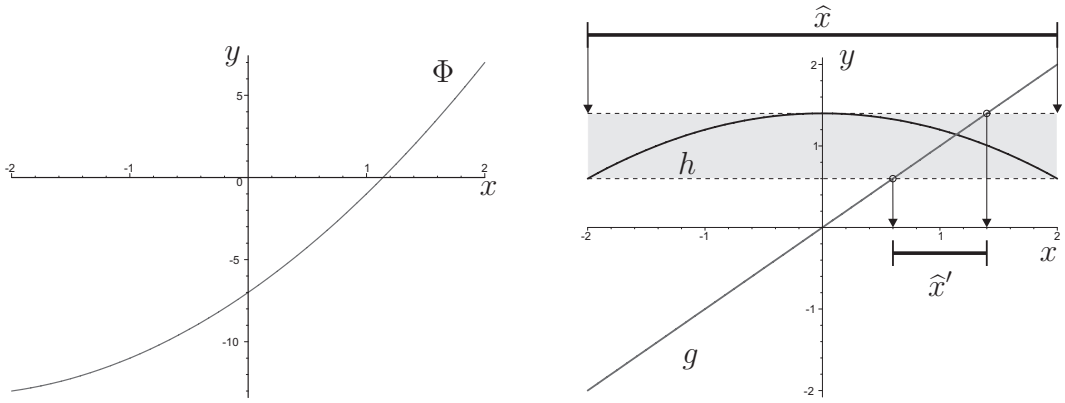
$$\widehat{c}_i' = g^{-1}(h(\widehat{\underline{c}})) \;\cap\; \widehat{c}_i. \tag{4.16}$$

Der Konsistenztest wird auch verwendet, um die gesamte Box $\widehat{\underline{c}}$ zu verwerfen, denn wenn die Intervalle $g^{-1}(h(\widehat{\underline{c}}))$ und $\widehat{c}_i$ disjunkt sind, enthält $\widehat{\underline{c}}$ keine Lösungen von Φ. Die Betrachtung der Schnittmenge in Gleichung (4.16) garantiert, dass der Iterationsschritt die Box $\widehat{\underline{c}}$ verkleinert oder zumindest unberührt lässt. Daher ist der Konsistenztest dazu geeignet, als *Schrumpf*-Operator zu fungieren. Auf die in dieser Arbeit betrachteten Ungleichungen kann der Konsistenztest angewendet werden, indem für den Wertebereich der Gleichung $\Phi = g - h = [0, \infty]$ betrachtet wird.

Der 2B-Test (auch *Hull Consistency Test* genannt [20, 39]) ist eine spezielle Form des Konsistenztests, bei dem ein linearer ($g(c_i) = c_i$) oder ein quadratischer Term ($g(c_i) = c_i^2$) einer Unbekannten c_i abgespalten wird. Für beide Varianten gilt, dass sich die Umkehrfunktion g^{-1} leicht bestimmen lässt. Durch Vergleich des Wertebereichs der Unbekannten mit dem Wertebereich der Funktion h kann gegebenenfalls das Intervall der Unbekannten verkleinert werden (Berechnung) oder die Box kann verworfen werden (Verifikation). Im Folgenden wird der 2B-Test an einem einfachen Beispiel veranschaulicht. Dazu wird die Gleichung $\Phi : x^2 + 5x - 7 = 0$ auf dem Intervall $\widehat{x} = [-2, 2]$ betrachtet (Abb. 4.3a). Für den 2B-Test wird die Gleichung folgendermaßen zerlegt:

$$\Phi : g - h \;=\; 0 \quad \text{mit} \tag{4.17}$$

[4] Es ist also nicht nötig, $\Phi(\underline{c})$ nach c_i aufzulösen.



a) ursprüngliche Funktion $y = \Phi(x)$

b) Zerlegung der Funktion in g, h und Auswertung mit Intervallen

Abbildung 4.3: Schritte beim 2B-Test für eine einfache Funktion

$$g = x, \tag{4.18}$$

$$h = \frac{1}{5}(7 - x^2). \tag{4.19}$$

Zunächst wird eine Intervallauswertung von h für das Intervall $\widehat{x}$ zu $h([-2,2]) = [\frac{3}{5}, \frac{7}{5}]$ bestimmt (Abb. 4.3b grauer Wertebereich). Damit ergibt sich eine garantierte Abschätzung des Wertebereichs von h. Um die Gleichung $\Phi = 0$ zu lösen, muss die Funktion g denselben Wertebereich wie $h(\widehat{x})$ besitzen. Da die Funktion $g(x) = x$ die triviale Umkehrfunktion $g^{-1}(x) = x$ besitzt, lässt sich unmittelbar folgern, dass die gesuchte Nullstelle (der Schnittpunkt von g und h) im Intervall $\widehat{x}' = g^{-1}(h(\widehat{x})) \cap \widehat{x}$ liegen muss. Daher kann das Intervall $\widehat{x}$ auf das Intervall $\widehat{x}' = [\frac{3}{5}, \frac{7}{5}]$ reduziert werden. Ein weiterer Iterationsschritt liefert $\widehat{x}'' = [1.008, 1.328]$. In ähnlicher Weise lässt sich der 2B-Test für die Ungleichung $\Phi > 0$ verwenden. Es ergibt sich die erweiterte Funktion $h^* = \frac{1}{5}([0, \infty] + 7 - x^2)$ und die Intervallauswertung liefert $h^*([-2,2]) = [\frac{3}{5}, \infty]$. Die Schnittmenge mit dem Intervall $\widehat{x}$ ergibt $\widehat{x}' = [\frac{3}{5}, 2]$. Dieses Intervall beinhaltet mögliche Lösungen, kann aber durchaus die Lösungsmenge überschätzen. Der Konsistenztest beweist nur, dass in dem Teil $[-2, \frac{3}{5}]$ keine Lösungen der Ungleichung existieren. Ein zweiter Iterationsschritt liefert für die Ungleichung $\Phi > 0$ ebenfalls $\widehat{x}'' = \widehat{x}' = [\frac{3}{5}, 2]$, d. h. der 2B-Test konvergiert in einigen Fällen nicht.

Der 2B-Test ist ein sehr mächtiges Werkzeug, wenn die Box $\underline{\widehat{c}}$ relativ groß ist. Die Konvergenzrate nimmt in der Regel für kleinere Boxen ab und kann – wie das Beispiel oben zeigt – vorzeitig zum Stillstand kommen. Dennoch kann durch einige Iterationsschritte eine deutliche Reduktion der Intervallbreite und damit auch der Rechenzeit erzielt werden. In der Praxis ist es leider sehr aufwendig, den 2B-Test zu implementieren, da die betrachteten Gleichungen für jede mögliche Unbekannte umgestellt werden müssen. Diese kann jedoch nur von Hand oder durch Computeralgebra-Systeme geschehen, welche außerhalb von Programmen wie Maple nicht verfügbar sind. Andererseits steht innerhalb von Computeralgebra-Systemen nur eine stark eingeschränkte Programmierumgebung zur Verfügung.

Beim 3B-Test (auch als *Bound Consistency Test* bekannt) werden *Scheiben* der Box von den Rändern abgetrennt und analysiert [20]. Entsprechend kann durch den 3B-Test die

Größe der Box verringert werden. Daher gehört der 3B-Test zu den *Schrumpf-* (prune-) Operationen und kann sowohl für die Verifikation als auch für die Berechnung einsetzt werden. Der 3B-Test scheint für die Behandlung von Ungleichungssystemen nur eine begrenzte Effizienz zu besitzen, während er bei der Lösung von Gleichungssystemen gute Dienste leistet [20].

Für den `hybriden CSP-Löser` (Abschnitt 4.2.4) hat es sich beim Algorithmus `Verifizieren` bewährt, eine maximale Lebensdauer (*Time to Live*) einzuführen. Die Idee einer Lebensdauer ist an das Internetprotokoll (IP) [100] angelehnt und besteht darin, die maximale Anzahl der Funktionsauswertungen innerhalb des Verifikationsverfahren zu begrenzen. Sind bei einem hybriden Problem die Breiten der Berechnungsvariablen $\operatorname{diam} \underline{\widehat{c}}$ zu groß, dann ist es für beliebig kleine $\underline{\widehat{v}}$ bei einer Verifikation nicht möglich, eine verlässliche Aussage (*gültig* oder *ungültig*) zu erhalten. Das normale Verifikationsverfahren bricht jedoch die Suche erst ab, wenn die Box der Verifikationsvariablen $\underline{\widehat{v}}$ kleiner als ε_v ist. Diesem Effekt wirkt die Lebensdauer der Boxen entgegen, indem sie die Anzahl der Funktionsauswertungen $\underline{\Phi}$ des Algorithmus `Verifizieren` begrenzt. Wird die zulässige Anzahl der Funktionsauswertungen überschritten, liefert der Algorithmus den Rückgabewert *finit*, da dieser Rückgabewert veranlasst, eine Bisektion der Berechnungsvariablen $\underline{\widehat{c}}$ durchzuführen, um so eine Konvergenz zu erreichen.

Da die Wahrscheinlichkeit einer abgeschlossenen Verifikation zunimmt, wenn die Größe der Boxen $\underline{\widehat{c}}$ reduziert wird, ist es sinnvoll, die Lebensdauer für kleinere Boxen zu erhöhen. Bevor eine Box $\underline{\widehat{c}}$ endgültig aufgrund einer zu geringen Größe verworfen wird, wird die Lebensdauer für einen letzten Aufruf von `Verifizieren` auf unendlich erhöht.

4.3.4 Verwendung des Gradienten

Mithilfe der Ableitung Φ' der betrachteten Funktion Φ lassen sich nützliche Schlüsse über die Eigenschaften der Funktion ziehen. Die Intervallauswertung der Ableitung $\widehat{h}' = \Phi'(\widehat{c})$ kann eine Aussage über die Monotonie der Funktion liefern. Wenn der Wert $0 \notin \widehat{h}'$ nicht innerhalb des Intervalls liegt, also $\inf \widehat{h}' > 0$ (oder $\sup \widehat{h}' < 0$) gilt, dann ist die Funktion auf dem Intervall $\widehat{c}$ streng monoton steigend (bzw. fallend). Also nimmt die Funktion ihre extremalen Werte an den Intervallgrenzen $\inf \widehat{c}_i, \sup \widehat{c}_i$ an. Für die Verifikation ist eine untere Schranke für den Wertebereich der Funktion ausreichend, und eine notwendige und hinreichende Bedingung für die Gültigkeit des Intervalls $\widehat{c}$ ist $\Phi(\inf \widehat{c}_i) > 0$ (bzw. $\Phi(\sup \widehat{c}_i) > 0$). Diese Bedingung gilt auch bei der Berechnung als notwendige und hinreichende Bedingung für die Beurteilung *gültig*. Analog kann anhand von $\Phi(\sup \widehat{c}_i) < 0$ (bzw. $\Phi(\sup \widehat{c}_i) < 0$) gezeigt werden, dass $\widehat{c}$ *ungültig* ist.

Im Fall eines Vektorfelds $\underline{\Phi} : \mathbb{I}^n \to \mathbb{I}^m$ ergibt sich anstelle einer Ableitung die JACOBI-Matrix der partiellen Ableitungen. Die Überlegungen lassen sich hier analog anwenden. Dazu werden die Elemente $[\underline{\underline{J}}]_{ij}$ der JACOBI-Matrix betrachtet. Falls ein Element mit $0 \notin [\underline{\underline{J}}]_{ij}$ existiert, ist die i-te Funktion in der j-ten Variable monoton. Diese Intervallvariable kann daher durch die reellen Werte ihres Infimums und Supremums ersetzt werden. Diese Vereinfachung verbessert in der Regel auch die Schranken für die Berechnung der übrigen partiellen Ableitungen, so dass der Test rekursiv angewendet werden kann. Auch wenn diese Strategie wirkungsvoll den Wertebereich der Funktion abschätzt, besteht in der Praxis das Problem darin, im richtigen Moment die Rekursion abzubrechen, um nicht zu viel Zeit auf diesen Test zu verwenden.

Um die notwendigen Auswertungen des Gradienten durchzuführen, gibt es zwei verschiedene Ansätze. Entweder werden die partiellen Ableitungen mit Computeralgebra symbolisch berechnet oder durch *automatische Differentiation*[5] bestimmt.

4.3.5 Nachträgliches Zusammenfassen von Boxen

Die Effizienz eines Algorithmus hängt vom Verbrauch der beiden Ressourcen *Speicher* und *Prozessorzeit* ab. Die bisher genannten Techniken verringern die Anzahl der notwendigen Funktionsauswertungen und damit die Prozessorzeit des Algorithmus. Neben der Rechenzeit spielt für die Leistung eines Algorithmus auch der benötige Speicherplatz eine Rolle. Um möglichst sparsam mit dem Speicherplatz umzugehen, ist es wünschenswert, die Lösungsmenge durch wenige und daher große Boxen darzustellen. Zur Veranschaulichung des Einsparpotentials wird der Algorithmus `Verifizieren` mit dem Algorithmus `Berechnen` für eine Box $\underline{\widehat{c}}$ verglichen. Für nichttriviale Bindungen kann der Algorithmus `Verifizieren` das Ergebnis nicht in einem Schritt liefern, sondern muss diverse Bisektionsschritte ausführen, bis die gesamte Box $\underline{\widehat{c}}$ bestätigt wird. Wird der Algorithmus `Berechnen` für das gleiche Problem gestartet, müssen die gleichen Bisektionsschritte ausgeführt werden, so dass die Menge der Lösungen $\mathcal{L}_\mathrm{S}$ diverse Boxen enthält, obwohl die Lösung durch die ursprüngliche Box $\underline{\widehat{c}}$ dargestellt werden kann. Dieser Effekt rührt daher, dass bei der Intervallauswertung einer komplexen Funktion der Wertebereich stark überschätzt wird.

Dieser Effekt kann vermieden werden, indem bei der Bisektion die ursprüngliche Box in einem Zwischenspeicher abgelegt wird. Gelingt es dann, alle Subboxen einer so gespeicherten Box zu verifizieren, können diese Subboxen gelöscht werden und stattdessen wird die ursprüngliche Box der Lösung $\mathcal{L}_\mathrm{T}$ hinzugefügt. Damit diese Technik effizient funktioniert, muss sie bei jedem Iterationsschritt angewendet werden. Der Zwischenspeicher kann dabei effizient als Stapel organisiert werden und verbraucht in der Praxis nur $n_\mathrm{cache} \sim \log \varepsilon_c$ Speicherplatz.

4.3.6 Parallele Implementierung

Die Berechnung des Arbeitsraums und vor allem die im folgenden Kapitel 5 behandelte Maßsynthese führen auf sehr komplexe hybride CSP. Besonders bei der später eingeführten Maßsynthese wird die Funktion $\underline{\Phi}$ milliardenfach ausgewertet, so dass die Rechenzeit eine wichtige Rolle spielt. Daher werden im Folgenden Möglichkeiten zur Verteilung des Problems auf mehrere Prozessoren vorgestellt. Für die parallele Berechnung haben sich anstelle der klassischen Supercomputer zunehmend Netzwerke von gewöhnlichen PCs etabliert. Eine Parallelisierung des CSP-Lösers hat jedoch auch für den Einsatz an einem einzelnen Computer eine wichtige Perspektive. Während der ersten Jahrzehnte der Prozessorentwicklung wurde die stetig voranschreitende Miniaturisierung vor allem zur Anhebung der Taktfrequenz genutzt. Trotz kontinuierlicher Verkleinerung der Struktur gestaltet sich eine weitere Steigerung schwierig, da mit einer höheren Taktfrequenz die thermischen Verluste zu einer schwer beherrschbaren Aufheizung des Chips führen. Daher zeichnet sich ein Paradigmenwechsel ab, der dazu führt, die Rechenleistung nicht mehr

[5] Mit der sogenannten *automatischen Differentiation* werden von symbolischen Ausdrücken oder von ganzen Algorithmen gleichzeitig die partiellen Ableitungen und der Funktionswert berechnet.

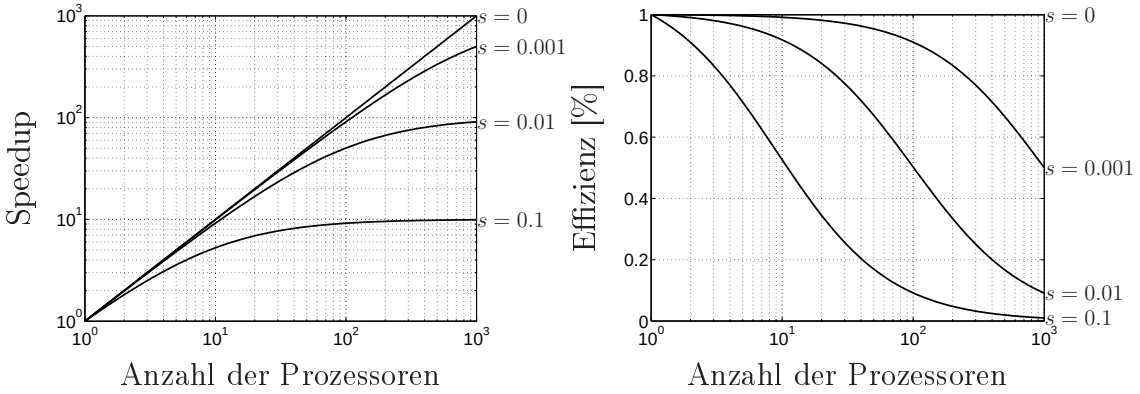


Abbildung 4.4: Grenzen der Skalierbarkeit paralleler Programme nach dem AMDAHL'schen Gesetz

durch die Taktfrequenz sondern durch eine Integration gleichartiger Prozessorkerne auf einem Chip zu steigern. Die ersten Chips mit zwei, vier und acht Prozessorkernen werden bereits produziert. Die Planungen der Chiphersteller sehen aber bereits in absehbarer Zeit Computer mit 64 und mehr Kernen vor. Die im Folgenden vorgestellte parallele Version des CSP-Lösers ist ideal dazu geeignet, die höhere Rechenleistung dieser neuen Computer auszuschöpfen.

Um diesen Zuwachs der Rechenleistung ausnutzen zu können, muss sich eine Aufgabe jedoch sinnvoll auf mehrere Prozessoren aufteilen lassen. Der Algorithmus für den CSP-Löser beruht auf einer einfachen *Teile-und-Herrsche*-Strategie, welche die Auswertungen des Systems $\underline{\Phi}$ für verschiedene Boxen $\widehat{c}, \widehat{v}$ unabhängig von einander durchführen kann. Somit zerfällt die Aufgabe von Natur aus in kleinere Probleme, die sehr gut auf einem parallelen Computer berechnet werden können. Eine Analyse der Laufzeit zeigt, dass der Hauptteil der Rechenzeit für die Auswertung von $\underline{\Phi}$ verwendet wird. Die Effizienz und Skalierbarkeit eines parallelen Programms hängt substanziell davon ob, wie viel zusätzlicher Aufwand für die Verwaltung und Kommunikation zwischen den Prozessoren erforderlich ist. Dies lässt sich anhand des AMDAHL'schen Gesetzes untersuchen, das den möglichen Geschwindigkeitszuwachs paralleler Programme durch den sogenannten *Speedup* beschreibt. Mit Speedup wird der Faktor bezeichnet, um den die Rechenzeit gegenüber einem seriellen Programm reduziert werden kann. Sei $s \in [0, 1]$ der nicht parallelisierbare Anteil eines Programms, T_0 die Zeit, die das Programm auf einem Prozessor benötigt und p die Anzahl der Prozessoren, dann kann die Ausführungszeit T des Programms auf

$$T = \left(s + \frac{1 - s}{p} \right) T_0 \tag{4.20}$$

reduziert werden. Das Gesetz zeigt jedoch auch eine klare Grenze für den Speedup eines Programmes auf, denn ein Programm kann sogar mit unendlich vielen Prozessoren ($p \to \infty$) nicht in weniger Zeit ausgeführt werden, als für den rein seriellen Teil benötigt wird. Ein Vergleich der theoretisch möglichen Skalierbarkeit und der Effizienz je Prozessor für verschiedene serielle Anteile s des Programms ist in Abbildung 4.4 dargestellt. Bezüglich des hier betrachteten CSP-Lösers bedeutet dies, dass der relative Anteil der Kommunikation und Verwaltung verglichen mit der Zeit der Berechnung minimiert werden muss. Da alle komplexen Probleme mithilfe des hybriden CSP-Lösers bearbeitet werden, kann die Parallelisierung jedoch auf einem höheren Niveau als der Auswertung

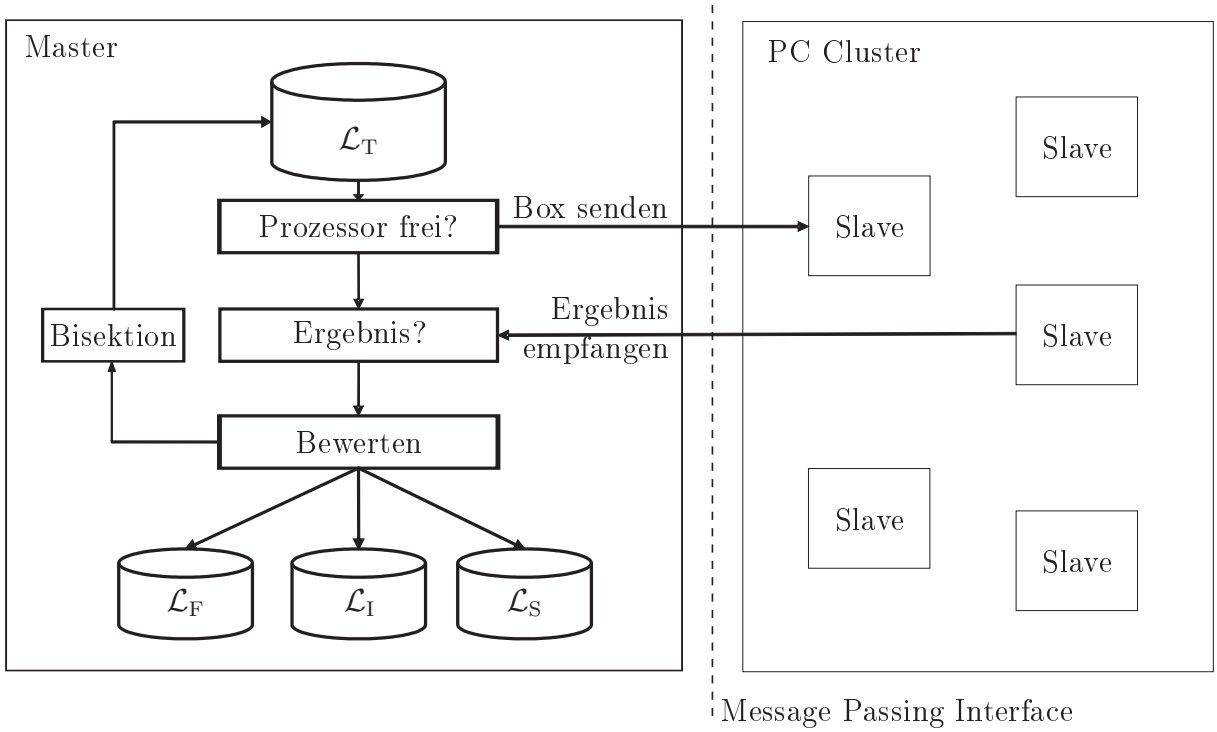


Abbildung 4.5: Algorithmusprinzip für einen parallelen CSP-Löser

der Funktion $\underline{\Phi}$ vorgenommen werden. Dazu werden die Auswertungen des Algorithmus `Verifizieren` parallelisiert, der wiederum zahlreiche Auswertungen des Systems $\underline{\Phi}$ erfordert. Da die Schnittstelle des Algorithmus `Verifizieren` nur wenige Daten überträgt, entsteht ein geringer Aufwand für die Kommunikation.

Der parallelisierte CSP-Löser besteht aus zwei Programmteilen (Abb. 4.5), die als Master/Slave organisiert sind. Die Aufgabe des Master besteht in der Verwaltung der Arbeit, während eine variable Anzahl Slaves die Berechnungen durchführt. Im Unterschied zum seriellen Algorithmus (Abschnitt 4.2.1) wird bei einem parallelen Programm der Aufruf von `Verifizieren` durch das Senden einer Nachricht an einen freien Slave ersetzt. In ähnlicher Weise wird das Ergebnis der Funktion durch eine Nachricht vom Slave an den Master übergeben. Der grundlegende Unterschied zum seriellen Algorithmus besteht darin, das Programm asynchron ablaufen zu lassen, d. h. der Master wartet nicht das Ergebnis der gestellten Aufgabe ab. Stattdessen prüft der Master zyklisch, ob weitere Boxen an freie Slaves verteilt werden können und ob Ergebnisse von den Slaves zurückgeliefert werden. Damit ergibt sich für den Master der folgende Algorithmus für den hybriden CSP-Löser:

Algorithmus 8: `Paralleler CSP-Löser (Master)`

1. Speichere die zu untersuchenden Boxen $\{\underline{\widehat{c}}_1, \ldots, \underline{\widehat{c}}_n\}$ in der Liste $\mathcal{L}_T$.
2. Erstelle leere Listen $\mathcal{L}_S$,$\mathcal{L}_I$,$\mathcal{L}_F$ für die Lösungsmenge ($\mathcal{L}_S$), für die ungültigen Boxen ($\mathcal{L}_I$) und für die Boxen, die zu klein sind ($\mathcal{L}_F$).
3. Markiere alle verfügbaren Slaves als *frei*.
4. Falls die Liste $\mathcal{L}_T$ leer ist und alle Slaves *frei* sind, beende den Algorithmus.
5. Falls mindestens ein Slave *frei* ist, führe aus:

   (a) Entnimm der Liste $\mathcal{L}_T$ die nächste Box $\underline{\widehat{c}}$.

(b) Falls diam $\widehat{\underline{c}} < \varepsilon_c$, d. h. die Breite einer Komponente des Intervallvektors $\widehat{\underline{c}}$ ist kleiner als eine gegebene Genauigkeit ε_c, bewerte die Box als zu klein und lege sie in der Liste $\mathcal{L}_\mathrm{F}$ ab; gehe zu Schritt (5).
(c) Sende eine Nachricht mit der Box $\widehat{\underline{c}}$ an einen freien Slave, markiere diesen als *beschäftigt*; gehe zu Schritt (5).

6. Falls von mindestens einem Slave eine Nachricht vorliegt, führe aus:

(a) Entnimm der Nachricht die Box $\widehat{\underline{c}}$ und deren Bewertung. Markiere den Slave als *frei*.
(b) Falls die Box $\widehat{\underline{c}}$ *gültig* oder *ungültig* ist, speichere sie in den entsprechenden Listen $\mathcal{L}_\mathrm{T}$ oder $\mathcal{L}_\mathrm{I}$; gehe zu Schritt (6).
(c) Falls die Box *unsicher* ist, teile die Box $\widehat{\underline{c}}$ in m Teilboxen $\{\widehat{\underline{c}}_1, \ldots, \widehat{\underline{c}}_m\}$ auf und speichere diese neuen Boxen in $\mathcal{L}_\mathrm{T}$; gehe zu Schritt (6).

7. Gehe zu Schritt (4).

Entsprechend wird der Teil zum Berechnen der Gleichungen und zur Bewertung der Ergebnisse von Slaves ausgeführt. Das Programm wird dabei so ausgestaltet, dass der Slave sich so lange in einem Wartezustand befindet, bis er einen Befehl vom Master bekommt, den er umgehend ausführt. Nachdem dieser Befehl verarbeitet wurde, wird das Ergebnis in Form einer Nachricht an den Master geschickt. Danach geht der Slave wieder in den Wartezustand. Um die Rechenleistung der Slaves optimal auszunutzen, ist es günstig, komplexe Teilprobleme zu delegieren, um das Verhältnis zwischen Rechen- und Wartezeit zu optimieren. Passend zum Master führt der Slave folgenden Algorithmus aus:

Algorithmus 9: `Paralleler CSP-Löser (Slave)`

1. Warte auf eine Nachricht vom Master. Entnimm der Nachricht die Box $\widehat{\underline{c}}$.
2. Führe den Algorithmus `Verifizieren` für die Box $\widehat{\underline{c}}$ aus.
3. Falls $\inf \widehat{\underline{h}} > \underline{0}$, d. h. das Infimum von $\widehat{\underline{h}}$ ist für alle Bedingungen positiv und die Bedingung ist somit für alle $\underline{c} \in \widehat{\underline{c}}$ erfüllt, sende das Ergebnis *gültig* und die Box $\widehat{\underline{c}}$ an den Master; gehe zu Schritt (1).
4. Falls ein $\sup \widehat{h}_i < 0$ existiert, d. h. mindestens ein Supremum ist negativ, dann existieren keine Lösungen $\underline{c} \in \widehat{\underline{c}}$. Sende das Ergebnis *ungültig* und die Box $\widehat{\underline{c}}$ an den Master; gehe zu Schritt (1).
5. Sende das Ergebnis *unsicher* und die Box $\widehat{\underline{c}}$ an den Master; gehe zu Schritt (1).

In der Implementierung wurde das Versenden und Empfangen von Nachrichten mithilfe der weit verbreiteten Bibliothek MPI (Message Passing Interface) realisiert [84, 106]. MPI hat sich im Bereich der Supercomputer praktisch zum Standard entwickelt und wird von jeder Hard- und Software für Großrechner unterstützt. Insbesondere ist es auch möglich, Standard-PCs mit unterschiedlichen Prozessoren und einfacher Hardware mit MPI zu vernetzen.

4.4 Prozessbedingte Anforderungen als Bindungen

Unter Bindungen (engl. *Constraints*) werden in diesem Zusammenhang die Bedingungen oder Anforderungen verstanden, die eine PKM erfüllen muss, um eine gegebene Aufgabe

erfüllen zu können. An erster Stelle muss daher qualitativ überprüft werden, ob eine Position zum erreichbaren Arbeitsraum gehört. Darüber hinaus existieren weitere Ausschlusskriterien, die über eine technische Nutzbarkeit entscheiden. Dazu gehören z. B. die Freiheit des gewünschten Arbeitsraums von Singularitäten und Autokollisionen sowie die Einhaltung der Begrenzungen der passiven Gelenke. Schließlich werden quantitative Aspekte geprüft, welche die Erfüllung von Mindestanforderungen im Arbeitsraum gewährleisten. Aufgrund des modularen Aufbaus der Betrachtung können beliebig viele Anforderungen miteinander kombiniert werden, so dass es zweckmäßig ist, mit der Betrachtung der einfachsten Bedingung zu beginnen.

4.4.1 Kinematisch erreichbarer Arbeitsraum der Beine

Als kinematisch erreichbarer Arbeitsraum wird die Menge aller Posen der Plattform bezeichnet, die sich rein geometrisch realisieren lassen. Technische Details der Realisierung wie Kollisionen und Begrenzungen der passiven Gelenke werden daher zunächst nicht betrachtet. Für die unterschiedlichen Beine ergeben sich verschiedene Gleichungen, die in den folgenden Abschnitten beschrieben werden.

Als Grundlage für die Betrachtung des kinematisch zulässigen Arbeitsraums wird untersucht, unter welchen Bedingungen sich die Funktion der inversen Kinematik auswerten lässt. Dazu wird zunächst der einfache Fall des UPS-Beins betrachtet. Die Länge des Beins kann für beliebige Positionen und Orientierungen der Plattform nach Gleichung (2.21) berechnet werden. Reale Schubgelenke sind jedoch auf einen endlichen Weg begrenzt, so dass die folgende Abschätzung für jedes Bein gilt

$$0 \leq s_{\min} \leq s_i \leq s_{\max}, \tag{4.21}$$

wobei $s_{\min}$ die minimale und $s_{\max}$ die maximale Länge des Schubgelenks ist. Es können auch verschiedene Längen der Schubgelenke für jedes einzelne Bein betrachtet werden und alle vorgestellten Verfahren funktionieren auch mit diesen flexibleren Parametern. Aus Gründen der Übersichtlichkeit wird auf diese zusätzliche Unterscheidung verzichtet. Durch Einsetzen von Gleichung (2.21) und Übergang auf Komponenten[6] lassen sich die beiden Ungleichungen

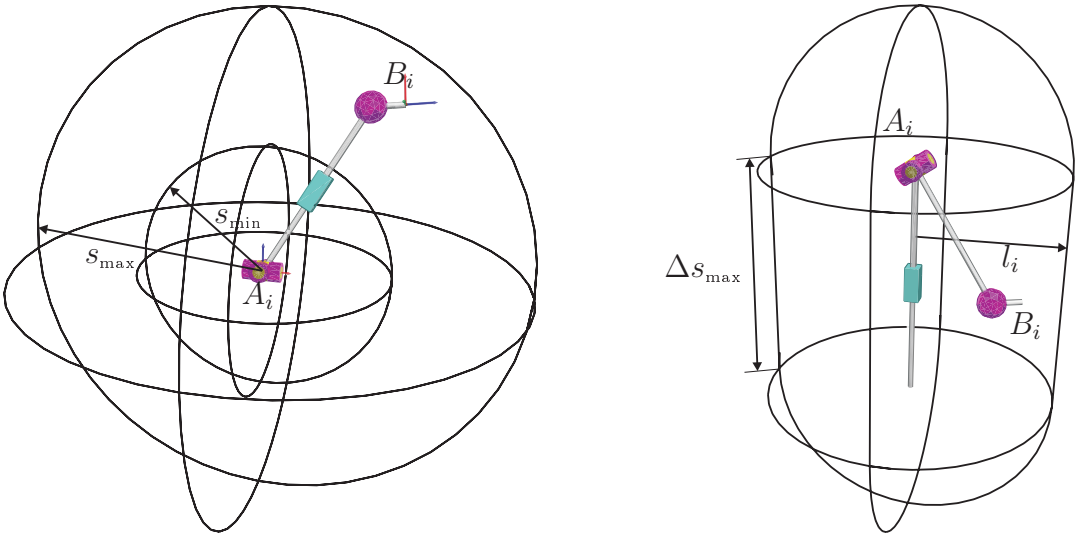
$$\left.\begin{array}{rcl} q_i - s_{\min} & > & 0 \\ s_{\max} - q_i & > & 0 \end{array}\right\} \quad \text{mit} \quad q_i^2 = \left(\underline{r} + \underline{\underline{R}}\,\underline{b}_i - \underline{a}_i\right)^2 \quad \text{für} \quad i = 1, \ldots, 6 \tag{4.22}$$

für jedes Bein ableiten. Die geometrische Interpretation dieser Gleichungen ist, dass die plattformseitigen Anlenkpunkte im Inneren einer Kugelschale liegen (Abb. 4.6a).

Der erreichbare Arbeitsraum eines PUS-Manipulators wird durch die konstante Länge des Beins und durch die Länge der Schubgelenke begrenzt. Auch hier können Kriterien aus der inversen Kinematik abgeleitet werden, die den Arbeitsraum beschränken. Aus Gleichung (2.23) folgt, dass nur dann eine Lösung für die inverse Kinematik existiert, wenn der Radikand positiv ist. Es folgt also

$$(\underline{n}_i \cdot \underline{d}_i)^2 - \underline{d}_i^2 + l_i^2 > 0. \tag{4.23}$$

[6] An dieser Stelle wird das Weltkoordinatensystem zur Zerlegung aller Vektoren zugrunde gelegt. Daher werden alle physikalischen Vektoren $\boldsymbol{a}$ durch ihre Komponenten $\underline{a}$ dargestellt, um sie einer Auswertung mit der Intervallanalyse zugänglich zu machen.



a) UPS-Bein b) PUS-Bein

Abbildung 4.6: Erreichbarer Arbeitsraum

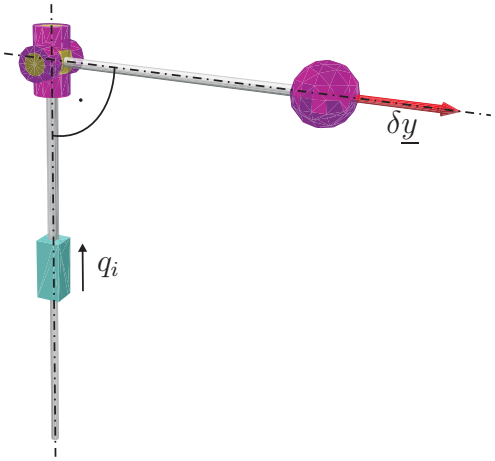
Unter der Voraussetzung, dass eine solche Lösung existiert, lässt sich die Position s_i des Schubgelenks berechnen, so dass geprüft werden kann, ob das Schubgelenk innerhalb seines zulässigen Bereichs liegt. Dazu wird die inverse Kinematik berechnet, so dass die Position des Schubgelenks ähnlich wie in Gleichung (4.22) bezüglich der maximalen Länge des Schubgelenks geprüft werden kann. Dies geschieht durch

$$\left.\begin{array}{rcl} q_i - l_{\min} & > & 0 \\ l_{\max} - q_i & > & 0 \end{array}\right\} \quad \text{für} \quad i = 1, \ldots, 6, \tag{4.24}$$

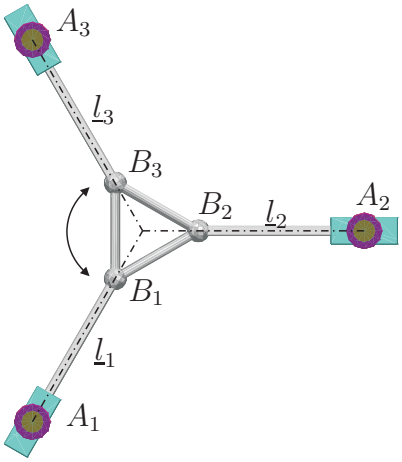
wobei sich auch hier die verallgemeinerte Koordinate q_i des Schubgelenks nach Gleichung (2.23) ergibt. Diese beiden Kriterien lassen sich gut anhand von Abbildung 4.6b veranschaulichen. Hier bilden alle erreichbaren Positionen des plattformseitigen Anlenkpunkts eine zylinderförmige Menge, deren oberes und unteres Ende durch Halbkugeln abgeschlossen ist.

4.4.2 Konvexität des Arbeitsraums

Der erreichbare translatorische Arbeitsraum des PUS-Beins ist konvex und der Arbeitsraum der gesamten Maschine ergibt sich als Schnittmenge der Arbeitsräume der sechs Beine. Da die Schnittmenge beliebig vieler konvexer Mengen ebenfalls konvex ist, ist auch der translatorische Arbeitsraum der gesamten Maschine konvex. In der Regel wird gefordert, dass der Arbeitsraum die Form eines Quaders hat. Folglich kann die Untersuchung auf einen Test der Eckpunkte dieses Quaders reduziert werden. Die Untersuchung der Eckpunkte dient als notwendiges und hinreichendes Kriterium. Dazu werden in Anlehnung an den Test diskreter Punkte (Abschnitt 4.3.3) die Eckpunkte des Suchraums in die Gleichungen eingearbeitet, so dass geprüft wird, ob alle Eckpunkte des gewünschten Arbeitsraums innerhalb des Arbeitsraums der Maschine liegen. Dieser Test kann besonders effizient bei dem Algorithmus `Verifizieren` für den Arbeitsraum angewendet werden, da das Einsetzen der Eckpunkte in die Gleichungen einige Intervallvariablen eliminiert.



a) Singularität erster Art am Beispiel des PUS-Beins



b) Singularität zweiter Art in der Mitte des Arbeitsraums bei einem symmetrischen ebenen RPR-Manipulator

Abbildung 4.7: Singuläre Stellung von PKM

Dadurch wird die Überschätzung reduziert und eine raschere Konvergenz ermöglicht. Die Ausnutzung der Konvexität ermöglicht bei der Maßsynthese eine relevante Verbesserung der Laufzeit, da die Weltkoordinaten dort als Verifikationsvariablen auftreten.

4.4.3 Singularitäten

Der Begriff der Singularität wird nach dem allgemeinen Ansatz von Gosselin [35] eingeführt. Die impliziten Schließbedingungen für PKM lauten

$$\underline{\nu}(\underline{y}, \underline{q}) = \underline{0} \tag{4.25}$$

und eine Differentiation liefert

$$\underbrace{\frac{\partial \underline{\nu}(\underline{y}, \underline{q})}{\partial \underline{y}}}_{\underline{\underline{A}}} \delta\underline{y} + \underbrace{\frac{\partial \underline{\nu}(\underline{y}, \underline{q})}{\partial \underline{q}}}_{\underline{\underline{B}}} \delta\underline{q} = \underline{0}. \tag{4.26}$$

Falls die Matrix $\underline{\underline{A}}$ regulär ist, wird die Gleichung zu

$$\delta\underline{y} = -\underline{\underline{A}}^{-1}\underline{\underline{B}}\delta\underline{q} \tag{4.27}$$

umgestellt, so dass sich die JACOBI-Matrix als $\underline{\underline{J}}^{\mathrm{DK}} = -\underline{\underline{A}}^{-1}\underline{\underline{B}}$ identifizieren lässt. Für vollständig parallele Roboter ist die Matrix $\underline{\underline{B}}$ eine Diagonalmatrix, da jedes Bein genau eine verallgemeinerte Koordinate enthält. Nach Gosselin befindet sich der Roboter in einer singulären Pose, wenn $\underline{\underline{A}}$ oder $\underline{\underline{B}}$ singulär ist. Es lassen sich drei verschiedene Arten von Singularitäten unterscheiden [35]. Wenn die Determinante von $\underline{\underline{B}}$ verschwindet, verliert die Plattform einen oder mehr Freiheitsgrade. Gleichzeitig existieren aufgrund des kinetostatischen Dualismus bestimmte Kraftwinder, die auf die Antriebe des Roboters keine Kräfte ausüben. Diese Art von Singularität kann z. B. bei Stabkinematiken auftreten, wenn die

Stäbe senkrecht zur Gelenkachse des Schubgelenks stehen (Abb. 4.7a). Verschwindet die Determinante der Matrix $\underline{\underline{A}}$, dann gewinnt der Roboter einen oder mehr Freiheitsgrade an der Plattform. Das bedeutet, dass sich die Plattform auch dann bewegen lässt, wenn alle verallgemeinerten Koordinaten konstant sind (Abb. 4.7b). Die dritte Art von Singularität tritt auf, wenn gleichzeitig $\det \underline{\underline{A}} = \det \underline{\underline{B}} = 0$ auftritt. Dies ist jedoch nur dann möglich, wenn die geometrischen Parameter der Maschine ganz speziell gewählt werden.

Alternativ ergibt sich die JACOBI-Matrix aus der Geschwindigkeitsübertragungsfunktion nach Gleichung (2.31) zu

$$\dot{\underline{y}} = \underline{\underline{J}}^{\mathrm{DK}}(\underline{q})\dot{\underline{q}}. \tag{4.28}$$

Dabei hängt die JACOBI-Matrix $\underline{\underline{J}}^{\mathrm{DK}}$ nur von der Geometrie des Roboters und der aktuellen Konfiguration $\underline{q}$ ab. Der Roboter befindet sich in einer singulären Konfiguration (Singularität), wenn die Determinante $\det \underline{\underline{J}}^{\mathrm{DK}} = 0$ verschwindet, die Matrix also nicht den vollen Rang hat. Es existieren folglich von Null verschiedene Geschwindigkeiten $\dot{\underline{q}}$ an den Antrieben, die keine Bewegung $\dot{\underline{y}}$ der Plattform erzeugen. Dieser Zustand wird als *Unterbeweglichkeit* bezeichnet. Ein analoges Kriterium gilt für die JACOBI-Matrix $\underline{\underline{J}}^{\mathrm{IK}}$ der inversen Kinematik. Verschwindet die Determinante $\det \underline{\underline{J}}^{\mathrm{IK}}$, befindet sich der Roboter ebenfalls in einer singulären Stellung, die als *Überbeweglichkeit* (engl. Self-Motion) bezeichnet wird: hier ist es möglich, die Plattform bei festgehaltenen verallgemeinerten Koordinaten $\underline{q}$ zu bewegen.

Die Prüfung auf singuläre Stellungen im Arbeitsraum kann durch die Entwicklung der Determinante von $\underline{\underline{J}}^{\mathrm{IK}}$ geschehen. Da die Determinante eine kontinuierliche Funktion in den Koeffizienten von $\underline{\underline{J}}^{\mathrm{IK}}$ ist, existiert genau dann eine Singularität im Arbeitsraum, wenn zwei verschiedene Punkte $\underline{y}_1, \underline{y}_2$ verschiedene Vorzeichen der Determinante aufweisen. Ohne Beschränkung der Allgemeinheit kann daher gefordert werden, dass für die Determinante an jeder Stelle des Arbeitsraums $\det \underline{\underline{J}}^{\mathrm{IK}} > 0$ gelten muss. Diese Bedingung kann prinzipiell als Bedingung für den CSP-Löser verwendet werden. In der Praxis empfiehlt es sich jedoch, spezialisierte Verfahren zur Untersuchung von Intervallmatrizen zu verwenden, da die allgemeinen Verfahren eine sehr lange Rechenzeit erfordern [114]. Ein für die besonderen Erfordernisse von PKM zugeschnittenes Verfahren wurde von Merlet vorgeschlagen [83].

Neben den algebraischen Methoden existieren auch geometrische Verfahren zum Auffinden von singulären Konfigurationen. Eine vollständige Klassifikation wurde von Merlet [74] vorgenommen. Dabei werden die Beine durch Linienkoordinaten (PLÜCKER-Koordinaten) beschrieben und mithilfe der GRASSMANN-Geometrie untersucht. Singuläre Konfigurationen treten genau dann auf, wenn diese Linienkoordinaten in bestimmten geometrischen Verhältnissen zueinander liegen [74].

4.4.4 Manipulierbarer Arbeitsraum

Im vorangegangenen Abschnitt wurde der Begriff der Singularität als qualitative kinematische Eigenschaft eines Manipulators eingeführt und diskutiert. Eine quantitative Bewertung der Beweglichkeit der Plattform wird durch einen sogenannten *Manipulierbarkeitsindex* (engl. Dexterity Index) vorgenommen. Die Grundidee bei der Definition von Manipulierbarkeitsindizes besteht darin, die Übersetzung zwischen Aktuatoren (Eingängen) und den Weltkoordinaten (Ausgängen) zu beschreiben. Die JACOBI-Matrix beinhaltet wesentliche Informationen zur Beurteilung der Manipulierbarkeit, hat jedoch den Nachteil, dass

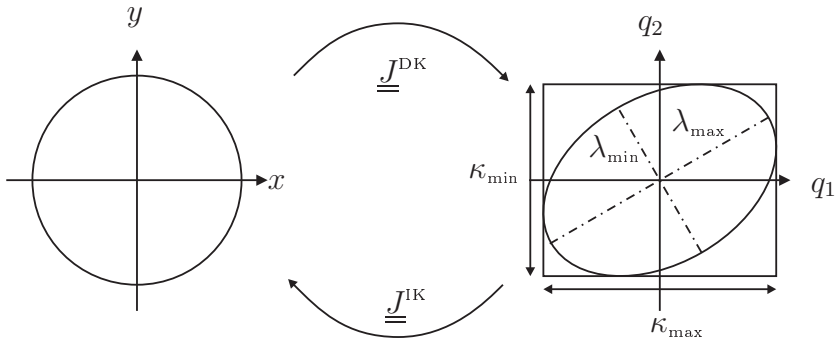


Abbildung 4.8: Manipulierbarkeit als Maß für Verzerrung der Abbildung zwischen Weltkoordinaten (x, y) und Gelenkkoordinaten (q_1, q_2)

sich Matrizen für verschiedene Posen unzureichend vergleichen lassen. Wie im vorangegangenen Abschnitt diskutiert wurde, ist bereits die Suche nach singulären Konfigurationen für räumliche Roboter nichttrivial. Ein Manipulierbarkeitsindex ist eine skalare Größe, die eine Vergleichbarkeit verschiedener Posen und Roboterdesigns gewährleisten soll. Zu diesem Zweck wurden in der Literatur zahlreiche Möglichkeiten vorgeschlagen [111, 123]. Die wichtigsten Manipulierbarkeitsindizes sind die Determinanten $\det \underline{\underline{J}}, \det(\underline{\underline{J}}\underline{\underline{J}}^{\mathrm{T}}), \det(\underline{\underline{J}}^T \underline{\underline{J}})$ [141], der kleinste Eigenwert $\lambda_{\min}$ von $\underline{\underline{J}}$ und die Konditionszahl κ von $\underline{\underline{J}}$ [34]. Für Roboter, die sowohl rotatorische als auch translatorische Freiheitsgrade besitzen, entsteht bei der Definition von Manipulierbarkeitsindizes das Problem, Verschiebungen und Drehungen ineinander umzurechnen. Dies geschieht durch die Einführung einer *Metrik*, welche die verschiedenen physikalischen Dimensionen homogenisiert. Gleichzeitig werden durch die Metrik willkürliche Koeffizienten eingeführt und es wurde gezeigt, dass das Konzept der Manipulierbarkeit daher weder invariant noch natürlich ist [123]. Im mathematischen Sinne kann durch einen Manipulierbarkeitsindex mithin nicht der Abstand von Singularitäten ermittelt werden. In dieser Arbeit wird daher geprüft, ob innerhalb eines gegebenen Arbeitsraums gegebene untere und obere Werte für die Manipulierbarkeit nicht über- bzw. unterschritten werden.

Im Zusammenhang mit Manipulierbarkeitsindizes ist es wichtig anzumerken, dass viele dieser Indizes durch eine Änderung des Bezugspunkts auf der Plattform (vgl. Abschnitt 3.3.2) oder durch ein Getriebe beeinflusst werden können. Das Problem der Manipulierbarkeit wird hier vor dem Hintergrund einer Werkzeugmaschine betrachtet, die eine bekannte Aufgabe ausführen muss, d. h. die für den Prozess nötigen maximalen Geschwindigkeiten des TCP sind bekannt. Wird nun die Geschwindigkeit des End-Effektors vorgegeben, können die Aktuatoren ihre maximale Geschwindigkeit unabhängig voneinander erreichen. Es muss daher im gesamten Arbeitsraum geprüft werden, ob jede gewünschte Geschwindigkeit am TCP durch die Antriebe realisiert werden kann. Damit die Geschwindigkeit $\underline{\dot{q}}_{\max}$ der Antriebe nicht überschritten wird, muss stets

$$\underline{\dot{q}}_{\max} > \underline{\underline{J}}\,\underline{\dot{y}} \tag{4.29}$$

gelten. Eine Abschätzung kann durch Normen erfolgen, wobei hier im Gegensatz zu den oben genannten Beispielen für die Manipulierbarkeit nicht notwendigerweise die euklidische Norm verwendet werden muss. Allgemein folgt unter Verwendung verträglicher Normen für die Vektoren und Matrizen

$$||\underline{\dot{q}}_{\max}|| = ||\underline{\underline{J}}\;\underline{\dot{y}}|| \leq ||\underline{\underline{J}}||\;||\underline{\dot{y}}||. \tag{4.30}$$

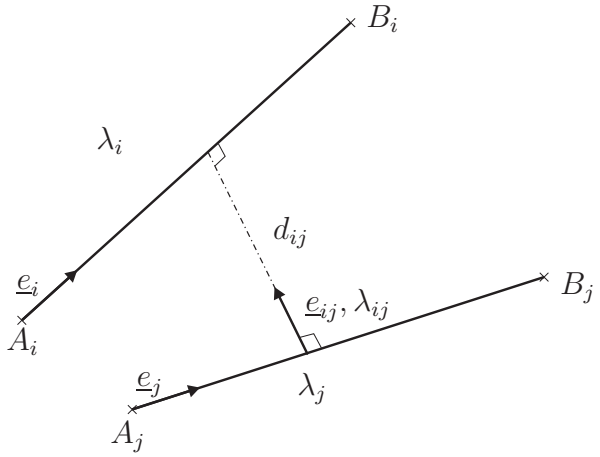


Abbildung 4.9: Abstand zwischen zwei Liniensegmenten

Für die Unendlich-Norm $||\cdot||_\infty$ ergibt sich für einen beliebigen Vektor $\underline{v} \in \mathbb{R}^n$ und eine beliebige Matrix $\underline{\underline{A}} \in \mathbb{R}^{n \times n}$

$$||\underline{v}||_\infty = \max_{1 \le i \le n} |v_i|, \tag{4.31}$$

$$||\underline{\underline{A}}||_\infty = \max_{1 \le i \le n} \sum_{j=1}^{n} |a_{ij}|. \tag{4.32}$$

Für eine dimensionslose Abschätzung wird die normalisierte Geschwindigkeit $\underline{\dot{y}}$ in Weltkoordinaten betrachtet. Um dieses Kriterium als Teil des Systems $\underline{\Phi} > \underline{0}$ zu behandeln, werden alle Argumente des max-Operators geprüft, so dass sich die einfachen Gleichungen

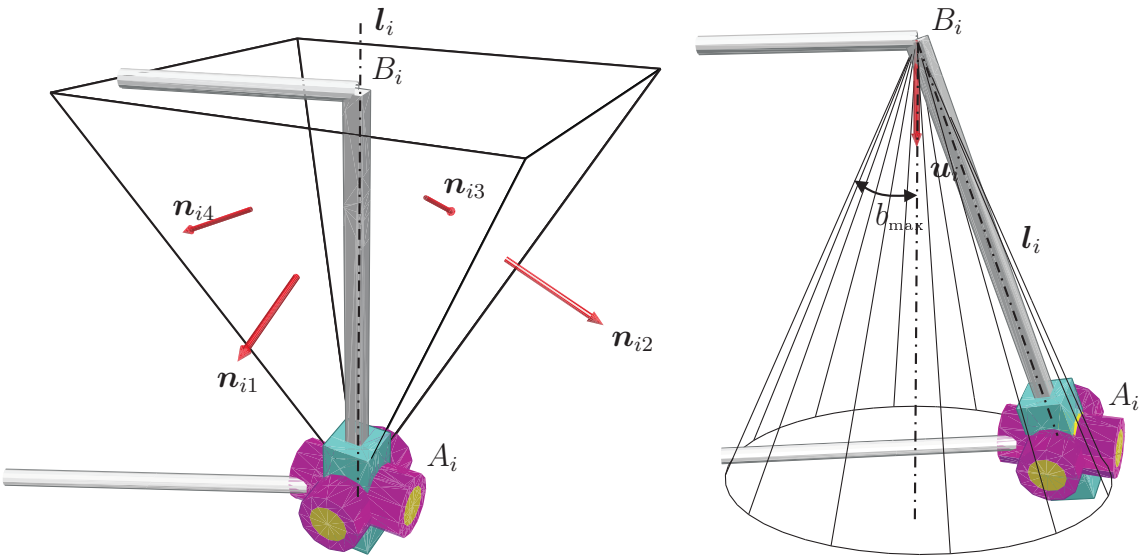
$$\sum_{k=1}^{6} |[\underline{\underline{J}}]_{ik}| < j_{\max} \quad \forall\ 1 \le i \le 6 \tag{4.33}$$

ergeben. Damit kann garantiert werden, dass bei einem bekannten Fertigungsprozess die zulässigen Geschwindigkeiten der Aktuatoren eingehalten werden.

4.4.5 Stabkollisionen

Bei PKM kann der Arbeitsraum durch Autokollisionen erheblich beeinträchtigt werden [78]. Entsprechend muss daher sichergestellt werden, dass solche Beeinträchtigungen nicht auftreten. Autokollisionen treten immer dann auf, wenn sich die beweglichen Teile der Maschine überschneiden können. Die relevanten Paarungen für die Kollision bilden Bein/Bein, Bein/Rahmen und Bein/Plattform. Die Kollision zwischen zwei Beinen wird in diesem Abschnitt näher beleuchtet, während die beiden anderen Typen durch eine geeignete Beschränkung der passiven Gelenke berücksichtigt werden kann.

Eine verbreitete Modellierung zur Untersuchung der Stabkollision besteht darin, die Stäbe als Zylinder zu beschreiben und die Abstände zwischen den Zylindern zu bestimmen (Abb. 4.9). Sind die Anlenkpunkte A_i, B_i, A_j, B_j bekannt, kann diese Berechnung relativ leicht durchgeführt werden, obwohl eine große Anzahl von Spezialfällen unterschieden werden muss, z. B. wenn die Stäbe parallel sind. Die zahlreichen Fallunterscheidungen führen jedoch zu Gleichungen, die sich nicht direkt in den Formalismus des CSP-Lösers



a) Kardangelenk b) Kugelgelenk

Abbildung 4.10: Begrenzung des Arbeitsraums durch die maximalen Gelenkwinkel

einfügen lassen. Stattdessen werden hier Hilfsvariablen eingeführt, die zu der einfachen Formulierung des Abstands zwischen zwei Stäben

$$g_i : \underline{v}_i = \underline{a}_i + \lambda_i(\underline{b}_i - \underline{a}_i) \quad \text{für} \quad \lambda_i \in [0,1],\ i = 1,\ldots,6 \tag{4.34}$$

führen, wobei hier das Supremum $d_{\min}$ von

$$d_{ij} = |\underline{v}_i - \underline{v}_j| \quad \forall\ \lambda_i, \lambda_j \in [0,1], i \neq j \tag{4.35}$$

gesucht wird. Durch die Einführung der Intervallvariablen $\lambda_i, \lambda_j \in \mathbb{I}$ lässt sich die komplexe Fallunterscheidung durch eine Intervallauswertung einer einfachen Betragsfunktion ersetzen und es ergibt sich

$$d_{\min} < d_{ij} = |\underline{a}_i + \lambda_i(\underline{b}_i - \underline{a}_i) - \underline{a}_j - \lambda_j(\underline{b}_j - \underline{a}_j)| \tag{4.36}$$

mit $1 \leq i \leq 6, 1 \leq j \leq 6, i \neq j$ als Ungleichungen. Somit lassen sich Stabkollisionen ebenfalls als Teil des Systems $\underline{\Phi} > 0$ berücksichtigen.

4.4.6 Passive Gelenke

Im Gegensatz zu seriellen Robotern verfügen PKM über *passive Gelenke*, welche nicht durch Aktuatoren angetrieben werden. Die Mehrzahl der verwendeten Gelenke sind aufgrund ihrer technischen Realisierung in ihrer Bewegung eingeschränkt. Schubgelenke haben z. B. nur eine endliche Länge, während bei Kugel- und Kardangelenken bestimmte Schwenkbereiche nicht überschritten werden können. Drehgelenke lassen sich prinzipiell so aufbauen, dass ein Bewegungsbereich von 360° erreicht werden kann, jedoch sind viele praktisch verwendete Gelenke ebenfalls in ihrer Bewegung beschränkt. Die Begrenzung der Gelenke lässt sich analog auf alle Beinkinematiken anwenden und die Begrenzungen müssen bei der Berechnung des Arbeitsraums berücksichtigt werden. Je nach Typ des

Gelenks schränken dessen Begrenzungen den Arbeitsraum bezüglich Größe und Gestalt erheblich ein. Die Behandlung von Dreh- und Schubgelenken mit einem Freiheitsgrad gestaltet sich weitgehend trivial. Anhand der Kinematik lassen sich für eine Pose $\underline{y}$ die erforderlichen Gelenkkoordinaten $\beta_i(\underline{y})$ bestimmen. Aus der Geometrie der Maschine sind andererseits die maximalen zulässigen Gelenkkoordinaten $\beta_i^{\min}, \beta_i^{\max}$ bekannt, so dass sich unmittelbar die Bewertung anhand von

$$\beta_i^{\min} \leq \beta_i(\underline{y}) \leq \beta_i^{\max} \quad \text{für} \quad i = 1, \ldots, n_g \tag{4.37}$$

vornehmen lässt. Bei Kugel- und Kardangelenken kann eine direkte geometrische Betrachtung gewählt werden. Für Kardangelenke lässt sich der Bereich der möglich Schwenkwinkel durch eine vierseitige Pyramide beschreiben (Abb. 4.10), deren Normalenvektoren $\underline{n}_1, \underline{n}_2, \underline{n}_3, \underline{n}_4$ aus der Geometrie des Gelenks bekannt sind [77, p. 205f]. Dies führt auf die einfachen Bedingungen

$$-\underline{l}_i \cdot \underline{n}_{ij} > 0 \quad \text{für} \quad i = 1, \ldots, 6, \quad j = 1, \ldots, 4, \tag{4.38}$$

wobei $\underline{n}_{ij}$ die vier normalen Vektoren für jedes der sechs Beine sind. Bei Kugelgelenken wird der Bewegungsbereich dagegen durch einen Kegel beschrieben, dessen Achse $\underline{u}_i$ und Öffnungswinkel $\beta_{\max}$ bekannt sind. Damit ergibt sich die Bedingung

$$\underline{l}_i \cdot \underline{u}_i - l_i \cos \beta_{\max} > 0 \quad \text{für} \quad i = 1, \ldots, 6. \tag{4.39}$$

Sowohl für Kugel- als auch für Kardangelenke muss dabei beachtet werden, dass sich die Normalenvektoren $\underline{n}_i, \underline{u}$ auf das Koordinatensystem des Gelenks beziehen. Werden also die plattformseitigen passiven Gelenke betrachtet, müssen die Normalenvektoren zunächst mit der Matrix $\underline{\underline{R}}$ in Weltkoordinaten umgerechnet werden. Für die Kugelgelenke, die meist an der bewegten Plattform verwendet werden, ergibt sich beispielsweise

$$\underline{l}_i \cdot \underline{\underline{R}}\, \underline{u}_i - l_i \cos \beta_{\max} > 0 \quad \text{für} \quad i = 1, \ldots, 6. \tag{4.40}$$

4.4.7 Automatische Codegenerierung

Beim Aufstellen der Bindungen treten zum Teil sehr umfangreiche algebraische Ausdrücke auf, die sich effizient mit einem Computeralgebra-System (CAS) behandeln lassen. Innerhalb eines CAS ist es meist nicht möglich, Intervallauswertungen von Funktionen effizient durchzuführen. Dies begründet sich darin, dass die Intervallanalyse nur unzureichend für diese Programme implementiert wurde. Selbst wenn eine Erweiterung für die Intervallanalyse vorliegt, besitzt diese nur eine vergleichsweise geringe Effizienz, da ein CAS zur Auswertung einer Funktion einen *Interpreter* benutzt, um den symbolischen Ausdruck zu berechnen. Im Gegensatz dazu bieten Hochsprachen wie z. B. C++ eine übersichtliche Möglichkeit, neue Datentypen wie die hier benötigten Intervalle einzuführen, indem sie sich des Konzepts der Überladung von Operatoren bedienen [125]. Die Lücke zwischen der einfachen Formelmanipulation eines CAS und der effizienteren Numerik einer Hochsprache lässt sich durch einen automatischen Quellcode-Generator schließen, der die symbolischen Ausdrücke des CAS in eine Hochsprache übersetzt. Während dieser Übersetzung können mehrfach auftretende Terme zusammengefasst und dahingehend optimiert werden. Im Fall der Intervallanalyse kann dieser Quellcode-Generator auch kleine Anpassungen

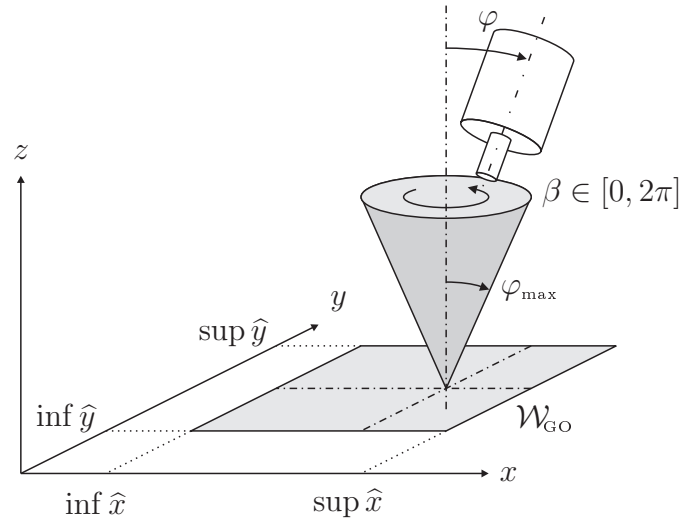


Abbildung 4.11: Parametrierung des Gesamtorientierungsarbeitsraums $\mathcal{W}_{\mathrm{GO}}$

vornehmen, um die Ausdrücke auf die besonderen Erfordernisse der Intervallanalyse vorzubereiten (siehe Abschnitt 2.3). Für diese Arbeit wurde der Quellcode-Generator von ALIAS/Maple verwendet, der für die Intervallanalyse optimiert wurde [21]. Darüber hinaus ermöglicht der Compiler einer Hochsprache gegenüber einem Interpreter spezifische Optimierungen des Codes für die schnelle Ausführung auf einem bestimmten Prozessor. Dadurch wird die Ausführungsgeschwindigkeit insgesamt im Vergleich zum CAS um drei Größenordnungen verbessert.

4.5 Ergebnisse

In den folgenden Abschnitten werden beispielhaft Ergebnisse für die Berechnung des Arbeitsraums verschiedener PKM angegeben, wobei sowohl nach dem Typ der Kinematik als auch nach den untersuchten Eigenschaften des Arbeitsraums unterschieden wird. Das System $\underline{\Phi}$ wird durch Einsetzen der kinematischen Gleichungen (Kapitel 2) in die Bindungen (Abschnitt 4.4) zusammengestellt. Die resultierenden Gleichungen werden dann mit einem CAS aufbereitet und in C++-Code übersetzt. Bei der Untersuchung wird die Rechenzeit des CSP-Lösers, die Genauigkeit der Ergebnisse und der Einfluss der unterschiedlichen Kriterien auf Größe und Gestalt des Arbeitsraums untersucht.

4.5.1 Linapod

Als erstes Beispiel für die Berechnung des Arbeitsraums einer PKM wird der Prototyp des *Linapod* betrachtet [108, 139, 140]. Dabei werden die nominellen Parameter des *Linapod* für die Geometrie, gemäß Tabelle 2.4, S. 29 zugrunde gelegt. Aufgrund der parallelen Anordnung der Schubgelenke ist der Arbeitsraum invariant gegenüber einer Verschiebung entlang der gemeinsamen z-Achse der Schubgelenke.[7] Weiterhin wird für die Orientierung

[7] Der Arbeitsraum hängt auch von der Länge der Führungsschienen der Schubgelenke ab, jedoch wird diese Begrenzung bei der Betrachtung des xy-Querschnitts vernachlässigt.

Tabelle 4.1: Parameter und Rechenzeit für die Berechnung des Arbeitsraums mit CSP-Löser am Beispiel des *Linapod*

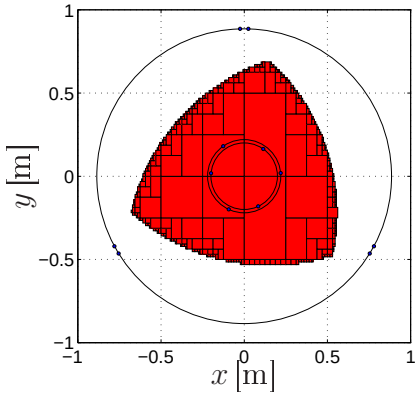
Abb.	Kriterien	$\dim \underline{c}$	$\dim \underline{v}$	ε_c	ε_v	$\#\mathcal{L}_\mathrm{S}$	$\#\mathcal{L}_\mathrm{I}$	$\#\mathcal{L}_\mathrm{F}$	Rechenzeit
4.12a	Beinlängen	2	0	0.01	–	317	321	962	0.6 s
4.12b	Beinlängen	2	2	0.01	0.01	204	345	6838	17.8 s
4.13a	Beinlängen	2	2	0.1	0.01	23	43	243	0.3 s
4.13b	Beinlängen	2	2	0.05	0.005	45	84	622	1.4 s
4.13c	Beinlängen	2	2	0.01	0.001	240	319	5417	33.6 s
4.13d	Beinlängen	2	2	0.005	0.0005	500	629	18942	157.8 s
4.15a	Manipulierbarkeit	2	0	0.01	–	263	825	1782	0.2 s
4.15b	Manipulierbarkeit	2	2	0.01	0.01	232	835	4160	2.2 s
4.16a	Stabkollision	2	2	0.01	0.01	168	321	13878	23.0 s
4.16b	Stabkollision	2	4	0.01	0.01	105	424	10085	1580.0 s

die Transformationsmatrix nach Gleichung (2.35) verwendet, so dass die Anzahl der Drehparameter auf zwei reduziert wird. Damit verbleiben die vier relevante Weltkoordinaten $\underline{y} = [x, y, \varphi, \beta]^\mathrm{T}$, welche den Gesamtorientierungsarbeitsraum aufspannen (Abb. 4.11).

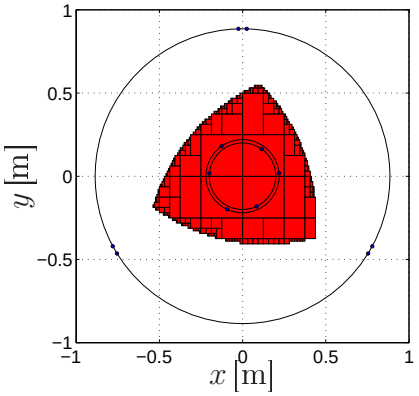
Mit dem CSP-Löser kann nun untersucht werden, welche Posen $\underline{y}$ die unterschiedlichen Bindungen erfüllen und daher zum Arbeitsraum der Maschine gehören. Bei den folgenden Berechnungen wurde der Suchraum $\mathcal{X}_s = \{\underline{x} \in [-1, 1], \underline{y} \in [-1, 1]\}$ angenommen. Die verwendeten Parameter, die Rechenzeit und die Anzahl der Elemente in den Listen $\mathcal{L}_\mathrm{S}$,$\mathcal{L}_\mathrm{I}$,$\mathcal{L}_\mathrm{F}$ des Algorithmus sind in Tabelle 4.1 aufgeführt. In den Diagrammen des Arbeitsraums (Abb. 4.12–4.16) ist die Geometrie der Maschine schematisch eingezeichnet. Der äußere Kreis im Diagramm deutet den Umkreis des Maschinenrahmens mit den rahmenseitigen Anlenkpunkten (schwarze Punkte) an. Die beiden inneren Kreise repräsentieren die Plattform und deren Anlenkpunkte.

Zunächst wird der erreichbare translatorische Arbeitsraum $\mathcal{W}_\mathrm{T}$ für eine konstante Orientierung der Plattform $\underline{\underline{R}}(\varphi = 0, \beta = 0) = \underline{\underline{I}}_3$ untersucht. Für den CSP-Löser wurden daher als Berechnungsvariablen $\underline{c} = [x, y]^\mathrm{T}$ gewählt, und die Lösungsmenge wurde mit dem Algorithmus `Berechnen` ermittelt. Die Laufzeit, Parameter und Ergebnisse sind in Tabelle 4.1 zusammengefasst. In der Abbildung 4.12a ist der Querschnitt des Arbeitsraums aufgetragen, wobei der Arbeitsraum für die nominellen Parameter der Maschine durch die Länge der unteren Stäbe limitiert wird.

Der erreichbare Gesamtorientierungsarbeitsraum $\mathcal{W}_\mathrm{GO}$ enthält für jede Position $[x, y]^\mathrm{T}$ alle Orientierungen $[\beta, \varphi]^\mathrm{T}$ der Plattform aus einer vorgegebenen Menge $\mathcal{X}_v$. Für Werkzeugmaschinen ist wichtig, dass sich das Werkzeug um *jede* Achse in der xy-Ebene um einen gegebenen Winkel φ_max schwenken lässt. Daher wird die Menge der Orientierungen durch die Verifikationsvariablen $\underline{v} = [\beta, \varphi]^\mathrm{T}$ beschrieben. Der Winkel β beschreibt die Richtung für das Schwenken des Werkzeugs und φ bezeichnet den Winkel, um den das Werkzeug ausschwenkt. Um ein Schwenken um den maximalen Winkel $\varphi_\mathrm{max} = 5.7°$ zu ermöglichen, wird die Verifikationsmenge zu $\mathcal{X}_v = \{\varphi = [0, \varphi_\mathrm{max}], \beta = [0, 2\pi]\}$ gesetzt. Daher wurde der hybride CSP-Löser für die Berechnung des Gesamtorientierungsarbeitsraums verwendet (Abb. 4.12b).



a) konstante Orientierung $\mathcal{W}_{\mathrm{T}}$



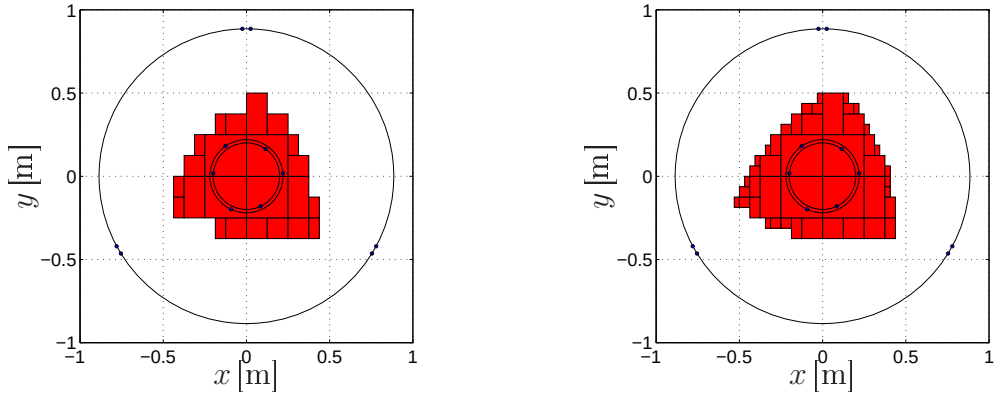
b) variable Orientierung $\mathcal{W}_{\mathrm{GO}}$ ($\varphi < 5.7°$)

Abbildung 4.12: Kinematisch erreichbarer Arbeitsraum des *Linapod* unter Berücksichtigung der Beinlängen

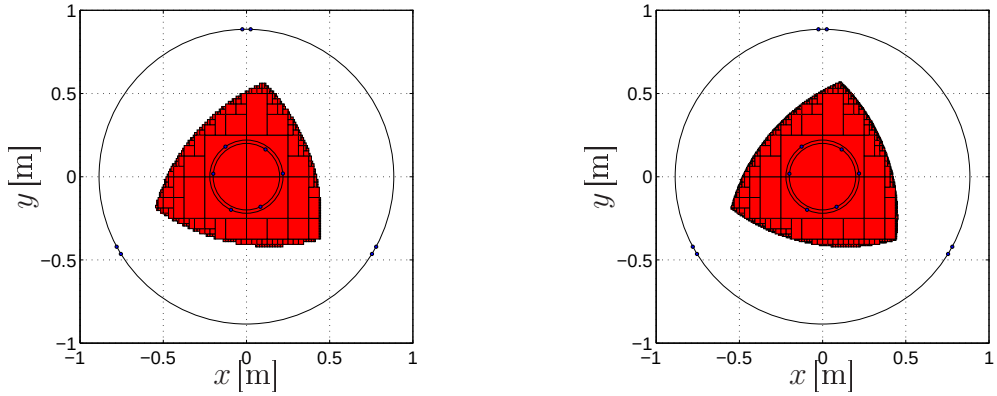
Der Einfluss der Genauigkeit ε_c bei der Berechnung des Gesamtorientierungsarbeitsraums ist in Abbildung 4.13 dargestellt. Hier wurde ε_c zwischen 0.1 und 0.005 variiert, um Rechenzeit und Speicherbedarf zu vergleichen. Abbildung 4.13 gibt einen qualitativen Eindruck des Einflusses von ε_c und Tabelle 4.1 kann die Laufzeit und der benötigte Speicherplatz entnommen werden. Ein graphischer Vergleich von Genauigkeit, Laufzeit und Speicherbedarf ist in Abbildung 4.14 doppelt logarithmisch dargestellt. Der lineare Verlauf bei doppelt logarithmischer Auftragung lässt erkennen, dass sowohl der Speicherbedarf als auch die Rechenzeit für kleinere Werten von ε_c polynominal steigen. Der Grad des Polynoms und damit die Ordnung des Verfahrens hängt von der Anzahl der betrachteten Variablen $\underline{c}, \underline{v}$ ab.

Als manipulierbarer Arbeitsraum (Abschnitt 4.4.4) wird der Teil des erreichbaren Arbeitsraums bezeichnet, in dem die Übersetzungsverhältnisse zwischen der Geschwindigkeit der Plattform und der Geschwindigkeit in den Antrieben einen vorgegebenen Grenzwert nicht übersteigen. Im folgenden Beispiel wurde das maximale Übersetzungsverhältnis mit $j_{\max} = 4$ vorgegeben, um den translatorischen Arbeitsraum und den Gesamtorientierungsarbeitsraum zu berechnen. Die Variablen $\underline{c} = [x, y], \underline{v} = [\varphi, \beta]$ und Mengen $\mathcal{X}_s, \mathcal{X}_v$ wurden daher wie oben gewählt. Die übrigen Parameter sind in Tabelle 4.1 angegeben. Die so berechneten Arbeitsräume $\mathcal{W}_{\mathrm{T}}, \mathcal{W}_{\mathrm{GO}}$ sind in Abbildung 4.15 dargestellt. Bemerkenswert ist die leichte Asymmetrie des Arbeitsraums rechts unten, die aus der Abschätzung durch die Unendlichkeitsnorm resultiert.

Das letzte Beispiel demonstriert den Einfluss der Stabkollisionen auf Größe und Gestalt des translatorischen Arbeitsraums und des Gesamtorientierungsarbeitsraums. Bei der Betrachtung der Stabkollisionen müssen, wie in Abschnitt 4.4.5 beschrieben, Hilfsvariablen λ_i eingeführt werden, um die der Vektor der Verifikationsvariablen erweitert wird. Zur effizienten Berechnung wird die Verifikationsmenge separiert, so dass sich für jede betrachtete Beinpaarung eine eigene Teilaufgabe ergibt. Für die nominelle Geometrie des *Linapod* werden die Beinpaarungen (1,4), (2,5) und (3,6) berücksichtigt. Für die Berechnung des translatorischen Arbeitsraums wird beispielsweise für die Beinpaarung (1,4) der Vektor $\underline{c}^{(1)} = [\lambda_1, \lambda_4]^{\mathrm{T}}$ und die Menge $\mathcal{X}_v^{(1)} = \{\lambda_1 \in [0,1], \lambda_4 \in [0,1]\}$ gesetzt. Entsprechend werden $\underline{c}^{(2)}, \underline{c}^{(3)}$ und $\mathcal{X}_v^{(2)}, \mathcal{X}_v^{(3)}$ gewählt. Damit ergibt sich für einen



a) $\varepsilon_c = 0.1\,\mathrm{m}$ b) $\varepsilon_c = 0.05\,\mathrm{m}$



c) $\varepsilon_c = 0.01\,\mathrm{m}$ d) $\varepsilon_c = 0.005\,\mathrm{m}$

Abbildung 4.13: Gesamtorientierungsarbeitsraum $\mathcal{W}_{\mathrm{GO}}$ für verschiedene Genauigkeiten ε_c

Stabdurchmesser von $d_l = 0.12\,\mathrm{m}$ der in Abbildung 4.16a gezeigte Arbeitsraum. Für die Bestimmung des Gesamtorientierungsarbeitsraums müssen die Winkel φ, β und die Hilfsvariablen λ_i gemeinsam betrachtet werden. Daher wird für die erste Beinpaarung (1,4) der Vektor $\underline{c}^{(1)} = [\varphi, \beta, \lambda_1, \lambda_4]^{\mathrm{T}}$ gesetzt und die Verifikationsmenge zu $\mathcal{X}_v^{(1)} = \{\varphi \in [0, \varphi_{\max}], \beta \in [0, 2\pi], \lambda_1 \in [0, 1], \lambda_4 \in [0, 1]\}$ gewählt. Der kollisionsfreie Arbeitsraum ist in Abbildung 4.16b visualisiert. Die Größe des Arbeitsraums hängt stark vom Durchmesser d_l der Stäbe ab. In Abbildung 4.17 wird der translatorische Arbeitsraum für verschiedene Durchmesser d_l verglichen. Es ist deutlich zu erkennen, dass der Arbeitsraum für größere Stabdurchmesser stark abnimmt. Folglich ist bei gegebenem Durchmesser $d_l = 0.12\,\mathrm{m}$ die Stabkollision ein entscheidendes Kriterium für die Größe des Arbeitsraums.

4.5.2 Gough-Plattform

Für die Gough-Plattform mit SSM-Parametrisierung wird im Folgenden der Arbeitsraum auf eine ähnliche Weise berechnet wie zuvor für die Maschine *Linapod*. Zunächst wird eine generische Geometrie mit den nominellen Parametern nach Tabelle 4.2 (S. 93) untersucht. Im Anschluss daran wird die Werkzeugmaschine *PaLiDA* (Mechatronik-Zentrum Hannover) und ein Fahrsimulator (DLR Braunschweig) mit den hier vorgestellten Methoden analysiert.

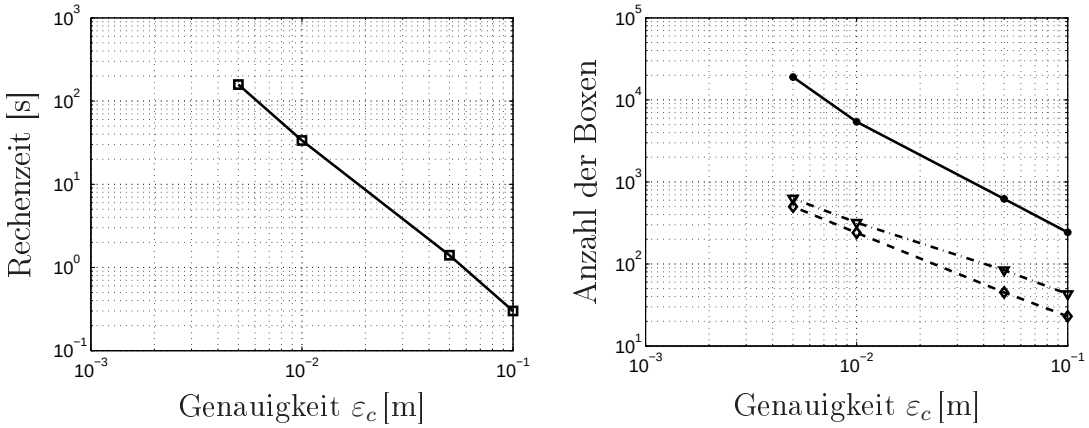
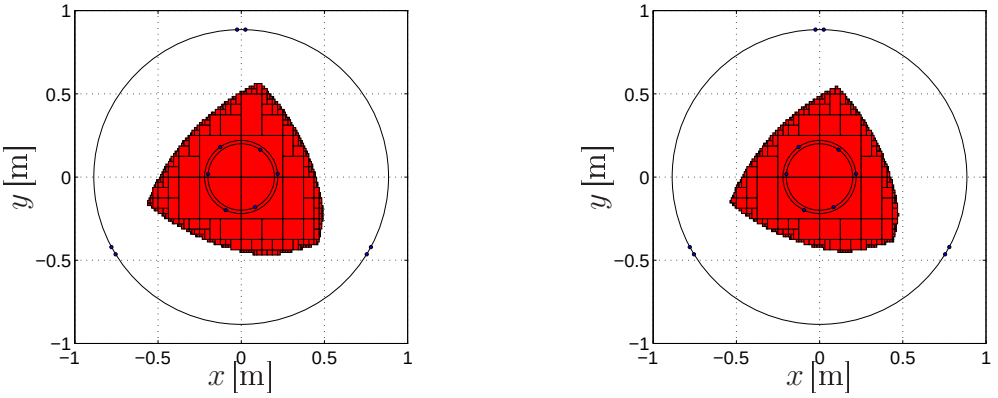


Abbildung 4.14: Rechenzeit und Anzahl der Boxen in den Listen $\mathcal{L}_\mathrm{S}(\diamond)$, $\mathcal{L}_\mathrm{I}(\triangleleft)$, $\mathcal{L}_\mathrm{F}(\bullet)$ in Abhängigkeit der Genauigkeit ε_c bei der Berechnung des Gesamtorientierungsarbeitsraums des *Linapod*

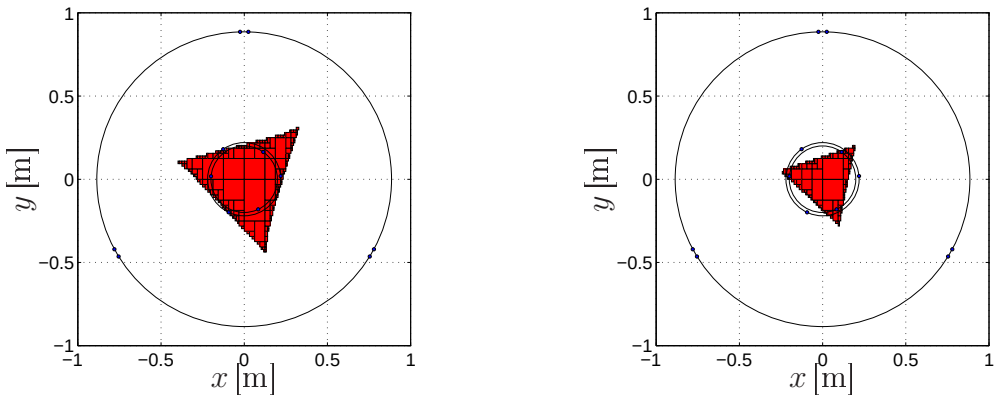


a) konstante Orientierung $\mathcal{W}_\mathrm{T}$ b) variable Orientierung $\mathcal{W}_\mathrm{GO}$ ($\varphi < 5.7°$)

Abbildung 4.15: Arbeitsraum des *Linapod* unter Berücksichtigung der Manipulierbarkeit

Der Arbeitsraum des SSM wird mithilfe des CSP-Lösers berechnet. In Abbildung 4.18a ist der translatorische Arbeitsraum $\mathcal{W}_\mathrm{T}$ abgebildet ($\underline{\underline{R}}_\mathrm{TCP} = \underline{\underline{I}}_3$). Dazu wurden für den Vektor der Berechnungsvariablen $\underline{c} = [x, y, z]^\mathrm{T}$ die kartesischen Koordinaten der Plattform verwendet, während für die Verifikationsvariablen die leere Menge verwendet wurde $\underline{v} = \emptyset$. Dagegen wurde der Vektor der Verifikationsvariablen $\underline{v} = [\varphi, \theta, \psi]^\mathrm{T}$ beim Gesamtorientierungsarbeitsraum $\mathcal{W}_\mathrm{GO}$ mit den KARDAN-Winkeln der Plattform initialisiert und die Verifikationsmenge zu $\mathcal{X}_v = \{\varphi \in [-0.1, 0.1], \theta \in [-0.1, 0.1], \psi \in [-0.1, 0.1]\}$ gesetzt. Für jede in Abbildung 4.18b dargestellte Position sind daher alle Orientierungen $\underline{v} \in \mathcal{X}_v$ möglich. Die Genauigkeit für die Berechnung wurde auf $\varepsilon = 0.025$ eingestellt, und die Berechnungszeit beträgt 1.7 s für den translatorischen Arbeitsraum $\mathcal{W}_\mathrm{T}$ und 41 s für den Gesamtorientierungsarbeitsraum $\mathcal{W}_\mathrm{GO}$.

Die Versuchsträger *PaLiDA* des Mechatronik-Zentrums Hannover wurde von Tönshoff et al. [129] beschrieben und ist in Abbildung 4.19 dargestellt. Die nominellen Parameter des *PaLiDA* sind in Tabelle 4.3 angegeben. Der translatorische Arbeitsraum und der Gesamtorientierungsarbeitsraum wurden mit dem CSP-Löser berechnet (Abb. 4.20). Ein



a) konstante Orientierung $\mathcal{W}_{\mathrm{T}}$ b) variable Orientierung $\mathcal{W}_{\mathrm{GO}}$ ($\varphi < 5.7°$)

Abbildung 4.16: Arbeitsraum des *Linapod* unter Berücksichtigung der Stabkollisionen

Tabelle 4.2: Nominelle geometrische Parameter des SSM

Parameter	Wert	Beschreibung
r_b	1 m	Radius des Rahmens
r_p	0.25 m	Radius der beweglichen Plattform
$\Delta\alpha_b$	45°	Winkel Anordnung der rahmenseitigen Anlenkpunkte
$\Delta\alpha_p$	15°	Winkel Anordnung der plattformseitigen Anlenkpunkte
$l_{\min}$	1 m	minimale Länge der Antriebs
$l_{\max}$	2 m	maximale Länge der Antriebs
$\Delta\beta_{\max}$	45°	maximaler Schwenkwinkel des Kugelgelenks

Vergleich mit den Untersuchungen des Arbeitsraums am Mechatronik-Zentrum Hannover zeigt eine gute Übereinstimmung der Ergebnisse.

In Abbildung 4.21 ist der Fahrsimulator des Deutschen Zentrums für Luft- und Raumfahrt (DLR) in Braunschweig abgebildet [11, 126]. Dieser Manipulator entspricht der Geometrie eines SSM mit den in Tabelle 4.4 angegebenen Parametern. Die Berechnung des Arbeitsraums konnte daher direkt anhand des Modells für SSM durchgeführt werden, und es ergibt sich der in Abbildung 4.22 dargestellte Arbeitsraum. Diese Ergebnisse liefern eine gute Übereinstimmung mit den am Simulator ermittelten Werten.

4.5.3 Parallelisierung des CSP-Lösers

Die Effizienz und Skalierbarkeit der parallelen Implementierung des CSP-Lösers wurde auf dem heterogenen Computer-Cluster des Lehrstuhls für Mechatronik getestet. An dem Test waren insgesamt 19 PCs beteiligt, wobei einer dieser Computer als Master fungierte, während die übrigen 18 PCs als Slaves konfiguriert waren. Als Protokoll wurde das verbreitete *Message Passing Interface* (MPI) verwendet [84], das in einer speziellen Implementierung für Microsoft Windows basierte PCs vorliegt [17]. Der Cluster war bezüglich drei verschiedener Punkte heterogen: die PCs sind mit verschiedenen Betriebssystemen ausgestattet (Microsoft Windows 2000 und XP), die Netzwerkverbindung erfolgt über 100 MBit Twisted-Pair-Ethernet sowie über Lichtwellenleiter-Ethernet. Schließlich

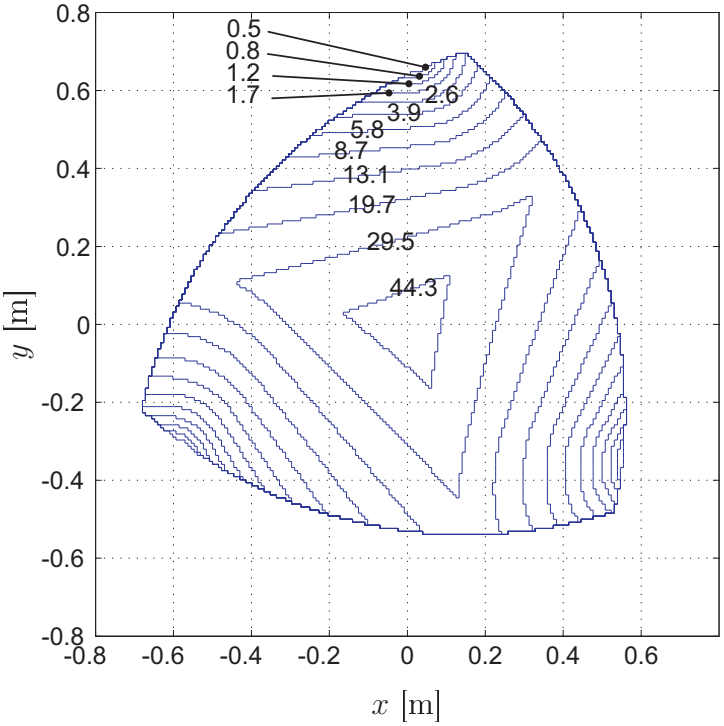
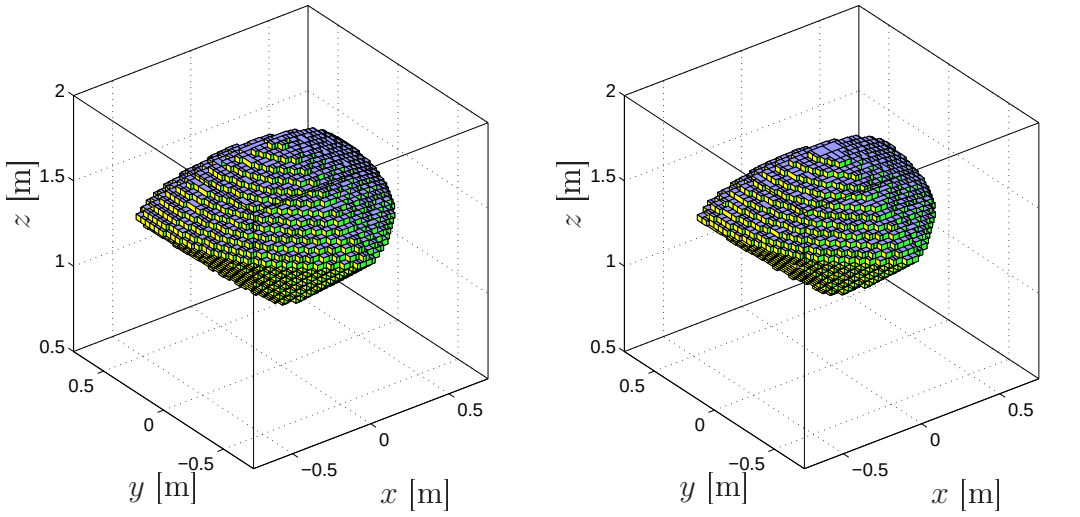


Abbildung 4.17: Rand des translatorischen Arbeitsraum $\mathcal{W}_{\mathrm{T}}$ für verschiedene Durchmesser d_l [cm] der Stäbe

Tabelle 4.3: Nominelle Parameter des SSM *PaLiDA*

Parameter	Wert	Beschreibung
r_b	0.425 m	Radius des Rahmens
r_p	0.09 m	Radius der beweglichen Plattform
$\Delta\alpha_b$	17.5°	Winkel Anordnung der rahmenseitigen Anlenkpunkte
$\Delta\alpha_p$	16.1276°	Winkel Anordnung der plattformseitigen Anlenkpunkte
$l_{\min}$	0.391 m	minimale Länge der Antriebs
$l_{\max}$	0.955 m	maximale Länge der Antriebs

verfügten die einzelnen PCs über stark unterschiedliche Prozessoren (u. a. Intel Pentium III-300 MHz, Intel Pentium IV 3.2 GHz, AMD Athlon und AMD Athlon XP, jeweils in verschiedenen Taktfrequenzen). Zusammenfassend kann festgestellt werden, dass keine zwei gleichen Systeme an den Cluster angeschlossen waren. Die Infrastruktur des Netzwerks wurde während der Testläufe nicht exklusiv zum parallelen Rechnen verwendet. Die Rechenzeit je PC wurde mit dem *Multi Processing Environment* (MPE) überwacht, das Teil der meisten MPI Implementierungen ist. Der Testlauf zeigte, dass die Arbeit trotz des stark inhomogenen PC Clusters sehr gleichmäßig verteilt wurde (Abb. 4.23) und dass die Kommunikation vernachlässigbare Verzögerungen verursachte. Die Testergebnisse lassen daher erwarten, dass auch auf größeren Clustern eine gute Skalierbarkeit erreicht werden kann.



a) translatorischer Arbeitsraum $\mathcal{W}_{\mathrm{T}}$ b) Gesamtorientierungsarbeitsraum $\mathcal{W}_{\mathrm{GO}}$

Abbildung 4.18: Arbeitsraum des SSM

Tabelle 4.4: Nominelle Parameter des DLR-Fahrsimulators

Parameter	Wert	Beschreibung
r_b	5.518 m	Radius des Rahmens
r_p	3.957 m	Radius der beweglichen Plattform
$\Delta\alpha_b$	49.56°	Winkel Anordnung der rahmenseitigen Anlenkpunkte
$\Delta\alpha_p$	7.258°	Winkel Anordnung der plattformseitigen Anlenkpunkte
$l_{\min}$	4.45 m	minimale Länge der Antriebs
$l_{\max}$	6.45 m	maximale Länge der Antriebs

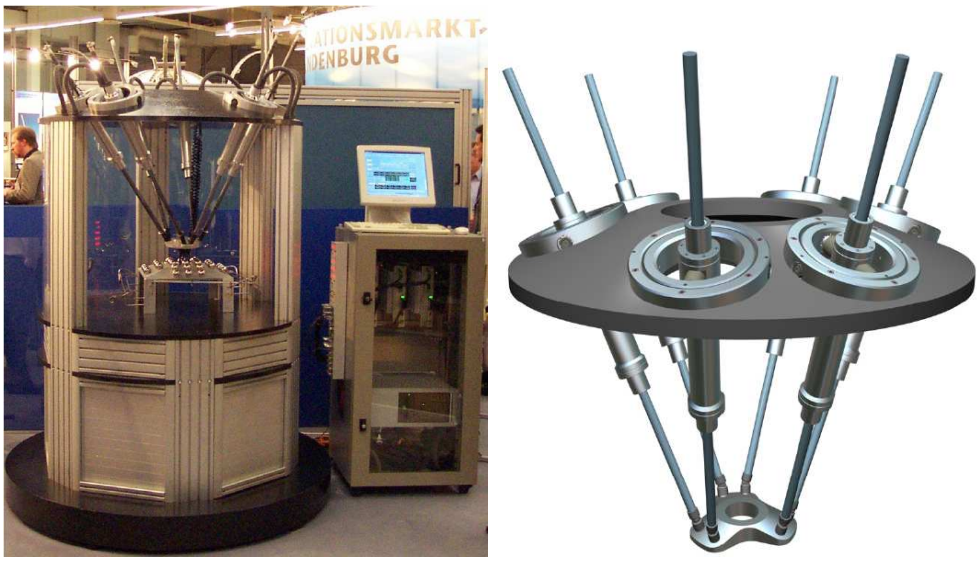
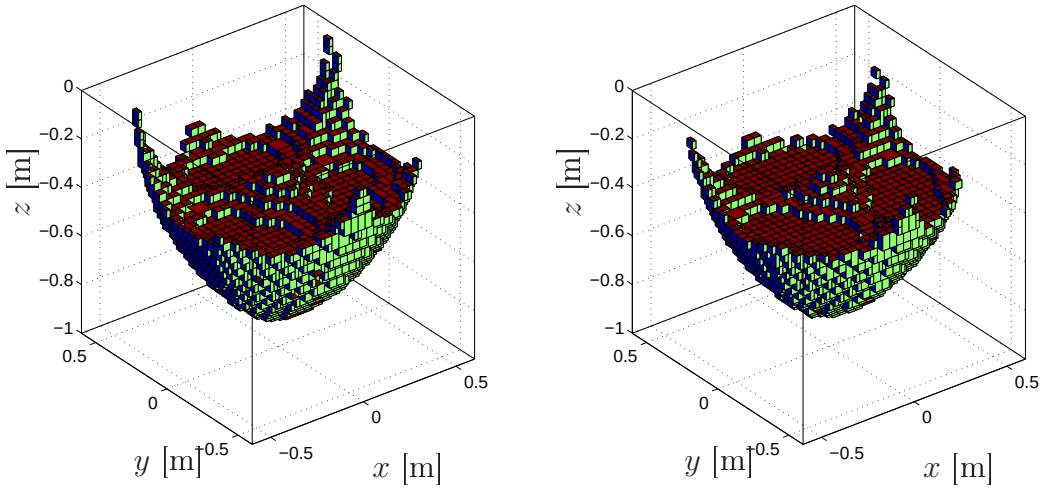


Abbildung 4.19: *PaLiDA* Prüfstand (© Photo Mechatronik-Zentrum Hannover)



a) translatorischer Arbeitsraum $\mathcal{W}_{\mathrm{T}}$ b) Gesamtorientierungsarbeitsraum $\mathcal{W}_{\mathrm{GO}}$

Abbildung 4.20: Arbeitsraum des *PaLiDA*

Abbildung 4.21: Fahrsimulator am DLR Braunschweig auf der Basis eines SSM (© Photo DLR Braunschweig)

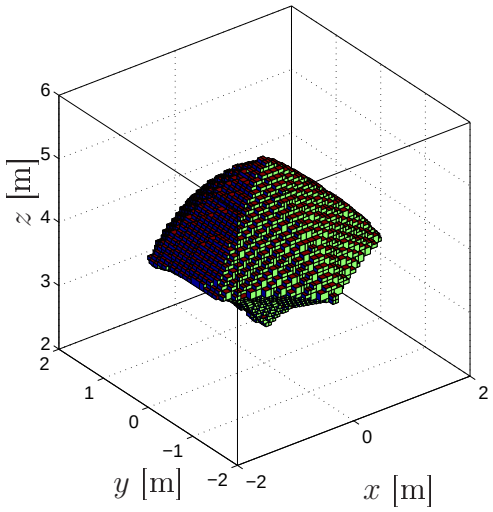


Abbildung 4.22: Translatorischer Arbeitsraum $\mathcal{W}_{\mathrm{T}}$ des DLR-Fahrsimulators

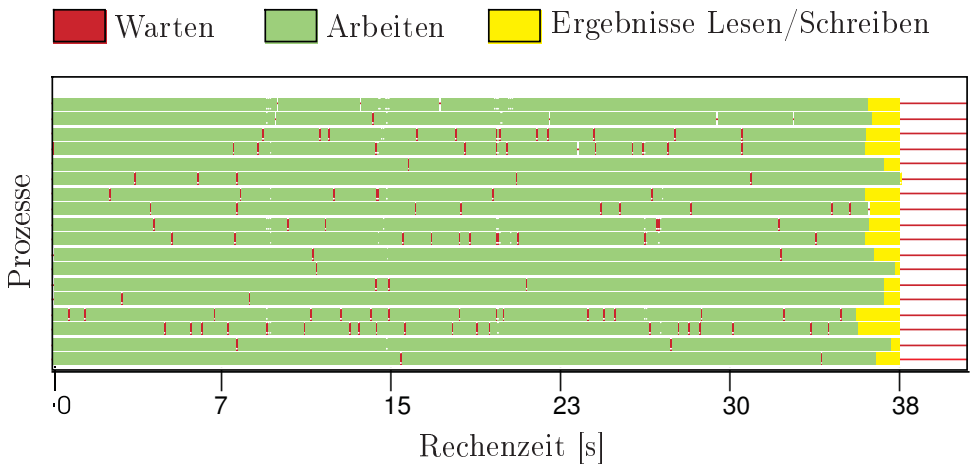


Abbildung 4.23: Verteilung der Arbeitslast beim parallelen CSP-Löser auf einem heterogenen Cluster mit 18 Slaves

Kapitel 5

Maßsynthese

In diesem Kapitel wird eine Methode zum Entwurf von PKM vorgestellt und in einem Programmgerüst strukturiert. Die Maßsynthese für gegebene Prozessanforderungen wird in Abschnitt 5.2 als CSP formuliert und in Abschnitt 5.3 mit einem Algorithmus zur globalen Optimierung kombiniert. Die Bestimmung der technologischen Parameter wird in Abschnitt 5.4 behandelt. Ein Programmgerüst zur vereinheitlichten Analyse, Synthese und Optimierung von PKM wird in Abschnitt 5.5 vorgestellt und auf einige Beispiele angewendet (Abschnitt 5.6).

5.1 Einführung

Die kinematischen Eigenschaften von Robotern hängen stark von ihrer Geometrie ab und ändern sich über dem Arbeitsraum. Dies gilt in besonderem Maße für PKM. Nach Merlet ist eine gute Wahl der Geometrie einer PKM wichtiger als deren Architektur [76]. Die Maßsynthese spielt daher eine zentrale Rolle beim Entwurf und der Konstruktion von PKM. An eine Methode zur Synthese von Mechanismen müssen zwei grundlegende Anforderungen gestellt werden. Zunächst müssen bestimmte Kriterien ohne Einschränkungen erfüllt werden. Diese Kriterien erwachsen sowohl aus prozessbedingten Anforderungen als auch aus technologischen Limitierungen. Daneben gibt es Kriterien, die optimal realisiert werden sollen, d. h. es gibt keine zwingende Vorgabe für diese Kriterien, aber die generelle Forderung nach einer Minimierung.[1] Diese werden durch eine Kostenfunktion repräsentiert, die beispielsweise den Energiebedarf, die äußeren Abmessungen oder die monetären Kosten beschreibt.

In der Literatur wird zum Entwurf von Maschinen hauptsächlich eine Methode eingesetzt, die als *optimales Design* bezeichnet wird (siehe z. B. [94, 99]). Dazu wird der Manipulator durch *Designparameter* $\underline{g}$ beschrieben, die innerhalb einer *Klasse* von Maschinen verschiedene *Varianten* definieren. Jeder Wert für $\underline{g}$ entspricht dabei genau einer Variante. Um die verschiedenen Varianten miteinander vergleichen zu können, werden Kennzahlen η_i

[1] Für die mathematische Optimierung ist die Suche nach einem Maximum und Minimum äquivalent, denn die Maximierung der Funktion f kann als Minimierung der Funktion $-f$ betrachtet werden. Ohne Beschränkung der Allgemeinheit wird in dieser Arbeit davon ausgegangen, dass die gesuchten Extremwerte Minima sind.

definiert, welche die Güte einer Maschine bezüglich eines bestimmten Kriteriums repräsentieren. Beispiele für wichtige Kriterien sind u. a.:

- Volumen des Arbeitsraums,
- Steifigkeit des Manipulators,
- Manipulierbarkeit,
- Genauigkeit,
- Energiebedarf,
- Nähe zu Autokollisionen,
- Nähe zu Singularitäten.

Für ein optimales Design wird nun gefordert, dass eine aus diesen Kennzahlen zusammengesetzte Zielfunktion minimiert wird. Da die Kennzahlen von Natur aus verschiedene Dimensionen besitzen, müssen dazu Gewichtungsfaktoren k_i eingeführt werden, um die verschiedenen Dimensionen anzupassen und relative Gewichtungen zwischen den Kriterien vorzunehmen. In der Praxis wird häufig die Zielfunktion

$$f(\underline{g}) = \sum_i k_i \eta_i(\underline{g}) \tag{5.1}$$

verwendet. Einige wichtige Kriterien, wie z. B. die Existenz von Singularitäten im Arbeitsraum, können nur durch die Zustände *erfüllt* oder *nicht erfüllt* beschrieben werden. Dementsprechend wird der Kennzahl η_i der Wert 0 (erfüllt) oder 1 (nicht erfüllt) zugeordnet. Durch ein ausreichend hohes Gewicht k_i (Penalty) soll in diesen Fällen die Erfüllung des Kriteriums erzwungen werden. Dieser Ansatz hat zwei entscheidende Nachteile. Erstens ist die Zielfunktion nicht stetig, was die Lösung des Optimierungsproblems erheblich erschwert. Zweitens ist es auch möglich, Lösungen zu finden, die nicht alle gestellten Kriterien erfüllen und daher im Sinne der Aufgabe ungültig sind. Selbst wenn dieses Problem nicht auftritt, führt die Zielfunktion nach Gleichung (5.1) zu Lösungen, die sehr empfindlich von der Wahl der Gewichtungsfaktoren k_i abhängen. Da die Kriterien normalerweise antagonistisch sind, entsprechen die Lösungen für gegebene k_i einem Kompromiss zwischen den Anforderungen. Daher ist das optimale Design stark durch die Wahl der Gewichte geprägt. Leider gibt es a priori keine Möglichkeit, die Gewichte k_i geeignet zu wählen.

Der in dieser Arbeit vorgestellte Ansatz für die Maßsynthese und Optimierung von PKM basiert auf der Idee, sämtliche technischen und prozessbedingten Anforderungen als zwingend erforderlich zu betrachten und folglich als Nebenbedingungen zu formulieren. Dies erscheint vor allem für praktische Aufgaben als angemessen, denn in der Regel hat ein Anwender genaue Vorstellungen davon, was eine Maschine leisten muss. Ferner ist es so möglich, verschiedene Nebenbedingungen gleichzeitig zu berücksichtigen. Diese Betrachtung ist intuitiv, führt jedoch auf das komplexe Optimierungsproblem

$$\text{minimiere} \quad f(\underline{g}) \tag{5.2}$$

$$\underline{\Phi}(\underline{g}, \underline{y}) \;>\; \underline{0} \quad \forall\, \underline{y} \in \mathcal{W}, \tag{5.3}$$

dessen Lösung in diesem Kapitel diskutiert wird. Gesucht ist ein globales Optimum der Kostenfunktion f unter den Nebenbedingungen $\underline{\Phi}$ [5, 38]. Dies ist ein selten betrachtetes

Optimierungsproblem, denn die Nebenbedingungen werden durch ein *Constraint Satisfaction Problem* (CSP) beschrieben und müssen über der Menge $\mathcal{W}$ erfüllt werden. Die Grundlagen für die Lösung dieses Typs von CSP wurden ausführlich im Kapitel 4 untersucht. Das grundsätzliche Problem bei der globalen Optimierung besteht darin, dass es im Allgemeinen sehr schwierig ist festzustellen, ob ein gefundenes Extremum globale Gültigkeit besitzt. Verbreitete Verfahren zur globalen Optimierung basieren auf *heuristischen Methoden*, *Approximationsmethoden* oder *systematischen Methoden* [5]. Zu den heuristischen Methoden gehören *evolutionäre Verfahren* (genetische Algorithmen), *Simulated Annealing* und *Monte-Carlo-Methoden*. Diese Methoden beinhalten alle eine stochastische Komponente, die einerseits für eine breit gestreute Durchmusterung des Suchraums sorgt, aber den Nachteil hat, dass das Auffinden von globalen Extrema nicht garantiert werden kann. Weiterhin sind diese Methoden meist für einfache Nebenbedingungen entwickelt worden.

In diesem Kapitel wird ein Verfahren zur globalen Optimierung mit Nebenbedingungen betrachtet, das auf der Intervallanalyse basiert [40, 101]. Dieser Algorithmus betrachtet die Nebenbedingungen als CSP und kombiniert die Methode zur Lösung aus Kapitel 4 mit einem intervallbasierten Verfahren für die globale Optimierung. Daher lassen sich für die Analyse und Synthese identische Gleichungen verwenden. In Abschnitt 5.5 wird gezeigt, dass dies mit einer integrierten Implementierung möglich ist [105]. Der besondere Vorteil des CSP-Lösers besteht darin, dass die Nebenbedingungen mit Verifikationsvariablen formuliert werden können, die über einer Menge $\mathcal{X}_v$ erfüllt werden müssen. Vor dem Hintergrund der Synthese von PKM wird in diesem Kapitel die Menge $\mathcal{X}_v$ meist mit dem gewünschten Arbeitsraum $\mathcal{W}$ identifiziert.

5.2 Maßsynthese als Constraint Satisfaction Problem

Die Maßsynthese für parallele Roboter wird wie die Arbeitsraumberechnung und -verifikation als CSP formuliert und kann daher mithilfe des hybriden CSP-Lösers (Abschnitt 4.2.4) behandelt werden. Dies führt wiederum auf ein Problem vom Typ

$$\underline{\Phi}(\underline{c}, \underline{v}) > \underline{0} \quad \forall\, \underline{v} \in \mathcal{X}_v \tag{5.4}$$

mit dem System der Bindungsgleichungen $\underline{\Phi}$, den Berechnungsvariablen $\underline{c}$, den Verifikationsvariablen $\underline{v}$ und der Verifikationsmenge $\mathcal{X}_v$. Für die Maßsynthese werden die geometrischen Parameter $\underline{g}$ mit den Berechnungsvariablen $\underline{c}$ und die Weltkoordinaten $\underline{y}$ mit den Verifikationsvariablen $\underline{v}$ identifiziert. Der gewünschte Arbeitsraum $\mathcal{W}$ wird dabei als Verifikationsmenge $\mathcal{X}_v$ vorgegeben. Damit repräsentiert dieses CSP die Aufgabe, alle Varianten $\underline{g}$ der Maschine zu bestimmen, die einen Arbeitsraum mit vorgegebener Größe $\mathcal{W}$ und vorgegebenen Eigenschaften $\underline{\Phi}$ besitzen. Der im Kapitel 4 beschriebene CSP-Löser mit der Intervallanalyse garantiert, dass die so gefundenen Varianten der Maschine an *jedem* Punkt im gewünschten Arbeitsraum $\mathcal{W}$ die gestellten Anforderungen erfüllen. Für den Aufbau des Systems $\underline{\Phi}$ werden dieselben Bindungen verwendet, die bereits für die Arbeitsraumuntersuchung in Abschnitt 4.4 eingeführt wurden. Daher kann das für die Analyse erstellte Modell der Maschine unmittelbar für die Synthese verwendet werden. Dadurch entsteht eine durchgängige Methodik, die beginnend mit der vorläufigen Analyse der PKM über die Maßsynthese und Optimierung bis hin zur Verifikation der Ergebnisse

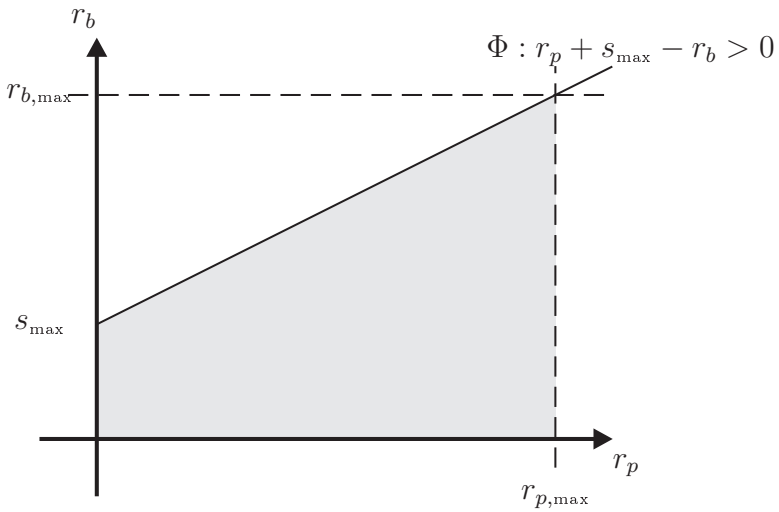


Abbildung 5.1: Parameterrelation am Beispiel des Radius von Plattform r_p und Radius des Rahmens r_b beim SSM

einen einheitlichen Rahmen bietet. Die speziellen Aspekte für die Implementierung eines *Programmgerüsts* werden in Abschnitt 5.5 erläutert.

5.2.1 Bindungen für die Maßsynthese

In Abschnitt 4.4 wurden Bindungen eingeführt, die die Einhaltung von kinematischen und technologischen Anforderungen erzwingen. Für die Maßsynthese ist es zweckmäßig, weitere Bindungen zu formulieren, welche die geometrischen Abmessungen der Maschine direkt in Relation zueinander setzen. Da die Geometrie bei der Arbeitsraumberechnung konstant ist, haben diese Bindungen dort keine Bedeutung. Zunächst können grobe Abschätzungen über den Wertebereich der Parameter aufgestellt werden. Weiterhin lassen sich einige Parameter derart zueinander in Beziehung setzen, dass ein Teil des Suchraums $\mathcal{X}_s$ ausgeschlossen werden kann. Dies hat den Vorteil, dass diese Bedingungen unabhängig von dem unterlagerten Algorithmus `Verifizieren` getestet werden können, d. h. diese Bindungen werden als separiertes Subsystem behandelt (Abschnitt 4.3.1), dessen Verifikationsmenge $\mathcal{X}_v$ leer ist und daher besonders schnell überprüft werden kann. Entsprechend groß ist das Potential zur Einsparung von Rechenzeit. Diese Art der Parameterrelation soll an einem einfachen Beispiel für den SSM illustriert werden. Damit der Arbeitsraum das Manipulators nicht prinzipiell leer ist, darf der Radius des Rahmens r_b nicht größer sein als die Summe aus Radius der Plattform r_p und maximaler Länge der Aktuatoren $s_{\max}$. Es gilt also

$$r_p + s_{\max} - r_b > 0. \tag{5.5}$$

Solche Regeln erscheinen trivial und können leicht aufgestellt und implementiert werden. In Abbildung 5.1 ist der gültige Teil des Parameterraums nach Gleichung (5.5) dargestellt. Es ist zu erkennen, dass diese einfache Ungleichung eine große Anzahl ungültiger Designs ausschließt, ohne die aufwendige Verifikation des Arbeitsraums einer Maschine durchzuführen. Diese Bindung kann effizient auf Konsistenz untersucht werden und auch eine Untersuchung des Gradienten führt zu einer rascheren Konvergenz. Aufgrund der modularen Struktur des CSP-Lösers können mit wenig Aufwand beliebig viele solcher

Tabelle 5.1: Grenzen für die geometrischen Parameter des *Linapod*

Symbol	Minimum	Maximum	Beschreibung
r_b	0	—	Radius des Rahmens der Maschine
r_u	0	r_b	Radius der oberen Plattformebene
r_l	0	r_b	Radius der unteren Plattformebene
l_l	$r_b - r_l$	—	Länge des unteren Beins
l_u	$r_b - r_u$	—	Länge des oberen Beins
Δa	$-60°$	$60°$	Tangentialer Versatz der rahmenseitigen Anlenkpunkte
Δh	0	—	Abstand zwischen unterer und oberer Plattformebene
Δt	0	—	Abstand zwischen $\mathcal{K}_p$ und unterer Plattformebene
$\Delta\alpha_p$	$-60°$	$60°$	Verdrehung der unteren und oberen plattformseitigen Anlenkpunkte gegenüber $\mathcal{K}_p$

Relationen berücksichtigt werden. Die Betrachtung von Parameterrelationen wirkt sich ausgesprochen günstig auf die Laufzeit des Algorithmus aus, denn Parameterrelationen erlauben, ungültige Varianten einer Maschine mit wenigen Rechenoperationen auszuschließen. Durch Relationen können einfache Regeln und anwendungsspezifische Restriktionen systematisch eingearbeitet werden.

5.2.2 Parameterrelationen für PKM

Im Folgenden wird beispielhaft eine Relation für die vorliegende Parametrisierung des *Linapod* aufgezeigt. In Tabelle 5.1 und Tabelle 5.2 sind Schranken des Wertebereichs der geometrischen Parameter für die Maschinen *Linapod* und SSM zusammengefasst.

Als Beispiel für eine Relation zwischen den geometrischen Parametern wird die Kollision der Beine mit der Plattform betrachtet. Hier wird davon ausgegangen, dass die Plattform durch einen zylinderförmigen Spindelkasten mit einer Länge l_s und einem Radius r_s beschrieben wird (Abb. 5.2). Der maximal mögliche Schwenkwinkel der Plattform wird durch die Kollision mit den oberen Stäben begrenzt. Unter Vernachlässigung des Durchmessers des oberen Stabs ergibt sich die Bedingung für eine Kollision (Abb. 5.2) zu

$$\varphi_s = \arctan \frac{r_u - r_s}{l_s - \Delta h - \Delta t}. \tag{5.6}$$

Der minimale Winkel $\beta_{\min}$ kann aus verschiedenen Gründen begrenzt sein. Das Kardangelenk könnte einen beschränkten Schwenkbereich besitzen oder das maximale Überset-

Tabelle 5.2: Grenzen für die geometrischen Parameter des SSM

Symbol	Minimum	Maximum	Beschreibung
r_b	0	—	Radius des Rahmens der Maschine
r_p	0	—	Radius der Plattform
$\Delta\alpha_b$	$-60°$	$60°$	Versatz der rahmenseitigen Anlenkpunkte
$\Delta\alpha_p$	$-60°$	$60°$	Versatz der plattformseitigen Anlenkpunkte

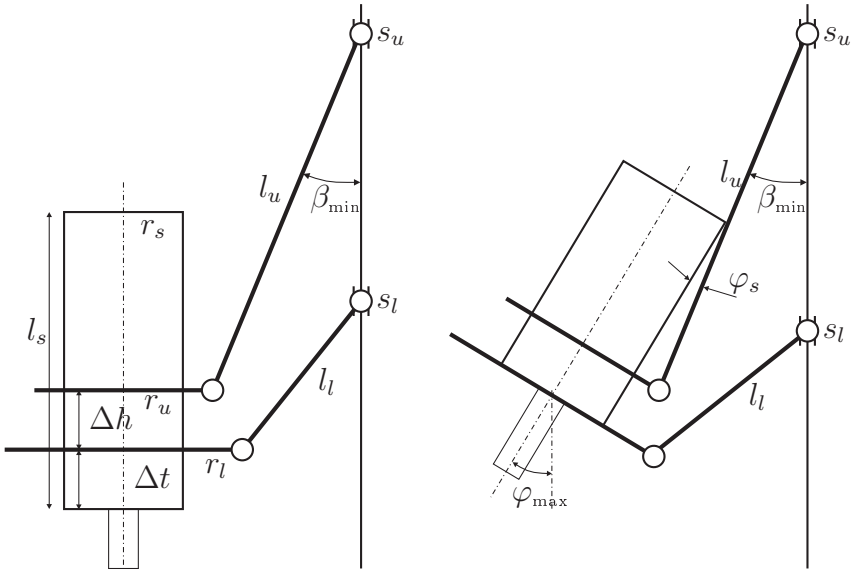


Abbildung 5.2: Maximal mögliche Schwenkwinkel der Plattform unter Berücksichtigung der Kollision zwischen Beinen und Spindelkasten

zungsverhältnis $j_{\max}$ zwischen der Bewegung des linearen Aktuators und der radialen Bewegung soll auf

$$\beta_{\min} = \arcsin \sqrt{\frac{1}{1 + j_{\max}^2}} \tag{5.7}$$

limitiert werden. Für den zweiten Fall folgt nach Abbildung 5.2 für den maximalen Schwenkwinkel $\varphi_{\max}$

$$\varphi_{\max} = \varphi_s + \beta_{\min} = \arctan \frac{r_u - r_s}{l_s - \Delta h - \Delta t} + \arcsin \sqrt{\frac{1}{1 + j_{\max}^2}}. \tag{5.8}$$

Die Bestimmung von $\varphi_{\max}$ ist dabei unabhängig von den Verdrehwinkeln der plattformseitigen Anlenkpunkte, da der maximale Schwenkwinkel in allen Richtungen realisiert werden soll. Daher muss das Schwenken insbesondere auch in Richtung der Stäbe möglich sein. Für die Synthese wird Gleichung (5.8) zu

$$\Phi : \varphi_{\max} - \arctan \frac{r_u - r_s}{l_s - \Delta h - \Delta t} - \arcsin \sqrt{\frac{1}{1 + j_{\max}^2}} > 0 \tag{5.9}$$

umgeformt, so dass sich eine Relation zwischen den geometrischen Parametern ergibt. In ähnlicher Weise kann überprüft werden, ob die Schwenkwinkel an den Eckpunkten des Arbeitsraums eingehalten werden. Da der Schwenkwinkel $\varphi_{\max}$ als Eigenschaft des Gesamtorientierungsarbeitsraums vorgegeben ist, ergibt sich so eine Ungleichung für die Plattformgeometrie und die Länge der oberen Stäbe, die nicht von den Weltkoordinaten abhängt. Bei der Maßsynthese ist dies besonders günstig, da damit ein separierbares Subsystem entsteht, das keine Verifikationsvariablen enthält und damit sehr schnell ausgewertet werden kann.

5.3 Globale Optimierung

Die Maßsynthese wurde für gegebene Prozessanforderungen als CSP formuliert, und alle gültigen Varianten der Maschine können mit dem hybriden CSP-Löser (Abschnitt 4.2.4) ermittelt werden. Da garantiert werden kann, dass jede dieser Varianten alle gestellten Anforderungen erfüllt, stellt sich nun die Frage, wie die optimale Maschine aus diesen Varianten ausgewählt werden kann. Dazu wird zunächst unterstellt, dass es eine Zielfunktion f gibt, deren Minimum die optimale Konfiguration repräsentiert. Diese Zielfunktion könnte beispielsweise die äußeren Abmessungen der Maschine beschreiben oder ein Maß für die monetären Kosten der Maschine sein. Dazu wird in den folgenden Abschnitten eine Methode zur globalen Optimierung anhand der Intervallanalyse untersucht.

5.3.1 Freie globale Optimierung

Zunächst wird die globale Optimierung ohne Nebenbedingungen betrachtet. Bei der Suche nach einem Minimum der Funktion soll eine Stelle gefunden werden, die kleinere Funktionswerte besitzt als alle Stellen in ihrer Umgebung. Die Auswertung einer Funktion liefert jedoch nur einen lokalen Funktionswert an genau der ausgewerteten Stelle. Das Problem bei der globalen Optimierung besteht darin, dass eine normale Funktionsauswertung keine Informationen mit globaler Gültigkeit liefert. Es wurde daher lange Zeit vermutet, dass es keinen Algorithmus geben kann, der garantiert das globale Minimum findet. Die Intervallarithmetik schließt diese Lücke, denn sie ermöglicht es, rigorose Schranken für den Wertebereich einer Funktion über einer Menge von Werten zu berechnen. Daher kann die Intervallauswertung einer Funktion globale Informationen über diese Funktion liefern. Obwohl die so ermittelten Schranken den Wertebereich der Funktion möglicherweise erheblich überschätzen, bildet diese Auswertung eine garantierte Einschachtelung der tatsächlichen auftretenden Werte. Seien $\widehat{f}_1 = f(\widehat{c}_1)$ und $\widehat{f}_2 = f(\widehat{c}_2)$ zwei Funktionsauswertungen über zwei disjunkten Intervallen $\widehat{c}_1, \widehat{c}_2$, dann kann durch den Vergleich von $\widehat{f}_1$ und $\widehat{f}_2$ die folgende global gültige Aussage getroffen werden

- $\sup f_1 < \inf f_2$: Das Intervall $\widehat{c}_2$ kann nicht das globale Minimum enthalten.
- $\sup f_2 < \inf f_1$: Das Intervall $\widehat{c}_1$ kann nicht das globale Minimum enthalten.

Auf der Basis dieses einfachen Vergleichs lässt sich ein *Branch-and-Bound* Algorithmus aufbauen, der garantiert das globale Minimum im Suchbereich findet [39]. Der prinzipielle Aufbau des Algorithmus sieht folgendermaßen aus:

Algorithmus 10: `Generischer globaler Optimierer`

1. Werte für den Suchraum $\{\underline{\widehat{c}}_1, \ldots, \widehat{c}_n\}$ die Zielfunktion $\widehat{h}_i = f(\underline{\widehat{c}}_i)$ für jede Box aus und speichere die Paare $(\underline{\widehat{c}}_i, \widehat{h}_i)$ in der Liste $\mathcal{L}_\mathrm{T}$.
2. Erzeuge eine leere Liste $\mathcal{L}_\mathrm{S}$ für die Kandidaten für das globale Minimum; setze die garantierte obere Grenze $h^* = \infty$.
3. Falls die Liste $\mathcal{L}_\mathrm{T}$ leer ist, beende den Algorithmus.
4. Aktualisiere h^*, falls $h^* > \min_{\mathcal{L}_\mathrm{T}}(\sup \widehat{h})$, d. h. es gibt eine Box in $\mathcal{L}_\mathrm{T}$ mit einem kleineren Supremum.
5. Entferne alle Paare mit $\inf \widehat{h} > h^*$ aus der Liste $\mathcal{L}_\mathrm{T}$ und $\mathcal{L}_\mathrm{S}$.

6. Entnimm der Liste $\mathcal{L}_\mathrm{T}$ das Paar $(\widehat{\underline{c}}, \widehat{h})$ mit dem kleinsten Infimum $\inf \widehat{h}$.
7. Falls verfügbar, wende Operationen zum *Schrumpfen* von $\widehat{\underline{c}}$ an.
8. Falls diam $\widehat{\underline{c}} < \varepsilon$ und diam $\widehat{h} < \mu$, d. h. die Breite aller Komponenten der Box ist kleiner als ein gegebenes ε und die Breite des Werts der Zielfunktion ist kleiner als μ, dann ist die Box hinreichend genau bestimmt; speichere $(\widehat{\underline{c}}, \widehat{h})$ in der Liste $\mathcal{L}_\mathrm{S}$; gehe zu (3).
9. Teile die Box $\widehat{\underline{c}}$ in m Subboxen $\{\widehat{\underline{c}}_1, \ldots, \widehat{\underline{c}}_m\}$, Werte die Zielfunktion $\widehat{h}_i = f(\widehat{\underline{c}}_i)$ für jede Subbox aus und speichere die neuen Paare in der Liste $\mathcal{L}_\mathrm{T}$; gehe zu (3).

Das Verhalten des Algorithmus wird durch zwei wichtige Parameter gesteuert, die zu Beginn festgelegt werden müssen. ε ist die Genauigkeit für die Größe der betrachteten Boxen und μ ist der maximale Fehler der Zielfunktion. Wenn der Algorithmus abgeschlossen wurde, erfüllt jedes Paar $(\widehat{\underline{c}}_i, \widehat{h}_i)$ in der Liste $\mathcal{L}_\mathrm{S}$

$$\operatorname{diam} \widehat{\underline{c}}_i < \varepsilon, \tag{5.10}$$

$$\operatorname{diam} \widehat{h}_i < \mu, \tag{5.11}$$

$$h_* \leq \widehat{h}_i \leq h^*. \tag{5.12}$$

Eine untere Schranke für den Wert der Zielfunktion wurde ebenfalls ermittelt, denn die Zielfunktion fällt nicht unter den Wert $h_* = \min_{\mathcal{L}_\mathrm{S}}(\inf \widehat{h})$. Zusammenfassend besitzt der Algorithmus folgende bemerkenswerte Eigenschaften:

- Der Algorithmus findet garantiert das globale Minimum. Falls mehrere, gleichwertige globale Minima existieren, werden alle ermittelt.
- Es werden automatisch garantierte obere und untere Schranken für den Wert der Zielfunktion gefunden.
- Es werden garantierte Schranken für die Designparameter gefunden, die das globale Minimum enthalten.
- Die Zielfunktion muss nicht stetig sein.
- Der Algorithmus eignet sich gut zur Parallelisierung.
- Durch die Verwendung der Intervallanalyse ist der Algorithmus robust gegenüber numerischen Rundungsfehlern.

Eine einfache Kombination von Maßsynthese und globaler Optimierung kann bereits anhand dieses einfachen Algorithmus vorgenommen werden. Dazu wird die Lösungsmenge der Maßsynthese als Eingangsdaten für den globalen Optimierer übernommen, d. h. die Liste $\mathcal{L}_\mathrm{T}$ des `generischen globalen Optimierers` wird mit dem Inhalt der Liste $\mathcal{L}_\mathrm{S}$ des `hybriden CSP-Lösers` initialisiert. Ein Vergleich der Algorithmen zur Lösung von CSP und der globalen Optimierung zeigt jedoch, dass beide Verfahren große Ähnlichkeit miteinander haben. Bei den in dieser Arbeit betrachteten Problemen ist die Auswertung der Bindungsgleichungen um Größenordnungen aufwendiger als die Auswertung der Zielfunktion, da für die Bindungsgleichungen der Arbeitsraum verifiziert werden muss. Im folgenden Abschnitt 5.3.2 wird eine synergetische Koppelung der beiden Verfahren vorgenommen, die zunächst eine zulässige Lösung für die Bindungsgleichungen ermittelt und dann anhand der Zielfunktion überprüft, ob eine Verbesserung der besten bekannten Variante möglich ist. Falls h^* nicht verbessert werden kann, wird die Variante verworfen, ohne die aufwendige Untersuchung der Bindungsgleichungen durchzuführen.

5.3.2 Globale Optimierung mit Nebenbedingungen

Eine Zusammenfassung von CSP und globaler Optimierung führt zu einem globalen Optimierungsproblem mit Nebenbedingungen, das sich folgendermaßen darstellen lässt:

$$\text{minimiere} \quad f(\underline{c}) \tag{5.13}$$

$$\underline{\Phi}(\underline{c}, \underline{v}) \quad > \quad \underline{0} \quad \forall\, \underline{v} \in \mathcal{X}_v. \tag{5.14}$$

Der Vergleich zwischen dem hybriden CSP-Löser und dem Algorithmus zur globalen Optimierung zeigt, dass die Algorithmen einen fast identischen Aufbau aufweisen. Der wichtigste Unterschied besteht darin, durch welchen Test eine Box verworfen wird. Beim CSP-Löser wird das System $\underline{\Phi}(\underline{c}, \underline{v}) > 0$ betrachtet, während für die globale Optimierung die Zielfunktion $f(\underline{c})$ mit der besten bekannten Lösung h^* verglichen wird. Damit eine Box $\widehat{\underline{c}}$ als Lösung in Betracht kommt, müssen beide Bedingungen erfüllt sein. Für die hier betrachtete Optimierung von PKM kann die Zielfunktion $f(\widehat{\underline{c}})$ um Größenordnungen schneller ausgewertet werden als der Algorithmus `Verifizieren` für die Box $\widehat{\underline{c}}$. Daher ist es günstiger, zunächst die Zielfunktion auf eine mögliche Verbesserung hin zu überprüfen. Weiterhin ist festzustellen, dass sich die obere Schranke h^* für das Minimum sukzessiv verkleinert, so dass sich die Schärfe dieses Kriteriums stetig verbessert.

5.3.3 Konsistenztest für die Zielfunktion

Ähnlich wie beim CSP-Löser kann die Laufzeit des Optimierers durch zusätzliche Maßnahmen drastisch verbessert werden. Die folgenden Abschnitte beschreiben Techniken, die insbesondere für die Behandlung von PKM ausgewählt wurden. Diese und weitere Tests sind in der Literatur zur globalen Optimierung mit der Intervallanalyse zu finden [5, 39].

Sobald eine gültige Lösung $\widehat{\underline{c}}$ des CSP gefunden wurde, wird eine obere Schranke für den Funktionswert des globalen Minimums ermittelt, indem eine Intervallauswertung der Zielfunktion für diese Box berechnet wird. Das Supremum $h^* = \sup f(\widehat{\underline{c}})$ ist eine globale obere Schranke für das Minimum. Diese Schranke kann häufig verbessert werden, indem die Zielfunktion f für einen diskreten Punkt $\underline{c} = \operatorname{mid} \widehat{\underline{c}} \in \widehat{\underline{c}}$ ausgewertet wird. Die Eigenschaften der Intervallanalyse garantieren dabei, dass $f(\underline{c}) \in f(\widehat{\underline{c}})$ und damit $h'^* = f(\underline{c}) \leq h^*$ gilt. Vor allem für relativ große Boxen $\widehat{\underline{c}}$ wird der Einfluss von Überschätzung und Abhängigkeiten durch die Auswertung für einen Punkt $\underline{c}$ stark reduziert, so dass h'^* häufig eine deutlich kleinere obere Schranke liefert als h^*. Da per Konvention die kleinste bekannte obere Schranke mit h^* bezeichnet wird, wird h^* durch den Wert von h'^* ersetzt.

Das globale Minimum von f muss offensichtlich kleiner sein als h^*, so dass sich eine zusätzliche Bindungsgleichung

$$\Phi : h^* - f(\underline{c}) \geq 0 \tag{5.15}$$

formulieren lässt, die in das CSP eingearbeitet werden kann. Im Unterschied zu den übrigen Bindungen hängt diese Ungleichung von einem Parameter des Algorithmus ab. Auf diese Ungleichung lassen sich viele der Techniken zur Effizienzsteigerung anwenden (Abschnitt 4.3.3), die für CSP bekannt sind. Vor allem die Separation der Verifikationsmenge, der Test auf Konsistenz und die Verwendung von Abhängigkeiten können genutzt werden, um die Rechenzeit zu reduzieren.

Diese zusätzliche Bindung hat eine bemerkenswerte Eigenschaft. Da die obere Schranke h^* für das Minimum während der Optimierung fortlaufend verbessert wird, verschärft sich

die daraus gewonnene Bindung progressiv, d. h. die Bindung wird im Laufe der Berechnung immer effektiver bei der Eliminierung von Boxen $\widehat{\underline{c}}$, die nicht das globale Minimum enthalten können.

5.3.4 Verwendung des Gradienten der Zielfunktion

Für eine stetig differenzierbare Zielfunktion f können dem Gradienten ∇f wertvolle Informationen entnommen werden. Eine Intervallauswertung $\widehat{\underline{z}} = \nabla f(\widehat{\underline{c}})$ liefert für eine Box $\widehat{\underline{c}}$ eine garantierte Einschachtelung des Wertebereichs der partiellen Ableitungen der Funktion. Die Funktion f ist bezüglich ihrer i-ten Komponente *monoton steigend*, falls für diese Komponente $\widehat{z}_i > 0$ gilt, d. h. $\inf \widehat{z}_i > 0$ ist. Sie besitzt dann garantiert kein Minimum im Inneren der Box $\widehat{\underline{c}}$ sondern höchstens auf dem Rand der Box. Daher kann die Box auf das teilweise degenerierte Intervall $\widehat{\underline{c}}' = [\widehat{c}_1, \ldots, \inf \widehat{c}_i, \ldots, \widehat{c}_n]$ reduziert werden, und mit $h^* = f(\widehat{\underline{c}}')$ ergibt sich eine verbesserte obere Schranke für das globale Minimum. Entsprechend ist die Funktion f bezüglich ihrer i-ten Komponente *monoton fallend*, wenn für diese Komponente $\widehat{z}_i < 0$ gilt. Dann nimmt die Funktion ihr lokales Minimum im Intervall $\widehat{\underline{c}}' = [\widehat{c}_1, \ldots, \sup \widehat{c}_i, \ldots, \widehat{c}_n]$ an. Die Box $\widehat{\underline{c}}$ wird daher durch $\widehat{\underline{c}}'$ ersetzt. Für die übrigen partiellen Ableitungen ergibt sich möglicherweise durch die Reduktion von $\widehat{\underline{c}}$ ebenfalls eine schärfere Auswertung, so dass der Test rekursiv wiederholt werden kann, um die obere Schranke h^* zu verbessern.

5.3.5 Zielfunktion

Für die Optimierung von PKM stellt sich die Frage, welche Eigenschaft der Maschine optimiert werden soll. Die kinematischen und technischen Anforderungen werden bereits durch die Nebenbedingungen der globalen Optimierung berücksichtigt, und die Verifikation über der Menge $\mathcal{X}_v = \mathcal{W}$ garantiert, dass alle gestellten Anforderungen in jedem Punkt des gewünschten Arbeitsraums $\mathcal{W}$ erfüllt werden. Daher können in der Zielfunktion f Kriterien berücksichtigt werden, für die keine strengen Grenzen bestehen, die aber unter der Anzahl der möglichen Varianten minimiert werden sollen. Beispiele für solche Kriterien sind die monetären Kosten der Maschine oder der gesamte benötigte Bauraum. Die Frage, welches Kriterium in der Praxis minimiert werden soll, führt jedoch thematisch von den in dieser Arbeit schwerpunktmäßig betrachteten Fragestellungen weg. Daher soll hier nur eine klare Schnittstelle zwischen dem kinematischen und technischen Entwurf von PKM und den möglichen Auswahlkriterien erarbeitet werden. Die Abschätzung der tatsächlichen Kosten für die Herstellung einer Maschine gehört zum Gebiet der Produktionstechnik und kann im Rahmen dieser Arbeit nicht näher betrachtet werden. Es ist jedoch möglich, die Fertigungskosten für eine gegebene Geometrie $\underline{g}$ anzugeben und als Zielfunktion in die globale Optimierung zu integrieren. Durch die hier gewählte Formulierung des Problems werden die beiden Fachgebiete an einer klar definierten Schnittstelle getrennt.

Als rein geometrisches Kriterium soll daher beispielhaft der Bauraum der Maschine minimiert werden. Eine vernünftige Abschätzung für das Volumen V kann aus den geometrischen Parametern berechnet werden und wird für jedes der Beispiele in Abschnitt 5.6 angegeben.

5.4 Synthese abhängiger technologischer Parameter

Durch die globale Optimierung lässt sich eine Maschine finden, die alle gestellten Anforderungen $\underline{\Phi}$ erfüllt und gleichzeitig die gegebene Zielfunktion f minimiert. Damit werden die kinematisch wesentlichen Abmessungen der Maschine ermittelt. Es verbleibt jedoch noch die Optimierung technologisch wichtiger Parameter, die bislang als gegeben betrachtet wurden. Am Beispiel der Gelenke soll die Möglichkeit zur Optimierung veranschaulicht werden. Bei der Analyse und Synthese wird angenommen, dass der maximale Schwenkwinkel der Gelenke als technologische Kenngröße bekannt ist, und während der Maßsynthese werden alle Varianten eliminiert, die diesen maximal möglichen Winkel nicht einhalten. Insofern ist eine von der Maßsynthese ermittelte Variante technisch machbar. Es wird nun überprüft, ob auch ein Gelenk mit einem geringeren Schwenkwinkel ausreichend ist.

Diese Frage lässt sich auch als CSP formulieren, bei dem für eine bekannte Variante $\underline{g}$ der Maschine und für einen gegebenen Arbeitsraum $\underline{y} \in \mathcal{W}$ geprüft wird, welche Werte der technologischen Parameter zu einer gültigen Variante führen. Die technologischen Kenngrößen werden dazu als Berechnungsvariablen $\underline{c}$ aufgefasst, und eine Verifikation muss über dem vorgegebenen Arbeitsraum $\mathcal{W}$ durchgeführt werden. Daher werden die Weltkoordinaten der Plattform als Verifikationsvariablen $\underline{v}$ verwendet, und die Verifikationsmenge $\mathcal{X}_v$ ist der gegebene Arbeitsraum $\mathcal{W}$. Durch die Maßsynthese wurde bereits die Existenz mindestens einer Lösung bewiesen, jedoch ist es durchaus möglich, dass technisch bessere Varianten existieren. Durch eine einfache Zielfunktion kann dann der optimale Wert für die technologischen Parameter bestimmt werden. Wie im folgenden Abschnitt gezeigt wird, ist diese Vertauschung der Rollen der einzelnen Parameter durch eine geeignete Gestaltung des Programmgerüsts ohne weiteren Aufwand möglich.

5.5 Implementierung eines objektorientierten Programmgerüsts

Die Verfahren zur Arbeitsraumberechnung und Maßsynthese wurden als CSP formuliert, so dass dieselben Bindungsgleichungen für sehr unterschiedliche Aufgaben verwendet werden können. Daher wird in diesem Abschnitt eine Implementierung betrachtet, bei der die Gleichungen nur einmal aufgestellt und eingegeben werden müssen. Danach stehen die Bindungen für jede der betrachteten Berechnungen zur Verfügung. Damit gelingt die selten realisierte Verzahnung von Analyse und Synthese innerhalb einer Methode und der gesamte Entwicklungsprozess wird vereinheitlicht. Zunächst kann eine vorläufige Analyse von speziellen vorgegebenen Maschinen durchgeführt werden. Dabei wird der Arbeitsraum berechnet, in dem alle betrachteten Eigenschaften erfüllt sind. Der Entwurf von Maschinen mit gegebenen Eigenschaften lässt sich nahtlos anschließen, da durch die Beschreibung als Bindungen $\underline{\Phi}$ die Rollen[2] von geometrischen Parametern und Weltkoordinaten vertauscht werden können. Bei der Maßsynthese werden damit alle Varianten der Maschine ermittelt, die garantiert die geforderten Eigenschaften im Arbeitsraum besitzen. Der besondere Vorteil besteht darin, dass die Bindungsgleichungen dazu nicht neu programmiert werden müssen. Da die Maßsynthese ein Teil des vorgestellten Verfahrens zur globalen Optimie-

[2]Die *Rolle* einer Variablen beschreibt deren Funktion bei dem jeweiligen Problem. Die Rolle einer Variablen kann *Berechnungsvariable*, *Verifikationsvariable* oder *Konstante* sein.

rung mit Nebenbedingungen ist, können die Bindungen auch bei der Optimierung weiterverwendet werden. Den letzten Schritt im Entwurfsprozess bildet die Bestimmung der technologischen Parameter. Dies wird ebenfalls als CSP behandelt, indem auch hier die Rollen der Variablen vertauscht werden. Nachdem die optimale Geometrie ermittelt und die Eigenschaften vorgegeben wurden, können die technologischen Parameter als unbekannt betrachtet werden, während Geometrie und Arbeitsraum vorgeben sind. Auch hier kann auf die bereits zur Analyse erarbeiteten Bindungen zurückgegriffen werden.

Damit der CSP-Löser mit dieser Flexibilität für Analyse und Synthese von PKM eingesetzt werden kann, wird ein Programmgerüst (Framework) benötigt, das die folgenden Eigenschaften haben sollte:

- Die Implementierung der Bindungen kann verwendet werden zur
  - Verifikation des Arbeitsraums (Abschnitt 4.2.2),
  - Berechnung des Arbeitsraums (Abschnitt 4.2.3),
  - Maßsynthese (Abschnitt 5.2),
  - Optimierung (Abschnitt 5.3),
  - Synthese der technologischen Parameter (Abschnitt 5.4).
- Das Programmgerüst kann um neue Intervallalgorithmen erweitert werden.
- Die Bindungen sind voneinander unabhängig, so dass sich verschiedene Bindungen kombinieren lassen.
- Neue Bindungen können auch nachträglich hinzufügt werden, ohne dass bestehende Bindungen überarbeitet werden müssen.
- Die Bindungsgleichungen für verschiedene Kriterien sind in einer Bibliothek organisiert und können anwendungsspezifisch zur Berücksichtigung der jeweils relevanten Eigenschaften kombiniert werden.
- Die Konfiguration kann unabhängig von den Bindungen eingestellt werden (z. B. über eine Konfigurationsdatei).

Um diese Anforderungen zu erfüllen, wird im Folgenden eine Aufteilung in Klassen vorgeschlagen.

5.5.1 Objektorientierte Analyse

Für das Programmgerüst wird durch die objektorientierte Analyse das Gesamtsystem CSP in Klassen aufgeteilt, die miteinander kommunizieren. Diese Klassen besitzen abstrakte Schnittstellen, die die prinzipiellen Kommunikationswege des Programmgerüsts beschreiben. Durch geeignete virtuelle Methoden der Klassen wird eine effektive Kapselung der individuellen Eigenschaften ermöglicht, die zu modularen und wiederverwendbaren Programmbausteinen führt.

Der prinzipielle Aufbau des Programmgerüsts ist in Abbildung 5.3 skizziert. Die Klasse `CSP-Löser` repräsentiert die verschiedenen Algorithmen, die in diesem und dem vorangegangenen Kapitel vorgestellt wurden. Zur Kommunikation mit dem Programmgerüst wird eine Methode `auswerten` vereinbart, die das System $\underline{\Phi}$ berechnet und die Bewertung

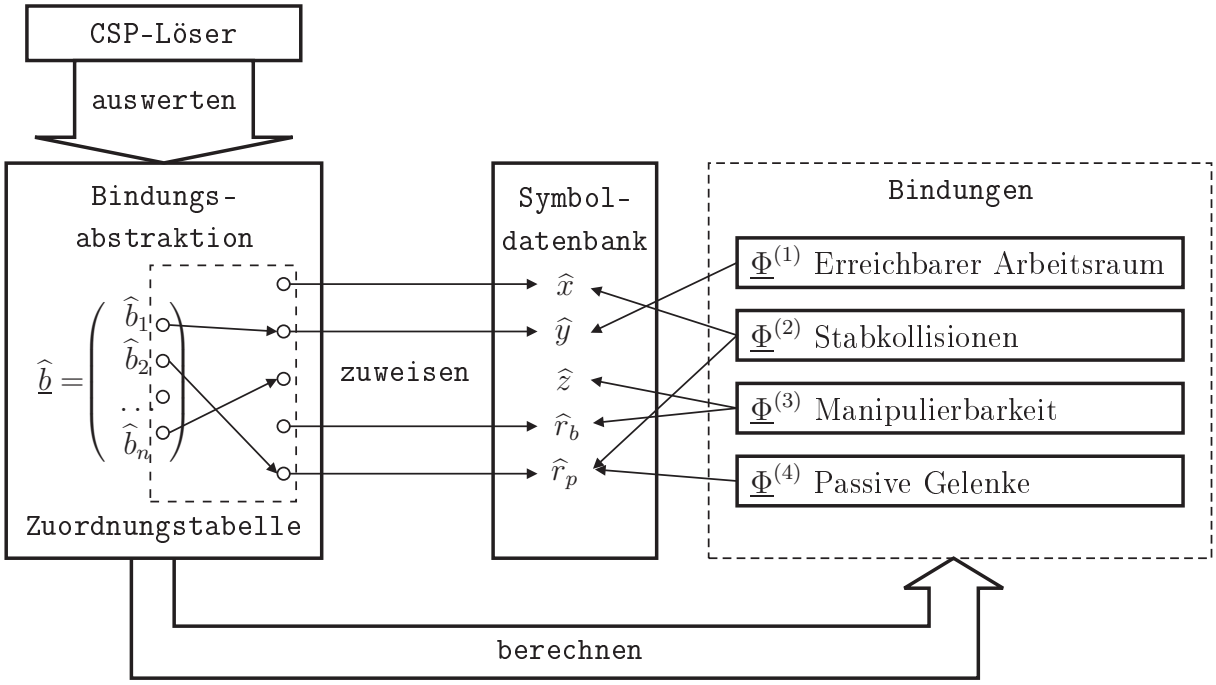


Abbildung 5.3: Aufbau des modularen Programmgerüsts für einen CSP-Löser

des Systems zurückliefert. Die Klasse für die `Bindungsabstraktion` enthält die eigentliche Beschreibung des Systems der Bindungsgleichungen $\underline{\Phi}$. Die `Bindungsabstraktion` enthält als wichtigste Komponenten eine Liste der `Bindungen` und eine dynamische zuweisbare `Zuordnungstabelle`. Die Bindungen entsprechen den in Abschnitt 4.4 und 5.2.1 beschriebenen Ungleichungen und alle in den Bindungen enthaltenden Variablen werden mit den Symbolen der `Symboldatenbank` gekoppelt. Die `Zuordnungstabelle` verknüpft für das jeweilige Problem die Box $\widehat{\underline{b}}^{\mathrm{T}} = [\widehat{\underline{c}}^{\mathrm{T}}, \widehat{\underline{v}}^{\mathrm{T}}]$ bestehend aus den Berechnungsvariablen $\widehat{\underline{c}}$ und den Verifikationsvariablen $\widehat{\underline{v}}$ mit der `Symboldatenbank`. Die Auswertung des Systems $\underline{\Phi}$ geschieht daher in zwei Schritten. Zunächst werden mithilfe der `Zuordnungstabelle` die aktuellen Werte der Box $\widehat{\underline{b}}$ der `Symboldatenbank` zugewiesen. Der zweite Schritt besteht darin, die Bindungen zu berechnen.

Die `Symboldatenbank` enthält Speicher für jede Variable, die in den Bindungen vorkommt. Daher kann die Symboldatenbank als schlichter Vektor $\widehat{\underline{s}} \in \mathbb{I}^{n_s}$ interpretiert werden, wobei n_s die Anzahl der Variablen ist. Bei einer variablen Anzahl von Gleichungen wird dadurch sichergestellt, dass alle Symbole genau einmal gespeichert werden. Die zentrale Speicherung der Variablen ist nötig, um zu gewährleisten, dass die in mehr als einer Gleichung auftretenden Variablen überall den gleichen Wert haben. Die Minimierung des Bedarfs an Arbeitsspeicher spielt bei dieser Maßnahme keine Rolle. Durch die `Symboldatenbank` werden die aktuellen Werte implizit in die Gleichungen eingesetzt und müssen nicht beim Funktionsaufruf als Parameter übergeben werden. Dies hat den Vorteil, dass bei der Implementierung der Gleichung nicht festgelegt werden muss, welche Rollen die Variablen spielen. Es ist daher möglich, über die `Zuordnungstabelle` erst zur Laufzeit auszuwählen, welche Variablen als Konstanten, Verifikations- oder Berechnungsvariablen fungieren. Weiterhin müssen die Werte der Box $\widehat{\underline{b}}$ pro Auswertung von $\underline{\Phi}$ nur einmal in die Datenbank

kopiert werden. Dies ist sehr effizient, da komplexe Probleme viele Bindungen enthalten und die Box $\widehat{\underline{b}}$ wenige Komponenten besitzt.

Den Symbolen werden zur Identifikation eindeutige Namen zugeordnet, die beim Aufbau der `Zuordnungstabelle` verwendet werden, um die Rolle der einzelnen Symbole festzulegen. Durch die dynamisch anpassbare Zuordnungstabelle wird die Rolle eines speziellen Symbols als Konstante, Berechnungs- oder Verifikationsvariable festgelegt.

Die einfache Klasse `Bindung` besitzt nur eine Funktion zur Auswertung der Vektorfunktion $\underline{\Phi}^{(i)} : \mathbb{I}^{n_s} \rightarrow \mathbb{I}^{n_i}$, wobei n_i die Anzahl der Funktionen in der Bindung ist. Bei der Initialisierung werden die für diese Bindung benötigten Variablen mit den Variablen in der `Symboldatenbank` verknüpft.[3] Damit kann sich eine a priori nicht bekannte Anzahl von Bindungen die gleichen Variablen in der Datenbank teilen. Gleichzeitig müssen die Werte für die Variablen nicht bei der Auswertung übergeben werden. Stattdessen verwendet die Bindung ohne zusätzlichen Aufwand die aktuellen Werte aus der `Symboldatenbank`. Die Auswertung der Bindung kann also ohne Parameter aufgerufen werden und liefert nur die Werte der Bindungsgleichungen zurück.

Die Zielfunktion der Optimierung kann als spezielle skalare Bindungsgleichung betrachtet werden. Die Verknüpfung mit der `Symboldatenbank` geschieht auf die gleiche Weise wie bei den `Bindungen`.

Die `Bindungsabstraktion` repräsentiert für den `CSP-Löser` das System der Ungleichungen $\underline{\Phi}$ und, sofern erforderlich, die Zielfunktion f. Die wichtigste Aufgabe der Klasse besteht darin, die vom `CSP-Löser` übergebene Box $\widehat{\underline{b}}$ speziell für die aktuelle Aufgabe auf die `Symboldatenbank` abzubilden. Dies geschieht durch die `Zuordnungstabelle`, welche die Komponenten der Box $\underline{b}$ in die `Symboldatenbank` überträgt.[4] Im direkten Anschluss daran werden die Bindungsgleichungen nacheinander ausgewertet.

Die `Zuordnungstabelle` kann aus einer Konfigurationsdatei erstellt werden, da die Symbole anhand ihrer Namen identifiziert werden können. Damit wird es möglich, die verschiedenen Aufgaben Verifikation eines gegebenen Arbeitsraums, Berechnung des Arbeitsraums, Maßsynthese und Optimierung mit der gleichen Implementierung der Bindungen zu behandeln. Dazu werden jeweils die vom `CSP-Löser` gesuchten Variablen mit dem Vektor $\widehat{\underline{b}}^{\mathrm{T}} = [\widehat{\underline{c}}^{\mathrm{T}}, \widehat{\underline{v}}^{\mathrm{T}}]$ verknüpft, während für die übrigen Variablen konstante Werte angenommen werden, die am Anfang der Berechnung in die `Symboldatenbank` geschrieben werden. Die Anzahl der Bindungen sowie deren Abhängigkeiten können als Detail vor dem `CSP-Löser` verborgen werden.

Die `Bindungsabstraktion` bewertet die Ergebnisse der Auswertung von $\underline{\Phi}$, d. h. die von den Bindungen zurückgelieferten numerischen Werte werden in die logischen Bewertungen *gültig*, *ungültig* und *unsicher* übersetzt.

In den vorangegangenen Abschnitten wurden die Algorithmen

- `Verifizieren`,
- `Berechnen`,

[3] In der Implementierung in C++ wird dies durch *Referenzen* realisiert [125]. Referenzen sind vergleichbar mit Zeigern, verhalten sich aber wie normale Variablen. Beim Anlegen einer Referenz wird einer bestehenden Speicherstelle in der `Symboldatenbank` lokal in der `Bindung` ein neuer Name zugeordnet.

[4] In der Implementierung in C++ wird die `Zuordnungstabelle` durch ein Feld von Zeigern abgebildet. Anhand der Indizes dieses Felds können die Komponenten von $\underline{b}$ den entsprechenden Variablen in der `Symboldatenbank` zugewiesen werden.

- `hybrider CSP-Löser`,
- `Paralleler CSP-Löser`,
- `Globaler Optimierer`,
- `Globaler Optimierer mit Nebenbedingungen`,

beschrieben, die als Ableitungen der Klasse `CSP-Löser` eingesetzt werden. Für die Implementierung lassen sich die Algorithmen `Verifizieren` und `Berechnen` als Spezialfälle des `hybriden CSP-Lösers` integrieren. Der Algorithmus `Berechnen` wird durch den `hybriden CSP-Löser` emuliert, wenn keine Verifikationsvariablen $\underline{v}$ zugewiesen werden und der `hybride CSP-Löser` verhält sich wie der Algorithmus `Verifizieren`, wenn keine Berechnungsvariablen $\underline{c}$ verwendet werden. In ähnlicher Weise wird die globale Optimierung ohne Nebenbedingungen als Spezialfall der Optimierung mit Nebenbedingungen betrachtet.

Die Algorithmen lassen sich mit der `Bindungsabstraktion` koppeln, ohne dass der Algorithmus zusätzlich Informationen über die Bindungen braucht. Die Auswertung von $\underline{\Phi}$ für eine Box $\widehat{\underline{b}}$ wird von der `Bindungsabstraktion` übernommen. Als Resultat erhält der `CSP-Löser` eine logische Bewertung der Box, denn für den Algorithmus sind die numerischen Werte der Bindungen irrelevant. Die Aufgabe des `CSP-Löser` besteht darin, die Listen $\mathcal{L}_\mathrm{T}$, $\mathcal{L}_\mathrm{S}$, $\mathcal{L}_\mathrm{F}$ und $\mathcal{L}_\mathrm{I}$ zu verwalten. Diese werden als verkettete Listen organisiert, so dass das Hinzufügen und Entfernen der Boxen bei beliebig langen Listen effizient durchgeführt werden kann. Die maximale Anzahl der Boxen in den Listen hängt nur vom verfügbaren Arbeitsspeicher ab.

5.5.2 Übersicht über mögliche Konfigurationen

Ein Vorteil des modularen Aufbaus besteht in der Möglichkeit, die `Zuordnungstabelle` der Symbole und die verwendeten Bindungen beliebig auszuwählen. In Tabelle 5.3 sind verschiedene Szenarien zusammengefasst, die in dieser Arbeit diskutiert werden. Dabei bezeichnet $\underline{g}$ die geometrischen und $\underline{t}$ die technologischen Parameter. Die Weltkoordinaten der Plattform setzten sich aus $\underline{y}^\mathrm{T} = [\underline{x}^\mathrm{T}, \underline{\theta}^\mathrm{T}]$ zusammen, wobei $\underline{x}$ die translatorischen Weltkoordinaten und $\underline{\theta}$ die Orientierungen beschreiben. Alle diese Aufgaben können mit dem hybriden CSP-Löser bearbeitet werden, wobei den Variablen $\underline{x}, \underline{\theta}, \underline{g}, \underline{t}$ in jedem Szenario unterschiedliche Rollen zugewiesen werden. Dazu wird die `Zuordnungstabelle` dynamisch zur Laufzeit des Programms aus einer Konfigurationsdatei erstellt. Es müssen daher nur die folgenden Informationen festgelegt beziehungsweise aus einer Liste von Optionen ausgewählt werden:

Tabelle 5.3: Verschiedene `Zuordnungstabellen` für den hybriden CSP-Löser

Aufgabe	$\underline{c}$	$\underline{v}$	konst.
Verifikation des Arbeitsraums	$\emptyset$	$\underline{x}, \underline{\theta}$	$\underline{g}, \underline{t}$
Berechnung des translatorischen Arbeitsraums	$\underline{x}$	$\emptyset$	$\underline{\theta}, \underline{g}, \underline{t}$
Berechnung des Gesamtorientierungsarbeitsraums	$\underline{x}$	$\underline{\theta}$	$\underline{g}, \underline{t}$
Maßsynthese für gegebene Arbeitsraum $\mathcal{W}_\mathrm{GO}$	$\underline{g}$	$\underline{x}, \underline{\theta}$	$\underline{t}$
Synthese technologischer Parameter	$\underline{t}$	$\underline{x}, \underline{\theta}$	$\underline{g}$

- verwendete Bindungen,
- Berechnungsvariablen $\underline{c}$ und deren Suchraum $\mathcal{X}_s$,
- Verifikationsvariablen $\underline{v}$ und die Verifikationsmenge $\mathcal{X}_v$,
- Werte der Konstanten,
- Genauigkeit $\varepsilon_v, \varepsilon_c$.

Für den Suchraum $\mathcal{X}_s$ wird in der Regel eine hinreichend große Box verwendet, d. h. es werden Intervalle für jede Komponente der Berechnungsvariablen $\underline{c}$ vorgegeben.

5.6 Ergebnisse

Die flexible Verwendung des CSP-Lösers und globalen Optimierers wird anhand der Maßsynthese und Optimierung der Maschinen SSM und *Linapod* demonstriert. Bei der Maßsynthese und Optimierung für SSM bzw. *Linapod* kommen die gleichen Bindungen zum Einsatz, die bereits im Kapitel 4 für die Berechnung des Arbeitsraums verwendet wurden.

5.6.1 Maßsynthese und Optimierung einer Gough-Plattform

Als Beispiel für die Maßsynthese beim SSM werden die folgenden Anforderungen betrachtet: Der Arbeitsraum soll ein Quader mit 200 mm Kantenlänge sein, wobei die Plattform in alle Richtungen 5.7° gedreht[5] werden kann; für die Verifikationsvariablen $\underline{v} = [x, y, z, \varphi, \theta, \psi]^{\mathrm{T}}$ wird also die Verifikationsmenge

$$\begin{aligned} \mathcal{X}_v \;=\; & \{x \in [-0.1, 0.1], y \in [-0.1, 0.1], z \in [0.0, 0.2], \\ & \varphi \in [-0.1, 0.1], \theta \in [-0.1, 0.1], \psi \in [-0.1, 0.1]\} \end{aligned}$$

angenommen, wobei die Längen x, y, z in Meter und die Winkel φ, θ, ψ in Radiant gemessen sind. Gesucht sind die Berechnungsvariablen $\underline{c} = [r_p, r_b, \Delta\alpha_p, \Delta\alpha_b]^{\mathrm{T}}$. Die Lösungsmenge ist also ein 4-dimensionaler Parameterraum, für den es keine triviale Visualisierung gibt. Der CSP-Löser ermittelt für die Lösungsmenge eine Approximation mit 214547 Boxen, die 7.2% des gesamten ursprünglichen Suchraums einnehmen. Die Rechenzeit für dieses Ergebnis beträgt ∼27 h auf einem einzelnen Rechner.

Die Lösungsmenge der Maßsynthese kann direkt als Suchraum für die freie globale Optimierung verwendet werden, da durch die Maßsynthese bereits sichergestellt wurde, dass

[5] In alle Richtungen bedeutet, dass *alle* Orientierungen erreicht werden können, die durch die KARDAN-Winkel φ, θ, ψ im angegebenen Winkelbereich [-0.1,0.1] rad generiert werden. Diese Beschreibung wurde der Einfachheit halber ausgewählt, da sie sich durch eine einzige Box beschreiben lässt und da sie näherungsweise einer Rotation von 5.7° um eine beliebige Achse entspricht. Das Verfahren kann genauso auf komplexere Beschreibungen von $\mathcal{X}_v$ und andere Parametrisierungen der Rotation angewendet werden.

Tabelle 5.4: Optimale Parameter des SSM für einen Gesamtorientierungsarbeitsraum von $200 \times 200 \times 200$ mm

Parameter:	r_p [m]	r_b [m]	$\Delta\alpha_b$ [rad]	$\Delta\alpha_p$ [rad]	Δz [m]
	0.38125	0.278125	0.78125	0.625	0.753125

alle gestellten Anforderungen erfüllt wurden. Als Beispiel für die Zielfunktion wird im Folgenden das Volumen der Maschine und der Abstand zwischen den plattformseitigen Anlenkpunkten verwendet. Damit ergibt sich die Zielfunktion zu

$$f(r_p, r_b, \Delta\alpha_b, \Delta\alpha_p, \Delta z) = r_p^2 + \Delta z\ r_b^2 + (\Delta\alpha_p - 1)^2 + (\Delta\alpha_b - 1)^2 \quad . \tag{5.16}$$

Mit dieser Zielfunktion und der Lösungsmenge der Maßsynthese ergeben sich für die optimalen geometrischen Parameter die Werte in Tabelle 5.4. Die Rechenzeit auf einem Pentium IV mit 3.2 GHz beträgt 56 min. Durchschnittlich werden dabei 57500 Auswertungen der Bindungen pro Sekunde durchgeführt. Dies belegt die hohe Effizienz der Implementierung. Durch eine Kombination von Maßsynthese und Optimierung kann auch hier die Rechenzeit signifikant reduziert werden.

Nachdem die optimale Geometrie für die spezifizierte Aufgabe des SSM gefunden wurde, stellt sich die Frage, ob die maximal möglichen Werte für technologische Parameter tatsächlich ausgenutzt werden. Daher werden nun die technologischen Parameter als gesuchte Parameter $\underline{c}$ angenommen, während die oben ermittelte optimale Geometrie verwendet wird. Anhand des `hybriden CSP-Lösers` wird über dem ganzen Arbeitsraum geprüft, welche technologischen Parameter zu gültigen Varianten der Maschine führen. Die Verifikationsvariablen werden daher wie bei der Maßsynthese zu $\underline{v} = [x, y, z, \varphi, \theta, \psi]^{\mathrm{T}}$ und für die Verifikationsmenge wird der vorgegebene Arbeitsraum $\mathcal{W}_{\mathrm{GO}}$ als

$$\begin{aligned} \mathcal{X}_v \;=\; & \{x \in [-0.1, 0.1], y \in [-0.1, 0.1], z \in [0.0, 0.2], \\ & \varphi \in [-0.1, 0.1], \theta \in [-0.1, 0.1], \psi \in [-0.1, 0.1]\} \end{aligned}$$

verwendet. Zunächst wird der maximal nötige Schwenkwinkel für die Gelenke an der Basis des SSM berechnet. Dazu wird $\underline{c} = [\beta]$ gesetzt und der Suchraum wird zu $\mathcal{X}_s = \{\beta \in [0, \beta_{\max}]\}$ gewählt, wobei für $\beta_{\max} = 45°$ angenommen wird. Der erforderliche Wert für den Schwenkwinkel ergibt $\beta = 21.56°$.

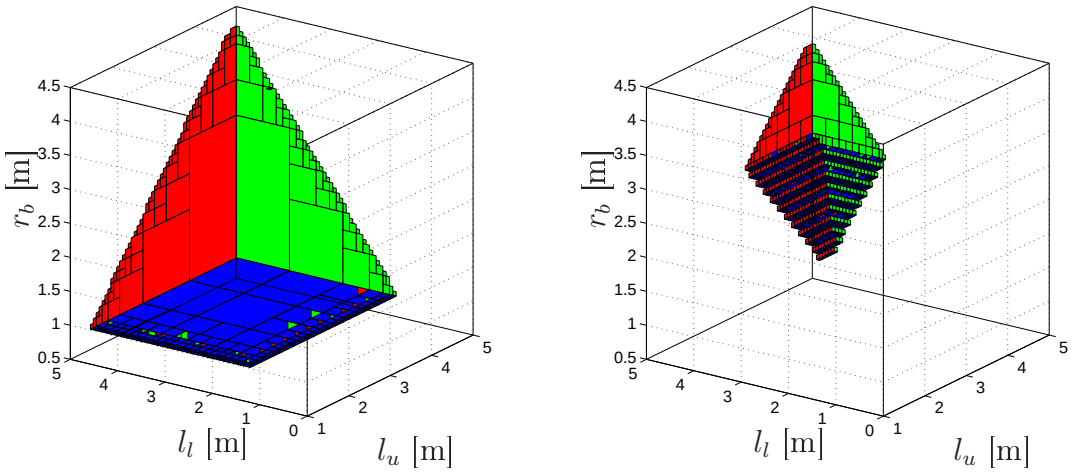
In ähnlicher Weise können die tatsächlich benötigten Hübe der Aktuatoren bestimmt werden. Dazu werden die minimale Länge des Aktuators $s_{\min}$ und die maximale Länge $s_{\max}$ als Berechnungsvariablen gewählt $\underline{c} = [s_{\min}, s_{\max}]^{\mathrm{T}}$. Die Verifikationsvariablen werden wie im vorigen Beispiel verwendet. Es stellt sich heraus, dass der Hub der Schubgelenke von $[1, 2]$ m auf $[0.684, 1.046]$ m reduziert werden kann.

5.6.2 Maßsynthese und Optimierung des *Linapod*

Im folgenden Abschnitten wird die Maßsynthese sowie die Optimierung für die PKM *Linapod* betrachtet. Sofern nicht anders angegeben, besitzen alle Parameter den nominellen Wert nach Tabelle 2.4 (Seite 29).

Für dieses Beispiel werden die folgenden Anforderungen an die Maschine gestellt. Der Arbeitsraum in der xy-Ebene soll mindestens eine Größe von 600×600 mm besitzen, wobei ein Schwenken der Plattform von 30° in jede beliebige Richtung möglich ist. Weiterhin wird die Manipulierbarkeit durch ein maximales Übersetzungsverhältnis von $j_{\max} = 4$ begrenzt. Die Genauigkeit für diese Berechnung beträgt $\varepsilon = 0.05$.

Gesucht werden die Längen der oberen und unteren Beine l_u, l_l sowie der Radius des Maschinenrahmens r_b. Der Vektor der Berechnungsvariablen ergibt sich also zu $\underline{c} = [l_u, l_l, r_b]^{\mathrm{T}}$. Wird nur der erreichbare Arbeitsraum ohne weitere Anforderungen betrachtet, ergibt sich



a) ohne Begrenzung der Manipulierbarkeit

b) mit einem maximalen Übersetzungsverhältnis von $j_{\max} = 4$

Abbildung 5.4: Zulässige Varianten der PKM *Linapod* für einen vorgegebenen Arbeitsraum von $600 \times 600\,\text{mm}$ bei einem maximalen Schwenkwinkel von $\varphi_{\max} = 30°$ für die Parameter l_l, l_u, r_b.

die in Abbildung 5.4a dargestellte Lösungsmenge. In Abbildung 5.4b wurde zusätzlich die Manipulierbarkeit des Systems durch einen maximalen Wert von $j_{\max} = 4$ begrenzt. Der Vergleich der Diagramme in Abbildung 5.4 zeigt deutlich, dass die Manipulierbarkeit einen großen Einfluss auf die möglichen Lösungen der Maßsynthese hat.

Die Ergebnisse der Maßsynthese können dann als Suchraum des globalen Optimierers verwendet werden. Die Zielfunktion wurde so gewählt, dass die Abmessungen der Maschine minimiert werden. Wird unterstellt, dass der Rahmen der Maschine die Gestalt eines Zylinders hat, ergibt sich das Volumen zu

$$f(\underline{g}) = f(l_l, r_b) = \pi l_l r_b^2. \tag{5.17}$$

Mit dieser Zielfunktion wurden die in Tabelle 5.5 angegebenen Parameter für die optimale Geometrie ermittelt. Für die praktische Realisierung der Maschine ist es durchaus nützlich, ein Intervall für die optimalen Parameter zu kennen, denn jede Variante aus diesem Intervall verfügt über die geforderten Eigenschaften. Wenn für die nominellen Parameter der optimalen Maschine der Mittelpunkt dieser Intervalle gewählt wird und die Toleranz kleiner als die halbe Intervallbreite vorgegeben wird, dann erfüllt die Maschine selbst unter Fertigungsfehlern garantiert alle geforderten Eigenschaften.

Tabelle 5.5: Geometrische Parameter der optimalen Variante des *Linapod* für einen Arbeitsraum von $600 \times 600\,\text{mm}$ und einen Schwenkwinkel der Plattform von $\varphi_{\max} = 30°$

Parameter	Wertebereich [m]	Beschreibung
l_l	$[2.39844, 2.46875]$	Länge der unteren Beine
l_u	$[2.45117, 2.45996]$	Länge der oberen Beine
r_b	$[2.0293, 2.04688]$	Radius des Maschinenrahmens

Die Rechenzeit für die Lösung des CSP beträgt etwa 45 min während die integrierte Implementierung des globale Optimierer mit Nebenbedingungen die Lösung bereits nach 2 min ermitteln kann. Dies belegt, wie groß das Potential zur Effizienzsteigerung durch Kombination des CSP-Lösers und des globalen Optimierers ist.

Kapitel 6

Zusammenfassung und Ausblick

In dieser Arbeit werden Methoden zur Analyse, Synthese und Optimierung von Werkzeugmaschinen mit Parallelkinematik untersucht. Für die exakte lokale Analyse wird die kinetostatische Methode verwendet. Ein modularer Baukasten ermöglicht die Modellierung der Kinematik und Dynamik einer großen Klasse räumlicher PKM mit sechs Freiheitsgraden. Die Maschine wird dazu in die drei intelligenten Module Plattform, Rahmen und Beine zerlegt. Durch eine Parametrisierung der einzelnen Module kann eine große Anzahl verschiedener PKM aus dem Baukasten zusammengestellt werden. Der Zusammenbau zum Gesamtsystem für die direkte und inverse Kinematik erfolgt automatisch, und es entsteht ein generisches Modell. Anhand dieses Modells lässt sich die Maschine mit allgemeinen Verfahren untersuchen.

Für die Analyse des kinetostatischen Modells wird ein neuer Algorithmus vorgestellt, der durch die Berechnung von Schnittkräften das Modell bezüglich der Geometrie vollständig linearisiert. Der Algorithmus ermöglicht auch die Linearisierung bezüglich einiger Parameter, die zur Vereinfachung in die Bindungsgleichungen idealisiert wurden. Beispielsweise kann mit der vorgestellten Methode der Abstand der Achsen in einem Kardangelenk untersucht werden. Das Modell muss für die Linearisierung nicht um zusätzliche Elemente erweitert werden. Der Algorithmus ist nicht spezifisch für die kinetostatische Methode, sondern kann mit jedem Verfahren angewendet werden, das eine Ermittlung der Schnittkräfte ermöglicht. Die Komplexität des Algorithmus ist linear in der Anzahl der betrachteten Körper und Gelenke.

Das linearisierte Modell wird verwendet, um die Empfindlichkeit der PKM bezüglich Fertigungs- und Montagefehlern zu untersuchen. Weiterhin wird die linearisierte Form benutzt, um die Steifigkeit der Maschine zu ermitteln. Dabei können lineare Elastizitäten in allen Bauteilen berücksichtigt werden. Anhand von Beispielen wird demonstriert, dass PKM eine systeminhärente Empfindlichkeit gegenüber geometrischen Fehlern besitzen. Die Ursache dafür liegt in der großen Anzahl verwendeter Bauteile, aus denen viele mögliche Fehlerquellen resultieren können. Eine wichtige Anwendung des linearisierten Modells ist daher die Kalibrierung von PKM.

Die kinetostatische Methode erlaubt eine präzise Analyse der Eigenschaften an endlich vielen diskreten Punkten im Arbeitsraum einer PKM. Für die Berechnung des Arbeitsraums und die Maßsynthese von PKM ist es jedoch wichtig, dass die Eigenschaften an allen Punkten des Arbeitsraums mit einem verifizierten Verfahren bewertet werden. Durch die Formulierung der Eigenschaften einer PKM als *Constraint Satisfaction Problem* (CSP)

wird ein neuer Ansatz aufgezeigt, mit dem der ganze Entwicklungsprozess von PKM erfasst werden kann. Der Algorithmus zur Lösung des CSP basiert auf der Intervallanalyse und ist daher numerisch sehr robust. Die Gültigkeit der berechneten Eigenschaften beschränkt sich nicht auf ein Gitter diskretisierter Punkte. Die Untersuchung mit der Intervallanalyse beweist, dass die Eigenschaften an *jedem* Punkt des Arbeitsraums gültig sind. Möglicherweise auftretende Rundungsfehler werden berücksichtigt und sicher abgeschätzt. Eine Implementierung für parallele Computer erlaubt auch bei großen Problemen eine akzeptable Laufzeit und ist für die kommende Generation von Computern mit vielen Prozessorkernen prädestiniert.

Die Formulierung als CSP wird verwendet, um den Arbeitsraum von PKM zu berechnen. Die vorgeschlagenen Kriterien erlauben die Überprüfung auf kinematische Erreichbarkeit, Freiheit von Singularitäten und Autokollisionen. Es wird verifiziert, ob die Begrenzungen sowohl der aktiven als auch der passiven Gelenke eingehalten werden und ob eine gegebene Schranke für die Übersetzungsverhältnisse nicht überschritten wird. Die Methoden werden auf die Berechnung des translatorischen Arbeitsraums sowie des Gesamtorientierungsarbeitsraums angewendet. Die Beispiele belegen die praktische Verwendbarkeit für relevante räumliche PKM, und die berechneten Arbeitsräume stimmen mit anderen Studien überein.

Durch eine konsequente Weiterentwicklung der Arbeitsraumberechnung wird die Maßsynthese für PKM als CSP formuliert. Dabei werden die Bindungen zur Berechnung des Arbeitsraums ohne Modifikation für die Synthese und Optimierung weiterverwendet. Der Fluss der Berechnung von gegebener Geometrie zum gesuchten Arbeitsraum wird vollständig umgekehrt, so dass alle gültigen Varianten der Geometrie einer PKM bestimmt werden, die über einen vorgegebenen Arbeitsraum verfügen. Es kann also ohne zusätzlichen Aufwand von der Arbeitsraumberechnung zur Maßsynthese übergegangen werden, wobei sämtliche bei der Analyse betrachteten Eigenschaften für die Synthese berücksichtigt werden. Damit gelingt die selten realisierte Verknüpfung der Analyse mit der Maßsynthese und folglich der Entwurf von PKM für vorgegebene Eigenschaften.

Für den optimalen Entwurf wird der Schwerpunkt auf die Erfüllung gegebener Eigenschaften gelegt. Dies stellt gegenüber der verbreiteten Maximierung der Eigenschaften von PKM einen Paradigmenwechsel dar. Als Zielfunktion tritt daher eine Bewertung der Kosten oder der äußeren Abmaße an die Stelle einer gewichteten Summe von Gütekriterien. Ein wichtiges Ergebnis dieser Arbeit ist ein neuer Algorithmus, der das für die Maßsynthese verwendete CSP als Nebenbedingung für einen Intervall-Algorithmus zur globalen Optimierung verwendet und damit unter den gültigen Varianten die optimale Maschine auswählt. Die Intervallanalyse garantiert, das globale Optimum im Suchraum zu finden. Gleichzeitig werden verifizierte Schranken für die Designparameter ermittelt, die dieses Optimum enthalten.

Als wichtigstes Ergebnis dieser Arbeit wird ein Programmgerüst vorgestellt, das alle CSP-Löser umfasst und sich gleichermaßen für die Analyse, Synthese und Optimierung von PKM eignet. Damit gelingt es, eine neue Verbindung zwischen Analyse und Synthese zu schaffen und beide Methoden synergetisch in einem methodischen Rahmen zu verbinden. Einem Konstrukteur steht damit ein integriertes Werkzeug zur Verfügung, das den gesamten Entwicklungsprozess neuer Maschinen unterstützt. Beginnend von einer vorläufigen Berechnung des Arbeitsraums, über eine verfeinerte Betrachtung der Eigenschaften bis hin zur Maßsynthese und Optimierung werden alle Schritte abgedeckt. Der modulare Aufbau

erlaubt es, die für die Analyse erstellten Bindungen für die Synthese wiederzuverwenden und nachträglich weitere Anforderungen zu ergänzen. Die Implementierung mit der Intervallanalyse ersetzt die übliche Diskretisierung des Arbeitsraums und bietet stattdessen ein verifiziertes Verfahren, das die Erfüllung der Anforderungen an jedem Punkt des Arbeitsraums garantiert. Eine ähnliche Methode kommt bei der globalen Optimierung zum Einsatz, so dass garantiert das globale Optimum gefunden wird.

In zukünftigen Forschungsarbeiten sollten Erweiterungen am CSP-Löser vorgenommen werden. Hierzu existieren verschiedene Verfahren in der Theorie der globalen Optimierung mit der Intervallanalyse, die noch nicht auf PKM angewendet wurden. Zur Reduzierung der Rechenzeit sollte untersucht werden, wie durch den Algorithmus geeignete Heuristiken automatisch ausgewählt werden können. Weiterhin sollte untersucht werden, wie sich Computeralgebra und Intervallanalyse enger verzahnen lassen, so dass das Aufstellen der Bindungen stärker automatisiert wird. Mit einer verbesserten Aufbereitung der Bindungen könnten auch weitere Eigenschaften analysiert und synthetisiert werden. Beispielsweise wäre eine Vorgabe des dynamischen Verhaltens von praktischem Interesse.

Die hier für PKM entwickelten Werkzeuge können auch beim Entwurf von anderen Mechanismen hilfreich sein. Für Seilroboter konnten bereits Bindungen zur Berechnung des Arbeitsraums formuliert werden. Daher sollte erforscht werden, inwiefern sich die Techniken zur Analyse und Synthese bei dieser Art von Robotern anwenden lassen. Eine Anwendung der vorgestellten Methode zur Analyse und Maßsynthese auf seriellen Robotern erscheinen ebenfalls erfolgversprechend.

Mit den monetären Kosten als Zielfunktion für die Optimierung ließe sich kundenspezifisch die PKM entwerfen, die bei einem minimalen Preis alle gestellten technischen Anforderungen erfüllt. Daher sollten die Fertigungskosten von PKM für gegebene technologische und geometrische Parameter untersucht werden. Dazu müssten die Herstellungs- und Montagekosten zusammengestellt und bewertet werden.

Literaturverzeichnis

[1] Rasim I. Alizade, Nazim R. Tagiyev, and Joseph Duffy. A forward and reverse displacement analysis of a 6-dof in-parallel manipulator. *Mechanism and Machine Theory*, 29(1):115–124, 1993.

[2] R. Altenburger. Calibration and parameter estimation on PKM – a statistical approach with application. In Reimund Neugebauer, editor, *Proceedings of the 4th Chemnitz Parallel Kinematics Seminar*, pages 151–163, Chemnitz, Germany, 2004.

[3] Patricia Ben-Horin and Moshe Shoham. Singularity analysis of a class of parallel robots based on Grassmann-Cayley algebra. *Mechanism and Machine Theory*, 41:958–970, 2006.

[4] H. Böhler, J. O. Hestermann, and N. Plischke. Leistungen und Grenzen des PARALIX – eine innovative Werkzeugmaschine. In *Fortschritt-Berichte VDI, Reihe 2: Fertigungstechnik Band 620*, 2002.

[5] Christian Bliek, Peter Spellucci, Luís N. Vicente, Arnold Neumaier, Laurent Granvilliers, Eric Monfroy, Frederic Benhamou, Etienne Huens, Pascal Van Hentenryck, Djamila Sam-Haroud, and Boi Faltings. COCONUT deliverable D1: Algorithms for solving nonlinear constrained and optimization problems: The state of the art. Technical report, The COCONUT Project, 2001.

[6] C. R. Boër, L. Molinari-Tosatti, and K. S. Smith, editors. *Parallel Kinematic Machines – Theoretical Aspects And Industrial Requirements*. Springer, 1999.

[7] O. Bottema and B. Roth. *Theoretical kinematics*. Dover Publications, New York, 1979.

[8] Tim Boye, Alexander Verl, and Andreas Pott. Optimal tolerance, model and pose selection for calibration of parallel manipulators. In *37th International Symposium on Robotics*, Munich, Germany, 2006.

[9] C. Brecher and F. Hoffmann. Multi-criteria comparison of standardised kinematic structures for machine tools. In Reimund Neugebauer, editor, *Proceedings of the 5th Chemnitz Parallel Kinematics Seminar*, pages 65–82, Chemnitz, Germany, 2006.

[10] Cornel Brisan, Daniel Franitza, and Manfred Hiller. Modelling and analysis of errors for parallel robots. In *Proceedings of the Kolloquium of SFB 562*, pages 83–96, Braunschweig, Germany, 2002.

[11] Martin Brünger-Koch. Motion parameter tuning and evaluation for the DLR automotive simulator. In *Proceedings of Driving Simulator Conference*, Orlando, Florida, 2005.

[12] Tobias Bruckmann, Andreas Pott, Daniel Franitza, and Manfred Hiller. A modular controller for redundantly actuated tendon-based Stewart platforms. In *1st European Conference on Mechanism Science*, Obergurgl, Austria, 2006.

[13] Tobias Bruckmann, Andreas Pott, and Manfred Hiller. Calculating force distributions for redundantly actuated tendon-based Stewart platforms. In *Advances in Robot Kinematics*, pages 403–412, Ljubljana, Slovenia, 2006. Springer-Verlag.

[14] K. L. Cappel. Motion simulator. Technical report, US Patent No. 3,295,224, January 3, 1967.

[15] Damien Chablat, Phipippe Wenger, Félix Majou, and Jean-Pierre Merlet. An interval analysis based study for the design and the comparison of 3-DOF parallel kinematic machines. *International Journal of Robotics Research*, 2004.

[16] Damien Chablat, Phipippe Wenger, and Jean-Pierre Merlet. Workspace analysis of the orthoglide using interval analysis. In *Advances in Robot Kinematics*, pages 397–406, Caldes de Malavella, Spain, 2002. Kluwer Academic Publishers.

[17] Chair for Operating Systems, RWTH-Aachen, University. MP-MPICH – User Documentation & Technical Notes. Technical report, 2005.

[18] Kenneth W. Chase, William H. Greenwood, Bruce G. Loosli, and Loren F. Hauglund. Least cost tolerance allocation for mechanical assemblies with automated process selection. *Manufactoring Review*, 3(1):49–59, 1990.

[19] Raymond Clavel. Delta, a fast robot with parallel geometry. In *18th Int. Symp. on Industrial Robot*, pages 91–100, 1988.

[20] COPRIN. A C++ algorithm library of interval analysis for equation systems. Technical report, INRIA, Sophia-Antipolis, France, 2003.

[21] COPRIN. The maple interface for ALIAS. Technical report, INRIA, Sophia-Antipolis, France, 2003.

[22] Nicholas G. Dagalakis, James S. Albus, Ben-Li Wang, Joseph Unger, and James D. Lee. Stiffness study of a parallel link robot crane for shipbuilding applications. *Transactions of the ASME, Journal of Mechanical Design*, 111(3):183–193, August 1989.

[23] David Daney. Optimal measurement configurations for Gough platform calibration. *IEEE Int. Conf. on Robotics and Automation*, 1:147–152, 2002.

[24] David Daney, Yves Papegay, and Arnold Neumaier. Interval methods for certification of the kinematic calibration of parallel robots. In *Proceedings of the IEEE International Conference on Robotics & Automation*, 2004.

[25] Anjan Kumar Dash, Song Huat Yeoy, Guilin Yangz, and I-Ming Cheny. Workspace analysis and singularity representation of three-legged parallel manipulators. In *Seventh International Conference on Control, Automation, Robotics And Vision*, pages 962–967, Singapore, 2002.

[26] J. Denavit and R. Hartenberg. A kinematic notation for lower pair mechanisms based on matrices. *Journal of Applied Mechanics*, 77, 1955.

[27] B. Denkena, H.-C. Möhring, E. Bedhief, and G. Günther. Universal method for calibration of parallel and hybrid kinematics. In Reimund Neugebauer, editor, *Proceedings of the 4th Chemnitz Parallel Kinematics Seminar*, pages 71–87, Chemnitz, Germany, 2004.

[28] P. Dietmaier. The Stewart-Gough platform of general geometry can have 40 real postures. In *Advances in Robot Kinematics*, pages 7–16, Salzburg, Austria, 1998. Kluwer Academic Publishers.

[29] Peter Eberhard. *Zur Mehrkriterienoptimierung von Mehrkörpersystemen.* Fortschritt-Berichte VDI, Reihe 11, Nr. 227. VDI Verlag, Düsseldorf, 1996.

[30] Bashar. S. El-Khasawneh and Placid. M. Ferreira. Computation of stiffness and stiffness bounds for parallel link manipulators. *International Journal of Machine Tools and Manufacture*, 39:321–342, 1998.

[31] Shiqing Fang. *Design, Modeling and Motion Control of Tendon-Based Parallel Manipulators.* Fortschritt-Berichte VDI, Reihe 8, Nr. 1076. VDI Verlag, Düsseldorf, 2005.

[32] Rainer Gloess. Nanometer precision for positioning mechanism with long travel range. In *1st European Conference on Mechanism Science*, Obergurgl, Austria, 2006.

[33] Clément Gosselin. Stiffness mapping of parallel manipulators. *IEEE Transactions on Robotics and Automation*, 6(3):377–382, 1990.

[34] Clément Gosselin. Optimum design of robotic manipulators using dexterity indices. *Robotics and Autonomous Systems*, 9(4):213–226, 1992.

[35] Clément Gosselin and Jorge Angeles. Singularity analysis of closed-loop kinematic chains. *IEEE Transactions on Robotics and Automation*, 4:281–290, 1990.

[36] V. Eric Gough. Contribution to discussion of papers on research in automobile stability, control and tyre performance. *Proc. Auto Div. Inst. Mech. Eng.*, 1956.

[37] R. Hammer, M. Hocks, U. Kulisch, and D. Ratz. *Numerical Toolbox for Verified Computing I: Basic Numerical Problems.* Springer-Verlag, Heidelberg, New York, 1993.

[38] Eldon Hansen. *Global Optimization using Interval Analysis.* Marcel Dekker, Inc., 1992.

[39] Eldon Hansen and G. William Walster. *Global Optimization using Interval Analysis.* Marcel Dekker, Inc., 2004.

[40] Fang Hao and Jean-Pierre Merlet. Multi-criteria optimal design of parallel manipulators based on interval analysis. *Mechanism and Machine Theory*, 40(2):157–171, February 2005.

[41] A. M. Hay and J. A. Snyman. The optimal synthesis of parallel manipulators for desired workspaces. In *Advances in Robot Kinematics*, pages 337–346, Caldes de Malavella, Spain, 2002. Kluwer Academic Publishers.

[42] A. M. Hay and J. A. Snyman. Optimal synthesis for a continuous prescribed dexterity interval of a 3-dof parallel planar manipulator for different prescribed output workspaces. *International Journal For Numerical Methods In Engineering*, 2006.

[43] Matthias Hebsacker. *Entwurf und Bewertung Paralleler Werkzeugmaschinen – das Hexaglide*. PhD thesis, ETH Zürich, 2000.

[44] U. Heisel and I. Esteban. DFG SPP 1099 - Production machines with parallel kinematics. In Reimund Neugebauer, editor, *Proceedings of the 5th Chemnitz Parallel Kinematics Seminar*, pages 13–38, Chemnitz, Germany, 2006.

[45] Thomas Heyden. *Bahnregelung eines seilgeführten Handhabungssystems mit kinematisch unbestimmter Lastführung*. Fortschritt-Berichte VDI, Reihe 8, Nr. 1100. VDI Verlag, Düsseldorf, 2006.

[46] Manfred Hiller, Shiqing Fang, Sonja Mielczarek, Richard Verhoeven, and Daniel Franitza. Design, analysis and realization of tendon-based parallel manipulators. *Mechanism and Machine Theory*, 40(4):429–445, April 2005.

[47] Manfred Hiller and Andrés Kecskeméthy. Equations of motion of complex multibody systems using kinematical differentials. *Transactions of the Canadian Society of Mechanical Engineering*, 13(4):113–121, 1989.

[48] Manfred Hiller, Andrés Kecskeméthy, and C. Woernle. A loop-based kinematical analysis of complex mechanisms. In *Design Engineering Technical Conference,*. American Society of Mechanical Engineers, October 1986.

[49] J. M. Hollerbach and C. W. Wampler. The calibration index and taxonomy for robot kinematic calibration methods. *The International Journal of Robotics Research*, 15, 1996.

[50] Manfred L. Husty. An algorithm for solving the direct kinematic of Stewart-Gough-type platforms. *Mechanism and Machine Theory*, 31(4):365–380, 1996.

[51] Manfred L. Husty, Adolf Karger, Hans Sachs, and Waldemar Steinhilper. *Kinematik und Robotik*. Springer-Verlag, 1997.

[52] Manfred L. Husty, Sonja Mielczarek, and Manfred Hiller. Constructing an overconstrained planar 4RPR manipulator with maximal forward kinematics solution set. In *Proceedings of the RAAD'01 Robotics in the Alpe-Adria-Danube Region*, Vienna, Austria, 2001.

[53] Manfred L. Husty, Sonja Mielczarek, and Manfred Hiller. A redundant spatial Stewart-Gough platform with maximal forward kinematics solution set. In *Advances in Robot Kinematics*, pages 147–154, Caldes de Malavella, Spain, 2002. Kluwer Academic Publishers.

[54] Carlo Innocenti and Vincenzo Parenti-Castelli. Closed-form direct position analysis of a 5-5 parallel mechanism. *ASME Journal of Mechanical Design*, 115:515–521, 1993.

[55] C. C. Iurascu and F. C. Park. Geometric algorithms for closed chain kinematic calibration. *IEEE Int. Conf. on Robotics and Automation*, pages 1752–1757, 1999.

[56] Leonardo Jelenkovic and Leo Budin. Error analysis of a Stewart platform based manipulator. In *6th International Conference on Intelligent Engineering Systems*, Opatija, 2002.

[57] S. Ji, X. Li, Y. Ma, and H. Cai. Optimal tolerance allocation based on fuzzy comprehensive evaluation and genetic algorithm. *International Journal of Advanced Manufacturing Technology*, 16:461–468, 2000.

[58] Yan Jin and I.-Ming Chen. Effects of constraint errors on parallel manipulators with decoupled motion. *Mechanism and Machine Theory*, 41:912–928, 2006.

[59] Andrés Kecskeméthy. *Objektorientierte Modellierung der Dynamik von Mehrkörpersystemen mit Hilfe von Übertragungselementen*. Fortschritt-Berichte VDI, Reihe 20, Nr. 88. VDI Verlag, Düsseldorf, 1993.

[60] Andrés Kecskeméthy. *MOBILE – User's guide and reference manual*. Fachgebiet Mechatronik, University Duisburg-Essen, Germany, 1994.

[61] W. Khalil and S. Besnard. Self calibration of Stewart-Gough parallel robots without extra sensors. *IEEE Transactions on Robotics and Automation*, 15(6):1116–1121, 1999.

[62] Han S. Kim and Yong J. Choi. The kinematic error bound analysis of the Stewart platform. *Journal of Robotic Systems*, 17(1):63–73, 2000.

[63] J. Kirchner. *Mehrkriterielle Optimierung von Parallelkinematik*. Verlag Wissenschaftliche Skripten, Zwickau, 2001.

[64] R. Klatte, U. Kulisch, A. Wiethoff, C. Lawo, and M. Rauch. *C-XSC – A C++ Class Library for Extended Scientific Computing*. Springer-Verlag, Berlin, New York, 1993.

[65] C. A. Klein and B. E. Blaho. Dexterity measures for the design and control of kinematically redundant manipulators. *The International Journal of Robotics Research*, 6(2):72–83, 1987.

[66] Olaf Knüppel. PROFIL / BIAS – A fast interval library. *Computing*, 53:277–287, 1994.

[67] Olaf Knüppel. Profil/Bias v2.0. Technical Report 99.1, Technische Universität Hamburg - Harburg, Technische Informatik III, Hamburg, Germany, 1999.

[68] Lars Kübler, Christoph Henninger, and Peter Eberhard. Multi-criteria optimization of a hexapod machine. *Multibody System Dynamics*, 14:225–250, 2005.

[69] Daniel Lazard. On the representation of rigid-body motions and its application to generalized platform manipulators. In J. Angeles, P. Kovacs, and G. Hommel, editors, *Computational Kinematics*, pages 175–182. Kluwer Academic Publishers, 1993.

[70] Oliver Lenord, Shiqing Fang, Daniel Franitza, and Manfred Hiller. Numerical linearisation method to efficiently optimize the oscillation damping of an interdisciplinary system model. *Multibody System Dynamics*, 10:201–217, 2003.

[71] Yangmin Li and Qingsong Xu. Kinematic analysis of a 3-PRS parallel manipulator. *Robotics and Computer-Integrated Manufacturing*, 2006.

[72] Thomas Maier. *Bahnsteuerung eines seilgeführten Handhabungssystems.* Fortschritt-Berichte VDI, Reihe 8, Nr. 1047. VDI Verlag, Düsseldorf, 2004.

[73] Jean-Pierre Merlet. Parallel manipulators part 2: Theory – Singular configurations and Grassmann geometry. Technical Report 791, Institut National de Recherche en Informatique et en Automatique, Sophia-Antipolis (France), February 1988.

[74] Jean-Pierre Merlet. Singular configurations of parallel manipulators and Grassmann geometry. *The International Journal of Robotics Research*, 8(5):45–56, October 1989.

[75] Jean-Pierre Merlet. Designing a parallel manipulator for a specific workspace. Technical Report 2527, Institut National de Recherche en Informatique et en Automatique, Sophia-Antipolis (France), April 1995.

[76] Jean-Pierre Merlet. The importance of optimal design for parallel structures. In *Parallel Kinematic Machines*, pages 99–110, London, 1999. Springer-Verlag.

[77] Jean-Pierre Merlet. *Parallel Robots.* Solid Mechanics and its Application. Kluwer Academic Publishers, Dordrecht, 2000.

[78] Jean-Pierre Merlet. Analysis of the influence of wires interference on the workspace of wire robots. In *Advances in Robot Kinematics*, pages 211–218, Sestri Levante, Italy, 2004. Kluwer Academic Publishers.

[79] Jean-Pierre Merlet. Guaranteed in-the-workspace improved trajectory/surface/volume verification for parallel robots. In *IEEE International Conference on Robotics and Automation*, New Orleans, USA, 2004.

[80] Jean-Pierre Merlet. Solving the forward kinematics of a Gough-type parallel manipulator with interval analysis. *The International Journal of Robotics Research*, 23(3):221–235, March 2004.

[81] Jean-Pierre Merlet. Computing the worst case accuracy of a PKM over a workspace or a trajectory. In Reimund Neugebauer, editor, *Proceedings of the 5th Chemnitz Parallel Kinematics Seminar*, pages 83–98, Chemnitz, Germany, 2006.

[82] Jean-Pierre Merlet. Jacobian, manipulability, condition number, and accuracy of parallel robots. *Journal of Mechanical Design*, 128(1):199–206, January 2006.

[83] Jean-Pierre Merlet and Peter Donelan. On the regularity of the inverse jacobian of parallel robots. In *Advances in Robot Kinematics*, pages 41–48, Ljubljana, Slovenia, 2006. Springer-Verlag.

[84] Message Passing Interface Forum. MPI: A message-passing interface standard. Technical report, 1994.

[85] Aiguo Ming and Toshiro Higuchi. Study on multiple degree-of-freedom positioning mechanism using wires (part 1) – concept, design and control. *Int. Journal of the Jap. Soc. for Precision Engineering*, 28(2):131–138, June 1994.

[86] Aiguo Ming and Toshiro Higuchi. Study on multiple degree-of-freedom positioning mechanism using wires (part 2) – development of a planar completely restrained positioning mechanism. *Int. Journal of the Jap. Soc. for Precision Engineering*, 28(3):235–242, September 1994.

[87] Jörg Müller. *Entwicklung virtueller Prototypen des hydraulisch angetriebenen Schreitfahrwerks ALDURO*. Fortschritt-Berichte VDI, Reihe 1, Nr. 356. VDI Verlag, Düsseldorf, 2002.

[88] Ramon E. Moore. *Interval Analysis*. Prentice-Hall, Englewood Cliffs, New Jersey, 1966.

[89] Ramon E. Moore. *Intervallanalyse*. R. Oldenbourg Verlag München – Wien, 1969.

[90] Ali Nahvi and John M. Hollerbach. The noise amplification index for optimal pose selection in robot calibration. *IEEE Int. Conf. on Robotics and Automation*, pages 647–654, 1996.

[91] Reimund Neugebauer. *Parallelkinematische Maschinen. Entwurf, Konstruktion, Anwendungen*. Springer-Verlag, 2006.

[92] Arnold Neumaier. *Interval methods for systems of equations*. Encyclopedia of mathematics and its application. Cambridge University Press, Cambridge, New York, Port Chester, Melbourne, Sydney, 1990.

[93] Arnold Neumaier. *Introduction to Numerical Analysis*. Cambridge University Press, Cambridge, New York, Port Chester, Melbourne, Sydney, 2001.

[94] Erika Ottaviano and Marco Ceccarelli. Optimal design of CaPaMan (cassino parallel manipulator) with a specific orientation workspace. *Robotica*, 20(2):159–166, 2002.

[95] Vincenzo Parenti-Castelli and Stefano Venanzi. Clearance influence analysis on mechanisms. *Mechanism and Machine Theory*, 40(12):1316–1329, Dezember 2005.

[96] A. Pashkevich, P. Wenger, and D. Chablat. Design strategies for the geometric synthesis of orthoglide-type mechanisms. *Mechanism and Machine Theory*, 40:907–930, 2005.

[97] Friedrich Pernkopf and Manfred L. Husty. Workspace analysis of Stewart-Gough manipulators using orientation plots. In *MUSME 2002, International Symposium on Multibody Systems and Mechatronics*, Mexico City, 2002.

[98] Friedrich Pernkopf and Manfred L. Husty. Reachable workspace and manufacturing errors of Stewart-Gough manipulators. In *MUSME 2005, International Symposium on Multibody Systems and Mechatronics*, Uberlandia, Brazil, 2005.

[99] Huy Hoang Pham and I-Ming Chen. Optimal synthesis for workspace and manipulability of parallel flexure mechanism. In *Proceedings of the 11th World Congress in Mechanism and Machine Science*, Tianjin, China, 2003.

[100] Jon Postel. RFC 791: Internet protocol. http://www.ietf.org/rfc/rfc791.txt, September 1981.

[101] Andreas Pott, Tim Boye, and Manfred Hiller. Parameter synthesis for parallel kinematic machines from given process requirements. In *Proceedings of the International Conference on Advanced Intelligent Mechatronics*, Monterey, California, USA, 2005. IEEE/ASME.

[102] Andreas Pott, Tim Boye, and Manfred Hiller. Design and optimization of parallel kinematic machines under process requirements. In Reimund Neugebauer, editor, *Proceedings of the 5th Chemnitz Parallel Kinematics Seminar*, pages 193–212, Chemnitz, Germany, 2006.

[103] Andreas Pott, Daniel Franitza, and Manfred Hiller. Orientation workspace verification for parallel kinematic machines with constant leg length. In *Proceedings of Mechatronics and Robotics 2004*, pages 984–989, Aachen, Germany, 2004.

[104] Andreas Pott and Manfred Hiller. A force based approach to error analysis of parallel kinematic mechanisms. In *Advances in Robot Kinematics*, pages 293–302, Sestri Levante, Italy, 2004.

[105] Andreas Pott and Manfred Hiller. A framework for analysis, synthesis and optimization of parallel kinematic machines. In *Advances in Robot Kinematics*, pages 103–112, Ljubljana, Slovenia, 2006. Springer-Verlag.

[106] Andreas Pott and Manfred Hiller. A parallel implementation for the optimization of parallel kinematic machines under process requirements. In *1st European Conference on Mechanism Science*, Obergurgl, Austria, 2006.

[107] Andreas Pott, Manfred Hiller, and Andrés Kecskeméthy. A simplified force-based method for the linearization and sensitivity analysis of complex manipulation systems. *Mechanism and Machine Theory*, 42(11):1445–1461, November 2007.

[108] Günter Pritschow, Tim Boye, and Till Franitza. Potentials and limitations of the Linapod's basic kinematic model. In Reimund Neugebauer, editor, *Proceedings of the 4th Chemnitz Parallel Kinematics Seminar*, volume 24 of *Reports from the IWU*, pages 331–345, Chemnitz, 2004. Verlag Wissenschaftliche Scripten.

[109] Günter Pritschow, C. Eppler, and T. Garber. Influence of the dynamic stiffness on the accuracy of PKM. In Reimund Neugebauer, editor, *Proceedings of the 3rd Chemnitz Parallel Kinematics Seminar*, pages 313–333, Chemnitz, 2002. Verlag Wissenschaftliche Scripten.

[110] Günter Pritschow and Karl-Heinz Wurst. Systematic design of hexapods and other parallel link systems. *Annals of the CIRP*, 46:291–295, 1997.

[111] F. Ranjbaran, J. Angeles, and A. Kecskeméthy. On the kinematic conditioning of robotic manipulators. In *IEEE International Conference on Robotics and Automation*, pages 3167–3172, Minneapolis, Minnesota, 1996.

[112] Helmut Ratschek and Jon Rokne. *Geometric Computations with Interval and New Robust Methods*. Horwood Publishing, Chichester, 2003.

[113] Erik Rebeck and Guangming Zhang. A method for evaluating the stiffness of a hexapod machine tool support structure. *International Journal of Flexible Automation and Integrated Manufacturing*, 7:149–165, 1999.

[114] Georg Rex and Jiri Rohn. Sufficient conditions for regularity and singularity of interval matrices. *SIAM J. MATRIX ANAL. APPL.*, 20(2):437–445, 1998.

[115] Felice Ronga and Thierry Vust. Stewart platforms without computer? preprint, Université de Genève, 1992.

[116] Siegfried M. Rump. Fast and parallel interval arithmetic. *BIT Numerical Mathematics*, 39(3):539–560, 1999.

[117] Siegfried M. Rump. INTLAB – Interval laboratory. *Developments in Reliable Computing*, pages 77–104, 1999.

[118] Jeha Ryu and Abdul Rauf. Efficient kinematic calibration of parallel manipulators using a double ball bar system. In *Proceedings of the 32nd ISR (International Symposium on Robotics)*, pages 1713–1718, April 2001.

[119] J. K. Salisbury and J. J. Craig. Articulated hands: Force control and kinematic issues. *The International Journal of Robotics Research*, 1(1):4–17, 1982.

[120] Martin Schneider. *Modellbildung, Simulation und nichtlineare Regelung elastischer, hydraulisch angetriebener Großmanipulatoren.* Fortschritt-Berichte VDI, Reihe 8, Nr. 756. VDI Verlag, Düsseldorf, 1999.

[121] J. A. Snyman and A. M. Hay. Analysis and optimization of a planar tendon-driven parallel manipulator. In *Advances in Robot Kinematics*, pages 303–312, Sestri Levante, Italy, 2004. Kluwer Academic Publishers.

[122] J. Song, J.-I. Mou, and C. King. Error modeling and compensation for parallel kinematic machines. In *Parallel Kinematic Machines*, pages 172–187, London, 1999. Springer-Verlag.

[123] E. Staffetti, H. Bruyninckx, and J. De Schutter. On the invariance of manipulability indices. In Jadran Lenarčič and Federico Thomas, editors, *Advances in Robot Kinematics*, pages 57–66, Caldes de Malavella, Spain, 2002. Kluwer Academic Publishers.

[124] D. Stewart. A platform with six degrees of freedom. *Proc, Inst. Mech. Eng. London, Part 1*, 180-5:371–386, 1965.

[125] Bjarne Stroustrup. *Die C++ Programmiersprache.* Addison-Wesley Publishing Company, Berlin, 2000.

[126] R. Suikat. The new dynamic driving simulator at DLR. In *Proceedings of Driving Simulator Conference*, Orlando, Florida, 2005.

[127] Sun Microsystems, Inc. *Forte Developer 7, C++ Interval Arithmetic Programming Reference.* Sun Microsystems, Inc., 2002.

[128] K. Takamasu, I. Murui, O. Sato, G. Olea, and R. Furutani. Calibration of three dimensional mechanism – Novel calibration D115 method for 3DOF parallel mechanism. *IEEE Int. Conf. on Industrial Technology ICIT*, pages 394–398, December 2002.

[129] H. K. Tönshoff, H. Grendel, J. Jacobsen, and S. Scherger. Auslegung und Regelung eines lineardirekt angetriebenen Hexapoden. In *4. VDI-Mechatroniktagung 2001, Frankenthal, VDI Berichte 1631*, pages 425–446, 2001.

[130] H. K. Tönshoff, H. Grendel, and R. Kaak. Structure and characteristics of the hybrid manipulator Georg V. In *Proceedings of the 1st European-American Forum on Parallel Kinematic Machines*, Milan, Italy, 1998.

[131] Richard Verhoeven. *Analysis of the Workspace of Tendon-based Stewart Platforms*. PhD thesis, University of Duisburg-Essen, 2004.

[132] Richard Verhoeven and Manfred Hiller. Estimating the controllable workspace of tendon-based Stewart platforms. In *Advances in Robot Kinematics*, pages 277–284, Portorož, Slovenia, 2000.

[133] Richard Verhoeven and Manfred Hiller. Tension distribution in tendon-based Stewart platforms. In *Advances in Robot Kinematics*, pages 117–124, Caldes de Malavella, Spain, 2002.

[134] P. Vischer. *Improve the accuracy of parallel robots*. PhD thesis, Institut des Systèmes Robotiques, École Polytechnique Fédérale de Lausanne, 1996.

[135] Andreas Wiethoff. *Verifizierte globale Optimierung auf Parallelrechnern*. PhD thesis, Universität Karlsruhe, 1997.

[136] Jonathan W. Wittwer, Kenneth W. Chase, and Larry L. Howell. The direct linearization method applied to position error in kinematic linkages. *Mechanism and Machine Theory*, 39(7):681–693, July 2004.

[137] Christoph Woernle. *Ein systematisches Verfahren zur Aufstellung der geometrischen Schließbedingungen in kinematischen Schleifen mit Anwendung bei der Rückwärtstransformation für Industrieroboter*. Fortschritt-Berichte VDI, Reihe 18, Nr. 59. VDI Verlag, Düsseldorf, 1988.

[138] Weidong Wu and S. S. Rao. Interval approach for the modeling of tolerances and clearances in mechanism analysis. *Transactions of the ASME, Journal of Mechanical Design*, 126:581–592, 2004.

[139] Karl-Heinz Wurst. Linapod – Machine tools as parallel link system in a modular design. In *Proceedings of the 1st European-American Forum on Parallel Kinematic Machines*, Milan, Italy, 1998.

[140] Karl-Heinz Wurst and L. Dubois. Das Maschinenkonzept LINAPOD. In *Umdruck zum Seminar 'Hexapod, Linapod, Dyna-M'*, Aachen, 1998.

[141] T. Yoshikawa. Manipulability of robotic mechanisms. *International Journal of Robotics Research*, 4(2):3–9, 1985.

[142] Yu Zhang, Joseph Duffy, and Carl D. Crane. The optimum quality index for a spatial redundant 4-8 in-parallel manipulator. In *Advances in Robot Kinematics*, pages 239–249, Portorož, Slovenia, 2000.

[143] J.-W. Zhao, K.-C. Fan, T.-H. Chang, and Z. Li. Error analysis of a serial-parallel type machine tool. *International Journal of Advanced Manufactoring Technology*, 19:174–179, 2002.

Lebenslauf

Persönliche Daten

Name	Andreas Alexander Pott
Geburtsdatum	4. August 1977
Geburtsort	Solingen
Familienstand	verheiratet mit Ariane Pott (geb. Bauer)

Schulausbildung

08/1984 – 07/1988	Grundschule Meigen in Solingen
08/1988 – 06/1997	August-Dicke-Gymnasium Solingen, Abschluss: Abitur

Zivildienst

07/1997 – 08/1998	Club der Behinderten und ihrer Freunde Solingen e.V.

Hochschulausbildung

10/1998 – 03/2003	Studium des allgemeinen Maschinenbaus an der Universität Duisburg-Essen, Abschluss: Diplom-Ingenieur

Beruf

04/2003 – 09/2006	Wissenschaftlicher Mitarbeiter am Lehrstuhl für Mechatronik, Universität Duisburg-Essen