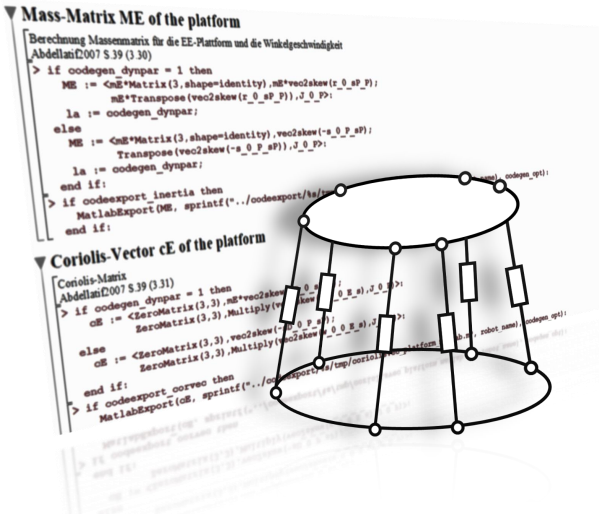
Institut für
Mechatronische Systeme
imes Leibniz Universität Hannover

Leibniz
Universität
Hannover

# Implementierung einer strukturunabhängigen Dynamikmodellierung für parallelkinematische Maschinen




Studienarbeit S-12/18-759

Tim-David Job
Matrikelnummer 3020710

Hannover, 01. Dezember 2018

Fachprüfer Prof. Dr.-Ing. Tobias Ortmaier
Betreuer Moritz Schappler, M.Sc.

Ich, Tim-David Job, versichere hiermit, dass die vorliegende Arbeit selbstständig verfasst wurde, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt wurden, alle Stellen der Arbeit, die wörtlich oder sinngemäß aus anderen Quellen übernommen wurden, als solche kenntlich gemacht sind und die Arbeit in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen hat.

Hannover, 01. Dezember 2018

(Tim-David Job)

Institut für
Mechatronische Systeme
Leibniz Universität Hannover
imes

# Studienarbeit

Herr Tim-David Job, Matr.-Nr. 3020710

## Implementierung einer strukturunabhängigen Dynamikmodellierung für parallelkinematische Maschinen

Implementation of a Structurally Independant Dynamics Modelling of Parallel Kinematic Machines

**Allgemeines:**
Im Rahmen des Projektes „Generierung aufgabenspezifischer Roboterkinematiken durch kombinierte Struktur- und Maßsynthese“ werden speziell für vorgegebene Aufgaben optimierte serielle sowie parallele Roboterstrukturen durch einen Algorithmus automatisch generiert, ausgewählt und dimensioniert. Dazu ist auch eine Berechnung der inversen Dynamik für verschiedene Systeme notwendig um bspw. nötige Antriebskräfte und daraus abgeleitete Leistungskennzahlen zu berechnen.

**Aufgabe:**
Im Rahmen dieser Studienarbeit soll eine strukturunabhängige Dynamikmodellierung in Form einer Toolbox in den Programmen MATLAB und MAPLE implementiert werden. Ein üblicher generischer Ansatz zur Dynamikmodellierung ist die Aufteilung des parallelen Systems in einzelne Ketten, die Berechnung der Dynamik für die Einzelketten und Rückrechnung auf das geschlossene System. Dazu soll ein publizierter Ansatz[1] basierend auf der Projektion auf die Endeffektorkoordinaten implementiert werden und mit weiteren bereits vorhandenen Implementierungen verglichen werden, die auf der Invertierung der Jacobi-Matrix der impliziten Form der kinematischen Zwangsbedingungen basieren. Dabei soll auch die Aufstellung der Gleichungen in parameterlinearer Form untersucht werden.

Im Rahmen dieser Arbeit ergeben sich insbesondere die folgenden Aufgabenpunkte:

- Literaturrecherche und Einarbeitung,
- Inbetriebnahme der existierenden Programme „ParaDyn“ und „openSYMORO“ und Test mit Beispielsystemen,
- Definition von Schnittstellen und Programmstruktur,
- Implementierung des zu untersuchenden Ansatzes (nach Abdellatif et al.),
- Herleitung der Dynamikgleichungen für unterschiedliche Systeme mit bekannter Lösung mittels der verschiedenen Ansätze bzw. Implementierungen,
- Vergleich der Ergebnisse (u. a. hinsichtlich der Struktur der Lösung, Recheneffizienz und der Beschränkung auf bestimmte Robotertypen)

Die Bearbeitungszeit beträgt 300h.
Ausgabe der Aufgabenstellung: 01.06.2018 Abgabe der Arbeit: Gemäß Prüfungsamt

Betreuer: M. Sc. Moritz Schappler

..... .....
(Prof. T. Ortmaier) (Tim-David-Job)

[1] H. Abdellatif, B. Heimann: Computational efficient inverse dynamics of 6-DOF fully parallel manipulators by using the Lagrangian formalism, Mechanism and Machine Theory, 2009

# Kurzfassung

Das Ziel dieser Arbeit ist die Implementierung einer automatischen Dynamikmodellierung für parallelkinematische Maschinen. Diese dient unter anderem der automatischen Struktur- und Maßsynthese neuartiger Roboter in Abhängigkeit einer vorgegeben Aufgabenspezifikation. Als Grundlage wird eine bereits vorhandene MAPLE-Toolbox verwendet, mit der aber bisher nur serielle und hybride Strukturen berechnet werden können. Da der zu implementierende Ansatz nach Abdellatif et al. auf der getrennten Betrachtung der Beinketten und der Plattform basiert, wurde die Toolbox so erweitert, dass auf die vorhandenen Lösungen für serielle Ketten zurückgegriffen werden kann. Um das Dynamikmodell, das über den LAGRANGE-sche Formalismus aufgestellt wird, in Minimalform zu erhalten, werden die Bewegungsgleichungen der Teilsysteme in den Plattform-Koordinatenraum projiziert. Als Referenz dienen in dieser Arbeit zwei weitere Implementierungen zur Berechnung der Dynamik, die auf einer ähnlichen Projektion auf Basis der kinematischen Zwangsbedingungen bzw. auf der Verwendung der NEWTON-EULER-Methode basieren. Die Lösungen wurden hinsichtlich Recheneffizienz, Struktur der Lösung und verfahrensbedingte Begrenzungen für verschiedene Roboterstrukturen untersucht und mit denen der Referenzen verglichen.

# Inhaltsverzeichnis

# Bildverzeichnis



# Tabellenverzeichnis

# Nomenklatur

Selten bzw. nur abschnittsweise verwendete Symbole und Formelzeichen sowie abweichende Bedeutungen werden ausschließlich im Text beschrieben.

### Allgemeine Konventionen

| | |
|---|---|
| Skalar | Klein- oder Großbuchstabe (kursiv): $a$, $A$ |
| Vektor | Kleinbuchstabe (fett und kursiv): $\boldsymbol{a}$ |
| Matrix | Großbuchstabe (fett und kursiv): $\boldsymbol{A}$ |
| Punkt | Klein- oder Großbuchstabe: $\mathrm{a}$, $\mathrm{A}$ |
| Körper | Großbuchstabe (fett): $\mathbf{A}$ |

### Lateinische Buchstaben

| | |
|---|---|
| $\boldsymbol{r}_A^B$ | Richtungsvektor von Punkt $A$ nach $B$ beschrieben in $(\mathrm{KS})_0$ |
| $\boldsymbol{r}_A$ | Ortsvektor zum Punkt $B$ in $(\mathrm{KS})_0$ beschrieben in $(\mathrm{KS})_0$ |
| $\boldsymbol{J}_y^z$ | Jacobi-Matrix aus $\dot{\boldsymbol{z}} = \boldsymbol{J}_y^z \dot{\boldsymbol{y}}$ |
| $\boldsymbol{T}_{\mathrm{r}_{\mathrm{Achse}}}$ | rotatorische, homogene Transformationsmatrizen um die angegebene Achse |
| $\boldsymbol{T}_{\mathrm{t}_{\mathrm{Achse}}}$ | tranlatorische, homogene Transformationsmatrizen entlang der angegebene Achse |
| $\boldsymbol{q}_{\mathrm{ges}}$ | Alle Systemkoordinaten des Roboters |
| $\boldsymbol{q}$ | Alle Gelenkkoordinaten des Roboters |
| $\boldsymbol{q}_{\mathrm{a}}$ | Aktive Gelenkkoordinaten des Roboters |
| $\boldsymbol{q}_{\mathrm{p}}$ | Passive Gelenkkoordinaten des Roboters |
| $\boldsymbol{x}_{\mathrm{E}}$ | Endeffektorkoordinaten des Roboters |
| $\boldsymbol{x}_{\mathrm{P}}$ | Plattformkoordinaten des Roboters |
| $\boldsymbol{x}_{\mathrm{K}_j}$ | Koppelpunktkoordinaten des Roboters |
| $\boldsymbol{z}$ | Minimalkoordinaten des Roboters |
| $\dot{\boldsymbol{s}}$ | Quasigeschwindigkeiten des Roboters |
| $\boldsymbol{\tau}_{q_{\mathrm{ges}}}$ | Kräfte und Momente aller Systemkoordinaten |
| $\boldsymbol{\tau}_q$ | Kräfte und Momente aller Gelenke |
| $\boldsymbol{\tau}_{\mathrm{a}}$ | Kräfte und Momente der aktiven Gelenke |
| $\boldsymbol{\tau}_{\mathrm{p}}$ | Kräfte und Momente der passiven Gelenke |
| $\boldsymbol{\tau}_{\mathrm{E}}$ | Kräfte und Momente am Endeffektor |
| $\boldsymbol{\tau}_{\mathrm{P}}$ | Kräfte und Momente an der Plattform |

| | |
|---|---|
| $\boldsymbol{\tau}_{\mathrm{K}_j}$ | Kräfte und Momente am Koppelpunkt |
| $\boldsymbol{p}$ | Parametervektor |
| $\boldsymbol{M}$ | Inertialmatrix der Dynamikgleichung |
| $\boldsymbol{c}$ | Coriolisvektor der Dynamikgleichung |
| $\boldsymbol{g}$ | Gravitationsvektor der Dynamikgleichung |
| $m_i$ | Masse des Teilkörpers $i$ |
| $\boldsymbol{f}_{\mathrm{t}}$ | Dynamische Kraft |
| $\boldsymbol{f}$ | Reaktionskraft |
| $\boldsymbol{m}_{\mathrm{t}}^{(A)}$ | Dynamisches Moment um Punkt $A$ |
| $\boldsymbol{m}^{(A)}$ | Reaktionsmoment um Punkt $A$ |
| $\boldsymbol{a}$ | Beschleunigung |
| $\boldsymbol{I}^{(A)}$ | Trägheitstensor um Punkt $A$ beschrieben in $(\mathrm{KS})_0$ |
| $\boldsymbol{v}_i$ | Absoluter Geschwindigkeitsvektor für den Körper $i$ beschrieben in $(\mathrm{KS})_0$ |
| $\boldsymbol{g}_{\mathrm{grav}}$ | Gravitationsvektor |
| $\boldsymbol{h}$ | kinematische Zwangsbedingungen |

### Griechische Buchstaben

| | |
|---|---|
| $\psi,\ \theta,\ \phi$ | Rotationswinkel um die $x$-, $y$- und $z$-Achse |
| $\boldsymbol{\lambda}$ | LAGRANGE-Multiplikatoren |
| $\boldsymbol{\omega}_i$ | Vektor Winkelgeschwindigkeiten für den Körper $i$ beschrieben in $(\mathrm{KS})_0$ |

### Koordinatensysteme

| | |
|---|---|
| $(\mathrm{KS})_i$ | Koordinatensystem $i$ |
| $(\mathrm{KS})_0$ | ortsfestes Inertialkoordinatensystem |
| $(\mathrm{KS})_{\mathrm{E}}$ | Endeffektorkoordinatensystem |
| $(\mathrm{KS})_{\mathrm{P}}$ | Plattformkoordinatensystem |

### Abkürzungen

| | |
|---|---|
| FHG | Freiheitsgrad |
| EE | Endeffektor |
| PKM | Parallelkinematische Maschine |
| DH | DENAVIT-HARTENBERG |
| MDH | modifizierte DENAVIT-HARTENBERG |

# 1 Einleitung

Die hohe Nachfrage in der Robotik- und Automatisierungstechnologie bewirkt, dass immer wieder neue Ansprüche an die aktuelle Robotertechnik gestellt werden. Aufgrund des enormen Wachstums entsteht der Bedarf, spezifische Roboter zu entwickeln, die auf eine einzige Aufgabe zugeschnitten sind und dabei bestimmte Spezifikationen einhalten. Der Hauptabnehmer ist nach wie vor die Industrie mit dem Haupteinsatzzweck in der Automobilbranche, aber auch die Medizin- oder Servicerobotik sind wachsende Einsatzfelder. In dem übergeordneten Projekt dieser Arbeit soll eine Methode zur automatischen Struktur- und Maßsynthese von parallelkinematischen Maschinen in Abhängigkeit einer vorgegeben Aufgabenspezifikation entwickelt werden. Dabei soll der Roboter gewisse Vorgaben wie z. B. einen geringen Energieverbrauch, eine hohe Dynamik oder ein geringes Gewicht einhalten. Bisherige Roboter werden in den meisten Fällen auf breite Anwendungsfelder ausgelegt und sind für den letztendlichen Einsatzzweck somit häufig überdimensioniert, wodurch sie die gewünschten Vorgaben nicht einhalten können. Da eine solche Methode bereits für serielle Roboter vorliegt, liegt der Schwerpunkt hier auf der Betrachtung parallelkinematischer Maschinen.

Das Grundkonzept der Methode, auf das die Entwicklung aufbauen soll, besteht aus zwei Schritten: Zunächst sollen alle Roboterstrukturen, die für die Aufgabe in Frage kommen, bestimmt werden (Struktursynthese), um in einem zweiten Schritt die Dimensionierung aller ausgewählten Roboterkinematiken so festzulegen, dass die gewünschten Spezifikationen erfüllt werden (Maßsynthese). Am Ende liegt eine Löungsmenge aller Roboter, die für die gewünschte Aufgabe und den einzuhaltenden Spezifikationen in Frage kommen, vor. Hierzu wird ein Optimierungsverfahren entwickelt, in dem eine aus den Vorgaben abgeleitete Gütefunktion in Abhängigkeit der kinematische und dynamische Leistungskenngrößen minimiert wird. Voraussetzung für die Optimierung ist, dass die Modellierung der Kinematik und der Dynamik aus den ausgewählten Roboterstrukturen automatisch berechnet werden kann.

Für diesen Zweck, aber auch für spätere Steuerungs- und Regelungsaufgaben wie z. B. Vorsteuerungen oder Simulationen, soll in dieser Arbeit eine automatische Dynamikmodellierung implementiert werden. Für serielle Strukturen liegt diese basierend auf den LAGRANGE'schen Gleichungen bereits vor und kann als Grundlage weiterverwendet werden [Sch]. Für parallele Roboterstrukturen existieren verschiedene Implementierungen zur Dynamikberechnung, die auf unterschiedlichen Ansätzen beruhen. In dieser Arbeit soll ein weiterer Ansatz für parallele

Roboter implementiert werden [AH09], während zwei weitere Verfahren zu Vergleichszwecken dienen [DTKHO09a], [KVK+14]. Alle drei Ansätze basieren entweder auf der Berechnung der Dynamik mit Hilfe des NEWTON-EULER-Verfahrens oder mittels der LAGRANGE'schen Gleichungen beziehungsweise der HAMEL-BOLTZMANN-Gleichungen für Quasikoordinaten und -geschwindigkeiten. Das Hauptaugenmerk liegt hierbei auf der Berechnung der inversen Dynamik, da die vorgegebene Aufgabenspezifikation in den meisten Fällen eine Endeffektorbewegung ist und die zu minimierende Gütefunktion die Kräfte und Momente der Antriebe beinhaltet. Durch Umstellung nach den Beschleunigungen mittels Matrixinvertierung und anschließende Integration lässt sich die direkte Dynamik aus der inversen Dynamik bestimmen. Am Ende liegt eine Implementierung vor, die für jeden Roboter die vollständige Dynamikmodellierung übernimmt, ohne das der Anwender sich mit dem Fachgebiet der Mehrkörpersysteme auseinandersetzen muss.

Um den oben genannten Einsatzzwecken gerecht zu werden, soll das Ergebnis der Modellierung einen möglichst geringen Rechenaufwand verursachen. In diesem Punkt unterscheiden sich die verschiedenen Ansätze (NEWTON-EULER und LAGRANGE) voneinander, was durch einen Vergleich am Ende dieser Arbeit mit untersucht werden soll. In jedem Fall muss das Dynamikmodell in Abhängigkeit der Minimalkoordinaten bestimmt werden. Sie entsprechen der minimalen Anzahl an Koordinaten, mit denen jeder Zustand des Roboters beschrieben werden kann und ihre Verwendung führt zu einer Reduzierung des Modells auf die geringste Anzahl Bewegungsgleichungen. Im Allgemeinen entsprechen die Minimalkoordinaten bei parallelen Robotern den EE- bzw. Plattformkoordinaten [Abd07]. Darüber hinaus erfolgt die Implementierung des Verfahrens über das Computer-Algebra-System MAPLE. Der Vorteil gegenüber der numerischen Berechnung, wie sie beispielsweise in MATLAB erfolgen würde, liegt ebenfalls in der Optimierung der Recheneffizienz [DPF13].

In dieser Arbeit wird zunächst in die wichtigsten Grundlagen der Roboterkinematik und -dynamik in Abschnitt 2 eingeführt, um darauf aufbauend in Abschnitt 3 auf die unterschiedlichen Ansätze der Dynamikmodellierung einzugehen. Während die Funktionsweisen der Vergleichsimplementierungen nur kurz angeschnitten werden, wird das zu implementierende Verfahren umfangreicher erläutert. Im Anschluss wird in Abschnitt 4 der Aufbau der Toolbox, die um die Dynamikmodellierung für parallele Roboter erweitert wurde, vorgestellt. In Abschnitt 5 wird auf die Roboterkinematiken, die zu Testzwecken der jeweiligen Implementierungen herangezogen werden, und deren Besonderheiten bzw. gegebenenfalls eingeführten Vereinfachungen eingegangen. Am Ende des Kapitels folgt ein Vergleich der neuen Implementierung gegenüber der bestehenden unter Betrachtung der genannten Beispielroboter. Eine Zusammenfassung mit kurzem Ausblick in Abschnitt 6 schließen diese Arbeit ab.

# 2 Grundlagen

Im Folgenden wird zunächst auf die wichtigsten Grundlagen für die Dynamikberechnungen bei parallelen Robotern eingegangen. In Abschnitt 2.1 wird zunächst auf Roboterkinematik eingegangen, auf der die nachfolgenden Berechnungen basieren. Anschließend beschreibt Abschnitt 2.2 die verschiedenen Verfahren zur Dynamikmodellierung, sowie Besonderheiten, die bei parallelen Robotern auftreten.

## 2.1 Kinematik

Die Kinematik bildet die Grundlage für die Robotermodellierung. Sie beschreibt den Zusammenhang zwischen den Gelenkkoordinaten $\boldsymbol{q}$ und den Endeffektorkoordinaten $\boldsymbol{x}_{\mathrm{E}}$. Die Kinematik kann entweder in impliziter oder in expliziter Form beschrieben werden. Wenn explizit von den gegeben aktiven Winkeln $\boldsymbol{q}_{\mathrm{a}}$ auf $\boldsymbol{x}_{\mathrm{E}}$ geschlossen werden soll, wird dies als direkte Kinematik bezeichnet. Den umgekehrten Fall nennt man inverse Kinematik.

### 2.1.1 Impliziter Zusammenhang

Um die Gelenkkoordinaten $\boldsymbol{q}$ und die Endeffektorkoordinaten $\boldsymbol{x}_{\mathrm{E}}$ in eine funktionale Relation setzen zu können, müssen kinematischen Zwangsbedingungen

$$\boldsymbol{h}(\boldsymbol{q}, \boldsymbol{x}_{\mathrm{E}}) = \boldsymbol{0} \tag{2.1}$$

aufgestellt werden. Grundsätzlich erhält man diese Zwangsbedingungen durch Betrachtung eines Punktes auf dem Roboter, der über zwei Wege entlang der Robotergeometrie berechnet wird. Durch Gleichsetzen dieser beiden Vektoren, erhält man den impliziten Zusammenhang aus (2.1). Ein bewährtes Vefahren ist hierbei das Gleichsetzen des Vektors der Koppelpunktkoordinaten $\boldsymbol{x}_{\mathrm{K}}$, der über die folgenden Wege berechnet wurde: Der erste Vektor $\boldsymbol{h}_{1i}(\boldsymbol{q})$ wird über die serielle Kette eines Beines $i$ und der zweite $\boldsymbol{h}_{2i}(\boldsymbol{x}_{\mathrm{E}})$ über die Endeffektorbewegung und dem Abstand der Koppelpunkte $\boldsymbol{x}_{\mathrm{K}i}$ zum Endeffektor berechnet. Durch Stapeln aller so enstandenen $i$ Gleichungen erhält man (2.1) [Mer06].

### 2.1.2 Inverse Kinematik

Die inverse Kinematik beschreibt die Winkel der aktiven Koordinaten $\boldsymbol{q}_\mathrm{a}$ in Abhängigkeit der Endeffektorkoordinaten $\boldsymbol{x}_\mathrm{E}$:

$$\boldsymbol{q}_\mathrm{a} = \boldsymbol{g}(\boldsymbol{x}_\mathrm{E}) \tag{2.2}$$

Bei seriellen Robotern kann die analytische Herleitung sehr aufwendig sein, da kein allgemeingültiger Lösungsformalismus existiert. Für parallele Roboter wird die inverse Kinematik im Allgemeinen über die minimalen Zwangsbedingungen $\boldsymbol{h}(\boldsymbol{q}_\mathrm{a}, \boldsymbol{x}_\mathrm{E}) = \boldsymbol{0}$ bestimmt. Der Vektor hat die Größe $(i \times 1)$, wenn, wie gewöhnlich, jedes Roboterbein nur ein angetriebenes Gelenk besitzt. Man erhält sie für die gängigen Robotertypen, deren Plattform-Koppelgelenke so viele rotatorische Freiheitsgrade wie der Endeffektor haben, aus (2.1), indem die passiven Gelenkwinkel eliminiert werden [Mer06]. Um die Gleichungen in der expliziten Form (2.2) zu erhalten, werden sie nach $\boldsymbol{q}_\mathrm{a}$ umgestellt.

### 2.1.3 Direkte Kinematik

Während die inverse Kinematik über den oben beschriebenen Ansatz gelöst wird, existiert für die direkte Kinematik

$$\boldsymbol{x}_\mathrm{E} = \boldsymbol{f}(\boldsymbol{q}_\mathrm{a}) \tag{2.3}$$

bei parallelen Robobter kein analytisches Lösungsverfahren. Dies liegt daran, dass für eine Gelenkstellungung der aktiven Koordinaten $\boldsymbol{q}_\mathrm{a}$ mehrere Endeffektorstellungen $\boldsymbol{x}_\mathrm{E}$ existieren [KD02]. Für eine eindeutige Zuordnung müssten auch die passiven Gelenkkoordinaten $\boldsymbol{q}_\mathrm{p}$ bekannt sein. Werden diese nicht zusätzlich gemessen, kann die Lösung nur über eine iterative numerische Lösung erfolgen.

### 2.1.4 Differentielle Kinematik

Die Beschreibung des Zusammenhang der Ableitungen der Systemkoordinaten und Endeffektorkoordinaten wird differentielle Kinematik genannt. Wie bei der direkten bzw. inversen Kinematik sind im Allgemeinen nur die aktiven Koordinaten von Interesse, sodass die Lösung

$$\dot{\boldsymbol{x}}_\mathrm{E} = \frac{\mathrm{d}\boldsymbol{f}(\boldsymbol{q}_\mathrm{a})}{\mathrm{d}t} = \frac{\partial \boldsymbol{f}(\boldsymbol{q}_\mathrm{a})}{\partial \boldsymbol{q}_\mathrm{a}} \frac{\mathrm{d}\boldsymbol{q}_\mathrm{a}}{\mathrm{d}t} = \frac{\partial \boldsymbol{f}(\boldsymbol{q}_\mathrm{a})}{\partial \boldsymbol{q}_\mathrm{a}} \dot{\boldsymbol{q}}_\mathrm{a} = \boldsymbol{J} \dot{\boldsymbol{q}}_\mathrm{a} \tag{2.4}$$

gesucht wird. Für die differentielle Kinematik reicht es demnach aus, die Gesamt-Jacobi-Matrix $\boldsymbol{J}$ zu kennen. Genau wie bei den Kinematikbetrachtungen, ist auch hier bei PKM nur die Berechnung der inversen differentiellen Kinematik $\dot{\boldsymbol{q}}_\mathrm{a} = \boldsymbol{J}^{-1} \dot{\boldsymbol{x}}_\mathrm{E}$ analytisch effizient

möglich. Bei komplizierten Robotermechanismen mit vielen Freiheitsgraden, erhält man $\boldsymbol{J}$ durch numerisches Invertieren von $\boldsymbol{J}^{-1}$. Bei parallelen Robotern wird für die Berechnung von $\boldsymbol{J}^{-1}$ auf die kinematischen Zwangsbedingungen $\boldsymbol{h}(\boldsymbol{q}_\mathrm{a}, \boldsymbol{x}_\mathrm{E}) = \boldsymbol{0}$ zurückgegriffen. Durch zeitliches Ableiten der Zwangsbedingungen erhält man

$$\frac{\mathrm{d}\boldsymbol{h}(\boldsymbol{x}_\mathrm{E}, \boldsymbol{q}_\mathrm{a})}{\mathrm{d}t} = \frac{\partial \boldsymbol{h}(\boldsymbol{x}_\mathrm{E}, \boldsymbol{q}_\mathrm{a})}{\partial \boldsymbol{x}_\mathrm{E}} \dot{\boldsymbol{x}}_\mathrm{E} + \frac{\partial \boldsymbol{h}(\boldsymbol{x}_\mathrm{E}, \boldsymbol{q}_\mathrm{a})}{\partial \boldsymbol{q}_\mathrm{a}} \dot{\boldsymbol{q}}_\mathrm{a} = \boldsymbol{A}\dot{\boldsymbol{x}}_\mathrm{E} + \boldsymbol{B}\dot{\boldsymbol{q}}_\mathrm{a} = 0. \tag{2.5}$$

Umgestellt nach den angetriebenen Gelenkgeschwindigkeiten ergibt sich

$$\dot{\boldsymbol{q}}_\mathrm{a} = -\boldsymbol{B}^{-1}\boldsymbol{A}\dot{\boldsymbol{x}}_\mathrm{E} = \boldsymbol{J}^{-1}\dot{\boldsymbol{x}}_\mathrm{E}. \tag{2.6}$$

Diese Lösung hat zwei Vorteile: Zum einen müssen die Zwangsbedingungen nicht erst aufwendig umgestellt werden, zum anderen liegen nun auch die Jacobimatrix der direkten Kinematik $\boldsymbol{A}$ und die Jacobimatrix der inverse Kinematik $\boldsymbol{B}$ vor. $\boldsymbol{A}$ und $\boldsymbol{B}$ sind vor allem für Singularitätsuntersuchungen von Relevanz [Abd07]. Durch erneutes Ableiten von (2.6) können die Beschleunigungen der aktiven Gelenke berechnet werden:

$$\ddot{\boldsymbol{q}}_\mathrm{a} = \boldsymbol{J}^{-1}\ddot{\boldsymbol{x}}_\mathrm{E} + \dot{\boldsymbol{J}}^{-1}\dot{\boldsymbol{x}}_\mathrm{E} \tag{2.7}$$

Da mit der Gesamt-Jacobimatrix $\boldsymbol{J}$ die Geschwindigkeiten verknüpft werden, gilt der gleiche Zusammenhang auch für kleine Positions- oder Winkeländerungen

$$\Delta\boldsymbol{q}_\mathrm{a} = \boldsymbol{J}^{-1}\Delta\boldsymbol{x}_\mathrm{E}. \tag{2.8}$$

Mit (2.8) kann nun auch die Beziehung zwischen Kräften und Momenten hergeleitet werden. Wenn davon ausgegangen wird, dass die Energie, die zur Endeffektorbewegung und zu aktiven Gelenkbewegungen vom System aufgebracht werden muss, die gleiche ist und in den passiven Gelenken keine Energie dissipiert wird, erhält man

$$\begin{aligned} \Delta\boldsymbol{q}_\mathrm{a}^\mathrm{T}\boldsymbol{\tau}_\mathrm{a} &= \Delta\boldsymbol{x}_\mathrm{E}^\mathrm{T}\boldsymbol{\tau}_\mathrm{E} \\ \Delta\boldsymbol{x}_\mathrm{E}^\mathrm{T}\boldsymbol{J}^{-\mathrm{T}}\boldsymbol{\tau}_\mathrm{a} &= \Delta\boldsymbol{x}_\mathrm{E}^\mathrm{T}\boldsymbol{\tau}_\mathrm{E} \\ \boldsymbol{J}^{-\mathrm{T}}\boldsymbol{\tau}_\mathrm{a} &= \boldsymbol{\tau}_\mathrm{E}. \end{aligned} \tag{2.9}$$

Die erste Zeile von (2.9) ist auch unter dem Prinzip der virtuellen Arbeit bekannt. $\boldsymbol{\tau}_\mathrm{a}$ sind die Kräfte bzw. Momente in den aktiven Gelenken und $\boldsymbol{\tau}_\mathrm{E}$ entspricht den resultierenden Kräften bzw. Momenten am Endeffektor [OK14].

### 2.1.5 MDH-Parameter

Für die Beschreibung des seriellen Robotermechanismus hat sich die Verwendung der DENAVIT-HARTENBERG-Notation etabliert. Sie dient vor allem dazu die Wahl der Koordinatensysteme für jeden Teilkörper des Roboters zu normen, sodass die Transformation von einem Teilkörper zum nächsten mit nur vier DENAVIT-HARTENBERG-Parametern beschrieben werden kann. Mit einer Tabelle, die diesen Parameter für jeden Teilkörper bzw. jedes Koordinatensystem besteht, lässt sich die Kinematik des Roboter eindeutig bestimmen. Die homogene Transformation vom Koordinatensystem $(\mathrm{KS})_{\mathrm{i}}$ nach $(\mathrm{KS})_{\mathrm{j}}$ lautet somit

$$^{\mathrm{i}}\boldsymbol{T}_{\mathrm{j}}(q_j) = \boldsymbol{T}_{\mathrm{r_z}}(\theta_j)\boldsymbol{T}_{\mathrm{t_z}}(d_j)\boldsymbol{T}_{\mathrm{t_x}}(a_j)\boldsymbol{T}_{\mathrm{r_x}}(\alpha_j), \tag{2.10}$$

wobei $\boldsymbol{T}_{\mathrm{r_x}}$, $\boldsymbol{T}_{\mathrm{r_y}}$ und $\boldsymbol{T}_{\mathrm{r_z}}$ für die rotatorischen homogenen Transformationen um die $x$-, $y$- und $z$-Achse stehen und $\boldsymbol{T}_{\mathrm{t_x}}$, $\boldsymbol{T}_{\mathrm{t_y}}$ und $\boldsymbol{T}_{\mathrm{t_z}}$ die translatorische homogene Transformation entlang der entprechenen Achsen beschreiben. Die vier Parameter, die vorgegeben werden müssen, lauten $\theta_j$, $d_j$, $a_j$ und $\alpha_j$ [OK14].

Für geschlossenen Ketten, wie die hybriden oder parallelen Roboter, reicht die Notation mit vier Parametern nicht aus, weshalb die modifizierte DENAVIT-HARTENBERG-Notation entstanden ist. Genauso wie bei der klassischen DENAVIT-HARTENBERG-Notation, ist die Wahl der Koordinatensysteme bis auf wenige Sonderfälle exakt vorgegeben. Durch den komplizierteren Mechanismus von geschlossenen Ketten, müssen aber mehr Parameter vorgegeben werden. Neben den sechs Parametern, die die Transformation von einem Körper zum nächsten beschreiben, muss angegeben zwischen welchen Gelenken die Transformation erfolgt. Darüber hinaus muss vorgegeben werden, ob es sich bei dem jeweiligen Gelenk des Teilkörpers um eine Dreh- ($\sigma_j = 0$) oder Schubgelenk ($\sigma_j = 1$) und ob es sich um ein aktives ($\mu_j = 1$) oder passives ($\mu_j = 0$) handelt. Da ein Teilglied mehrere Folgeglieder, wird mit $a(j) = i$ angegeben, dass es sich um die Transformation von $(\mathrm{KS})_{\mathrm{i}}$ nach $(\mathrm{KS})_{\mathrm{j}}$

$$^{\mathrm{i}}\boldsymbol{T}_{\mathrm{j}}(q_j) = \boldsymbol{T}_{\mathrm{r_z}}(\gamma_j)\boldsymbol{T}_{\mathrm{t_z}}(b_j)\boldsymbol{T}_{\mathrm{r_x}}(\alpha_j)\boldsymbol{T}_{\mathrm{t_x}}(d_j)\boldsymbol{T}_{\mathrm{r_z}}(\theta_j)\boldsymbol{T}_{\mathrm{t_z}}(r_j) \tag{2.11}$$

handelt. Somit müssen für die vollständige Beschreibung die acht Parameter $\sigma_j$, $a(j)$, $\mu_j$, $\gamma_j$, $b_j$, $\alpha_j$, $d_j$, $\theta_j$ und $r_j$ vorgegeben werden [KD02]. Da es ausreicht, bei den in dieser Arbeit betrachteten Ansätzen, die Kinematik der jeweiligen Beine der PKM vorzugeben und diese für gewöhnlich keine geschlossene Ketten enthalten, ist sowohl die klassische als auch die modifizierte Notation möglich. Während in dem Programm nach [DTKHO09a], die auf die Berechnung serieller und paralleler Strukturen beschränkt ist, DENAVIT-HARTENBERG Parameter vorgegeben werden, verwendet die in dieser Arbeit für PKM nach [Abd07] erweiterte Toolbox [Sch] die modifizierten Parameter, da hiermit auch hybride Roboter berechnet werden können.

## 2.2 Dynamik

Die Dynamik eines Roboters gibt die Relation zwischen den eingebrachten Kräften und der daraus resultierenden Bewegung für die massebehafteten Körper an. Im Gegensatz zu seriellen Robotern, ist die Berechnung der Dynamik für parallele Roboter mit geschlossene Ketten wesentlich aufwendiger. Dies liegt vor allem daran, dass sich der Minimalkoordinatenraum vom Antriebskoordinatenraum unterscheidet und jedes Bein zudem aus zahlreichen passiven Gelenken besteht. Beim Aufstellen der Differentialgleichungen kommt es daher zu vielen Transformationen der Bewegungsgleichungen in unterschiedliche Koordinatenräume [AH09]:

Für einen Roboter mit $n$ Freiheitsgraden besteht das Dynamikmodell aus $n$ Differentialgleichungen. Darüber hinaus wird hierbei zwischen der direkten und inversen Dynamik unterschieden. Die direkte Dynamik

$$\ddot{\boldsymbol{x}}_{\mathrm{E}} = f_{\mathrm{dir}}(\dot{\boldsymbol{x}}_{\mathrm{E}}, \boldsymbol{x}_{\mathrm{E}}, \boldsymbol{q}, \boldsymbol{\tau}_{\mathrm{a}}, \boldsymbol{p}) \tag{2.12}$$

gibt die Beschleunigung der Minimalkoordinaten im Bezug auf die eingebrachten Kräfte an und wird vor allem in der Simulation von Roboterbewegungen verwendet. In der realen Anwendung hat die direkte Dynamik allerdings keine Relevanz, weshalb der Schwerpunkt dieser Arbeit auf der inversen Dynamik

$$\boldsymbol{\tau}_{\mathrm{a}} = f_{\mathrm{inv}}(\ddot{\boldsymbol{x}}_{\mathrm{E}}, \dot{\boldsymbol{x}}_{\mathrm{E}}, \boldsymbol{x}_{\mathrm{E}}, \boldsymbol{q}, \boldsymbol{p}) \tag{2.13}$$

liegt. Sie beschreibt die Kräfte, die in die aktiven Gelenke eingeleitet werden müssen, um eine vorgegebene Endeffektorbewegung durchführen zu können und wird hauptsächlich zum Zweck der Steuerung bzw. Regelung des Roboters sowie für die Simulation und Auslegung in der Maßsynthese eingesetzt. Der Vektor $\boldsymbol{p}$ aus (2.13) enthält alle physikalischen Parameter des Systems. In (2.12) und (2.13) ist zu erkennen, dass sowohl die Endeffektorkoordinaten als auch sämtliche passive und aktive Gelenkkoordinaten vorgegeben werden müssen. Bei den Geschwindigkeiten und Beschleunigungen reicht es hingegen die Endeffektorkoordinaten anzugeben. Dies hängt mit der inversen Kinematik aus Abschnitt 2.1.2 zusammen, bei der kein analytisches Verfahren zur Bestimmung aller aktiven und passiven Gelenkkoordinaten auf Basis der Endeffektorkoordinaten existiert. Die inverse Dynamik kann in verschiedenen Formen aufgestellt werden. Die in der Literatur am häufigsten auftretende allgemeine Form

$$\boldsymbol{\tau}_{\mathrm{a}} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{x}}_{\mathrm{E}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{x}}_{\mathrm{E}}) + \boldsymbol{g}(\boldsymbol{q}) \tag{2.14}$$

entspricht (2.13) mit expliziter Darstellung der einzelnen Dynamikterme. Wenn $n$ der Anzahl an Freiheitsgraden entspricht, ist $\boldsymbol{M}$ die $(n \times n)$-Massenmatrix, $\boldsymbol{c}$ der $(n \times 1)$-Coriolisvektor und

$\boldsymbol{g}$ der $(n \times 1)$-Gravitationsvektor. Die Gleichung der Form (2.14) ist zwar rechenaufwendiger als (2.13), es lässt sich daraus aber durch einfaches Umformen die Gleichung für die direkte Dynamik bestimmen [Abd07].

Zur Berechnung der Dynamikgleichungen eines Roboters existieren verschiedene Verfahren, die eingesetzt werden können: Das NEWTON-EULER-Verfahren, das LAGRANGE'sche Verfahren und Das Prinzip der virtuellen Arbeit bzw. Energie. Im Folgenden werden die ersten beiden Methoden vorgestellt, da die in dieser Arbeit betrachteten Algorithmen ausschließlich darauf beruhen.

### 2.2.1 Der LAGRANGE'sche Formalismus

Bewegungsgleichungen, die mittels des LAGRANGE'sche Formalismus aufgestellt werden, haben den Vorteil, dass sich die jeweiligen Dynamikterme direkt aus der LAGRANGE-Funktion berechnen lassen und somit eine gut strukturierte Form der Gleichung vorliegt. Bei dem Formalismus kann zwischen den Gleichungen 1. Art, die aus $n + m$ Differentialgleichungen stehen und die kinematischen Zwangsbedingungen enthalten, und 2. Art aus $n$ Differentialgleichungen, in denen die Zwangsbedingungen bereits eliminiert wurden, unterschieden werden. Während $n$ die Anzahl der Freiheitsgrade des Roboters und somit die unabhängigen Koordinaten angibt, entspricht $m$ der Anzahl aller abhängigen Koordinaten.

### 2.2.2 Die LAGRANGE'schen Gleichungen 1. Art

Die Grundlage der LAGRANGE'schen Gleichungen sind kinetische und potentielle Energie $T$ bzw. $U$, die für das Gesamtsystems aufgestellt und zur LAGRANGE-Funktion $L = T - U$ zusammengesetzt werden. Durch partielles Ableiten von $L$ nach den Roboterkoordinaten ergibt sich das Dynamikmodell zu

$$\boldsymbol{\tau}_q + \frac{\partial \boldsymbol{h}(\boldsymbol{q})}{\partial(\boldsymbol{q})}^{\mathrm{T}} \boldsymbol{\lambda} = \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{\boldsymbol{q}}} - \frac{\partial L}{\partial \boldsymbol{q}}, \tag{2.15}$$

das $n + m$ Differentialgleichungen enthält. Hierbei besteht der Vektor $\boldsymbol{q}$ aus $n$ unabhängigen und $m$ abhängigen Koordinaten und $\boldsymbol{\tau}_q$ aus den entsprechenden Kräften und Momenten. Da die Koordinaten durch $m$ kinematische Zwangsbedingungen miteinander verknüpft sind, müssen diese in den Gleichungen berücksichtigt werden. Dazu wird die Matrix $\partial \boldsymbol{h}(\boldsymbol{q})/\partial(\boldsymbol{q})$ multipliziert mit dem Vektor der LAGRANGE-Multiplikatoren $\boldsymbol{\lambda}$ addiert. Es existieren genauso viele LAGRANGE-Multiplikatoren wie Zwangsbedingungen vorhanden sind. $(\partial \boldsymbol{h}(\boldsymbol{q})/\partial(\boldsymbol{q}))^{\mathrm{T}}\boldsymbol{\lambda}$ ist der Vektor der Zwangskräfte, die eingebracht werden müssen, um die Zwangsbedingungen

einzuhalten. Mit allen $n + m$ Differentialgleichungen und den $m$ Zwangsbedingungen stehen genug Gleichungen zur Verfügung, um das Gleichungssystem zu lösen. Damit liegt sowohl die Lösung für die Zwangskräfte als auch für die gesuchte Bewegungsgleichung des Systems vor [Mei70].

Da die Zwangskräfte bei der Berechnung der inversen Dynamik nicht von Interesse sind und die Berechnung unnötig aufwendiger macht, wird deshalb auf die LAGRANGE'schen Gleichungen 2. Art zurückgegriffen. Diese lassen sich aus den Gleichungen 1. Art ableiten, indem die Zwangskräfte eliminiert werden, wie im Folgenden näher beschrieben wird.

### 2.2.3 Die LAGRANGE'schen Gleichungen 2. Art

Die LAGRANGE'schen Gleichungen 2. Art ergeben sich aus (2.15), wenn die partiellen Ableitungen der LAGRANGE-Funktion ausschließlich nach den generalisierte Koordinaten (Minimalkoordinaten) $\boldsymbol{z}$ gebildet werden. Die Minimalkoordinaten sind jene Koordinaten, die die kinematischen Zwangsbedingungen immer erfüllen. Dies hat zur Folge, dass die Zwangsbedingungen konstant bleiben und die Ableitung nach den Minimalkoordinaten $\partial \boldsymbol{h}(\boldsymbol{q})/\partial(\boldsymbol{z}) = \boldsymbol{0}$ ergibt. Somit erhält man ausgehend von (2.15) die allgemeine Form

$$\boldsymbol{\tau}_z + \underbrace{\frac{\partial \boldsymbol{h}(\boldsymbol{q})}{\partial(\boldsymbol{z})}^{\mathrm{T}}}_{=0} \boldsymbol{\lambda} = \boldsymbol{\tau}_z = \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{\boldsymbol{z}}} - \frac{\partial L}{\partial \boldsymbol{z}}. \tag{2.16}$$

Die Anzahl an Minimalkoordinaten eines Systems entsprechen der Anzahl dessen Freiheitsgrade $n$. Wie aus den Kinematikgrundlagen aus Abschnitt 2.1 bereits bekannt, ist die direkte Kinematik bei parallelen Robotern analytisch nicht lösbar, da für eine Konfiguration der Antriebskoordinaten mehrere Konfigurationen des Endeffektors existieren. Die kinematischen Zwangsbedingungen, die ebenfalls in Abschnitt 2.1 aufgestellt wurden, werden von den Antriebskoordinaten nicht immer erfüllt. Bei parallelen Systemen kommen für die Wahl der Minimalkoordinaten somit nur die Endeffektorkoordinaten in Frage [Mei70].

### 2.2.4 Das NEWTON-EULER-Verfahren

Die zweite, weitverbreitete Methode zur Berechnung der Bewegungsgleichungen ist das NEWTON-EULER-Verfahren. Hierbei wird das Gesamtsystem in seine $N$ Teilkörper freigeschnitten und der Impuls- und Drallsatz für jeden Teilkörper aufgestellt. Wenn das dabei entstehende Gleichungssystem mit $6N$ Gleichungen nach den aktiven Gelenkkräften aufgelöst wird, erhält man die inverse Dynamik des Systems. Ein Vorteil des Verfahrens gegenüber dem LAGRANGE'schen Formalismus ist, dass es eine höhere Recheneffizienz bietet. Dadurch ist das

Verfahren bei der Modellierung von PKMs verbreitet [Abd07]. Ein Nachteil ist, dass mit dem Verfahren auch die Schnittkräfte mitberechnet werden müssen. Die translatorische Bewegung wird für jeden Körper $i$ mit dem Impulssatz in differentieller Form

$$m_i \boldsymbol{a}_i = \boldsymbol{f}_{\mathrm{t}i} \tag{2.17}$$

beschrieben. Der Vektor $\boldsymbol{f}_{\mathrm{t}i}$ ist die Summe alle äußeren Kräfte, die auf den Körper $i$ wirken. $m_i$ ist die Masse des Körpers und $\boldsymbol{a}_i$ die Beschleunigung des Körpers in seinem Massenschwerpunkt $S_i$. Zur Berechnung der rotatorischen Bewegungsgleichung werden für jeden Teilkörper $i$ die Drallsätze um den Schwerpunkt $S_i$ gebildet

$$\boldsymbol{I}_i^{(S_i)} \dot{\boldsymbol{\omega}}_i + \tilde{\boldsymbol{\omega}}_i \left( \boldsymbol{I}_i^{(S_i)} \boldsymbol{\omega}_i \right) = \boldsymbol{m}_{\mathrm{t}i}^{(S_i)}. \tag{2.18}$$

$\boldsymbol{I}_i^{(S_i)}$ ist der Trägheitstensor, $\boldsymbol{\omega}_i$ der Vektor der Winkelgeschwindigkeiten und $\boldsymbol{m}_{\mathrm{t}i}^{(S_i)}$ der Vektor der äußeren Momente um den Schwerpunkt $S_i$. Bei parallelen Robotern wird das System in seine Teilsysteme zerlegt und die aufgestellten Drall- und Impulssätze rekursiv in einander eingesetzt [Abd07].

### 2.2.5 Dynamik bei parallelen Robotern

Durch die komplizierte Struktur paralleler Roboter muss ein erhöhtes Augenmerk auf die Anwendung der oben beschriebenen Berechnungsverfahren gelegt werden. Während man bei der Verwendung der NEWTON-EULER-Methode wenige Variationsmöglichkeiten beim Aufstellen der Gleichungen hat, können mit der LAGRANGE-Methode verschiedene Ansätze in Betracht gezogen werden. Grundsätzlich lässt sich das Prinzip der LAGRANGE'schen Gleichungen auf das geschlossene System des parallelen Roboters anwenden. Allerdings erhält man auf Grund der partiellen Ableitungen nach den Minimalkoordinaten sehr aufwendige Gleichungen, womit das Ergebnis für einen Parallelroboter mit vielen Freiheitsgraden nicht recheneffizient wäre. In [AH09] wird bewiesen, dass die geschlossenen Ketten der Grund für die ineffiziente Lösung sind und dass die Aufsummierung der Dynamikgleichungen für die einzelnen Teilketten und die Plattform auf das gleiche Ergebnis, wie das des geschlossenen Mechanismus, führen:

Es werde die LAGRANGE-Funktion $L = T - U$ mit den Minimalkoordinaten $\boldsymbol{z} = (\boldsymbol{r}_{\mathrm{P}}^{\mathrm{T}}, \boldsymbol{\varphi}_{\mathrm{P}}^{\mathrm{T}})^{\mathrm{T}}$ für 6 FHG für den geschlossenen Mechanismus und $L_t = T_t - U_t$ mit den Minimalkoordinaten jedes Teilsystems, die in $\boldsymbol{z}_t$ zusammengefasst werden, für das offene System betrachtet. Der LAGRANGE'sche Formalismus ergibt

$$\boldsymbol{\tau} = \frac{\mathrm{d}}{\mathrm{d}t} \frac{\partial L}{\partial \dot{\boldsymbol{z}}} - \frac{\partial L}{\partial \boldsymbol{z}} \tag{2.19}$$

und

$$\boldsymbol{\tau}_t = \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L_t}{\partial \dot{\boldsymbol{z}}_t} - \frac{\partial L_t}{\partial \boldsymbol{z}_t}. \tag{2.20}$$

Der Beweis aus [AH09] liefert für die virtuelle Arbeit, dass

$$\boldsymbol{\tau}\mathrm{d}\boldsymbol{z} = \left(\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{\boldsymbol{z}}} - \frac{\partial L}{\partial \boldsymbol{z}}\right)\mathrm{d}\boldsymbol{z} = \left(\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L_t}{\partial \dot{\boldsymbol{z}}_t} - \frac{\partial L_t}{\partial \boldsymbol{z}_t}\right)\mathrm{d}\boldsymbol{z}_t = \boldsymbol{\tau}_t\mathrm{d}\boldsymbol{z}_t. \tag{2.21}$$

Das heißt, dass beide Systeme die gleiche virtuelle Arbeit verrichten und deshalb bei Verwendung des D'ALAMBERT'sche Prinzips die gleichen Bewegungsgleichungen entstehen müssten. Mit anderen Worten wird die gleiche Energie vom System aufgebracht, unabhängig davon, ob die Koordinaten $\boldsymbol{z}$ mit $\boldsymbol{\tau}$ oder die Koordinaten $\boldsymbol{z}_t$ mit $\boldsymbol{\tau}_t$ bewegt werden. Vorausgesetzt, das offene und das geschlossene System führen die gleichen Bewegungen durch, was aber durch die kinematischen Zwangsbedingungen erzwungen wird. Eine weitere Erklärung für die Machbarkeit dieses Verfahrens ist die Berechnung über das NEWTON-EULER-Verfahren. Hierbei kann die Dynamik ebenfalls zunächst für jedes Subsystem des Roboters für sich erstellt werden. Anschließend werden die externen Kräfte der Teilsysteme durch geeignete Transformationen und die NEWTON-EULER-Methode miteinander verrechnet.

Es ist demnach möglich die Bewegungsgleichungen für jede Teilsysteme einzeln zu bestimmen und diese anschließend aufzusummieren. Dabei muss allerdings beachtet werden, dass die Gleichungen zuvor in ein gemeinsames Koordinatensystem transformiert werden. Bei geschickter Einteilung des Gesamtssystems in seine Teilsysteme, können die Abhängigkeiten von den Minimalkoordinaten vereinfacht und somit der Aufwand zur Berechnung der partiellen Ableitungen reduziert werden. Bei parallelen Robotern hat sich die Einteilung in die jeweiligen seriellen Ketten und die Roboterplattform bewährt. Die Gleichungen der seriellen Ketten $j$ werden bezüglich ihrer Gelenkkoordinaten $\boldsymbol{q}_j$ gebildet. Zur Berechnung der Dynamikgleichungen der Roboterplattform werden die Minimalkoordinaten des Gesamtsystems $\boldsymbol{z}$ verwendet [AH09].

# 3 Ansätze

In diesem Kapitel werden das zu implementierende Verfahren sowie die zu vergleichenden Ansätze für die Berechnung der PKM-Dynamik beschrieben und dabei die jeweiligen Gemeinsamkeiten und Unterschiede erläutert. Zunächst wird in Abschnitt 3.1 auf das Programm OPENSYMORO eingegangen, das die inverse Dynamik mittels des NEWTON-EULER-Verfahren bestimmt [KVK+14]. Abschnitt 3.2 und Abschnitt 3.3 beschreiben die Funktionsweise von PARADYN aus [DTKHO09a] bzw. dem zu implementierenden Ansatz (nach Abdellatif et al.), der in [AH09] beschrieben wird.

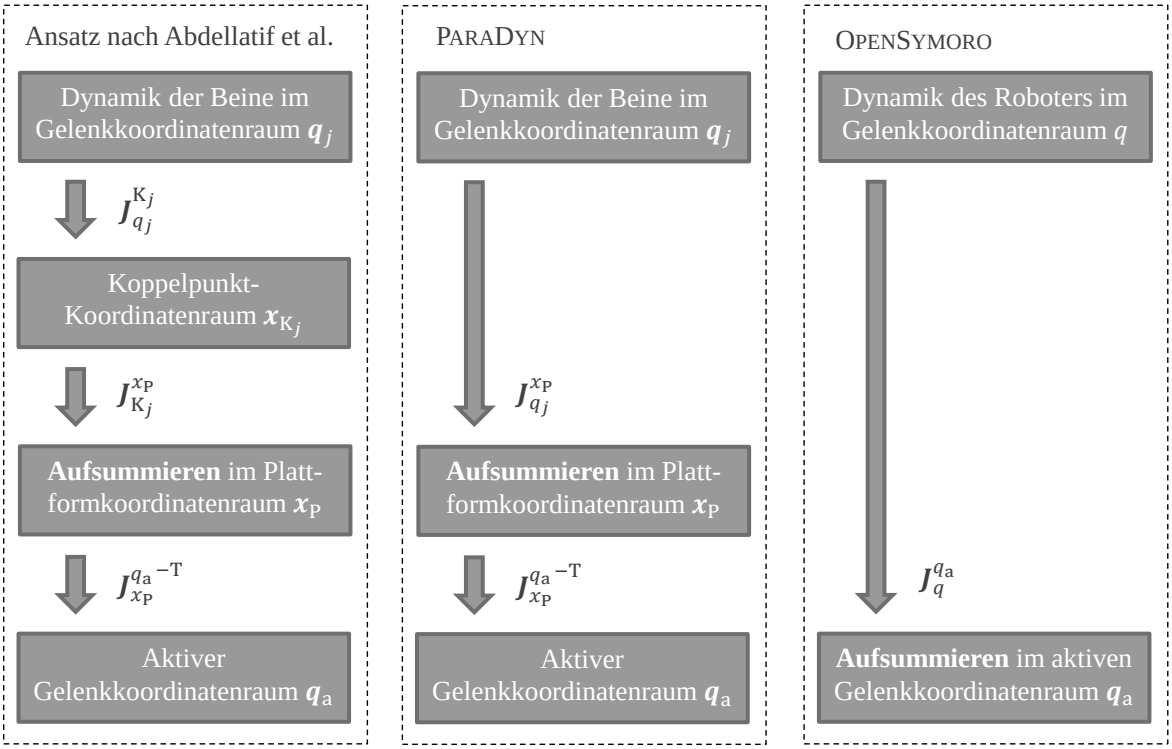



**Bild 3.1**: Schematische Darstellung der betrachteten Verfahren

Beide Ansätze verwenden die LAGRANGE'schen Gleichungen zur Dynamikmodellierung und unterscheiden sich hauptsächlich in der Projektion der Dynamikgleichungen in ein gemeinsames Koordinatensystem. In Bild 3.1 sind die Unterschiede dieser Projektionen schematisch

dargestellt. Grundsätzlich unterscheidet sich das Koordinatensystem der Plattform $(\mathrm{KS})_{\mathrm{P}}$ von dem des EE-Koordinatensystem $(\mathrm{KS})_{\mathrm{E}}$, was durch eine weitere Transformation berücksichtigt werden muss. Da diese Transformation aber nicht von der Konfiguration des Roboters abhängt, werden für alle Verfahren statt der EE-Koordinaten $\boldsymbol{x}_{\mathrm{E}}$ die Plattformkoordinaten $\boldsymbol{x}_{\mathrm{P}}$ verwendet.

## 3.1 Projektion durch implizit definierte Schließbedingungen (OPENSYMORO)

OPENSYMORO ist ein Open-Source Programm, dass 2014 von Khalil et al. veröffentlicht wurde und der symbolischen Modellierung von Robotern dient. Mit dem, in der Programmiersprache Python geschriebenen Programm, können sowohl Roboter mit offenen als auch mit geschlossenen Ketten modelliert werden. Zudem ist es möglich Roboter mit fester und beweglicher Basis, sowie solche mit Rädern zu berechnen. Als Vorgabe erwartet die graphische Oberfläche die Anzahl der Gelenke und bei Robotern mit geschlossenen Ketten die Gliederanzahl. Die Geometrie des Roboters wird mit modifizierten DENAVIT-HARTENBERG-Parametern bestimmt [KVK+14].

### 3.1.1 Form des Ergebnisses

Das Ziel bei der Bestimmung eines Dynamikmodells ist es, die aktiven Gelenkkräfte und -momente in Abhängigkeit der Roboter Bewegung zu erhalten. Grundsätzlich ist es möglich diese Bewegung alleine durch alle Systemkoordinaten und die Minimalgeschwindigkeiten und -beschleunigungen beschreiben zu können. Da in OPENSYMORO die Dynamik mittels der NEWTON-EULER-Methode im Gelenkkoordinatenraum aufgestellt wird, liegt wird das Dynamikmodell in der Form

$$\begin{aligned} \boldsymbol{\tau}_q &= \boldsymbol{f}_{\mathrm{dyn}}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}, \boldsymbol{p}), \\ \boldsymbol{\tau}_{\mathrm{a}} &= \boldsymbol{J}_q^{q_{\mathrm{a}}} \boldsymbol{\tau}_q \end{aligned} \tag{3.1}$$

mit dem Parametervektor $\boldsymbol{p}$ und allen Gelenkwinkeln $\boldsymbol{q}$ ausgegeben. Mit dieser Formel werden zunächst alle Kräfte der passiven und aktiven Gelenke $\boldsymbol{\tau}_q$ bestimmt, um das Ergebnis im Anschluss über $\boldsymbol{J}_q^{q_{\mathrm{a}}}$ in den Raum der aktiven Gelenke $\boldsymbol{\tau}_{\mathrm{a}}$ zu projizieren. Aufgrund der Verwendung der NEWTON-EULER-Methode zur Bestimmung des Modells, ist es nicht möglich, die Gleichung mit getrennten Dynamiktermen für den Intertial-, Coriolis- und Gravitationsanteil zu erhalten.

### 3.1.2 Das Verfahren

Im Gegensatz zu den anderen, später beschriebenen Verfahren, ist der Ansatz aus OPENSYMORO nicht auf PKMs beschränkt. Hiermit können jegliche Systeme von seriellen und baumstrukturartigen Robotern bis hin zu Robotern mit geschlossenen Ketten berechnet werden. Dies wird unter anderem dadurch ermöglicht, dass hier zur Dynamikmodellierung das NEWTON-EULER-Verfahren herangezogen wird. Denn im Gegensatz zum LAGRANGE'schen Verfahren fallen bei der NEWTON-EULER-Methode die Überlegungen zum Koordinatenraum, in dem die Bewegungsgleichungen aufgestellt werden, weg. Im Folgenden wird zunächst die Bestimmung der Kinematik in Abschnitt 3.1.2 und darauf aufbauend das Dynamikmodell in Abschnitt 3.1.2 beschrieben.

#### Kinematik

Zur Modellbestimmung bei Robotern mit geschlossenen Ketten, wird das System in eine äquivalente Baumstruktur überführt. Dies geschieht indem die Schleifen an bestimmten Gelenken aufgeschnitten werden, aber davon ausgegangen wird, dass die beiden Koordinatensysteme der so entstandenen freien Enden übereinander liegen. Eine geeignete Wahl der Schnittgelenke bei parallelen Robotern sind die Koppelgelenke. Das Ziel bei der Bestimmung des Kinematikmodells bei parallelen Robotern ist der differentielle Zusammenhang zwischen den aktiven Gelenken und den Plattformkoordinaten. Bei OPENSYMORO wird hierfür zunächst der differentiellen Zusammenhangs zwischen den Plattformkoordinaten und allen Gelenken entlang einer Kette

$$\dot{\boldsymbol{x}}_{\mathrm{P}} = \frac{\partial \boldsymbol{f}(\boldsymbol{q})}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} = \boldsymbol{J}_q^{x_{\mathrm{P}}} \dot{\boldsymbol{q}} \tag{3.2}$$

gebildet. Da im Prinzip an jedem Teilglied der Endeffektor montiert werden könnte, muss das Plattformkoordinatensystem vorgegeben werden. Da die in Abschnitt 2.1.4 beschriebene differentielle Kinematik nur von den aktiven Gelenken abhängt, müssen die passiven Gelenke in (3.2) durch kinematische Zwangsbedingungen eliminiert werden [KVK+14].

Zur Bestimmung der Beziehung zwischen den passiven und aktiven Gelenken werden die Geschwindigkeiten der Koordinatensysteme an den Enden freien Enden einer geöffneten Kette gebildet. Da davon ausgegangen wird, dass beide Koordinatensysteme exakt übereinander liegen, können die so berechneten Geschwindigkeiten gleich gesetzt werden und sodass man

$$(\boldsymbol{v}_k^{\mathrm{T}}, \boldsymbol{\omega}_k^{\mathrm{T}})^{\mathrm{T}} = \boldsymbol{J}_{q1}^k \dot{\boldsymbol{q}}_1 = \boldsymbol{J}_{q2}^k \dot{\boldsymbol{q}}_2 \tag{3.3}$$

erhält. $k$ steht für das geschnittene Koordinatensystem und $\dot{\boldsymbol{q}}_1$ bzw. $\dot{\boldsymbol{q}}_2$ für die Gelenkgeschwin-

digkeiten entlang einer der beiden Ketten. Multipliziert mit den beiden Jacobi-Matrzen $\boldsymbol{J}_{q1}^{k}$ und $\boldsymbol{J}_{q2}^{k}$ ergibt sich die Geschwindigkeiten $(\boldsymbol{v}_k^{\mathrm{T}}, \boldsymbol{\omega}_k^{\mathrm{T}})^{\mathrm{T}}$ das geschnittene Koordinatensystem. Durch Kombination dieser Gleichungen für alle geschlossenen Ketten ergibt sich

$$\begin{pmatrix} \boldsymbol{W}_{\mathrm{a}} & \boldsymbol{W}_{\mathrm{P}} & 0 \\ \boldsymbol{W}_{\mathrm{ac}} & \boldsymbol{W}_{\mathrm{pc}} & \boldsymbol{W}_{\mathrm{c}} \end{pmatrix} \begin{pmatrix} \dot{\boldsymbol{q}}_{\mathrm{a}} \\ \dot{\boldsymbol{q}}_{\mathrm{p}} \\ \dot{\boldsymbol{q}}_{\mathrm{c}} \end{pmatrix} = 0, \tag{3.4}$$

wobei $\dot{\boldsymbol{q}}_{\mathrm{a}}$, $\dot{\boldsymbol{q}}_{\mathrm{p}}$ und $\dot{\boldsymbol{q}}_{\mathrm{c}}$ die aktiven, passiven und geschnittenen Gelenke repräsentieren. Durch Umstellen der ersten Zeile aus (3.4) erhält man

$$\dot{\boldsymbol{q}}_{\mathrm{p}} = -\boldsymbol{W}_{\mathrm{P}}^{-1} \boldsymbol{W}_{\mathrm{a}} \dot{\boldsymbol{q}}_{\mathrm{a}} = \boldsymbol{J}_{q_{\mathrm{a}}}^{q_{\mathrm{p}}} \dot{\boldsymbol{q}}_{\mathrm{a}}. \tag{3.5}$$

Durch Einsetzen von (3.5) in (3.2) ergibt sich die gewünschte differentielle Kinematik in Abhängigkeit der aktiven Gelenkwinkel [KC97].

### Dynamik

In Abschnitt 2.2.4 wurde erklärt wie mittels des NEWTON-EULER-Verfahrens die Bewegungsgleichungen für die einzelnen Starrkörper eines Roboters bestimmt werden können. Zur Berechnung der inversen Dynamik werden die durch Freischneiden der einzelnen Glieder entstandenen Gleichungen rekursiv in einander eingesetzt. Für das in OPENSYMORO eingesetzte Verfahren zur Bildung des Dynamikmodells wird das geschlossene System in eine äquivalente Baumstruktur überführt. Hierbei müssen Schnittgelenke gewählt werden, in denen die geschlossenen Ketten geöffnet werden.

Das Verfahren besteht demnach aus zwei Schritten: In der Vorwärts-Rekursion werden die Dynamikkräfte in Abhängigkeit der Geschwindigkeiten und Beschleunigungen bestimmt. Die Reaktionskräfte zwischen den Teilkörpern werden hierbei vorerst nicht berücksichtigt. Für jeden Körper $i = 1, ..., N$ des Systems ergeben sich die Impuls- und Drallsätze zu

$$\begin{aligned} \boldsymbol{f}_{\mathrm{t}i} &= m_i \boldsymbol{a}_i, \\ \boldsymbol{m}_{\mathrm{t}i} &= \boldsymbol{I}_i^{(S_i)} \dot{\boldsymbol{\omega}}_i + \tilde{\boldsymbol{\omega}}_i \left( \boldsymbol{I}_i^{(S_i)} \boldsymbol{\omega}_i \right). \end{aligned} \tag{3.6}$$

Den zweiten Schritt bildet eine Rückwärts-Rekursion, bei der die entstandenen Bewegungsgleichungen der Teilglieder über die jeweiligen Schnittkräfte miteinander verknüpft werden. Für jedes freigeschnitten Teilglied ergeben sich für $i = N, ..., 1$ die Funktionen

$$\begin{aligned} \boldsymbol{f}_i &= \boldsymbol{f}_{\mathrm{t}i} + \boldsymbol{f}_j, \\ \boldsymbol{m}_i &= \boldsymbol{m}_{\mathrm{t}i} + \boldsymbol{m}_j + \tilde{\boldsymbol{r}}_i^{S_i} \boldsymbol{f}_i + \tilde{\boldsymbol{r}}_j^{S_i} \boldsymbol{f}_j. \end{aligned} \tag{3.7}$$

Hierbei ist $i$ das auf $j$ folgende Gelenk, sodass $i = a(j)$. Der Tilde-Operator beschreibt das Kreuzprodukt zweier Vektoren durch ein Produkt aus Matrix und Vektor $a \times b = \tilde{a}b$. Bei mehreren Nachfolgern müssen die Schnittkräfte und -momente aller Folgeglieder berücksichtigt werden. Die Kräfte bzw. Momente $\boldsymbol{f}_i$ bzw. $\boldsymbol{m}_i$ sind die die Reaktionskräfte zwischen den Teilgliedern [KC97].

Die Bestimmung der Antriebskräfte $\tau_i$ jedes Gelenks ist davon abhängig, ob es es sich um ein translatorisches oder rotatorisches Gelenk handelt sowie entlang welcher Achse es sich bewegen kann. Mit Verwendung der klassischen bzw. modifizierten DENAVIT-HARTENBERG-Notation ist bereits festgelegt, dass sich die Gelenke um bzw. entlang der $z$-Achse bewegen können. Für das Gelenk $i$ ergibt sich somit

$$\begin{aligned} \tau_i &= \boldsymbol{f}_i^{\mathrm{T}} \boldsymbol{a}_{z_i} \ \text{ bzw.} \\ \tau_i &= \boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{a}_{z_i}, \end{aligned} \tag{3.8}$$

wobei $\boldsymbol{a}_{z_i} = (0,0,1)^{\mathrm{T}}$ zur Auswahl der $z$-Komponente des Vektors genutzt wird. Die so entstandenen Bewegungsgleichungen liefern die Kräfte aller Gelenke $\boldsymbol{\tau_q} = (\boldsymbol{\tau}_{\mathrm{a}}^{\mathrm{T}}, \boldsymbol{\tau}_{\mathrm{p}}^{\mathrm{T}})^{\mathrm{T}}$ des Roboters. Um die vollständige inverse Dynamik, die ausschließlich die Kräfte der aktuierten Gelenke enthält, zu erhalten, ist es notwendig alle Kräfte auf die aktiven Gelenke zu projizieren. Für diese Projektion wird Gleichung (3.5) verwendet, sodass

$$\boldsymbol{\tau}_{\mathrm{a_m}} = \boldsymbol{\tau}_{\mathrm{a}} + \boldsymbol{J}_{q_{\mathrm{a}}}^{q_{\mathrm{p}}-\mathrm{T}} \boldsymbol{\tau}_{\mathrm{p}} \tag{3.9}$$

gilt. Der Vektor $\boldsymbol{\tau}_{\mathrm{a_m}}$ gibt hierbei die endgültigen Motorkräfte an, während $\boldsymbol{J}_{q_{\mathrm{a}}}^{q_{\mathrm{p}}\,\mathrm{T}} \boldsymbol{\tau}_{\mathrm{p}}$ der Projektion des passiven auf den aktiven Gelenkkoordinatenraum entspricht [KC97].

## 3.2 Projektion auf Plattformkoordinaten durch Schließbedingungen (PARADYN)

PARADYN wurde am Institut für mechatronische Systeme an der Leibniz Universität Hannover von Do Tanh et al. in dem Computer-Algebra-System MAPLE entwickelt. Dem Programm werden lediglich die DENAVIT-HARTENBERG- und die pyhsikalischen Parameter sowie die aktuierten und Minimalkoordinaten, -geschwindigkeiten und -beschleunigungen vorgegeben. Als Ergebnis erhält man das symbolisch berechnete Kinematik- und Dynamikmodell in optimiertem Matlab-Code [DTKHO09a]. Dieses Kapitel ist in zwei Teile gegliedert: Abschnitt 3.2.1 beschreibt, wie das Ergebnis des Dynamikmodells aussehen soll und Abschnitt 3.2.2 erläutert die Vorgehensweise in dem Verfahren.

Während beim Ansatz nach Abdellatif et al. das Dynamikmodell für jedes Subsystem mit eigenen Gleichungen, getrennt voneinander beschrieben wird, betrachtet PARADYN die Gleichungen zusammengesetzt in einem einzigen Vektor. In dieser Koordinatenteilungsmethode werden die Gleichungen aller Subsysteme zusammengefügt und anschließend über eine Projektionsmatrix $\boldsymbol{J}_{q}^{q_a}$ auf die Minimalform reduziert. Bei Abdellatif et al. wird jede Gleichung für sich projiziert, um sie darauffolgend zur Minimalform aufzusummieren. Dies ist lediglich eine Stilfrage und hat somit keinen Einfluss auf die Struktur der endgültigen Lösung. Der Übersicht halber wird in dieser Arbeit für beide Verfahren die selbe Darstellung verwendet.

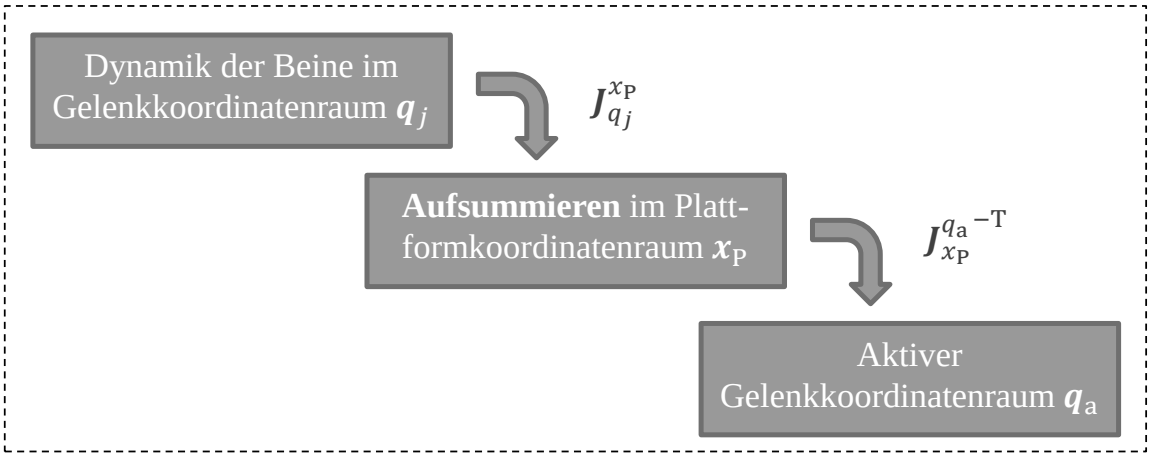



**Bild 3.2**: Schematische Darstellung von PARADYN

### 3.2.1 Form des Ergebnisses

Die inverse Dynamik beschreibt die eingeprägten Kräfte in Abhängigkeit einer vorgegebenen Systembewegung bzw. der Koordinaten, Geschwindigkeiten und Beschleunigungen. Das Ziel ist es das Dynamikmodell in Minimalform

$$\boldsymbol{\tau}_a = \boldsymbol{f}_{\mathrm{dyn}}(\ddot{\boldsymbol{z}}, \dot{\boldsymbol{z}}, \boldsymbol{z}, \boldsymbol{q}, \boldsymbol{p}) \tag{3.10}$$

zu erhalten, bei der die Anzahl der Gleichungen der Anzahl an Freiheitsgraden und der aktiven Gelenke des Roboters entsprechen. Mit der expliziten Darstellung der einzelnen Terme wie in Abschnitt 2.2.1 als Differentialgleichung 2. Ordnung erhält man

$$\boldsymbol{\tau}_\mathrm{a} = \boldsymbol{M}_\mathrm{a}(\boldsymbol{q}_\mathrm{ges})\ddot{\boldsymbol{z}} + \boldsymbol{c}_\mathrm{a}(\boldsymbol{q}_\mathrm{ges}, \dot{\boldsymbol{z}}) + \boldsymbol{g}_\mathrm{a}(\boldsymbol{q}_\mathrm{ges}), \tag{3.11}$$

wobei $\dot{\boldsymbol{z}}$ bzw. $\ddot{\boldsymbol{z}}$ den $(n \times 1)$ Minimalgeschwindigkeiten bzw. -beschleunigungen und $\boldsymbol{\tau}_a$ dem Kraftvektor der Antriebe für $n$ unabhängige Koordinaten entspricht. $\boldsymbol{M}$ ist die $(n \times n)$ Massenmatrix, $\boldsymbol{c}$ der $(n \times 1)$ Coriolisvektor, $\boldsymbol{g}$ der $(n \times 1)$ Gravitationsvektor und $\boldsymbol{p}$ der $(n \times 1)$ Parametervektor. $\boldsymbol{q}_{\mathrm{ges}}$ ist der $(f \times 1)$ Vektor, der alle $f = n + m$ Systemkoordinaten einbezieht. Für die Minimalkoordinaten stehen sowohl die aktuierten Gelenke als auch die Koordinaten der Plattform zur Auswahl. Aufgrund der Schwierigkeit die Gleichungen in Abhängigkeit der aktiven Gelenke in symbolischer Form zu bilden, kommt aber nur letzteres in Frage [DTKHO09a].

Wie bereits in Abschnitt 2.2.1 erwähnt, hat die Form in (3.11) den Vorteil, dass die einzelnen Terme des Dynamikmodells explizit dargestellt sind. Aufgrund dieser Tatsache eignet sich die Gleichung sehr gut für die Berechnung der direkten Dynamik oder zum Entwurf von Regelungen und bietet im Allgemeinen die größte Flexibilität in der Darstellung. Z. B. können durch die enstprechende Jacobi-Matrix die Minimalbeschleunigungen $\ddot{\boldsymbol{z}}$ durch den Vektor der Antriebsbschleunigungen $\ddot{\boldsymbol{q}}_a$ ersetzt werden, wie in [AH09] gefordert. In dieser Arbeit wird sich auf die Darstellung nach (3.11) beschränkt, um eine einheitliche Darstellung zu gewährleisten.

### 3.2.2 Das Verfahren

Nachdem in Abschnitt 2.2.5 gezeigt wurde, dass das Aufteilen in Subsysteme zu kürzeren Ausdrücken in den Bewegungsgleichungen führt, wird der Parallelroboter in seine einzelnen seriellen Ketten sowie die bewegte Plattform separiert. Wenn $j$ der Indizierung der Beine entspricht, kann die Dynamik der Beine $\boldsymbol{\tau}_j$ mit Hilfe der LAGRANGE'schen Gleichungen bezüglich der Gelenkkoordinaten $\boldsymbol{q}_j$ bestimmt werden. Häufig treten symmetrische Roboter auf, bei denen es genügt die Gleichungen für ein Bein aufzustellen und anschließend für alle anderen Beine zu übernehmen. Die Roboterplattform besteht lediglich aus einem Starrkörper, dessen Dynamikgleichungen $\boldsymbol{\tau}_{\mathrm{P}}$ ebenfalls mittels des LAGRANGE'schen Formalismus oder des NEWTON-EULER-Verfahrens bestimmt werden können. Um die Ergebnisse der Teilsysteme zusammenzuführen, müssen die Gleichungen in einen gemeinsamen Koordinatenraum überführt werden. Anschließend können die Ergebnisse aufsummiert werden. Hierzu wird in PARADYN das kartesische Koordinatensystem, in dem die Bewegung der Roboterplattform beschrieben wird, verwendet. Die Kräfte bzw. Momente der aktuierten Gelenke $\boldsymbol{\tau}_{\mathrm{a}}$ erhält man, indem die Kräfte über die entsprechende Jacobi-Matrix $\boldsymbol{J}_{\boldsymbol{z}}^{\boldsymbol{q}_{\mathrm{a}}-\mathrm{T}}$ in den Raum des aktiven Gelenkraums projiziert werden. Damit das Ergebnis der Form aus (3.10) bzw. (3.11) entspricht, gilt es alle Gelenkgeschwindigkeiten und -beschleunigungen $\dot{\boldsymbol{q}}_j$ und $\ddot{\boldsymbol{q}}_j$ durch die Minimalgeschwindigkeiten und -beschleunigungen $\dot{\boldsymbol{z}}$ und $\ddot{\boldsymbol{z}}$ zu ersetzen [DTKHO09b].

### Kinematik

Bild 3.2 kann entnommen werden, inwiefern sich die Ansätze in der Wahl und der Projektion in ein gemeinsames Koordinatensystem unterscheiden. Diese Transformation erfolgt, wie bereits aus Abschnitt 2.2 bekannt, über eine Multiplikation mit der jeweiligen Jacobi-Matrix $\boldsymbol{J}$. Die Bewegungsgleichungen der Plattform liegen bereits bezüglich der kartesischen Koordinaten vor, sodass hier ausschließlich die Dynamikmodelle der seriellen Ketten betrachtet werden müssen. In Bild 3.2 ist zu erkennen, dass in PARADYN die Bewegungsgleichungen aus dem Gelenkraum ohne Umweg über ein Zwischenkoordinatensystem in den kartesischen Raum transformiert werden. Da ein Parallelroboter geschlossene kinematische Ketten aufweist, müssen zuvor die kinematischen Zwangsbedingungen aufgestellt werden. Durch zeitliches Ableiten der so entstanden Gleichungen erhält man

$$\frac{\partial \boldsymbol{h}(\boldsymbol{z}, \boldsymbol{q}_j)}{\partial \boldsymbol{z}} \dot{\boldsymbol{z}} + \frac{\partial \boldsymbol{h}(\boldsymbol{z}, \boldsymbol{q}_j)}{\partial \boldsymbol{q}_j} \dot{\boldsymbol{q}}_j = \boldsymbol{A}\dot{\boldsymbol{z}} + \boldsymbol{B}\dot{\boldsymbol{q}}_j = \boldsymbol{0}. \tag{3.12}$$

Durch Umstellen der Gleichung

$$\dot{\boldsymbol{q}}_j = -\boldsymbol{B}^{-1}\boldsymbol{A}\dot{\boldsymbol{z}} = \boldsymbol{J}_z^{q_j}\dot{\boldsymbol{z}} \tag{3.13}$$

kann die Jacobi-Matrix für einen parallelen Roboter bestimmt werden. $\boldsymbol{h}(\boldsymbol{z}, \boldsymbol{q}_j)$ beschreibt die aufgestellten kinematischen Zwangsbedingungen, $\boldsymbol{z}$ die Minimal- bzw. Plattformkoordinaten des parallelen Mechanismus, $\boldsymbol{q}_j$ die Minimalkoordinaten der seriellen Kette $j$ und $\boldsymbol{J}_z^{q_j}$ die gesuchte Projektionsjacobi-Matrix [DTKHO09a]. Für eine detaillierte Erklärung für Herleitung der kinematischen Zwangsbedingungen sei auf Abschnitt 2.1 verwiesen.

### Dynamik der seriellen Ketten

Für die Dynamik der seriellen Ketten ist es das Ziel, das Dynamikmodell in der Form

$$\boldsymbol{\tau}_j = \boldsymbol{M}_j(\boldsymbol{q}_j)\ddot{\boldsymbol{q}}_j + \boldsymbol{c}_j(\dot{\boldsymbol{q}}_j, \ddot{\boldsymbol{q}}_j) + \boldsymbol{g}_j(\boldsymbol{q}_j) \tag{3.14}$$

mit den expliziten Termen für die Inertialmatrix $\boldsymbol{M}_j$ sowie den Coriolis- und Gravitationsvektor $\boldsymbol{c}_j$ bzw. $\boldsymbol{g}_j$ zu erhalten. $\boldsymbol{\tau}_j$ beschreibt die eingeprägten Gelenkmomente und -kräfte. In Abschnitt 2.2.1 wurde gezeigt, wie die Bewegungsgleichungen eines mechanischen Systems mit Hilfe des LAGRANGE'schen Ansatzes und der enthaltenen Energie hergeleitet werden können. Für serielle Strukturen aus $N_j$ Teilkörpern ergeben sich die kinetische und potentielle Energie-Gleichung zu

$$T_j = \frac{1}{2} \sum_{k=1}^{N_j} \left( \boldsymbol{\omega}_k^{\mathrm{T}} \boldsymbol{I}_k^{(S_k)} \boldsymbol{\omega}_k + m_k \boldsymbol{v}_{S_k}^{\mathrm{T}} \boldsymbol{v}_{S_k} \right) \tag{3.15}$$

und

$$U_j = \sum_{k=1}^{N_j} m_k \boldsymbol{g}_{\mathrm{grav}}^T \boldsymbol{r}_{S_k}. \tag{3.16}$$

Die Winkelgeschwindigkeiten bzw. Schwerpunktgeschwindigkeiten eines Körpers können über $\boldsymbol{\omega}_k$ bzw. $\boldsymbol{v}_{S_k}$ ausgedrückt werden. $m_k$ bzw. $\boldsymbol{I}_k^{(S_k)}$ entsprechen der Masse bzw. Trägheitstensor eines Körpers. Zur Berechnung der potentiellen Energie $U_j$ wird der Gravitationsvektor $\boldsymbol{g}$ mit den Schwerpunktkoordinaten $\boldsymbol{r}_{S_k}$ eines jeden Körpers multipliziert. Die Winkel- und Schwerpunktgeschwindigkeiten sind linear von den Gelenkgeschwindigkeiten $\dot{\boldsymbol{q}}_j$ abhängig und können in Matrixform

$$\boldsymbol{\omega}_k = \boldsymbol{J}_{\mathrm{R}_k} \dot{\boldsymbol{q}}_j \quad \text{und} \quad \boldsymbol{v}_{\mathrm{C}_k} = \boldsymbol{J}_{\mathrm{T}_k} \dot{\boldsymbol{q}}_j \tag{3.17}$$

ausgedrückt werden. $\boldsymbol{J}_{\mathrm{R}_k}$ und $\boldsymbol{J}_{\mathrm{T}_k}$ sind die rotatorische und translatorische Jacobi-Matrix für den Körper $k$. Über die partiellen Ableitungen der LAGRANGE'schen Funktion $L_j = T_j - U_j$ und durch Einsetzen von (3.17) in (3.15) kann die kinetische Energie der seriellen Kette mit

$$T_j = \frac{1}{2} \left( \dot{\boldsymbol{q}}_j^{\mathrm{T}} \boldsymbol{M}_j \dot{\boldsymbol{q}}_j \right) \tag{3.18}$$

ausgedrückt werden, wobei die Massenmatrix als

$$\boldsymbol{M}_j = \sum_{k=1}^{N_j} \left( \boldsymbol{J}_{\mathrm{R}_k}^{\mathrm{T}} \boldsymbol{I}_k^{(S_k)} \boldsymbol{J}_{\mathrm{R}_k} + m_k \boldsymbol{J}_{\mathrm{T}_k}^{\mathrm{T}} \boldsymbol{J}_{\mathrm{T}_k} \right) \tag{3.19}$$

definiert ist. Somit kann die Berechnung der Massenmatrix $\boldsymbol{M}_j$ direkt über (3.19) implementiert werden [DTKHO09a]. Aus der partiellen Ableitung $\partial T_j / \partial \boldsymbol{q}_j$ in (2.16) kann von $\boldsymbol{M}_j$ auf den Coriolisvektor $\boldsymbol{c}_j$

$$c_{j,\epsilon} = \sum_{k=1}^{N_j} \sum_{l=1}^{N_j} \left( \frac{\partial M_{j,\epsilon k}}{\partial z_{j,l}} - \frac{1}{2} \frac{\partial M_{j,kl}}{\partial z_{j,\epsilon}} \right) \dot{z}_{j,k} \dot{z}_{j,l} \tag{3.20}$$

in (3.14) geschlossen werden. Der Index $\epsilon = 1...n_j$ gibt eine der Komponenten von $c_j$ wieder. Die Terme

$$g_{j,\epsilon} = \sum_{k=1}^{N_j} m_k \boldsymbol{g}_{\mathrm{grav}}^{\mathrm{T}} \frac{\partial \boldsymbol{r}_{\mathrm{C}_k}}{\partial z_{j,\epsilon}} \tag{3.21}$$

des Gravitationsvektors $\boldsymbol{g}_j$ in (3.14) werden aus der partiellen Ableitung $\partial U_j / \partial \boldsymbol{q}_j$ in (2.16) ermittelt [AH09].

#### Dynamik der Plattform

Bei der Plattform handelt es sich um einen einzelnen Starrkörper, dessen Energie-Gleichungen sich durch

$$T_{\mathrm{P}} = \frac{1}{2}\left(\boldsymbol{\omega}_{\mathrm{P}}^{\mathrm{T}} \boldsymbol{I}_p^{(S_{\mathrm{P}})} \boldsymbol{\omega}_{\mathrm{P}} + m_{\mathrm{P}} \boldsymbol{v}_{S_{\mathrm{P}}}^{\mathrm{T}} \boldsymbol{v}_{S_{\mathrm{P}}}\right) \quad \text{und} \quad U_{\mathrm{P}} = m_{\mathrm{P}} \boldsymbol{g}_{\mathrm{grav}}^{T} \boldsymbol{x} \tag{3.22}$$

beschreiben lassen. Die Position der Plattform wird mit dem Vektor $\boldsymbol{x}$ beschrieben, der die Position des Plattformursprungs relativ zum kartesischen Inertialkoordinatensystem $(\mathrm{KS})_0$ beschreibt. Die Orientierung wird mit EULER-Winkeln beschrieben [DTKHO09a], da dies, im Gegensatz zu z. B. Quaternionen, eine Minimaldarstellung ist. Basierend auf dem LAGRANGE'schen Verfahren, können zur Berechnung der Dynamikgleichungen der Plattform die selben Gleichungen wie die der serielle Kette aus Abschnitt 3.2.2 verwendet werden. Wird die Form aus (3.14) verwendet, erhält man für die Dynamik

$$\boldsymbol{\tau}_{\mathrm{P}} = \boldsymbol{M}_{\mathrm{P}}(\boldsymbol{z})\ddot{\boldsymbol{z}} + \boldsymbol{c}_{\mathrm{P}}(\boldsymbol{z}, \dot{\boldsymbol{z}}) + \boldsymbol{g}_{\mathrm{P}}(\boldsymbol{z}). \tag{3.23}$$

#### Projektion und Summieren der Gleichungen

Die nun vorliegenden Gleichungen für $\boldsymbol{\tau}_p$ und $\boldsymbol{\tau}_j$ aus (3.23) und (3.14) mit den oben genannten kinematischen Zwangsbedingungen $\boldsymbol{h}(\boldsymbol{z}, \boldsymbol{q}_j)$ aus (3.12) decken sich mit den Ergebnissen, die die LAGRANGE'schen Gleichungen 1. Art

$$\boldsymbol{\tau} + \boldsymbol{\lambda}\frac{\partial \boldsymbol{h}(\boldsymbol{z}, \boldsymbol{q}_j)}{\partial \boldsymbol{q}_{\mathrm{ges}}} = \boldsymbol{M}(\boldsymbol{q}_{\mathrm{ges}})\ddot{\boldsymbol{q}}_{\mathrm{ges}} + \boldsymbol{c}(\dot{\boldsymbol{q}}_{\mathrm{ges}}, \dot{\boldsymbol{q}}_{\mathrm{ges}}) + \boldsymbol{g}(\boldsymbol{q}_{\mathrm{ges}}) \tag{3.24}$$

mit $\boldsymbol{q}_{\mathrm{ges}} = (\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{z})^{\mathrm{T}}$ wie in Abschnitt 2.2.2 ergeben. Damit liegt das vollständige Dynamikmodell der parallelen Struktur vor, allerdings in Abhängigkeit der Gelenkkoordinaten und der Minimalkoordinaten. Das Ziel ist es jedoch dieses Modell einerseits in Minimalform darzustellen, um die geringste Anzahl an Gleichungen zu erhalten und damit den Rechenaufwand zu minimieren. Andererseits sollen ausschließlich die eingeprägten Momente bzw. Kräfte der aktiven Gelenke $\boldsymbol{\tau}_a$, wie in (3.11), explizit in den Gleichungen stehen.

Um die gewünschte Form zu erhalten, werden zunächst die Dynamikmodelle jeder seriellen Kette in den kartesischen Raum und somit in den Raum der bereits aufgestellten Plattformdynamik transformiert. Dabei werden die jeweiligen Gleichungen mit der Jacobi-Matrix $\boldsymbol{J}_z^{q_j}$ aus (3.13), die von den kinematischen Zwangsbedigungen abgeleitet wird, multipliziert. Da nun

alle Gleichungen in kartesischen Koordinaten vorliegen, können diese aufsummiert werden. In [DTKHO09a] wird dies durch eine Multiplikation von (3.24) mit der transponierten Jacobi-Matrix aus $\dot{\boldsymbol{q}} = \boldsymbol{J}_z^{q\mathrm{T}}\dot{\boldsymbol{z}} = (\boldsymbol{J}_z^{q_1\mathrm{T}}, \boldsymbol{J}_z^{q_2\mathrm{T}}, \boldsymbol{J}_z^{q_{Ns}\mathrm{T}}, ..., \boldsymbol{I})^{\mathrm{T}}\dot{\boldsymbol{z}}$ für alle $N_s$ Beine, die die Beziehung zwischen allen Systemkoordinaten und den Minimalkoordinaten beschreibt, ausgedrückt. Als Ergebnis erhält man die Minimalform der Dynamikgleichungen des gesamten Systems

$$\boldsymbol{\tau}_z = \boldsymbol{M}(\boldsymbol{q}, \boldsymbol{z})\ddot{\boldsymbol{z}} + \boldsymbol{c}(\boldsymbol{q}, \boldsymbol{z}, \dot{\boldsymbol{q}}, \dot{\boldsymbol{z}}) + \boldsymbol{g}(\boldsymbol{q}, \boldsymbol{z}), \tag{3.25}$$

wobei

$$\boldsymbol{\tau}_z = \boldsymbol{\tau}_p + \sum_{j=1}^{N_s} \boldsymbol{J}_z^{q_j\mathrm{T}}\boldsymbol{\tau}_j, \tag{3.26}$$

$$\boldsymbol{M}(\boldsymbol{q}, \boldsymbol{z}) = \boldsymbol{M}_p(\boldsymbol{z}) + \sum_{j=1}^{N_s} \boldsymbol{J}_z^{q_j\mathrm{T}}\boldsymbol{M}_j(\boldsymbol{q}_j)\boldsymbol{J}_z^{q_j}, \tag{3.27}$$

$$\boldsymbol{c}(\boldsymbol{q}, \boldsymbol{z}, \dot{\boldsymbol{q}}, \dot{\boldsymbol{z}}) = \boldsymbol{c}_p(\boldsymbol{z}, \dot{\boldsymbol{z}}) + \sum_{j=1}^{N_s} (\boldsymbol{J}_z^{q_j\mathrm{T}}\boldsymbol{M}_j(\boldsymbol{q}_j)\boldsymbol{J}_z^{q_j}\dot{\boldsymbol{z}} + \boldsymbol{J}_z^{q_j\mathrm{T}}\boldsymbol{c}_j(\boldsymbol{q}_j, \dot{\boldsymbol{q}}_j)) \tag{3.28}$$

und

$$\boldsymbol{g}(\boldsymbol{q}, \boldsymbol{z}) = \boldsymbol{g}_p(\boldsymbol{z}) + \sum_{j=1}^{N_s} (\boldsymbol{J}_z^{q_j\mathrm{T}}\boldsymbol{g}_j(\boldsymbol{q}_j)). \tag{3.29}$$

In einem weiteren Schritt werden alle Gelenkgeschwindigkeiten durch die Minimalgeschwindigkeiten

$$\dot{\boldsymbol{q}}_{\mathrm{ges}} = (\dot{\boldsymbol{q}}_1^{\mathrm{T}}, \dot{\boldsymbol{q}}_2^{\mathrm{T}}, ..., \dot{\boldsymbol{z}}^{\mathrm{T}})^{\mathrm{T}} = (\boldsymbol{J}_z^{q_1\mathrm{T}}, \boldsymbol{J}_z^{q_2\mathrm{T}}, \boldsymbol{J}_z^{q_{Ns}\mathrm{T}}, ..., \boldsymbol{I})^{\mathrm{T}}\dot{\boldsymbol{z}} \tag{3.30}$$

ersetzt, sodass nun die Gleichung

$$\boldsymbol{\tau}_z = \boldsymbol{M}(\boldsymbol{q}_{\mathrm{ges}})\ddot{\boldsymbol{z}} + \boldsymbol{c}(\boldsymbol{q}_{\mathrm{ges}}, \dot{\boldsymbol{z}}) + \boldsymbol{g}(\boldsymbol{q}_{\mathrm{ges}}) \tag{3.31}$$

in Abhängigkeit aller Gelenkkoordinaten $\boldsymbol{q}_{\mathrm{ges}}$, aber nur der Minimalgeschwindigkeiten und -beschleunigunen $\dot{\boldsymbol{z}}$ bzw. $\ddot{\boldsymbol{z}}$ vorliegt.

#### Inverse Dynamik

Wie oben bereits erklärt, dient das Dynamikmodell als Vorsteuerung im Regelkreis des Roboters oder dazu, die Roboterbewegung zu simulieren. In beiden Fällen wird die inverse Dynamik benötigt, bei der das Modell als einzige Momente die aktiven Gelenkmomente enthält. Es wird

demnach der Zusammenhang zwischen den Kräften und Momenten im kartesischen Raum $\boldsymbol{\tau}_z$ und im aktiven Gelenkraum $\boldsymbol{\tau}_a$ gesucht. Dieser ergibt sich aber genau aus der Gesamt-Jacobi-Matrix $\boldsymbol{J}_z^{q_a}$ des parallelen Roboters, sodass $\boldsymbol{\tau}_\mathrm{a} = \boldsymbol{J}_z^{q_\mathrm{a}\,-\mathrm{T}}\boldsymbol{\tau}_z$. Die Gesamt-Jacobi-Matrix ergibt sich aus dem Zusammenhang $\dot{\boldsymbol{q}}_\mathrm{a} = \boldsymbol{J}_z^{q_\mathrm{a}}\dot{\boldsymbol{z}}$ und kann ähnlich wie in (3.12) bzw. (3.13) über die minimalen kinematischen Zwangsbedingungen hergeleitet werden:

$$\frac{\partial \boldsymbol{h}(\boldsymbol{q})}{\partial \boldsymbol{z}}\dot{\boldsymbol{z}} + \frac{\partial \boldsymbol{h}(\boldsymbol{q})}{\partial \boldsymbol{q}_a}\dot{\boldsymbol{q}}_a = \boldsymbol{A}_\mathrm{ges}\dot{\boldsymbol{z}} + \boldsymbol{B}_\mathrm{ges}\dot{\boldsymbol{q}}_a = 0 \tag{3.32}$$

$$\dot{\boldsymbol{q}}_\mathrm{a} = -\boldsymbol{B}_\mathrm{ges}^{-1}\boldsymbol{A}_\mathrm{ges}\dot{\boldsymbol{z}} = \boldsymbol{J}_z^{q_\mathrm{a}}\dot{\boldsymbol{z}} \tag{3.33}$$

Der Vollständigkeit halber sei erwähnt, dass die Gesamt-Jacobi-Matrix $\boldsymbol{J}_z^{q_\mathrm{a}}$ auch aus $(\boldsymbol{J}_z^{q_1\,\mathrm{T}}, \boldsymbol{J}_z^{q_2\,\mathrm{T}}, \boldsymbol{J}_z^{q_{Ns}\,\mathrm{T}}, ..., \boldsymbol{I})^\mathrm{T} = \boldsymbol{J}_z^{q_\mathrm{ges}}$ abgeleitet werden kann. In dieser Matrix sind die aktiven Gelenkanteile der Gesamt-Jacobi-Matrix bereits enthalten und müssen lediglich über die Auswahl-Matrix $\partial \boldsymbol{q}_\mathrm{ges}/\partial \boldsymbol{q}_\mathrm{a}$ extrahiert werden [DTKHO09a]: Aus

$$\boldsymbol{\tau}_\mathrm{a} = \boldsymbol{J}_z^{q_\mathrm{a}\,-\mathrm{T}}\boldsymbol{\tau}_z \quad \text{und} \quad \boldsymbol{\tau}_\mathrm{a} = \frac{\partial \boldsymbol{q}_\mathrm{ges}}{\partial \boldsymbol{q}_\mathrm{a}}\boldsymbol{\tau}_{q_\mathrm{ges}} = \frac{\partial \boldsymbol{q}_\mathrm{ges}}{\partial \boldsymbol{q}_\mathrm{a}}^\mathrm{T}\boldsymbol{J}_z^{q_\mathrm{ges}\,-\mathrm{T}}\boldsymbol{\tau}_z \tag{3.34}$$

ergibt sich

$$\boldsymbol{J}_z^{q_\mathrm{a}\,-\mathrm{T}}\boldsymbol{\tau}_z = \frac{\partial \boldsymbol{q}_\mathrm{ges}}{\partial \boldsymbol{q}_\mathrm{a}}^\mathrm{T}\boldsymbol{J}_z^{q_\mathrm{ges}\,-\mathrm{T}}\boldsymbol{\tau}_z. \tag{3.35}$$

Durch Multiplikation von (3.25) mit der transponierten Gesamt-Jacobi-Matrix $\boldsymbol{J}_z^{q_\mathrm{a}\,-\mathrm{T}}$ erhält man die gewünschte Form aus (3.11)

$$\begin{aligned}\boldsymbol{\tau}_\mathrm{a} &= \boldsymbol{J}_z^{q_\mathrm{a}\,-\mathrm{T}}\boldsymbol{\tau}_z \\ &= \boldsymbol{J}_z^{q_\mathrm{a}\,-\mathrm{T}}\left(\boldsymbol{M}(\boldsymbol{q}_\mathrm{ges})\ddot{\boldsymbol{z}} + \boldsymbol{c}(\boldsymbol{q}_\mathrm{ges}, \dot{\boldsymbol{z}}) + \boldsymbol{g}(\boldsymbol{q}_\mathrm{ges})\right) \\ &= \boldsymbol{M}_\mathrm{a}(\boldsymbol{q}_\mathrm{ges})\ddot{\boldsymbol{z}} + \boldsymbol{c}_\mathrm{a}(\boldsymbol{q}_\mathrm{ges}, \dot{\boldsymbol{z}}) + \boldsymbol{g}_\mathrm{a}(\boldsymbol{q}_\mathrm{ges}).\end{aligned} \tag{3.36}$$

## 3.3 Projektion über Zwischenkoordinatenraum (Abdellatif)

Der in dieser Arbeit zu implementierende Ansatz wurde 2007 am damaligen Institut für Robotik (heute Institut für Mechatronische Systeme) und am Mechatronik-Zentrum Hannover (MZH) der Leibniz Universität Hannover von Abdellatif et al. entwickelt [Abd07]. Es hat viele Ähnlichkeiten mit PARADYN [DTKHO09a], unterscheidet sich allerdings trotzdem in mehreren, wesentlichen Berechnungen. Dies führt zu einer veränderten Struktur des symbolischen Ergebnisses und somit auch zu Unterschieden in der Rechenzeit. Im Folgenden wird auf die Funktionsweise des Verfahrens eingegangen.

### 3.3.1 Form des Ergebnisses

Zu den Gemeinsamkeiten der beiden Ansätze gehört, dass die Dynamikmodelle des Gesamtsystems durch Aufspaltung in Subsysteme berechnet werden. Abschnitt 2.2.5 zeigte bereits, was der Vorteil einer solchen Methode ist und bewies durch Energiebetrachtungen, dass es zu den gleichen Dynamikgleichungen führt. Die Ergebnisse der Subsysteme werden ins kartesische Koordinatensystem transformiert und aufsummiert, um die Minimalform zu erhalten. Durch Multiplikation mit der transponierten Gesamt-Jacobi-Matrix $\boldsymbol{J}_s^{q_\mathrm{a}\,-\mathrm{T}}$ werden die Antriebsmomente berechnet. Unterschiede gibt es in der Berechnung der Gesamt-Jacobi-Matrix und der Jacobi-Matrix aus $\dot{\boldsymbol{q}}_j = \boldsymbol{J}_s^{q_j}\,\dot{\boldsymbol{s}}$, die von Minimalgeschwindigkeiten in Gelenkgeschwindigkeiten transformiert. Hier werden die Jacobi-Matrizen nicht anhand der kinematischen Zwangsbedingungen berechnet, wie das beispielsweise in (3.12) bzw. (3.13) geschehen ist, sondern über die seriellen Jacobi-Matrix der jeweiligen Beine und dem Zusammenhang zwischen den Koppelpunktgeschwindigkeiten und den Plattformgeschwindigkeiten der PKM. Die Gesamt-Jacobi-Matrix ergibt sich durch Auswahl derjenigen Zeilen der Jacobi-Matrizen $\boldsymbol{J}_s^{q_j}$ , die auf die aktiven Gelenkgeschwindigkeiten führen.

Eine weiterer Besonderheit bei dem Ansatz nach Abdellatif et al. ist die Verwendung von Quasikoordinaten $\boldsymbol{s}$ und Quasigeschwindigkeiten $\dot{\boldsymbol{s}} = (\dot{\boldsymbol{r}}_\mathrm{P}^\mathrm{T}, \boldsymbol{\omega}_\mathrm{P}^\mathrm{T})^\mathrm{T}$. Sie werden genutzt, weil die Berechnungen der Ableitungen in den LAGRANGE'schen Gleichungen nach den Geschwindigkeiten $\dot{\boldsymbol{z}} = (\dot{\boldsymbol{r}}_\mathrm{P}^\mathrm{T}, \dot{\boldsymbol{\varphi}}_\mathrm{P}^\mathrm{T})^\mathrm{T}$, die konsistent mit den Minimalkoordinaten sind, schnell sehr aufwendig werden. Da $\dot{\boldsymbol{r}}_P$ und $\boldsymbol{\omega}_\mathrm{P}$ direkt in der Berechnung der kinetische Energie der Plattform involviert sind, bietet es sich an, die Winkelgeschwindigkeiten $\boldsymbol{\omega}_\mathrm{P}$ anstatt der EULER-Winkeländerungsraten $\dot{\boldsymbol{\varphi}}_\mathrm{P}$ als Ableitung der Minimalkoordinaten zu verwenden [Qui90]. Bei der Darstellung der Orientierung durch Rotationen um die drei Raumachsen muss es eine Konvention geben, die besagt, um welche Achsen und in welcher Reihenfolge die Koordinatensysteme rotiert werden. Der Angabe der Rotationswinkel $\boldsymbol{\varphi}_\mathrm{P}$ muss demnach die genutzte Konvention beigefügt sein, um die Orientierung exakt beschreiben zu können. $\boldsymbol{\omega}_\mathrm{P}$ gibt die Drehrate um eine momentane Drehachse an, bei der die infinitesimale Bewegung wie eine Bewegung in der Ebene betrachtet werden kann. Die momentane Drehachse wird durch die in $\boldsymbol{\omega}_\mathrm{P}$ enthaltenen EULER-Winkeln beschrieben. Bis auf den Fall, dass Rotationen nur in einer Ebene ausgeführt werden, entsprechen die Komponenten der Winkelgeschwindigkeit $\boldsymbol{\omega}_\mathrm{P}$ im Allgemeinen allerdings nicht den EULER-Winkeländerungsraten $\dot{\boldsymbol{\varphi}}_\mathrm{P}$. Da $\boldsymbol{\omega}_\mathrm{P}$ nur für infenitesimale Bewegungen definiert ist, kann $\mathrm{d}\boldsymbol{\Omega}_\mathrm{P} = \boldsymbol{\omega}_\mathrm{P}\mathrm{d}t$ nur auf infinitesimale Drehungen um die aktuelle Achse führen. Das Integral $\int \boldsymbol{\omega}_\mathrm{P}\mathrm{d}t$ hat somit keine physikalische Bedeutung. Diese Eigenschaft hat dazu geführt dazu, dass Variablen, wie die Winkelgeschwindigkeiten $\boldsymbol{\omega}_\mathrm{P}$, als Quasi-Geschwindigkeiten und $\boldsymbol{\Omega}_\mathrm{P}$ als Quasi-Koordinaten bezeichnet werden. Der Zusatz „Quasi"bedeutet, dass die Koordinate $\boldsymbol{\Omega}_\mathrm{P}$ keine Bedeutung als Positionskoordinate besitzt, wo-

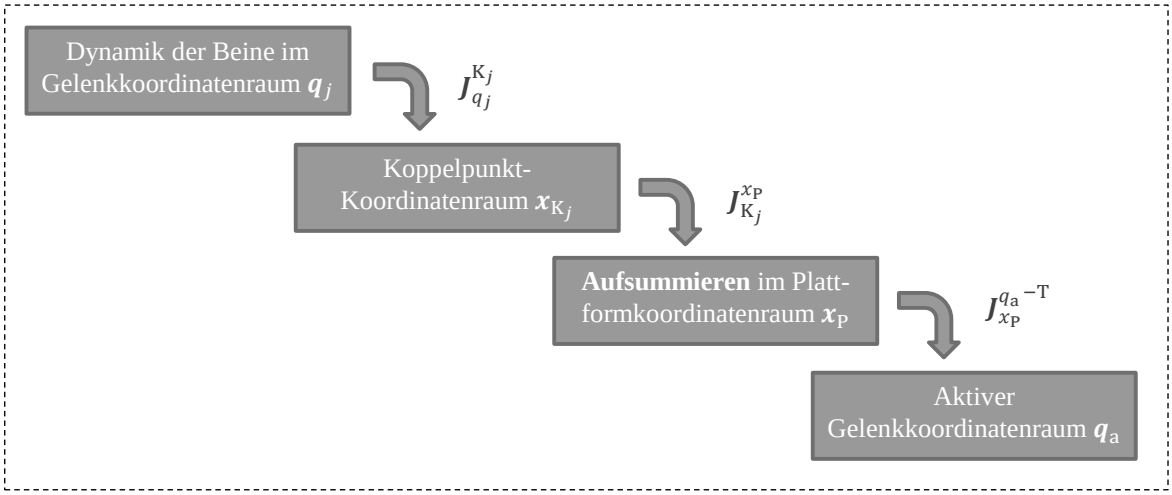



**Bild 3.3**: Schematische Darstellung nach dem Ansatz von Abdellatif et al.

hingegen die Geschwindigkeit $\boldsymbol{\omega}_\mathrm{P}$ und das differentielle Inkrement $\mathrm{d}\boldsymbol{\Omega}_\mathrm{P} = \boldsymbol{\omega}_\mathrm{P}\mathrm{d}t$ physikalisch gedeutet werden können [Gin08].

Für das Dynamikmodell ergibt sich mit Verwendung der Quasikoordinaten die von (3.11) verschiedene Gleichung

$$\boldsymbol{\tau}_\mathrm{a} = \boldsymbol{f}_\mathrm{dyn}(\ddot{\boldsymbol{s}}, \dot{\boldsymbol{s}}, \boldsymbol{q}_{ges}, \boldsymbol{p}) = \boldsymbol{M}_\mathrm{a}(\boldsymbol{q}_{ges})\ddot{\boldsymbol{s}} + \boldsymbol{c}_\mathrm{a}(\boldsymbol{q}_{ges}, \dot{\boldsymbol{s}}) + \boldsymbol{g}_\mathrm{a}(\boldsymbol{q}_{ges}). \tag{3.37}$$

$\dot{\boldsymbol{s}}$ bzw. $\ddot{\boldsymbol{s}}$ entsprechen den $(n \times 1)$ Minimalgeschwindigkeiten bzw. -beschleunigungen, die hier aus Quasigeschwindigkeiten und -beschleunigungen bestehen, und $\boldsymbol{\tau}_a$ dem Kraftvektor der Antriebe.

### 3.3.2 Das Verfahren

Wie oben bereits erklärt, sind sich das Verfahren von Abdellatif und PARADYN in den meisten Punkten sehr ähnlich. Übereinstimmungen gibt es in dem verwendeten Ansatz, in dem das Gesamtsystem der parallelen Struktur in deren Teilketten und die Plattform aufgeteilt werden, um die Gleichungen anschließend in einem gemeinsamen Koordinatenraum zu summieren. Die Unterschiede liegen vor allem in der Berechnung der Jacobi-Matrizen zur Projektion der Gleichungen und der Wahl der Minimalkoordinaten bzw. -geschwindigkeiten. In Abschnitt 3.3.1 wurde hierzu bereits auf die Verwendung von Quasigeschwindigkeiten eingegangen. Im Folgenden werden die Schritte zur Berechnung des Dynamikmodells ausführlich vorgestellt. Zunächst

wird die Kinematik, insbesondere die entsprechenden Jacobi-Matrizen zur Transformation der Gleichungen, hergeleitet. Um nicht die Zwangsbedingungen der geschlossenen Ketten bestimmen zu müssen, kombiniert Abdellatif die Kinematik der Beine und den Zusammenhang zwischen Koppelpunkt- und Plattformgeschwindigkeiten. Die Kinematik der Beine, die in den meisten Fällen aus seriellen Ketten bestehen, kann analytisch mit wenig Aufwand bestimmt werden. Die Jacobi-Matrix, die den Zusammenhang zwischen Koppelpunkt- und Plattformgeschwindigkeiten beschreibt, ist bei jedem parallelen Roboter gleich. Durch die Wahl dieser Transformationsvorschrift wird erhofft den Rechenaufwand weiter zu reduzieren [Abd07].

Zur Berechnung der Dynamikgleichungen $\boldsymbol{\tau}_j$ der seriellen Ketten werden die LAGRANGE'schen Gleichungen bezüglich der Gelenkkoordinaten $\boldsymbol{q}_j$ bestimmt. Da die Roboterplattform lediglich aus einem Starrkörper besteht, kann das Dynamikmodell ebenfalls mittels der LAGRANGE'schen Gleichungen oder mit Hilfe des NEWTON-EULER-Verfahrens hergeleitet werden. Bei allen Berechnungen ist darauf zu achten, dass als Minimalgeschwindigkeiten und -beschleunigungen die in Abschnitt 3.3.1 gewählten Quasigeschwindigkeiten und -beschleunigungen verwendet werden. Dies gilt es sowohl in der Plattformdynamik als auch bei der Transformation in den gemeinsamen Koordinatenraum zu beachten. Wie auch in dem Verfahren von PARADYN, müssen alle Gelenkgeschwindigkeiten und -beschleunigungen $\dot{\boldsymbol{q}}_j$ und $\ddot{\boldsymbol{q}}_j$ durch die Minimalgeschwindigkeiten und -beschleunigungen $\dot{\boldsymbol{s}}$ und $\ddot{\boldsymbol{s}}$ ersetzt werden, um die Form in (3.37) zu erhalten. Durch numerische Verfahren kann $\boldsymbol{q}_j$ aus der inversen Kinematik bestimmt werden, sodass es ausreicht die Minimalkoordinaten $\boldsymbol{z}$ vorzugeben. Da es das Ziel ist, das Modell in symbolischer Form zu generieren, müssen den Gleichungen alle Systemkoordinaten $\boldsymbol{q}_{ges}$ vorgegeben werden. Die numerischen Berechnungen müssen deshalb in einem vorherigen Schritt, beispielsweise in MATLAB erfolgen [AH09].

#### Kinematik

Sowohl beim Kinematik- als auch beim Dynamikmodell ist auf die Verwendung der Quasigeschwindigkeiten Rücksicht zu nehmen. Der Vorteil dieser Definition wurde in Abschnitt 3.3.1 erläutert. Abdellatif verwendet in seinem Verfahren die Minimalkoordinaten, -geschwindigkeiten und -beschleunigungen

$$\begin{aligned} \boldsymbol{z} &= \boldsymbol{x}_\mathrm{P} = (\boldsymbol{r}_\mathrm{P}^\mathrm{T}, \boldsymbol{\varphi}_\mathrm{P}^\mathrm{T})^\mathrm{T}, \\ \dot{\boldsymbol{s}} &= \boldsymbol{v}_\mathrm{P} = (\dot{\boldsymbol{r}}_\mathrm{P}^\mathrm{T}, \boldsymbol{\omega}_\mathrm{P}^\mathrm{T})^\mathrm{T}, \\ \ddot{\boldsymbol{s}} &= \dot{\boldsymbol{v}}_\mathrm{P} = (\ddot{\boldsymbol{r}}_\mathrm{P}^\mathrm{T}, \dot{\boldsymbol{\omega}}_\mathrm{P}^\mathrm{T})^\mathrm{T}, \end{aligned} \tag{3.38}$$

wobei $\boldsymbol{r}_\mathrm{P} = (x_\mathrm{P}, y_\mathrm{P}, z_\mathrm{P})^\mathrm{T}$ der Ortsvektor zum Plattformkoordinatensystems bzw. zu dem Punkt auf der Plattform, an dem ein möglicher Endeffektor montiert werden kann, ist. $\boldsymbol{\varphi}_\mathrm{P}^\mathrm{T} =$

$(\psi_\mathrm{P}, \theta_\mathrm{P}, \phi_\mathrm{P})^\mathrm{T}$ gibt die Orientierung des Plattformkoordinatensystems an. Bei Angabe der Rotationswinkel ist auf die gewählte EULER-Winkel-Konvention zu achten. In der Herleitung der Gleichungen wird sich auf die am meisten genutzte KARDAN-Winkel-Konvention beschränkt, bei der nacheinander um die mitgedrehte $x$-, $y$- und $z$-Achse rotiert wird. Die Verfahren funktionieren aber genauso mit jeder anderen, denkbaren Winkel-Konvention [Abd07].

$\dot{\boldsymbol{r}}_\mathrm{P} = \mathrm{d}\boldsymbol{r}_\mathrm{P}/\mathrm{d}t$ ist die zeitliche Ableitung des Ortsvektors $\boldsymbol{r}_\mathrm{P}$ und gibt demnach den translatorischen Anteil an der Geschwindigkeit $\boldsymbol{v}_\mathrm{P}$ an. Die Winkelgeschwindigkeit der Plattform wird durch $\boldsymbol{\omega}_\mathrm{P}$ ausgedrückt und kann über die Rotationsmatrizen der Plattform $\boldsymbol{R}_\mathrm{P}$ zu

$$\boldsymbol{\omega}_\mathrm{P} = \dot{\boldsymbol{R}}_\mathrm{P}\boldsymbol{R}_\mathrm{P}^\mathrm{T} \tag{3.39}$$

berechnet werden. Für die KARDAN-Winkel-Konvention ergibt sich die Winkelgeschwindigkeit in Abhängigkeit der Winkelraten zu

$$\begin{aligned}\boldsymbol{\omega}_\mathrm{P} &= \frac{\partial \boldsymbol{\omega}_\mathrm{P}}{\partial \dot{\boldsymbol{\varphi}}_\mathrm{P}} \dot{\boldsymbol{\varphi}}_\mathrm{P} \\ &= \begin{pmatrix} 1 & 0 & \mathrm{s}_{\theta_P} \\ 0 & \mathrm{c}_{\psi_\mathrm{P}} & -\mathrm{s}_{\psi_\mathrm{P}}\mathrm{c}_{\theta_\mathrm{P}} \\ 0 & \mathrm{s}_{\psi_\mathrm{P}} & \mathrm{c}_{\psi_\mathrm{P}}\mathrm{c}_{\theta_\mathrm{P}} \end{pmatrix} \begin{pmatrix} \dot{\psi}_\mathrm{P} \\ \dot{\theta}_\mathrm{P} \\ \dot{\phi}_\mathrm{P} \end{pmatrix} \\ &= \boldsymbol{Q}_\mathrm{p}(\boldsymbol{\varphi}_\mathrm{P})\dot{\boldsymbol{\varphi}}_\mathrm{P}.\end{aligned} \tag{3.40}$$

Die Abkürzungen $\mathrm{s}_\psi$ und $\mathrm{c}_\psi$ stehen dabei für die trigonometrischen Funktionen $\sin(\psi)$ und $\cos(\psi)$ [AH09]. Mit (3.38) kann somit der Zusammenhang zwischen den Quasigeschwindigkeiten $\dot{\boldsymbol{s}}$ und den Ableitungen der Plattformkoordinaten $\dot{\boldsymbol{x}}_\mathrm{P}$ mit

$$\begin{aligned}\dot{\boldsymbol{s}} = \boldsymbol{v}_\mathrm{P} = \begin{pmatrix} \dot{\boldsymbol{r}}_\mathrm{P} \\ \boldsymbol{\omega}_\mathrm{P} \end{pmatrix} &= \begin{pmatrix} \boldsymbol{I} & 0 \\ 0 & \boldsymbol{Q}_\mathrm{p}(\boldsymbol{\varphi}_\mathrm{P}) \end{pmatrix} \dot{\boldsymbol{x}}_\mathrm{P} \\ &= \boldsymbol{H}(\boldsymbol{\varphi}_\mathrm{P})\dot{\boldsymbol{x}}_\mathrm{P} \\ &= \boldsymbol{H}(\boldsymbol{\varphi}_\mathrm{P})\dot{\boldsymbol{z}}\end{aligned} \tag{3.41}$$

beschrieben werden. Damit existiert ein linearer Zusammenhang zwischen den Minimal- bzw. Quasigeschwindigkeiten und den Ableitungen der Minimalkoordinaten.

In Abschnitt 2.2.5 wurde bereits erklärt, dass die Dynamikmodelle der einzelnen Subsysteme in ein gemeinsames Koordinatensystem transformiert werden müssen, um die Gleichungen addieren zu können. Aus Bild 3.3 wird deutlich, dass die Gleichungen zunächst in den Raum der Koppelpunkte $\boldsymbol{x}_{\mathrm{K}_j}$ und anschließend in den Raum der Minimalkoordinaten $\boldsymbol{z}$ transformiert werden. Ein Vorteil bei der Verwendung dieser Zwischentransformation liegt in der einfachen

Berechnung der jeweiligen Jacobi-Matrizen, bei der die kinematischen Zwangsbedingungen nicht aufgestellt müssen.

Der Zusammenhang $\dot{\boldsymbol{q}}_j = \boldsymbol{J}_s^{q_j}\dot{\boldsymbol{s}}$ wird deshalb über die Transformationen mit einer zusätzliche Matrix-Multiplikationen

$$\dot{\boldsymbol{q}}_j = \boldsymbol{J}_{\mathrm{K}_j}^{q_j}\dot{\boldsymbol{x}}_{\mathrm{K}_j} = \boldsymbol{J}_{\mathrm{K}_j}^{q_j}\boldsymbol{J}_s^{\mathrm{K}_j}\dot{\boldsymbol{s}} \tag{3.42}$$

ausgedrückt. $\dot{\boldsymbol{q}}_j = \boldsymbol{J}_{\mathrm{K}_j}^{q_j}\dot{\boldsymbol{x}}_{\mathrm{K}_j} = \boldsymbol{J}_{q_j}^{\mathrm{K}_j\,-1}\dot{\boldsymbol{x}}_{\mathrm{K}_j}$ entspricht der direkten differentiellen Kinematik der Beine und kann aus der Ableitung

$$\dot{\boldsymbol{x}}_{\mathrm{K}_j} = \frac{\mathrm{d}\boldsymbol{x}_{\mathrm{K}_j}}{\mathrm{d}t} = \frac{\partial \boldsymbol{f}(q_j)}{\partial \boldsymbol{q}_j}\frac{\mathrm{d}\boldsymbol{q}_j}{\mathrm{d}t} = \boldsymbol{J}_{q_j}^{\mathrm{K}_j}\dot{\boldsymbol{q}}_j \tag{3.43}$$

der direkten Kinematik $\boldsymbol{x}_{\mathrm{K}_j} = \boldsymbol{f}(q_j)$ bestimmt werden. Aufgrund der seriellen Struktur der Beine gestaltet sich die Berechnung recht einfach. Im Gegensatz zu der Berechnung von $\boldsymbol{J}_{\mathrm{K}_j}^{q_j}$ ist die Jacobi-Matrix aus $\dot{\boldsymbol{x}}_{\mathrm{K}_j} = \boldsymbol{J}_s^{\mathrm{K}_j}\dot{\boldsymbol{s}}$ nicht abhängig von der Roboterstruktur. Da beide Punkte $\boldsymbol{z}$ und $\boldsymbol{x}_{\mathrm{K}_j}$ auf der Plattform, die als Starrkörper betrachtet wird, liegen, handelt es sich lediglich um eine translatorische Transformation

$$\begin{aligned}\dot{\boldsymbol{x}}_{\mathrm{K}_j} = \dot{\boldsymbol{r}}_{\mathrm{K}_j} &= \dot{\boldsymbol{r}}_{\mathrm{P}} + \boldsymbol{r}_{\mathrm{P}}^{\mathrm{K}_j} \times \boldsymbol{\omega}_{\mathrm{P}} \\ &= \dot{\boldsymbol{r}}_{\mathrm{P}} + \tilde{\boldsymbol{r}}_{\mathrm{P}}^{\mathrm{K}_j}\boldsymbol{\omega}_{\mathrm{P}} \\ &= (\boldsymbol{I}, \tilde{\boldsymbol{r}}_{\mathrm{P}}^{\mathrm{K}_j})(\dot{\boldsymbol{r}}_{\mathrm{P}}^{\mathrm{T}}, \boldsymbol{\omega}_{\mathrm{P}}^{\mathrm{T}})^{\mathrm{T}} \\ &= \boldsymbol{J}_s^{\mathrm{K}_j}\dot{\boldsymbol{s}}.\end{aligned} \tag{3.44}$$

Die Inverse der Gesamt-Jacobi-Matrix ergibt sich aus den Zeilenvektoren

$$\boldsymbol{J}_s^{q_{\mathrm{a}_j}} = \frac{\partial q_{\mathrm{a}_j}}{\partial \boldsymbol{q}_j}\boldsymbol{J}_s^{q_j} = \frac{\partial q_{\mathrm{a}_j}}{\partial \boldsymbol{q}_j}\boldsymbol{J}_{\mathrm{K}_j}^{q_j}\boldsymbol{J}_s^{\mathrm{K}_j} \tag{3.45}$$

zu

$$\boldsymbol{J}_s^{q_{\mathrm{a}}} = (\boldsymbol{J}_s^{q_{\mathrm{a}_1}\,\mathrm{T}}, \boldsymbol{J}_s^{q_{\mathrm{a}_2}\,\mathrm{T}}, ..., \boldsymbol{J}_s^{q_{\mathrm{a}_{N_s}}\,\mathrm{T}})^{\mathrm{T}}, \tag{3.46}$$

wobei $\partial q_{\mathrm{a}_j}/\partial \boldsymbol{q}_j$ die Zeilen von $\boldsymbol{J}_s^{q_j}$ auswählt, die auf $q_{\mathrm{a}_j}$ führen, wenn damit das aktive Gelenk des Beines $j$ [Abd07] bezeichnet wird. Damit sind alle nötigen Transformation der Kinematik für parallele Roboter bekannt und das Dynamikmodell, das im folgenden beschrieben wird, kann vollständig aufgestellt werden.

### Dynamik der Plattform

Da die Berechnung der Dynamik der Beine dem gleichen Prinzip wie dem in Abschnitt 3.2.2 folgt, wird nur noch auf die Plattformgleichungen eingegangen. Auf Grund der Verwendung von Quasikoordinaten und der Betrachtung der Plattform als Starrkörper lassen sich die Energien $T$ und $U$ aus (3.22) nach dem Verfahren von LAGRANGEanalytisch ableiten. Dies gehört zu den in Abschnitt 3.3.1 genannten Vorteilen der Wahl von Quasikoordinaten. Nach dem Ableiten und Zusammenführen in die jeweiligen Dynamikterme erhält man

$$\boldsymbol{\tau}_{\mathrm{P}} = \boldsymbol{M}_{\mathrm{P}}(\boldsymbol{z})\ddot{\boldsymbol{s}} + \boldsymbol{c}_{\mathrm{P}}(\boldsymbol{z}, \dot{\boldsymbol{s}}) + \boldsymbol{g}_{\mathrm{P}}\boldsymbol{z} \tag{3.47}$$

als strukturierte und allgemeine Lösung. Die jeweiligen Dynamikterme werden hier nicht aufgelistet und sind [Abd07] zu entnehmen.

### Projektion und Summieren der Gleichungen

Nachdem nun alle Dynamikgleichungen vorliegen, gilt es diese zusammenzuführen. In [AH09] erfolgt dieser Schritt mittels Projektion in einen gemeinsamen Koordinatenraum analog zu dem in Abschnitt 3.2.2 beschriebenen Verfahren. Hierzu werden die Dynamikgleichungen der seriellen Kette mit Hilfe die Projektionsmatrix $\boldsymbol{J}_s^{q_j} = \boldsymbol{J}_{\mathrm{K}_j}^{q_j}\boldsymbol{J}_s^{\mathrm{K}_j}$ in den Raum des Inertialkoordinatensystem projiziert. Anschließend können alle Gleichungen, die nun bezüglich des selben Raumes vorliegen, aufsummiert werden. Als Ergebnis erhält man die Minimalform der Dynamikgleichungen

$$\boldsymbol{\tau}_s = \boldsymbol{M}(\boldsymbol{q}, \boldsymbol{z})\ddot{\boldsymbol{s}} + \boldsymbol{c}(\boldsymbol{q}, \boldsymbol{z}, \dot{\boldsymbol{q}}, \dot{\boldsymbol{s}}) + \boldsymbol{g}(\boldsymbol{q}, \boldsymbol{z}), \tag{3.48}$$

wobei

$$\boldsymbol{\tau}_s = \boldsymbol{\tau}_{\mathrm{P}} + \sum_{j=1}^{N_s} \boldsymbol{J}_s^{q_j\,\mathrm{T}}\boldsymbol{\tau}_j, \tag{3.49}$$

$$\boldsymbol{M}(\boldsymbol{q}, \boldsymbol{z}) = \boldsymbol{M}_{\mathrm{P}}(\boldsymbol{z}) + \sum_{j=1}^{N_s} \boldsymbol{J}_s^{q_j\,\mathrm{T}}\boldsymbol{M}_j(\boldsymbol{q}_j)\boldsymbol{J}_s^{q_j}, \tag{3.50}$$

$$\boldsymbol{c}(\boldsymbol{q}, \boldsymbol{z}, \dot{\boldsymbol{q}}, \dot{\boldsymbol{s}}) = \boldsymbol{c}_{\mathrm{P}}(\boldsymbol{z}, \dot{\boldsymbol{s}}) + \sum_{j=1}^{N_s} (\boldsymbol{J}_s^{q_j\,\mathrm{T}}\boldsymbol{M}_j(\boldsymbol{q}_j)\dot{\boldsymbol{J}}_s^{q_j}\dot{\boldsymbol{s}} + \boldsymbol{J}_s^{q_j\,\mathrm{T}}\boldsymbol{c}_j(\boldsymbol{q}_j, \dot{\boldsymbol{q}}_j)) \tag{3.51}$$

und

$$\boldsymbol{g}(\boldsymbol{q},\boldsymbol{z}) = \boldsymbol{g}_p(\boldsymbol{z}) + \sum_{j=1}^{N_s} (\boldsymbol{J}_s^{q_j\,\mathrm{T}} \boldsymbol{g}_j(\boldsymbol{q}_j)). \tag{3.52}$$

Die Berechnung der Jacobi-Matrix $\dot{\boldsymbol{J}}_s^{q_j}$ ist sehr viel aufwendiger als die dessen inverse Jacobi-Matrix $\dot{\boldsymbol{J}}_{q_j}^{s} = \dot{\boldsymbol{J}}_s^{q_j\,-1}$. Der zweite Term aus (3.51) wird demnach so aufgestellt, dass nur noch die Ableitung dieser nicht-inversen Jacobi-Matrix bestimmt werden muss. Eine detaillierte Beschreibung dieser Umformschritte ist in [Abd07] zu finden. In dieser Substitution der Ableitungen und der damit einhergehenden Steigerung der Recheneffizienz zeigt sich der Vorteil einer Aufteilung der Projektionen in zwei Teilschritte aus (3.42). In einem letzten Schritt werden alle Gelenkgeschwindigkeiten $\dot{\boldsymbol{q}}_j$ durch die Minimalgeschwindigkeiten $\dot{\boldsymbol{s}}$ ersetzt. Damit erhält man die geforderderte Funktion aus (3.37), die nur noch von den Systemkoordinaten und den Minimalgeschwindigkeiten und -beschleunigungen abhängig ist.

### Inverse Dynamik

Die vollständige inverse Dynamik liegt nun bezüglich des Raumes der Plattformkoordinaten vor. Das Ziel ist es allerdings die Funktion der Form (3.37) zu erhalten, in der explizit die aktiven Gelenkmomente $\boldsymbol{\tau}_\mathrm{a}$ enthalten sind. Es muss daher ein Zusammenhang zwischen den angetriebenen Gelenkmomenten bzw. -kräften $\boldsymbol{\tau}_\mathrm{a}$ und den Plattformmomenten bzw. -kräften $\boldsymbol{\tau}_\mathrm{s}$ hergestellt werden. Diese Abhängigkeit kann aber genau über die Gesamt-Jacobi-Matrix $\boldsymbol{\tau}_\mathrm{a} = \boldsymbol{J}_s^{q_a\,-\mathrm{T}} \boldsymbol{\tau}_\mathrm{s}$ ausgedrückt werden. Die Matrix $\boldsymbol{J}_s^{q_a}$ ergibt sich in [Abd07] aus der Matrix $(\boldsymbol{J}_s^{q_1\,\mathrm{T}}, \boldsymbol{J}_s^{q_2\,\mathrm{T}}, ..., \boldsymbol{J}_s^{q_{N_s}\,\mathrm{T}})^\mathrm{T} = \boldsymbol{J}_s^q$, indem sie mit der Auswahlmatrix $\partial \boldsymbol{q} / \partial \boldsymbol{q}_a$ multipliziert wird. Es werden also nur die Einträge aller Transformationen $\boldsymbol{J}_s^q$ ausgewählt, die die aktiven Gelenke betreffen. Man erhält somit für die Gesamt-Jacobi-Matrix

$$\boldsymbol{\tau}_\mathrm{a} = \boldsymbol{J}_s^{q_a\,-\mathrm{T}} \boldsymbol{\tau}_s = \frac{\partial \boldsymbol{q}}{\partial \boldsymbol{q}_a}^\mathrm{T} \boldsymbol{J}_s^{q\,-\mathrm{T}} \boldsymbol{\tau}_s. \tag{3.53}$$

Setzt man (3.48) für $\boldsymbol{\tau}_s$ in (3.53) ein, erhält man

$$\boldsymbol{\tau}_\mathrm{a} = \boldsymbol{J}_s^{q_a\,-\mathrm{T}} \boldsymbol{\tau}_s = \boldsymbol{J}_s^{q_a\,-\mathrm{T}} \left( \boldsymbol{M}(\boldsymbol{q}_{ges}) \ddot{\boldsymbol{s}} + \boldsymbol{c}(\boldsymbol{q}_{ges}, \dot{\boldsymbol{s}}) + \boldsymbol{g}(\boldsymbol{q}_{ges}) \right) \tag{3.54}$$

und somit die geforderte Funktion aus (3.37). Aufgrund der rechenintensiven Berechnung von $\boldsymbol{J}_s^{q_a\,-\mathrm{T}}$, wird die Invertierung insbesondere bei Robotern mit sechs Freiheitsgraden numerisch durchgeführt.

# 4 Die Toolbox

Das Ziel dieser Arbeit ist die Implementierung des Ansatzes (nach Abdellatif et al.), der in [AH09] beschrieben wird. Hierzu wurde eine bereits vorhandene Toolbox, die bisher die Kinematik sowie die Dynamik für serielle und hybride Roboter berechnen konnte, erweitert. Sie basiert auf mehreren MAPLE-Arbeitsblättern, die sowohl manuell als auch automatisch über Bash-Skripte nacheinander ausgeführt werden können.

## 4.1 Maple und Bash-Skripte

Mit dem Programm MAPLE können symbolische Berechnung durchgeführt werden, indem alle Funktionen, die man vorgibt, analytisch berechnet werden. Hierbei rechnet das Programm bekannte Ausdrücke automatisch aus und versucht diese, durch Vorgabe von entsprechenden Funktionen, zu vereinfachen. Die symbolische Berechnungsmethode hat den erheblichen Vorteil gegenüber der numerischen, dass viele Terme eliminiert werden, sobald sie Null ergeben. Dies trifft auch auf trigonometrischen Funktionen, wie z. B. $\cos(\pi/2) = 0$ zu. Dadurch fallen die überflüssigen Berechnungsschritte weg, was zu einer effizienteren Lösung und somit zu einer geringeren Rechenzeit führt. Bei numerischen Verfahren hingegen, wird jede Operation, die vorgegeben wird, ausgeführt. Nullen sind vor der Laufzeit nicht bekannt, sodass nicht darauf geachtet wird, ob eine Berechnung überflüssig ist, wenn es sich bei einem der Faktoren um eine Null handelt.

Die Verwendung von Bash-Skripten hat mehrere Vorteile. Auf der einen Seite werden damit die Lösungen vollkommen automatisch erstellt, auf der anderen Seite können die Arbeitsblätter, im Gegensatz zu einer reinen MAPLE-Implementierung, auch komplett manuell ausgeführt werden. Darüber hinaus können durch die Aufteilung in mehrere Arbeitsblätter im Vergleich zu den verschachtelten Funktion, wie in PARADYN, die Zwischenergebnisse leichter geprüft werden. Ein weiterer Vorteil ist, dass die Ergebnisse mit den Informationen zu den Schnittstellen automatisch generiert und standardisiert getestet werden können. Im Folgenden Abschnitt wird der Aufbau der Toolbox näher erläutert. Da hier zum Teil auf bereits vorhandene Ergebnisse zurückgegriffen wird, kann mittels Skripten überprüft werden, ob diese Ergebnisse schon existieren und diese bei Bedarf neu berechnen. Zum Testen und zur Weiterverwendung des erstellten Dynamikmodells können am Ende automatisch MATLAB-Funktionen generiert werden.

## 4.2 Aufbau

Aufbauend auf dem Dynamikmodell der Beine, das mit der Toolbox bereits bestimmt werden kann, besteht die erweiterte Struktur aus den vier Teilkomponenten, die die Aufgaben

- Erstellung der Variablen-Definitionen
- Berechnung der Kinematik des parallelen Roboters
- Berechnung der Plattformdynamik
- Projektion und Aufsummieren aller Dynamikgleichungen

ausführen. Für jede dieser Komponenten wurden MAPLE-Arbeitsblätter erstellt, die nacheinander ausgeführt werden müssen. Bild 4.1 gibt einen Überblick über das Schema der um die parallele Berechnung erweiterten Toolbox. Ausgehend von den bereits vorhanden Berechnungen der Beine und den in Abschnitt 3.3.2 aufgestellten Dynamikgleichungen der Plattform, werden diese in einem nächsten Schritt in den gleichen Koordinatenraum projiziert. Dies geschieht über die zuvor berechnete Kinematik aus Abschnitt 3.3.2. Liegen alle Differentialgleichungen im selben Koordinatenraum vor, können diese in einem letzten Schritt wie in Abschnitt 3.3.2 aufaddiert werden. Bei allen Berechnungen wird auf die vorhergegangenen Variablen-Definitionen zurückgegriffen.

Die Toolbox ermöglicht es außerdem die Dynamikgleichungen in parameterlinearer Form aufzustellen. Parameterlinearer Form bedeutet, dass das Dynamikmodell linear bezüglich der Dynamik-Parameter vorliegt [KD02]. Um die Berechnungen von der normalen Dynamikberechnung zu trennen, wurden hierzu zwei weitere Arbeitsblätter erstellt. Bei Verwendung der Berechnung in parameterlinearer Form werden demnach die Teilfunktionen

- Erstellung der Variablen-Definitionen
- Berechnung der Kinematik des parallelen Roboters
- Berechnung der Plattformdynamik in Parameterlinearer Form
- Projektion und Aufsummieren aller Dynamikgleichungen in parameterlinearer Form

nacheinander ausgeführt.

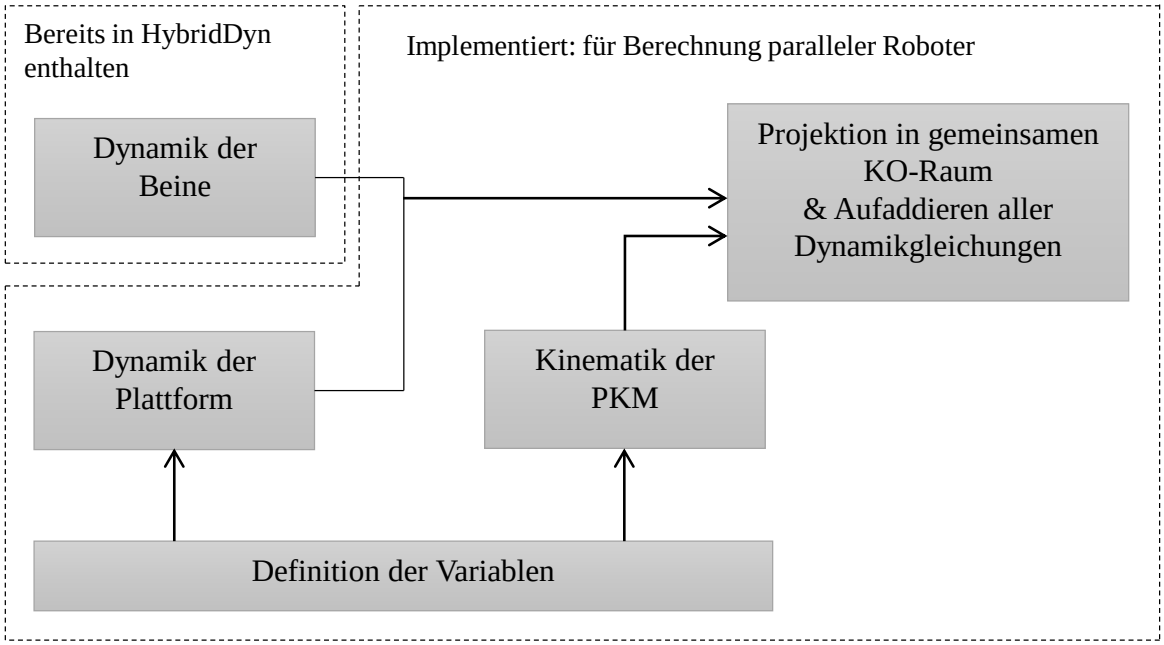



**Bild 4.1**: Aufbau der um die parallele Berechnung erweiterten Toolbox

## 4.3 Definitionen und Übergabeargumente

Die Berechnung baut auf den Informationen aus zwei Definitionsdateien auf, die für jeden Roboter erstellt werden müssen, bevor die Berechnungen ausgeführt werden können. Die eine Datei enthält alle Informationen für die Beine des Roboters, während die zweite die Struktur des parallelen Systems beschreibt. Indem die Berechnungen für die jeweiligen Beine und für die PKM getrennt voneinander berechnet werden, können die Lösungen auch vollkommen getrennt voneinander erstellt werden. Diese modulare Struktur der Toolbox hat den Vorteil, auf bereits vorhandene Lösungen zurückgreifen zu können ohne alle Gleichungen wiederholt berechnen zu müssen, und dadurch Rechenzeit einzusparen.

Die Definitionsdatei der Beine wurde aus der vorhandenen Toolbox übernommen. Sie enthält hauptsächlich den Namen des Roboters, die Anzahl der Teilkörper und Gelenke, sowie die MDH-Parameter für serielle oder hybride Roboter. Alle anderen Informationen werden entweder über Bash-Argumente übergeben oder, wie die physikalischen Parameter, als allgemein angenommen. Für die Berechnung paralleler Roboter wurde eine weitere Definitionsdatei eingeführt. Sie muss sowohl den Namen der PKM als auch die Namen der Beine enthalten, auf dessen Lösung zurückgegriffen werden soll. Aus diesem Grund muss bei der Definition auf

eine konsistente Namensgebung geachtet werden. Zudem enthält die Datei die Freiheitsgrade des Endeffektors und die Anzahl der Beine. Bei der Berechnung des Beines wurde bisher davon ausgegangen, dass alle Gelenke aktiv betrieben werden. Bei parallelen Robotern existiert für gewöhnlich aber nur ein Antriebsgelenk pro Bein, dessen Index angeben werden muss. Durch die Angabe des Basis-Koppelpunkts durch einen Ortsvektor und EULER-Winkel wird die Definition abgeschlossen.

Durch Übergabeargumente an das Bash-Skript ist es möglich zu beeinflussen, welche Berechnungen ausgeführt werden sollen. Für parallele Systeme muss übergeben werden, dass es sich um einen parallelen Roboter handelt. Zudem kann vorgegeben werden, ob das Dynamikmodell der Beine neu berechnet oder auf ein bereits vorhandenes Ergebnis zurückgegriffen werden soll. Auch hierbei muss auf die kontinuierliche Namensgebung geachtet werden. MAPLE greift bei der Berechnung standardmäßig nur auf einen Prozessorkern zu. Um Rechenzeit einzusparen, kann angegeben werden, dass die Arbeitsblätter, soweit möglich, parallel und auf mehreren Kernen ausgeführt werden.

## 4.4 Ausgabe

Die berechneten Ausdrücke können direkt in MAPLE als optimierten MATLAB-Code ausgegeben werden. Dafür wird die Gesamtfunktion möglichst weit vereinfacht und dann so in Teilausdrücke aufgeteilt, dass kein Ausdruck doppelt berechnet wird. Dadurch kann mehrfach auf ein Ergebnis zurückgegriffen werden, ohne es erneut berechnen zu müssen. Um die Ergebnisse ohne manuelles Eingreifen direkt in numerisch-rechnenden Programmen weiterverwenden zu können, werden aus den erstellten Code-Dateien MATLAB-Funktionen generiert. Hierbei werden alle zusammengehörige Variablen aus MAPLE durch MATLAB-Vektoren ersetzt. So können die Übergabeargumente an die Funktion möglichst übersichtlich gehalten werden.

# 5 Auswertung

Im Folgenden soll der implementierte Ansatz genauer untersucht werden. Hierbei werden in Abschnitt 5.1 zunächst klassische Roboterstrukturen vorgestellt, anhand derer die Untersuchungen erfolgen. Anschließend wird die Implementierung in der Korrektheit der Lösung in Abschnitt 5.2, der Struktur der Lösung in Abschnitt 5.3, der Recheneffizienz in Abschnitt 5.4 und den Limitierungen in Abschnitt 5.5 erforscht.

## 5.1 Verwendete Roboterstrukturen

In diesem Kapitel werden die Roboter, die zum Testen und Vergleichen der Implementierungen betrachtet werden, vorgestellt. In erster Linie wird auf die jeweilige Kinematik und gegebenenfalls eingeführte Vereinfachungen eingegangen. Um ein möglichst weites Spektrum abzudecken, werden Roboter mit verschiedenen Freiheitsgraden betrachtet. Zunächst werden rein planare Roboter in Abschnitt 5.1.1 und Abschnitt 5.1.2 beschrieben, um mit dem Delta-Roboter in Abschnitt 5.1.3 ein räumliches System mit drei Freiheitsgraden einzuführen. In Abschnitt 5.1.4 wird der Aufbau eines räumlichen Roboters mit einem rotatorischen Freiheitsgrad erläutert. Abschnitt 5.1.5 widmet sich der 6UPS-PKM, einem räumlichen Roboter mit vollen sechs Freiheitsgraden. Es konnte kein symmetrischer Roboter mit drei translatorischen und zwei rotatorischen FHG (3T2R) gefunden werden, dessen Modellierung mit dem implementierten Verfahren möglich ist. Auf die Erklärung hierzu wird in Abschnitt 5.5 eingegangen.

### 5.1.1 3RRR

Der 3RRR ist ein planarer Roboter mit insgesamt drei Freiheitsgraden. Aus zwei translatorischen und einem rotatorischen Freiheitsgrad ergibt sich die Kurzbeschreibung 2T1R. Als Minimalkoordinaten wird der Vektor $\boldsymbol{z} = (x_\mathrm{P}, y_\mathrm{P}, \varphi_\mathrm{P})^\mathrm{T}$ gewählt. Die Beine bestehen aus zwei Teilkörpern mit jeweils einem Drehgelenk, die wiederum über ein Drehgelenk als Koppelgelenk mit der Plattform verbunden sind. Aus Tabelle 5.1 können die verwendeten klassischen sowie die modifizierten DENAVIT-HARTENBERG-Parameter nach [BK15] entnommen werden. Das erste Gelenk jedes Beines ist das Antriebsgelenk $q_1$.

**Tabelle 5.1**: DH-Parameter der RRR-Beine des 3RRR-Roboters. Links: Standard DH-Parameter. Rechts: Modifizierte DH-Parameter

| $i$ | $\theta_i$ | $d_i$ | $a_i$ | $\alpha_i$ |
|---|---|---|---|---|
| 1 | $q_1$ | 0 | $l_1$ | 0 |
| 2 | $q_2$ | 0 | $l_2$ | 0 |
| 3 | $q_3$ | 0 | 0 | 0 |

| $i$ | $a(i)$ | $\sigma_i$ | $\gamma_i$ | $b_i$ | $\alpha_i$ | $d_i$ | $\theta_i$ | $r_i$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | $q_1$ | 0 |
| 2 | 1 | 0 | 0 | 0 | 0 | $l_1$ | $q_2$ | 0 |
| 3 | 2 | 0 | 0 | 0 | 0 | $l_2$ | $q_3$ | 0 |

### 5.1.2 3RPR

Eine weitere, oft betrachteter planare PKM ist der 3RPR-Roboter, der ebenfalls drei Freiheitsgrade besitzt, sodass die Minimalkoordinaten aus $\boldsymbol{z} = (x_\mathrm{P}, y_\mathrm{P}, \varphi_\mathrm{P})^\mathrm{T}$ (2T1R) bestehen. Die Beine bestehen aus zwei Teilkörpern mit einem Drehgelenk $q_1$ und dem aktiven Schubgelenk $q_2$ und werden über das passive Schnittgelenk $q_3$ mit der Plattform verbunden. Die klassischen sowie die modifizierten DENAVIT-HARTENBERG-Parameter nach [BK15] sind in Tabelle 5.2 zu finden.

**Tabelle 5.2**: DH-Parameter der PPR-Beine des 3RPR-Roboters. Links: Standard DH-Parameter. Rechts: Modifizierte DH-Parameter

| $i$ | $\theta_i$ | $d_i$ | $a_i$ | $\alpha_i$ |
|---|---|---|---|---|
| 1 | $q_1$ | 0 | 0 | $\pi/2$ |
| 2 | 0 | 0 | $q_2$ | $\pi/2$ |
| 3 | $q_3$ | 0 | 0 | 0 |

| $i$ | $a(i)$ | $\sigma_i$ | $\gamma_i$ | $b_i$ | $\alpha_i$ | $d_i$ | $\theta_i$ | $r_i$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | $q_1$ | 0 |
| 2 | 1 | 1 | 0 | 0 | $\pi/2$ | 0 | 0 | $q_2$ |
| 3 | 2 | 0 | 0 | 0 | $\pi/2$ | 0 | $q_3$ | 0 |

### 5.1.3 3RUU (Delta)

Der Delta-Roboter ist eine PKM, die räumliche Bewegungen mit drei translatorischen Freiheitsgraden (3T0R) durchführen kann, sodass als Minimalkoordinaten der Vektor $\boldsymbol{z} = (x_\mathrm{P}, y_\mathrm{P}, z_\mathrm{P})^\mathrm{T}$ gewählt wird. Die Beine bestehen aus dem aktiven Drehgelenk $q_1$, gefolgt von einer hybriden Struktur, die sich in ihrer Bewegung wie ein Parallelogramm verhält. Um die kompliziertere hybride Kinematik zu umgehen und die Beine als serielle Ketten betrachten zu können, wird eine Modellierung nach [LCGB06] gewählt. Hier wird davon ausgegangen, dass die Plattform immer parallel zu Inertialkoordinatensystem bewegt wird und rein translatorische Bewegungen ausführt ohne zu rotieren. Damit kann das hybride System durch einen Starrkörper der Länge $l_2$, der sich zwischen zwei Kardangelenken befindet, ersetzt werden. Durch Ignorieren des letzten

Kardangelenks jedes Beines, kann das System, durch Vorgabe der drei Minimalkoordinaten $\boldsymbol{z}$, vereinfacht modelliert werden. Tabelle 5.3 zeigt die DENAVIT-HARTENBERG-Parameter für die gesamte Beinkette.

**Tabelle 5.3**: DH-Parameter der Beine des Delta-Roboters. Links: Standard DH-Parameter. Rechts: Modifizierte DH-Parameter

| $i$ | $\theta_i$ | $d_i$ | $a_i$ | $\alpha_i$ |
|---|---|---|---|---|
| 1 | $q_1$ | 0 | $l_1$ | 0 |
| 2 | $q_2$ | 0 | 0 | $\pi/2$ |
| 3 | $q_3$ | 0 | $l_2$ | 0 |
| 4 | $q_4$ | 0 | 0 | $\pi/2$ |
| 5 | $q_5$ | 0 | 0 | 0 |

| $i$ | $a(i)$ | $\sigma_i$ | $\gamma_i$ | $b_i$ | $\alpha_i$ | $d_i$ | $\theta_i$ | $r_i$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | $q_1$ | 0 |
| 2 | 1 | 0 | 0 | 0 | 0 | $l_1$ | $q_2$ | 0 |
| 3 | 2 | 0 | 0 | 0 | $\pi/2$ | 0 | $q_3$ | 0 |
| 4 | 3 | 0 | 0 | 0 | 0 | $l_2$ | $q_4$ | 0 |
| 5 | 4 | 0 | 0 | 0 | $\pi/2$ | 0 | $q_5$ | 0 |

### 5.1.4 4(P)RRR

Der 4(P)RRR-Roboter baut auf auf der 3RRR-PKM, bei der als erstes Gelenk zusätzlich ein Linearaktor verwendet wird. Mit vier solchen Beinketten, kann der Roboter räumliche Bewegungen mit drei translatorische Freiheitsgraden und einem rotatorischen Freiheitsgrad durchführen, sodass als Minimalkoordinaten der Vektor $\boldsymbol{z} = (x_\mathrm{P}, y_\mathrm{P}, z_\mathrm{P}, \phi_\mathrm{P})^\mathrm{T}$ (3T1R) gewählt wird. Während bei einem Bein das erste Lineargelenk $q_1$ aktuiert wird, ist das aktive Gelenk bei den drei anderen Beinketten das Drehgelenk $q_2$. Die DENAVIT-HARTENBERG-Parameter des Roboters können Tabelle 5.4 entnommen werden.

**Tabelle 5.4**: DH-Parameter der Beine des 4(P)RRR-Roboters. Links: Standard DH-Parameter. Rechts: Modifizierte DH-Parameter

| $i$ | $\theta_i$ | $d_i$ | $a_i$ | $\alpha_i$ |
|---|---|---|---|---|
| 1 | 0 | $q_1$ | 0 | 0 |
| 2 | $q_2$ | 0 | $l_1$ | 0 |
| 3 | $q_3$ | 0 | $l_2$ | 0 |
| 4 | $q_4$ | 0 | 0 | 0 |

| $i$ | $a(i)$ | $\sigma_i$ | $\gamma_i$ | $b_i$ | $\alpha_i$ | $d_i$ | $\theta_i$ | $r_i$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | $q_1$ |
| 2 | 1 | 0 | 0 | 0 | 0 |  | $q_2$ | 0 |
| 3 | 2 | 0 | 0 | 0 | 0 | $l_1$ | $q_3$ | 0 |
| 4 | 3 | 0 | 0 | 0 | 0 | $l_2$ | $q_4$ | 0 |

### 5.1.5 6UPS (STEWART-GOUGH)

Die 6UPS-PKM ist ein 3T3R-Roboter, der sich in alle Raumrichtungen bewegen kann und somit alle sechs Freiheitsgrade besitzt. Der Vektor der Minimalkoordinaten besteht aus $\boldsymbol{z} = (\boldsymbol{r}_\mathrm{P}^\mathrm{T}, \boldsymbol{\varphi}_\mathrm{P}^\mathrm{T})^\mathrm{T}$, wobei $\boldsymbol{r}_\mathrm{P}$ die Position der Plattform und $\boldsymbol{\varphi}_\mathrm{P}$ die Orientierung in EULER-Winkeln ist. Der Roboter besteht aus sechs Beinen, die wiederum aus dem passiven Kardangelenk, das durch zwei Drehgelenke $q_1$ und $q_2$ modelliert wird, gefolgt von dem aktiven Schubgelenk $q_3$ bestehen. Das Koppelgelenk ist ein Kugelgelenk und wird mit den drei Drehgelenken $q_4$, $q_5$ und $q_6$ modelliert. Die Beschreibung mit (modifizieren) DENAVIT-HARTENBERG-Parameter ist in Tabelle 5.5 zu finden.

**Tabelle 5.5**: DH-Parameter der Beine des 6UPS-Roboters. Links: Standard DH-Parameter. Rechts: Modifizierte DH-Parameter

| $i$ | $\theta_i$ | $d_i$ | $a_i$ | $\alpha_i$ |
|---|---|---|---|---|
| 1 | $q_1$ | 0 | 0 | $\pi/2$ |
| 2 | $q_2$ | 0 | 0 | $\pi/2$ |
| 3 | 0 | 0 | $q_3$ | 0 |
| 4 | $q_4$ | 0 | 0 | 0 |
| 5 | $q_5$ | 0 | 0 | $\pi/2$ |
| 6 | $q_6$ | 0 | 0 | $\pi/2$ |

| $i$ | $a(i)$ | $\sigma_i$ | $\gamma_i$ | $b_i$ | $\alpha_i$ | $d_i$ | $\theta_i$ | $r_i$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | $q_1$ | 0 |
| 2 | 1 | 0 | 0 | 0 | $\pi/2$ | 0 | $q_2$ | 0 |
| 3 | 2 | 1 | 0 | 0 | $\pi/2$ | 0 | 0 | $q_3$ |
| 4 | 3 | 0 | 0 | 0 | 0 | 0 | $q_4$ | 0 |
| 5 | 4 | 0 | 0 | 0 | $\pi/2$ | 0 | $q_5$ | 0 |
| 6 | 5 | 0 | 0 | 0 | $\pi/2$ | 0 | $q_6$ | 0 |

## 5.2 Vergleich der Ergebnisse

Zu Beginn der Auswertung wird überprüft, ob die neue Implementierung die Dynamikmodelle überhaupt korrekt berechnen kann. Hierzu dienen die anderen beiden Verfahren als Referenzlösung. Für den Test werden die Ergebnisse in der bereits beschriebenen Form aus (3.1), (3.11) und (3.37) verglichen. Die unterschiedlichen Transformationen, die durch die verschiedenen Übergabeargumente in den Formeln nötig sind, wurden zuvor in MATLAB durchgeführt. Die physikalischen Parameter sowie die Konfiguration des Roboters sind zufällig gewählt worden. Da die Ergebnisse für alle, in dieser Arbeit betrachteten, Beispielroboter mit den Referenzlösungen übereinstimmen, wird davon ausgegangen, dass der Ansatz korrekt implementiert ist. Aus Gründen der Übersichtlichkeit werden die Gleichungen, in denen die jeweiligen Ergebnisse vorliegen, hier noch einmal aufgelistet:

$$\begin{aligned} \boldsymbol{\tau}_q &= \boldsymbol{f}_\mathrm{dyn}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}, \boldsymbol{p}), \\ \boldsymbol{\tau}_\mathrm{a} &= \boldsymbol{J}_q^{q_\mathrm{a}} \boldsymbol{\tau}_q \end{aligned} \tag{5.1}$$

$$\boldsymbol{\tau}_{\mathrm{a}} = \boldsymbol{M}_{\mathrm{a}}(\boldsymbol{q}_{ges})\ddot{\boldsymbol{z}} + \boldsymbol{c}_{\mathrm{a}}(\boldsymbol{q}_{ges}, \dot{\boldsymbol{z}}) + \boldsymbol{g}_{\mathrm{a}}(\boldsymbol{q}_{ges}), \tag{5.2}$$

$$\boldsymbol{\tau}_{\mathrm{a}} = \boldsymbol{M}_{\mathrm{a}}(\boldsymbol{q}_{ges})\ddot{\boldsymbol{s}} + \boldsymbol{c}_{\mathrm{a}}(\boldsymbol{q}_{ges}, \dot{\boldsymbol{s}}) + \boldsymbol{g}_{\mathrm{a}}(\boldsymbol{q}_{ges}) \tag{5.3}$$

## 5.3 Vergleich der Struktur der Lösung

In PARADYN wird zunächst unterschieden, ob es sich um einen Roboter mit mehr als vier Freiheitsgraden handelt oder nicht. Falls das System mehr als vier Freiheitsgrade hat, wird die inverse Dynamik im Plattformkoordinatenraum und die Inverse der Gesamt-Jacobi-Matrix getrennt von einander ausgegeben. Dies hat den Grund, dass aus den bisherigen analytischen Berechnungen nur die Inverse der Gesamt-Jacobi-Matrix hervorgeht. Bei komplizierten Robotermechanismen, wie dem 6UPS-Roboter, wäre die analytische Invertierung sehr kostspielig. Durch die separate Ausgabe kann die Invertierung numerisch mit MATLAB erfolgen, um die inverse Dynamik im Plattformkoordinatenraum anschließend mit Hilfe der Gesamt-Jacobi-Matrix in den aktiven Antriebsraum zu transformieren. Während der Implementierung in dieser Arbeit wurde ebenfalls festgestellt, dass die symbolische Invertierung einer $(6 \times 6)$-Matrix eine lange Rechenzeit in Anspruch nimmt. Darum wird auch hier die Inverse der Gesamt-Jacobi-Matrix in einer getrennten Funktion ausgegeben, um die Invertierung numerisch vorzunehmen.

Die Ausgabe von OPENSYMORO liefert im Gegensatz zu den anderen beiden Verfahren das Dynamikmodell bezüglich aller aktiven und passiven Gelenke. Für das vollständige Modell müssen die Dynamikgleichungen mit einer Projektionsmatrix multipliziert werden, die ebenfalls ausgegeben wird. Darüber hinaus müssen dem Modell aus OPENSYMORO die Geschwindigkeiten und Beschleunigungen aller Gelenkwinkel vorgegeben werden.

Für die Dynamikmodellierung werden bei PARADYN für einen Roboter mit sechs Freiheitsgraden drei Funktionen zur Berechnung der inversen Dynamik ausgegeben. Neben der inversen Dynamik im Plattformkoordinatenraum und der Inversen der Gesamt-Jacobi-Matrix wird zusätzlich eine Funktion zur Berechnung der Minimalparameter erstellt. Die Funktion der inversen Dynamik greift auf die Rückgabewerte dieser Funktion zu. Bei der Implementierung in dieser Arbeit wurde auf die explizite Erzeugung der Minimalparameter in einer zusätzlichen Funktion verzichtet. Da in dem Ansatz nach Abdellatif et al. Quasigeschwindigkeiten verwendet werden, liegen auch die Momente der inversen Dynamik im Plattformkoordinatenraum im Raum der Winkelgeschwindkeiten vor, während die Momente bei PARADYN bezüglich seiner Drehraten berechnet werden. Durch die gleiche Berücksichtigung in der Gesamt-Jacobi-Matrix hebt sich der Unterschied bei der Berechnung der aktiven Momente wieder auf.

## 5.4 Vergleich der Rechenoperationen

Obwohl der Schwerpunkt in dieser Arbeit nicht auf einer möglichst guten Recheneffizienz der Lösung des inversen Dynamikmodells liegt, ist sie dennoch ein wichtiges Gütekriterium für die Implementierungen. Sowohl für die Regelung bzw. Steuerung als auch für die Simulation ist es notwendig die Berechnungen mit möglichst wenig Rechenschritten ausführen zu können. Aus diesem Grund steht beim Vergleich der neuen Implementierung mit den vorhandenen der Rechenaufwand im Vordergrund. Ein Vorteil der Berechnungen mit MAPLE ist, dass man mit dem Befehl `cost` die Anzahl der Rechenoperationen ermitteln lassen kann. Durch die Funktion `optimize` mit dem Übergabeargument `tryhard` kann das Modell optimiert und auf die Berechnung einzelner Ausdrücke reduziert werden. Da die Wahl der Dynamikparameter-Form einen erheblich Einfluss auf die Anzahl der Rechenoperationen hat, werden drei verschiedene Formen unterschieden:

- DynPar 1: Parameter bezogen auf den Schwerpunkt des Teilkörpers
- DynPar 1: Parameter bezogen auf den Koordinatenursprung und Angabe des 1. Moments
- DynPar 3: Minimalparameter gemäß [GK90], [Abd07]

### 5.4.1 Vergleich mit Abdellatif

Da sich die Form des Ergebnisses der Bewegungsgleichungen nach [Abd07] von der aus PARADYN unterscheidet und dies Auswirkungen auf die Anzahl der Rechenschritte hat, muss sich beim Vergleich auf eine gemeinsame Form geeinigt werden. Zunächst wurde aber überprüft, ob mit dem implementierten Ansatz die gleiche Anzahl an Operationen wie in [Abd07] erreicht werden kann. Dazu wird die Implementierung so verändert, dass sie der Gleichung gemäß (3.48) entspricht. Sie unterscheidet sich einerseits darin, dass die passiven Gelenkgeschwindigkeiten nicht durch die Minimalgeschwindigkeiten ersetzt werden und andererseits in der Verwendung von Quasigeschwindigkeiten. Betrachtet wird hier der PaLiDA-Roboter, der sich aber in seiner Struktur und der Kinematik nicht vom 6UPS-Roboter unterscheidet. Da sowohl die Beine des 6UPS- als auch die des PaLiDA-Roboters nur aus drei Gelenken und somit aus nur drei Teilkörpern bestehen, werden die physikalischen Parameter der restlichen modellierten Teilkörper nicht betrachtet. Um die überflüssigen Parameter nicht in den Berechnungen berücksichtigen zu müssen, wurden diese bereits vorher eliminiert. Näheres zur Auswahl der notwendigen physikalischen Parameter ist in [Abd07] zu finden.

Die Tabelle 5.6 zeigt, dass sich die Gesamtanzahl der Operationen der inversen Dynamik gemäß (3.48) der eigenen Implementierung deutlich gegenüber den Ergebnissen aus [Abd07]

**Tabelle 5.6**: Rechenoperationen für PaLiDA-Roboter mit notwendigen physikalischen Parametern.

| | $\pm$ | $\times$/: | sin/cos |
|---|---|---|---|
| PARADYN (DynPar 3) | 1075 | 1779 | 42 |
| Abdellatif (DynPar 3) | 1276 | 2045 | 42 |
| [Abd07] (DynPar 3) | 669 | 1480 | kA |

unterscheidet. Obwohl für beide Lösungen das Programm MAPLE zum Einsatz kam, hat man bei der Implementierung vielseitige Möglichkeiten die analytische Entwicklung der Formeln zu beeinflussen. So verwendet MAPLE z.B. nicht selbstständig alle Additionstheoreme zum Vereinfachen trigonometrischer Terme. Auch die Stellen im Code, an denen die Vereinfachungen durchgeführt werden, können ein Grund für die Abweichung sein. Bei der getrennten Betrachtung der Terme, fällt auf, dass die Dynamikterme der serielle Kette annähernd übereinstimmen, während die der Plattform sich fast um einen Faktor zwei unterscheiden. Da die vereinfachte Parametrisierung bezüglich der Vorannahmen übernommen wurde, lässt dies darauf schließen, dass für die Berechnung der Operationen weitere Vereinfachungen getroffen worden sind, die in [Abd07] nicht erwähnt werden. Desweiteren wird der Rechenaufwand der einzelnen Terme aufsummiert, ohne weitere mögliche Operationen bei der Addition aller Terme zu berücksichtigen. Damit liegen keine fairen Ausgangsbedingungen vor und ein aussagekräftiger Vergleich zwischen der Implementierung aus [Abd07] und dieser Arbeit ist nicht möglich.

### 5.4.2 Vergleich mit PARADYN und OPENSYMORO

Um den Vergleich der Implementierung nach dem Ansatz von Abellatif et al. und dem aus PARADYN auf die Projektion beschränken zu können, werden beide Dynamikmodelle gemäß (5.2) dargestellt. Da OPENSYMORO durch die Wahl der NEWTON-EULER-Methode einen von Grund auf anderen Ansatz wählt, werden die Ergebnisse nicht weiter aneinander angeglichen. Bei der generischen Robotererzeugung ist im Voraus nicht bekannt, welche physikalischen Parameter notwendig sind, sodass bei der Dynamikmodellierung alle berücksichtigt werden. Da die Bewegungsgleichungen von PARADYN und Abdellatif in der gleichen Zielfunktion vorliegen, ist davon auszugehen, dass der Rechenaufwand in der gleichen Größenordnung liegt.

Die Ergebnisse der in Abschnitt 5.1 beschriebenen Roboter können Tabelle 5.7 und Bild 5.1 entnommen werden. In Tabelle 5.7 werden alle Rechenoperationen nach ihrer Gewichtung aufsummiert. Hierfür wurden die Gewichtungsfaktoren, die für die Höhe ihres Rechenaufwandes stehen, aus [SLH19] übernommen. Bei allen Modellen wurde als EULER-Winkel-Konvention

stets die $xyz$-Konvention gewählt. Durch die getrennte Berechnung vom Dynamikmodell und der Gesamt-Jacobi-Matrix für Systeme mit mehr als vier FHG, werden die Ergebnisse in Tabelle 5.7 für den 6UPS separat in zwei Zeilen angegeben. Für einen differenzierten Vergleich werden die Operationen für die Projektionsmatrix aus OPENSYMORO ebenfalls getrennt aufgelistet. Bild 5.1 zeigt die Ergebnisse außerdem bezüglich der drei verschiedenen Dynamikparameter und bereits in den aktiven Gelenkraum projiziert in einem Balkendiagramm.

**Tabelle 5.7**: Anzahl der gewichteten Rechenoperationen für alle betrachteten Roboter.

| Faktor | $\pm$ 1 | $\times$ 1 | / 4 | sin/cos 10 | := 1 | (Gewichtete) Summe |
|---|---|---|---|---|---|---|
| Verfahren | 3RPR | | | | | |
| OPENSYMORO (DynPar 2) | 189 | 246 | 12 | 35 | 176 | 1009 |
| Abdellatif (DynPar 3) | 181 | 302 | 6 | 8 | 124 | 711 |
| PARADYN (DynPar 3) | 169 | 275 | 6 | 8 | 136 | 684 |
| Verfahren | 3RRR | | | | | |
| OPENSYMORO (DynPar 2) | 251 | 308 | 0 | 53 | 199 | 1288 |
| Abdellatif (DynPar 3) | 233 | 432 | 8 | 20 | 206 | 1103 |
| PARADYN (DynPar 3) | 209 | 321 | 8 | 20 | 193 | 955 |
| Verfahren | Delta | | | | | |
| OPENSYMORO (DynPar 2) | 890 | 1094 | 32 | 193 | 592 | 4634 |
| Abdellatif (DynPar 3) | 415 | 901 | 22 | 30 | 368 | 2072 |
| PARADYN (DynPar 3) | 427 | 730 | 22 | 24 | 420 | 1905 |
| Verfahren | 4(P)RRR | | | | | |
| OPENSYMORO (DynPar 2) | 628 | 642 | 0 | 79 | 358 | 2418 |
| Abdellatif (DynPar 3) | 293 | 543 | 9 | 26 | 253 | 1385 |
| PARADYN (DynPar 3) | 266 | 409 | 7 | 18 | 249 | 1132 |
| Verfahren | 6UPS - $\boldsymbol{\tau}_\mathbf{P}$ bzw. $\boldsymbol{\tau}_q$ | | | | | |
| OPENSYMORO (DynPar 2) | 1579 | 1904 | 30 | 41 | 1046 | 5059 |
| Abdellatif (DynPar 3) | 1472 | 2478 | 12 | 42 | 811 | 5229 |
| PARADYN (DynPar 3) | 1194 | 1864 | 14 | 42 | 942 | 4476 |
| Verfahren | 6UPS - Jacobi-Matrix $\boldsymbol{J}$ | | | | | |
| OPENSYMORO (DynPar 2) | 205 | 930 | 30 | 505 | 465 | 6770 |
| Abdellatif (DynPar 3) | 150 | 318 | 0 | 42 | 172 | 1060 |
| PARADYN (DynPar 3) | 120 | 312 | 0 | 42 | 245 | 1097 |

Bei den planaren System fallen die Unterschiede in der Anzahl der Operationen aller Verfahren gering aus. Wie erwartet, ist der Aufwand mit der Wahl von DynPar 1 am höchsten und mit DynPar 3 am niedrigsten. Während PARADYN für alle betrachteten Roboter die höchste Effi-

zienz besitzt, bildet OPENSYMORO immer das Schlusslicht. Mit zunehmender Komplexität des Roboters fällt OPENSYMORO immer weiter ab, sodass sich die Anzahl der Rechenschritte beim 6UPS fast verdoppelt hat. Das Verfahren nach Abdellatif liegt mit seinem Rechenaufwand stets in einem mittleren Bereich. Wie zu erwarten war, steigt die Anzahl der Operationen mit zunehmenden Freiheitsgraden an. Auffallend ist, dass der 4(P)RRR-Roboter mit mehr FHG weniger Operationen als der 3RUU-Roboter aufweist. Dies ist damit zu begründen, dass die Beinketten aus drei Drehgelenken in einer Ebene und dem Schubgelenk einfacher aufgebaut sind.

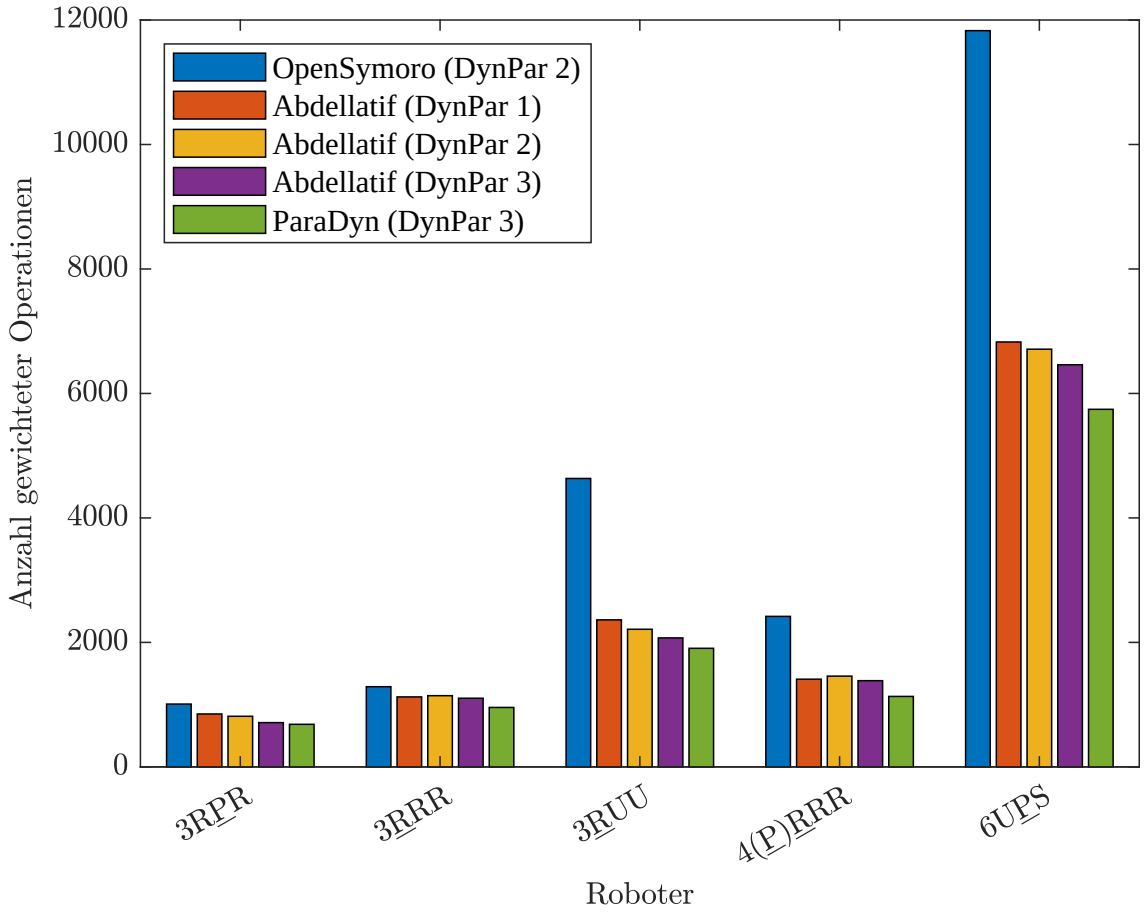



**Bild 5.1**: Vergleich der gewichteten Rechenoperationen aller drei Verfahren und mit verschiedenen Dynamik-Parametern.

Obwohl Abdellatif mit seinem Ansatz nahe an die Ergebnisse von PARADYN herankommt, ist der Aufwand für jeden hier betrachteten Roboter immer etwas größer. Der Hauptgrund für die Abweichungen in den Ergebnissen ist, dass PARADYN auf ein möglichst rechen-

effizientes Ergebnis ausgelegt ist. Neben den vielen Zwischenvereinfachungen, die in dem MAPLE-Code von PARADYN eingesetzt worden sind, wird am Ende ein selbst geschriebener Optimierungsalgorithmus ausgeführt. Zur Vereinfachung werden hier vor allem die Vielzahl an Additionstheoremen, die aus den analytischen Berechnungen bekannt sind, MAPLE aber von sich aus nicht einsetzt, verwendet. Aus den letzten beiden Zeilen aus Tabelle 5.7 wird deutlich, dass der hohe Rechenaufwand OPENSYMORO aus der separaten Berechnung der Projektionsmatrix resultiert. Besonders hervorzuheben ist die enorme Anzahl der sin- und cos-Funktionen, die darauf zurückzuführen ist, dass redundante Ausdrücke in den Berechnungen nicht eliminiert werden.

Der absteigende Rechenaufwand von DynPar 1 bis DynPar 3 lässt sich damit erklären, dass die Dynamikparameter zu immer kürzeren Ausdrücken zusammengefasst werden. Z. B. werden in DynPar 2 die Trägheitsparameter auf den Ursprung bezogen, wodurch die Steiner-Anteile bereits in den Parametern enthalten sind. Die Abweichung dieser Tendenz beim 3RRR und 4(P)RRR ist damit zu begründen, dass es MAPLE gelingt, die trigonometrischen Funktionen für DynPar 1 zu vereinfachen. In DynPar 3 werden die einzelnen Dynamikparameter aus DynPar 2 weiter zu Minimalparametern zusammengefasst, was wiederum zu einer Reduzierung der Rechenoperationen führt.

### 5.4.3 Einfluss der EULER-Winkel

Es existieren 12 EULER-Winkel-Konventionen, mit denen die Orientierung der Plattform beschrieben werden kann. Durch die Implementierung der automatischen Dynamikmodellierung ist es möglich, die Bewegungsgleichungen für alle Konventionen zu generieren. Der Vergleich der Anzahl an gewichteten Rechenoperationen hat allerdings ergeben, dass die Wahl der EULER-Winkel keinen Einfluss auf den Rechenaufwand hat. Dies ist damit zu begründen, dass die unterschiedlichen Konventionen die gleiche Anzahl an trigonometrischen Funktionen enthalten. Die in MAPLE genutzte Optimierung kann den Rechenaufwand für keine der gewählten Konventionen maßgeblich reduzieren.

## 5.5 Begrenzungen

Die Verwendung des Ansatzes nach Abdellatif et al. bringt gewisse Begrenzung mit sich. Es müssen bestimmte Vorgaben eingehalten werden, damit das Verfahren zum Ziel kommt. Aus der Wahl der Beispielroboter wurde bereits ersichtlich, welche Aufgabentypen möglich sind. Im Folgenden wird erläutert, welche PKM mit dem Ansatz verwendet werden können und

warum das Verfahren auf Robotertypen beschränkt ist, deren Gelenkanzahl eines Beines bis zum Koppelpunkt den translatorischen FHG des Roboters entspricht.

### 5.5.1 Schnittgelenk

Sowohl in dem neuen Ansatz als auch in dem von PARADYN wird das Koppelgelenk als Schnittgelenk gewählt, sodass nur die Kinematik der Beine bis zum Koppelpunkt mit der Plattform vorgegeben wird. Das Koppelgelenk wird dabei nicht betrachtet. Wenn davon ausgegangen wird, dass es sich um ein reines Drehgelenk handelt, hat es auf die kinematischen Zwangsbedingungen keinen Einfluss. Wie oben bereits erklärt, werden vor dem Ausführen des Programms die MDH-Parameter für jedes Bein definiert. Die MDH-Parameter sind im Gegensatz zu den DH-Parametern so definiert, dass sich die Parameter des aktuellen Koordinatensystems immer auf das letzte beziehen. Für die vollständige Beschreibung der seriellen Kette müssen demnach auch die Parameter des Koppelgelenks angegeben werden. Beim Aufstellen der translatorischen Jacobi-Matrix der seriellen Kette wird das Koppelgelenk allerdings automatisch eliminiert und hat somit keinen Einfluss auf die Kinematik des Roboters.

Da das Schnittgelenk eliminiert wird, fehlen dem Verfahren die Informationen des Roboters. Es geht deshalb automatisch davon aus, dass das letzte Gelenk der Beine auf die nötigen Freiheitsgrade „auffüllt“. Z.B. besitzt die Plattform beim 3RRR drei Freiheitsgrade, die definierten Beine bis zum Koppelpunkt allerdings nur zwei. Das Verfahren geht demnach davon aus, dass es sich bei dem Schnittgelenk um ein einfaches Drehgelenk handelt. Die Beine des 6UPS hingegen besitzen bis zum Koppelpunkt nur drei Freiheitsgrade. Um auf die sechs Freiheitsgrade der Plattform zu kommen, geht das Verfahren von einem Kugelgelenk als Koppelgelenk aus. Der Delta-Roboter ist ein Sonderfall, bei dem im Vorhinein die Parallelogrammstruktur durch ein Kardangelenk ersetzt worden ist. Da das Kardangelenk schon alle Freiheitsgrade der Parallelogrammstruktur abdeckt, geht das Verfahren davon aus, dass das letzte Gelenk keinen Einfluss auf die Freiheitsgrade der seriellen Kette hat.

### 5.5.2 Redundanzen

Die Bisherige Struktur der Implementierung lässt nur die Modellierung von Parallelrobotern mit Antriebsredundanz durch eine zusätzliche serielle Kette zu. Hier unterscheidet sich das Ergebnis darin, dass die Inverse der Gesamt-Jacobi-Matrix nicht quadratisch ist. Eine klassische Invertierung dieser Matrix kann nicht durchgeführt. Unter der Nebenbedingung, dass die Geschwindigkeiten der Antriebsgelenke möglichst gering sein sollen, kann zur Invertierung die Moore-Penrose-Inverse verwendet werden. Dies hat keine Auswirkung auf das Aufstellen der Dynamikmodelle sowie auf die Transformation der Gleichungen. Auch die Antriebsredundanz

durch Ersetzen eines passiven durch ein aktives Gelenk betrifft lediglich die Inverse der Gesamt-Jacobi-Matrix. Hierzu wird der differentielle Zusammenhang zwischen Plattformkoordinaten und dem zusätzlichen aktiven Gelenk hergestellt. Dadurch ist Erweiterung der Implementierung auf solche Redundanzen problemlos möglich.

Für parallele Roboter mit kinematischer Redundanz existieren redundante serielle Ketten. Da die Jacobi-Matrix der seriellen Kette nicht quadratisch ist, kann sie nicht invertiert werden. Die Nutzung der Moore-Penrose-Inverse, die impliziert, dass die Gelenkgeschwindigkeiten minimal gehalten werden, ist hier nachteilig. Dies bringt eine vorherige Einschränkung der Lösung mit sich, obwohl die passiven Gelenkgeschwindigkeiten nicht relevant sind. In PARADYN wird dieses Problem durch die Vorgabe, dass das zusätzliche, kinematische redundante Gelenk als eine Komponente der Minimalkoordinaten verwendet wird, gelöst. Dies ist möglich durch das Aufstellen aller Transformationsmatrizen aus den kinematischen Zwangsbedingungen. Ein solcher Ansatz ist mit dem Verfahren aus Abdellatif et al. nicht möglich, da das Aufstellen der Transformationsmatrizen speziellen Regeln folgt. Somit können kinematische Redundanzen mit dem zu implementierenden Ansatz nicht gelöst werden und stellen somit eine Limitierung dar.

### 5.5.3 Singularitäten

Singularitäten treten durch Rangabfall ($det(\boldsymbol{J}) = 0$) einer zu invertierenden quadratischen Jacobi-Matrix auf und limitieren das Verfahren. Die Inverse einer solchen Matrix kann nicht bestimmt werden und es kommt grundsätzlich zu einem Verlust bzw. Gewinn von Freiheitsgraden des Endeffektors. Die Implementierung des Ansatzes aus Abdellatif et al. enthält jene Invertierungen an zwei Stellen des Verfahrens. In (3.43) wird die Jacobi-Matrix eines Beines $\boldsymbol{J}_{q_j}^{\mathrm{K}_j}$, die den Zusammenhang zwischen Gelenkgeschwindigkeiten und Koppelpunktgeschwindigkeiten herstellt, invertiert. Eine quadratische Matrix wird dadurch sichergestellt, dass eine Beinkette bei räumlichen Systemen aus drei Gelenken und bei planaren Systemen aus zwei Gelenken besteht. Die Koppelgelenke werden hierbei nicht berücksichtigt. Singuläre Stellungen entstehen hierbei sowohl auf als auch innerhalb der Grenzen des Arbeitsraumes einer seriellen Kette und müssen vermieden werden. In (3.54) wird die Gesamt-Jacobi-Matrix für parallele Roboter invertiert. Auch hier kann es auf und innerhalb der Grenzen des Arbeitsraumes zum Rangabfall kommen. Für eine differenziertere Untersuchung der singulären Stellungen sei auf [Mer06] verwiesen, in der drei verschiedene Arten unterschieden werden.

# 6 Zusammenfassung

Im Rahmen dieser Arbeit wurde eine automatische Dynamikmodellierung für parallelkinematischen Maschinen implementiert. Hierzu wurde eine bereits vorhandene Toolbox für die Modellierung serieller und hybrider Strukturen um die Berechnung paralleler Roboter erweitert. Diese basiert auf Arbeitsblättern des Computer-Algebra-Systems MAPLE, die über Bash-Skripte ausgeführt werden. Damit wurden die Vorteile beider Systeme ausgenutzt, um eine allgemeine, automatische und flexible Modellierung für PKM zu erhalten. Hierbei können die Bewegungsgleichungen bezüglich unterschiedlicher Dynamikparameter sowie in parameterlinearer Form erzeugt werden. Als Vorgabe werden modifizierte DENAVIT-HARTENBERG-Parameter, die Nummer des aktiven Gelenks, die Transformation zum Basis-Koordinatensystem der Beinketten und die Plattformkoordinaten im Inertialkoordinatensystem benötigt.

Der Rechenaufwand der erhaltenen Lösung ist vergleichbar mit dem Aufwand aus den Ergebnissen bereits vorhandener Implementierungen, wobei sich die Anzahl der Rechenoperationen durch die Wahl der Dynamikparameter beeinflussen lässt. Durch die Realisierung einer Optimierung mit dem Schwerpunkt auf die Anwendung trigonometrischer Additionstheoreme lässt sich die Recheneffizienz vermutlich noch verbessern. Die automatische Dynamikmodellierung in der jetzigen Form ermöglicht es die Bewegungsgleichungen für symmetrische PKM zu bestimmen. Hierbei muss beachtet werden, dass die Anzahl der Gelenke eines Beines bis zum Koppelpunkt den translatorischen FHG des Roboters entspricht. Dadurch können auch Roboter mit Antriebsredundanz durch eine zusätzliche serielle Kette berechnet werden. Durch den gewählten Ansatz ist es grundsätzlich auch möglich Roboter mit Antriebsredundanz durch mehrere aktive Gelenke in einer Beinkette zu modellieren.

Die Implementierung liegt nun für die Berechnung klassischer PKM vor und bietet die passenden Voraussetzungen, um in einer weiteren Arbeit darauf aufzubauen. Hierbei ist die Untersuchung über die Verwendung hybrider Beinketten denkbar. Die Toolbox, die hier erweitert wurde, kann die Modelle für hybride Roboter mit eliminierten Zwangsbedingungen bereits berechnen. Damit kann überprüft werden, ob die Modellierung solcher Systeme mit dem hier genutzten Projektionsverfahren besser funktioniert, als mit PARADYN.

# Literaturverzeichnis


[Abd07] ABDELLATIF, H.: *Modellierung, Identifikation und robuste Regelung von Robotern mit parallelkinematischen Strukturen*, Leibniz Universität Hannover, Dissertation, 2007

[AH09] ABDELLATIF, H. ; HEIMANN, B.: Computational efficient inverse dynamics of 6-DOF fully parallel manipulators by using the Lagrangian formalism. In: *Mechanism and Machine Theory* 44 (2009), Nr. 1, S. 192–207

[BK15] BRIOT, S. ; KHALIL, W.: *Dynamics of Parallel Robots*. Springer, 2015

[DPF13] DOCQUIER, N. ; PONCELET, A. ; FISETTE, P.: ROBOTRAN: a powerful symbolic gnerator of multibody models. In: *Mechanical Sciences* 4 (2013), Nr. 1, 199–219. `https:/www.mech-sci.net/4/199/2013/ms-4-199-2013.pdf`

[DTKHO09a] DO THANH, T ; KOTLARSKI, J ; HEIMANN, B ; ORTMAIER, T: A new program to automatically generate the kinematic and dynamic equations of general parallel robots in symbolic form. In: *Proc. of the 1st IFToMM International Symposium on Robotics and Mechatronics*, 2009, S. 122–128

[DTKHO09b] DO THANH, T. ; KOTLARSKI, J. ; HEIMANN, B. ; ORTMAIER, T.: On the inverse dynamics problem of general parallel robots. In: *Proc. of the 2009 IEEE International Conference on Mechatronics* IEEE, 2009, S. 1–6

[Gin08] GINSBERG, J.: *Engineering Dynamics 2008*. Cambridge University Press, 2008

[GK90] GAUTIER, M. ; KHALIL, W.: Direct calculation of minimum set of inertial parameters of serial robots. In: *IEEE Transactions on Robotics and Automation* 6 (1990), June, Nr. 3, S. 368–373

[KC97] KHALIL, W. ; CREUSOT, D.: SYMORO+ : A system for the symbolic modelling of robots. In: *Robotica* 15 (1997), Nr. 2, S. 153–161

[KD02] KHALIL, W. ; DOMBRE, E.: *Modeling, Identification and Control of Robots*. Hermes Penton Science, 2002

[KVK+14] KHALIL, W. ; VIJAYALINGAM, A. ; KHOMUTENKO, B. ; MUKHANOV, I. ; LEMOINE, P. ; ECORCHARD, G.: OpenSYMORO: An open-source software package for Symbolic Modelling of Robots. In: *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. Besançon, France, Juli 2014, 1206-1211

[LCGB06] LÓPEZ, M. ; CASTILLO, E. ; GARCÍA, G. ; BASHIR, A.: Delta robot: Inverse, direct, and intermediate Jacobians. In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 220 (2006), jan, Nr. 1, S. 103–109

[Mei70] MEIROVITCH, L.: *Methods of analytical dynamics*. McGraw-Hill, 1970 (McGraw-Hill classic textbook reissue series)

[Mer06] MERLET, J.: *Parallel Robots*. Bd. 74. 2nd. Springer Science & Business Media, 2006

[OK14] ORTMAIER, T. ; KOTLARSKI, J.: *Skript zur Vorlesung Robotik I*. Institut für mechatronische Systeme, 2014

[Qui90] QUINN, R. D.: Equations of Motion for Structures in Terms of Quasi-Coordinates. In: *Journal of Applied Mechanics* 57 (1990), Nr. 3

[Sch] SCHAPPLER, Moritz: HybridDyn-Repository. `https://phabricator.imes.uni-hannover.de/source/robot_dynamics-hybriddyn/`. – Version 10.08.2018; Abzweigung am 16.08.2018

[SLH19] SCHAPPLER, M. ; LILGE, T. ; HADDADIN, S.: Kinematics and Dynamics Model via Explicit Trigonometric Elimination of Kinematic Constraints for a Force Assistance Exoskeleton. In: *15th IFToMM World Congress, Krakow, Poland* (2019). – Vorläufige Version von 09/2018