# Steuerung von sechsbeinigen Laufrobotern unter dem Aspekt technischer Anwendungen

## Dissertation

Zur Erlangung des akademischen Grades

## Doktoringenieur (Dr.-Ing.)

angenommen durch die Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg

von: Dipl.-Ing. Thomas Ihme
geb. am: 21. 11. 1968 in Magdeburg

Gutachter:
Prof. Dr. Edgar Nett
Prof. Dr. Rüdiger Dillmann
Prof. Dr. Peter Hauptmann

Promotionskolloquium:
Magdeburg, den 21. März 2002

Gedruckt auf alterungsbeständigem Papier °° ISO 9706

## Zusammenfassung

Die vorliegende Arbeit hat das Ziel, Methoden zur Steuerung von sechsbeinigen Laufrobotern unter dem Aspekt technischer Anwendungen aufzuzeigen und mit Hilfe eines Laufroboterprototypen zu demonstrieren. Dabei wird die Betrachtung des Laufens als Fortbewegungsmethode auf die Unterstützung technischer Operationen durch Körperbewegungen erweitert. Bei technischen Anwendungen spielt die autonome Steuerung eine wichtige Rolle, so dass ein geeignetes Steuerungssystem in den Roboter integriert werden muss. Die Steuerungsalgorithmen müssen trotz Ressourcenbeschränkungen in einer bestimmten Zeit Steuerungsinformationen bereitstellen, um den durch das mechatronische System bedingten Echtzeitanforderungen zu genügen. Es wird gezeigt, wie deterministische Bewegungen, die für technische Anwendungen wichtig sind, erzeugt werden können. Dazu werden Laufbewegungen, Körperbewegungen und deren Kombination sowie kraftgeführte Bewegungen betrachtet, die Grundlage zur Ausführung technischer Operationen sind.

## Danksagung

Diese Arbeit entstand als Ergebnis meiner mehrjährigen Tätigkeit am Fraunhofer Institut für Fabrikbetrieb und –automatisierung Magdeburg und an der Otto-von-Guericke-Universität Magdeburg, bei der ich mich mit der Steuerung eines Laufroboterprototypen beschäftigte.

Ich möchte mich an dieser Stelle bei Herrn Prof. Edgar Nett bedanken, der mir bei der Verbesserung meines Verständnisses für Probleme der Informationsverarbeitung unter Echtzeitbedingungen sehr geholfen hat. Durch seine Betreuung hat er wesentlich zum Gelingen dieser Arbeit beigetragen.

Mein Dank gilt in diesem Zusammenhang auch den Mitarbeitern des Lehrstuhls Echtzeitsysteme und Kommunikation, die mir durch Kommentare, Diskussionen und technische Unterstützung geholfen haben.

Mein Dank gilt weiterhin Prof. Schenk, Dr. Schmucker und Dr. Schneider, die das Laufroboterprojekt ins Leben gerufen haben und mir die Möglichkeit gaben, daran mitzuwirken. In diesem Zusammenhang möchte ich allen Mitarbeitern und ehemaligen Studenten der Abteilung Automatisierung des IFF für ihre direkte und indirekte Mitarbeit am Laufroboterprojekt danken.

Nicht zuletzt möchte ich mich bei meiner Familie bedanken, die viel Verständnis und Geduld aufgebracht hat und so zum Gelingen dieser Arbeit betrug. Mein besonderer Dank gilt meiner Frau, die mich beim Korrekturlesen und bei der schreibtechnischen Erstellung der Arbeit unterstützt hat.

## Inhaltsverzeichnis

# Verzeichnis der verwendeten Formelzeichen und Abkürzungen

## Zur Notation von Formelzeichen und Transformationen

Die angegebenen Größen beziehen sich in der Regel auf das aktuell verwendete Koordinatensystem. Zur Vereinfachung der Notation wird dieses Koordinatensystem nicht immer explizit gekennzeichnet. In den Fällen, wo sich Größen auf ein anderes Koordinatensystem beziehen, wird dies zur eindeutigen Kennzeichnung angegeben. Für eine Größe X in einem Koordinatensystem A gilt folgende Notation

$${}^{A}\mathbf{X}^{(i)}_{soll}$$

*A* – Koordinatensystem in dem die Größe dargestellt wird
(*i*) – Bezugnahme, z.B. Bezug auf Bein *i*
soll – Bedeutung bzw. Verwendung der Größe

Die Transformation einer Größe X von einem Koordinatensystem A in ein Koordinatensystem B wird wie folgt angegeben:

$${}^{B}\mathbf{X} = {}^{B}_{A}\mathrm{T}\left\{{}^{A}\mathbf{X}\right\}$$

Dabei kennzeichnet der tiefgestellte Index am Transformationssymbol das Ursprungskoordinatensystem und der hochgestellte Index das Zielkoordinatensystem [Wloka 92].

## Verzeichnis der wichtigsten verwendeten Formelzeichen

| | |
|---|---|
| $A$ | Querschnitt, Übertragungsverhalten |
| $B$ | Breite, geometrischer Parameter |
| $B_i$ | Nummer des Beines, i=1..6 |
| $C$ | Ausführungszeit, geometrischer Parameter, Kapazität |
| $D$ | Durchmesser, geometrischer Parameter |
| $d_n$ | Achsversatz |
| $E$ | Elastizitätsmodul |
| $F,\ \vec{F}$ | Kraft, Kraftvektor |
| $F_g$ | Gewichtskraft |
| $f$ | Frequenz |
| $G_0$ | Übertragungsfunktion der offenen Kette |
| $G_k$ | Beingruppe für den Dreifussgang; k=1,2 |
| $G_R$ | Reglerübertragungsfunktion |
| $G_S$ | Streckenübertragungsfunktion |
| $H$ | Dicke |
| $I,\ i$ | Strom |
| $\mathbf{K}_{FN}$ | Nachgiebigkeitsmatrix |
| $\mathbf{K}_{F2}$ | Anpassungsmatrix |
| $K_R$ | Übertragungskonstante des Reglers |

| | |
|---|---|
| $k_{FN}$ | Nachgiebigkeitsfaktor |
| $L$ | Entfernung, Länge, Induktivität |
| $L_{Stütz}$ | Schrittlänge |
| $L_{Transfer}$ | zurückgelegter Weg in der Transferphase |
| $M$ | Moment |
| $m$ | Massepunkt, Masse |
| $\vec{n}$ | Normalenvektor |
| $n$ | Anzahl, Auflösung |
| $P$ | Leistung |
| $\vec{q}$ | Vektor der Freiheitsgrade $\alpha$, $\beta$, $\gamma$ |
| $R$ | Elektrischer Widerstand |
| $\vec{R}, \vec{r}$ | Ortsvektor |
| $\mathbf{S}$ | Selektionsmatrix |
| $s$ | Laplace-Operator |
| $T$ | Temperatur, Zeitdauer |
| $t$ | Zeit |
| $U$ | Auslastungsfaktor |
| $U, u$ | Spannung |
| $V$ | Verstärkung |
| $v$ | Geschwindigkeit |
| $w$ | Wortbreite |
| $X, x$ | Position |
| | |
| $\alpha, \beta, \gamma$ | Freiheitsgrad; Gelenkwinkel |
| $\alpha_T$ | Temperaturkoeffizient |
| $\beta$ | Stützverhältnis, Tastverhältnis |
| $\beta_{PWM}$ | Tastverhältnis |
| $\varepsilon$ | Dehnung |
| $\varepsilon_q$ | Querdehnung |
| $\mu$ | Poisson-Zahl |
| $\rho$ | Spezifischer elektrischer Widerstand |
| $\omega$ | Drehzahl, Laufrichtungswinkel; Laufrichtung |
| $\omega_g$ | Grenzfrequenz |

**Verzeichnis der verwendeten Abkürzungen**

| | |
|---|---|
| ADU | Analog-Digital-Umsetzer |
| ALU | Arithmetisch-logische Einheit |
| CAN | Controller Area Network |
| CCITT | Comité Consultatif International Télégraphique et Téléphonique |
| CHMOS | Complementary High-density Metal Oxide Semiconductor |
| CMOS | Complementary Metal-Oxide-Semiconductor |
| CPU | Central Processing Unit |
| CRC | Cyclic Redundancy Check |
| DKT | Direkte kinematische Transformation |
| DMS | Dehnmeßstreifen |
| E/A | Eingabe/Ausgabe |

| | |
|---|---|
| ECET | Expected Case Execution Time |
| EDF-Scheduling | Earliest-Deadline-Scheduling |
| EPA | Event Processor Array |
| FPGA | Field Programmable Gate Array |
| Fraunhofer AIS | Fraunhofer Institut für Autonome Intelligente Systeme |
| Fraunhofer IFF | Fraunhofer Institut für Fabrikbetrieb und -automatisierung |
| FZI Karlsruhe | Forschungszentrum Informatik Karlsruhe |
| GMD | Gesellschaft für Mathematik und Datenverarbeitung |
| IKT | Inverse kinetische Transformation |
| I/O | Eingabe/Ausgabe |
| ISA | Instruction Set Architecture |
| ISO | International Organization for Standardization |
| ISR | Interrupt-Service-Routine |
| MA | Multiple Access |
| MCA2 | Modular Controller Architecture |
| NMI | Nicht maskierbarer Interrupt |
| OSI | Open System Interconnection |
| PTS | Peripheral Transaction Server |
| PWM | Puls-Weiten-Modulation |
| RAM | Random Access Memory |
| ROM | Read Only Memory |
| RRR | Kennzeichnung für drei rotatorische Gelenke |
| SIMD | Single Instruction Multiple Data |
| SSIO | Synchrone serielle Kommunikationseinheit |
| TAFT | Time Aware Fault Tolerant |
| TDMA | Time Division Multiple Access |
| TU München | Technische Universität München |
| UART | Universal Asynchronous Receiver Transmitter |
| WCET | Worst Case Execution Time |

# 1 Einleitung

Mit Hilfe der natürlichen Fortbewegung, dem Laufen, ist es für Menschen möglich, an einen bestimmten Ort zu gelangen. Das Laufen bereitet den Menschen und an Land lebenden Tieren wenig Mühe. Dabei stellen kleine Gräben oder Hindernisse wie Felsblöcke keine wirklichen Probleme im Hinblick auf die Beinkoordination dar.

Seit Generationen versuchen Techniker und Ingenieure, die Laufbewegung auf Maschinen zu übertragen. Neben der Faszination, natürliche Systeme nachzuahmen, lässt sich mit Hilfe der Lauftechnologie potentiell ein großer Bereich von bisher unerschlossenen Gebieten unseres Planeten erreichen, denn etwa die Hälfte unseres Planeten ist für Räderfahrzeuge unpassierbar [Rosheim 94].

## 1.1 Motivation

Der Vorteil der schreitenden gegenüber der rollenden Fortbewegung besteht darin, dass die Anforderungen an die natürliche oder vorhandene Infrastruktur der Umgebung wesentlich geringer sind. Das trifft sowohl auf das Bodenrelief als auch auf die physikalischen Eigenschaften des Bodens zu. Besondere Vorteile von Laufmaschinen sind [Schmucker et al. 94]:

- hohe Beweglichkeit und Manövrierfähigkeit,
- Überwindung von Hindernissen bis etwa der Beingröße,
- Fähigkeit zum Treppensteigen und zum Durchqueren enger Durchbrüche,
- Möglichkeit der Arbeit in unstrukturiertem (z.B. zerklüfteten) Gelände,
- diskontinuierliche Spur, wählbare Stützstellen auf dem Untergrund,
- Adaptionsfähigkeit an Untergründe mit unbekannter und wechselnder Tragfähigkeit,
- Nutzung einzelner Beine als Manipulator und
- Nutzung des Körpers als frei positionierbare Trägerplattform für Instrumente.

Diese Vorteile werden im Vergleich zur rollenden Fortbewegung durch eine erheblich höhere Komplexität des Gesamtsystems erkauft. Hierbei sind beispielsweise Probleme zur Koordinierung aller Beine, der Sensorsignalverarbeitung und der Implementierung der Steuerungsfunktionen zu lösen. Die Steuerungsfunktionen müssen dabei in einer hinreichend kurzen Zeit auf Steuerkommandos und Sensorsignale reagieren. Die Funktion des Gesamtsystems ist erst dann möglich, wenn alle Teilkomponenten, wie Mechanik, Elektronik und Informationsverarbeitung aufeinander abgestimmt sind.

Die Anwendungsmöglichkeiten von Laufrobotern sind nicht allein auf ihre Transportfähigkeiten begrenzt. Bei technischen Anwendungen besteht in der Regel die Forderung, neben Transport bzw. Selbsttransport am Zielort bestimmte Arbeiten auszuführen. Das sind beispielsweise Manipulation von Objekten oder die Positionierung von Instrumenten und Sensoren. Diese Arbeiten können durch die Nutzung von Körperbewegungen wirkungsvoll unterstützt werden.

## 1.2 Ziel und Aufbau der Arbeit

Ziel dieser Arbeit ist es, Steuerungsalgorithmen zu entwickeln und zu testen, die den Einsatz von sechsbeinigen Laufrobotern unter dem Aspekt technischer Anwendungen erlauben. Eine wichtige Fragestellung ist die Autonomie des Roboters, die die autonome Steuerung seiner Bewegungen einschließt. Hierbei spielen Fragen des Zusammenspiels von Mechanik, Elektronik und zu implementierender Steuerungsalgorithmen eine zentrale Rolle.

Ein wichtiger Aspekt bei technischen Anwendungen ist, dass die Steuerung der Bewegungen deterministisch ist und sich somit ein vorhersagbares Verhalten ergibt. Die Roboterbewegungen sollen nach einer analytischen Beschreibung auf einem speziellen Roboter implementiert und getestet werden.

Im Kapitel zwei wird zunächst ein Überblick über die bisherige Entwicklung von Laufmaschinen gegeben und mögliche Anwendungsgebiete von Laufmaschinen näher betrachtet.

Kapitel drei beschreibt die mechanische Konstruktion des Laufroboters Katharina, der als Prototyp für die Entwicklung des Steuerungssystems zur Verfügung stand.

Ausgehend von der Zielstellung der Steuerung des Laufroboters wird in Kapitel 4 das Konzept für ein Steuerungssystem entwickelt. Dabei wird sowohl auf die logische Struktur als auch auf mögliche Varianten der Rechnerarchitektur eingegangen.

Mechatronische Systeme bilden eine Einheit aus Mechanik, Elektronik und Informationsverarbeitung. Kapitel 5.1 befasst sich deshalb mit Aspekten der Hardware, der Informationsgewinnung mit Hilfe von Sensoren, der Informationsverarbeitung, die mit Hilfe eines Mikrocontrollersystems realisiert wird und der Ansteuerung von Aktoren. Kapitel 5.2 beschreibt die zugrundeliegenden Steuerungsalgorithmen, mit deren Hilfe Lauf- und Körperbewegungen erzeugt werden. In Kapitel 5.3 wird die Funktionsfähigkeit der Steuerungsalgorithmen auf dem realen Roboter untersucht und experimentelle Ergebnisse gezeigt. Kapitel 5.4 geht auf die Implementierung der Steuerungsalgorithmen auf einem Mikrocontrollersystem ein. Dabei spielt das Zusammenwirken der einzelnen Teilalgorithmen auf den Mikrocontrollern ebenso eine Rolle wie die Implementierung unter Echtzeitbedingungen.

In Kapitel 6 wird gezeigt, wie die Steuerungsalgorithmen dazu genutzt werden können, technische Operationen auszuführen.

In Kapitel 7 wird ausgehend von den Erfahrungen mit der realisierten Laufrobotersteuerung darauf eingegangen, wie künftige Realisierungen verbessert werden können.

Abschließend erfolgt eine Zusammenfassung.

# 2 Stand der Forschung und Anwendungsgebiete von Laufrobotern

Im folgenden Kapitel soll die historische Entwicklung von Laufrobotern kurz dargestellt werden, sowie einige aktuelle Forschungsprojekte näher erläutert werden. Des weiteren erfolgt ein Überblick über mögliche Anwendungsgebiete von Laufrobotern.

## 2.1 Historische Entwicklung von Laufmaschinen

Das Bestreben, das Fortbewegungsprinzip Laufen mit Hilfe von Mechanismen und Maschinen umzusetzen, ist schon Jahrhunderte alt. Erst moderne Methoden und technische Möglichkeiten haben einen großen Entwicklungsvorschub leisten können. Die Entwicklung von Laufmaschinen kann in mehrere große Etappen eingeteilt werden, die eng mit dem Wissen und dem technischen Möglichkeiten der jeweiligen Zeit verbunden ist. Nachfolgend wird auf einzelne Entwicklungsetappen eingegangen.

*Mechanismen*

Mechanismen stellten für die Menschen die erste Möglichkeit dar, gezielt Bewegungen zu erzeugen. Mit ihrer Hilfe wurden erste automatische Einrichtungen oder animierte Figuren gebaut [Rosheim 94]. Der älteste bekannte Laufmechanismus stammt aus dem alten China und ist einem Rind (Niú) nachempfunden. Es handelt sich um eine Holzkonstruktion aus dem Jahr 231. Der Laufmechanismus konnte 200-250 kg Nutzlast bei einer Geschwindigkeit von 10 km pro Tag transportieren [Morecki 97]. Neue Konstruktionen kamen Ende des 19. Jahrhunderts auf und basierten auf einem vom russischen Mathematiker Tschebyschew entworfenen Mechanismus mit einem Koppelgetriebe. Dieser Mechanismus erzeugt eine Koppelkurve, die als Lauftrajektorie genutzt werden kann. Bekanntestes Beispiel ist das mechanische Pferd, wofür Lewis. A. Rygg 1893 ein Patent erteilt bekam [Rygg 1893], [Raibert 86a]. Allen Mechanismen gemeinsam ist, dass die für Bewegungsfunktionen notwendigen Informationen über Zusammenhänge von Ort und Zeit durch mechanische Konstruktionen kodiert worden sind. Durch die fest kodierten Bewegungssequenzen bestand keine Möglichkeit der Anpassung an ein Terrain.

*Maschinen*

Mit der Einführung von Kraftverstärkern erfolgte der nächste Entwicklungsschritt. Die Nutzung von Kraftverstärkern ermöglichte, die Anzahl von Bewegungsvarianten zu vergrößern. Den Meilenstein der Entwicklung markiert der vierbeinige Walking Truck von General Electric um 1965. Die Steuerung der Bewegungen wurde von einem Operator übernommen, der mit Händen und Füßen die vier Beine steuerte. Die Kraftverstärkung erfolgte mittels Hydraulik. Durch Kraftrückführung wurde dem Bediener ein Bodenkontakt vermittelt, so dass eine Geländeanpassung durch Operatorsteuerung möglich wurde [Raibert 86b]. Der Fortschritt besteht in der Trennung von Steuerung, Aktoren und Sensoren.

*Automatisierung*

Der nächste Entwicklungsschritt war, zur Steuerung der Bewegungen von Laufrobotern programmierbare Digitalrechner einzusetzen. Dadurch entstand die Möglichkeit, Bewegungssequenzen automatisch zu erzeugen und zu variieren. Damit wurde eine automatische Anpassung an das Gelände möglich. Erste Vertreter solcher Roboter sind der OSU Hexapod der Ohio State University [Orin 82] und der Laufroboter der Akademie der Wissenschaften der UdSSR [Gurfinkel et al. 81]. Die Steuerrechner waren extern, die Roboter bestanden nur aus der Mechanik, den Sensoren und Aktoren.

*Integration*

Der nächste Entwicklungsschritt ist die Integration des Steuerrechners in den Laufroboter. Die wesentliche Schwierigkeit besteht darin, alle Komponenten so aufeinander abzustimmen, dass die Gesamtfunktion des Roboters realisiert werden kann. Dabei werden verschiedene Teildisziplinen eng miteinander verzahnt. Restriktionen bei der Integration bestehen hauptsächlich in der Begrenztheit von Ressourcen, das heißt beim verfügbaren Platz und der nutzbaren Energie. Das äußert sich beispielsweise in der begrenzten Rechenleistung. Die Steuerungsalgorithmen müssen innerhalb eines vorgegebenen Zeitrahmens bei vertretbarem Aufwand Steuerinformationen liefern. Einer der ersten Laufroboter, in denen die Steuerung integriert wurde, war das Adaptive Suspension Vehicle (ASV) mit einer Gesamtmasse von 2,7 t [Pugh et al. 90], [Waldron, McGhee 86]. In den letzten zehn Jahren erfolgte die Integration von Steuerrechnern auch in kleinere Laufroboter.

*Autonomie*

Der Entwicklungsschritt zur Autonomie ist geprägt von den Fortschritten der einzelnen Teildisziplinen, die zu einer autonomen Funktion des Laufroboters beitragen. Wesentliche Merkmale sind, dass die Energieversorgung integriert ist und die Steuerung des Roboters ohne externe Hilfe erfolgen kann. Dabei können auch komplexe Aufgaben bewältigt werden, die über das bloße Bereitstellen der Lauffunktion hinausgehen. Notwendig dafür sind Fortschritte in der Informationsverarbeitung, der Rechentechnik und Elektronik sowie beim Laufsystem.

## 2.2 Aktuelle Forschungsprojekte sechsbeiniger Laufroboter

International sind in den letzten Jahren viele Projekte initiiert worden. Dabei kann hauptsächlich zwischen biologisch inspirierten und technisch orientierten Designs und Kontrollansätzen unterschieden werden. Bei biologisch inspirierten Designs werden Roboter speziellen natürlichen Vorbildern nachempfunden. Zur Steuerung werden reflexive oder musterbasierte Ansätze verfolgt, die beispielsweise mit Hilfe von neuronalen Netzen realisiert werden [Quinn et al. 01], [Ferrell 95]. Bei technisch orientierten Designs und Kontrollansätzen stehen analytische Lösungen im Vordergrund, beispielsweise zur Generierung von Laufmustern und der Interaktion mit dem Untergrund [Yoneda et al. 97], [Zielinska 01]. Bei einigen Projekten wird versucht, besonders einfache Lösungen zu finden, die mit wenig Freiheitsgraden auskommen, etwa beim Roboter RHex der McGill University [Buehler 01] oder beim Parawalker des Tokyo Institute of Technology [Yoneda 01].

In Deutschland können bei sechsbeinigen Laufrobotern fünf Projekte hervorgehoben werden. Das sind die TUM Walking Machine der TU München, der Laufroboter Katharina des Fraunhofer IFF Magdeburg, die Lauron-Roboter des FZI Karlsruhe, die Tarry-Roboter der Universität Duisburg und der Laufroboter Sir Arthur der GMD. Bei der TUM Walking Machine wurde das Ziel verfolgt, das Steuerungsprinzip und die Kinematik der Stabheuschrecke *Carausius morosus* in einem technischen System umzusetzen [Pfeiffer et al. 95]. Die zugrundeliegenden Steuerungsalgorithmen wurden von der Gruppe um Prof. Cruse in Bielefeld untersucht. Dazu existieren zahlreiche Veröffentlichungen, beispielsweise [Cruse et al. 95], [Cruse et al. 98]. Dieses Projekt orientiert sich am stärksten an einem biologischen Vorbild. Die Laufroboterprojekte zu Lauron, Tarry und Sir Arthur verfolgen biologisch motivierte Zielstellungen. Lauron und Tarry sind in der mechanischen Struktur der Stabheuschrecke ähnlich. Zur Steuerung der drei Roboter werden neuronale Netze genutzt. Beim Lauron-Projekt steht das Lernen von Laufbewegungen mit Methoden des Reinforcement Learning im Vordergrund [Ilg, Berns 95]. Diese Arbeiten wurden beim vierbeinigen Laufroboter Bisam mit dem Ziel fortgesetzt, Laufen online zu erlernen [Ilg 01]. Bei Sir Arthur werden Methoden des hierarchischen Q-Learning untersucht [Kirchner 98]. Beim Duisburger Laufroboter Tarry steht die Nutzung von neuronalen Netzen zur Approximation mechanischer Modelle für die Laufsteuerung im Vordergrund. Dabei werden Feed-Forward-Netzwerke auf Gelenkwinkelmuster trainiert [Frik et al. 99]. Beim Laufroboter Katharina erfolgt eine rein analytisch basierte Steuerung der Bewegungen. Auch die hexagonale Körperform findet keine Entsprechung in der Natur. Biologische Prinzipien finden nur in der Beinkonstruktion und bei der Strukturierung des Steuerungssystems Eingang. Dieses Laufroboterprojekt ist am stärksten technisch motiviert.

## 2.3 Technische Anwendungsgebiete von Laufrobotern

Typisch für technische Anwendungsgebiete ist, dass deterministische Bewegungen gefordert werden, um einen Verlust von Laufrobotern oder Schäden an der Umgebung durch nicht vorhersagbare Bewegungen zu vermeiden. Ein anderer wichtiger Aspekt ist die Möglichkeit den Körper in definierter Weise zu positionieren, um technische Operationen zu unterstützen. Nachfolgend sollen einige denkbare technische Anwendungsgebiete von Laufrobotern näher erläutert werden.

*Havarierte Umgebungen und Katastropheneinsätze*

Der Einsatz mobiler Roboter in Katastrophengebieten und havarierten Umgebungen wurde in der Vergangenheit mehrfach vorgeschlagen [Todd 85]. Der große Bedarf an Rettungsgeräten wird dadurch deutlich, dass Hilfsgeräte den Rettungskräften nur sehr eingeschränkt zur Verfügung stehen, andererseits die Arbeiten sehr gefährlich sind. Zu den havarierten Umgebungen zählen beispielsweise durch chemische und radioaktive Substanzen verseuchte Umgebungen, wie Kraftwerke oder chemische Anlagen, eingestürzte und einsturzgefährdete Bauwerke, schwer zugängliche Brandbereiche oder allgemein Rettungsarbeiten in schwer zugänglichen Umgebungen. Bei allen genannten Bereichen besteht insbesondere das Problem, dass sich die Hilfskräfte selbst einer großen Gefahr aussetzen. Die Schwierigkeit besteht jedoch darin, dass die Umgebung für konventionelles Gerät wegen zerstörter Infrastruktur nicht erreichbar ist oder schwere, geländegängige Fahrzeuge zu groß sind und weitere Schäden ver-

ursachen. In Innenbereichen sind Treppen ein besonderes Problem. In Außenbereichen könnte das Abseilen eine wichtige Technik sein.

Typische Anwendungen bei Havarieeinsätzen sind das Handhaben und Manipulieren von Objekten, z.B. Schließen von Ventilen, das Erkunden und ggf. Vermessen der Umgebung, die Handhabung von Feuerlöschgeräten oder die Vorbereitung von Wegen für weiteres Hilfsgerät. In [Kitano et al. 99] wird auch das Platzieren von Sensoren erwähnt.

Eine sehr wichtige Aufgabe ist das Bergen von Opfern. Für den RoboCup Rescue Wettbewerb wurde deshalb diese Aufgabe als zentrales Element in den Regelentwurf des Wettbewerbs aufgenommen [Robo 01].

*Inspektions- und Wartungsarbeiten*

Der Zielort für Inspektions- und Wartungsarbeiten sollte keine unbekannten Hindernisse aufweisen. Anlagen weisen typischerweise keine Infrastruktur auf, die primär für mobile Roboter vorgesehen ist. Es sollten aber in jedem Fall Zugangsmöglichkeiten für Wartungspersonal vorhanden sein. Damit ist die Einsatzumgebung für Laufroboter in groben Zügen bekannt und besteht aus engen Gängen, Treppen und übersteigbaren Hindernissen. Handhabungsoperationen sind insbesondere Montageoperationen, Platzierung von Sensoren, Probenentnahme oder allgemein das Handhaben von Objekten. Mögliche Einsatzgebiete sind Kernkraftwerke oder Bereiche, in denen mit radioaktiven Isotopen gearbeitet wird. Da Menschen wegen der benötigten Schutzkleidung nur eingeschränkt arbeitsfähig sind, ist der Einsatz von Laufrobotern sehr interessant [Byrd, DeVries 90].

Weitere Einsatzgebiete sind Inspektionsarbeiten an großen Bauwerken und Anlagen. Die Schwierigkeit besteht darin, Messgeräte und Kameras an bestimmte zu inspizierende Stelle zu positionieren. Hierbei rückt das Klettern in den Vordergrund, das dem Laufen sehr verwandt ist. Typisches Einsatzbeispiele sind Tanks, Brücken oder Metallkonstruktionen [Giménez et al. 98]. Der Einsatz von Robotern beispielsweise an Leitungsmasten oder Radarantennen ermöglicht es, Inspektionsarbeiten während des Betriebes der Anlage durchzuführen. Um Havarien und Schäden an den Anlagen zu vermeiden, müssen die Bewegungen des Roboters vorhersagbar und kontrollierbar sein.

Ein weiteres Einsatzgebiet ist die Inspektion von Rohren und Abwasserkanälen. Rohrleitungssysteme können Hindernisse enthalten oder eine rutschige Oberfläche, beispielsweise bei Abwasserleitungen, besitzen. Hindernisse können beispielsweise auch bei Nachinstallationen entstehen, wenn die herausgetrennte Trennwand im Rohr verbleibt. Konventionelle, radgetriebene Inspektionsfahrzeuge sind hier nur begrenzt einsetzbar, bei Steigungen oder senkrechten Rohren gar nicht. Rohrkrabbler können hingegen Hindernisse überwinden und mit Hilfe des Spreizganges die notwendige Andruckkraft erzeugen, um ausreichend Halt zu finden. Die Steuerung eines Rohrkrabblers ist beispielsweise in [Pfeiffer et al. 97] beschrieben.

Auch Kriechen, das dem vielbeinigen Laufen verwandt ist, wurde für die Inspektion von Rohrleitungen untersucht. Ein Beispiel hierfür ist die GMD Snake [Nett, Streich 97].

*Forst und Natur*

Beim Einsatz in Wald- oder Naturbereichen ist der Einsatz von Laufrobotern zunächst dann vorteilhaft, wenn aufgrund der Umgebungsbedingungen Räderfahrzeuge nicht geeignet sind, beispielsweise bei felsigem Untergrund oder an Hanglagen. Der Vorteil gegenüber Räderfahrzeugen besteht in der Möglichkeit der Auswahl geeigneter Fußstellungen, die beispielsweise zwischen Hindernissen liegen können.

Ein weiterer zu beachtender Effekt ist die Wechselwirkung zwischen den Füßen und dem Boden. Durch das Laufen können Querkräfte so minimiert werden, dass die Bodenstruktur nicht zerstört wird. Dadurch werden Wurzeln geschont. Außerdem können die Füße dort platziert werden, wo sie einen guten Halt haben. Durch die Minimierung von Querkräften ergibt sich eine bessere Energieeffizienz als bei Ketten- oder Räderfahrzeugen [Lehtinen 94]. Weitere Vorteile ergeben sich durch die Beweglichkeit der gesamten Maschine, die dadurch Manipulationsarbeiten unterstützen kann.

Beispiele für Laufroboter, die für Waldarbeiten konzipiert wurden, sind Mechant [Halme et al. 97] und der Plustech Walker. Der Plustech Walker wird zum Roden von Bäumen eingesetzt.

*Bau*

Im Baubereich sind Laufmaschinen für die Handhabung schwerer Lasten in unstrukturiertem Gelände interessant. Beispiele hierfür sind der Transport großer Objekte oder Baggerarbeiten. Bei Baggerarbeiten ist es von Interesse, auch an Hanglagen oder in schwierigem Gelände arbeiten zu können. Auch die Möglichkeit des Überwindens größerer Höhenunterschiede kann ein großer Vorteil sein. Für Baggerarbeiten wird derzeit „ALDURO“, ein Prototyp eines Schreitbaggers an der Universität Duisburg entwickelt [Hiller et al. 99][Müller et al. 98].

Andere Anwendungsbereiche sind mit dem Wasserbau oder Unterwasserarbeiten verbunden. Vorteile beim Wasserbau ergeben sich dadurch, dass der Roboter auf dem Untergrund stehen kann, während die Arbeitsplattform über Wasser bleibt. Eine Einsatzmöglichkeit ist der Bau von Küstenschutzanlagen oder von steinernen Hafenmauern. Bei Unterwasserarbeiten besteht oft die Notwendigkeit, größere Unebenheiten des Seebodens auszugleichen und dabei einen sicheren Stand zu gewährleisten. Das gilt auch beim Vorhandensein von Gezeitenströmungen [Todd 85], [Kizono et al. 89].

*Weltraumerkundung und Weltraumtechnologien*

Dieses Anwendungsfeld lässt sich in zwei wesentliche Bereiche einteilen. Das sind Serviceroboter für die Konstruktion, Inspektion und Instandsetzung von Teilen einer Raumstation sowie die Erkundung von Himmelskörpern wie Planeten oder Monde.

Für Arbeiten an Raumstationen haben kletternde oder auf der Oberfläche entlanglaufende Roboter den Vorteil, dass keine komplizierten Flugmanöver notwendig sind, um an einen Zielort zu gelangen. Wegen der fehlenden Gravitation besteht die Notwendigkeit, die Füße in geeigneter Weise zu verankern. Ein Demonstrator, der Verankerungspunkte an einer Konstruktion nutzt, wurde in [Xu et al. 94] vorgestellt. Dabei können die Verankerungspunkte sowohl für die Fortbewegung als zur Verankerung im Manipulationsmodus genutzt werden.

Großer Nutzen wird vom Einsatz von Laufmaschinen zur Erkundung insbesondere von Planeten erwartet. Da die Oberfläche fremder Himmelskörper nicht a priori bekannt ist, kann die Fortbewegung durch Laufen und die damit verbundene Möglichkeit, auch schwieriges Gelände zu durchqueren, von großem Vorteil sein und wird deshalb in zahlreichen Veröffentlichungen genannt, beispielsweise in [Eltze et al. 92], [Martin-Alvarez et al. 94]. In der Vergangenheit wurden bereits verschiedene Konzepte und Prototypen entwickelt. Beispiele hierfür sind der Laufroboter Ambler, der für Marsmissionen der NASA konzipiert wurde [Krotkov et al. 91], [Bares et al. 89], sowie ein Micro Walking Robot, der für die InterMarsnet Mission der ESA konzipiert wurde [Martin-Alvarez et al. 96]. Ein generelles Problem für mobile Roboter ist die von der Erde verschiedene Gravitation. Für sehr kleine Bodenkontaktkräfte, wie sie bei mg- oder µg-Umgebungen, beispielsweise bei Asteroieden oder Kometen, auftreten, verliert die von Bekker bekannte „Theory of land locomotion“ [Bekker 56] ihre Gültigkeit. Unter diesen Bedingungen erlangen Kohäsionseffekte für das Laufen an Bedeutung, um die notwendige Vorschubkraft zu erzeugen [Martin-Alvarez et al. 96].

Alle Fortbewegungsmethoden dienen dem Ziel, wissenschaftliche oder technische Instrumente an einen bestimmten Zielort zu bringen und dort in gewünschter Weise zu positionieren, was sogar für einen kompletten Lander erforderlich sein kann [Martin-Alvarez et al. 96]. Um Verluste und Schäden zu vermeiden, sollten die von Lauf- und Kletterrobotern erzeugten Bewegungen vorhersagbar sein und, sofern es in der konkreten Situation möglich ist, auch kontrollierbar sein.

*Militärische und Anti-Terror-Einsätze*

Der Einsatz von Laufmaschinen für militärische Zwecke ist Gegenstand von Forschungen seit den 60er Jahren. Wichtige Projekte waren der Iron Mule Train und das Adaptive Suspension Vehicle [Todd 85], [Waldron, McGhee 86]. Das Interesse in diesen Projekten gilt hauptsächlich den Transportfähigkeiten, die Laufmaschinen auch in schwierigem Gelände bereitstellen können. Das Ziel ist dabei die Mobilität, die der eines Menschen entsprechen soll. Aktuelle Laufroboterprojekte befassen sich mit kleineren Robotern, die für den Einsatz in wechselnden oder extremen Umgebungen geeignet sind, z.B. der RHex der McGill University Montreal [Moore, Buehler 01] und der Scorpion des Fraunhofer AIS St. Augustin [Spenneberg, Kirchner 00].

Ein weites Einsatzgebiet eröffnet sich bei der Suche und Entschärfung von Landminen und Bomben. Durch Kriege in den letzten Jahrzehnten wurden große Gebiete mit Landminen bestückt. Viele dieser Gebiete sind schwer zugänglich. Für eine humanitäre Entminung muss das Gebiet vollständig geräumt werden, so dass das Suchen und Entschärfen in schwierigem Gelände manuell vorgenommen werden muss. Mit Hilfe von Laufrobotern kann die notwendige Mobilität für technisches Hilfsgerät geschaffen werden, um die Notwendigkeit manueller Arbeiten zu minimieren [Baudoin, Colon 98], [Hirose, Kato 98].

Auch zur Entschärfung von Bomben, beispielsweise in Gebäuden, können Laufroboter eingesetzt werden. Probleme mit Raupenfahrzeugen treten insbesondere bei Treppen auf, wenn der Anfahrweg vor einer Treppe zu kurz ist oder das Fahrzeug auf der Treppe stehen bleiben muss und ein erneutes Anfahren auf der Treppe nicht möglich ist.

Außer der hohen Mobilität als Voraussetzung für den Einsatz von Laufrobotern ist eine wesentliche Forderung, dass Sensoren und Werkzeuge in vorgegebener Weise positioniert und gehandhabt werden können.

*Medizin und Rehabilitation*

Neben der Erforschung des Laufen, das durch Nachahmung mittels technischer Realisierungen, z.B. durch das Laufroboterprojekt der TU München, kann Lauftechnologie auch für medizinische Zwecke und zur Rehabilitation eingesetzt werden. Exoskelette und aktive Prothesen können dazu genutzt werden, Menschen mit Amputationen oder Lähmungen neue Mobilität zu verleihen. Mit Hilfe aktiver Prothesen kann die natürliche Bewegung so weit wie möglich nachgeahmt werden, um ein verbessertes Laufen zu erzielen. Exoskelette oder Orthesen können zum Bewegungstraining eingesetzt werden

*Animatronik und Entertainment*

Die Animatronik beschäftigt sich damit, animierte realistisch wirkende Modelle zu erzeugen. Diese Modelle werden insbesondere zur Produktion von Unterhaltungsfilmen oder zur Animation von Figuren in Unterhaltungsparks genutzt. Die Anforderungen liegen hier hauptsächlich bei einer zuverlässigen Ausführung vorgeplanter Bewegungen, um die gewünschten Effekte zu erzielen und so oft, wie gefordert zu wiederholen.

Das Gebiet der Entertainmentroboter ist noch sehr neu und setzt eine lange Entwicklung von Unterhaltungsautomaten fort. Beispiele hierfür sind die Sony-Hunde [Fujita, Kageyama 97]. In diesem Bereich spielt die Nachahmung von Verhaltensmustern eine große Rolle, um Interaktionsmöglichkeiten zu schaffen.

*Technologienutzung*

Um Laufmaschinen erfolgreich zu steuern, müssen aktuelle Lösungen aus verschiedenen Teilgebieten auf einem Punkt konzentriert werden. Dazu zählen beispielsweise Informatik, Elektronik, Antriebstechnik, Mechanik oder Biologie. Die Konzentration der Teiltechnologien und deren Integration zu einem gesamten komplexen System bringt neue Probleme hervor. Die erfolgreiche Lösung dieser Probleme hat positive Rückwirkungen auf die beteiligten Teilsdisziplinen.

# 3 Grundlegender Aufbau des sechsbeinigen Laufroboters Katharina

Laufmaschinen sollten in der Lage sein, sowohl über ebenes als auch über unebenes Gelände zu laufen, wobei das Gelände nicht bekannt sein muss. Dabei ist es wichtig, dass sie Hindernissen selbständig ausweichen bzw. diese überwinden können. Ferner müssen Laufmaschinen auch Absätze, Stufen und Treppen passieren können. Am Bestimmungsort sollen mit ihrer Hilfe technische Operationen, wie zum Beispiel Bohren oder Montageoperationen, ausgeführt werden.

Um die genannten Problemstellungen zu untersuchen, wurde am Fraunhofer Institut für Fabrikbetrieb und -automatisierung Magdeburg federführend der sechsbeinige Laufroboter Katharina als Prototyp entwickelt. Dieser wurde später für die Implementierung und den Test der Steuerungsalgorithmen am Institut für Fabrikautomatisierung (IFF) und am Institut für Verteilte Systeme der Otto-von-Guericke-Universität Magdeburg genutzt. Der Laufroboterprototyp stellte die mechatronischen Randbedingungen, die für die Implementierung der Steuerungsalgorithmen wichtig sind.

## 3.1 Auswahlkriterien für das Design des Laufroboters Katharina

Für ein sicheres Laufen über unbekanntem Untergrund ist statisch stabiles Laufen eine geeignete Fortbewegungsform, da der Roboter allein aufgrund der Beinkonfiguration zu jedem Zeitpunkt stehen kann. Aktive Stabilisierungsmaßnahmen sind im Gegensatz zur dynamischen Stabilität nicht notwendig.

Bei vierbeinigen Laufrobotern ist statisch stabiles Laufen nur in engen Grenzen möglich, da es aufgrund der diagonalen Anordnung der Beine vorkommt, dass der Masseschwerpunkt am Rand eines Stützpolygons liegt, wenn ein Bein angehoben wird. Falls der Schwerpunkt nicht über dem Stützpolygon liegt oder mehr als ein Bein angehoben werden soll, so kann der Roboter nur mit Hilfe dynamischer Stabilität aufrecht gehalten werden. Bei zweibeinigen Robotern ist kein statisch stabiles Stehen oder Laufen möglich.

Sechsbeinige Laufroboter unterstützen statisch stabile Laufmuster dagegen sehr gut. Dies wird beim Dreifußgang besonders deutlich. Diese Gangart ist das schnellste statisch stabile Laufmuster. Dabei wechseln sich beim Stützen jeweils zwei Beingruppen mit je 3 Beinen ab. Jede der Beingruppen ist in der Lage, ein großes Stützpolygon aufzuspannen, so dass zu jedem Zeitpunkt ein statisch stabiles Stehen möglich ist.

Um eine möglichst sichere Fortbewegung zu gewährleisten, wurde am IFF entschieden, einen Laufroboter mit sechs Beinen zu bauen, da statisch stabile Laufmuster genutzt werden können.

Viele bekannte Realisierungen von sechsbeinigen Laufmaschinen sind biologischen Vorbildern nachempfunden. Ein sehr beliebtes biologisches Vorbild ist die Stabheustrecke, da sie eine einfache Geometrie aufweist und gut untersucht ist [Dean 91]. Bei einigen Realisierungen wird hauptsächlich die Geometrie nachempfunden, so z.B. bei den Robotern des Typs Lauron [Cordes et al. 93], [Cordes et al. 97] oder bei der Laufmaschine Tarry. Bei anderen

wurden auch Steuerungsprinzipien übernommen, wie beispielsweise bei der Laufmaschine MAX der TU München [Pfeiffer et al. 95].

Weiterhin sollte der Roboter über eine gute Manövrierfähigkeit verfügen. Wie sich bei sechsbeinigen Insekten beobachten lässt, besitzt ihr Körperbau eine Vorzugsrichtung, so dass das Vorwärtslaufen sehr gut funktioniert, während andere Richtungen, z.B. seitlich, nur schwer zu bewältigen sind. Für einen Richtungswechsel führen sie deshalb komplizierte Wendemanöver aus. Zur Vermeidung dieses Nachteil muss der Körperbau des Roboters entsprechend angepasst und die Beine dementsprechend angeordnet werden. Verwendet man als Körper eine runde oder regelmäßige polygonale Form, so vermeidet man Vorzugsrichtungen beim Laufen. Beispiele für diese Bauform sind der ODEX I von Odetics Inc. [Byrd, DeVries 90] und der SILEX der Université Libre de Bruxelles [Preumont 94]. Durch eine gezielte aktive seitliche Anordnung der Beine lässt sich eine solche schmale Konfiguration herstellen, dass auch enge Durchbrüche durchquert werden können [Byrd, DeVries 90].

Zur Vermeidung einer Vorzugsrichtung wurde für den Laufroboter Katharina eine hexagonale Körperform gewählt. Die sechs Beine werden an den jeweiligen Seitenflächen des sechsseitigen prismatischen Körpers angebracht.

Zusammenfassend genügt der Laufroboter Katharina folgenden Parametern und Randbedingungen:

- sechs Beine zur Ermöglichung von statisch stabilem Laufen,
- Vermeidung einer Vorzugsrichtung durch Nutzung einer regulär prismatischen Körperform und durch regelmäßige Anordnung der Beine um den Körper,
- Integration von Sensoren,
- Messbarkeit von Interaktionskräften durch Integration von Kraftsensoren,
- Integration des Steuerungssystems in den Roboter und
- Erzeugung definierter Lauf- und Bewegungsmuster zur Vorhersagbarkeit von Verhalten des Roboters und ggf. direkte Manipulierung der Bewegungen.

## 3.2 Mechanischer Aufbau des Laufroboters Katharina

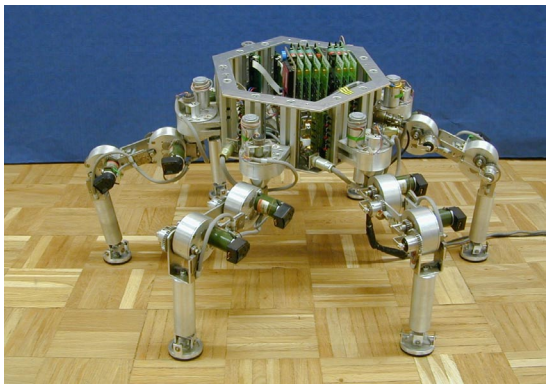


Abbildung 3.1: Gesamtansicht des Laufroboters

Entsprechend der aufgestellten Anforderungen an den zu entwickelnden Laufroboter Katharina, den Abbildung 3.1 zeigt, besitzt dieser einen sechseckigen prismatischen Grundkörper.

Um den Körper sind an den Seitenflächen des prismatischen Grundkörpers sechs Beine mit jeweils drei aktiven Freiheitsgraden angeordnet. Diese Anordnung der Beine bietet den zusätzlichen Vorteil, dass ein großer Bewegungsraum in horizontaler Ebene bei gleichzeitig kompaktem Körper entsteht. Die Beinkonstruktion ist eine Weiterentwicklung der Beine des Roboters Mascha der russischen Akademie der Wissenschaften.

Tabelle 3.1 gibt einen Überblick über die mechanischen Parameter des Roboters. Nachfolgend wird auf einige Details des mechanischen Aufbaus eingegangen, auf die später Bezug genommen wird.

Tabelle 3.1: Mechanische Parameter des Laufroboters Katharina

| **Grundlegende Parameter** | |
|---|---|
| Grundform | Sechskantiger prismatischer Körper |
| Anzahl der Beine | 6 |
| Aktive Freiheitsgrade gesamt | 18 |
| Max. theoretische Geschwindigkeit ca. | 0,4 km/h |
| Gewicht (Basiskonfiguration) | 22 kg |
| Nutzlast ca. | 5 kg |
| **Körper** | |
| Grundkörperabmessungen<br>Höhe ca.<br>Seitenlänge ca. | <br>10 cm<br>17 cm |
| Masse des Körpers mit Mikroprozessoren | 3,8 kg |
| **Beine** | |
| Aktive Freiheitsgrade | 3 |
| Passive Freiheitsgrade (im Fußgelenk) | 3 |
| Arbeitsbereich der Gelenke<br>1. Gelenk<br>2. Gelenk<br>3. Gelenk | <br>-60° .. +60°<br>-30° .. +60°<br>-30° .. +60° |
| Antriebe | Gleichstrommotoren |
| Übersetzungen<br>1. Gelenk<br>2. Gelenk<br>3. Gelenk | <br>1000<br>2000<br>2000 |
| Abmessungen eines Beins gesamt / Segmente | 40,4 cm / $l_0$=7,5 cm, $l_1$=13 cm, $l_2$=20 cm |
| Masse eines Beins ca. | 2,8 kg |
| Fußauflagefläche | 28,3 cm² |
| Sensoren<br>Gelenkwinkelsensoren<br>Geschwindigkeitssensoren<br>Kraftsensoren | <br>Potentiometer auf den Gelenkachsen<br>3 optische Encoder (am Motor)<br>3 orthogonale Komponenten im Unterschenkel |

### 3.2.1 Der Körper des Laufroboters Katharina

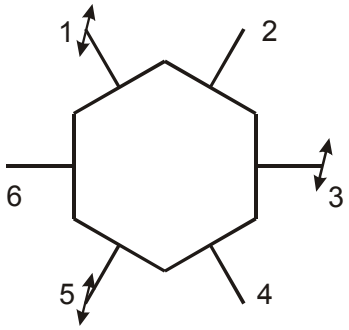



Abbildung 3.2: Roboterform ohne Vorzugsrichtung

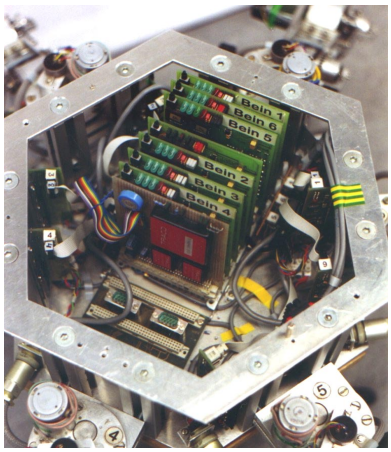



Abbildung 3.3: Der Roboterkörper

Der Grundkörper des Laufroboters Katharina, der schematisch in Abbildung 3.2 dargestellt ist, ist ein reguläres sechsseitiges gerades Prisma. An seinen Seitenflächen werden die Beine befestigt. Ferner ist eine mögliche Laufrichtung für die Beine 1, 3 und 5 angedeutet. Es ist zu erkennen, dass es keine Vorzugsrichtung gibt. Für jedes Bein ist aus lokaler Sicht die Laufrichtung unterschiedlich. Es ist daher wichtig, dass die Laufzyklen so ausgeführt werden, dass sie innerhalb der Arbeitsbereiche aller Beine realisierbar sind.

Das Steuerungssystem wird im prismatischen Roboterkörper untergebracht und auf dessen Grundfläche befestigt, wie in Abbildung 3.3 zu sehen ist. In der Mitte sind die Steuerplatinen mit den Mikrocontrollern und die Stromversorgungsplatine zu erkennen. Die Leistungsverstärker sind nah am zugehörigen Bein angeordnet. Die Deckfläche wird nur als Rahmen ausgeführt, an dem Aufbauten, beispielsweise Werkzeuge oder ein Kamerakopf, befestigt werden können.

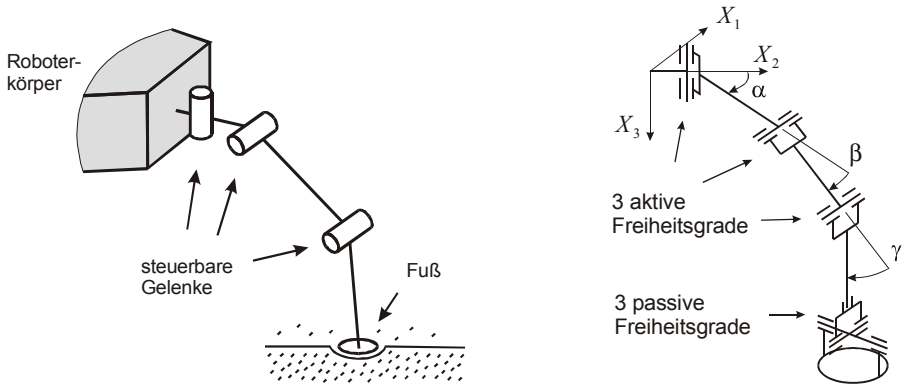
### 3.2.2 Die Beine des Laufroboters Katharina

Mit Hilfe der Beine können sowohl Körper- als auch Beinbewegungen erzeugt werden. Auf die technischen Besonderheiten des Aufbaus der Beine wird im folgenden detailliert eingegangen.

#### 3.2.2.1 Kinematisches Schema der Beine

Zum Erzeugen von räumlichen Trajektorien werden drei aktive Freiheitsgrade für ein Bein benötigt. Damit kann innerhalb des Arbeitsbereiches eines Beines ein Fuß an einem beliebigen Punkt positioniert werden. Zu dessen Realisierung können verschiedene Varianten aus linearen und rotatorischen Freiheitsgraden genutzt werden [Wloka 92]. Beim Laufroboter Katharina kommt eine Verkettung von drei rotatorischen Freiheitsgraden zum Einsatz (Typ

RRR). Das erste Gelenk ist senkrecht zur Ebene der Grundfläche des Roboterkörpers, das erste und zweite Gelenk stehen senkrecht aufeinander und das zweite und dritte Gelenk sind parallel. Die Anordnung der Gelenke zeigt Abbildung 3.4a.




a) Steuerbare Freiheitsgrade b) Kinematisches Schema eines Beines

Abbildung 3.4: Kinematisches Schema der Beine des Laufroboters Katharina

Mit drei Freiheitsgraden ist es nicht möglich, die räumliche Ausrichtung des Fußes zu beeinflussen. Zur Lösung des Problems werden deshalb oft Kugelfüße eingesetzt, die einen punktförmigen Bodenkontakt erzeugen. Beispiele hierfür sind die sechsbeinigen Laufroboter der TU München und des FZI Karlsruhe. Bei der Bewegung eines Beines während der Stützphase entstehen insbesondere bei weichem Untergrund Querkräfte durch die Beinbewegung und der damit verbundenen Veränderung der räumlichen Orientierung des Fußes. Zur Vermeidung dieses Effektes können drei zusätzliche Freiheitsgrade genutzt werden. Damit kann die räumliche Orientierung des Fußes während der Stützphase konstant gehalten werden. Beim Laufroboter Katharina erfolgt das mit Hilfe von drei passiven Freiheitsgraden im Fußgelenk. Es werden ein rotatorisches Gelenk mit einer Achse längs des Unterschenkels und einem Kardangelenk verwendet. Abbildung 3.4b zeigt das komplette kinematische Schema eines Beines für den Laufroboter Katharina.

#### 3.2.2.2 Arbeitsbereich eines Beines

Abbildung 3.5 zeigt den Arbeitsbereich eines Beines. Teilabbildung a) zeigt einen Schnitt durch das verfügbare Arbeitsvolumen in einer Ebene der Achse des Alphagelenkes. Es ist zu erkennen, dass der Arbeitsbereich durch Kreisbögen begrenzt wird, die durch die Rotationsgelenke und die maximal und minimal möglichen Gelenkwinkel entstehen. Die begrenzenden Geraden markieren Grenzen des effektiv nicht nutzbaren Arbeitsbereiches. Die begrenzende Gerade unterhalb des Alphagelenkes kennzeichnet eine Singularität. Das heißt, dass für die Position des Fußes auf dieser Geraden kein eindeutiger Winkel $\alpha$ existiert. Teilabbildung b) zeigt eine Projektion des Arbeitsvolumens auf eine Ebene senkrecht zur Achse des Alphagelenkes. Es ist zu erkennen, dass der Arbeitsbereich unterhalb des Alphagelenkes keine Ausdehnung besitzt. Links der Singularität existiert ein weiterer theoretisch nutzbarer Bereich, der gepunktet angedeutet ist. Es müssten jedoch zur Benutzung dieses Bereiches alle Trajektorien auch die Gerade schneiden, die die Singularität kennzeichnet. Deshalb ist eine Ausdehnung des Arbeitsbereiches über diese Gerade hinaus nicht sinnvoll.

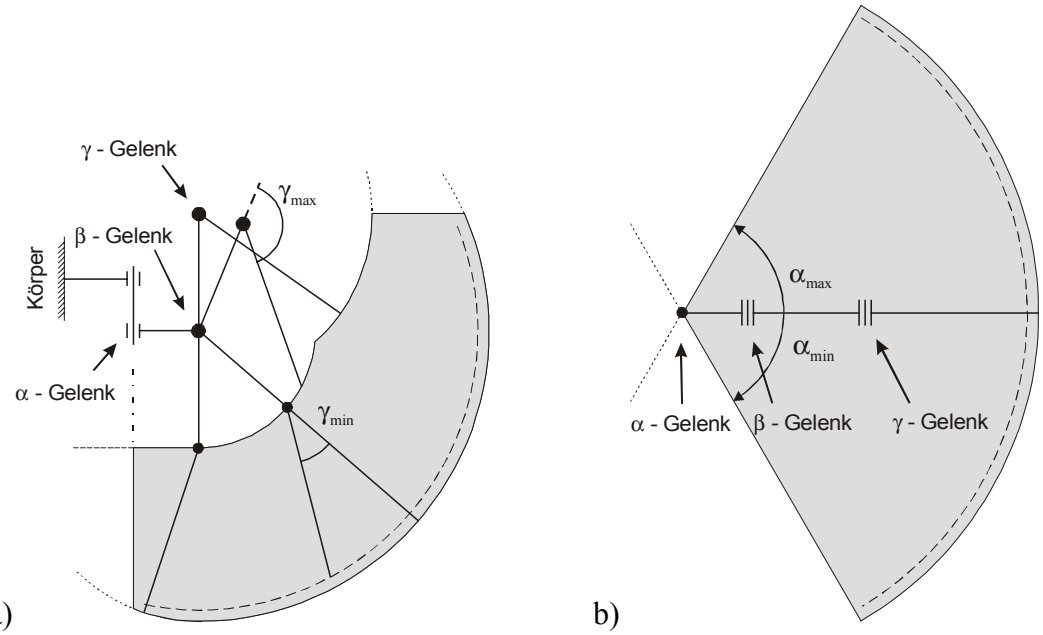



Abbildung 3.5: Arbeitsbereich eines Beines

Neben der durch die mechanische Konstruktion bedingten Begrenzung der Gelenkwinkel ist es auch sinnvoll, eine zusätzliche Begrenzung für den dritten Gelenkwinkel einzuführen, wie in Abbildung 3.5a gezeigt. Bei Bewegungen des Fußes senkrecht zur äußeren Arbeitsbereichsbegrenzung wird eine sehr hohe Bewegungsgeschwindigkeit des dritten Gelenkes in der Nähe der Arbeitsbereichsgrenze erforderlich. Je näher der Fuß der Arbeitsbereichsbegrenzung kommt, desto größer muss die Geschwindigkeit werden. Am Rand des Arbeitsbereiches strebt die erforderliche Gelenkgeschwindigkeit gegen unendlich, da das Gelenk gestreckt ist. Die äußere Begrenzung ist aus geometrischer Sicht ebenfalls singulär. Aus diesem Grund wird der Arbeitsbereich weiter eingegrenzt, was in Abbildung 3.5 durch die gestrichelten Kreisbögen im Arbeitsbereich gekennzeichnet ist.

#### 3.2.2.3 Aktoren und Sensoren

Die Aktoren dienen der Erzeugung der Bewegungen eines Roboterbeines. Bei Laufrobotern mit einer Masse von unter 100 kg werden Elektromotoren eingesetzt, da hydraulische oder pneumatische Antriebsprinzipien für diese Roboterklasse mit autonomen Betrieb nicht geeignet sind. Beim Laufroboter Katharina werden permanenterregte Gleichstrommotoren mit 10 W Leistungsaufnahme verwendet. Sie sind mit Stirnradgetrieben gekoppelt, die eine Untersetzung von 1:1000 bzw. 1:2000 realisieren [Schmucker et al. 00]. Durch die hohe Untersetzung wird die Rückwirkung auf den Motor stark gehemmt. Die mit Hilfe der Motoren beeinflussten Gelenkwinkel müssen mit Hilfe geeigneter Sensoren, z.B. Potentiometer, gemessen werden. Dazu werden die Gelenkwinkelsensoren direkt auf die Gelenkachse montiert. Dadurch kann die Gelenkstellung ohne den störenden Einfluss des Getriebespiels gemessen werden. Abbildung 3.6 zeigt ein Bein des Laufroboters Katharina mit den Motoren und den Gelenkwinkelsensoren.

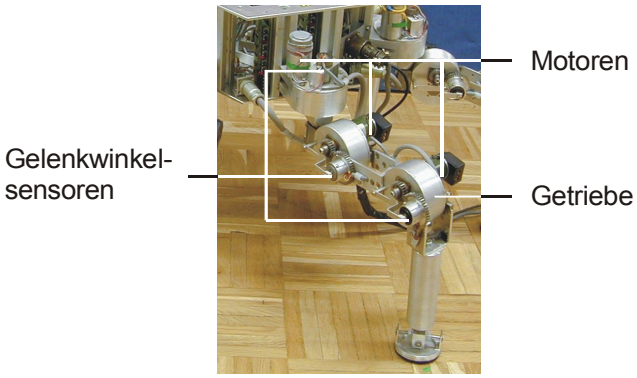



Abbildung 3.6: Einzelnes Roboterbein

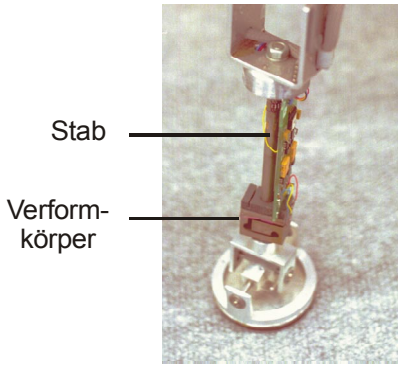



Abbildung 3.7: Kraftsensor im Unterschenkel

Kraftsensoren können zur Erfassung des Bodenkontaktes und von Kollisionen mit Hindernissen genutzt werden. Da die Getriebe beim Laufroboter Katharina stark untersetzen und die Rückwirkung hemmen, können die wirkenden Kräfte nicht über die Antriebsmomente mit Hilfe der Motorströme ermittelt werden. Daher werden bei Katharina Kraftsensoren zur Messung der wirkenden Kräfte eingesetzt. Abbildung 3.7 zeigt den in den Laufroboter Katharina eingebauten Kraftsensor, der Bestandteil der Beinkonstruktion ist. Im Kraftsensor erfolgt die Kraftbestimmung über Dehnmessstreifen (DMS), die die Dehnung an definierten Stellen registrieren. Da der Kraftangriffspunkt bekannt ist, können die Querkräfte mit Hilfe eines Biegebalkens bestimmt werden. Eine am Fuß angreifende Querkraft erzeugt an der Stelle, wo die Dehnung mit Hilfe von Dehnmessstreifen gemessen wird, auf der einen Seite eine Dehnung und auf der anderen Seite eine Stauchung. Die Dehnung ε eines Biegebalkens ist [Heimann et al. 98]

$$\varepsilon = \frac{6 \cdot F \cdot L}{E \cdot B \cdot H^2}. \tag{3.1}$$

Dabei ist $F$ die in der Entfernung $L$ angreifende Kraft. Das Elastizitätsmodul $E$ ist eine Materialkonstante. $B$ ist die Breite des Biegebalkens und $H$ dessen Dicke. Bei nicht quadratischem Querschnitt kann die Breite durch die Querschnittsfläche $A$ ersetzt werden, die aus den geometrischen Beziehungen oft einfacher zu bestimmen ist

$$\varepsilon = \frac{6 \cdot F \cdot L}{E \cdot A \cdot H}. \tag{3.2}$$

Beim Laufroboter Katharina ergibt sich bei einer angreifenden Kraft von 1 N für die Querkomponenten eine Dehnung von $\varepsilon = 20{,}23 \cdot 10^{-6}$.

Zur Erfassung der Längskraft wird ein Verformkörper genutzt. In Abbildung 3.7 ist er am unteren Ende des Biegebalken zu sehen. Abbildung 3.8 zeigt das Funktionsprinzip des Verformkörpers, der die Form eines „G“ besitzt. Wirkt eine Kraft wie eingezeichnet in Längsrichtung, so wird die linke Seite so gebogen, dass an der Oberfläche, wo der Dehnmessstreifen aufgebracht ist, eine Dehnung entsteht. Auf der rechten Seite entsteht eine Stauchung. Dieser gegenläufige Effekt wird zur Verschaltung mehrerer Dehnmessstreifen zu einer Messbrücke genutzt. Treten Querkräfte auf, so treten auf beiden Seiten gleiche Dehnungen bzw. Stauchungen auf, so dass sich diese in einer Messbrücke gegenseitig kompensieren. Bei Querkräften senkrecht zur Bildebene sind die auftretenden Dehnungen sehr klein. Pro

Seite können zwei Dehnmessstreifen aufgebracht werden, die dennoch auftretende Dehnungseffekte in einer Vollbrücke kompensieren können.

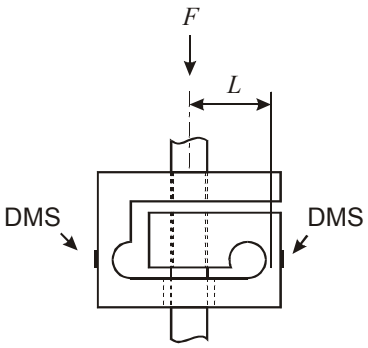



Abbildung 3.8: Verformkörper des Kraftsensors

Zur Bestimmung der auftretenden Dehnung lässt sich auch hier das Prinzip des Biegebalkens anwenden. Die Biegung entsteht durch ein angreifendes Biegemoment an der Stelle, wo nur ein dünner Steg vorhanden ist. In Abbildung 3.8 sind das die Stellen unter den Dehnmessstreifen. Die Kraft $F$ greift entlang des Stabes im Abstand $L$ von der neutralen Faser der Biegestelle an, so dass ein Biegemoment entsteht und auf der Oberfläche eine Dehnung hervorruft. Für diesen Fall wird zur Berechnung der Dehnung Gleichung (3.1) genutzt. Beim Laufroboter Katharina ergibt sich bei einer angreifenden Kraft von 1 N eine Dehnung für die Längskomponente von $\varepsilon = 5{,}33 \cdot 10^{-6}$.

# 4 Konzept eines Steuerungssystems

Ein Steuerungssystem für einen Laufroboter wird so entworfen, dass er mit Hilfe abstrakter Bewegungskommandos steuerbar ist. Dabei können gezielte Körperbewegungen, die Interaktionen mit der Umwelt und Transportmöglichkeiten des Roboters berücksichtigt werden. Dadurch wird es möglich, technische Operationen mit Hilfe des Roboters auszuführen. Die in diesem Abschnitt vorgestellten grundlegenden Konzepte und Ideen lassen sich bei verschiedenen Typen von Laufrobotern nutzen. An gegebenen Stellen wird auf die konkrete Realisierung beim Laufroboter Katharina eingegangen.

## 4.1 Entwurfsprinzip und Strukturierung

Nachfolgend sollen die verschiedenen Möglichkeiten ein Steuerungssystem zu entwerfen betrachtet werden, sowie deren jeweiligen Vor- und Nachteile diskutiert werden.

### 4.1.1 Grundlagen

Laufroboter sind komplexe mechatronische Systeme. Sie unterscheiden sich von üblichen Industrierobotern oder einfachen, radgetriebenen mobilen Robotern durch ihre Komplexität hinsichtlich Mechanik, Sensorik, Aktorik und Informationsverarbeitung. Ferner sollen Laufroboter über Autonomie verfügen, was nicht nur hinsichtlich des Bewegungsapparates in Form von Mechanik, Sensoren und Autoren zu realisieren ist, sondern auch durch die Integration der Energieversorgung und des alle Bewegungen kontrollierenden Steuerungssystems. Daher ergeben sich hinsichtlich der Rechenleistung und der zur Verfügung stehenden Energie Beschränkungen. Es muss beim Entwurf des Steuerungssystems darauf geachtet werden, dass Hardware, Steuerungsalgorithmen und deren Implementierung so aufeinander abgestimmt sind, dass deren Funktion trotz der bestehenden Beschränkung den Echtzeitanforderungen des technischen Systems genügt. Folgende Anforderungen ergeben sich aus der Sicht der technischen Realisierung:

- große Anzahl von Ein- und Ausgabekanälen (Sensoren, Aktoren),
- Signalauswertung von Sensorsignalen,
- Formung von Steuersignalen,
- Einbettung der Steuerung in den Roboter,
- Koordinierung komplexer Bewegungen, insbesondere solche mit speziellem technischem Charakter und
- Echtzeitfähigkeit.

Das zentrale Problem ist die Integration der Steuerung in den Roboter. In früheren Arbeiten ist dieses Problem dadurch umgangen worden, dass eine externe Steuerung zusammen mit Baugruppen der Motorsteuerung und der Messwertverarbeitung genutzt wurde. Stellvertretend sei das Laufroboterprojekt an der Akademie der Wissenschaften der UdSSR genannt, wo ein Digitalrechner NOVA 2/10 kombiniert mit drei Analogrechnern der Typen MN-17 und MN-14 zur Berechnung der Steuersignale genutzt wurde [Gurfinkel et al. 81]. Ein anderes

Beispiel ist der OSU Hexapod, wo die Steuerung mit Hilfe einer PDP11/45 realisiert und später durch fünf LSI-11-Rechner erweitert wurde [Orin 82], [Klein, Wahawisan 82]. In den achtziger Jahren wurden Steuerrechner in größere Laufmaschinen integriert, beispielsweise in das Adaptive Suspension Vehicle [Pugh et al. 90], wo eine ganze Reihe von allgemein nutzbaren (14 Intel 86/30 SBC, 1 Intel 386/21 SBC) und hoch spezialisierten Mikrorechnern integriert worden sind. Allerdings handelte es sich hier um einen Roboter mit 2,7 t Gesamtmasse [Waldron, McGhee 86], wo dafür ausreichend Platz und Tragfähigkeit vorhanden ist. In den Laufroboter MECHANT I mit 1100 kg wurde ein Rechner basierend auf Intel 486 und sechs Rechner basierend auf Intel 286 eingesetzt [Halme, Hartikainen 96].

Auch heute werden zu Forschungszwecken externe Steuerrechner genutzt. Dies erfolgt hauptsächlich, um Steuerungsalgorithmen zu testen. Dabei wird der Vorteil der Entwicklungs- und Testumgebung auf einem externen Rechner genutzt. Gleichzeitig stellen moderne Hochleistungs-PC's so viel Rechenleistung bereit, dass Steuerungsalgorithmen ohne nennenswerte zeitliche Beschränkungen getestet werden können. Ein Beispiel hierfür ist das Projekt der Laufmaschine Tarry der Universität Duisburg. Dabei sind die Steuerelektronik und der Steuerrechner extern, die Maschine selbst besteht nur aus Mechanik, Antrieben und Sensoren. [Frik et al. 98].

In den letzten Jahren ist dazu übergegangen worden, die Steuerrechner selbst auch in kleinere Roboter zu integrieren, die somit zu eingebetteten Systemen werden. Hervorzuheben sind hier Roboter der 20 kg-Klasse. Dazu zählen beispielsweise neben Katharina [Schmucker et al. 96b] die TUM Walking Machine der TU München [Pfeiffer et al. 95] und Lauron I und II des FZI Karlsruhe [Cordes et al. 93], [Cordes et al. 97].

Die Rechnerarchitektur der Laufmaschinen ist sehr stark auf die funktionale Struktur der Maschine und der zugrundeliegenden Steuerungsalgorithmen abgestimmt. Dabei ist es von Bedeutung, wie die Funktion der Maschine abstrahiert wird und die Steuerungsalgorithmen eingebettet werden.

Viele Arbeiten zu Laufmaschinen lehnen sich beim Entwurf der Steuerung sehr stark an biologische Vorbilder an. In der Regel ist die Mechanik der Laufmaschinen den anatomischen Merkmalen der Vorbilder nachempfunden. Ein gut untersuchtes und deshalb genutztes biologisches Vorbild ist die Stabheuschrecke [Ferrell 95], [Cruse et al. 95], [Pfeiffer et al. 95]. Bei ihr ist die Steuerung der Beine dezentral organisiert. Jedes Bein wird durch einen Nervenknoten gesteuert und ist für sich genommen autonom. Zur Koordinierung der Beine untereinander können die Beine mit ihren Nachbarn kommunizieren, was durch die leiterförmige Struktur des Nervensystems unterstützt wird. Dabei reagieren die Beine auf Reize der eigenen Sensoren und der Zustände der jeweiligen Nachbarbeine [Dean 91], [Cruse et al. 95]. Dadurch entsteht ein reaktives Laufen. Bei höher entwickelten Lebewesen ist das Prinzip der Mehrebenensteuerung zu finden. Dabei wird von einer höheren Instanz das Bewegungsmuster vorgegeben, die konkrete Realisierung erfolgt von tiefer liegenden Steuerschichten [Bernstein 88]. Eine tiefer gelegene Steuerungsebene ist dabei in der Lage, die Sensorinformationen schnell in Reaktionen umzusetzen, während höhere Steuerungsebenen langsamer reagieren, jedoch komplexere Operationen ermöglichen. Diese komplexen Operationen sind für technische Anwendungen interessant, bei denen es auf gezielte aktive Bewegungen ankommt. Dieses Ziel lässt sich mit einem hierarchischen Konzept verfolgen, bei dem abstraktere Bewegungen durch eine höhere Steuerungsebene realisiert werden, während untere Steuerungsebenen weniger abstrakte Bewegungen, z.B. Gelenkpositionierung, realisieren. Solch ein hierarchisches Konzept lässt sich durch weitere Ebenen ergänzen, wie sie für mobile Ro-

boter bekannt sind, z.B. Navigation und Missionsplanung. Die Laufsteuerung realisiert dann die abstrakten Bewegungskommandos der höheren Ebenen.

Für das Design eines Steuerungssystems mit hierarchischem Ansatz können zwei entgegengesetzte Methoden genutzt werden. Diese sind der Top-Down-Entwurf und der Bottom-Up-Entwurf. Bei der ersten Methode erfolgt eine Verfeinerung der Struktur beginnend mit der obersten Steuerungsebene. Das gestattet die generelle Sicht auf die Funktion des abstrakten Objekts Roboter, der schrittweise in funktionelle Elemente zerlegt wird. Dieser Ansatz führt möglicherweise nicht direkt zu einer Struktur mit den verwendeten mechatronischen Komponenten. Beim Bottom-Up-Entwurf wird des Gesamtsystems beginnend bei seinen Basisfunktionen, die durch die Sensoren und Aktoren gegeben sind, zusammengesetzt. Darauf aufbauend kann auf mechatronische Komponenten und Steueralgorithmen zurückgegriffen werden. Der Vorteil dieser Methode liegt in der Abbildung der Hardwarestruktur in eine entsprechende Softwarestruktur. Da beide Konzepte an ihren Startpunkten ihre Stärken haben, liegt es nahe, beide Methoden zu kombinieren [rti 99].

### 4.1.2 Top-Down-Entwurf

Aus der Sicht des Top-Down-Entwurfes führt ein abstraktes Bewegungskommando zu einer bestimmten Roboterbewegung. Für größere Distanzen erfolgt die Fortbewegung eines Laufroboters mittels Laufen. Dabei sollen Bodenunebenheiten durch die Laufbewegungen gefiltert werden. Die Fußpunkte der Beine müssen definierte Bewegungen gegenüber der Umwelt ausführen. Für eine Feinpositionierung des Körpers genügt hingegen eine Körperbewegung bei feststehenden Füßen. Da Instrumente und Werkzeuge bevorzugt am Körper befestigt werden, müssen die Körperbewegungen in bezug auf die Körperachsen erfolgen. Aus den beabsichtigten Körperpositionen müssen die dazu erforderlichen Beinbewegungen abgeleitet werden. In beiden Fällen müssen anschließend konkrete Bein- und Gelenkstellungen abgeleitet werden.

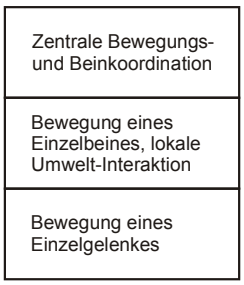



Abbildung 4.1: Hierarchische Strukturierung nach dem Top-Down-Entwurf

Der Ansatz des hierarchischen Steuerungssystems ist für den Top-Down-Ansatz sehr geeignet. Bis grundlegende Basisfunktionen vorliegen, werden ausgehend von einem zentralen, bewegungskoordinierenden Element schrittweise Funktionsmodule erzeugt. Konkret auf das Steuerungssystem für Laufroboter bezogen bedeutet das, dass ausgehend vom zentralen bewegungskoordinierenden Element die Beinbewegungen abgeleitet werden müssen, und daraus wiederum die Gelenkbewegungen. Abbildung 4.1 zeigt die entsprechende Ebenenstruktur. Als Ziel bei der Top-Down-Verfeinerung stehen Basisfunktionen zur Steuerung der mechatronischen Komponenten bereit.

### 4.1.3 Bottom-Up-Entwurf

Beginnend mit der kleinsten sinnvollen Einheit, dem Einzelgelenk, können Abstraktionen in der Steuerungshierarchie eingeführt werden, bis der gesamte Roboter als ein abstraktes Objekt betrachtet wird. Jedes Einzelgelenk für sich ist bereits ein komplexes Sensor-Aktor-System. Einzelne Gelenke sind mechanisch mit den Nachbargelenken oder dem Körper verbunden. Eine solche kinematische Kette wird als Bein bezeichnet. Die Geometrie der kinematischen Kette bzw. der Zusammenhang zwischen Gelenkpositionen und Beinposition wird mit Hilfe der direkten kinematischen Transformation (DKT) und der inversen kinematischen Transformation (IKT) beschrieben. Jedes Bein agiert wie ein kleiner, eigenständiger Manipulator („Roboterarm“). Hierbei gelten prinzipiell die gleichen Regeln wie für die Positionierung von Knickarmrobotern, die eine entsprechende Kinematik besitzen. Die Positionierungskommandos für den Endpunkt der kinematischen Kette (Fuß) werden von höherer Ebene vorgegeben.

Der gesamte Laufroboter lässt sich aus den sechs Beinen und dem Körper zusammensetzen. Da mehrere gleiche Komponenten zusammengesetzt werden, kann neben der vertikalen Strukturierung nach Abstraktionsebenen auch horizontal entsprechend der Teilsysteme unterteilt werden. Auf diese Weise können weitere Komponenten auf gleichem Abstraktionsniveau ergänzt werden. Dazu zählen beispielsweise ein Manipulator oder ein Kamerakopf. Das Prinzip der vertikalen und horizontalen Strukturierung verdeutlicht Abbildung 4.2. Der dick markierte Pfad kennzeichnet das Zusammenfügen dreier Gelenke zu einem Bein, welches Teil des Roboters ist.

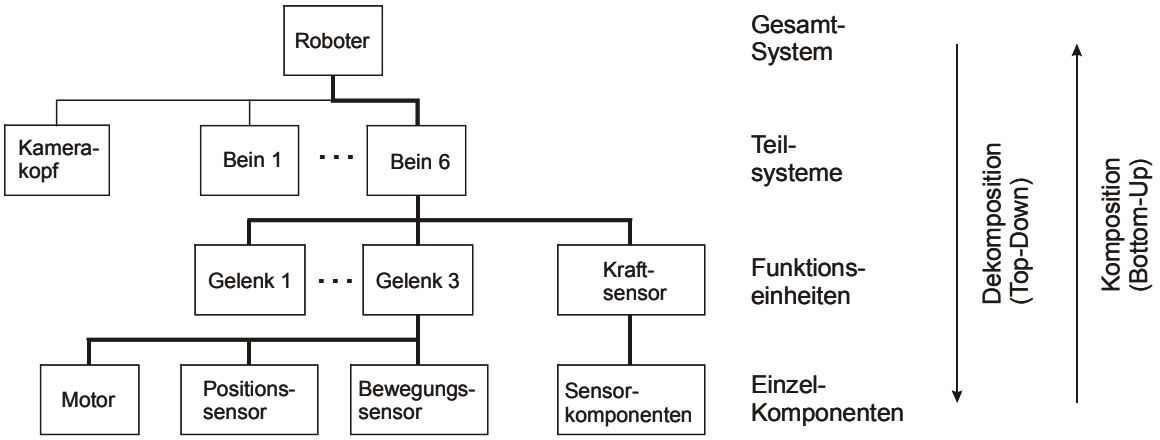



Abbildung 4.2: Vertikale und horizontale Strukturierung, entstanden durch ein Bottom-Up-Design

### 4.1.4 Synthese der Steuerungsstruktur durch Anwendung beider Prinzipien

Im Bottom-Up-Entwurf ist die klare Strukturierung eines Roboters nach Funktionseinheiten und Teilsystemen zu erkennen, die zusammen das Gesamtsystem Roboter ergeben. Die Funktion des komplexen Systems Laufroboter ergibt sich aber nicht durch die bloße Zusammenstellung der Komponenten, da die zusammenhängende Funktionsweise der einzelnen Komponenten und die Beziehungen dieser untereinander nicht unmittelbar aus dem Entwurf

ableitbar sind. Umgekehrt ist durch eine Betrachtung eines abstrakten Roboters im Top-Down-Entwurf nicht unmittelbar ersichtlich, wie die Realisierung der Laufbewegungen basierend auf existierende mechatronische Komponenten erfolgen könnte. Daher ist es sinnvoll, beide Ansätze in geeigneter Weise zu kombinieren. Nach dem Bottom-Up-Verfahren werden die mechatronischen Basisfunktionen identifiziert und entsprechend der gegebenen mechanischen Struktur zusammengesetzt. Dadurch entsteht eine Modellstruktur, die für den Top-Down-Entwurf genutzt wird. Diese Modellstruktur beschreibt die Basisfunktionen, die als Ziel für den Top-Down-Entwurf genutzt werden. Somit wird sichergestellt, dass die entstehende Struktur zum mechatronischen System passt.

Am Beispiel des Laufroboters Katharina soll im folgenden die Kombination beider Entwurfsverfahren erläutert werden, die sowohl für das Laufen des Roboters, als auch für seine Körperpositionierung angewendet werden kann. Im Bottom-Up-Verfahren ist es unter den gegebenen mechatronischen Voraussetzungen möglich, einem Gelenk einen Gelenkwinkel vorzugeben. Durch die Zusammensetzung eines Beines aus drei Gelenken kann damit eine Beinposition vorgegeben werden. Soll eine Laufbewegung bzw. eine Körperpositionierung mittels Top-Down-Entwurf realisiert werden, so wird das Laufen bzw. die Körperposition durch Verfeinerung zunächst in Funktionen für die Beinbewegung zerlegt. Anschließend werden diese in Funktionen für die Gelenkgeschwindigkeiten aufgesplittet. Die Vorgabe einer Gelenkgeschwindigkeit ist mit dem vorhandenen mechatronischem System jedoch nicht möglich. Sind hingegen mechatronische Basisfunktionen bzw. abstrakte Funktionen aus dem Bottom-Up-Entwurf bekannt, so ist es beim Top-Down-Entwurf möglich, gezielt auf diese Basisfunktionen zuzuarbeiten. Abbildung 4.3 verdeutlicht diese Vorgehensweise. Da keine Gelenkgeschwindigkeiten, sondern nur Gelenkwinkel als Vorgabe möglich sind, müssen alle Bewegungen mit Hilfe einer Folge von Positionierungen beschrieben werden.

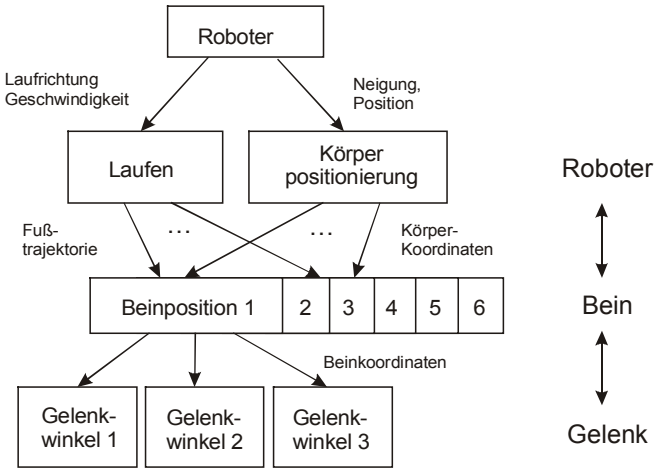



Abbildung 4.3: Dekomposition der Roboterbewegung

Beachtet man beim Top-Down-Entwurf die zusammensetzbaren Basisfunktionen, die man mit dem Bottom-Up-Entwurf realisieren kann, so ist gesichert, dass einerseits die Zusammenhänge des Gesamtsystems beachtet werden, andererseits auf technisch sinnvolle Basisfunktionen zurückgegriffen wird. Bei der Kombination beider Verfahren für den Entwurf des Steuerungssystems für den Laufroboter Katharina lassen sich die Abstraktionseben

- Gesamtsystem: Roboter;
- Teilsysteme: Beine, Manipulator;
- Funktionseinheiten: Gelenke, Sensoren und
- Einzelkomponenten: Einzelsensoren, Sensorkomponenten, Aktoren

aufstellen. Tabelle 4.1 zeigt die Zuordnung verschiedener Operationen und Informationen zu den Abstraktionsebenen.

Tabelle 4.1: Informationen und Operationen auf der jeweiligen Abstraktionsebene

| Abstraktion | Operation | Information |
|---|---|---|
| Roboter | Körperbewegung und –positionierung, Laufmuster | Gewichtsvektor, Massenschwerpunkt, Körperposition, Bodenneigung, allgemeiner Kraft- und Momentemnvektor |
| Bein, Manipulator, Kamerakopf | Beinpositionierung und –bewegung, entsprechend für Manipulator und Kamerakopf, aktive Nachgiebigkeit | Beinposition, lokale Untergrundeigenschaften (Plastizität, Elastizität, Reliefhöhe) |
| Gelenk, Sensor | Gelenkbewegung und –position | Gelenkposition und –geschwindigkeit, Kräfte |
| Aktor, Sensorkomponente | Signalformung zur Aktoransteuerung, Sensorsignalverarbeitung | Primäre Sensordaten, Stellwerte für Aktoren |

## 4.2 Rechnerarchitektur und Einbettung

Bei der Konstruktion eines Steuerungssystems für einen autonomen Laufroboter müssen folgende Aufgaben gelöst werden. Dies sind

- Steuerung der realen Maschine,
- Realisierung der Peripherieanbindung,
- Integration der Rechnerhardware und aller weiteren notwendigen Komponenten in die Maschine und
- Minimierung des Energiebedarfs.

Bei mehreren dieser Punkte sind Kompromisse einzugehen. Der Hauptkompromiss ist bei der Größe und Universalität der in den Roboter zu integrierenden Steuerungshardware einzugehen. Kommerziell verfügbare Standardkomponenten bzw. Baugruppen können in der Regel nicht oder nur sehr eingeschränkt genutzt werden. Auf die letzten drei Aufgaben soll nachfolgend näher eingegangen werden.

*Realisierung der Peripherieanbindung*

Peripherieelemente bei einem Laufroboter sind Sensoren, Aktoren und Kommunikationseinheiten. Da ein Laufroboter ein eingebettetes System ist, sind Bedienelemente, wie z.B. Anzeige oder Tasten, primär nicht vorgesehen.

Bei der Integration von Sensoren und Aktoren ist zu beachten, wie die von den Sensoren gelieferten Signale für die rechnerinterne Informationsverarbeitung aufbereitet werden müssen und welche physikalischen Signale aus den rechnerintern repräsentierten Informationen zur Ansteuerung der Aktoren generiert werden müssen. Da die Steuerung in den Roboter zu integrieren ist, können keine handelsüblichen Baugruppen für Messwertverstärker, Analog-Digital-Umsetzer bzw. Digital-Analog-Umsetzer oder Leistungsverstärker zur Ansteuerung

von Aktoren genutzt werden. Es ist eine Lösung zu finden, bei der die Sensorsignale mit möglichst geringem Aufwand zur Weiterverarbeitung in den Rechner gelangen. Umgekehrt müssen die Signale zur Aktorenansteuerung (Gleichstrommotoren) mit möglichst geringem Aufwand erzeugt und verstärkt werden.

*Integration der Rechnerhardware und aller weiteren notwendigen Komponenten in die Maschine*

Neben der Erbringung der Rechenleistung ist es in eingebetteten Systemen sehr wichtig, mit peripheren Komponenten über verschiedene elektrische Signale Informationen auszutauschen. Um geeignete Peripheriesignale zu produzieren bzw. auszuwerten, ist es möglich, sogenannte Mikrocontroller zu nutzen. Diese enthalten bereits eine Reihe von Peripherieschaltungen und unterstützen somit ein einfaches Design des Steuerungssystems. Die Auswahl eines geeigneten Mikrocontrollers ist somit nicht allein auf die Rechenleistung beschränkt, sondern es fließen auch die verfügbaren Peripheriekomponenten mit ein.

*Minimierung des Energiebedarfs*

Energie ist bei autonomen Systemen eine begrenzte Ressource. Deshalb muss Energie so sparsam wie möglich eingesetzt werden. Hauptsächlich wird bei einem Laufroboter Energie durch die Antriebe, die Leistungsverstärker und die Rechnerhardware genutzt. Dabei ist zu beachten, dass moderne Hochleistungsprozessoren heute zunehmend mehr Energie benötigen. Dieser Trend ist im wesentlichen auf die Instruction Set Architecture (ISA), die Taktfrequenz und die Prozessorspeicher zurückzuführen. Bei der ISA wird die Verarbeitungsgeschwindigkeit dadurch erhöht, dass Rechenfunktionen in der Hardware durch komplexe Logikschaltungen implementiert werden, was die Komplexität der arithmetisch-logischen Einheit (ALU) vergrößert und sich in einer erhöhten Gatteranzahl ausdrückt. Gleichzeitig wird versucht, den Durchsatz durch Parallelverarbeitung zu erhöhen. Zum Einsatz kommen beispielsweise Pipelining und SIMD-Techniken. In diesen Fällen wird das Chip-Design noch komplexer. Integrierte Caches leisten hier einen weiteren Beitrag. Durch Erhöhung der Taktfrequenz steigt der Energiebedarf bedingt durch CMOS-typische Effekte ebenfalls an [Kühn 86], [Hertzsch 86]. Das ist ein wichtiger Grund, zwischen Leistungsfähigkeit des Prozessors und dem vertretbaren Energieaufwand abzuwägen. Darum wird ein Designkompromiss notwendig.

Bei einigen Prozessoren, insbesondere bei Mikrocontrollern, besteht die Möglichkeit, inaktive Phasen mittels Power-Down-Techniken in einem Energiesparmodus zu überbrücken. Um die Energiesparmöglichkeiten auszunutzen, dürfen Steuerungsalgorithmen nur so oft ausgeführt werden, wie es für die Applikation notwendig ist, was durch geeignete Schedulingmethoden erreicht werden kann. Ein weiterer Effekt ist die dadurch verbesserte Performance eines Prozessors, so dass sich möglicherweise die Prozessorenanzahl bei einem Mehrprozessorsystem verringern oder einfachere Prozessoren verwenden lassen. Dadurch ergibt sich ein Energiespareffekt.

Leistungsverstärker und Antriebe bilden eine Einheit. Die elektrische Energie soll möglichst effektiv in Bewegungsenergie umgesetzt werden. Dabei sollen möglichst wenig Energieverluste in den Leistungsverstärkern auftreten. Mit Hilfe algorithmischer und elektrotechnischer Methoden muss die Verlustleistung in den Leistungsverstärkern gering gehalten werden und die Bewegungen so energiearm und effektiv wie möglich ausgeführt werden.

### 4.2.1 Rechnerarchitektur

Zur Steuerung eines Roboters muss eine geeignete Rechnerarchitektur zur Abarbeitung der Steuerungsalgorithmen zur Verfügung gestellt werden. Dabei ist es wichtig, die bisher genannten Randbedingungen zu erfüllen. Darüber hinaus sollten folgende Überlegungen einfließen:

- Lassen sich nebenläufige Prozesse identifizieren? Können diese Prozesse räumlich getrennt werden?
- Besteht das Ziel, relativ selbständige Subeinheiten zu erhalten (z.B. Beinsteuerung)?
- Ist eine Unterteilung in Teilsysteme für ein fehlertolerantes Verhalten sinnvoll?

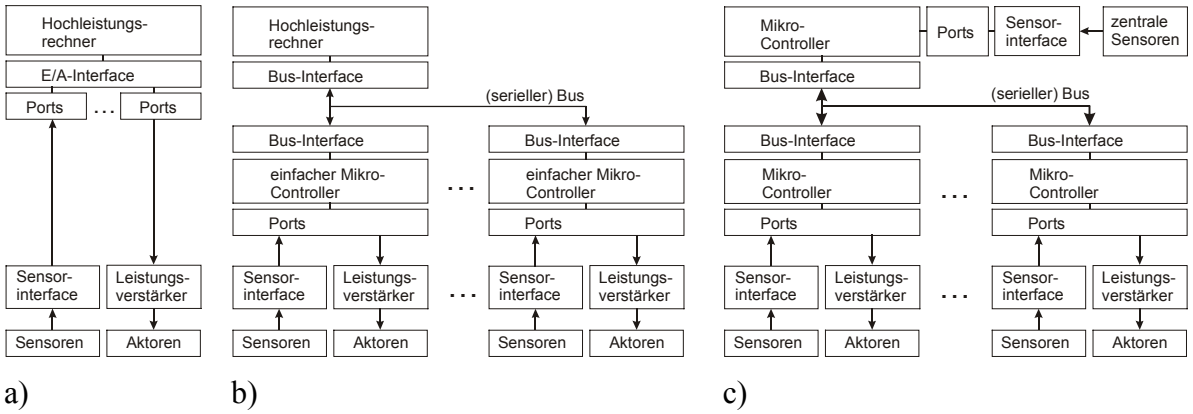



Abbildung 4.4: Verschiedene Rechnerarchitekturen zur Steuerung eines komplexen Sensor-Aktor-Systems

Zur Realisierung einer Rechnerarchitektur, die das hierarchische Konzept unterstützt, lassen sich drei verschiedene Grundvarianten unterscheiden, die in
Abbildung 4.4 dargestellt sind. Diese sind

1. zentraler Hochleistungsrechner mit einer speziellen Ein- und Ausgabeeinheit,
2. zentraler Hochleistungsrechner mit intelligenten Ein- und Ausgabeeinheiten, die die Signalformung und die Sensorinformationsvorverarbeitung übernehmen und
3. Verteilung der Steuerungsalgorithmen auf mehrere kleine Recheneinheiten, die untereinander kommunizieren und eigene Ein- und Ausgabefunktionalität besitzen.

Zentrale Hochleistungsrechner mit einer speziellen Ein- und Ausgabeeinheit sind von früheren und aktuellen Laufmaschinenprojekten z.B. [Frik et al. 98] und heute üblichen Industrierobotersteuerungen bekannt. Es wird ein zentraler Rechner genutzt, auf dem sämtliche Steueralgorithmen online abgearbeitet werden, die den Echtzeitanforderungen gerecht werden müssen. Die Hardware für solch einen Zentralrechner beinhaltet in der Regel einen Hochleistungsprozessor mit entsprechender Standardhardware. Zur Kopplung des Rechners mit den Sensoren und Aktoren ist ein Interface notwendig. Die Bereitstellung der erforderlichen Ein- und Ausgänge ist oft mit erheblichem Aufwand verbunden. Bei üblichen Industriesteuerungen besitzen Ein- und Ausgabeeinheiten nicht sehr viele Kanäle, so dass eine entsprechende Anzahl solcher Baugruppen notwendig ist.

Für den Laufroboter Katharina, bei dem die Sensorsignale in analoger Form vorliegen, ergibt sich eine Konfiguration der Ein- und Ausgänge, wie sie in Tabelle 4.2 dargestellt ist.

Tabelle 4.2: Konfiguration der Ein- und Ausgänge für den Laufroboter Katharina

| | Eingänge | Ausgänge |
|---|---|---|
| Antriebe | | 18 |
| Positionssensoren Gelenke | 18 | |
| Kraftkomponenten | 18 | |
| Gesamt | 36 | 18 |

In der Beispielkonfiguration müssen bereits 36 Eingangskanäle verarbeitet werden. Bei einem weiteren Ausbau mit Sensoren bzw. Aktoren vergrößert sich die Anzahl der benötigten Ein- und Ausgänge. So würden beispielsweise noch einmal 18 x 2 Kanäle für Encoder oder weitere Ein- und Ausgänge für einen Kamerakopf hinzukommen. Der Aufwand zur Bereitstellung einer ausreichenden Anzahl von Ein- und Ausgabe-Ports bzw. Analog-Digital- und Digital-Analog-Umsetzer am einzelnen Rechner ist hoch. Ferner ist zu beachten, dass das Interface nicht skalierbar ist.

Neben der Bereitstellung der Rechnerhardware und der Ein- und Ausgabeeinheiten selbst besteht das Problem, diese auch auf dem Roboter unterzubringen. Aufgrund des Platzproblems scheiden in der Regel übliche industrietypische Baugruppen aus. Ein Ausweg bietet sich durch den Einsatz eines speziellen, konfigurierbaren Sensor-Aktor-Interfaces, beispielsweise durch Einsatz eines FPGA (Field Programmable Gate Array).

Eine Konfiguration mit einem zentralen Hochleistungsrechner mit einer speziellen Ein- und Ausgabeeinheit ist wegen der mechanischen Ausmaße, des Energiebedarfs und der komplizierten Kopplung mit der Peripherie (Sensoren, Aktoren) nur für Laborzwecke mit einem externen Steuerrechner geeignet. Lange Zuleitungen zum Roboter bei externen Steuerrechnern können zusätzliche Probleme bewirken.

Bei der Nutzung eines zentralen Hochleistungsrechners mit intelligenten Ein- und Ausgabeeinheiten, die die Signalformung und die Sensorinformationsvorverarbeitung übernehmen, wird speziell das Problem des Ein- und Ausgabeinterfaces betrachtet. Hier wird das einzelne komplexe Interface durch eine Reihe von kleinen, passiven oder durch einfache Mikrocontroller gesteuerte Interfaces ersetzt. Der Vorteil bei dieser Variante besteht darin, dass das Einlesen der Sensorinformationen bzw. die Ausgabe der Steuersignale von den Mikrocontrollern übernommen werden kann. Diese besitzen integrierte Peripherieeinheiten, z.B. digitale Ein- und Ausgänge oder Analog-Digital-Umsetzer. Dadurch kann die Hardware sehr kompakt aufgebaut werden. Die Signalformung der Ausgangssignale bzw. die Ansteuerung der Analog-Digital-Umsetzer kann relativ einfach implementiert werden, da diese Mikrocontroller keine eigenen Steuerfunktionen zu übernehmen haben. Ihre Aufgabe besteht lediglich in der Ein- und Ausgabe sowie der Datenübermittlung über den Bus. Durch den Einsatz des Busses wird das Interface skalierbar. Das heißt, dass ohne großen Aufwand weitere Interfaces hinzugefügt werden können. Zusätzliche Interfaces könnten einen Kamerakopf oder einen Manipulator steuern. Die Grenze der Skalierbarkeit liegt beim Bus und den zu übertragenden Informationsmengen. Da die Mikrocontroller keine eigene Steuerungs- oder Regelungsfunktionen übernehmen, müssen diese vom Zentralrechner bereitgestellt werden. Dadurch müssen die Ein- und Ausgabedaten mit einer der Steuerungs- bzw. Regelungsperiode entsprechenden Updatefrequenz übertragen werden. Innerhalb der Steuerungsperiode sollten keine nennenswerten Verzögerungen auftreten, damit eine Reaktion rechtzeitig erfolgen kann bzw. sich dadurch das Systemverhalten nicht verändert. Bei einer durch viele Stationen und hohen Updatefrequenzen bedingten Buslast kann die Busbandbreite schnell zur begrenzenden Ressource werden. Bei dieser Variante bleibt das Problem des Energiebedarfes erhalten, da der

als Hochleistungsprozessor ausgelegte Zentralrechner aufgrund seiner Architektur und der hohen Taktung einen hohen Energiebedarf hat. Dieser hohe Energiebedarf ist für autonome Systeme ungünstig.

Bei der Verteilung der Steuerungsalgorithmen auf mehrere kleine Recheneinheiten, die untereinander kommunizieren und eigene Ein- und Ausgabefunktionalität besitzen, wird neben der Ein- und Ausgabefunktionalität auch die Steuerungsfunktionalität auf mehrere Rechner verteilt. Gegenüber der zuvor erwähnten Variante besteht nun die Möglichkeit, die Steuerungsfunktionen lokal auf dem jeweiligen Mikrocontroller zu realisieren. Dadurch wird ein unmittelbarer Zugriff auf die Peripherieeinheiten gewährleistet. Das ist für überlebenswichtige Funktionen von großer Bedeutung. Dadurch kann bei gestörter Datenübertragung zumindest ein konservatives Verhalten eines Teilsystems gesichert werden. Beispielsweise wird bei fehlender Kommunikation dafür gesorgt, dass das betroffene Bein seine Gelenkwinkel nicht verändert. Da auf der Seite des Kommunikationspartners diese Störung ebenfalls bemerkt wird, kann für die anderen Beine die gleiche Strategie genutzt werden, so dass der Roboter in einem kontrollierten Zustand bleibt. Somit können Methoden der Fehlertoleranz genutzt werden. Durch die Verlagerung von Steuerungsfunktionalität in die über den Bus gekoppelten Controller besteht zusätzlich die Möglichkeit, dass sie sich gegenseitig kontrollieren. Dazu können verschiedene Controller mit der Fähigkeit ausgestattet werden, andere Controller mittels eines nicht maskierbaren Interrupts (NMI) oder mittels eines Resets in einen definierten Zustand zu versetzen, so dass dieser seine Funktion wieder übernehmen kann.

Durch die lokale Realisierung von Steuerfunktionen sind die hohen Updatefrequenzen nur für das Sensor-Aktor-Interface selbst notwendig. Die Daten müssen nicht mit dieser hohen Rate über den Bus übertragen werden, was zu einer Entlastung des Busses führt. Dadurch wird weniger Bandbreite belegt. Es besteht somit die Möglichkeit, mehr Stationen als bei der vorhergehenden Variante hinzuzufügen, was zu einer Verbesserung der Skalierbarkeit führt.

Durch die Verteilung der Steuerungsfunktionalität auf mehrere Controller verringert sich die benötigte Rechenleistung für den zentralen Prozessor. Dadurch kann sowohl mit einer niedrigeren Taktung als auch mit wenigen komplexen Prozessoren gearbeitet werden, was zu einer Reduzierung des Energieverbrauchs führt. Die Verteilung auf mehrere kleine Prozessoren vereinfacht außerdem die Ableitung der Verlustwärme.

Die Verteilung der Steuerung auf mehrere Mikrocontroller ist allerdings nur dann sinnvoll, wenn Algorithmen nebenläufig sind oder erst eine Synchronisation nach deren Abarbeitung erforderlich ist. Lassen sie sich jedoch nur sequentiell abarbeiten, ist diese Methode nicht sinnvoll. Für den Laufroboter Katharina ist eine Verteilung der Steuerung auf mehrere Mikrocontroller durch die Struktur des Steuerungssystems sinnvoll. Durch die Trennung in eine Hauptsteuerungsebene und eine Beinsteuerungsebene entstehen zwei verschiedene Aufgabenbereiche, die nur eine geringe Abhängigkeit voneinander besitzen. Wahrend die Hauptsteuerungsebene die Planung der Beinbewegungen und deren Koordinierung übernimmt, ist die Beinsteuerungsebene für die Realisierung der Beinbewegungen zuständig. Die Realisierung der jeweiligen Beinbewegungen kann nebenläufig bearbeitet werden. Für die Realisierung einer Beinbewegung muss nicht bekannt sein, wie die aktuelle Position der Nachbarbeine ist, da die Koordinierung bereits durch die Hauptsteuerungsebene erfolgt. Somit ist die Realisierung einer Beinbewegung ein zu den anderen Beinbewegungen nebenläufiger Prozess. Das bedeutet, dass die Beinsteuerungsebene auf mehrere Controller verteilt werden kann. Dadurch ergibt sich eine baumartige Struktur, wie sie in Abbildung 4.2 durch das Zusammensetzen der Funktionseinheiten erkennbar ist und in Abbildung 4.3 bei der Dekomposition der Roboterbewegung ebenfalls erkennbar ist. Die gleiche Unterteilung lässt sich im Prinzip noch einmal

zur Trennung von Beinsteuerung und Gelenksteuerung vornehmen, hier liegt jedoch eine stärkere Kopplung der beiden Ebenen vor. Die Gelenkregelung mit Positionsrückführung, wie sie beim Laufroboter Katharina realisiert wurde, ist nicht so komplex, dass ein eigener Prozessor gerechtfertigt ist. Außerdem entsteht durch die Kraftregelung eine stärkere Bindung zwischen Bein- und Gelenkebene als zwischen Hauptsteuerungs- und Beinebene.

Die Vorteile der Verteilung der Steuerungsalgorithmen auf mehrere kleine Recheneinheiten sind in folgenden Punkten zu sehen

- Verteilung nebenläufiger Steuerungsalgorithmen auf mehrere Prozessoren,
- Reduzierung der Komplexität durch Verwendung mehrerer Mikrocontroller,
- vereinfachtes Design durch Vorhandensein der Ein- und Ausgabe-Interfaces auf den Mikrocontrollern,
- gegenseitige Kontrolle der Teilsysteme möglich,
- definiertes Verhalten im Fehlerfall durch direkten Zugriff auf die Ein- und Ausgabe-Interfaces,
- mittels Mikrocontroller gesteuerte Beine sind autonome Teilsysteme und
- Skalierbarkeit des Steuerungssystems durch Ankopplung weiterer Mikrocontroller an den Bus.

Beim Steuerungssystem des Laufroboters Katharina werden die Steuerungsalgorithmen auf sieben Mikrocontroller verteilt: einer für die zentrale Steuerung und je einer für ein Bein, genutzt, auf die die Steuerungsalgorithmen verteilt werden. Der Einsatz eines Mikrocontrollers für die zentralen Steuerungsfunktionen eröffnet die Möglichkeit, zentrale Sensoren, die Informationen in Bezug auf den Roboterkörper gewinnen, anzuschließen und direkt mit Hilfe der dort angesiedelten Steuerungsalgorithmen zu verarbeiten. Das gilt gleichermaßen für zentral gesteuerte Aktoren.

### 4.2.2 Kommunikation

Durch das Konzept, die Steuerungsalgorithmen auf mehrere Prozessoren zu verteilen, entsteht die Notwendigkeit der Kommunikation. Zum Austausch der Daten zwischen den Prozessoren gibt es nach [Oberschelp, Vossen 00] prinzipiell zwei Möglichkeiten. Diese sind

- Kommunikation über gemeinsamen Speicher und
- Kommunikation über Nachrichtenaustausch.

Bei der Kommunikation über einen gemeinsamen Speicher entsteht eine sehr starke Bindung zwischen den Prozessoren, da hier umfangreiche hardwaretechnische Maßnahmen zur Lösung des gemeinsamen Zugriffs auf diesen Speicher notwendig sind. Insbesondere beim Zugriff auf den gleichen Speicherbereich muss gesichert werden, dass beim Zugriff auf den Speicher von einem Prozessor die anderen Prozessoren warten müssen. Das Problem lässt sich durch einen Dual Ported bzw. Multi Ported RAM lösen. Dieser spezielle RAM besitzt alle Adress- und Dateneingänge mehrfach, so dass Zugriffe quasi simultan stattfinden können. Intern besitzt der RAM eine Arbitrierungseinheit, die die Zugriffe auf die Speicherzellen regelt. Werden nicht atomare Daten in den Speicher geschrieben, so muss zusätzlich dafür gesorgt werden dass sie konsistent sind. Das heißt, es darf erst darauf zugegriffen werden, wenn die Daten komplett im Speicher stehen. Beim Lesen eines Datenbereiches gilt diese Anforderung ent-

sprechend. Ein weiteres Problem besteht darin, dem Kommunikationspartner über die geänderten Daten zu informieren. Hierin besteht ein spezielles Problem bei gemeinsam genutzten Speichern. Zur Realisierung sind zwischen beiden Systemen eine Vielzahl von Verbindungen, d.h. Adress- und Datenleitungen notwendig. Dadurch entsteht ein kompliziertes, unflexibles Design, das nur in speziellen Fällen sinnvoll ist. Im Fall des Steuerungssystems für den Laufroboter Katharina wäre ein spezielles Design notwendig, das alle Controller in geeigneter Weise miteinander verbindet. Die Skalierung ist mit entsprechendem Aufwand möglich.

Die Kommunikation mittels Nachrichtenaustausch erfolgt mit einem seriellen Bus, da in diesem Fall der geringste Aufwand bei der Realisierung der Hardware besteht. Alle Maßnahmen zur Zugriffssteuerung sind nur für das Kommunikationsmedium notwendig. Eine Potentialtrennung, wie sie bei vielen Feldbussen eingesetzt wird, ist auch innerhalb eines Mikrocontrollersystems realisierbar. Durch die geringe Anzahl notwendiger elektrischer Verbindungen unterstützt ein serieller Bus das Hinzufügen neuer Kommunikationsteilnehmer, sofern dadurch keine elektrischen Probleme hervorgerufen werden. Zum Beispiel sind bei einem Bus basierend auf den RS495-Standard eine maximale Anzahl von 32 Knoten pro System und bei einem CAN-Bus 30 Knoten bei 1 Mbit/s angegeben [Lawrenz 99]. Die serielle Übertragung von Daten wird von vielen Mikrocontrollern unterstützt, so dass sich einfache Lösungen realisieren lassen.

Zur Kommunikation zwischen Rechnern wurde von der ISO ein Schichtenmodell standardisiert, das aus sieben Schichten besteht [Tanenbaum 96]. Dieses Schichtenmodell ist ein Universalmodell, mit dem sich Rechnerverbindungen unterschiedlichster Art realisieren lassen. Im Feldbusbereich werden für die Kommunikation der Teilnehmer nur die Schichten 1, 2 und 7 (z.B. Profibus, Interbus, CAN) genutzt [Phoenix 98], [Lawrenz 99], [Baginski 98], [Bender 93]. Abbildung 4.5 zeigt die Schichten des OSI-Modells. Die grau eingefärbten Schichten werden von den genannten Feldbussen genutzt. Wie bei den Feldbussen sind die Schichten 1, 2 und 7 wichtig.

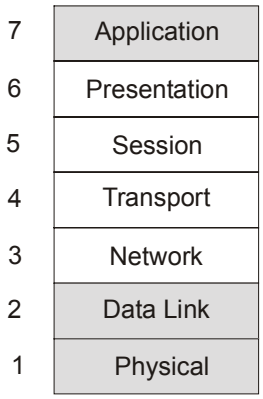



Abbildung 4.5: Das OSI-Schichtenmodell

Für die Realisierung der Datenübertragung auf physikalischer Ebene (Schicht 1) sind verschiedene Vernetzungstopologien, z.B. Ring, Baum, Stern oder Bus denkbar. Stern- und Baumtopologien sind für das Steuerungssystem nicht geeignet, da an den Verzweigungsknoten mehrere Busanschlüsse zur Verfügung gestellt werden müssen. Das Weiterleiten der übertragenen Daten in das Zielsegment würde zusätzlichen Protokollaufwand erfordern. Bei einem Ring müssten alle Informationen durch sämtliche Controller geleitet werden, was ebenfalls zu einem erhöhten Protokollaufwand führt. Für die Einfachheit der Realisierung in einem prototypischen Steuerungssystem spricht die Bustopologie, da hier alle Stationen die übertragenen Informationen mithören und die relevanten Informationen empfangen können. Beim

Senden ist ebenfalls sichergestellt, dass der Empfänger diese Nachricht erhalten kann. Die physikalische Realisierung des Busses kann mittels spezieller Transceiver oder direkt über logische Pegel erfolgen. Dabei muss sichergestellt werden, dass inaktive Stationen die Bussignale nicht stören. Die Verbindung zwischen den Controllern kann über die Rückverdrahtung der Controllersteckplätze erfolgen.

Da das Buskonzept Vorteile hinsichtlich der Einfachheit in Realisierung und Erweiterbarkeit bietet, wurde es für die Realisierung für den Laufroboter Katharina ausgewählt. Für diesen Einsatzfall sind ebenfalls nicht alle Schichten des OSI-Modells relevant, da nur wenige, genau spezifizierte Daten übertragen werden. Eine Weitervermittlung von Daten ist beispielsweise nicht notwendig, da alle Teilnehmer am gleichen Segment kommunizieren.

In der Schicht 2 werden Dienste zur Datensicherung und zur Buszugangskontrolle realisiert. Die Datensicherung lässt sich mittels fehlererkennender oder fehlerkorrigierender Codierung bzw. mittels bestätigter Übertragungen realisieren.

Bei der Datensicherung auf Telegrammebene durch fehlererkennende Codierung oder einen fehlerkorrigierenden Code können sporadische Störungen, die nur einzelne Bits beeinflussen, erkannt und gegebenenfalls behoben werden. Es ist jedoch auch möglich, dass die Synchronisation zwischen Sender und Empfänger gestört ist. In diesem Fall passt die Fehlersicherungsinformation nicht zum Telegramm und wird somit erkannt. Im Fall der fehlenden Synchronisation kann diese durch Abwarten der folgenden Telegrammkennung wieder hergestellt werden. Geht die Synchronisation während der Übertragung verloren, so sind die Daten unvollständig oder verändert, was durch einen fehlererkennenden Code entdeckt werden kann. Die fehlererkennende Codierung kann auf Telegrammebene mittels CRC (Cyclic Redundancy Check) realisiert werden.

Bei der Datensicherung mittels bestätigter Übertragungen schickt der Empfänger nach Erhalt eines Telegramms dem Sender eine Bestätigung zurück. Dabei ist zu unterscheiden, ob eine Bestätigung in jedem Fall, nur bei positivem Ergebnis oder nur bei negativem Ergebnis erfolgt. Ein bestätigter Dienst ist nur dann sinnvoll, wenn auf eventuell auftretende Übertragungsfehler auf Senderseite reagiert werden kann. In Echtzeitsystemen ist das genau dann nicht möglich, wenn dadurch eine Frist überschritten wird, die Daten nur kurzzeitig gültig sind oder durch eine wiederholte Übertragung andere Übertragungen in unzulässiger Weise verzögert werden. Beim Laufroboter Katharina sind keine bestätigten Dienste vorgesehen, da die Daten durch periodische Übertragungen nur eine kurze Lebensdauer haben und die wiederholte Übertragung deshalb nicht sinnvoll ist. Außerdem bestünde die Gefahr, dadurch unzulässige Verzögerungen zu verursachen. Wegen der periodischen Übertragung kann es toleriert werden, wenn einzelne Telegramme ausfallen, da sich die aufeinanderfolgend übertragenen Informationen (z.B. Beinposition) nur relativ wenig im Inhalt unterscheiden. Somit ist eine temporäre Redundanz gegeben [Ihme 00a]. Der Verzicht auf eine bestätigte Übertragung impliziert, dass kein verbindungsorientiertes Protokoll eingesetzt wird.
Für den Buszugang lassen sich die Verfahren

- Master-Slave-Kommunikation,
- Tokenverfahren,
- Spontaner Zugriff (MA) und
- Timesharingverfahren (TDMA)

herausstellen [Neumann 90]. Bei der Master-Slave-Kommunikation werden die Busteilnehmer durch einen Master nach Sendewünschen abgefragt. Dabei werden die zeitlichen Restrik-

tionen durch den Master vorgegeben und deren Einhaltung überwacht. Beim Tokenverfahren wird das Senderecht durch einen Token einer Station zugeteilt. Das Überwachen der Fristen muss dabei vom jeweiligen Tokeninhaber erfolgen, indem er die Tokenumlaufzeit prüft und in Abhängigkeit von der Restzeit Kommunikation initiieren kann. Dieses Verfahren wird beispielsweise beim Profibus genutzt [Bender 93]. Zwischen den ersten beiden Varianten sind auch Mischverfahren möglich, z.B. „flying master“ [Neumann 90], [Bender 93]. Beim spontanen Zugriffsverfahren kann jeder Teilnehmer nach Bedarf kommunizieren. Für den Buszugang ist eine spezielle Logik zur Erkennung bzw. Vermeidung von Kollisionen und Verfahren zur Kollisionsauflösung notwendig.

Die Einhaltung von Fristen kann nicht allein durch die Vergabe von Prioritäten, wie beim CAN-Bus möglich, garantiert werden, da die Übertragung nicht unterbrechbar ist. Für eine echtzeitfähige Kommunikation müssen spezielle Protokolle, z.B. mittels TDMA implementiert werden. Bei diesem Verfahren werden Zeitschlitze für einzelne Übertragungen reserviert. Dadurch können Fristen garantiert werden.

Beim Steuerungssystem des Laufroboters Katharina gibt es einen Hauptcontroller, der im Steuerungssystem koordinierende Funktionen übernimmt. Die Beincontroller übernehmen die Realisierung der Bewegungen, so dass sie periodisch mit dem Hauptcontroller kommunizieren müssen. Diese Kommunikationsaufgabe kann durch eine Master-Slave-Kommunikation effizient gelöst werden. Dabei fragt der Master, d.h. der Hauptcontroller, alle Slaves, d.h. die Beincontroller, nach aktuellen Sensorwerten ab, um anschließend alle Sollwerte für die Beinsteuerung zu übertragen. In einer weiteren optionalen Phase werden den Beincontrollern zusätzliche Daten gesandt oder es werden Daten abgefragt, beispielsweise für Diagnosezwecke. Abbildung 4.6 zeigt die drei Kommunikationsphasen am Beispiel der Kommunikation eines Masters mit einem Slave. Die Dienste des Application Layer werden durch Funktionen zum Senden und Empfangen der Daten bereitgestellt.

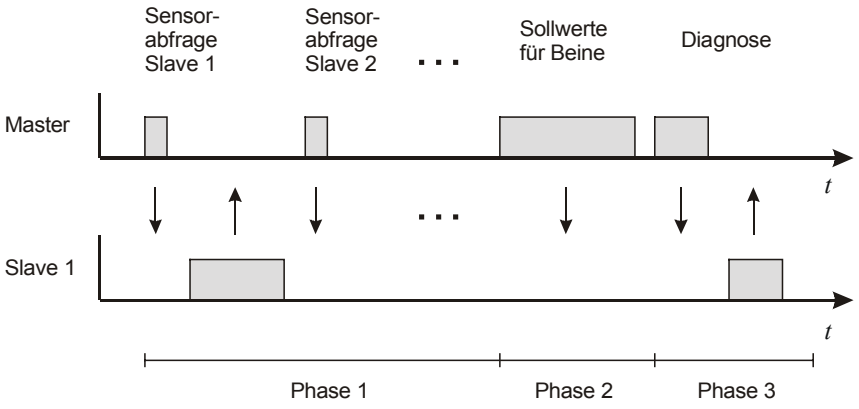



Abbildung 4.6: Konzept der Master-Slave-Kommunikation

### 4.2.3 Sensoren und Aktoren

Die Sensoren und Aktoren eines Laufroboters stellen die Schnittstelle zur Außenwelt her. Mit Hilfe der Sensoren werden Informationen über den Zustand des Roboters und der Außenwelt gewonnen. Es wird zwischen internen und externen Sensoren unterschieden. Interne Sensoren gewinnen Informationen über die inneren Zustände eines Roboters, während externe Sensoren

Informationen über die Umwelt eines Roboters gewinnen. Diese grobe Einteilung lässt sich in Anlehnung an [Dillmann, Huck 91] und [Luo, Kay 89] auf der Basis des Bereiches, in dem die Sensoren die Messgrößen erfassen, weiter verfeinern. Die Erfassung der Messgrößen kann im Roboter selbst, im taktilen Bereich, in näherer Umgebung oder in größerer Entfernung erfolgen. Dabei werden weiter entfernte Objekte zuerst von den Fernbereichssensoren erfasst, während der Roboter sich ihnen nähert. Danach erfassen die Nahbereichssensoren das Objekt, bis es in den taktilen Bereich gelangt.

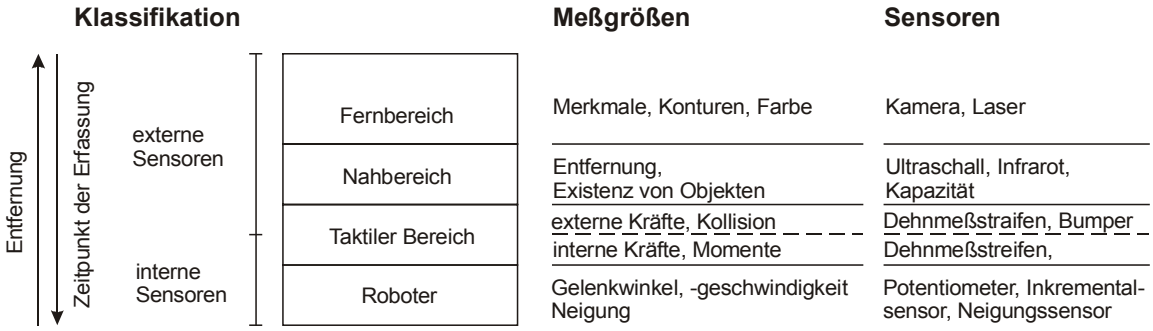



Abbildung 4.7: Klassifikation von Sensoren und Beispiele

Abbildung 4.7 zeigt diese Klassifikation und einige Beispiele zu Messgrößen und Sensoren. Für das Laufen eines Roboters selbst sind die Erfassung von Messgrößen im Fern- und Nahbereich nicht von Interesse, da sie Informationen über entfernte Objekte liefern, die für Wegplanung relevant sind, jedoch keinen unmittelbaren Einfluss auf das Laufen haben. Von hauptsächlichem Interesse sind Sensoren für den taktilen Bereich und interne Sensoren, da sie für die Interaktion mit der Umwelt und die unmittelbare Bewegung der Beine wichtig sind.

Eine andere Klassifizierung von Sensoren kann über das Messprinzip erfolgen. Danach werden relativ messende Sensoren und absolut messende Sensoren unterschieden. Relativ messende Sensoren benötigen eine Vergleichs- oder Referenzgröße, um ein Messergebnis zu liefern. Dazu muss das Referenzsignal zunächst erzeugt werden. Bei einem Inkrementalsensor beispielsweise werden Inkremente ausgehend von einer bekannten Position gezählt, die einmalig angefahren werden muss. Die aktuelle Position bestimmt sich also aus der bekannten Referenzposition und einem zurückgelegten Weg, der durch das Zählen von Inkrementen ermittelt wird. Bei einem Laufroboter können solche Sensoren zur Bestimmung der Gelenkwinkel eingesetzt werden. Dadurch ergibt sich aber die Notwendigkeit, für jedes Gelenk eine Referenzfahrt durchzuführen. Da jedoch die Beine zum Stehen benötigt werden und die Umgebung des Roboters in der Regel unbekannt ist, ist das sichere Stehen in einer solchen Situation nicht gewährleistet. Darüber hinaus besteht bei Referenzfahrten die Gefahr, dass unkontrollierte Kollisionen mit externen Objekten auftreten können. Auch bei absolut-relativ messenden Sensoren besteht dieses Problem, da die Referenzgröße erst nach Erreichen eines von mehreren Referenzpunkten bzw. nach Durchfahren eines Referenzbereiches bekannt ist. In einem Fehlerfall, zum Beispiel bei kurzzeitigem Stromausfall, kann eine sichere Funktion nicht mehr gewährleistet werden. Bei taktilen Sensoren ist sie Situation ähnlich. Relativ messende Sensoren, z.B. piezoelektrische Sensoren, können nicht sofort eine Aussage über einen Bodenkontakt liefern. Aus diesen Gründen werden für alle Funktionen, die mit dem sicheren Stehen eines Roboters in Verbindung stehen, absolut messende Sensoren eingesetzt. Dazu zählen Sensoren für Gelenkwinkel, Kraft- und Kontaktsensoren eingesetzt.

Bei einen autonomen Roboter müssen die Aktoren mit der benötigten Energie versorgt werden, die auf dem Roboter selbst bereitzustellen ist. Für einen Roboter der 20 kg-Klasse, zu der auch der Laufroboter Katharina zählt, sind Gleichstrommotoren für die Aktoren geeignet. Durch eine geeignete Ansteuerung der Motoren muss die elektrische Energie so effizient wie möglich in Bewegungen umgesetzt werden.

# 5 Aufbau und Erprobung des Steuerungssystems für den Laufroboter Katharina

## 5.1 Hardware

In den folgenden Abschnitten wird die Hardware des Steuerungssystems für den Laufroboter Katharina betrachtet. Bei eingebetteten Systemen spielt die Hardware im allgemeinen eine große Rolle. Bei solchen Systemen werden nicht nur Daten manipuliert und die darin enthaltenen Informationen verarbeitet, sondern es werden mit den angeschlossenen Sensor- und Aktorinterfaces Informationen und Signale ausgetauscht. Der Informationsverarbeitungsprozess ist hier sehr stark an die Möglichkeiten der Hardware gebunden, die im Bereich der eingebetteten Systeme zum Teil sehr leistungsfähige, aber spezialisierte Dienste bereitstellt. Dazu gehören größtenteils spezielle integrierte Sub- und Peripherieprozessoren, die die Informationsaufnahme und -ausgabe zum Teil erheblich automatisieren und damit vereinfachen können. Die genaue Kenntnis der Rechnerarchitektur und deren gezielte Nutzung ist eine wichtige Grundlage für die erfolgreiche Entwicklung eines eingebetteten Echtzeitsystems.

### 5.1.1 Mikrocontrollerplatinen

Die sieben Recheneinheiten, die auf dem Laufroboter Katharina untergebracht sind, enthalten Mikrocontroller des Typs Intel 80196KR. Die Platinen haben weiterhin entsprechend benötigte Speichereinheiten (RAM, Flash-ROM) und benötigte Signalverarbeitungseinheiten für Ein- und Ausgabe sowie zur Kommunikation.

#### 5.1.1.1 Mikrocontroller

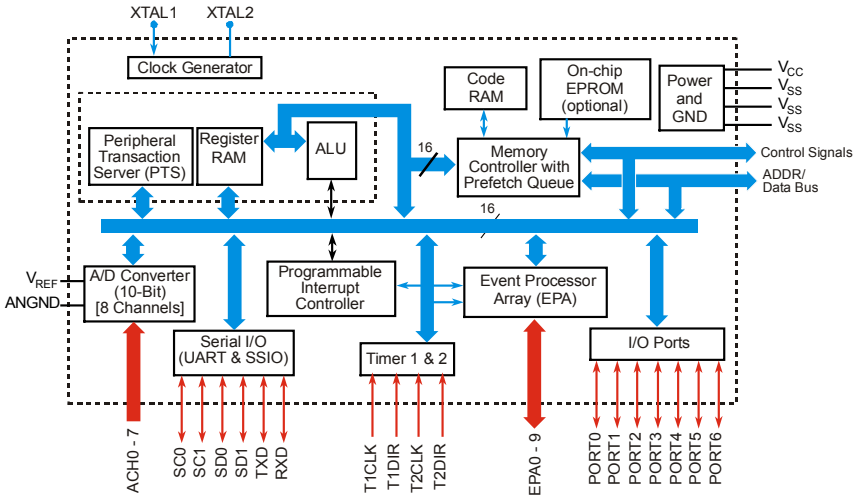



Abbildung 5.1: Blockschema des internen Aufbaus des Mikrocontrollers Intel 801996KR [Intel 98a]

Als Mikrocontroller wurde ein Intel 80196KR genutzt, der sich durch eine geeignete Architektur auszeichnet. Beim Intel 80196KR, dessen Blockschema Abbildung 5.1 zeigt, handelt es sich um einen 16 Bit Mikrocontroller in CHMOS-Technik für schnelle Berechnungen, sowie Ein- und Ausgaben. Dieser Mikrocontroller wurde speziell für Anwendungen entwickelt, bei denen es auf sehr schnelle Reaktionen auf Ereignisse ankommt, zum Beispiel zur Motorsteuerung, für elektronische Geräte, Airbags, Antiblockiersysteme, Festplattenlaufwerke und medizinische Geräte. Hauptkomponenten sind die ALU, verschiedene Speicher, sieben E/A-Ports und verschiedene integrierte Peripherieeinheiten. Diese beinhalten Analog-Digital-Umsetzer, ein Event Processor Array, zwei Timer, sowie synchrone und asynchrone serielle Schnittstellen. Der Adressraum beträgt 64 KB [Intel 95], [Intel 98a], [Intel 98b].

Bei der ALU handelt es sich um eine Register-ALU, die mit einem 512 Byte Registersatz arbeitet. 256 Byte können direkt für beliebige arithmetisch-logische Operationen verwendet werden, der restliche Bereich besteht aus sogenannten Special Function Registern, die die integrierten Peripheriekomponenten ansprechen. Außer auf den intern verfügbaren Speicher kann auf einen externen Speicher über einen Memory-Controller zugegriffen werden.

Der Mikrocontroller verfügt über eine flexible Interruptsteuerung. Sie besteht aus einem Interrupt-Controller, der mit den Hardwarefunktionen gekoppelt ist und einem Peripheral Transaction Server (PTS). Während über den Interrupt-Controller programmierte Interruptfunktionen angesprochen werden, stellt der PTS verschiedene mikrocodierte Interrupt-Serviceroutinen (ISR) bereit, die alternativ genutzt werden können. Mit ihnen ist es möglich, mittels Interrupt ereignisgesteuert Daten von Peripherieeinheiten in den Hauptspeicher zu transferieren und umgekehrt.

Die integrierten Peripheriekomponenten realisieren digitale Ein- und Ausgänge, asynchrone und synchrone serielle Schnittstellen, acht 10 Bit Analog-Digital-Umsetzer (ADU), das Event Processor Array (EPA) sowie einen Watchdog-Timer. Die Ports können entweder als digitale Ein- und Ausgänge oder mit einer speziellen Funktion genutzt werden. Dazu zählen beispielsweise die ADU's. Einige Port-Pins werden für die Ansteuerung des externen Speichers benötigt. Das Event Processor Array ermöglicht es, durch Vergleich mit einem der Timer (compare mode) Ausgaben an Port-Pins zu erzeugen bzw. bei Eintreten eines vordefinierten Ereignisses (Zustandswechsel am Port-Pin) den aktuellen Zählerstand zu speichern (capture mode). Die genannten Ereignisse können Interrupts auslösen, die z.B. durch den Peripheral Transaction Server bedient werden können. Auf diese Weise werden leistungsfähige, ereignisgesteuerte Steuerungsalgorithmen unterstützt.

#### 5.1.1.2 Hauptcontroller

Die Aufgaben des Hauptcontrollers sind hauptsächlich bestimmt durch Transformationen und planerische Aufgaben, etwa der Generierung von Laufzyklen. Die physikalische Strukturierung des Hauptcontrollers ist in Abbildung 5.2 dargestellt. Der Hauptcontroller ist aus dem Mikrocontroller 80196KR und einigen peripheren Komponenten aufgebaut. Durch die in den Mikrocontroller integrierten Peripheriekomponenten vereinfacht sich der Aufbau des Hauptcontrollers deutlich. Die in Abbildung 5.2 gezeigten Blöcke CPU, asynchrone serielle Kommunikationseinheit (UART), synchrone serielle Kommunikationseinheit (SSIO), analoge Eingabeeinheit (ADU und Signalverarbeitung) und Ausgabeeinheit auf Basis der Pulsweitenmodulation (PWM-Ausgabeeinheit) werden mit Hilfe der in den Mikrocontroller integ-

rierten Komponenten realisiert und nur durch eine Schaltung zur elektrischen Signalformung ergänzt. Zum Grundausbau des Hauptcontrollers zählen entsprechend des vorgesehenen Aufgabenbereiches lediglich CPU, asynchrone serielle Kommunikationseinheit, synchrone serielle Kommunikationseinheit, RAM und Flashspeicher. ADU und PWM-Ausgabeeinheit wurden implementiert, sind jedoch nur optional vorgesehen, etwa zum Auslesen zentraler Sensoren oder zur Ansteuerung eines Manipulators oder eines Kamerakopfes.

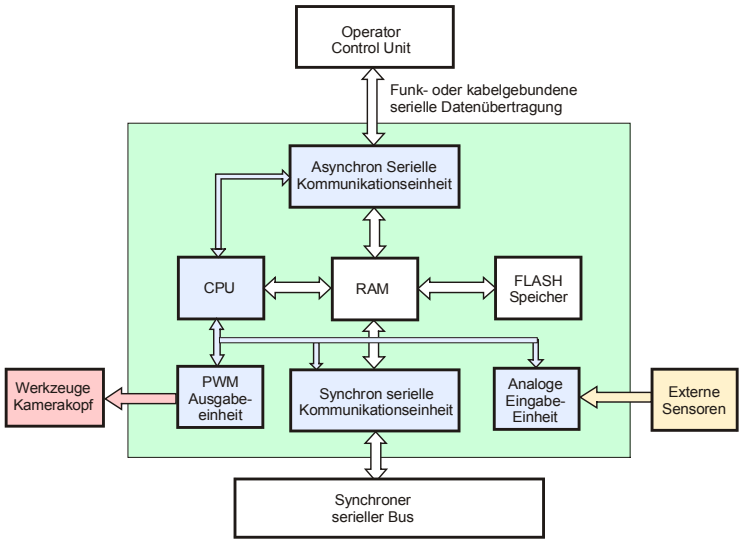



Abbildung 5.2: Physikalische Struktur des Hauptcontrollers

Die asynchrone serielle Kommunikationseinheit dient zur Kommunikation mit der externen Experimentsteuerung. Um hier eine elektrisch robuste Informationsübertragung, auch über längere Leitungen, zu ermöglichen, wird für die Datenübertragung eine RS422-Schnittstelle genutzt. Dabei handelt es sich um eine standardisierte physikalische Übertragungsebene. Die elektrischen Signale werden über einen differentiellen Zweidrahtbus übertragen. Zur Realisierung wird ein SN75179 von Texas Instruments genutzt [TI 98]. Um Problemen mit Potentialunterschieden vorzubeugen, wird für die seriellen Schnittstellen und die digitalen Ein- und Ausgänge eine Potentialtrennung über Optokoppler realisiert.

Für die synchrone serielle Kommunikationseinheit wird ebenfalls ein differentieller Zweidrahtbus genutzt. Hier wird ein SN75176 eingesetzt, der für einen Mehrstationenbetrieb am synchron seriellen Bus geeignet ist [TI 99].

#### 5.1.1.3 Beincontroller

Der Beincontroller ist ebenfalls mit dem Mikrocontroller 80196KR und einigen peripheren Komponenten aufgebaut. Auch hier vereinfacht sich der Aufbau erheblich, da viele Peripherieeinheiten bereits in den Mikrocontroller integriert sind.

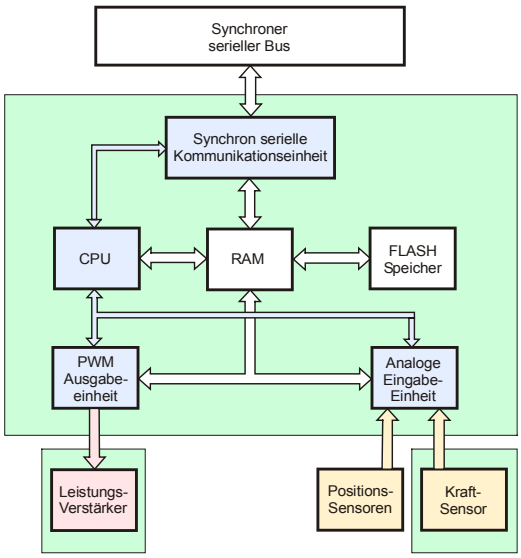



Abbildung 5.3: Physikalische Struktur des Beincontrollers mit Peripherie

Die in Abbildung 5.3 gezeigten Blöcke CPU, asynchrone serielle Kommunikationseinheit, synchrone serielle Kommunikationseinheit, analoge Eingabeeinheit und PWM-Ausgabeeinheit werden ebenfalls mit Hilfe der in den Mikrocontroller integrierten Komponenten realisiert und nur durch eine Schaltung zur elektrischen Signalformung ergänzt. RAM und Flashspeicher sind hier ebenfalls durch externe Schaltkreise realisiert.

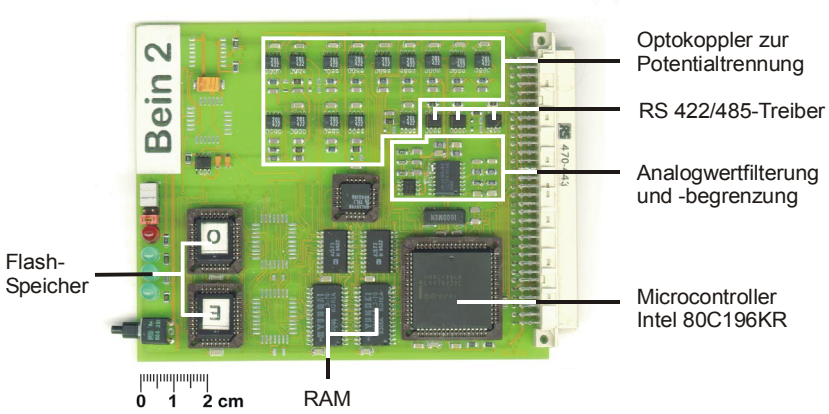



Abbildung 5.4: Physikalischer Aufbau des Beincontrollers

Abbildung 5.4 zeigt den Aufbau einer Beincontrollerplatine. Die Hälfte der Platine wird vom Mikrocontroller und von externen Speichern belegt, während die andere Hälfte Signalformungs- und Schutzfunktionen dient. Die Analogwertfilter sollen unerwünschte Störungen, die durch Leitungen eingekoppelt werden können, unterdrücken. Das ist auch deshalb von besonderer Bedeutung, weil beispielsweise durch die Transformationen alle drei Achsenmesswerte miteinander verknüpft werden und so Störsignale auf alle drei kartesischen Koordinaten gleichzeitig Einfluss haben können. Zum Schutz der digitalen Ein- und Ausgänge, sowie zur Trennung der Digital- und Leistungselektronik werden auch beim Beincontroller Optokoppler eingesetzt.

#### 5.1.1.4 Joystickcontroller

Die Operatorsteuerung soll Steuerkommandos vom Bediener entgegennehmen und notwendige Informationen über den Roboter und seiner Steuerung dem Bediener vermitteln. Ein geeignetes Interface für Operatoreingaben sind Joysticks. Mit ihnen können Bewegungsabläufe oder proportionale Steuergrößen aufgenommen werden, wobei zwei oder mehr Größen gleichzeitig verarbeitet werden können. Eine direkte und intuitive Steuerung lässt sich mit proportionalen Auslenkungen oder durch proportionalen Krafteinsatz erreichen. Proportionale Steuergrößen haben den Vorteil, dass feinmotorische Fähigkeiten des Bedieners genutzt werden können. Die empfindlichen Sinnesorgane der Hände erlauben es außerdem, dem Bediener taktile Informationen zu vermitteln, beispielsweise Kräfte, Vibrationen oder Oberflächenstruktur. Durch solche Rückkopplungseffekte kann dem Bediener auch vermittelt werden, welche Einschränkungen in den Freiheitsgraden bestehen oder um Eingaben auf bestimmte Freiheitsgrade zu begrenzen. Mit diskreten Eingaben ist es hingegen möglich, bestimmte Funktionsmodi oder binäre Aktoren zu aktivieren bzw. inkrementelle Vorgänge zu steuern, z.B. das Drehen eines Gelenkes um einen bestimmten voreingestellten Betrag. Diese Bedienmöglichkeiten sollen hauptsächlich der Tastatur vorbehalten bleiben. Einige wenige binäre Bedienelemente sollten jedoch unmittelbar am Joystick vorhanden sein, um schnell zwischen wichtigen Funktionsmodi wechseln zu können.

Für den beabsichtigten Ausbau des Bedieninterfaces für künftige Untersuchungen ist ein Joystick mit drei Eingabeachsen und Kraftrückführung für jede Achse vorgesehen. Zur Realisierung ist ein Interface notwendig, das drei Achsenpositionen auslesen, drei wirkende Kräfte ermitteln und drei Stellgrößen zur Krafterzeugung ausgeben kann. Des weiteren sollen Bedienereingaben an das Steuerungssystem weitergeleitet und die Sollwerte für die Kraftrückführung vom Steuerungssystem entgegengenommen werden. Die spezifizierten Anforderungen lassen erkennen, dass das Interface ähnliche Peripheriefunktionen zu erfüllen hat, wie die Beincontroller. Es liegen ähnliche Aufgaben zur Sensorinformationsverarbeitung und zur Steuerung von Freiheitsgraden vor, wie es bei den anderen Controllern der Fall ist. Daher ist es naheliegend, von den Haupt- und Beincontrollern abgeleitete Hard- und Software einzusetzen.

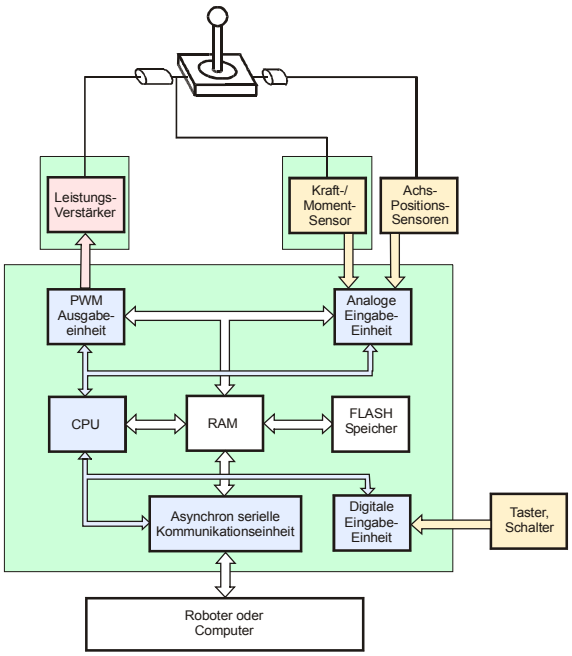



Abbildung 5.5: Physikalische Struktur des Joystickcontrollers

Die physikalische Struktur des Joystickcontrollers ist in Abbildung 5.5 dargestellt. Es ist zu erkennen, dass bereits bekannte Komponenten verwendet werden können. Eine Erweiterung findet hier nur durch digitale Ein- und Ausgänge statt.

Zweckmäßigerweise sollte das Joystickinterface einfach mit dem Rechner zur Experimentsteuerung oder dem Roboter selbst zu verbinden sein. So entsteht eine abgeschlossene Funktionseinheit, die sich durch folgende Merkmale auszeichnet:

1. flexible Anschaltung von neuen Bedienelementen oder Erweiterung ihrer Eigenschaften,
2. einfache Verbindung mit Roboter bzw. Computer über serielle Standardschnittstelle und
3. Erweiterung der Bedienfunktionalität durch Anpassung der Datenprotokolle.

In einer ersten Ausbaustufe wurde die Joystickeinheit für drei Auslenkungen realisiert. Die Funktionen für die Kraftrückführung (Stellwertausgabe, Einlesen der Sensorwerte) sind durch Verwendung gleicher Controllerhardware einschließlich der Anschaltbarkeit der Peripheriekomponenten vorbereitet bzw. durch Verwendung der Steuerungsalgorithmen der Beincontroller bereits vorhanden.

### 5.1.2 Messwerterfassung

Bevor die Messdaten von den Sensoren als Informationen im Mikrocontroller verarbeitet werden können, müssen die Messgrößen von den Sensoren in elektrische Größen umgewandelt werden, die dann wiederum in binär darstellbare Informationen gewandelt werden. Nachfolgend wird auf die Gewinnung der Sensorsignale und deren Bereitstellung zur Weiterverarbeitung durch den Mikrocontroller eingegangen.

#### 5.1.2.1 Gelenkwinkel

Für die Fortbewegung ist die gezielte Manipulation der Gelenkwinkel von grundlegender Bedeutung. Zur Messung des aktuellen Gelenkwinkels können absolute oder relative Sensoren bzw. analoge oder inkrementelle Sensoren verwendet werden.

Da die Umgebung des Roboters als unbekannt angenommen wird, sollten Messprinzipien, die eine Referenzfahrt des Gelenkes erfordern, vermieden werden. Damit sind relativ messende Sensoren, die für ihre Funktion eine Referenzfahrt benötigen, nicht einsetzbar. Durch den Einsatz absolut messender Sensoren ist sichergestellt, dass zu jedem Zeitpunkt eine Information über den aktuellen Gelenkwinkel gewonnen werden kann.

Die Messung des Gelenkwinkels ist mit absolut-diskreten und absolut-analogen Sensoren möglich. Bei absolut-diskreten Sensoren besteht aus Implementierungssicht in der Regel das Problem, dass sie viele Anschlussleitungen benötigen oder bereits eine Anschaltung an ein Bussystem besitzen, das mit dem eingesetzten Mikrocontroller gekoppelt werden muss. Der Anschluss eines Sensors ist allgemein dann mit geringem Aufwand zu realisieren, wenn dies durch den Mikrocontroller selbst unterstützt wird. Der geeignetste Sensor ist hier ein Potentiometer, das eine absolute Positionsinformation liefert und einfach anschließbar ist.

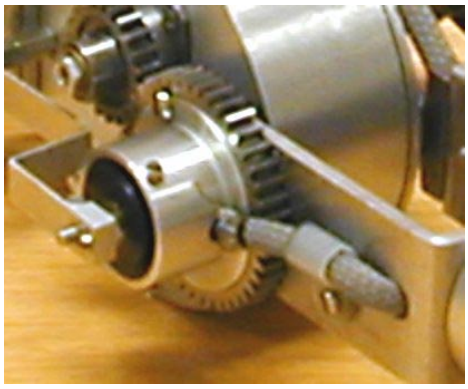


Abbildung 5.6: Potentiometer zur Messung des Gelenkwinkels

Abbildung 5.6 zeigt ein Leitplastikpotentiometer, das zur Messung des Gelenkwinkels direkt auf die Gelenkachse des Laufroboters Katharina montiert ist.
Der Nachteil bei der verwendeten Methode besteht darin, dass aufgrund der analogen Informationsdarstellung durch Einkopplung von Störsignalen Messwerte verfälscht werden können. Am wahrscheinlichsten ist die Einkopplung von Störsignalen durch die Leistungssignale der Motoransteuerung, die mit einer Puls-Weiten-Modulation von 4 kHz bzw. 8 kHz arbeitet. Für die Unterdrückung von Störungen bei der Messung der Gelenkwinkel werden aktive Filter zweiter Ordnung benutzt. Das Übertragungsverhalten eines solchen Tiefpasses ist [Tietze, Schenk 99]

$$A(s_n) = \frac{1}{1 + a_1 s_n + b_1 s_n^2} \quad . \tag{5.1}$$

Für die Bewegung der Gelenke wird eine Grenzfrequenz von 10 Hz als ausreichend angesehen. Bei einer vorgesehenen Abtastzeit von 10 ms ist die größte darstellbare Frequenz 50 Hz, die im vorliegenden Fall auch als größte sinnvolle Grenzfrequenz gelten kann. Das Übertragungsverhalten des verwendeten Tiefpasses zweiter Ordnung mit Einfachmitkopplung lässt sich mit

$$A(s_n) = \frac{1}{1 + \omega_g C_1 (R_1 + R_2) s_n + \omega_g^2 R_1 R_2 C_1 C_2 s_n^2} \tag{5.2}$$

beschreiben. Aus Koeffizientenvergleich von (3.1) und (5.2) erhält man die Koeffizienten $a_1$ und $b_1$

$$a_1 = \omega_g C_1 (R_1 + R_2) \tag{5.3}$$

$$b_1 = \omega_g^2 R_1 R_2 C_1 C_2 \quad . \tag{5.4}$$

Die Filterkoeffizienten wurden beim Laufroboter Katharina mit $a_1$=1,414 und $b_1$=1,000 so gewählt, dass eine Butterworth-Charakteristik entsteht [Seifart 87]. Die Widerstände wurden identisch gewählt, d. h. es gilt $R_1$=$R_2$=$R$. Damit lässt sich die Kapazität der Kondensatoren entsprechend der gewünschten Grenzfrequenz mit

$$C_1 = \frac{a_1}{2 R \omega_g} \tag{5.5}$$

$$C_2 = \frac{b_1}{R^2 C_1 \omega_g^2} \tag{5.6}$$

bestimmen. Die Dämpfung des Filters bei einer angenommenen Störfrequenz von 4 kHz, was der Grundwelle bei einer 4 kHz-PWM entspricht, und einer Grenzfrequenz von 50 Hz beträgt dann

$$A(s_n) = \frac{1}{1 + 1{,}414 \cdot \frac{4\,kHz}{50\,Hz} + 1{,}00 \cdot \left(\frac{4\,kHz}{50\,Hz}\right)^2} = 1{,}53 \cdot 10^{-4} = -76\,dB\,. \tag{5.7}$$

Um bei einem mit 10 Bit auflösenden Analog-Digital-Wandler ($n_{ADU}$), der einen Spannungsbereich $U_{ADU,\max}$ von 5 V verarbeitet, eine Messwertänderung um 1 Digit zu bewirken, müsste eine Störspannung von

$$U_{St} = \frac{1}{A_{4kHz}} \cdot \frac{U_{ADU,\max}}{2 \cdot n_{ADU}} = \frac{1}{1{,}53 \cdot 10^{-4}} \cdot \frac{5V}{2 \cdot 1024} \approx 16\ V \tag{5.8}$$

eingekoppelt werden. Abbildung 5.7 zeigt den entsprechenden Amplitudengang im Bereich von 1 Hz bis 10 kHz.

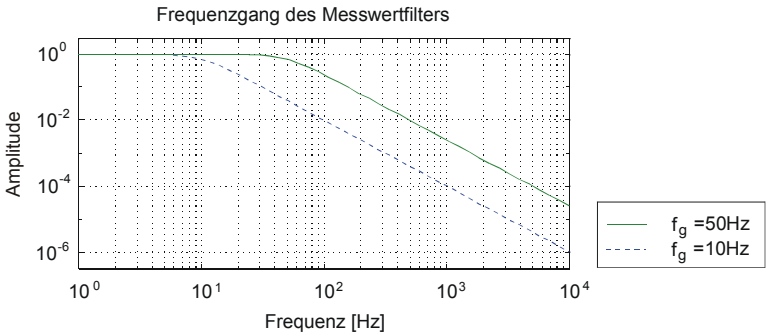



Abbildung 5.7: Frequenzgang des Messwertfilters

### 5.1.2.2 Fußkräfte

Die Erfassung der Fußkräfte eröffnet die Möglichkeit, einen Boden- oder Hinderniskontakt eines Roboters zu detektieren oder seine auf den Boden übertragene Kraft zu bestimmen. Zur Bestimmung dieser Kraft stehen prinzipiell zwei verschiedene Methoden zur Verfügung. Eine Methode basiert auf der Bestimmung der Kraft als Funktion der Momente, die durch die Gelenke eingebracht werden und eine andere basiert auf der direkten Messung der Kräfte durch einen Sensor im oder in der Nähe des Fußes ähnlich der Kraftmessung im Hauptgelenk bei Industrierobotern [Hauptmann 91]. Die erste Methode erfordert entweder einen speziellen Momentensensor für jeden Antrieb oder das Antriebsmoment muss über den Ankerstrom des Motors bestimmt werden. Letzteres gestaltet sich schwierig für stark untersetzende Getriebe, wie sie beim Laufroboter Katharina verwendet werden, da eine hohe Selbsthemmung des Getriebes existiert und somit neben dem abgegebenen Antriebsmoment auch ein Reibmoment

und eventuelle Beschleunigungsmomente in die Messung eingehen. Bei der zweiten Methode bestehen diese Probleme nicht. Das Hauptproblem ist hier die Integration eines geeigneten Sensors in die Roboterkonstruktion. Zur Kraftmessung wurden auch für andere Laufmaschinen Kraftsensoren entwickelt. Im Gegensatz zu Industrierobotern sind die Anforderungen an Genauigkeit und Größe des Messbereiches nicht so hoch. Während es bei Industrierobotern darauf ankommt, kleine Kräfte resultierend aus kleinen Verformungen zu messen, müssen bei Laufrobotern große Stützkräfte in den Beinen gemessen werden. Fußkraftsensoren für Laufroboter besitzen deshalb eine größere Steifheit, was der Integration in solche Roboter entgegenkommt [Gorinevsky et al. 97].

Das Prinzip zur Kraftmessung basiert darauf, dass ein Verformkörper aufgrund von wirkenden Kräften gedehnt oder gestaucht wird. Diese Dehnungen oder Stauchungen können mit Hilfe von längenabhängigen Widerständen gemessen werden, die als Dehnmessstreifen auf den Verformkörper appliziert werden. Der Widerstand $R$ eines elektrischen Leiters ist

$$R = \rho \cdot \frac{L}{A} \tag{5.9}$$

mit der Länge $L$, dem Querschnitt $A$ und dem spezifischen elektrischen Widerstand $\rho$. Wird der Widerstand gedehnt, so wird $L$ größer und durch Querkontraktion der Querschnitt $A$ verringert. Die verwendeten Dehnmessstreifen besitzen metallische Leiter, bei denen der spezifische elektrische Widerstand nahezu konstant ist [Schrüfer 95]. Somit ergibt sich die relative Widerstandsänderung $\Delta R$ zu

$$\frac{\Delta R}{R_0} = \frac{\Delta L}{L_0} - \frac{\Delta A}{A_0} \ . \tag{5.10}$$

Dabei ist der Quotient $\Delta L/L_0$ die Dehnung $\varepsilon$. Wird anstelle des Querschnittes $A$ der Durchmesser $D$ eingeführt, so geht (5.10) über in

$$\frac{\Delta R}{R_0} = \varepsilon - 2\frac{\Delta D}{D_0} \ . \tag{5.11}$$

Das Verhältnis $\Delta D/D_0$ wird als Querdehnung $\varepsilon_q$ bezeichnet. Die Widerstandsänderung kann nun auf die Dehnung $\varepsilon$ bezogen werden. Es gilt

$$\frac{\Delta R/R_0}{\varepsilon} = 1 - 2\frac{\varepsilon_q}{\varepsilon} \ . \tag{5.12}$$

Das Verhältnis $\varepsilon_q/\varepsilon$ ist die Poisson-Zahl $\mu$ und ist eine Materialkonstante [Stroppe 88]. Folglich ist auch die linke Seite eine Materialkonstante, die die relative Widerstandsänderung bei einer bestimmten Dehnung bezeichnet

$$k = 1 - 2\mu \ . \tag{5.13}$$

Die Widerstandsänderung ist demzufolge direkt abhängig von der Dehnung und einer Materialkonstanten

$$\frac{\Delta R}{R_0} = k \cdot \varepsilon \tag{5.14}$$

und somit direkt proportional zur mechanischen Verformung des Verformkörpers.

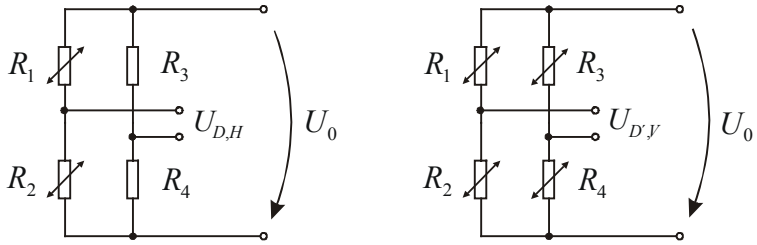



Abbildung 5.8: Messbrücken

Die Widerstandsänderung muss noch in eine für den Analog-Digital-Umsetzer geeignete Größe umgesetzt werden. Dazu wird mit Hilfe einer Messbrücke die Widerstandsänderung in eine Spannungsänderung umgesetzt. Da bei den Verformkörpern jeweils gegensinnige Stauch- und Dehnungszonen vorhanden sind, können an diesen Zonen Dehnmessstreifen angebracht werden, die sich in einer Halb- oder Vollbrücke zusammenschalten lassen. Abbildung 5.8 zeigt die verwendeten Brückenschaltungen. Als Dehnmessstreifen werden zwei identische Typen eingesetzt, so dass der Normalwiderstand und die relative Widerstandsänderung bei Dehnung gleich sind. Es gilt somit $R_{1,0}=R_{2,0}=R_0$. Die Dehnmessstreifen werden so appliziert, dass sich der Wert von $R_1$ mit $R_1=R_0-\Delta R$ bei positiver Kraftrichtung verringert und der von $R_2$ mit $R_2=R_0+\Delta R$ vergrößert. Die anderen beiden Widerstände bei der Halbbrücke werden mit $R_3=R_4=R$ ebenfalls gleich gewählt, so dass die Brückenspannung im unbelasteten Zustand Null ergibt. Für die Halbbrücke ergibt sich die Brückenspannung zu

$$U_{D0,H} = \left( \frac{R_0 + \Delta R}{2R_0 + \Delta R - \Delta R} - \frac{R}{2R} \right) \cdot U_0 = \frac{1}{2} \frac{\Delta R}{R_0} \cdot U_0 \ . \tag{5.15}$$

Für die Vollbrücke gilt

$$U_{D0,V} = \left( \frac{R_0 + \Delta R}{2R_0 + \Delta R - \Delta R} - \frac{R_0 - \Delta R}{2R_0 - \Delta R + \Delta R} \right) \cdot U_0 = \frac{\Delta R}{R_0} \cdot U_0 . \tag{5.16}$$

Damit hängen die Änderungen der Brückenspannung direkt von den Widerstandsänderungen ab. Da die Widerstandsänderungen in der Größenordnung von $10^{-5}$ bis $10^{-3}$ sehr klein sind, können Temperatureinflüsse eine Widerstandsänderung hervorrufen, die das Messergebnis verfälschen könnte. Die Temperaturabhängigkeit ist mit $R=(R_0+\Delta R)(1+\alpha_T \Delta T)$ für kleine Temperaturdifferenzen gegeben. Dabei ist $\alpha_T$ der Temperaturkoeffizient und $\Delta T$ die Temperaturdifferenz. Für die beiden Brückenschaltungen können die Formeln (5.15) und (5.16) um die Temperaturabhängigkeit ergänzt werden. Es gilt

$$U_{D,H} = \left( \frac{(R_0 + \Delta R)\left(1 + \alpha_{T,R_{1,2}} \cdot \Delta T\right)}{(2R_0 + \Delta R - \Delta R)\left(1 + \alpha_{T,R_{1,2}} \cdot \Delta T\right)} - \frac{R \cdot \left(1 + \alpha_{T,R_{3,4}} \cdot \Delta T\right)}{2R \cdot \left(1 + \alpha_{T,R_{3,4}} \cdot \Delta T\right)} \right) \cdot U_0 \tag{5.17}$$

und

$$U_{D,V} = \left( \frac{(R_0 + \Delta R)\left(1 + \alpha_{T,R_{1,2}} \cdot \Delta T\right)}{(2R_0 + \Delta R - \Delta R)\left(1 + \alpha_{T,R_{1,2}} \cdot \Delta T\right)} - \frac{(R_0 - \Delta R)\left(1 + \alpha_{T,R_{3,4}} \cdot \Delta T\right)}{(2R_0 - \Delta R + \Delta R)\left(1 + \alpha_{T,R_{3,4}} \cdot \Delta T\right)} \right) \cdot U_0 . \quad (5.18)$$

In den Gleichungen (5.17) und (5.18) ist zu sehen, dass sich die temperaturabhängigen Änderungen gegenseitig aufheben, so dass diese Brücken temperaturkompensiert sind [Haug 93]. Trotz dieser messtechnisch günstigen Bedingungen ist es in der Praxis möglich, dass die Brücken aufgrund von Fertigungstoleranzen der Bauteile oder durch Ungenauigkeiten bei der Aufbringung der Dehnmessstreifen nicht ganz symmetrisch sind. Auch sollten die Kriecheigenschaften des Dehnmessstreifen mit denen des Materials des Verformkörpers abgestimmt sein.

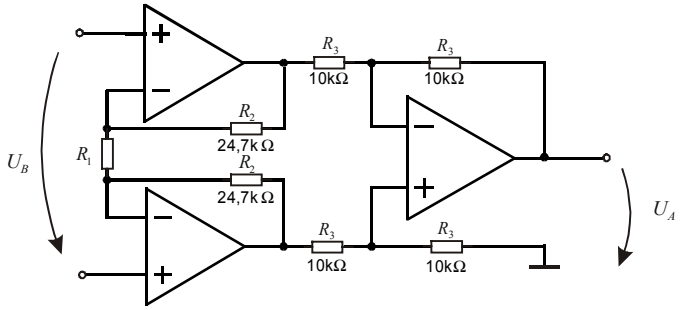



Abbildung 5.9: Instrumentationsverstärker für die Dehnmessstreifenbrücke

Am Brückenausgang liegt die der Dehnung proportionale Spannung an. Diese bewegt sich im Bereich der $10^{-5}$ bis $10^{-3}$fachen Brückenspannung. Die Brückenspannung muss hochohmig abgegriffen werden, damit die Messbrücke als unbelasteter Spannungsteiler erhalten bleibt. Dazu kann ein Instrumentationsverstärker benutzt werden, der in Abbildung 5.9 dargestellt ist [Tietze, Schenk 99], [Heimann et al. 98]. Bei diesem Verstärkertyp sind beide Eingänge sehr hochohmig, so dass keine das Messergebnis verfälschenden Querströme fließen. Bei dem verwendeten integrierten Instrumentationsverstärker AD620 liegt der Eingangswiderstand bei 10 GΩ [Analog 95]. Der Verstärker misst eine Spannungsdifferenz und ist in der Eingangsstufe symmetrisch. Die Schaltung besteht aus zwei Teilen. Das sind Verstärker und Impedanzwandler sowie ein Subtrahierer. Die Verstärkung des ersten Teils lässt sich durch getrennte Berechnungen der Einzelverstärkungen für beide Eingänge ermitteln und dann überlagern. Die Eingangsspannung des oberen Verstärkers in Abbildung 5.9 sei $U_1$ und die des unteren Verstärkers $U_2$. Die gemessene Brückenspannung ist die Spannungsdifferenz $U_D$=$U_2$-$U_1$. Die Ausgangsspannung des oberen Verstärkers ergibt sich mit

$$U_1' = -\frac{R_1 + R_2}{R_1} \cdot (U_2 - U_1) + U_2 \quad (5.19)$$

und die des unteren Verstärkers mit

$$U_2' = \frac{R_1 + R_2}{R_1} \cdot (U_2 - U_1) + U_1 . \quad (5.20)$$

Für die verstärkte Spannungsdifferenz gilt somit

$$U_2' - U_1' = \left(1 + 2\frac{R_2}{R_1}\right) \cdot U_D \ . \tag{5.21}$$

Der zweite Teil der Schaltung ist ein Subtrahierer. Da alle Widerstände des Subtrahierers gleich sind ($R_3$), gibt es weder eine Gewichtung der einzelnen Operanden noch eine Verstärkung. Für den Subtrahierer gilt folglich

$$U_A = U_2' - U_1' \ . \tag{5.22}$$

Durch Einsetzen von (5.22) in (5.21) ergibt sich die Gesamtverstärkung des Instrumentationsverstärkers zu

$$U_A = \left(1 + 2\frac{R_2}{R_1}\right) \cdot U_D \, . \tag{5.23}$$

Somit hängt die Verstärkung nur von den Widerständen $R_1$ und $R_2$ ab. Da die Widerstände $R_2$ aus Symmetriegründen bereits fest in den Schaltkreis integriert worden sind, lässt sich die Verstärkung $V_I$=$U_A$/$U_D$ nur mit Hilfe des Widerstandes $R_1$ einstellen. Es gilt

$$R_1 = \frac{2 \cdot R_2}{V_I - 1} \ . \tag{5.24}$$

Bei einer angenommenen maximalen Differenzspannung mit $10^{-3} \cdot U_B$ muss eine Verstärkung von 500 angesetzt werden, um maximal je eine halbe positive oder negative Brückenspannung als Ausgangsspannung $U_A$ zu erhalten. Der Widerstand $R_1$ muss demzufolge folgendermaßen dimensioniert werden

$$R_1 = \frac{2 \cdot 24{,}7\,\mathrm{k\Omega}}{500 - 1} = 99\ \Omega \, . \tag{5.25}$$

Mit dieser Verstärkung kann eine Auflösung von ca. 0,1 V/N erreicht werden.

Messbrücke und Messverstärker benötigen eine Spannungsversorgung. Die Versorgungsleitungen und die Signalleitungen werden entlang des Beines bis zum Kraftsensor geführt. Aufgrund dieser langen Leitungen ist es möglich, dass Störsignale eingekoppelt werden, die die Messsignale verfälschen können. Um das Auftreten bzw. die Wirkung derartiger Fremdsignale zu unterdrücken, können mehrere Maßnahmen ergriffen werden. Diese sind:

a) passive Maßnahmen
   Die Leitungen zum Kraftsensor wurden abgeschirmt, damit eingekoppelte Störungen abgeleitet werden können. Innerhalb der Abschirmung sind die Leitungen parallel geführt, so dass eventuell auftretende Störungen symmetrisch eingekoppelt werden und sich somit kompensieren können.

b) aktive Maßnahmen
   Zur Unterdrückung der Wirkung von möglichen Störsignalen ist es zunächst wichtig, die Versorgungsspannung frei von Störungen zu halten. Dazu wird die Versorgungsspannung für Messverstärker und Messbrücke gefiltert und elektronisch geregelt, was zur Unterdrü-

ckung möglicherweise eingekoppelter Signale führt. Die Brückenspannung wird möglichst groß gewählt, so dass das Nutzsignal gegenüber möglichen Störungen groß ausfällt.

Nach der Verstärkung der Sensorsignale stehen diese am Ausgang des Messverstärkers niederohmig bereit. Durch die relativ niederohmige Eingangsbeschaltung am Controllereingang wird die Wirkung von Störungen vermindert. Zusätzlich wird für die Übertragung des Messsignals ein möglichst großer Signalbereich gewählt, der am Controllereingang über einen Spannungsteiler auf den für den Analog-Digital-Umsetzer geeigneten Bereich vermindert wird. Auch die Signale der Kraftsensoren werden gefiltert.

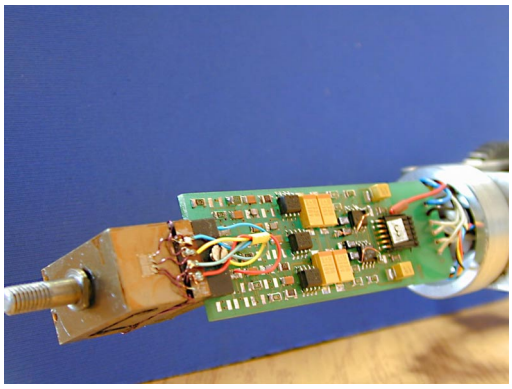


Abbildung 5.10: Kraftsensor mit integrierter Elektronik

Abbildung 5.10 zeigt den geöffneten Unterschenkel des Laufroboters Katharina, in den der gesamte Kraftsensor einschließlich Elektronik integriert ist. Links im Bild ist der Verformkörper zur Messung der Kraftkomponente entlang des Unterschenkels mit einem applizierten Dehnmessstreifen zu sehen. Die Sensorelektronik ist unmittelbar hinter diesem Verformkörper an der Kraftsensormechanik befestigt. Im vorderen Bereich auf der Sensorplatine sind zunächst die Widerstände für die Halbbrücken und die Einstellung der Verstärkung der Instrumentationsverstärker platziert, danach folgen die drei Verstärkerschaltkreise. Im hinteren Teil der Platine befinden sich die Spannungsaufbereitung für Messbrücke und Messverstärker, sowie ein Steckverbinder zum Anschluss der Zuleitungen.

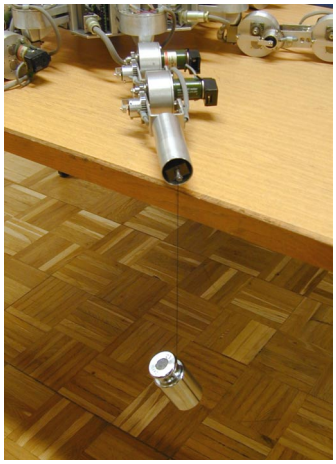
Da die Parameter des Sensors von Fertigungstoleranzen aller verwendeten Bauteile, der exakten Positionierung der Dehnmessstreifen und der Verbindung des Dehnmessstreifens mit dem Verformkörper abhängen, ist eine Kalibrierung des Sensors notwendig. Mit einer Kalibrierung des Sensors ist es dann möglich, solche Abweichungen auszugleichen. Grundsätzlich stehen für das Kalibrieren eines solchen Sensors mehrere Methoden zur Verfügung. Das sind

- Kalibrierung durch Nullpunktabgleich und Einstellen des Verstärkungsfaktors
- Kalibrierung durch Offset und Skalierungsfaktor per Software und
- Kalibrierung von Nullpunkt und Verstärkung durch Programmierung; Rückführung über Digital-Analog-Umsetzer.

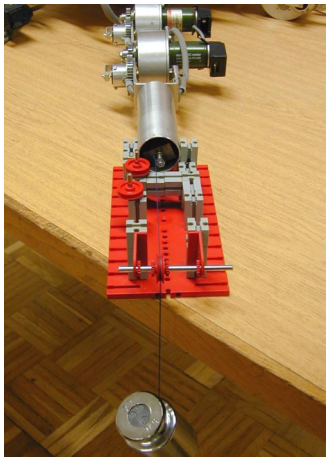
Die erste Methode bietet den Vorteil, dass sich jeder Sensor so einstellen lässt, dass er sich nach vorgegebenen Parametern verhält und sich so alle Sensoren gleich verhalten. Dazu ist es notwendig, alle Sensoren mit Einstellmöglichkeiten zu versehen. Da jedoch bei einer Laufmaschine insbesondere in den Beinen Erschütterungen und Vibrationen zu erwarten sind, die potentiell eine Verstimmung der Einstellwiderstände hervorrufen könnten, wurde auf diese Methode verzichtet. Die zweite Methode ist besonders bei kleinen Korrekturen vorteilhaft, wenn durch die vorhandene Verstimmung die Messbereiche und die Auflösung bzw. die Ge-

nauigkeit nicht unzulässig beeinträchtigt wird. Die dritte Methode kombiniert die Möglichkeiten der beiden vorgenannten Methoden, indem die Einstellung ohne bewegte mechanische Teile vorgenommen wird. Die Messbrücke wird durch eine zusätzliche, mittels Digital-Analog-Umsetzer erzeugte Spannung verstimmt oder der Verstärkungsfaktor wird elektronisch eingestellt. Diese ideal erscheinende Lösung erfordert einen hohen Aufwand. Für die Rückführung sind Digital-Analog-Umsetzer notwendig. Die Auflösung des Umsetzers muss groß genug sein (Quantisierung) und die Schaltung muss entsprechend ergänzt und angepasst werden. Ein anderes Problem ist die hohe Anzahl von erforderlichen Datenleitungen.

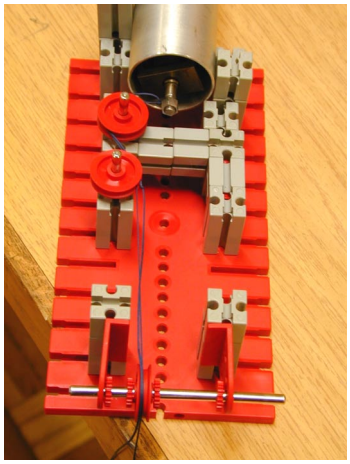
Um die beschriebenen Nachteile zu minimieren, wird eine Lösung durch Kombination der ersten beiden Varianten gewählt. Die Kalibrierung des Nullpunktes und der Verstärkung wird mittels Festwiderstände vorgenommen. Dabei können geringe Abweichungen des Nullpunktes und der Verstärkung toleriert werden. Diese sind auf Softwareseite durch einen Offset und Anpassung der Übertragungsfaktoren zu kompensieren. Somit wird eine zweistufige Kalibrierung eingesetzt, die aus einer Grobkalibrierung mittels Festwiderständen und einer Feinkalibrierung mittels Software besteht.



a) Komponente $F_{X_2}$



b) Komponente $F_{X_3}$



c) Komponente $F_{X_1}$

Abbildung 5.11: Kalibrierung des Kraftsensors des Laufroboters Katharina

Entscheidend für eine gute Kalibrierung ist neben der Einstellmöglichkeit auch der Vergleich mit einer Referenzgröße. Bei dem benutzten Sensor ist die geeignete Referenzgröße eine Kraft. Beim vorliegenden Mehrkomponentenkraftsensor muss die Referenzkraft in einer definierten Raumrichtung wirken und die betreffende Sensorkomponente eingestellt werden. Gleichzeitig kann auf Übersprechen auf andere Sensorkomponenten geprüft und gegebenenfalls bewertet werden. Eine definierte Referenzkraft lässt sich mit Hilfe von Wägestücken erzeugen, da die wirkende Gewichtskraft aufgrund bekannter Masse ebenfalls bekannt ist. Um eine definiert gerichtete Kraft zu erzeugen, kann mittels Umlenkrollen die Kraftrichtung geändert werden. Die notwendige Gegenkraft wird durch Teile der Roboterkonstruktion aufgenommen, die keinen Einfluss auf die Kraftmessung haben. Abbildung 5.11 zeigt den Aufbau beim Kalibrieren. Der Roboter ist auf einer horizontalen Ebene mit gestreckten Beinen gelagert, so dass die Achse des Kraftsensors ebenfalls horizontal ist, was durch einen Parallelreißer geprüft werden kann. Die Gewichtskraft des Wägestückes kann nun direkt vertikal (Abbildung 5.11a) oder über Umlenkrollen horizontal (Abbildung 5.11b und c) angreifen.

### 5.1.3 Stellgrößenausgabe

Die Stellgrößenausgabe realisiert die Ansteuerung der Motoren. Bei autonomen mobilen Robotern stehen bei der Motorenansteuerung zwei wichtige Aspekte im Vordergrund. Zum einen ist es wichtig, die Motoren möglichst energieeffizient anzusteuern, zum anderen müssen alle Steuerungskomponenten in den Roboter integriert sein. Daher ist die Motoransteuerung mit wenigen Bauteilen möglichst effizient zu gestalten.

#### 5.1.3.1 Grundlegende Methoden der Motoransteuerung

Für die Motoransteuerung stehen verschiedene grundlegende Varianten zur Verfügung, deren Blockschemen in Abbildung 5.12 dargestellt sind. Die in den jeweiligen Verstärkern realisierte Motoransteuerung durch Spannungsteilung ist in Abbildung 5.13 zu sehen.

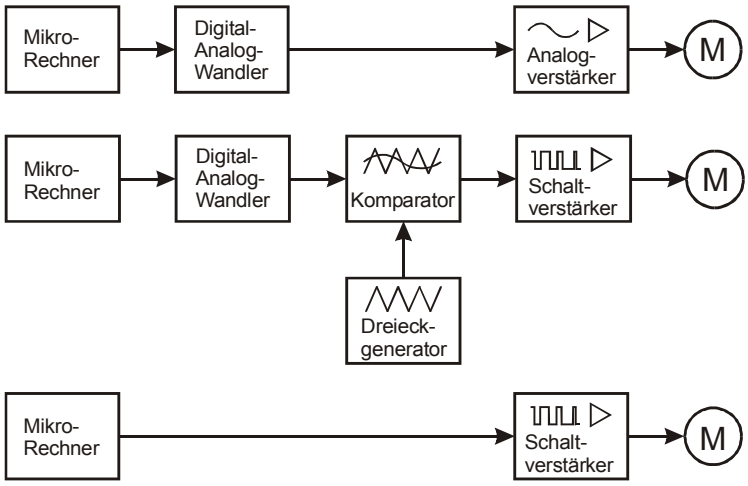



Abbildung 5.12: Varianten zur Motoransteuerung

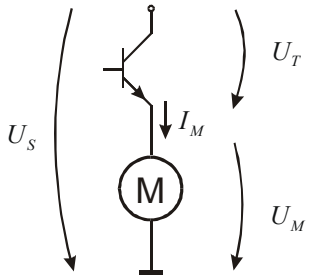



Abbildung 5.13: Motoransteuerung als Spannungsteiler

Die erste Variante in Abbildung 5.12 ist die Digital-Analog-Wandlung des Stellwertes, der an den Leistungsverstärker weitergegeben wird. Dieser kann den Motor mit der gewünschten Spannung ansteuern. Nachteilig hieran ist, dass sehr viel Energie im Verstärker in Wärme umgesetzt wird. Die Gesamtspannung $U_S$ teilt sich in die Spannung über den Transistor $U_T$ und die Spannung über den Motor $U_A$ auf, wobei der Transistor stellvertretend für den Leistungsverstärker steht. Die im Motor umgesetzte Leistung ist $P_M = U_A \cdot I_A$ und die im Transis-

tor umgesetzte Leistung beträgt $P_T = U_T \cdot I_A$. Bei Ansteuerung des Motors mit der halben Speisespannung fällt der gleiche Betrag auch über den Transistor ab. In diesem Fall wird die gleiche Leistung im Motor und im Transistor umgesetzt. Es gilt

$$P_T = P_M = \frac{1}{2} U_S \cdot I_A \ . \tag{5.26}$$

Da in der Regel eine Geschwindigkeit größer Null und kleiner der Maximalgeschwindigkeit benötigt wird, muss der Motor stets mit einer Spannung aus dem mittleren Bereich betrieben werden. Da so ständig eine große Verlustleistung im Verstärker anfällt, ist diese Variante für autonome Systeme ungeeignet.

In der zweiten Variante in Abbildung 5.12 wurde der analoge Leistungsverstärker durch einen Schaltverstärker ersetzt. Die erforderliche Schaltfolge für den Verstärker wird mittels eines Dreieckgenerators und eines Vergleichers erzeugt. Der Transistor im Verstärker ist entweder voll geöffnet oder voll geschlossen. Im voll geöffneten Zustand fließt der größte mögliche Strom, während die über den Transistor abfallende Spannung nahezu Null ist. Im voll geschlossenen Zustand fließt kein Strom, während die volle Spannung über den Transistor abfällt. In diesen beiden Fällen wird im Idealfall keine Leistung im Transistor umgesetzt. Diese zwei Extremfälle werden für den Betrieb des Schaltverstärkers genutzt. Es wird in sehr kurzer Folge zwischen den beiden Zuständen umgeschaltet. Bei dieser Methode wird die Trägheit des Motors ausgenutzt, so dass sich eine mittlere Drehzahl einstellt. Es ist jedoch ein großer Aufwand notwendig, um erst das analoge Signal und daraus das Schaltsignal zu erzeugen. Das widerspricht der Forderung nach einer einfachen und in einen mobilen Roboter integrierbaren Schaltung, die für alle eingebauten Antriebe notwendig ist.

In der dritten Variante in Abbildung 5.12 wird die Schaltfolge für den Verstärker direkt durch den Mikrocontroller erzeugt. Der Vorteil besteht darin, dass der Digital-Analog-Wandler und der Schaltsignalerzeuger eingespart werden können. Außerdem ist der Schaltverstärker energieeffizient. Der Nachteil an dieser direkten Methode ist, dass beim Erzeugen des Signals die elektrischen Effekte des Antriebes mit beachtet werden müssen. Ferner geht die standardisierbare analoge Schnittstelle verloren. Die Vorteile durch die Einfachheit der Hardware überwiegen bei diesem Verfahren, so dass es für die Motoransteuerung für den Laufroboter Katharina ausgewählt wurde.

#### 5.1.3.2 Generierung der Puls-Weiten-Modulation (PWM)

Zur Ansteuerung des Leistungsverstärkers kann entweder eine Puls-Dauer-Modulation oder eine Puls-Weiten-Modulation (PWM) verwendet werden. Bei der ersten Variante werden Pulse fester Länge erzeugt und die Abstände zwischen den Pulsen variiert, bei der Puls-Weiten-Modulation hingegen bleibt die Periodendauer der aufeinanderfolgenden Pulse $T_{PWM}$ konstant und die Pulsweite $T_{on}$ wird verändert. Die PWM besitzt Vorteile bezüglich des elektrischen Verhaltens und wird deshalb zur Ansteuerung der Antriebe verwendet [Riefenstahl 00].

Für die Ansteuerung des Leistungsverstärkers, der als Schaltverstärker arbeitet, müssen die Schaltsignale der PWM direkt vom Mikrocontroller erzeugt werden. Das Erzeugen von Schaltfolgen ist im Prinzip eine Aufgabe von Software. Jedoch unterstützen dies häufig moderne, für den Einsatz in eingebetteten Systemen vorgesehene Mikrocontroller durch in die

Hardware implementierte Funktionen. Diese Funktionen können als Mikrocode realisiert sein, der durch spezialisierte Sub-Prozessoren ausgeführt wird und dessen Parameter konfigurierbar sind [Intel 95]. Des weiteren kann ihre Realisierung auch durch fest programmierte Funktionen erfolgen [Infineon 00].

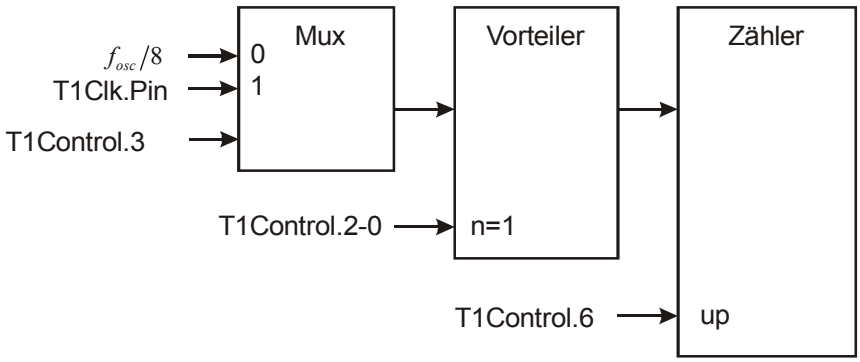



Abbildung 5.14: Prinzip der Erzeugung des Zähler- bzw. des Zeitgebertaktes

Das Verhalten des Motors hängt sowohl von der Periodendauer als auch vom Tastverhältnis ab. Eine genaue Einhaltung dieser beiden Parameter ist für die gewünschte Funktion des Motors erforderlich. Die Erzeugung der PWM ist bei weitem nicht die einzige Aufgabe, die ein Prozessor zu bearbeiten hat. Hier ist eine ereignisgesteuerte Taskbearbeitung notwendig. Dazu muss zunächst ein zentraler Zeitgeber implementiert werden, mit dessen Hilfe zu bestimmten Zeiten die gewünschten Tasks aktiviert werden können. Solche Zeitgeber sind in der Regel in ein- oder mehrfacher Ausführung in Mikrocontrollern vorhanden. Beim Intel 80196KR besteht die Möglichkeit, entweder einen externen Takt oder einen von der Oszillatorfrequenz abgeleiteten Takt zu nutzen (Abbildung 5.14). Über einen 3-Bit Vorteiler wird dann die eigentliche Basisfrequenz für den Zeitgeber abgeleitet. Bei einem Prozessortakt $f_{osc}$=16 MHz und einem Vorteiler von 1 ist die Basisfrequenz

$$f_b = \frac{f_{osc}}{8 \cdot n_{Vorteiler}} = \frac{16\ \text{MHz}}{8 \cdot 1} = 2\ \text{MHz} \tag{5.27}$$

bzw. ergibt sich eine Periodendauer von 0,5 µs. Sollen für die PWM 256 Einstellwerte zur Verfügung stehen, so ergibt sich eine PWM-Frequenz $f_{\text{PWM}}$ von

$$f_{PWM} = \frac{f_b}{n_{PWM}} = \frac{2\ \text{MHz}}{256} = 7{,}8\ \text{kHz} \tag{5.28}$$

bzw. eine Periodendauer von 128 µs. Während einer solchen Periode wird ein Zyklus von Ein- und Ausschaltphase durchlaufen, wie er in Abbildung 5.15 dargestellt ist.

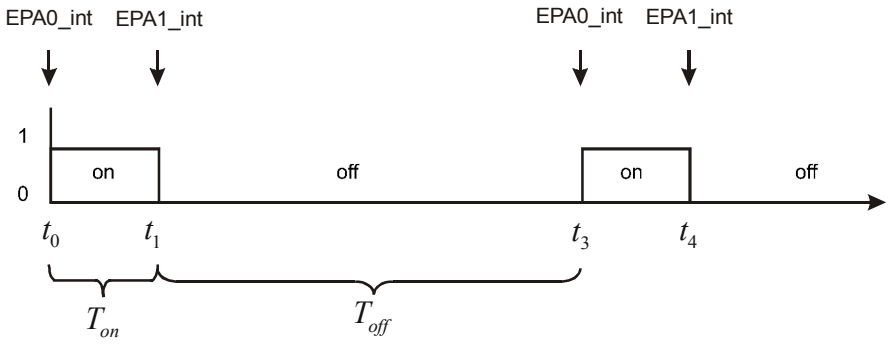



Abbildung 5.15: Zeitdiagramm zur PWM-Erzeugung

Eine Möglichkeit zum Erzeugen des PWM-Signals besteht darin, dass zu den Ein- und Ausschaltzeitpunkten mit Hilfe des Timers ein Interrupt erzeugt wird und das Umschalten des Signals durch Programmcode in der Interrupt-Serviceroutine erfolgt. Interrupts werden gegenüber dem normal auszuführenden Code mit einer höheren Priorität behandelt. Probleme können insbesondere dann auftreten, wenn verschiedene Anforderungen mit einer hohen Priorität behandelt werden müssen. Sind die Anforderungen an die Reaktionszeit bekannt und in Grenzen akzeptierbar, so kann durch Vergabe von unterschiedlichen Prioritäten an die einzelnen Interrupts eine Reaktion innerhalb einer Zeitschranke garantiert werden. Eingebettete Systeme müssen jedoch oft harten Echtzeitbedingungen genügen, so dass zumindest eine Teilreaktion mit möglichst geringer Verzögerung erfolgen muss, die bestimmte Zeitgrenzen nicht überschreiten darf. Solche Restriktionen sind insbesondere dann einzuhalten, wenn damit überlebenswichtige oder zeitkritische Reaktionen verbunden sind. Beim Antrieb eines Gelenkes des Laufroboters Katharina besteht die überlebenswichtige Funktion darin, dass jeder der eingebauten Motoren korrekt angesteuert werden muss, um die Stabilität des Roboters zu gewährleisten. Bei der Ansteuerung mittels PWM bedeutet das, dass das Verhältnis zwischen Ein- und Ausschaltzeit korrekt eingehalten werden muss. Da die Auflösung durch die Stellwerte der PWM im µs-Bereich liegt, machen sich die Latenzzeiten von einigen µs deutlich bemerkbar.

Die PWM wird dadurch erzeugt, dass ein Ausgangssignal periodisch eingeschaltet und nach einer gewissen Dauer wieder ausgeschaltet wird. Maßgeblich hierbei ist die Periodendauer $T_{PWM}$ und die Einschaltzeit $T_{on}$. Der Anteil der Einschaltzeit an der Gesamtzeit ist dabei das Tastverhältnis

$$\beta_{PWM} = \frac{T_{on}}{T_{PWM}} \ . \tag{5.29}$$

Die Periodendauer ergibt sich entsprechend Abbildung 5.15 mit

$$T_{PWM} = T_{on} + T_{off} \ . \tag{5.30}$$

Ein Beispiel soll das Funktionsprinzip näher erläutern. Während einer Periode wird das Ausgabesignal zum Zeitpunkt $t_0$ auf logisch 1 gesetzt und zum Zeitpunkt $t_1=t_0+T_{on}$ wieder auf logisch 0 zurückgesetzt. Die nächste Periode beginnt dann bei $t_2=t_0+T_{PWM}$, wobei das Signal wieder auf logisch 1 gesetzt wird. Soll das Tastverhältnis nicht verändert werden, so gilt Gleichung (5.30) und somit $t_3=t_1+T_{\mathrm{PWM}}$. Sollte das Tastverhältnis jedoch verändert werden, so wäre $t_3=t_2+T_{\mathrm{on}}$ zu setzen. Es stehen zwei Regelsätze zur Verfügung. Das sind

a) für den kontinuierlicher Betrieb ohne Änderung des Tastverhältnisses

$$t_{on,neu} = t_{on} + T_{PWM} \tag{5.31}$$

$$t_{off,neu} = t_{off} + T_{PWM} \quad \text{und} \tag{5.32}$$

b) für die Änderung des Tastverhältnisses

$$t_{on,neu} = t_{on} + T_{PWM} \tag{5.33}$$

$$t_{off,neu} = t_{on,neu} + T_{on} \ . \tag{5.34}$$

Dabei sind $t_{on}$ und $t_{on,neu}$ die alte und neue Einschaltzeit, sowie $t_{off}$ und $t_{off,neu}$ die alte und neue Ausschaltzeit.

Zur Implementierung der PWM kann beim Intel 80C196KR das Event Processor Array genutzt werden. Dieses besteht aus einer Reihe von Registern (EPA0 bis EPA9) und Vergleichern. Wird die Gleichheit von einem der Registerinhalte und einem spezifizierten Timer festgestellt, kann ein Interrupt ausgelöst und ein zugeordnetes I/O-Pin in einer festgelegten Weise angesteuert werden. Diese Funktion kann für das Setzen oder Rücksetzen des Ausgabewertes zu einem bestimmten Zeitpunkt an einem Pin genutzt werden. Es besteht jedoch noch die Notwendigkeit, diese Zeitpunkte zu bestimmen und die Register des EPA entsprechend zu konfigurieren. Hierzu kann der Peripheral Transaction Server genutzt werden, der eine Reihe von Mikrocodes für Datentransfers von Peripherieoperationen zur Verfügung stellt. Einer der Mikrocodes stellt die Funktion

```
A := A + B
```

bereit, womit sich die Operationen in Gleichung (5.31) und (5.32) ausführen lassen. PTS-Operationen werden als Interrupt ausgeführt. Da sie in einem eigenen Subprozessor ausgeführt werden, sind weder Stackoperationen noch Registersicherungsmaßnahmen notwendig. Ihre Latenzzeit ist deshalb mit maximal 43 Prozessorzyklen (5,375 µs) deutlich kürzer als bei normalen Interrupts mit 56 Prozessorzyklen (7 µs). Dies sind die theoretisch ungünstigsten Fälle, die bei Nutzung aller verfügbaren Instruktionen möglich sind. Im erzeugten Code für den Laufroboter Katharina fanden jedoch nicht alle denkbaren Instruktionen Verwendung, so dass sich maximale Latenzzeiten von 27 Prozessorzyklen (3,375 µs) für die Interrupts und 16 Prozessorzyklen (2 µs) für PTS-Interrupts ergaben. Der Vorteil der Verwendung des PTS wird hier besonders deutlich. Hinzu kommt, dass durch die Behandlung mittels PTS effektiv keine CPU-Zeit benötigt wird. Jedoch ergeben sich Verzögerungen durch Buszugriffe.

Für die Veränderung des Tastverhältnisses muss noch eine Funktion für Gleichung (5.34) bereitgestellt werden, die nicht durch einen PTS erfüllt werden kann. Diese muss mit Hilfe eines normalen Interrupts bereitgestellt werden. Abbildung 5.16 zeigt die entsprechende Realisierung eines PWM-Signals. Das Register EPA0_Time enthält den Zeitpunkt für die positive Flanke. Wird durch Vergleich mit dem Timer-Wert der Zeitpunkt erkannt, so wird über das EPA das Ausgangssignal auf logisch 1 gesetzt. Gleichzeitig wird ein PTS-Interrupt generiert, der die PWM-Periodendauer zum Register EPA0_Time hinzuaddiert und so den nächsten Einschaltzeitpunkt festlegt. Entsprechend wird auch für den Ausschaltzeitpunkt verfahren, falls das Tastverhältnis konstant bleibt. Ändert es sich jedoch, so wird der PTS einmalig für einen Interrupt deaktiviert und die normale ISR tritt in Funktion, die Gleichung (5.34) realisiert. Mit dieser Methode ergeben sich drei wichtige Vorteile. Diese sind

- Realisierung der verzögerungsfreien Umschaltung des Signals direkt durch den Vergleicher,
- Latenzzeit für den PTS-Interrupt ist kürzer als bei normalen Interrupts und
- Nutzung des PTS verringert die CPU-Last.

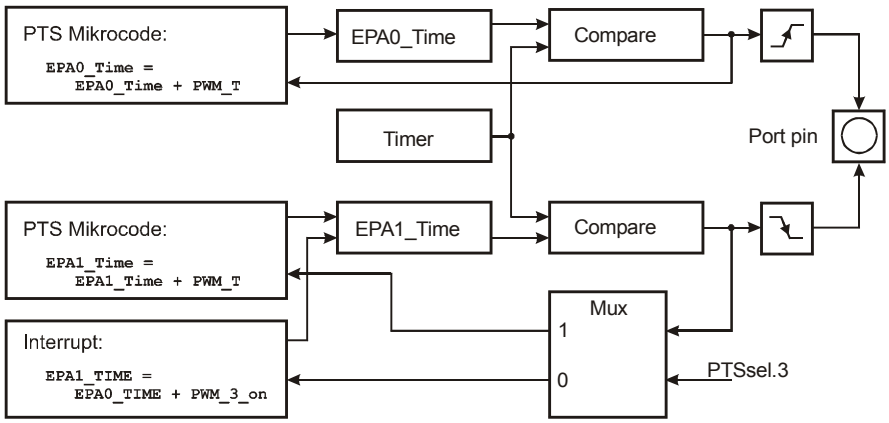



Abbildung 5.16: Realisierung der PWM-Ausgabe

### 5.1.3.3 Leistungsverstärkung

In Realisierung von beiden Bewegungsrichtungen für die Gelenkantriebe beim Laufroboter Katharina werden Schaltverstärker in Vollbrückenschaltung genutzt, dessen Prinzip Abbildung 5.17 zeigt. Die vier Leistungstransistoren V1 bis V4 sind so zusammengeschaltet, dass sie eine Vollbrücke bilden. Die Dioden leiten die Induktionsspannung der Motorwicklung gegen die Versorgungsspannung $U_S$ ab. Zur Ansteuerung der Brücke gibt es zwei grundlegende Methoden. Das sind die Gegenphasen-Ansteuerung (Locked Anti-Phase Control) und Vorzeichen-Betrags-Ansteuerung (Sign/Magnitude Control) [Regan 99].

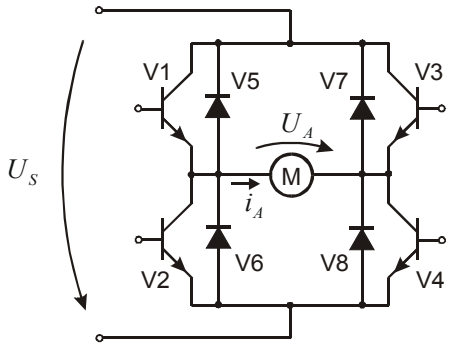



Abbildung 5.17: Prinzip des Brückenverstärkers

Bei der Gegenphasen-Ansteuerung werden die Transistoren V1 und V4 abwechselnd mit den Transistoren V2 und V3 durchgeschaltet. Der Strom wird durch die elektrische Filterwirkung der Motorwicklung geglättet und es stellt sich ein Strommittelwert ein. Diese Ansteuerung setzt voraus, dass die elektrische Zeitkonstante wesentlich größer ist als die Periodendauer der PWM

$$T_{el} \gg T_{PWM} \ . \tag{5.35}$$

Bei der Vorzeichen-Betrags-Ansteuerung lassen sich wiederum zwei Betriebsarten unterscheiden, die gleichzeitige und die alternierende Pulsung [Riefenstahl 00]. Bei der gleichzeitigen Pulsung wird während der Einschaltphase $T_{on}$ entweder das Transistorpaar V1/V4 oder das Transistorpaar V2/V3 geöffnet. Ein besonderes Richtungssignal bestimmt, welches Paar aktiv wird. Während der Zeit $T_{off}$ sind alle Transistoren geschlossen. Bei der alternierenden Pulsung wird das aktive Transistorpaar ebenfalls durch das Richtungssignal bestimmt, jedoch bleibt während der Zeit $T_{off}$ abwechselnd einer der Transistoren des aktiven Paares geöffnet.

Für den Leistungsverstärker im Laufroboter Katharina wurde der Vollbrückentreiber UDN2954W bzw. dessen Nachfolger UDN3951SW eingesetzt [Dewey 00]. Diese lassen sich in Gegenphasen-Ansteuerung und in Vorzeichen-Betrags-Ansteuerung mit gleichzeitiger Pulsung betreiben. Da Bedingung (5.35) nicht erfüllt war, wurde die Schaltung in Vorzeichen-Betrags-Ansteuerung betrieben.

#### 5.1.3.4 Puls-Weiten-Modulation und Gleichstrommotor

Wird ein permanenterregter Gleichstrommotor mit einer Gleichspannung angesteuert, so kann er mit folgenden Gleichungen beschrieben werden

$$u_A = u_{EMK} + R_A \cdot i_A + L_A \cdot \frac{di_A}{dt} \tag{5.36}$$

$$u_{EMK} = k_1 \cdot \omega \tag{5.37}$$

$$M_I = k_2 \cdot i_A \ . \tag{5.38}$$

Dabei ist $u_A$ die Ankerspannung, $R_A$ der ohmsche Ankerwiderstand, $L_A$ die Ankerinduktivität und $u_{EMK}$ die induzierte Spannung, welche über die Motorkonstante $k_1$ direkt proportional zur Drehzahl ist. Das erzeugte Moment $M_I$ ist über die Drehmomentkonstante $k_2$ direkt proportional zum Ankerstrom $i_A$.

Im stationären Fall, d.h. wenn das Antriebssystem eingeschwungen ist, ändert sich die Drehzahl nicht mehr und es treten keine Beschleunigungsmomente in Erscheinung. Das heißt, dass der letzte Term in Gleichung (5.36) verschwindet. In diesem Fall ist das erzeugte Moment $M_I$ konstant. Setzt man (5.37) und (5.38) in (5.36) ein, so kann man durch Umstellung die Drehzahl $\omega$ berechnen

$$\omega = \frac{u_A}{k_1} - \frac{R_A \cdot M_I}{k_1 \cdot k_2} \ . \tag{5.39}$$

Folglich ist der Steigungsteil der Kennlinie linear. Die Drehzahl ist für den Fall Null, dass das erzeugte Drehmoment die Last nicht treiben kann. Abbildung 5.18 zeigt den prinzipiellen Verlauf einer solchen Kennlinie.

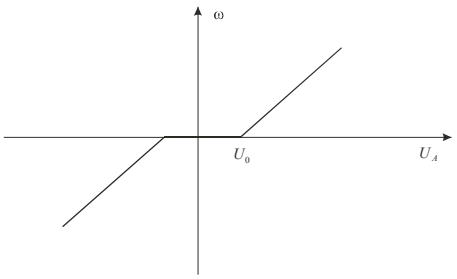
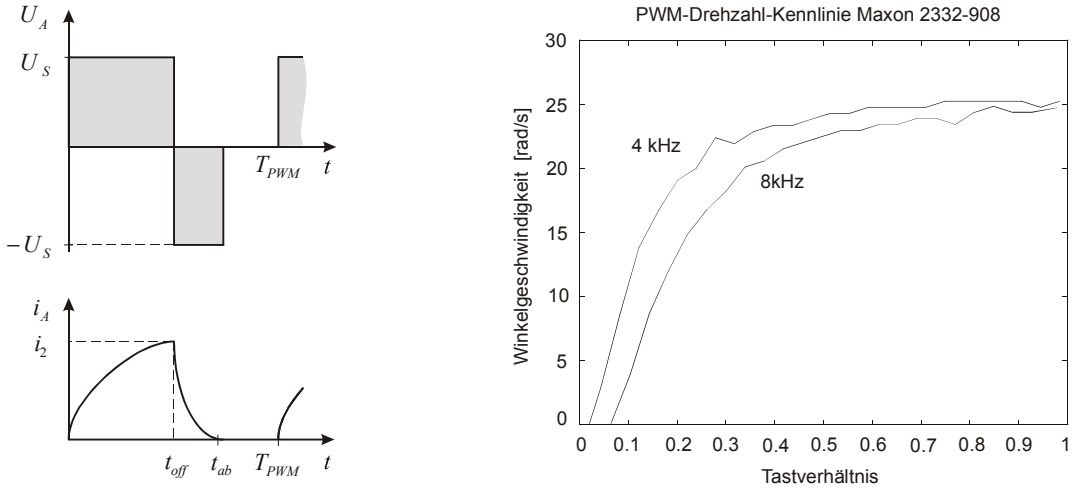



Abbildung 5.18: Ankerspannungs-Drehzahl-Kennlinie bei Gleichstrombetrieb

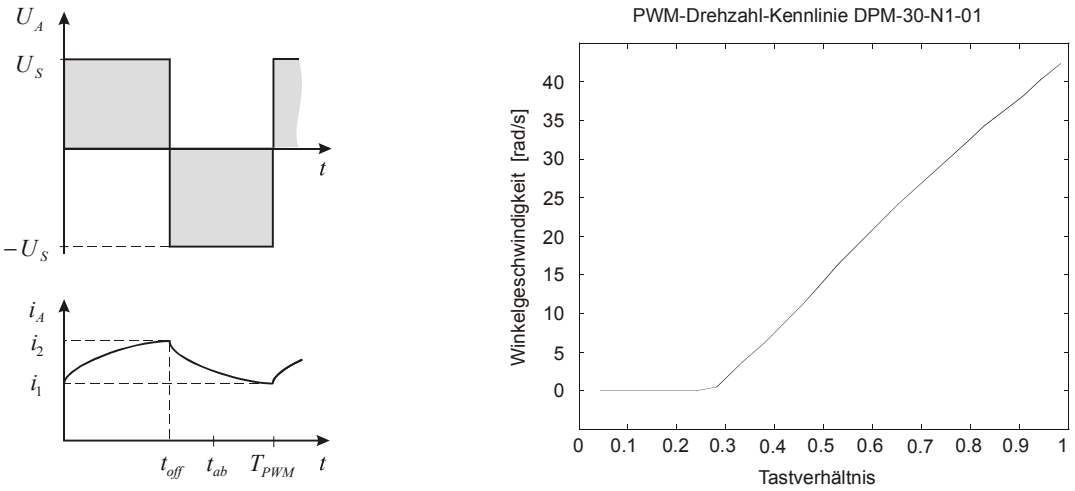
Wird jedoch die Ankerspannung gepulst, so ergeben sich weitere Effekte, so dass der letzte Term in Gleichung (5.36) nicht mehr verschwindet. Das Verhalten wird durch die Ansteuerung des Vollbrückenmotortreibers mit der PWM, dem erzeugten Moment und der Induktivität beeinflusst. Dabei kann der Fall auftreten, dass der Strom während der PWM-Periode wieder den Wert Null annimmt, was als lückender Betrieb bezeichnet wird. Im Zeitintervall $0 \leq t < t_{off}$ der PWM-Periode ist ein Transistorpaar (z.B. V1/V4 in Abbildung 5.17) aktiv und die Brückenspannung $U_S$ liegt am Motor an und der Strom beginnt zu fließen, wie in Abbildung 5.19a dargestellt. Zum Zeitpunkt $t = t_{off}$ sperren die Transistoren. Da es sich bei Motoren um ohmsch-induktive Lasten handelt, an denen der Strom nicht springen kann, fließt dieser durch die Feilaufdioden (z.B. V6/V7 in Abbildung 5.17) gegen die Spannungsquelle. Im Intervall $T_{off} \leq t < t_{ab}$ liegt infolgedessen die negative Spannung $U_S$ am Motor an. Zum Zeitpunkt $t_{ab}$ wird der Strom gleich Null, das heißt, der Strom lückt. Die Dioden sperren, die Spannung bricht zusammen und der Motor ist vom Netz getrennt. Das Stromlücken führt zu dem Effekt, dass die PWM-Drehzahl-Kennlinie keine lineare Steigung besitzt, wie Abbildung 5.19b zeigt.




a) Spannungs- und Stromverläufe b) PWM-Drehzahl-Kennlinie

Abbildung 5.19: Verhalten bei lückendem Betrieb

Nimmt der Strom während der PWM-Periode nicht den Wert Null an, so spricht man von nicht lückendem Betrieb. Dabei ist $t_{ab} > T_{\mathrm{PWM}}$ und folglich fließt Strom während der gesamten Zeit $t_{off} \leq t < T_{\mathrm{PWM}}$, wie in Abbildung 5.20a dargestellt. In diesem Fall stellt sich ein Mittelwert des Stromes ein und die Kennlinie erhält eine lineare Steigung, was Abbildung 5.20b an Hand einer mit dem Laufroboter Katharina aufgenommenen Kennlinie zeigt.




a) Spannungs- und Stromverläufe b) PWM-Drehzahl-Kennlinie

Abbildung 5.20: Verhalten bei nichtlückendem Betrieb

Um ein lineares Verhalten des Systems bei gegebener Last zu erhalten, müssen diese Kennlinien linearisiert werden. Dies kann durch eine inverse Kennlinie erfolgen, z.B. mittels einer Look-Up-Tabelle. Interessiert nur ein Teil der Kennlinie, kann der betreffende Bereich durch eine Gerade approximiert werden. Diese Variante wurde für den Laufroboter Katharina gewählt, da nur der untere und mittlere Drehzahlbereich verwendet wurde.

### 5.1.4 Stromversorgung

Für autonome mobile Roboter ist auch die Energieversorgung von großer Bedeutung. Bei Robotern der 20 kg-Klasse werden elektrische Antriebe verwendet. Neben den Antriebssystemen müssen auch sämtliche Rechnerhardware und alle Sensorsysteme mit elektrischer Energie versorgt werden. Jedes dieser Teilsysteme benötigt unterschiedliche Spannungen und unterschiedlich viel Energie. Dabei dürfen Stromspitzen der Leistungselektronik die Funktion anderer Teilsysteme nicht stören. Da ein mobiler Roboter nicht mit beliebig vielen Spannungsquellen ausgestattet werden kann, müssen alle erforderlichen Spannungen von einer zentralen Spannungsquelle abgeleitet werden. Dazu werden beim Laufroboter Katharina von einer zentralen Spannungsquelle die Spannungen für die Sensoren und die Mikrocontroller mit Hilfe von Transvertern gewonnen. Sie ermöglichen gleichzeitig eine galvanische Trennung der Teilsysteme. So können beispielsweise die analog messenden Sensoren mit einer eigenen, potentialfreien Spannung versorgt werden. Um ein Übersprechen von Leistungssignalen auf die Digital- oder Sensorelektronik zu verhindern, werden diese Schaltungsteile mit Hilfe von Optokopplern galvanisch getrennt. Abbildung 5.21 zeigt schematisch die Stromversorgung und die Trennung der Sensor- und Leistungssignale beim Laufroboter Katharina.

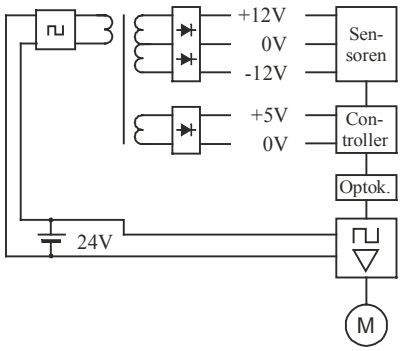



Abbildung 5.21: Stromversorgung und Potentialtrennung beim Laufroboter Katharina

## 5.2 Steuerungsalgorithmen

Steuerungsalgorithmen dienen der Erzeugung und Realisierung von Lauf- und Körperbewegungen von Laufmaschinen. Die in diesem Kapitel beschriebenen Algorithmen sind prinzipiell auf verschiedene Typen sechsbeiniger Roboter anwendbar, sofern sie einen starren Körper und Beine mit einer ähnlichen Kinematik besitzen. Die folgenden Darstellungen erfolgen mit Bezug auf den Laufroboter Katharina.

### 5.2.1 Koordinatensysteme

Zur Steuerung des Laufroboters werden Koordinatensysteme als Bezugsrahmen genutzt, auf die sich die Koordinierung von Bewegungen (Kinematik) und die Darstellung von Sensorinformationen bezieht. Meist werden kartesische Koordinatensysteme verwendet, deren Koordinatenachsen eine orthonormale Basis bilden. Das heißt, dass die Koordinatenachsen paarweise senkrecht aufeinander stehen und die Basisvektoren sämtlich Einheitsvektoren sind. Dadurch können reale geometrische Maße auf die Modellwelt übertragen werden.

#### 5.2.1.1 Definition der Koordinatensysteme

Nachfolgend sollen die Koordinatensysteme näher beschrieben werden, die bei der Beschreibung der Bewegung des Laufroboters und zur Verarbeitung seiner Sensorinformationen benutzt werden. Abbildung 5.22 zeigt die verwendeten Koordinatensysteme am Roboter.

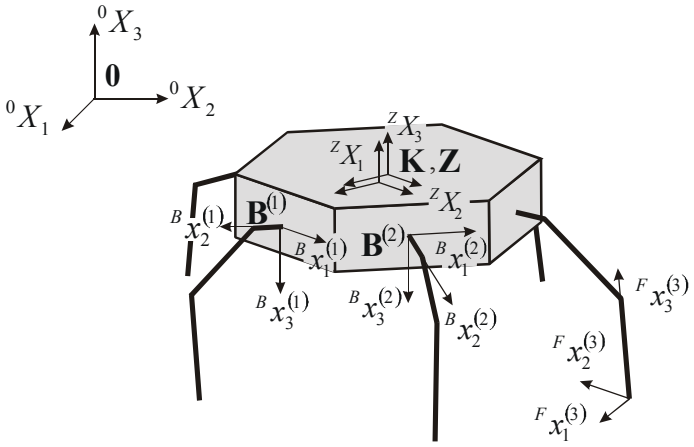



Abbildung 5.22: Koordinatensysteme am Laufroboter Katharina

*Das Weltkoordinatensystem* $\{O; X_1, X_2, X_3\}$

Das Weltkoordinatensystem ist ein kartesisches, erdfestes Koordinatensystem. Das heißt, es ist in der Umwelt des Roboters verankert. Seine Achse $X_3$ ist so gerichtet, dass sie dem Gravitationsvektor antiparallel ist. Die Achsen $X_1$ und $X_2$ liegen in der Horizontalebene. Es gilt

$$^0\vec{X}_3 \uparrow\downarrow \vec{g} \tag{5.40}$$

*Das Körperkoordinatensystem* $\{K; X_1, X_2, X_3\}$

Das Körperkoordinatensystem ist ein mit dem Roboterkörper verbundenes kartesisches Koordinatensystem. Sein Ursprung $K$ liegt in der Mitte des Roboterkörpers und ist bezüglich dem Roboterkörper ortsfest. Die positive Richtung der Koordinatenachse $X_1$ zeigt in Richtung des ersten Beines. Die durch die Koordinatenachsen $X_1$ und $X_2$ aufgespannte Koordinatenebene ist parallel zur Grundfläche des Roboterkörpers. Das Körperkoordinatensystem dient zur Beschreibung der Geometrie des Roboterkörpers und daran befestigter Objekte.

$$^K\vec{X}_3 \parallel \vec{n}_{Grundfläche} \tag{5.41}$$

*Das Zentralkoordinatensystem* $\{Z; X_1, X_2, X_3\}$

Das Zentralkoordinatensystem ist ein kartesisches Koordinatensystem. Es dient als Bezugskoordinatensystem für Bein- und Körperbewegungen und wird zur Lokalisierung des Roboters genutzt. Zentralkoordinatensystem und Körperkoordinatensystem sind deckungsgleich, wenn sich der Körper in der Ausgangsposition befindet.

*Das Beinkoordinatensysteme* $\left\{B^{(i)}; X_1^{(i)}, X_2^{(i)}, X_3^{(i)}\right\}$

Die Beine des Roboters werden durchnumeriert. Ihnen werden die kartesischen Beinkoordinatensysteme $\left\{B^{(i)}; X_1^{(i)}, X_2^{(i)}, X_3^{(i)}\right\}$ mit $i=\{1,2 \ldots 6\}$ für den betrachteten Laufroboter zugeord-

net. Der Koordinatenursprung $B^{(i)}$ liegt jeweils am Beinbefestigungspunkt. Die Richtung der Koordinatenachse $X_3^{(i)}$ ist antiparallel zur Koordinatenachse $X_3$ des Körperkoordinatensystems. In der durch die Koordinatenachsen $X_1^{(i)}$ und $X_3^{(i)}$ aufgespannten Ebene liegt die Seitenfläche des Prismas, an der das Bein befestigt ist. $X_2^{(i)}$ ist demzufolge parallel zum Normalenvektor der Seitenfläche.

$$ {}^{B}\vec{X}_3^{(i)} \uparrow\downarrow {}^{K}\vec{X}_3 \tag{5.42} $$

$$ {}^{B}\vec{X}_2^{(i)} \parallel \vec{n}_{Seitenfläche}^{(i)} \tag{5.43} $$

*Das Gelenkkoordinatensystem* $\left\{q^{(i)};\alpha^{(i)},\beta^{(i)},\gamma^{(i)}\right\}$

Jedes Bein besitzt Gelenkkoordinaten, die die aktiven, rotatorischen Freiheitsgrade beschreiben. Sie werden mit $\alpha^{(i)}$ für das vom Körper aus gesehene erste Gelenk, $\beta^{(i)}$ für das zweite Gelenk und $\gamma^{(i)}$ für das dritte Gelenk bezeichnet. Der Vektor $\vec{q}$ fasst die drei beschriebenen Freiheitsgrade zusammen. [Wloka 92]

*Die Fußkoordinatensysteme* $\left\{F^{(i)};X_1^{(i)},X_2^{(i)},X_3^{(i)}\right\}$

Die Fußkoordinatensysteme $\left\{F^{(i)};X_1^{(i)},X_2^{(i)},X_3^{(i)}\right\}$ mit $i=\{1, 2 .. 6\}$ sind kartesische Koordinatensysteme. Sie haben ihren Ursprung jeweils im Kreuzungspunkt der passiven Freiheitsgrade des jeweiligen Fußes. Die Koordinatenachse $X_3^{(i)}$ ist senkrecht zur dritten Gelenkachse. Die Koordinatenachse $X_1^{(i)}$ ist parallel zur dritten Gelenkachse.

### 5.2.1.2 Beziehungen der Koordinatensysteme zueinander

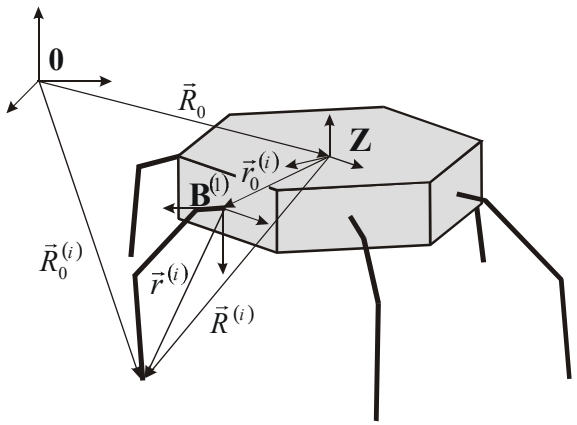



Abbildung 5.23: Beziehungen zwischen den Koordinatensystemen am Laufroboter

Die verschiedenen, im vorherigen Kapitel vorgestellten Koordinatensysteme werden zur Beschreibung bestimmter Bewegungen bzw. Vorgänge des Gesamtsystems Roboter oder eines

Teilsystems eines Roboters benötigt. Zwischen den Koordinatensystemen lassen sich Transformationen ausführen. Abbildung 5.23 zeigt die Lage der verschiedenen Koordinatensysteme zueinander.

Das Weltkoordinatensystem liefert einen konstanten Bezugspunkt zur Bestimmung der Position des Roboters und von Objekten aus seiner Umgebung. Die Position des Roboters wird mit dem Vektor $\vec{R}_0$ beschrieben, der den Koordinatenursprung des Weltkoordinatensystems mit dem Ursprung des Zentralkoordinatensystems verbindet.

Das Zentralkoordinatensystem dient zur Beschreibung der Bewegung und zur Positionierung des Roboters. In ihm werden die Orte der Beinbefestigungspunkte und der Fußpunkte angegeben. Der Vektor $\vec{R}^{(i)}$ verbindet den Koordinatenursprung des Zentralkoordinatensystems mit dem Ursprung des Fußkoordinatensystems, der als Beinende angenommen wird. Dessen Position lässt sich mit Hilfe einer Vektoraddition in Bezug zum Weltkoordinatensystem

$$\vec{R}_0^{(i)} = \vec{R}_0 + \vec{R}^{(i)} \tag{5.44}$$

darstellen.

Die Beinbefestigungspunkte seien durch die Vektoren $\vec{r}_0^{(i)}$ mit dem Zentralkoordinatensystem verbunden. Mindestens drei dieser Vektoren $\vec{r}_0^{(i)}$ bestimmen eindeutig die räumliche Lage des Roboterkörpers gegenüber dem Zentralkoordinatensystem. Die Beinbefestigungspunkte sind identisch mit dem Ursprung der Beinkoordinatensysteme. Diese werden durch die Vektoren $\vec{r}^{(i)}$ mit den Fußkoordinatensystemen verbunden. Der Zusammenhang zwischen den Vektoren lässt sich mit folgender Vektorgleichung ausdrücken

$$\vec{R}^{(i)} = \vec{r}_0^{(i)} + \vec{r}^{(i)} \ . \tag{5.45}$$

Die Gelenkkoordinaten sind über inverse Kinematikbeziehungen mit den kartesischen Beinkoordinaten verbunden, die die Position des Beinendes beschreiben.

Der Übergang vom Fußkoordinatensystem, in dem die Kraftsensormesswerte dargestellt werden, zu einem dem Beinkoordinatensystem parallelen System wird Sensordatentransformation oder im Falle von Kraftmesswerten Krafttransformation genannt.

Das Körperkoordinatensystem ist über zwei Translations- und eine Rotationsoperation mit dem Zentralkoordinatensystem verbunden. Dabei beschreiben die Operationen die Lage des Roboterkörpers gegenüber dem Zentralkoordinatensystem.

### 5.2.2 Grobstruktur des Steuerungssystems

Das Steuerungssystem ist in mehrere Ebenen unterteilt. Abbildung 5.24 gibt einen Überblick zum Aufbau des verwendeten Steuerungssystems.

Die externen Steuerungsebenen, in denen die Steuerung des Roboters aufgabenbezogen erfolgt, befinden sich außerhalb des Roboters. Sie dienen zur Experimentsteuerung durch den Operator und zur Experimentauswertung.

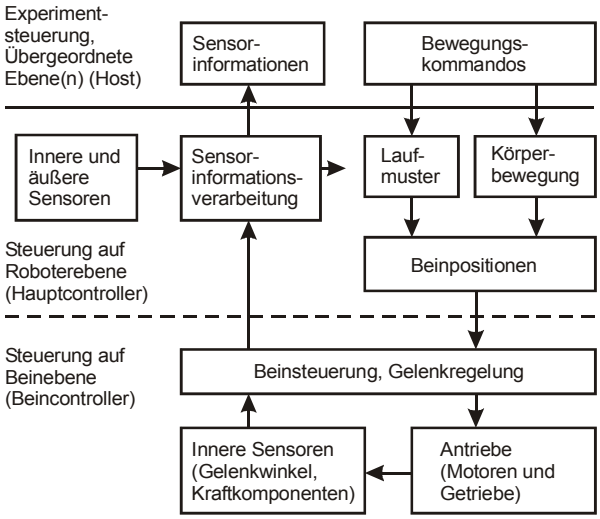



Abbildung 5.24: Grobstruktur des Steuerungssystems

Die in den Roboter implementierte Steuerung unterteilt sich in die Hauptsteuerungsebene und die Beinsteuerungsebene. Die Hauptsteuerungsebene generiert und koordiniert alle notwendigen Beinbewegungen. Sie verarbeitet Kommandos aus der höhergelegenen Steuerungsebene und setzt diese in Beinbewegungen um. Das betrifft sowohl die Laufbewegungen als auch die gewünschten Körperbewegungen. Die Hauptsteuerungsebene erhält Sensorinformationen von der unteren Steuerungsebene zur Koordinierung von Lauf- und Körperbewegung. In der Beinsteuerungsebene werden die von der Hauptsteuerungsebene vorgegebenen Bewegungen umgesetzt, indem die notwendigen Gelenkbewegungen mittels Koordinatentransformationen ermittelt und mit Hilfe des Servoregelsystems realisiert werden. Von den Sensoren werden Informationen über Gelenkpositionen und wirkende Kräfte gewonnen. Sie werden über Koordinatentransformationen miteinander verknüpft, so dass Istwerte, die für die Realisierung der aktiven Nachgiebigkeit erforderlich sind, mit Bezug auf das Beinkoordinatensystem zur Verfügung stehen. Zwischen Beinsteuerungsebene und Hauptsteuerungsebene erfolgt der Datenaustausch mit Bezug auf das Beinkoordinatensystem, dessen Ursprung sich an der Verbindung Bein/Körper befindet.

### 5.2.3 Externe Steuerungsebene

Die externe Steuerungsebene ist extern auf einem Hostrechner implementiert. Sie dient der Experimentsteuerung und der Experimentauswertung.

Mit der Experimentsteuerung ist es möglich, Kommandos und Parameter an den Roboter zu übergeben, so dass sich je nach Bedarf verschiedene Funktionen des Roboters aktivieren lassen. Darüber hinaus können verschiedene Parameter, z.B. Laufgeschwindigkeit, Laufrichtung und Parameter für die aktive Nachgiebigkeit übergeben werden.

Zur Experimentauswertung werden in umgekehrter Richtung aktuelle Zustände und Istwerte übertragen, so dass anhand von Messwerten das Verhalten des Steuerungssystems beobachtet werden kann.

Die externe Steuerungsebene wird hier anstelle von höheren Steuerungsebenen eingesetzt. Die höheren Steuerungsebenen haben allgemein bei mobilen Robotern die Aufgabe, Navigation und Steuerung aufgabenbezogen zu realisieren. Sie sind deshalb nicht spezifisch für Laufroboter [Preumont 94].

### 5.2.4 Hauptsteuerungsebene

Die Hauptsteuerungsebene hat die Aufgabe, Bewegungskommandos aus höheren Steuerungsebenen zu verarbeiten und in notwendige Beinbewegungen umzusetzen. Dabei besteht eine wichtige Aufgabe darin, die Beine so zu koordinieren, dass die Stabilität des Roboters erhalten bleibt. Diese Bedingung muss beim Erzeugen der Beinbewegungen beachtet werden, insbesondere hinsichtlich des Laufmusters. Eine andere Aufgabe besteht darin, die Beinbewegungen so zu koordinieren, dass sich die gewünschte Fortbewegung oder Körperbewegung ergibt. Dabei ist es notwendig, Sensorsignale aus der Umwelt zu verarbeiten und die Beinbewegungen entsprechend anzupassen. Dazu müssen entweder die Laufbewegungen modifiziert oder die Körperbewegung in Abhängigkeit von Sensorsignalen gesteuert werden.

Durch die Navigationsalgorithmen in höheren Steuerungsebenen werden bereits Sensorsignale über die entferntere Umwelt des Roboters verarbeitet. Ab der Hauptsteuerungsebene abwärts sind Informationen über die nähere Umwelt von Interesse, zum Beispiel taktile Informationen. Mit ihrer Hilfe ist es möglich die Bewegungen in gewünschter Weise anzupassen.

#### 5.2.4.1 Generierung von Laufzyklen

In der oben beschriebenen Hauptsteuerungsebene werden die notwendigen Beinbewegungen berechnet. Der Laufzyklusgenerator übernimmt dabei die Aufgabe, das Laufmuster für die Fortbewegung zu erzeugen. Es müssen folgende Randbedingungen eingehalten werden:

- Sicherung der statischen Stabilität durch ein geeignetes Laufmuster und
- Synchronisation der Beine, insbesondere in der Stützphase.

Zur Erfüllung der beiden Bedingungen wird das Laufmuster durch ein zentrales Element, dem Laufzyklusgenerator, erzeugt. Er koordiniert die zeitliche Zuordnung der Beine als Stützbeine. Die Regeln dazu werden durch das Laufmuster (z.B. Dreifußgang) vorgegeben. Gleichzeitig wird durch das Laufmuster gesichert, dass durch die sich in Stützphase befindlichen Beine ein Stützpolygon aufgespannt werden kann, das die statische Stabilität sichert.

#### 5.2.4.1.1 Anforderungen an das Laufmuster hinsichtlich statischer Stabilität

Der Laufroboter wurde mit dem Ziel entwickelt, sicheres Laufen in unbekanntem Gelände zu ermöglichen. Die größte Sicherheit lässt sich mit Hilfe statischer Stabilität während des Laufens erzielen. Dabei muss gewährleistet werden, dass durch die Stützbeine ein Stützpolygon so aufgespannt wird, dass der projizierte Masseschwerpunkt des Roboters vom Polygon eingeschlossen wird. Die Bedingung ist dann erfüllt, wenn sich die orthogonale Parallelprojektion $X'^{(0)}$ des Schwerpunktes $X^{(0)}$ des Roboters entlang des Gravitationsvektors $\vec{g}$ innerhalb des projizierten Stützpolygons (Stabilitätspolygon) $P$ befindet (siehe Abbildung 5.25).

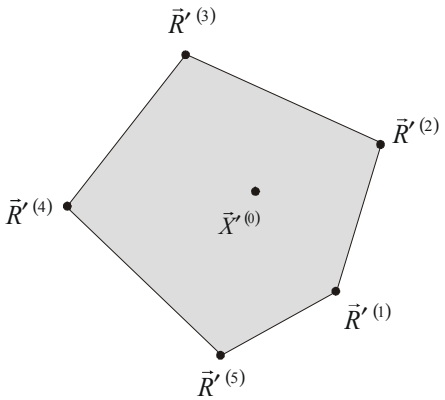



Abbildung 5.25: Projiziertes Stützpolygon mit fünf Stützpunkten und projiziertem Massenpunkt innerhalb des Polygons

Zur Projektion des Schwerpunktes $X^{(0)}$ wird eine Projektionsebene $E_{proj}$ eingeführt, deren Normalenvektor $\vec{n}$ parallel zum Gravitationsvektor $\vec{g}$ ist. Für die Ebene $E_{proj}$ gilt

$$E_{proj}: \quad \vec{n} \cdot \vec{x} = d \text{ mit } \vec{n} \| \vec{g}\,;\ |\vec{n}| = 1\,;\ d \geq 0\ . \tag{5.46}$$

Die das Stützpolygon ergebende Parallelprojektion der Fußpunkte erfolgt mit

$$\vec{R}'^{(i)} = \vec{R}^{(i)} + \left(d - \vec{n} \cdot \vec{R}^{(i)}\right) \cdot \vec{n}\,, \tag{5.47}$$

wobei $\vec{x}'$ die Projektion der Fußpunkte $\vec{x}$ ist.

Entsprechend erfolgt die Projektion des Massenschwerpunktes $X^{(0)}$ in die Ebene

$$\vec{X}'^{(0)} = \vec{X}^{(0)} + \left(d - \vec{n} \cdot \vec{X}^{(0)}\right) \cdot \vec{n}\,. \tag{5.48}$$

Um die Bedingung der statischen Stabilität zu erfüllen, muss das Stabilitätspolygon den projizierten Masseschwerpunkt umschließen. Der Abstand der projizierten Masseschwerpunkte von den Rändern des Stabilitätspolygons kann als Maß für die statische Stabilität benutzt werden und wird als Stabilitätsreserve bezeichnet. Voraussetzung dafür ist, das die Fläche des Polygons größer Null ist. Das heißt, dass die Anzahl der stützenden Füße mindestens drei betragen muss und die projizierten Stützpunkte nicht auf einer Linie liegen.

Tabelle 5.1: Anzahl der frei verfügbaren Beine bei Sicherung statischer Stabilität und unterschiedlicher Gesamtzahl von Beinen

| Gesamtzahl der Beine | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|
| Zur Sicherung der statischen Stabilität notwendige Anzahl von Stützbeinen | 3 | 3 | 3 | 3 | 3 |
| Anzahl der freien Beine bei statischer Stabilität | 1 | 2 | 3 | 4 | 5 |

Wenn sich stets drei der Beine in der Stützphase befinden müssen, so können die anderen Beine genutzt werden, um sie in eine neue Stützposition zu bewegen. Die Anzahl der „freien" Beine ist die Differenz zwischen der Gesamtzahl der vorhandenen Beine einer Laufmaschine und der zur Sicherung der statischen Stabilität benötigten Beine. Tabelle 5.1 zeigt die Anzahl der freien Beine bei einer unterschiedlichen Gesamtanzahl von Beinen. Bei einem Sechsbeiner fällt auf, dass die Anzahl der Stützbeine und die Anzahl frei verfügbarer Beine gleich ist, was die Nutzung einfacher, symmetrischer Laufmuster ermöglicht.

Ziel des Laufens ist es, eine Fortbewegung des Roboters zu erzielen. Dazu müssen Schritte ausgeführt werden, wobei sich Stützphasen und Returnphasen abwechseln. Da sich die Beine beim Stützen abwechseln, unterliegt der Laufzyklus auch zeitlichen Bedingungen. Der Laufzyklus teilt sich in zwei grundlegende Phasen auf. Dies sind

- Stützphase und
- Returnphase.

Entsprechend setzt sich die für einen Laufzyklus benötigte Zeit $T$ (Zykluszeit) aus der Zeit für die Stützphase $T_{Stütz}$ (Stützzeit) und die Returnphase $T_{Return}$ (Returnzeit) zusammen

$$T = T_{Stütz} + T_{\mathrm{Return}} . \tag{5.49}$$

Das Stützverhältnis β, auch duty factor genannt, ist das Verhältnis von Stützzeit zur Zykluszeit

$$\beta = \frac{T_{Stütz}}{T} . \tag{5.50}$$

Damit zu jeder Zeit mindestens drei Füße auf dem Boden stehen, ist ein Stützverhältnis von $\beta \geq 0{,}5$ erforderlich. Das bedeutet ferner, dass die Laufgeschwindigkeit von der Returnzeit begrenzt wird. Wird das Stützverhältnis $\beta < 0{,}5$, so sind nicht immer genügend Füße in Bodenkontakt, um statische Stabilität zu sichern und das Laufen muss somit dynamisch stabilisiert werden. Je kleiner das Stützverhältnis β, um so schneller kann der Roboter laufen. Für statisch stabiles Laufen folgt daraus, dass bei $\beta = 0{,}5$ die schnellste, statisch stabile Fortbewegung erzielt wird. Für einen Vierbeiner folgt bei gleicher Bedingung (statische Stabilität) ein notwendiges Stützverhältnis von $\beta \geq 0{,}75$. Damit kann mit einem Sechsbeiner die schnellste, statisch stabile Fortbewegung erzielt werden.

Die beschriebenen Regeln werden implizit durch das sogenannte Laufmuster beschrieben, indem festgehalten wird, wie und in welcher Reihenfolge die Beine zur Fortbewegung genutzt

werden. Eines der einfachsten Laufmuster, die bei Sechsbeinern vorkommen, ist der Dreifußgang, der bei Insekten beobachtet werden kann [Cruse et al. 95]. Zur Implementierung wurde der Dreifußgang ausgewählt.

#### 5.2.4.1.2 Aufbau des Laufzyklus‘

Unabhängig vom Laufmuster ist es notwendig, Trajektorien für alle Beine zu erzeugen. Diese unterliegen bestimmten Randbedingungen, die durch das Laufmuster und der beabsichtigten Form der Trajektorie vorgegeben werden. Durch die Bedingung zur Sicherung der statischen Stabilität ($\beta \geq 0{,}5$; s. Abschnitt 5.2.4.1.1) und der Gleichungen (5.49) und (5.50) ergibt sich, dass die Stützzeit größer oder gleich der Returnzeit sein muss

$$T_s \geq T_r \ . \tag{5.51}$$

Zur Konstruktion des Laufzyklus wird die Returnphase in drei Phasen

- Ablösephase,
- Transferphase und
- Absenkphase

aufgesplittet, so dass nun der Laufzyklus aus vier Phasen besteht. Dies hat den Vorteil, dass Absenk- und Ablösephase für die Adaption des Laufzyklus‘ benutzt werden können. In Abbildung 5.26 ist die Geometrie eines Laufzyklus dargestellt.

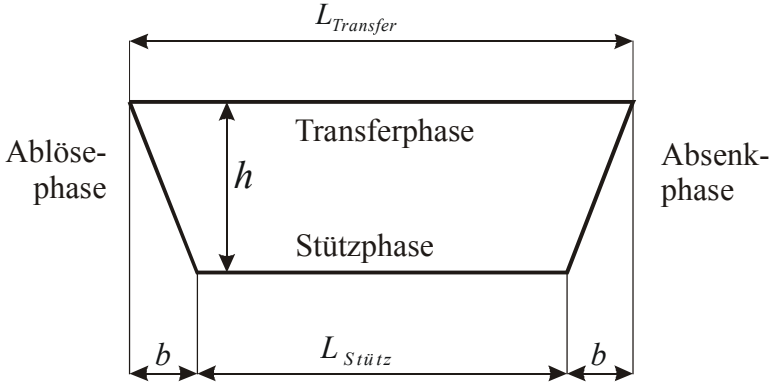



Abbildung 5.26: Geometrischer Aufbau eines Laufzyklus‘

Es ist zu erkennen, dass die zurückgelegte Strecke in der Stützphase kleiner ist als die anderen drei Teilstrecken zusammen. Aus der Bedingung (5.51) folgt, dass die Zeit für Stützphase mindestens so groß sein muss wie die Zeit für die anderen drei Phasen. Daraus folgt, dass die durchschnittliche Bewegungsgeschwindigkeit in der Returnphase größer als in der Stützphase sein muss. Die Maximalgeschwindigkeit des Roboters wird also von der Geschwindigkeit in der Returnphase bestimmt.

Zunächst wird von den geometrischen Größen des Schreitzyklus ausgegangen. Aus Abbildung 5.26 ist zu ersehen, dass für den in der Transferphase zurückgelegten Weg $L_{Transfer}$

$$L_{Transfer} = 2 \cdot b + L_{Stütz} \tag{5.52}$$

gilt. Der Parameter $L_{Stütz}$ in (5.52) kennzeichnet die Schrittlänge. Der Parameter $b$ ist für die Ablöse- und Absenkphase gleich und ist die Projektion des in der jeweiligen Phase zurückgelegten Weges auf die Ebene. Es ist offensichtlich, dass der auf der Trajektorie in Ablöse-, Transfer- und Absenkphase zurückgelegte Weg länger ist, als der in der Stützphase zurückgelegte. Nach Gleichung (5.51) steht für die ersten drei Phasen maximal genau so viel Zeit zur Verfügung wie für die Stützphase. Für einen symmetrischen Laufzyklus gilt weiterhin

$$T_{Ablöse} = T_{Absenk} = T_b \,. \tag{5.53}$$

Für den Fall des Dreifußganges gilt

$$T_{Stütz} \geq T_{Ablöse} + T_{Transfer} + T_{Absenk} = T_{Transfer} + 2T_b \,. \tag{5.54}$$

Alle drei geometrischen Größen sind von den kinematischen Parametern der Beine abhängig, da sich die Trajektorie des Laufzyklus‘ im Arbeitsraum des Beines befinden muss. Der Konstruktion des Laufzyklus liegt die Idee zugrunde, dass durch Überlagerung von verschiedenen Teilbewegungen die eigentliche, resultierende Laufbewegung entsteht. Unter der vereinfachenden Voraussetzung, dass sich der Roboterkörper linear bewegt, entspricht $v_{Stütz}$ genau der Bewegungsgeschwindigkeit des Roboterkörpers. Es gilt

$${}^{K}\vec{v}_{Stütz} \uparrow\downarrow {}^{0}\vec{v}_{Körper}, \left|{}^{K}\vec{v}_{Stütz}\right| = \left|{}^{0}\vec{v}_{Körper}\right| . \tag{5.55}$$

Dabei ist ${}^{K}\vec{v}_{Stütz}$ die Relativgeschwindigkeit des Fußes gegenüber dem Körper und ${}^{0}\vec{v}_{Körper}$ die Bewegungsgeschwindigkeit des Körpers im Weltkoordinatensystem.

Für Ablöse- und Absenkphasen wird gefordert, dass sich der Fuß zum Untergrund hin vertikal absenkt bzw. mit einer vertikalen Bewegung ablöst. Dadurch werden horizontale Kraftwirkungen gegenüber dem Untergrund vermieden. Der gewünschte Effekt kann durch Überlagerung von zwei Teilbewegungen erzielt werden. Dazu wird die horizontale Bewegungskomponente der Ablöse- und Absenkphase $v_{h,b}$ gleich der Bewegung in der Stützphase gesetzt. Es gilt folglich

$$v_{h,b} = v_{Stütz} \,. \tag{5.56}$$

Die vertikale Bewegungskomponente realisiert das eigentliche Ablösen bzw. Absenken des Fußes. Ablöse-, Transfer- und Absenkphase sollten so kurz wie möglich gestaltet werden, da in ihnen einerseits die längste Strecke der Trajektorie zurückgelegt wird, andererseits die benötigte Zeit Einfluss auf die erzielbare Laufgeschwindigkeit hat.

#### 5.2.4.1.3 Erzielbare Laufgeschwindigkeit

Geht man in Gleichung (5.54) davon aus, dass die Gleichheitsbedingung gilt und dass es sich bei den Teilbewegungen der Fußpunkte der Beine um geradlinige, gleichförmige Bewegungen handelt, so gilt

$$\frac{L_{Stütz}}{v_{Stütz}} = 2\frac{b}{v_{h,b}} + \frac{L_{Transfer}}{v_{Transfer}}. \tag{5.57}$$

Unter Verwendung von Gleichung (5.56) erhält man

$$v_{Stütz} = \frac{L_{Stütz} - 2b}{L_{Transfer}} \cdot v_{Transfer}. \tag{5.58}$$

Ersetzt man $L_{Transfer}$ durch Gleichung (5.52), so ergibt sich ein Verhältnis von Stütz- und Transfergeschwindigkeit, das nur von geometrischen Größen des Laufzyklus‘ abhängt

$$\frac{v_{Stütz}}{v_{Transfer}} = \frac{L_{Stütz} - 2b}{L_{Stütz} + 2b}. \tag{5.59}$$

Ablöse- und Absenkphase werden durch Überlagerung zweier Geschwindigkeiten erzeugt. Da nach Bedingung (5.54) auch Ablöse- und Absenkphase möglichst kurz ausfallen sollten, kann die maximale vertikale Geschwindigkeit genutzt werden. Diese kann näherungsweise gleich der maximalen Transfergeschwindigkeit angenommen werden, da es sich jeweils um die maximale Geschwindigkeit für die freie Bewegung eines Beines handelt. Das gilt nur für den Fall, dass zum Zweck der Adaption an den Untergrund keine geringeren Geschwindigkeiten vorgesehen sind. Für Ablöse- und Absenkphase gilt dann

$$t_b = \frac{b}{v_{Stütz}} = \frac{h}{v_{Transfer}}. \tag{5.60}$$

Gleichung (5.59) lässt sich so umformen, dass nur noch Schrittlänge und –höhe, sowie die Geschwindigkeiten für die freie Bewegung der Beine und der Stützbewegung vorkommen

$$v_{Stütz} \cdot \left(v_{Transfer} \cdot L_{Stütz} + 2 \cdot v_{Stütz} \cdot h\right) = v_{Transfer} \cdot \left(v_{Transfer} \cdot L_{Stütz} - 2 \cdot v_{Stütz} \cdot h\right). \tag{5.61}$$

Dabei ist die Stützgeschwindigkeit gleich der Fortbewegungsgeschwindigkeit des Roboters. Zur Bestimmung der Stützgeschwindigkeit ergibt sich damit eine quadratische Gleichung

$$v_{Stütz}^2 + v_{Stütz} \cdot \left(\frac{v_{Transfer} \cdot L_{Stütz}}{2h} + v_{Transfer}\right) - \frac{v_{Transfer}^2 \cdot L_{Stütz}}{2h} = 0. \tag{5.62}$$

Durch Lösen dieser quadratischen Gleichung lässt sich die erzielbare Fortbewegungsgeschwindigkeit bei gegebener Schrittlänge, Schritthöhe und maximaler Bewegungsgeschwindigkeit des Beines berechnen

$$v_{Stütz} = v_{Transfer} \cdot \left[-\frac{L_{Stütz} + 2h}{4h} \pm \frac{1}{4h}\sqrt{L_{Stütz}^2 + 12L_{Stütz} \cdot h + 4h^2}\right]. \tag{5.63}$$

In Gleichung (5.63) ist nur das positive Ergebnis als sinnvolle Lösung interessant. Das Verhältnis von maximaler Fußgeschwindigkeit $v_{Transfer}$ und Stützgeschwindigkeit $v_{Stütz}$ bei gegebenen Schreitzyklusparametern $L_{Stütz}$ und $h$ ergibt sich somit zu

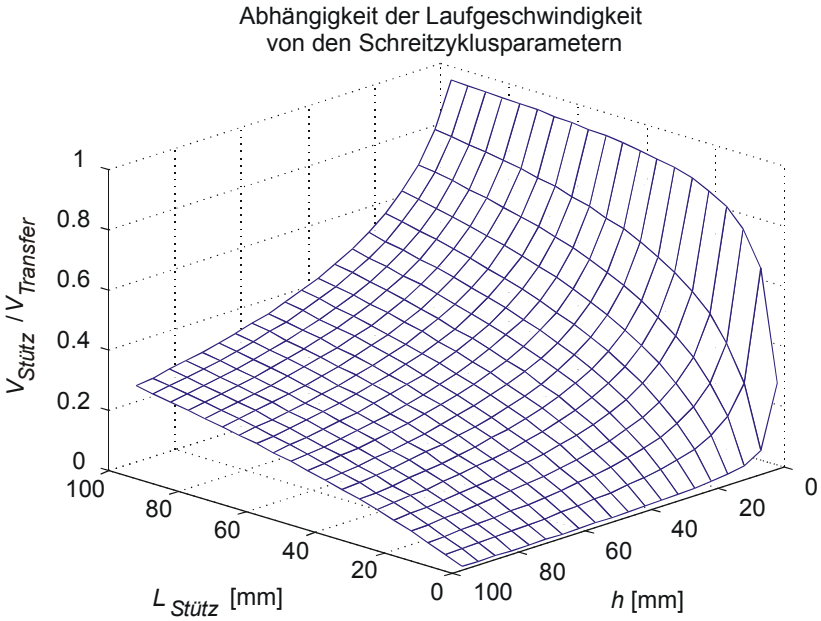



Abbildung 5.27 Abhängigkeit der Laufgeschwindigkeit von den Schreitzyklusparametern

$$\frac{v_{Stütz}}{v_{Transfer}} = -\frac{L_{Stütz} + 2h}{4h} + \frac{1}{4h}\sqrt{L_{Stütz}^2 + 12L_{Stütz} \cdot h + 4h^2} \quad . \tag{5.64}$$

Abbildung 5.27 zeigt das Verhältnis (5.64) für die zwei variablen Parameter $L_{Stütz}$ und $h$. Es ist zu erkennen, dass einerseits die Laufgeschwindigkeit größer wird, wenn die Schrittlänge größer wird. Andrerseits wirkt sich die Schritthöhe negativ auf die Laufgeschwindigkeit aus. Die Schreitzyklen sollten deshalb so geformt werden, dass sich eine möglichst große Schrittlänge bei einer möglichst niedrigen, noch sinnvollen Schritthöhe ergibt. Durch die Optimierung dieser beiden Parameter wird die benötigte Zeit für die Returnphase in Relation zur Schrittlänge minimiert. Zum Erzielen der bestmöglichen Laufgeschwindigkeit lässt sich folgende Strategie ableiten:

- die Schrittlänge sollte möglichst groß gewählt werden,
- die Schritthöhe sollte möglichst klein sein und
- zur Minimierung der notwendigen Schritthöhe sollten die Laufzyklen bestmöglich an die zu erwartende Untergrundhöhe angepasst werden.

Somit steht eine einfache Methode bereit, um Muster für Laufzyklen in Abhängigkeit von gegebenen Parametern wie Schrittweite und -höhe zu generieren. Dabei wird die Methode der Superposition von zwei Teilgeschwindigkeiten genutzt. Ferner stehen Informationen zur Auslegung der Parameter bereit.

#### 5.2.4.2 Abbildung von Laufzyklen und Adaption

Als Ausgangspunkt steht in jedem Fall ein Musterlaufzyklus bereit, der in Abhängigkeit von den gegebenen Parametern konfiguriert werden kann. Dabei fließen Randbedingungen zur Sicherung der statischen Stabilität mit ein. Nachfolgend soll näher auf das Abbildungsprinzip von Laufzyklen und deren Adaption eingegangen werden.

#### 5.2.4.2.1 Prinzip der Abbildung von Laufzyklen

Der Laufzyklusgenerator erzeugt einen Musterlaufzyklus in einer zweidimensionalen Koordinatenebene, von dem alle individuellen Beinbewegungen abgeleitet werden können. Alle Beine, die gleichartige Bewegungen ausführen sollen, werden zu einer Beingruppe zusammengefasst, so dass ihre Bewegungen von der gleichen Musterbewegung abgeleitet werden können. Auf diese Weise wird für jede Beingruppe jeweils nur ein Laufzyklus erzeugt. Die Anzahl der Beingruppen ist vom gewählten Laufmuster abhängig und kann zwischen zwei und sechs betragen. Zur Sicherung der statischen Stabilität müssen die Beine den Beingruppen so zugeordnet werden, dass sie in der Stützphase ein ausreichend großes Stützpolygon aufspannen. Weiterhin muss durch eine Phasenverschiebung der Musterlaufzyklen gesichert werden, dass sich stets ausreichend Beine in der Stützphase befinden. Für den Dreifußgang bedeutet das, dass zwei Beingruppen eingeführt werden, deren Musterzyklen so phasenverschoben sind, so dass jeweils eine Beingruppe die Stützbeine bereitstellt. Die Zuordnung der Beine $B_i$ *(i=1, 2,..., 6)* zu den Beingruppen $G_k$ *(k=1, 2)* erfolgt mit

$$\begin{matrix} B_1, B_3, B_5 \in G_1 \\ B_2, B_4, B_6 \in G_2 \end{matrix} . \tag{5.65}$$

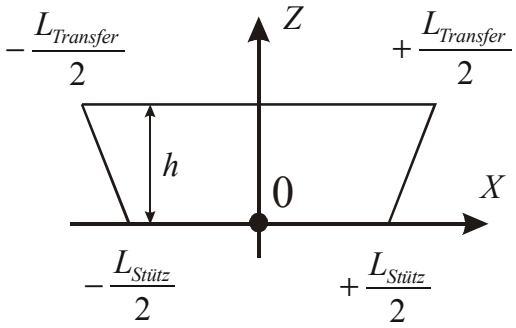



Abbildung 5.28: Musterschreitzyklus mit Koordinatensystem

Somit stehen die Musterbewegungen für jedes Bein als Relativbewegungen zur Verfügung, die mit Hilfe einer geeigneten Transformation in den Arbeitsbereich der Beine projiziert werden müssen. Da es sich jeweils um Relativbewegungen handelt, müssen diese noch mit einer Bezugsposition verknüpft werden. Als Bezugsposition der Relativbewegung wird die mittlere Position der Stützphase genutzt, wie Abbildung 5.28 zeigt, so dass die Relativbewegungen symmetrisch bezüglich der Z-Achse sind. Die Bezugsposition des Beines und die Relativbewegung werden mit Hilfe folgender Beziehung überlagert

$$\vec{R}^{(i)} = \vec{R}_0^{(i)} + T^{(i)}\left\{\vec{S}^{(G_k)}\right\} \tag{5.66}$$

mit
- $G_k$ - Beingruppe mit k=[1, 2] für den Dreifußgang
- $\vec{S}^{(G_k)}$ - Position auf der Trajektorie des Musterlaufzyklus' der Beingruppe $G_k$
- $T^{(i)}$ - Transformation für das *i*-te Bein
- $\vec{R}_0^{(i)}$ - Bezugsposition für das *i*-te Bein
- $\vec{R}^{(i)}$ - Resultierende Position für das *i*-te Bein.

Mit der Transformation $T^{(i)}$ in Gleichung (5.66) wird bestimmt, wie die Abbildung des Laufzyklus im Arbeitsraum der Beine erfolgt. Der Laufzyklus wird für jedes Bein so gedreht und verschoben, dass die Laufzyklen den gewünschten Vortrieb erzeugen.

Für ein einfaches, gerichtetes Laufen in eine Richtung, die durch den Laufrichtungswinkel ω vorgegeben wird, wird folgende Transformation verwendet

$$\vec{R}^{(i)} = \vec{R}_0^{(i)} + \begin{pmatrix} \cos(\omega) & -\sin(\omega) & 0 \\ \sin(\omega) & \cos(\omega) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} S_x^{(G_k)} \\ 0 \\ S_z^{(G_k)} \end{pmatrix}. \tag{5.67}$$

Nunmehr stehen alle notwendigen Informationen zur Ausführung einer einfachen Laufbewegung zur Verfügung, deren Repräsentation im Zentralkoordinatensystem erfolgt. In Abbildung 5.29 wird die Trajektorie eines Laufzyklus gezeigt, wie sie nach Gleichung (5.67) erzeugt wird. Die Darstellung erfolgt mit Bezug auf das Zentralkoordinatensystem.

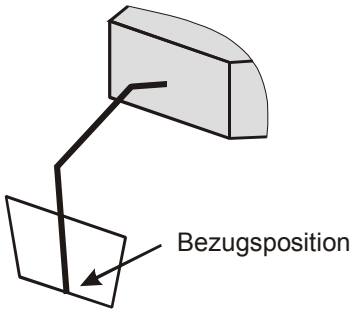
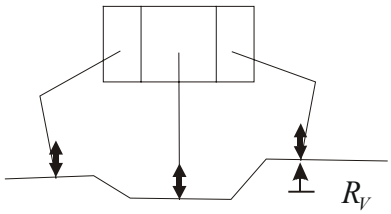



Abbildung 5.29: Fußtrajektorie und Bezugsposition

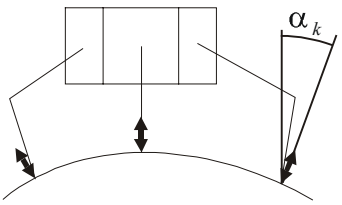
#### 5.2.4.2.2 Adaption durch Abbildungsvorschriften

Die Methode der Abbildung der Laufzyklen auf die Beinenden kann nicht nur für einfache Laufoperationen genutzt werden. Durch Veränderung der Transformation lassen sich die Laufzyklen kippen oder verschieben, so dass eine gezielte Anpassung an die Geländeform bzw. die Umgebung vorgenommen werden kann. Abbildung 5.30 zeigt Beispiele für mögliche Anpassungen. Durch eine lineare Verschiebung der Laufzyklen wird es beispielsweise möglich, die Beinstellungen beim Laufen so zu manipulieren, dass sich der Roboter in engen Umgebungen bewegen kann. Durch horizontale Verschiebung oder durch vertikales Verschieben der Laufzyklen richte sich der Roboter entweder stark auf oder er hält den Körper dicht über den Boden.

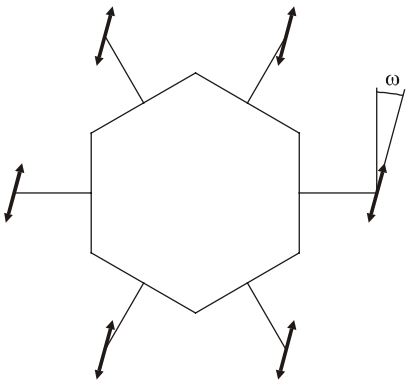
Die Anpassungen der Laufzyklen erfolgt durch Veränderung der Transformationen und damit werden weitere Parameter eingeführt. Das bereits genannte Verschieben der Laufzyklen erfolgt lediglich durch eine Addition eines Verschiebungsvektors $\vec{R}_V^{(i)}$ zur Bezugsposition des Beines. Dabei können für jedes Bein individuelle Verschiebungen vorgenommen werden, wie Abbildung 5.30a zeigt. Das Kippen der Laufzyklen erfolgt hingegen durch Rotation des Musterlaufzyklus, wie in Abbildung 5.30b für ein Beispiel dargestellt ist. Es ist zu sehen, dass



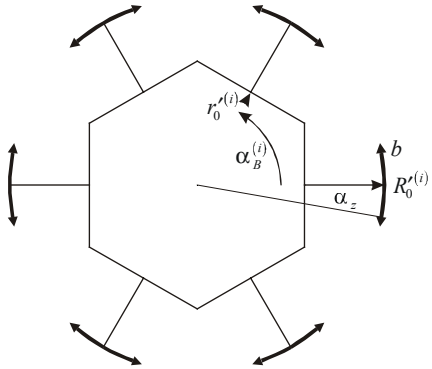

a) Verschiebung von Laufzyklen zur Adaption an die Geländeform




b) Kippen von Laufzyklen zur Adaption an die lokale Bodenneigung




c) Gerichtetes laufen




d) Zirkulare Schritte zum Drehen des Roboters

Abbildung 5.30: Beispiele für Laufzyklusabbildungen

es sinnvoll sein kann, die Laufzyklen unterschiedlich zu kippen. In einem solchen Fall kann der Kippwinkel in Abhängigkeit von der Beinposition unterschiedlich groß gewählt werden. Im Beispiel wird der Kippwinkel in Abhängigkeit vom seitlichen Abstand so gewählt, dass er in einem groben Zusammenhang mit der lokalen Bodenneigung steht. Die Schreitzyklen werden nicht nur entsprechend der gewünschten Laufrichtung gedreht, sondern auch für jedes Bein individuell gekippt. Die Abbildung der Laufzyklen würde sich entsprechend dem Beispiel wie folgt gestalten

$$\vec{R}^{(i)} = \vec{R}_0^{(i)} + \vec{R}_V^{(i)} + \begin{pmatrix} \cos(\omega) & -\cos(\alpha_k^{(i)})\sin(\omega) & \sin(\alpha_k^{(i)})\sin(\omega) \\ \sin(\omega) & \cos(\alpha_k^{(i)})\cos(\omega) & -\sin(\alpha_k^{(i)})\cos(\omega) \\ 0 & \sin(\alpha_k^{(i)}) & \cos(\alpha_k^{(i)}) \end{pmatrix} \cdot \begin{pmatrix} S_x^{(G_k)} \\ 0 \\ S_z^{(G_k)} \end{pmatrix} \qquad (5.68)$$

mit $\alpha_k^{(i)}$ - Kippwinkel für den Laufzyklus des *i*-ten Beines
$\omega$ - Laufrichtung
$\vec{R}_V^{(i)}$ - Laufzyklusverschiebung für das *i*-te Bein.

Nützlich ist ein solches Verfahren nicht nur bei einer konvexen, sondern auch bei einer konkaven Oberfläche, wie sie in Rohren anzutreffen ist.

Es ist nicht immer ausreichend, dass der Roboter ohne Wendemanöver in eine beliebige Richtung laufen kann (siehe Abbildung 5.30c). Sind auf dem Roboter Werkzeuge montiert, die nicht beliebig frei ausgerichtet werden können, so muss deren Positionierung auch durch entsprechende Ausrichtung des Roboterkörpers unterstützt werden. Eine Vorpositionierung wird dabei am Zielort durch Drehung des gesamten Roboters um seine eigene Achse erreicht, wie in Abbildung 5.30d dargestellt. Diese Aufgabe lässt sich ebenfalls durch eine passende Transformation lösen. Dazu erfolgt die Abbildung der Laufzyklen auf einem Zylindermantel.

### 5.2.4.3 Körperbewegung

Es wurde bereits gezeigt, dass das Interesse an den Möglichkeiten von Laufmaschinen nicht allein auf den Transport beschränkt ist. Eine große Zahl von Einsatzmöglichkeiten ergibt sich durch die Handhabung von Objekten, Werkzeugen oder Instrumenten. Um diese zu positionieren, bietet sich die Nutzung der Freiheitsgrade des Roboterkörpers an.

Zur Positionierung des Körpers müssen alle Vektoren bestimmt werden, die dessen Lage und Orientierung bei gegebenen Fußpunkten bestimmen. Das Problem ist, dass es bei einer Laufmaschine keine ortsfesten Fußpunkte gibt, auf die die Bewegungen bezogen werden können, wie es bei einer Stewart-Plattform [Graf et al. 98] der Fall ist. Vielmehr sind die Fußpunkte ebenfalls einer Ortsveränderung unterworfen, da sie zum Laufen genutzt werden. Es muss deshalb ein anderer Bezugspunkt gefunden werden, um die gewünschten Bewegungen zu beschreiben. Da sich der Roboter selbst fortbewegt, ist auch kein fester Bezug auf das Weltkoordinatensystem oder einen Punkt aus der Umwelt sinnvoll. Um diesen Widerspruch aufzulösen, wird das Zentralkoordinatensystem als Referenzkoordinatensystem genutzt, das bereits zur Beschreibung der Laufbewegungen genutzt wurde. Es hat den Vorteil, dass es lokal für den Roboter gültig ist und durch seine Position der Roboter innerhalb des Weltkoordinatensystems lokalisiert wird.

Im Unterschied zur Stewart-Plattform werden hier nicht lineare Gelenke, sondern die Beine des Roboters zur Positionierung genutzt. Sowohl zur Beschreibung der Körperposition als auch zur späteren Ableitung der Beinposition müssen die Vektoren der Beinbefestigungspunkte ermittelt werden. Zur Ausführung technischer Operationen, wie Positionierung von Objekten, sind geeignete Relativbewegungen notwendig, die durch Sensorinformationen beeinflusst werden.

Zur Unterstützung der technischen Operationen wird die Körperposition in mehreren Teilschritten ermittelt:

1. Linearverschiebung in Relation zu den Körperachsen,
2. Rotation um Zentralkoordinatenachsen und
3. Linearverschiebung in Relation zu Zentralkoordinatenachsen.

Ausgangspunkt sind die Beinbefestigungspunkte ${}^{K}r_{0,0}^{(i)}$, die über feste geometrische Beziehungen miteinander verbunden sind. Da die Zentral- und Körperkoordinatensysteme zu Beginn der drei Teilschritte als deckungsgleich angenommen werden können, sind demzufolge auch die Koordinaten der Beinbefestigungspunkte ${}^{K}r_{0,0}^{(i)}$ bekannt.

Im ersten Schritt werden die Koordinaten der Beinbefestigungspunkte entlang der Achsen des Zentralkoordinatensystems verschoben, das noch mit dem Körperkoordinatensystem zusammenfällt

$$ {}^{Z}\vec{r}_{0,1}^{(i)} = \mathrm{Trans}\left(Z, \Delta X_{1,K}\right) \cdot {}^{Z}\vec{r}_{0,0}^{(i)} = {}^{Z}\vec{r}_{0,0}^{(i)} + \Delta\vec{X}_{1,K} \quad . \tag{5.69} $$

Hiermit wird eine Verschiebung erzeugt, die auf den Roboterkörper, an dem Werkzeuge oder Instrumente montiert sein können, bezogen ist. Diese lineare Verschiebung bleibt weiterhin als Verschiebung entlang der Körperachsen erhalten, selbst bei geneigtem Körper. Sie kann

für Operationen genutzt werden, bei denen Bewegungen in Bezug auf die Körperachsen gefordert sind.

Im zweiten Schritt erfolgt eine Rotation um die Koordinatenachsen des Zentralkoordinatensystems

$$ {}^{Z}\vec{r}_{0,2}^{(i)} = \mathrm{Rot}\left(Z_z, \alpha_{K,z}\right) \cdot \mathrm{Rot}\left(Z_y, \alpha_{K,y}\right) \cdot \mathrm{Rot}\left(Z_x, \alpha_{K,x}\right) \cdot {}^{Z}\vec{r}_{0,1}^{(i)} \quad . \tag{5.70} $$

Sie sorgt für die gewünschte Neigung und Drehung des Roboterkörpers. Dabei kann im Gegensatz zu [Schneider et al. 96] um alle drei Koordinatenachsen gleichzeitig gedreht werden. Die Rotationsbewegungen stehen ohne Vorzugsrichtung zur Verfügung und unterstützen somit das Konzept der Richtungsunabhängigkeit.

Abschließend wird nochmals eine lineare Verschiebung entlang der Achsen des Zentralkoordinatensystems vorgenommen

$$ {}^{Z}\vec{r}_{0}^{(i)} = \mathrm{Trans}\left(Z, \Delta X_{2,K}\right) \cdot {}^{Z}\vec{r}_{0,2}^{(i)} = {}^{Z}\vec{r}_{0,2}^{(i)} + \Delta\vec{X}_{2,K} \quad . \tag{5.71} $$

Mit der zweiten Translation in Gleichung (5.71) erzeugt man eine Verschiebung in Bezug auf das Zentralkoordinatensystem, das eine bekannte Position und Orientierung gegenüber der Umwelt hat und genutzt werden kann, wenn ein Bezug zu äußeren Objekten besteht. Nach Ausführung der drei beschriebenen Bewegungen sind die gesuchten Koordinaten der Beinbefestigungspunkte in Bezug auf das Zentralkoordinatensystem bekannt. Abbildung 5.31 verdeutlicht die Teilschritte zur Erzeugung der Körperbewegungen.

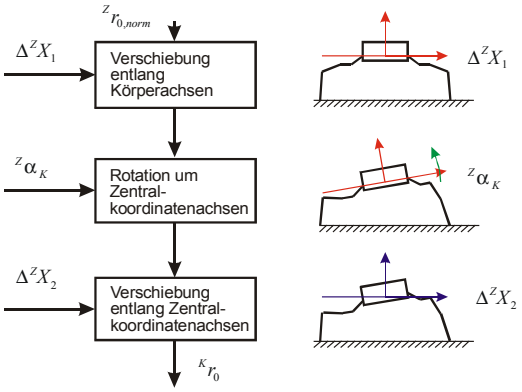



Abbildung 5.31: Erzeugen der Körperbewegung mit Hilfe mehrerer Teiloperationen

Mit Hilfe der Körperbewegung ist es möglich,

- den Körper gezielt auszurichten und zu orientieren,
- eine Feinpositionierung vorzunehmen, ohne zu laufen,
- Werkzeuge und Instrumente zu positionieren und zu führen und
- durch die Freiheitsgrade der Körperbewegung eine Einsparung von Freiheitsgraden bei Werkzeugen und Instrumenten zu erzielen.

Mit dem dreistufigen Algorithmus zur Körperbewegung ist es möglich,

- den Körper linear zu verschieben und zu drehen,
- lineare Verschiebungen mit Bezug auf den Roboterkörper oder den Untergrund vorzunehmen, so dass verschiedene Relativbewegungen entstehen und
- die Relativbewegungen gezielt zur Ausführung von Serviceoperationen einzusetzen.

#### 5.2.4.4 Ableitung der Beinbewegungen

Es wurde bereits gezeigt, wie Laufbewegungen und Körperbewegungen erzeugt werden können. Diese Teilbewegungen werden unabhängig voneinander erzeugt. Ihre Unabhängigkeit spiegelt sich im Steuerungssystem dadurch wider, dass im Strukturbild diesen beiden Teilfunktionen jeweils ein separater Block zugeordnet ist. Zur Realisierung der gesamten Bewegung müssen jedoch die Teilbewegungen zusammengeführt und in entsprechende Beinbewegungen umgesetzt werden [Ihme 00b].

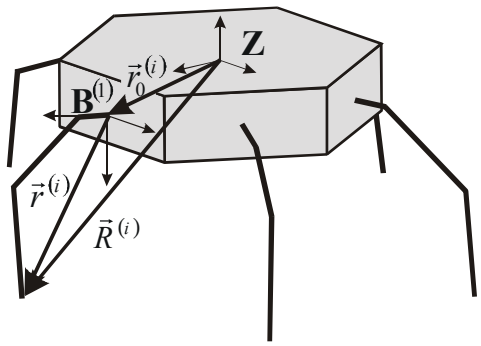



Abbildung 5.32: Geometrische Darstellung der Vektorgleichung zur Bestimmung der Beinposition

Das Zusammenführen der Teilbewegungen vereinfacht sich dadurch, dass beide bereits im Zentralkoordinatensystem beschrieben werden. Die Laufbewegung wird durch die Positionsfolge des Fußpunktes beschrieben und die Körperbewegung durch die Positionsfolge der Beinbefestigungspunkte. Damit existieren bereits zwei Parameter, aus denen sich die Beinposition gewinnen lässt. In Abbildung 5.32 wird dieser Zusammenhang graphisch dargestellt. Es ist zu sehen, dass der Vektor der Beinposition in einer einfachen Vektorgleichung enthalten ist

$$ {}^{Z}\vec{R}^{(i)} = {}^{Z}\vec{r}_0^{(i)} + {}^{Z}\vec{r}^{(i)} . \tag{5.72}$$

Somit gilt für die Beinposition

$$ {}^{Z}\vec{r}^{(i)} = {}^{Z}\vec{R}^{(i)} - {}^{Z}\vec{r}_0^{(i)} . \tag{5.73}$$

Die Vektoren ${}^{Z}\vec{r}^{(i)}$ sind gebundene Vektoren im Zentralkoordinatensystem, deren Anfangspunkt mit den Beinbefestigungspunkten zusammenfallen. Es liegen prinzipiell die notwendigen Informationen für die Sollwerte der Beinpositionen vor. Es handelt sich allerdings um gebundene Vektoren, die in die lokalen Beinkoordinatensysteme transformiert werden müs-

sen, um sie an die Beincontroller übergeben zu können. Die Anfangspunkte der Vektoren fallen jeweils mit dem Koordinatenursprung des entsprechenden Beinkoordinatensystems zusammen. Abschließend muss eine Rotation der Koordinaten vorgenommen werden, so dass die Beinkoordinaten im Beinkoordinatensystem dargestellt werden

$$ {}^{B}\vec{r}^{(i)} = {}^{B}_{Z}\mathrm{T} \cdot {}^{Z}\vec{r}^{(i)} = \mathrm{Rot}\left(\alpha_B^{(i)} - \alpha_{K,z}\right) \cdot \mathrm{Rot}\left(-\alpha_{K,y}\right) \cdot \mathrm{Rot}\left(-\alpha_{K,x}\right) \cdot {}^{Z}\vec{r}^{(i)} . \tag{5.74} $$

Es liegen somit alle Sollkoordinaten in den lokalen Beinkoordinatensystemen vor. Diese können zur weiteren Verarbeitung an die Beincontroller übergeben werden.

### 5.2.4.5 Richtungssteuerung, Umschaltpunkte

Bisher wurde zur Generierung von Laufbewegungen ein Algorithmus genutzt, der mit Hilfe einer Musterbewegung alle weiteren zum Laufen notwendigen Beinbewegungen erzeugt. Es ist damit prinzipiell möglich, in jede Richtung zu laufen. Eine Modifizierung der Laufparameter zu jedem beliebigen Zeitpunkt ist jedoch nicht sinnvoll, da dadurch nicht vorhersagbare oder nicht realisierbare Bewegungen entstehen können. Sind keine negativen Auswirkungen auf die aktuelle Bewegung zu erwarten, so kann eine Modifizierung dieser Parameter vorgenommen werden. Dies ist der Fall, wenn

- eine bestimmte Laufrichtung vorgegeben bzw. die aktuelle Laufrichtung geändert werden soll,
- neue Parameter, wie Schrittweite und Schritthöhe vorgegeben werden sollen und
- ein Laufvorgang initialisiert bzw. eine Ruheposition eingenommen werden soll.

Damit wird eine Erweiterung des Funktionsumfanges vorgenommen. Neben dem kontinuierlichen Laufen ist auch das sichere Stehen möglich, um beispielsweise technische Operationen durchzuführen. Damit wird gleichzeitig die Möglichkeit geschaffen, den Roboter in bestimmten Funktionsmodi zu betreiben. Solche Funktionsmodi können beispielsweise sein:

- Laufen mit einem bestimmten regelmäßigen Laufmuster,
- Laufen in einem unregelmäßigen Laufmuster (free gait),
- Stehen in einer Ruheposition und
- Ausführung einer technischen Operation, z.B. Montage- oder Bohroperation.

Damit entsteht die Frage, wie die Übergänge zwischen diesen Funktionsmodi realisiert werden können. Eine Lösung besteht darin, einen sicheren Übergang zwischen den Funktionsmodi mit Hilfe einer Zustandsmaschine vorzunehmen. Dabei werden die Schaltbedingungen (Zustandsübergang) an bestimmte zu erfüllende Voraussetzungen geknüpft. Sind die erforderlichen Bedingungen erfüllt, so ist ein definierter Zustand entstanden (Umschaltpunkt) und ein neuer Bewegungsvorgang kann eingeleitet werden. Am Beispiel des Überganges zwischen Stehen in Ruheposition und Laufvorgang beim Dreifußgang soll dies näher erläutert werden. Die Aufgabe besteht darin, den Übergang von einer Ruheposition, bei der alle Beine auf dem Boden stehen, zu einem definierten Startpunkt auf der Lauftrajektorie für jede Beingruppe zu vollziehen. Zunächst müssen drei Füße in die Startposition auf der Trajektorie entsprechend dem Laufmuster bewegt werden. Die drei verbleibenden, noch am Boden stehenden Füße, befinden sich bereits auf der Trajektorie, da diese durch den Bezugspunkt verläuft (siehe z-Achse in Abbildung 5.28). Damit ist der Übergangsprozess beendet. Die nun

eingenommene Position ist der Symmetriepunkt bezüglich der Richtung der Lauftrajektorien, die durch Transformationen räumlich orientiert werden (siehe Abschnitt 5.2.4.2). Da diese Punkte regelmäßig nach jedem Halbschritt durchlaufen werden, kann hier eine Änderung der Laufrichtung vorgenommen werden. Dieser Symmetriepunkt wird für die Richtungssteuerung und den Übergang zwischen Laufen und Stehen als Umschaltpunkt genutzt. Das beschriebene Verfahren wurde mit Hilfe einer Zustandsmaschine implementiert. Der entsprechende Zustandsgraph ist in Abbildung 5.33 dargestellt. Die vertikal in der Mitte angeordneten Zustände repräsentieren Übergangs- und Umschaltzustände.

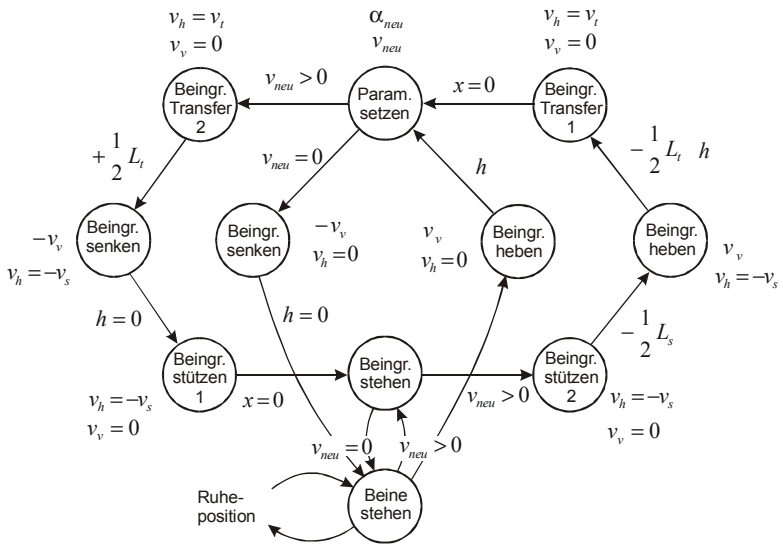



Abbildung 5.33: Zustandsgraph zur Realisierung eines Dreifußganges

### 5.2.5 Beinsteuerungsebene

In den vorangegangenen Abschnitten wurde ausgehend vom gewünschten Gesamtverhalten des Roboters gezeigt, wie geeignete Lauf- und Körperbewegungen erzeugt werden können. Daraus wurden die dafür notwendigen Bewegungen der Beine abgeleitet, die nun Ausgangspunkt für die Beinsteuerungsebene sind. Die Sollwerte für die Beinpositionen liegen in lokalen Beinkoordinatensystemen vor. Da alle Beine identisch aufgebaut sind, können für alle Beine die gleichen Algorithmen genutzt werden.

Die Anwendbarkeit der kinematischen Transformation ist nur für einen Laufroboter mit der gleichen Beinkinematik möglich. Da in diesem Abschnitt nur ein einzelnes Bein betrachtet wird, wird der Bezug durch einen hochgestellten Index weggelassen.

Die Schnittstelle zum Roboter ist der Befestigungspunkt des Beines, auf den sich auch das Beinkoordinatensystem bezieht. Durch die Unabhängigkeit der einzelnen Beine ergibt sich, dass alle Operationen und Transformationen nebenläufig oder in getrennten Recheneinheiten ausgeführt werden können.

Die Sollwerte liegen in kartesischen Koordinaten vor, die das Beinende einnehmen soll. Es besteht nun die Aufgabe, aus den kartesischen Koordinaten die Achs- oder Gelenkkoordinaten

zu gewinnen. Dieser Übergang wird durch die inversen kinematischen Koordinatentransformationen (weiterhin „inverse Kinematik“ genannt) beschrieben

$$\vec{q} = f^{-1}(\vec{x}) \tag{5.75}$$

mit $\vec{x} = (x_1 \quad x_2 \quad \cdots \quad x_n)^{\mathrm{T}}$ Positionsvektor im kartesischen Raum

$\vec{q} = (q_1 \quad q_2 \quad \cdots \quad q_m)^{\mathrm{T}}$ Positionsvektor in Gelenkkoordinaten.

Um festzustellen, wo sich ein Beinende aktuell gerade befindet, müssen aus den Gelenkkoordinaten die kartesischen Koordinaten des Beinendes gewonnen werden. Diese werden durch die direkten kinematischen Koordinatentransformationen (weiterhin „direkte Kinematik“ bezeichnet) beschrieben

$$\vec{x} = f(\vec{q})\,. \tag{5.76}$$

Die Gewinnung von Sensorsignalen selbst ist nicht ausreichend, um sie im Rahmen des Gesamtsystems oder zumindest im Rahmen des Teilsystems Bein zu interpretieren. Es spielt zusätzlich zur eigentlichen Sensorsignalinformation auch die räumliche bzw. örtliche Zuordnung eine Rolle, um die Sensorinformationen zu interpretieren. So ist es beispielsweise von Interesse, wo eine Kraft wirkt oder wie weit ein Hindernis oder Objekt von einem Bein oder dem Roboter entfernt ist. Deshalb müssen Sensorinformationen, wenn sie nicht nur lokal genutzt werden, etwa zur Positionsregelung eines einzelnen Gelenkes, auf ein gemeinsames Koordinatensystem bezogen werden. Dazu werden mehrere Sensorwerte zu einem Vektor zusammengefasst bzw. im Rahmen eines betrachteten Teilmodells interpretiert. Das Zusammenfassen mehrerer Sensorwerte zu einem neuen Sensorwert wird als Fusion oder virtueller Sensor bezeichnet. Beispielsweise können drei orthogonale Kraftkomponenten zu einem Kraftvektor zusammengefasst werden. Das Interpretieren der Sensordaten innerhalb eines Modells zur Ausführung einer Aufgabe wird als Integration bezeichnet [Ihme 00a], [Luo, Kay 89]. Beispielsweise werden durch die direkte Kinematik die Gelenkpositionen in das kinematische Beinmodell integriert, so dass die Beinposition in kartesischen Koordinaten mit Hilfe des Beinmodells interpretiert wird. Diese Vorgehensweise kann mehrstufig erfolgen. Der bereits erwähnte Kraftvektor kann nun mit der Beinposition zusammengefasst werden, so dass ein gebundener Kraftvektor entsteht. Außer den Positionswerten müssen noch die Kraftmesswerte einer Verarbeitung unterzogen werden, um sie im Beinkoordinatensystem interpretieren zu können. Der Übergang von den Sensorsignalen des Kraftsensors zum Kraftvektor im Beinkoordinatensystem wird weiterhin als Krafttransformation bezeichnet.

#### 5.2.5.1 Direkte Kinematik

Die Position des Beinendes im Raum wird durch die Stellung der Gelenke des Beines bestimmt. Die Darstellung kann in den kartesischen Raum überführt werden, was durch die direkte Kinematik beschrieben wird und ist durch Gleichung (5.76) repräsentiert.

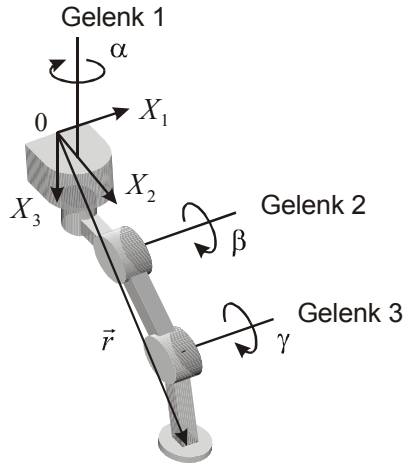
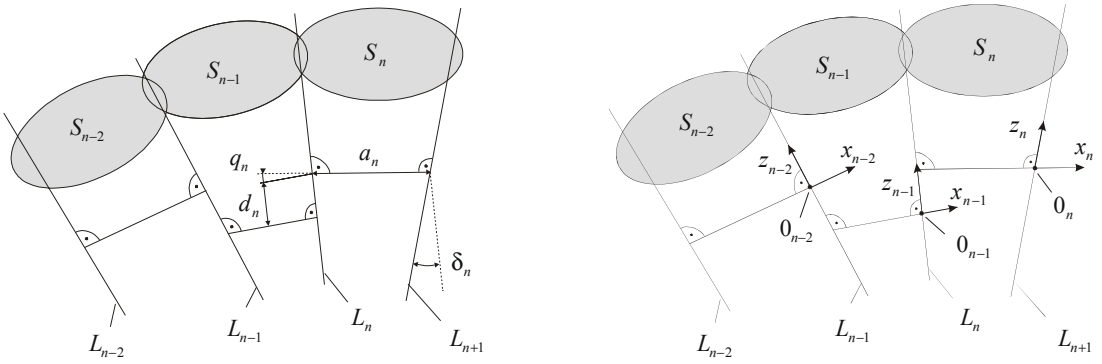



Abbildung 5.34: Koordinatensysteme eines Beines für die Kinematik

Herleitung der direkten Kinematik soll das Bein als ein System von starren Körpern aufgefasst werden. Die Gelenkwinkel stellen dabei die Freiheitsgrade dar und dienen als Eingangsgrößen. Abbildung 5.34 zeigt den Aufbau eines Beines und die dazugehörigen Koordinatensysteme. Die Pfeile markieren positive Richtungen. Das Koordinatensystem $BX_1X_2X_3$ repräsentiert das Beinkoordinatensystem, dessen Ursprung in der Mitte der den Roboter kontaktierenden Fläche liegt (siehe Abbildung 5.22 in Abschnitt 5.2.1.1). α, β und γ repräsentieren die Freiheitsgrade. Dabei ist der Winkel α gleich Null, wenn das erste Gelenk in Richtung $X_2$-Achse des Koordinatensystems $BX_1X_2X_3$ zeigt, der Winkel β ist Null, wenn das bewegte Beinsegment parallel zur Ebene $BX_1X_2$ ist und der Winkel γ ist Null, wenn das dritte Gelenk in die gleiche Richtung wie das zweite Gelenk zeigt. Der Vektor $\vec{r}$ ist der Vektor vom Ursprung des Beins zum Beinende.




a) Parameter nach Denavit und Hartenberg b) Achskoordinatensysteme

Abbildung 5.35: Kette von verbundenen starren Körpern

Durch eine systematische Zuordnung lässt sich eine Kette von starren Körpern durch vier Parameter pro Segment $S_n$ beschreiben [Wloka 92], die von Denavit und Hartenberg eingeführt worden sind. Abbildung 5.35 zeigt symbolisch eine Kette von starren Körpern. Jeder Freiheitsgrad wird durch eine Rotation $q_n$ um die Achse $L_n$ repräsentiert. Der Abstand zwischen den Achsen wird mit Hilfe der Normalen bestimmt, die sich zwischen den beiden Achsen konstruieren lässt und wird durch den Parameter $a_n$ gegeben. Wird die die beiden Geraden verbindende Normale als Flächennormale einer Projektionsebene für die beiden Achsen genutzt, so lässt sich mit Hilfe einer Parallelprojektion die Verdrehung der beiden Achsen gegeneinander bestimmen. Diese Verdrehung wird mit dem Parameter $\delta_n$ beschrieben. Der Achsversatz $d_n$ ist der Abstand zwischen den Normalen. Die Zuordnung der geometrischen

Parameter eines Beines zeigt Abbildung 5.36 und die Zuordnung zu den entsprechenden Parametern nach Denavit-Hartenberg (DH-Parameter) ist Tabelle 5.2 zu entnehmen.

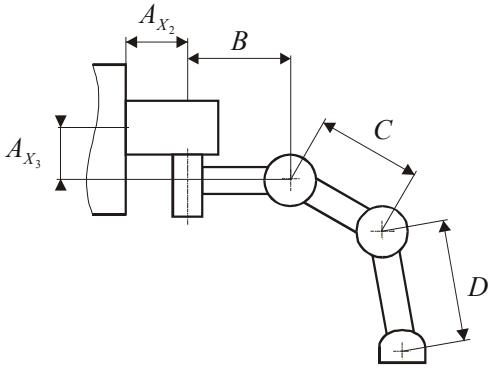



Abbildung 5.36: Geometrische Parameter eines Beins

Tabelle 5.2: Denavit-Hartenberg-Parameter für ein Bein

| Link | $a_n$ | $\delta_n$ | $d_n$ | $q_n$ |
|---|---|---|---|---|
| 1 | $A_{x2}$ | $\pi/2$ | 0 | 0 |
| 2 | $B$ | $-\pi/2$ | $A_{x3}$ | $\alpha$ |
| 3 | $C$ | 0 | 0 | $\beta$ |
| 4 | $D$ | 0 | 0 | $\gamma$ |

Der Übergang von einem Achskoordinatensystem (siehe Abbildung 5.35b) zu einem in der Kette folgenden Achskoordinatensystem lässt sich mit Hilfe von vier Elementartransformationen

$$ {}^{n-1}_{\mathrm{n}}\mathrm{T}_{\mathrm{n}} = \mathrm{Rot}(z_{n-1}, q_n) \cdot \mathrm{Trans}(0,0,d_n) \cdot \mathrm{Trans}(a_n,0,0) \cdot \mathrm{Rot}(x_n, \delta_n) \qquad (5.77) $$

ausdrücken. Die in Gleichung (5.77) angeführten Elementartransformationen entsprechen folgender Vorschrift:

1. Rotation um die Achse $z_{n-1}$ mit dem Winkel $q_n$,
2. Translation entlang der Achse $z_{n-1}$ mit dem Weg $d_n$,
3. Translation entlang der Achse $x_n$ mit dem Weg $a_n$ und
4. Rotation um die Achse $x_n$ mit dem Winkel $\delta_n$.

Durch Verwendung homogener Koordinaten lassen sich die Transformationen für die gesamte kinematische Kette durch Multiplikation der Transformationsmatrizen miteinander verknüpfen

$$ {}^{0}_{4}\mathrm{T} = {}^{0}_{1}\mathrm{T} \cdot {}^{1}_{2}\mathrm{T} \cdot {}^{2}_{3}\mathrm{T} \cdot {}^{3}_{4}\mathrm{T} \, . \qquad (5.78) $$

Durch die Anordnung der Gelenkachsen der Beine ergeben sich Vereinfachungen notwendiger Transformationsschritte dadurch, dass Gelenkachsen parallel sind oder senkrecht aufeinander stehen. Somit werden einige DH-Parameter in Tabelle 5.2 zu Null. Es gilt

$$
\begin{aligned}
{}^{0}_{4}\mathrm{T} \;=\; & \mathrm{Trans}\left(A_y,0,0\right)\cdot \mathrm{Rot}\left(x_1,90°\right)\cdot \\
& \mathrm{Rot}\left(z_1,\alpha\right)\cdot \mathrm{Trans}\left(0,0,A_z\right)\cdot \mathrm{Trans}\left(B,0,0\right)\cdot \mathrm{Rot}\left(x_2,-90°\right)\cdot \\
& \mathrm{Rot}\left(z_2,\beta\right)\cdot \mathrm{Trans}\left(C,0,0\right)\cdot \\
& \mathrm{Rot}\left(z_3,\gamma\right)\cdot \mathrm{Trans}\left(D,0,0\right)
\end{aligned}
\tag{5.79}
$$

Durch Ausmultiplizieren der entsprechenden homogenen Transformationsmatrizen erhält man die Transformationsmatrix für das gesamte Bein. Das Ergebnis ist eine Hypermatrix, die die Teilmatrizen für die Rotation und die Translation enthält

$$
{}^{0}_{4}\mathrm{T}={}^{0}_{1}\mathrm{T}\cdot{}^{1}_{1}\mathrm{T}\cdot{}^{2}_{3}\mathrm{T}\cdot{}^{3}_{4}\mathrm{T}=\begin{pmatrix} {}^{4}_{0}\mathbf{R} & {}^{0}\vec{r} \\ \mathbf{0} & 1 \end{pmatrix} . \tag{5.80}
$$

In Gleichung (5.80) interessiert der Vektor zum Fußende ${}^{0}\vec{r}$. Für die in Abbildung 5.34 gezeigte Beinkonfiguration ergibt sich dieser Vektor in Beinkoordinaten zum Fußende mit

$$
\begin{pmatrix} r_{X_1} \\ r_{X_2} \\ r_{X_3} \end{pmatrix} = \begin{pmatrix} \left(B + C\cdot\cos(\beta) + D\cdot\cos(\beta+\gamma)\right)\cdot\sin(\alpha) \\ A_{X_2} + \left(B + C\cdot\cos(\beta) + D\cdot\cos(\beta+\gamma)\right)\cdot\cos(\alpha) \\ A_{X_3} + C\cdot\sin(\beta) + D\cdot\sin(\beta+\gamma) \end{pmatrix} . \tag{5.81}
$$

Darin ist $(r_{X_1}\ r_{X_2}\ r_{X_3})^T$ der Ortsvektor im Beinkoordinatensystem zum Fußpunkt; $A_{X_2}$, $A_{X_3}$, $B$, $C$ und $D$ sind konstante geometrische Parameter (siehe Abbildung 5.36); $\alpha$, $\beta$ und $\gamma$ sind Gelenkwinkel, das heißt Freiheitsgrade $q_i$, die variabel sind.

Im Ergebnis entsteht eine Transformation, um aus gegebenen Gelenkwinkeln die kartesischen Koordinaten im Beinkoordinatensystem zu bestimmen.

#### 5.2.5.2 Inverse Kinematik

Besonders einfach ist die Beschreibung von Trajektorien der Roboterbewegung im kartesischen Raum. Daher werden die Bewegungen der Beine im kartesischen Raum koordiniert und erzeugt, so dass ausgehend von den vorgesehenen kartesischen Koordinaten des Beinendes bestimmt werden muss, welche Gelenkwinkel notwendig sind, um diese Position zu realisieren. Beispielsweise ist eine Gerade sehr einfach mit kartesischen Koordinaten beschreibbar, indem eine oder mehrere Koordinaten kontinuierlich verändert werden. Im Gelenkraum müssen sich jedoch zur Ausführung der gleichen Bewegung alle Gelenke in Abhängigkeit ihrer aktuellen Position unterschiedlich bewegen. Der Zusammenhang der inversen Kinematik wird formal durch Gleichung (5.75) beschrieben. Da es keine direkte inverse Funktion zur direkten Kinematik gibt, muss Gleichung (5.76) nach $q_i$ aufgelöst werden. Aus der Lösung für die direkte Kinematik (Gleichung (5.81)) ist zu erkennen, dass es keine einfach zu erzeugende inverse Funktion gibt. Um zu einer Lösung des Problems für die inverse Kinematik zu gelangen, gibt es zwei verschiedene Vorgehensweisen. Die eine Vorgehensweise versucht, eine analytische Lösung zu finden, die andere arbeitet mit numerischen Methoden.

Die Lösung der inversen Kinematik ist aus folgenden Gründen wesentlich komplizierter als die für die direkte Kinematik [Sciavicco, Siciliano 00]:

- die zu lösenden Gleichungen sind nichtlinear und somit kann nicht immer eine geschlossene Lösung gefunden werden,
- es existieren mehrfache Lösungen,
- es können unendlich viele Lösungen existieren, z.B. bei kinematisch redundanten Strukturen und
- es ist möglich, dass in Abhängigkeit von mechanischen Randbedingungen keine zulässige Lösung existiert.

Eine analytische Lösung ist in jedem Fall vorzuziehen, da sich dann die inverse Kinematik sehr effizient berechnen lässt. Das ist insbesondere dann möglich, wenn parallele oder senkrecht aufeinanderstehende Achsen vorhanden sind [Wloka 92]. Beim hier betrachteten Roboterbein sind zwei Achsen parallel und zwei Achsen stehen senkrecht aufeinander. Mit dieser Anordnung entsteht keine kinematische Redundanz.

Zur Herleitung einer analytischen Lösung wird das Problem auf zwei zweidimensionale Koordinatensysteme zurückgeführt:

1. Ein XY-System, das sich auf der von der $X_1$- und $X_2$-Achse des vom kartesischen Koordinatensystems aufgespannten Ebene befindet und dessen Ursprung auf der ersten Gelenkachse liegt ($X_1$-$X_2$-Ebene) und

2. Ein RZ-System, das sich auf der von der Normalen der ersten Gelenkachse zum nächsten Gelenk (R) und der Achse des Gelenkes aufgespannten Ebene liegt (Z) und dessen Ursprung auf dem Schnittpunkt zwischen zweiter Gelenkachse und der Normalen liegt.

Die Gelenkachsen des ersten und zweiten Gelenkes stehen senkrecht aufeinander, die Gelenkachsen des zweiten und dritten Gelenkes sind parallel und *r* sei der Abstand zum Fuß im XY-System, somit lässt sich mit Hilfe des XY-Systems der erste Gelenkwinkel $\alpha$ bestimmen

$$\begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = \begin{pmatrix} r \cdot \sin(\alpha) \\ r \cdot \cos(\alpha) \\ 1 \end{pmatrix} . \tag{5.82}$$

Da die Koordinaten aber nur im Beinkoordinatensystem bekannt sind, muss über die Koordinaten eine Translation vom Ursprung des XY-Systems zum Ursprung des Beinkoordinatensystems

$${}^{XY}_{B}\mathrm{T} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -A_{x_2} \\ 0 & 0 & 1 \end{pmatrix} \tag{5.83}$$

ausgeführt werden. Somit folgt

$$\begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = {}^{XY}_{B}\mathrm{T} \cdot {}^{B}\vec{\mathrm{r}}_{O,F} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -A_{x_2} \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ 1 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 - A_{x_2} \\ 1 \end{pmatrix}. \tag{5.84}$$

Setzt man Gleichung (5.82) in Gleichung (5.84) ein und dividiert X durch Y, so erhält man die Lösung für den Gelenkwinkel α

$$\alpha = \arctan\left(\frac{X}{Y}\right) = \arctan\left(\frac{r_{x_1}}{r_{x_2} - A_{x_2}}\right). \tag{5.85}$$

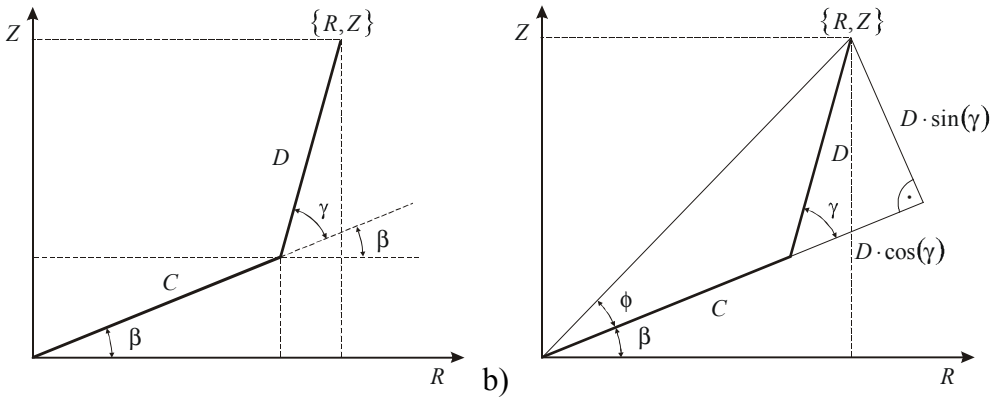



Abbildung 5.37: Geometrische Beziehungen im *RZ*-System

Abbildung 5.37 zeigt die geometrischen Beziehungen im *RZ*-System. Es ist ersichtlich, dass mit seiner Hilfe die noch fehlenden Gelenkwinkel β und γ bestimmt werden können. Die Koordinatentransformation für das *RZ*-System hat folgende Form

$$\begin{pmatrix} R \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} C \cdot \cos(\beta) + D \cdot \cos(\beta + \gamma) \\ C \cdot \sin(\beta) + D \cdot \sin(\beta + \gamma) \\ 1 \end{pmatrix}. \tag{5.86}$$

Wendet man die Additionstheoreme an, so ergibt sich

$$\begin{pmatrix} R \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} C \cdot \cos(\beta) + D \cdot \cos(\beta) \cdot \cos(\gamma) - D \cdot \sin(\beta) \cdot \sin(\gamma) \\ C \cdot \sin(\beta) + D \cdot \sin(\beta) \cdot \cos(\gamma) + D \cdot \cos(\beta) \cdot \sin(\gamma) \\ 1 \end{pmatrix}. \tag{5.87}$$

Die Gleichungen für R und Z in Gleichung (5.87) können genutzt werden, um die fehlenden Gelenkwinkel β und γ zu bestimmen. Zunächst quadriert man beide Gleichungen und addiert sie anschließend. Man erhält

$$R^2 + Z^2 = C^2 + 2 \cdot C \cdot D \cdot \cos(\gamma) + D^2. \tag{5.88}$$

Durch Umstellen dieser Gleichung erhält man den Gelenkwinkel γ

$$\gamma = \arccos\left(\frac{1}{2}\cdot\frac{R^2+Z^2-C^2-D^2}{C\cdot D}\right). \tag{5.89}$$

Aus Abbildung 5.37b ist ersichtlich, dass die folgenden Beziehungen gelten:

$$\frac{Z}{R} = \tan(\phi+\beta) = \frac{\tan(\phi)+\tan(\beta)}{1-\tan(\phi)\cdot\tan(\beta)} \tag{5.90}$$

$$\tan(\phi) = \frac{D\cdot\sin(\gamma)}{C+D\cdot\cos(\gamma)}\ . \tag{5.91}$$

Durch Einsetzen von Gleichung (5.91) in Gleichung (5.90) und Auflösung nach β ergibt sich

$$\beta = \arctan\left(\frac{Z\cdot C+Z\cdot D\cdot\cos(\gamma)-R\cdot D\cdot\sin(\gamma)}{R\cdot C+R\cdot D\cdot\cos(\gamma)+Z\cdot D\cdot\sin(\gamma)}\right). \tag{5.92}$$

Damit liegen für alle drei Gelenkwinkel die analytischen Lösungen vor. Das RZ-Koordinatensystem muss noch in eine entsprechende Darstellung im Beinkoordinatensystem überführt werden. Dazu wird eine Transformation benutzt, die den Ursprung des Beinkoordinatensystems mit dem des *RZ*-Koordinatensystems verbindet

$${}^{RZ}_{B}\mathrm{T} = \begin{pmatrix} 1 & 0 & 0 & A_{x_1} \\ 0 & 1 & 0 & A_{x_2} \\ 0 & 0 & 1 & A_{x_3} \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{E} & \mathbf{A} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{pmatrix}. \tag{5.93}$$

Damit ist die Koordinate *Z* im *RZ*-System bekannt

$$Z = r_{x_3} - A_{x_3}\,. \tag{5.94}$$

Die Koordinate *R* lässt sich aus den Koordinaten *X* und *Y* des *XY*-Systems über den Satz des Pythagoras unter Berücksichtigung des Koordinatenursprungs bestimmen. Es ist

$$R^2 = X^2 + Y^2\,. \tag{5.95}$$

Verwendet man die Transformation (5.84), so lässt sich *R* aus den Beinkoordinaten gewinnen

$$R = \sqrt{r_{x_1}^2 + \left(r_{x_2} - A_{x_2}\right)^2} - B\,. \tag{5.96}$$

Mit der Möglichkeit die Koordinaten X und Z, sowie die drei Gelenkwinkel zu bestimmen, ist die analytische Lösung der inversen Kinematik vollständig.

*Existenz von Lösungen*

Für die Gelenkwinkel α und ß existieren rein mathematisch gesehen, nur dann Lösungen, wenn die Nenner in den Gleichungen (5.85) und (5.92) ungleich Null sind. Da nicht auszu-

schließen ist, dass die Nenner in Einzelfällen doch einmal den Wert Null annehmen bzw. gegen den Grenzwert Null streben und dies zu Schwierigkeiten bei der rechentechnischen Implementierung der Gleichungen führen kann, wird für die Implementierung auf eine spezielle Arcustangensfunktion zurückgegriffen, wie sie in modernen Programmiersprachen, wie ANSI C, zur Verfügung steht. Diese Funktion wird mit atan2(X,Y) bezeichnet und mit ihr werden der Zähler und Nenner getrennt zur Berechnung übergeben. Diese Funktion hat den Vorteil, dass sie alle vier Quadranten berücksichtigt. Sie liefert nur dann keine Lösung, wenn Zähler und Nenner gleichzeitig den Wert Null annehmen, das heißt ein Singularitätsfall vorliegt.

Für den Gelenkwinkel $\alpha$ liegt nach Gleichung (5.85) eine Singularität vor, wenn $r_{x_1} = 0$ und $r_{x_2} - A_{x_2} = 0$ gilt. In diesem Fall befindet sich das Beinende genau auf der verlängerten Achse des ersten Gelenkes.

Für den Gelenkwinkel $\beta$ ergeben sich Singularitäten entsprechend Gleichung (5.92), wenn gilt

a) $C + D \cdot \cos(\gamma) = 0$ und $\sin(\gamma) = 0$
   Dieser Fall tritt nur bei $\gamma$=180° und *C*=*D* auf. Das bedeutet, dass das Beinende auf der $\beta$-Achse liegt. Dieser Fall kann durch Fallprüfung in Gleichung (5.89) ausgeschlossen werden. Außerdem ist diese Lösung mechanisch nicht sinnvoll.

b) R = 0 und $Z = 0$
   Der Fuß liegt auf der Beta-Achse, das heißt $\gamma$=180°. Das kann durch Fallprüfung in Gleichung (5.89) ausgeschlossen werden. Außerdem ist diese Lösung mechanisch nicht sinnvoll.

Die Gleichung (5.89) für den Gelenkwinkel $\gamma$ hat nur dann eine Lösung, wenn

$$-1 \le \frac{1}{2} \frac{R^2 + Z^2 - C^2 - D^2}{CD} \le 1 \tag{5.97}$$

gilt. Mit Hilfe von Umformungen und der Substitution $r^2 = R^2 + Z^2$ ergibt sich

$$(C - D)^2 \le r^2 \le (C + D)^2 . \tag{5.98}$$

Das von C und D aufgespannte Dreieck kann nur existieren, wenn diese Bedingung erfüllt ist.

*Eindeutigkeit von Lösungen*

Bei der inversen Kinematik zeigt sich das Problem, dass die Lösungen für die drei Gelenkwinkel nicht eindeutig sind. Bei den trigonometrischen Funktionen treten periodische Lösungen auf, die hier generell auf einen Lösungsbereich von ±90° eingeschränkt werden können. In Gleichung (5.89) gibt es zwei mögliche Lösungen, die durch die Symmetrie der Kosinusfunktion entstehen

$$\cos(\gamma) = \cos(-\gamma) . \tag{5.99}$$

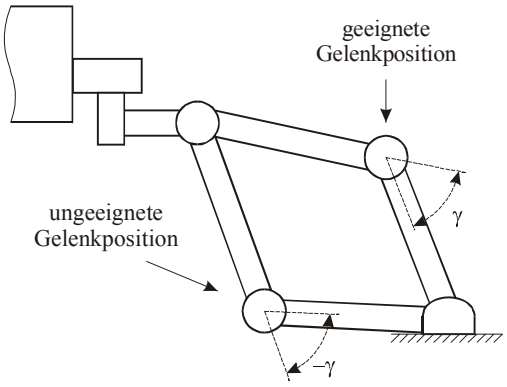



Abbildung 5.38: Mehrfachlösungen bei der inversen Kinematik

Abbildung 5.38 zeigt, dass zum Laufen auf dem Boden nur positive $\gamma$ sinnvoll sind. Für Gleichung (5.89) gilt deshalb folgende Einschränkung

$$0 \le \gamma \le \pi \,. \tag{5.100}$$

Der Gelenkwinkel $\beta$ ist abhängig vom gewählten Wert für $\gamma$ und somit eindeutig.

Eine weitere Mehrdeutigkeit kann durch Gleichung (5.96) entstehen, da die Wurzel eine positive und eine negative Lösung besitzt. Hier kann der Lösungsbereich durch Zulässigkeit von Lösungen eingeschränkt werden. Unter diesem Aspekt können folgende Lösungsmöglichkeiten unterschieden werden

$$R = \begin{cases} +\sqrt{r_{X_1}^2 + \left(r_{X_2} - A_{X_2}\right)^2} - B & \text{für} \quad r_{X_2} - A_{X_2} > 0 \\ \pm\sqrt{r_{X_1}^2 + \left(r_{X_2} - A_{X_2}\right)^2} - B & \text{für} \quad r_{X_2} - A_{X_2} = 0 \\ -\sqrt{r_{X_1}^2 + \left(r_{X_2} - A_{X_2}\right)^2} - B & \text{für} \quad r_{X_2} - A_{X_2} < 0 \end{cases} . \tag{5.101}$$

Die beiden letzten Fälle können ausgeschlossen werden, da sie nicht zu Lösungen innerhalb des erlaubten Lösungsbereiches führen. Diese Einschränkung kann einerseits durch Gleichung (5.85) geprüft werden, andererseits werden nur Fälle mit $r_{X_2} - A_{X_2} > 0$ betrachtet.

*Zulässigkeit von Lösungen*

Es werden nur solche Lösungen als zulässig angesehen, die

- innerhalb des Arbeitsraumes liegen und
- zu einer kinematisch sinnvollen Konfiguration führen.

Die meisten Beschränkungen ergeben sich durch die Eingrenzung des Arbeitsraumes durch sinnvolle Gelenkwinkelbereiche

$$-60° \le \alpha \le +60° \tag{5.102}$$
$$-30° \le \beta \le +90° \tag{5.103}$$
$$0° < \gamma \le +90° \,. \tag{5.104}$$

Eine weitere Einschränkung wird für die Fallunterscheidung in Gleichung (5.101) vorgenommen

$$r_{X_2} - A_{X_2} > 0\,. \tag{5.105}$$

Damit wird der Arbeitsraum nur auf den „äußeren“ Bereich eingeschränkt. Mit den Beschränkungen für zulässige Lösungen werden gleichzeitig Konfigurationen vermieden, die zu Singularitäten führen. Die Trajektorien müssen so geplant werden, dass sie innerhalb des Arbeitsraumes realisiert werden können.

Im Ergebnis liegen die Gelenkkoordinaten bzw. Gelenkwinkel $\alpha$, $\beta$ und $\gamma$ in Abhängigkeit von den Beinkoordinaten $r_{x_1}$, $r_{x_2}$ und $r_{x_3}$ vor. Es wurde gezeigt, dass für die inverse Kinematik eine analytische Lösung existiert. Somit können die Gelenkwinkel ohne den Umweg numerischer Näherungsverfahren direkt berechnet werden.

#### 5.2.5.3 Krafttransformationen

Der Kraftsensor jedes Beines liefert Informationen über wirkende Kräfte. Der Kraftsensor ist so aufgebaut, dass er drei senkrecht aufeinanderstehende Kraftkomponenten registriert. Die Messwerte werden auf den Fußpunkt bezogen. Ort und Richtung der registrierten Kräfte sind also abhängig von Position und Orientierung des Kraftsensors im Raum. Diese Sensorinformationen können also nur zusammen mit anderen Informationen sinnvoll genutzt werden, so dass die Sensorinformationen in Zusammenhang mit dem aktuell betrachteten System bzw. Teilsystem interpretiert werden müssen. Die Interpretation kann entweder auf das jeweilige Bein oder auf den gesamten Roboter bezogen erfolgen. Innerhalb des hierarchischen Systems erfolgt die Interpretation zunächst mit Bezug auf das aktuelle Bein. Liegt die Kraftinformation in dieser Form vor, kann sie später mit Hilfe der bekannten Robotergeometrie auf den gesamten Roboter bezogen werden.

Beim Laufroboter Katharina befindet sich der Drei-Komponenten-Kraftsensor im Unterschenkel des Beines. Durch die nicht veränderliche Geometrie des Unterschenkels kann davon ausgegangen werden, dass die Kräfte in einem bestimmten Punkt, dem Fuß, angreifen. Diese Annahme ist gerechtfertigt, da der Kontakt zur Umwelt über den Untergrund hergestellt wird. Die über passive Gelenke befestigte Fußplatte dient der Entkopplung von Momenten, die durch Drehung des Fußes auf dem Untergrund entstehen könnten. Ihre Gewichtskraft ist klein gegenüber den üblicherweise auftretenden Kräften, so dass ihre Wirkung vernachlässigt werden kann. Bei Bedarf kann diese zusätzliche Kraft später rechnerisch kompensiert werden.

Die vom Kraftsensor erfassten Kraftkomponenten $F_A$, $F_B$ und $F_C$ sind orthogonal zueinander und entsprechend der räumlichen Lage des Kraftsensors ausgerichtet. Da er fest in den Unterschenkel integriert ist, ist seine Lage immer identisch mit der des Unterschenkels. Die räumliche Anordnung zeigt Abbildung 5.39. Da die wirkenden Kräfte immer als am Fuß angreifend angenommen werden, kann die räumliche Orientierung des Kraftangriffspunktes mit Hilfe kinematischer Beziehungen des Beines bestimmt werden, das heißt die Beziehungen für die direkte Kinematik können hier genutzt werden. Die Transformationsgleichung (5.80) enthält eine Hypermatrix, wobei ${}^0\vec{r}$ die Position des Fußpunktes, das heißt die des Kraftangriffs-

punktes und ${}^{4}_{0}\mathbf{R}$ die Rotation des Koordinatensystems am Beinende gegenüber dem Beinkoordinatensystem ist.

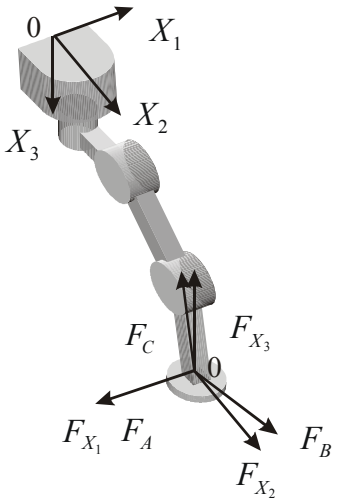



Abbildung 5.39: Koordinatensysteme für Kräfte am Bein

Die Rotationsmatrix ${}^{4}_{0}\mathbf{R}$ kann genutzt werden, um wirkende Kräfte so darzustellen, dass ihre Komponenten parallel zu denen des Beinkoordinatensystems sind. Für die Transformation der Signale der Kraftsensorkomponenten in zum Beinkoordinatensystem parallele Komponenten ergibt sich folgende Beziehung

$$\begin{pmatrix} F_{X_1} \\ F_{X_2} \\ F_{X_3} \end{pmatrix} = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha)\cdot\sin(\beta+\gamma) & -\sin(\alpha)\cdot\cos(\beta+\gamma) \\ \sin(\alpha) & \cos(\alpha)\cdot\sin(\beta+\gamma) & \cos(\alpha)\cdot\cos(\beta+\gamma) \\ 0 & -\cos(\beta+\gamma) & \sin(\beta+\gamma) \end{pmatrix} \cdot \begin{pmatrix} F_A \\ F_B \\ F_C \end{pmatrix} \tag{5.106}$$

mit $[BX_1X_2X_3] \parallel [0F_{X_1}F_{X_2}F_{X_3}]$.

Der angenommene Angriffspunkt der Kraft fällt mit dem Beinende zusammen, das mit $r_{X_1}$, $r_{X_2}$, $r_{X_3}$ gegeben ist. Diese beiden Informationen lassen sich zu einem gebundenen Kraftvektor $\vec{F}_{r/F}$ zusammenfügen.

#### 5.2.5.4 Gelenkregelung

In Abschnitt 5.2.5.2 wurde beschrieben, wie die Sollposition eines Beines von kartesischen Koordinaten im Beinkoordinatensystem in Gelenkwinkel überführt wird. Umgekehrt können Gelenkwinkel auch in kartesische Koordinaten überführt werden (Abschnitt 5.2.5.1). Der Nachteil der Darstellung im kartesischen Raum ist, dass bei Bewegung eines einzelnen Gelenkes gleich alle drei Koordinaten verändert werden können. Durch die Beschreibung der Beinposition mit den Gelenkwinkeln $\alpha$, $\beta$ und $\gamma$ können diese unabhängig voneinander betrachtet werden. Dadurch kann ein komplexes System auf mehrere einfachere Teilsysteme zurückgeführt werden. Die gewünschten Gelenkpositionen können so durch Regelung reali-

siert werden. Der Gelenkregelkreis ist eng an dem zur Verfügung stehenden Sensor-Aktor-System für ein einzelnes Gelenk gekoppelt. Die folgende Betrachtung gilt für die beim Laufroboter Katharina vorhanden Gelenkantriebe und –motoren. Da die Bewegungen des Roboters relativ langsam erfolgen, können sie kinematisch beschrieben werden. Das wird durch die Vorgabe einer Folge von Sollpositionen realisiert. Diese dienen nun als Sollwert für die Gelenkregelung. Der Istwert wird durch die Gelenkwinkelpotentiometer bereitgestellt. Wegen der relativ langsamen Bewegungen und den zur Verfügung stehenden Istwertinformationen kann ein einschleifiger Servo-Regelkreis aufgebaut werden.

Aus den a-priori-Informationen und der Identifikation ist bekannt, dass für jedes Gelenk ein $IT_1$-Modell angenommen werden kann. Damit werden im wesentlichen die mechanischen Zeitkonstanten erfasst. Sie liegen in der Größenordnung von 100 ms. Die elektrischen Zeitkonstanten liegen im Bereich von 0,1 bis 1 ms und werden für den Entwurf des Reglers nicht weiter betrachtet.

Um ein stabiles System zu erhalten, müssen die allgemeine Stabilitätsbedingungen für das geschlossene System

$$G_0 + 1 = 0 \tag{5.107}$$

$$G_R \cdot G_S + 1 = 0 \tag{5.108}$$

erfüllt werden [Föllinger 90]. Dabei ist $G_0$ die Übertragungsfunktion der offenen Kette, $G_R$ die Reglerübertragungsfunktion und $G_S$ die Streckenübertragungsfunktion. Dies kann durch einen $IT_1$- Entwurf erreicht werden, bei dem die offene Kette ($G_0$) eine $IT_1$-Struktur aufweist [Reinisch 82]. Ziel des Regelungsentwurfes ist, dass der Istwert dem Sollwert schnell folgt und die geschlossene Kette stabil ist. Dadurch kann auch sichergestellt werden, dass eine bleibende Regelabweichung bezüglich der Position ausgeregelt werden kann.

Für die Regelstrecke wird folgendes Modell angenommen. Die Streckenübertragungsfunktion $G_S$ sei

$$G_S = \frac{1}{T_I s \cdot (1 + T_1 s)} \tag{5.109}$$

mit $T_I$ als Integrationszeitkonstante und $T_1$ als Zeitkonstante. Zur Realisierung einer $IT_1$-Struktur der offenen Kette ist ein P-Regler notwendig. Für die geschlossene Kette ergibt sich somit folgendes Übertragungsverhalten

$$\frac{K_R}{T_I s \cdot (1 + T_1 s)} + 1 = 0\,, \tag{5.110}$$

wobei $K_R$ die Übertragungskonstante des Reglers ist. Durch die beiden Zeitkonstanten im Nenner ergibt sich ein schwingungsfähiges System

$$\frac{T_I T_1}{K_R} s^2 + \frac{T_I}{K_R} s + 1 = 0\,. \tag{5.111}$$

Durch Koeffizientenvergleich mit der Schwingungsgleichung

$$T_0^2 s^2 + 2DT_0 s + 1 = 0 \tag{5.112}$$

erhält man

$$T_0^{\,2} = \frac{T_I T_1}{K_R} \tag{5.113}$$

$$2DT_0 = \frac{T_I}{K_R} \ . \tag{5.114}$$

Durch Auflösen nach $T_0$ und anschließendes Quadrieren von (5.114) lässt sich diese mit (5.113) gleichsetzen und nach $K_R$ auflösen

$$K_R = \frac{T_I}{4D^2 T_1} \ . \tag{5.115}$$

Die Dämpfung wird für eine Überschwingweite von 2% mit D≈ 0,7 gewählt [Unbehauen 89].

Bei den bisherigen Betrachtungen wurde davon ausgegangen, dass die Regelstrecke lineares Verhalten aufweist. In der Realität zeigen sich jedoch einige Effekte, die der Regelstrecke ein nichtlineares Verhalten geben. Durch die Puls-Weiten-modulierte Ansteuerung des Gelenkantriebes treten Induktionseffekte der Ankerwicklung des Motors in Erscheinung. Nach Anlegen eines Pulses muss in der Ankerwicklung erst ein Magnetfeld aufgebaut werden, bevor ein Antriebsmoment wirken kann. Deshalb ist zusätzlich zur Bürstenspannung [Maxon 00] zu beobachten, dass sich bei kurzen Implulsen keine Drehzahl aufbaut. Dieser Effekt ist frequenzabhängig. Bei niedrigen Frequenzen ist die Totzone geringer, mit größer werdender Frequenz wird sie größer. Aus der PWM-Drehzahl-Kennlinie ist zu erkennen, dass der Kurvenanstieg außerhalb des Totbereiches zunächst linear ist. Da insbesondere in der Stützphase, wo es auf eine genauere Regelung ankommt, nur relativ langsame Bewegungen erforderlich sind, kann auf ein Modell mit Totzone und linearem Anstieg zurückgegriffen werden. Damit wird gleichzeitig der Arbeitsbereich der Regelung auf den unteren Drehzahlbereich begrenzt. Schnellere Bewegungen sind hingegen nur in der Returnphase notwendig. Allerdings ist hier die Genauigkeit nicht von primärem Interesse. Das Modell der PWM-Drehzahl-Kennlinie wird somit wie folgt dargestellt

$$d = \begin{cases} -\mathrm{PWM}_{\max} \le \mathrm{PWM} \le -\mathrm{PWM}_0 & : \ \omega = k_d \cdot (\mathrm{PWM} + \mathrm{PWM}_0) \\ -\mathrm{PWM}_0 < \mathrm{PWM} < +\mathrm{PWM}_0 & : \ \omega = 0 \\ +\mathrm{PWM}_0 \le \mathrm{PWM} \le -\mathrm{PWM}_{\max} & : \ \omega = k_d \cdot (\mathrm{PWM} - \mathrm{PWM}_0) \end{cases} \tag{5.116}$$

mit

| | | |
|---|---|---|
| PWM | - | Betrag der Puls-Weiten-Modulation |
| $\mathrm{PWM}_0$ | - | Betrag der Stelle des Kennlinienknicks im positiven und negativen Bereich der Streckenübertragungsfunktion |
| $\mathrm{PWM}_{\max}$ | - | Maximaler Betrag der Puls-Weiten-Modulation |
| $k_d$ | - | Steigungsfaktor der Kennlinie außerhalb des Totbereichs |
| ω | - | Drehzahl. |

Das Modell lässt sich nun zur Konstruktion einer inversen Kennlinie nutzen, mit der die Nichtlinearität kompensiert werden kann

$$\mathrm{PWM} = \mathrm{PWM}_R + \mathrm{PWM}_0 \cdot \mathrm{sign}\left(\mathrm{PWM}_R\right), \tag{5.117}$$

wobei $\mathrm{PWM}_R$ die Stellgröße bei Nutzung der kompensierten Kennlinie ist.

Nach dem Inneren-Modell-Prinzip werden die nichtlineare Regelstrecke und die Kompensations-Kennlinie zu einem linearen Modell der Regelstrecke abstrahiert

$$G_S = G_{S,NL} \cdot G_{S,lin}, \tag{5.118}$$

wobei $G_{S,NL}$ der nichtlineare Anteil der Streckenübertragungsfunktion und $G_{S,lin}$ die Kompensationsfunktion zur Linearisierung der Streckenübertragungsfunktion ist. Damit ergibt sich eine lineare Übertragungsfunktion der Regelstrecke. Abbildung 5.40 zeigt den Regelkreis mit der Kennlinienlinearisierung. Somit sind Methoden der linearen Regelungstheorie anwendbar.

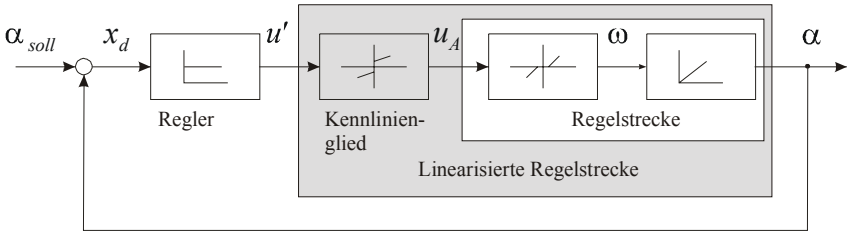



Abbildung 5.40: Gelenkregelkreis mit Linearisierung der Regelstrecke

### 5.2.5.5 Kraftgeführte Bewegungen

Bei Laufrobotern entsteht durch die Art der Fortbewegung ein hohes Maß an Interaktion mit dem Untergrund. Hinzu kommen weitere, aufgabenbezogene Interaktionen mit der Umgebung.

Aus den auftretenden Kraftreaktionen lassen sich zum einen mit Hilfe binärer taktiler Sensoren Entscheidungen für die Bewegungskoordinierung ableiten [Whitney 87], etwa das Erreichen des Untergrundes oder das Auftreffen auf ein Hindernis. Diese Informationen werden für die Bewegungsplanung weiterverwendet, sie führen zu einer lokalen, auf ein Bein bezogenen, oder zu einer globalen Neuplanung der Bewegungen.

Zum anderen lassen sich mit Hilfe von Kraftsensoren kontinuierliche Bewegungen beeinflussen. Diese Möglichkeit wird nachfolgend näher betrachtet.

#### 5.2.5.5.1 Prinzipien der Kraftregelung

Die Beeinflussung von Bewegungen bzw. die Bewegungsführung beruht auf dem Prinzip, dass ein bestimmter Zusammenhang zwischen Position bzw. Bewegung und Kraft hergestellt wird [Whitney 87]. Dieses Prinzip kann entweder dazu genutzt werden, um eine ausgeführte

Bewegung zu modifizieren oder um eine oder mehrere Bewegungskomponenten abhängig von wirkenden Kräften zu gestalten. Die erste Variante kann innerhalb des Konzeptes der Mehrebenensteuerung angewandt werden, bei der eine unterlagerte Ebene die vorgegebenen Bewegungen in Grenzen modifizieren kann [Bernstein 88]. Mit Hilfe der zweiten Variante kann gezielt mit externen Objekten interagiert werden. Dabei wird es mit einer bestimmten Kraft beaufschlagt, so dass sich das Objekt bewegt, verformt, eine geführte Bewegung entsteht oder eine Schneidkraft eingebracht wird. Beispiele hierfür sind das Einsetzen von Teilen, das Bohren oder das Verdichten des Bodens zum Test dessen Tragfähigkeit [Sinha, Bajcsy 92].

Je nach Zielstellung können unterschiedliche Methoden zur Kraftregelung eingesetzt werden, wie sie in [Whitney 87], [Shutter et al. 98] vorgestellt wurden. Neben der reinen Positions- oder Geschwindigkeitsregelung ist auch eine reine Kraftregelung möglich. Dabei spielen Position und Geschwindigkeit keine Rolle, außer in Form von Begrenzungen, wie maximale Geschwindigkeit oder Positionsgrenzen. Aktionskräfte können auch indirekt festgelegt werden, indem Stellkräfte eines Positionsreglers gezielt begrenzt werden, so dass sich ein Kräftegleichgewicht einstellt. Dies wird auch als implcit force control bezeichnet.

In der Regel besteht jedoch das Ziel, eine Position oder eine Geschwindigkeit bei einer Interaktion mit der Umwelt so zu beeinflussen, dass bestimmte Kraftwirkungen entstehen bzw. aus Kraftwirkungen bestimmte Geschwindigkeiten oder Positionen resultieren. Dazu müssen Positions- bzw. Geschwindigkeitsregelung und Kraftregelung miteinander kombiniert werden. Dieses Ziel kann mit Stiffness Control, Damping Control oder Impedance Control verfolgt werden. Diese Reglertypen kombinieren die Größen Position und/oder Geschwindigkeit mit der Kraft.

*Der Impedanzregler*

Der Impedanzregler ist eine allgemeine Form des Kraftreglers. Impedanz bedeutet, dass das Verhältnis einer kraftbezogenen Größe, wie zum Beispiel Drehmoment oder Kraft zu einer der Zeit reziproken Größe geregelt wird. Dies wird auch als mechanische Impedanz bezeichnet. Die mechanische Impedanz ist eine Analogie zur elektrischen Impedanz, die ein Verhältnis aus Spannung und Strom ist. Beim Impedanzregler wird nicht nur die Kraft selbst geregelt, sondern sie wird in Zusammenhang mit der Position oder der Geschwindigkeit geregelt. Das hat gegenüber der reinen Kraftregelung den Vorteil, dass auch Positionen und/oder Geschwindigkeiten einbezogen werden. In [Volpe, Khosla 95] wird hierzu näher auf einen Impedanzregler zweiter Ordnung eingegangen. Unter bestimmten Randbedingungen lassen sich aus dem Impedanzregler reine Kraftregler, Dämpfungs- oder Nachgiebigkeitsregler ableiten. [Whitney 87] verfolgt den umgekehrten Weg und kombiniert Nachgiebigkeits- und Dämpfungsregler zum Impedanzregler. Das allgemeine Regelungsgesetz eines Impedanzreglers zweiter Ordnung aus [Volpe, Khosla 95] lautet

$$F = \left(K_{FT} s^2 + K_{FD} s + K_{FS}\right) \cdot X \tag{5.119}$$

mit
- $F$ - Kraft
- $K_{FT}$ - Trägheit
- $K_{FD}$ - Dämpfung
- $K_{FS}$ - Steifheit
- $X$ - Position
- $s$ - Laplace-Operator.

Da die Bewegungen des Roboters relativ langsam erfolgen, so dass seine Bewegungen rein kinematisch beschrieben werden können, kann auch hier die Trägheit vernachlässigt werden. Somit gehen in die Beziehung nur noch die Dämpfung und die Steifheit ein, so dass sich einfachere Regler verwenden lassen. In [Whitney 87] wurde deren Nutzung zur Ausführung verschiedener Aufgaben gezeigt.

*Der Nachgiebigkeitsregler*

Der Nachgiebigkeitsregler wird in der Literatur auch mit den Namen Active Compliance, aktive Nachgiebigkeit sowie Stiffness Control bezeichnet [Mason 81], [Schmucker et al. 96b], [Gorinevsky, Schneider 90]. Die aktive Nachgiebigkeit erzeugt eine proportionale Positionsänderung entsprechend einer wirkenden Kraft

$$\Delta X = k_{FN} \cdot \left( F_{soll} - F \right) \tag{5.120}$$

mit $\Delta X$ - Positionsänderung
$k_{FN}$ - Nachgiebigkeitsfaktor
$F_{soll}$ - Sollkraft
$F$ - Aktuell wirkende Kraft, Istkraft.

Diese Beziehung entspricht dem Federkraftgesetz. Somit wird das Verhalten einer mechanischen Feder implementiert. Größere Werte für den Nachgiebigkeitsfaktor $k_{FN}$ haben eine größere Nachgiebigkeit zur Folge. Der Nachgiebigkeitsfaktor ist mit $k_N = 1/k_S$ indirekt proportional dem Steifigkeitsfaktor. Für den mehrdimensionalen Fall wird der Faktor $k_{FN}$ zu einer Matrix, [Gorinevsky, Schneider 90]. Das Gesetz gilt nicht nur für Translationen, sondern auch für Rotationen. Ein frei beweglicher Körper hat drei translatorische und drei rotatorische Freiheitsgrade. Die allgemeine Form des Nachgiebigkeitsgesetzes lässt sich wie folgt angeben

$$\begin{pmatrix} \Delta\vec{X} \\ \Delta\vec{\omega} \end{pmatrix} = \mathbf{K}_{FN} \cdot \begin{pmatrix} \vec{F}_{soll} - \vec{F} \\ \vec{M}_{soll} - \vec{M} \end{pmatrix} \tag{5.121}$$

mit $\Delta\vec{X}$ - Vektor der Positionsänderungen
$\Delta\vec{\omega}$ - Vektor der Rotationsänderungen
$\mathbf{K}_{FN}$ - Nachgiebigkeitsmatrix
$\vec{F}_{soll}$ - Vektor der Sollkräfte
$\vec{M}_{soll}$ - Vektor der Sollmomente
$\vec{F}$ - Vektor der Istkräfte
$\vec{M}$ - Vektor der Istmomente.

Sollen die auftretenden Kräfte und Momente eine Aktion in die gleiche Richtung bewirken, so wird die Nachgiebigkeitsmatrix nur in ihrer Hauptdiagonalen mit Elementen besetzt, so dass sie jeweils die Nachgiebigkeit für die einzelnen Freiheitsgrade angeben. Ein Beispiel ist

$$\mathbf{K}_{FN} = \begin{pmatrix} k_{FN,x} & 0 & 0 & 0 & 0 & 0 \\ 0 & k_{FN,y} & 0 & 0 & 0 & 0 \\ 0 & 0 & k_{FN,z} & 0 & 0 & 0 \\ 0 & 0 & 0 & k_{FN,\omega x} & 0 & 0 \\ 0 & 0 & 0 & 0 & k_{FN,\omega y} & 0 \\ 0 & 0 & 0 & 0 & 0 & k_{FN,\omega z} \end{pmatrix} = diag\left(k_{FN,x}, k_{FN,y}, k_{FN,z}, k_{FN,\omega x}, k_{FN,\omega y}, k_{FN,\omega z}\right) \quad (5.122)$$

Sind neben der Hauptdiagonalen auch andere Elemente der Matrix mit von Null verschiedenen Elementen besetzt, dann haben wirkende Kräfte und Momente auch Wirkung auf Verschiebungen und Drehungen anderer Richtungen.

*Der Dämpfungsregler bzw. die aktive Anpassung*

Der Dämpfungsregler, auch Damping Control genannt, bzw. die aktive Anpassung, auch als Active Accommodation bezeichnet, erzeugt eine Geschwindigkeit in Abhängigkeit einer wirkenden Kraft. Es gilt

$$\Delta\dot{\vec{X}} = k_{F2} \cdot \left(\vec{F}_{soll} - \vec{F}\right). \quad (5.123)$$

Diese Beziehung beschreibt das Verhalten eines Dämpfers, wobei größere Geschwindigkeiten durch größere wirkende Kraftdifferenzen realisiert werden. Dabei wird eine Bewegung in Richtung der wirkenden Kraft erzeugt. So lange die Kraft wirkt, hält auch die Bewegung an. Es entsteht eine Ausweich- oder Anpassbewegung, die deshalb aktive Anpassung genannt wird. Für größere Werte von $k_{F2}$ wird die Dämpfung geringer. In Bezug auf die Position wirkt dieser Regler integrierend.

Auch dieses Gesetz lässt sich für den mehrdimensionalen Fall formulieren:

$$\begin{pmatrix} \Delta\dot{\vec{X}} \\ \Delta\dot{\vec{\omega}} \end{pmatrix} = \mathbf{K}_{F2} \cdot \begin{pmatrix} \vec{F}_{soll} - \vec{F} \\ \vec{M}_{soll} - \vec{M} \end{pmatrix} \quad (5.124)$$

mit $\Delta\dot{\vec{X}}$ - Lineare Anpassungsgeschwindigkeit
$\Delta\dot{\vec{\omega}}$ - Rotatorische Anpassungsgeschwindigkeit
$\mathbf{K}_{F2}$ - Anpassungsmatrix.

Auch die Anpassungsmatrix ist im Normalfall nur in der Hauptdiagonalen besetzt. Nichthauptdiagonalelemente erzeugen Translationen bzw. Rotationen in andere Richtungen.

*Der Hybride Positions-Kraft-Regler*

In Bezug auf bestimmte Aufgabenstellungen bzw. zur Erzielung bestimmter gewünschter Effekte ist es mitunter erforderlich, selektiv Richtungen (linear, rotatorisch) entweder einer Lage- bzw. Geschwindigkeitsregelung zu unterziehen oder Methoden der Kraftregelung zu nutzen. Diese selektive Steuerung der Methoden zur Bewegungsregelung erfolgt mit der hyb-

riden Positions-Kraft-Regelung [Raibert, Craig 81]. Dabei können die oben aufgeführten Verfahren genutzt werden.

#### 5.2.5.5.2 Implementierung kraftgeführter Bewegungen

Bei autonomen Laufrobotern muss die Steuerung der Laufbewegungen auf dem Roboter selbst mit vertretbarerer Rechenleistung erfolgen. Einerseits sind der verfügbaren Rechenleistung Grenzen gesetzt (Energie, Volumen, Masse), andererseits müssen die Algorithmen zur Bewegungssteuerung leistungsfähig genug sein. Daher werden einfache Algorithmen für kraftgeführte Bewegungen implementiert. Gleichzeitig werden damit bisher nicht genannte Probleme, wie das überbestimmte mechanische System und die unvollkommene Positionierung der Füße gelöst. Durch die Beine werden kinematische Ketten geschlossen. Dies bedeutet, wenn sich ein Gelenk bewegt, müssen auch andere Gelenke synchron mitbewegt werden. In der Praxis treten dabei jedoch immer Fehler auf. Bei mechanisch steifen Strukturen entstehen bereits durch kleine Abweichungen große Kräfte, die in die mechanische Struktur hinein wirken und Energie binden. In der Literatur wurden bereits Methoden, mit denen sich solche auftretenden Querkräfte minimieren lassen für kleinere Roboter mit externer Steuerung und für große Roboter mit mehreren 100 kg bis 2 t Masse, die ausreichend Platz für eine größere Steuereinheit bieten, beschrieben [Lehtinen 94], [Gao, Song 90], [Gorinevsky, Schneider 90].

Zur Implementierung in den betrachteten Laufroboter wurde die Methode der aktiven Nachgiebigkeit ausgewählt. Sie hat den Vorteil, dass sie einfach ist und mit einer rein kinematisch arbeitenden Steuerung funktioniert.

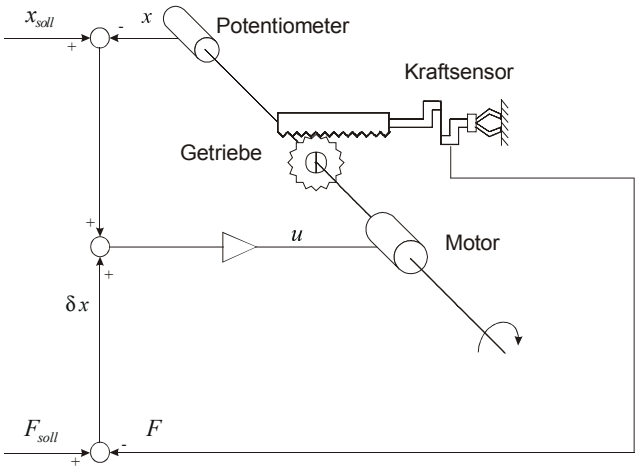



Abbildung 5.41: Prinzip der aktiven Nachgiebigkeit für den eindimensionalen Fall [Schmucker et al. 94]

Abbildung 5.41 zeigt den Zusammenhang der aktiven Nachgiebigkeit als Funktionsschema für den eindimensionalen Fall. Dabei wird die Information über die gemessene Kraftwirkung zur Modifikation des Sollwertes genutzt. Die Richtung einer Bewegung (Aktion) entsteht bei einem Bein erst durch das Zusammenwirken aller drei aktiven Freiheitsgrade. Die Reaktionskraft kann eine beliebige räumliche Richtung besitzen. Sie wird mit dem Kraftsensor im Unterschenkel erfasst, dessen räumliche Orientierung abhängig von der Position des Beines ist. Um die Größen miteinander verknüpfen zu können, müssen sie auf einer gemeinsamen Koor-

dinatenbasis dargestellt werden. Hierzu eignet sich das Beinkoordinatensystem. Die Sollposition liegt bereits im Beinkoordinatensystem vor, die wirkenden Kräfte und die aktuelle Beinposition lassen sich transformieren (siehe Abschnitte 5.2.5.1 und 5.2.5.3). Die vorgegebene Sollposition des Beinendes wird dann mit einer kraftabhängigen Positionsänderung, d.h. einer Nachgiebigkeit überlagert, so dass der nun modifizierte Sollwert durch die Gelenkregler realisiert wird. Die neue Sollposition wird dazu wie im Abschnitt 5.2.5.2 beschrieben in den Gelenkraum transformiert. Die Steuergesetze lauten

$$\vec{X}^{*}_{soll} = \vec{X}_{soll} + \mathbf{K}_{FN} \cdot \left(\vec{F}_{soll} - \vec{F}\right) \tag{5.125}$$

$$\vec{Y} = \mathbf{K}_{R} \cdot \left(\vec{q}_{soll} - \vec{q}\right) \tag{5.126}$$

$$\vec{q}_{soll} = {}^{q}_{B}\mathrm{T}\left\{\vec{X}^{*}_{soll}\right\} \tag{5.127}$$

$$\vec{F} = {}^{B}_{F}\mathrm{T}\left\{{}^{F}\vec{F}\right\} \tag{5.128}$$

mit
- $\vec{X}^{*}_{soll}$ - Vektor der durch Nachgiebigkeit modifizierten kartesischen Sollposition
- $\vec{X}_{soll}$ - Vektor der vom Bewegungsalgorithmus vorgegebenen kartesischen Sollposition
- $\vec{q}$ , $\vec{q}_{soll}$ - Ist- und Sollposition für die einzelnen Gelenke
- $\mathbf{K}_{R}$ - Regelparameter für den Positionsregelkreis
- $\vec{Y}$ - Stellgrößen für die einzelnen Gelenkantriebe
- ${}^{q}_{B}\mathrm{T}$ - Transformation vom Beinkoordinatensystem in den Gelenkraum
- ${}^{B}_{F}\mathrm{T}$ - Transformation vom Fußkoordinatensystem in das Beinkoordinatensystem.

Gleichung (5.125) realisiert die aktive Nachgiebigkeit im kartesischen Beinkoordinatensystem, Gleichung (5.126) die unterlagerte Positionsregelung im Gelenkraum, Gleichung (5.127) die Transformation der kartesischen Sollkoordinaten vom Beinkoordinatensystem in den Gelenkraum und Gleichung (5.128) die Transformation der gemessenen Kräfte vom Fußkoordinatensystem in das Beinkoordinatensystem.

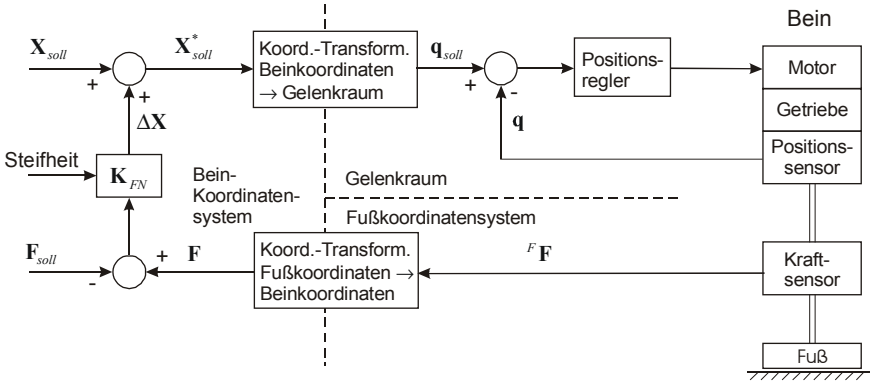



Abbildung 5.42: Prinzip der Implementierung der aktiven Nachgiebigkeit für den dreidimensionalen Fall

Das Wirkprinzip der aktiven Nachgiebigkeit für ein Bein ist in Abbildung 5.42 dargestellt. Es wird deutlich, dass die Positionsregelung als innerer Wirkkreis der Nachgiebigkeit unterlagert ist. Während die Positionsregelung auf Gelenkebene realisiert wird, nutzt die Nachgiebigkeitsregelung ein abstraktes Beinmodell und beschreibt die Bewegung in kartesischen Bein-

koordinaten. Der Übergang zwischen den Abstraktionsebenen wird durch Koordinatentransformationen vollzogen.

### 5.2.6 Kraftverteilungen

Beim Laufen eines Roboters stellt sich die Frage nach den Interaktionen mit dem Untergrund, welche durch Aktions- und Reaktionskräfte bestimmt werden. Daher ist es wichtig, diese Kräfte zu kennen und gegebenenfalls gezielt zu beeinflussen. Bei der Betrachtung der Interaktionskräfte soll im folgenden eine Fokussierung auf die vertikalen Kräfte erfolgen.

Ein Roboter wirkt während des Laufvorganges mit seiner Gewichtskraft auf den Untergrund. Da er mit mehreren Beinen auf dem Boden steht, verteilen sich die Kräfte auf die Stützfüße. Zur Beschreibung der Kräfte können die statischen Gleichgewichtsbedingungen herangezogen werden [Gross et al. 86]. Momente entstehen, wenn Kräftepaare wirken und somit eine Drehwirkung verursachen. Dabei ist es wichtig, wo und mit welchem Hebel ein Moment angreift. Allgemein gilt

$$\vec{M} = \vec{r} \times \vec{F} \,. \tag{5.129}$$

Wirken mehrere Momente und Kräfte auf den Körper, so entstehen resultierende Kräfte und Momente

$$\vec{M} = \sum \vec{M}_i \tag{5.130}$$

$$\vec{F} = \sum \vec{F}_i \quad . \tag{5.131}$$

Führen alle wirkenden Kräfte und Momente zu keiner Bewegung des Körpers, so befinden sie sich im Gleichgewicht, das heißt, dass sich alle Kräfte und Momente gegenseitig aufheben. Die statischen Gleichgewichtsbedingungen lauten somit

$$\sum \vec{M}_i = 0 \text{ mit } \sum M_{i,X_1} = 0 \,,\ \sum M_{i,X_2} = 0 \text{ und } \sum M_{i,X_3} = 0 \tag{5.132}$$

$$\sum \vec{F}_i = 0 \text{ mit } \sum F_{i,X_1} = 0 \,,\ \sum F_{i,X_2} = 0 \text{ und } \sum F_{i,X_3} = 0 \,. \tag{5.133}$$

Die Anzahl der skalaren Gleichgewichtsbedingungen entspricht den sechs möglichen Freiheitsgraden. Im betrachteten Fall wirken alle Kräfte, d. h. Gewichtskraft und Stützkräfte, in die gleiche Richtung. Da $F_{i,X_1} = 0$ und $F_{i,X_2} = 0$ reduziert sich die Anzahl der Gleichungen. Es gilt

$$\sum F_{i,X_3} = 0 \,,\ \sum M_{i,X_1} = 0 \text{ und } \sum M_{i,X_2} = 0 \,. \tag{5.134}$$

Diese Gleichgewichtsbedingungen bilden ein Gleichungssystem, das sich in Matrixform formulieren lässt

$$\begin{pmatrix} 1 \\ \vec{X}_1^{(i)} \\ \vec{X}_2^{(i)} \end{pmatrix} \cdot \vec{F}^{(i)} = \begin{pmatrix} 1 \\ X_1^{(0)} \\ X_2^{(0)} \end{pmatrix} \cdot F_g \ . \tag{5.135}$$

Dabei sind $\vec{X}_1^{(i)}$ und $\vec{X}_2^{(i)}$ die Koordinaten der am jeweiligen Bein angreifenden Kraft $\vec{F}^{(i)}$. $X_1^{(0)}$ und $X_2^{(0)}$ sind die Koordinaten des Massenschwerpunktes, von dem aus die Gewichtskraft $F_g$ wirkt. Werden drei Beine als Stützbeine genutzt, so ergeben sich drei Gleichungen mit drei Unbekannten. Die Verteilung der Kräfte $\vec{F}^{(i)}$ kann somit eindeutig durch die Lösung des Gleichungssystems bestimmt werden. Das mechanische System ist in diesem Fall bestimmt.

Beim Dreifußgang befinden sich je drei Füße in der Stützphase. Da der Stützfaktor β=0,5 ist, gibt es keine Überlappungen, so dass idealerweise stets drei Beine Bodenkontakt haben. Um den Wechsel der Stützfüße zu berücksichtigen, wird eine Selektionsmatrix **S** eingeführt, bei der nur die Hauptdiagonalelemente besetzt sind. Soll ein Element eines Vektors ausgewählt werden, so wird das entsprechende Hauptdiagonalenelement mit 1 besetzt, sonst mit 0. Entsprechend der vorhandenen Konfiguration von Stützfüßen werden die Hauptdiagonalenelemente mit 1 besetzt. Gleichung (5.135) lässt sich wie folgt ergänzen

$$\begin{pmatrix} 1 \\ \vec{X}_1^{(i)} \\ \vec{X}_2^{(i)} \end{pmatrix} \cdot \mathbf{S} \cdot \vec{F}^{(i)} = \begin{pmatrix} 1 \\ X_1^{(0)} \\ X_2^{(0)} \end{pmatrix} \cdot F_g \ . \tag{5.136}$$

Um die Stützkräfte zu ermitteln, müssen die benutzten Stützfüße, die Position der Stützfüße, die Gewichtskraft und die Koordinaten des Massenschwerpunktes bekannt sein. Der Masseschwerpunkt lässt sich bestimmen, wenn die Verteilung der Masse über den gesamten Roboter bekannt ist. Es gilt

$$\vec{X}^{(0)} = \frac{\int \vec{X}\, m\left(\vec{X}\right) d\vec{X}}{\int m\left(\vec{X}\right) d\vec{X}} \ . \tag{5.137}$$

Dabei ist $\vec{X}^{(0)}$ der Ortsvektor des Massenschwerpunktes und $\vec{X}$ der Vektor des Massenpunktes $m$. Da es unmöglich ist, die genaue Massenverteilung zu kennen, wird ein Modell genutzt, das jeweils starre Körper zu einem Massepunkt mit entsprechenden Koordinaten zusammenfasst. Die Bestimmung des Massenschwerpunktes reduziert sich dann auf wenige zu betrachtende Massenpunkte

$$\vec{X}^{(0)} = \frac{\sum \vec{X}_i\, m_i}{\sum m_i} \ . \tag{5.138}$$

Dazu werden der Roboterkörper und die Beinsegmente als Massepunkte zusammengefasst, da sie sich als starre Körper auffassen lassen. Die Beine werden wieder in die Segmente *A*, *B*, *C* und *D* zerlegt (siehe Abbildung 5.22 in Abschnitt 5.2.5.1). Das Segment *A* wird dem Körper zugeschlagen, da es mit ihm fest verbunden ist. Für die anderen drei Segmente werden die Massepunkte $m_B$, $m_C$ und $m_D$ bestimmt. Ihre Koordinaten werden zum vorhergehenden Gelenk mit $B_m$, $C_m$ und $D_m$ bestimmt. Um zunächst die Koordinaten des Beinschwerpunktes zu

bestimmen, müssen diese analog zur direkten kinematischen Transformation bestimmt werden. Da hier nur die Ebene senkrecht zum Gravitationsvektor betrachtet und der Roboterkörper als parallel zu dieser Ebene angenommen wird, müssen nur zwei Koordinaten bestimmt werden. Ist der Gravitationsvektor jedoch beliebig gerichtet und somit die Annahme über den Roboterkörper nicht zutreffend, müssen alle drei Koordinaten beachtet werden. Die Transformationsgleichungen für die Koordinaten der Massepunkte im ebenen Beinkoordinatensystem lauten

$$ {}^{B}\vec{X}^{(m_B)} = \begin{pmatrix} {}^{B}X_1^{(m_B)} \\ {}^{B}X_2^{(m_B)} \end{pmatrix} = \begin{pmatrix} B_m \cdot \sin(\alpha) \\ A_{X_2} + B_m \cdot \cos(\alpha) \end{pmatrix} \tag{5.139} $$

$$ {}^{B}\vec{X}^{(m_C)} = \begin{pmatrix} {}^{B}X_1^{(m_C)} \\ {}^{B}X_2^{(m_C)} \end{pmatrix} = \begin{pmatrix} (B + C_m \cdot \cos(\beta)) \cdot \sin(\alpha) \\ A_{X_2} + (B + C_m \cdot \cos(\beta)) \cdot \cos(\alpha) \end{pmatrix} \tag{5.140} $$

$$ {}^{B}\vec{X}^{(m_D)} = \begin{pmatrix} {}^{B}X_1^{(m_D)} \\ {}^{B}X_2^{(m_D)} \end{pmatrix} = \begin{pmatrix} (B + C \cdot \cos(\beta) + D_m \cdot \cos(\beta + \gamma)) \cdot \sin(\alpha) \\ A_{X_2} + (B + C \cdot \cos(\beta) + D_m \cdot \cos(\beta + \gamma)) \cdot \cos(\alpha) \end{pmatrix} . \tag{5.141} $$

Somit sind die Massepunkte für die einzelnen Beinsegmente im Beinkoordinatensystem bekannt. Zur Berechnung des Massenschwerpunktes müssen diese in das Roboterkoordinatensystem transformiert werden.

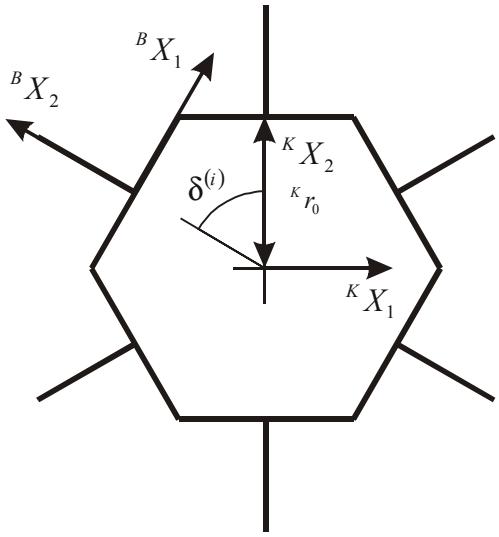



Abbildung 5.43: Ebene Koordinatensysteme zur Massenschwerpunktberechnung

Die verwendeten ebenen Koordinatensysteme sind in Abbildung 5.43 dargestellt. Für die Transformation müssen die Koordinaten um den Abstand $r_B$ vom Koordinatenursprung des Roboterkoordinatensystems zum Beinbefestigungspunkt verschoben und um den Seitenwinkel $\delta^{(i)}$ für das jeweilige Bein gedreht werden

$$ {}^{R}\vec{X}^{(m)} = {}^{R}_{B}\mathrm{T} \cdot {}^{B}\vec{X}^{(m)} = \mathrm{rot}\left(\delta^{(i)}\right) \cdot \mathrm{trans}\left({}^{K}r_0\right) {}^{B}\vec{X}^{(m)} \tag{5.142} $$

$$ {}^{R}\vec{X}^{(m)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & {}^{K}r_0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos(\delta) & -\sin(\delta) & 0 \\ \sin(\delta) & \cos(\delta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} {}^{B}X_1^{(m)} \\ {}^{B}X_2^{(m)} \\ 1 \end{pmatrix} \tag{5.143} $$

$$ {}^{R}\vec{X}^{(m)} = \begin{pmatrix} \cos(\delta) & -\sin(\delta) & 0 \\ \sin(\delta) & \cos(\delta) & {}^{0}r_0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} {}^{B}X_1^{(m)} \\ {}^{B}X_2^{(m)} \\ 1 \end{pmatrix}. \tag{5.144} $$

Damit ergeben sich alle Massepunktkoordinaten für die Beine und den Roboterkörper im Roboterkoordinatensystem. Der Massenschwerpunkt für den gesamten Roboter lässt sich mit

$$ X_1^{(0)} = \frac{m_B \sum X_1^{(i,m_B)} + m_C \sum X_1^{(i,m_C)} + m_D \sum X_1^{(i,m_D)} + X_1^{(K)} \cdot m_K}{6 \cdot (m_B + m_C + m_D) + m_K} \tag{5.145} $$

$$ X_2^{(0)} = \frac{m_B \sum X_2^{(i,m_B)} + m_C \sum X_2^{(i,m_C)} + m_D \sum X_2^{(i,m_D)} + X_2^{(K)} \cdot m_K}{6 \cdot (m_B + m_C + m_D) + m_K} \tag{5.146} $$

bestimmen.

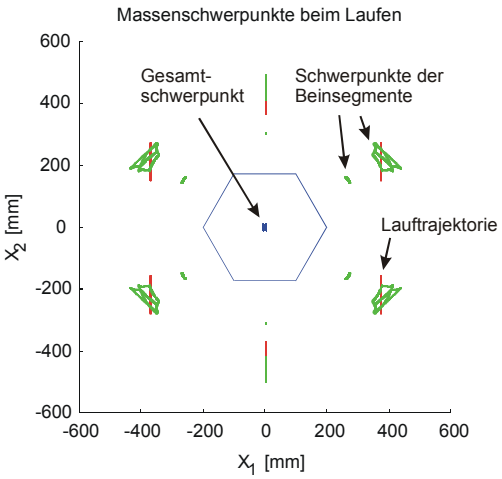



Abbildung 5.44: Massenschwerpunkte beim Laufen

Abbildung 5.44 zeigt die Positionen der Massepunkte und des Massenschwerpunktes für einen Laufzyklus. Durch Lösen von Gleichung (5.136) lässt sich für jeden Zeitpunkt die Kraftverteilung für die drei stützenden Füße mit

$$ \vec{F}^{(i)} = \begin{pmatrix} 1 \\ \vec{X}_1^{(i)} \\ \vec{X}_2^{(i)} \end{pmatrix}^{-1} \cdot \begin{pmatrix} 1 \\ X_1^{(0)} \\ X_2^{(0)} \end{pmatrix} \cdot F_g \tag{5.147} $$

für i=(1, 3, 5) oder i=(2, 4, 6) bestimmen.

Bei dem benutzten Laufmuster für den Dreifußgang ergibt sich die in Abbildung 5.45 gezeigte Kraftverteilung. Es ist zu sehen, dass sich die Stützkraft für ein Bein während der Stützphase stetig vergrößert oder verringert. Dieser Effekt entsteht dadurch, dass sich der Abstand zwischen stützendem Bein und dem Massenschwerpunkt im betrachteten Fall stetig verringert bzw. vergrößert. Dieser Effekt tritt dann nicht auf, wenn sich der Abstand zum Massenschwerpunkt während einer Stützphase verringert und wieder vergrößert. Dies ist der Fall, wenn Laufrichtung, Koordinatenursprung des Roboterkoordinatensystems und Beinbefestigungspunkt einen annähernd rechten Winkel bilden.

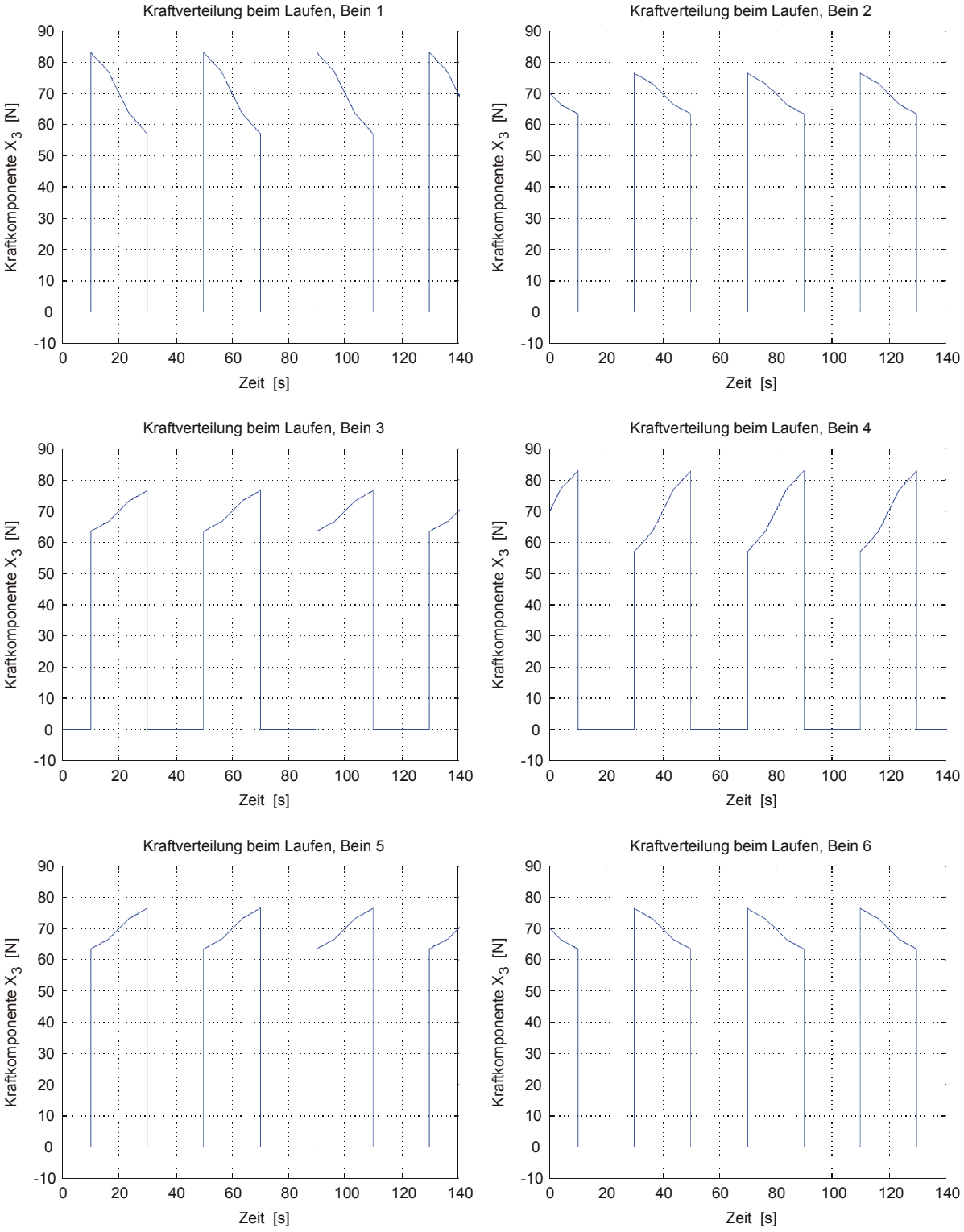



Abbildung 5.45: Simulierte Kraftverteilung beim Dreifußgang

Damit lässt sich die über die Füße einstellende Kraftverteilung bestimmen. Die nun bekannten zu erwartenden Stützkräfte können genutzt werden, um eine vertikale „Vorspannung“ für die aktive Nachgiebigkeit einzustellen, indem die Sollkräfte mit den zu erwartenden Stützkräften gleichgesetzt werden

$$\vec{F}_{soll}^{(i)} = \begin{pmatrix} 0 \\ 0 \\ F^{(i)} \end{pmatrix} . \tag{5.148}$$

Mit Hilfe der gemessenen Stützkräfte kann umgekehrt der Ort des Massenschwerpunktes mit Gleichung (5.135) bestimmt und somit die Stabilität des Roboters mit Hilfe von Sensorsignalen nachgeprüft werden.

Für Fälle, in denen mehr als drei stützende Füße genutzt werden, kann die Kraftverteilung nicht direkt auf diesem Weg bestimmt werden, da das statische System unbestimmt ist. Es sind dann zusätzliche Bestimmungsgleichungen notwendig, um eine eindeutige Lösung für alle Kräfte zu erhalten. Diese Zusatzbedingungen enthalten dann beispielsweise Kriterien für eine möglichst gleichmäßige Kraftverteilung. Das wird erreicht, indem die Stützkräfte minimiert werden [Gorinevsky, Schneider 90], [Lehtinen 94]

$$\left|\vec{F}\right|^2 = \sum_i \left|\vec{F}^{(i)}\right|^2 \rightarrow \min \ , \tag{5.149}$$

was den Effekt der Minimierung des Energieverbrauchs hat.

Diese Bedingung entspricht gleichzeitig der besten Näherungslösung für das Gleichungssystem (5.135) bzw. (5.136). Dieses Kriterium wird durch die Nutzung der pseudoinversen Matrix $A^+$ zur Lösung des Gleichungssystems erfüllt [Gantmacher 86]

$$\vec{F}^{(i)} = \left(\mathbf{X}^{(i)}\right)^{+} \cdot \vec{X}^{(0)} \cdot F_g \tag{5.150}$$

$$\vec{F}^{(i)} = \left(\mathbf{X}^{(i)}\right)^{\mathrm{T}} \cdot \left(\left(\mathbf{X}^{(i)}\right) \cdot \left(\mathbf{X}^{(i)}\right)^{\mathrm{T}}\right)^{-1} \cdot \vec{X}^{(0)} \cdot F_g \tag{5.151}$$

mit $\mathbf{X}^{(i)} = \begin{pmatrix} 1 & \vec{X}_1^{(i)} & \vec{X}_2^{(i)} \end{pmatrix}^{\mathrm{T}}$ für die Koordinaten der stützenden Füße

$\vec{X}^{(0)} = \begin{pmatrix} 1 & X_1^{(0)} & X_2^{(0)} \end{pmatrix}^{\mathrm{T}}$ für die Koordinaten des Masseschwerpunktes.

Mit den beschriebenen Methoden ist es möglich, eine geeignete Kraftverteilung zwischen den Füßen zu bestimmen und diese Stützkräfte für die Geländeanpassung und die aktive Nachgiebigkeit vorzugeben sowie die Position des Massenschwerpunktes mit Hilfe von Sensordaten zu bestimmen, um die Stabilität des Roboters zu prüfen.

## 5.3 Erprobung des Steuerungssystems

In diesem Kapitel soll die Funktionsfähigkeit der im vorangegangenen Abschnitt entwickelten Steuerungsalgorithmen gezeigt werden. Dies erfolgt hauptsächlich mit Hilfe von Experimenten mit dem Laufroboter Katharina.

### 5.3.1 Erprobung des Laufens

Im Abschnitt 5.2.4 wurde gezeigt, wie die zum Laufen notwendigen Bewegungen erzeugt werden. Dabei wurde von einem Musterlaufzyklus ausgegangen, der die prinzipiell notwendigen Relativbeinbewegungen bereitstellt. Zur Sicherung der statischen Stabilität muss das Laufmuster so gewählt werden, dass zu jedem Zeitpunkt mindestens drei Füße gleichzeitig auf dem Boden stehen und somit ein ausreichend großes Stützpolygon aufspannen. Dazu wurde der Dreifußgang ausgewählt, so dass abwechselnd je drei Beine ein Stützpolygon aufspannen. Abbildung 5.45a und b zeigen die möglichen Stützpolygone beim Dreifußgang, während Abbildung 5.45c ein Stützpolygon beim Stehen mit allen 6 Beinen zeigt.

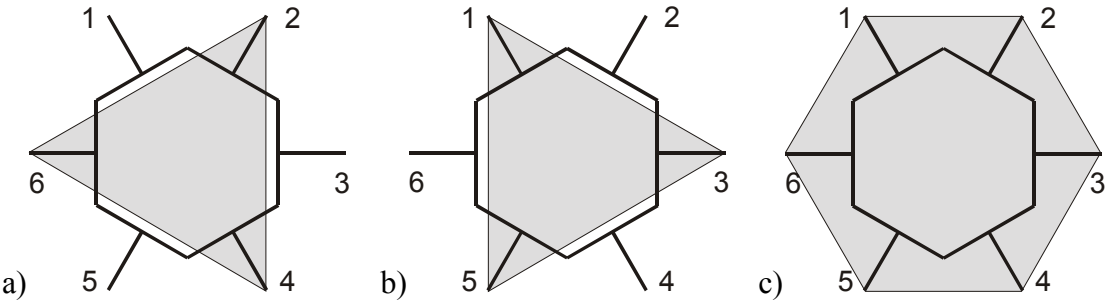



Abbildung 5.46: Beispiel für Stützpolygone beim Dreifußgang

Es ist zu sehen, dass durch die Wahl des Dreifußganges als Laufmuster ein großes Stützpolygon aufgespannt wird. Je drei Beine, die zum Stützen dienen, werden zu einer Beingruppe zusammengefasst. Für diese gilt die gleiche Mustertrajektorie. Die Mustertrajektorien zwischen den Beingruppen sind zeitlich derart verschoben, dass sich jeweils mindestens drei Beine auf dem Boden befinden. Aus der Mustertrajektorie werden die für die einzelnen Beine relevanten Trajektorien im kartesischen Raum abgeleitet. Die Beincontroller selbst leiten aus diesen kartesischen Koordinaten wiederum die einzelnen erforderlichen Gelenkbewegungen ab.

#### 5.3.1.1 Lauftrajektorie

Zur Überprüfung der korrekten Erzeugung der Lauftrajektorie aus der Mustertrajektorie wird diese im kartesischen Raum des Beinkoordinatensystems betrachtet. Dazu werden die während eines Laufzyklus gemessenen Gelenkwinkel mit Hilfe der direkten Kinematik in den

kartesischen Raum überführt, so dass sich die Trajektorie im Bein- oder Roboterkoordinatensystem darstellen lässt. Die gemessene Trajektorie muss der in Abbildung 5.27 in Abschnitt 5.2.4.1.2 entsprechen.

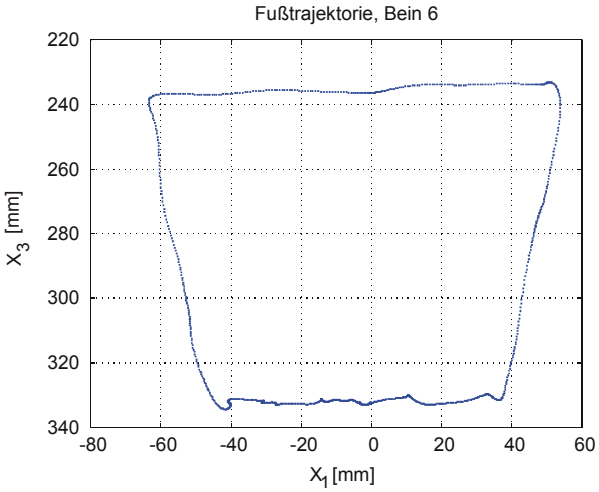



Abbildung 5.47: Gemessene Fußtrajektorie

Abbildung 5.47 zeigt die gemessene Fußtrajektorie für Bein 6 im Beinkoordinatensystem. Es sind alle Phasen entsprechend der Abbildung 5.27 gut zu erkennen. Das Verschleifen der Trajektorie ist mit einer ungenauen Positionierung durch das Servoregelsystem zu erklären. Weitere Ungenauigkeiten in der Darstellung entstehen durch das Quantisierungsrauschen in der Analog-Digital-Wandlung sowie durch weitere Quantisierungseffekte, die durch die Algorithmen zur Koordinatentransformation und der roboterinternen Darstellung der Beinkoordinaten in Integerformat bedingt sind. Die zu beobachtenden Streuungen liegen im Bereich von ±1,2 mm. Diese Abweichungen sind angesichts des konstruktiv vorhandenen Spiels in den Gelenken vertretbar. Dieses Spiel stellt eine besondere Schwierigkeit für den Regler dar. Eine Verbesserung der realisierten Trajektorie wird nur durch eine verbesserte Mechanik, ein komplexeres Regelungsprinzip (Position und Geschwindigkeit) und durch eine Behandlung der Daten mit erhöhter Genauigkeit möglich. Das bedeutet jedoch eine deutliche Erhöhung des mechanischen, elektrischen und rechentechnischen Aufwandes.

### 5.3.1.2 Einfacher Dreifußgang

Zur Prüfung der korrekten Ausführung der Laufbewegungen für den Dreifußgang ist es sinnvoll, die Beinbewegungen im kartesischen Raum zu betrachten, da dadurch eine gleichartige Darstellungsbasis für die Bewegungen existiert. Gleichzeitig entspricht diese Darstellungsform auch den für einen Beobachter wahrnehmbaren räumlichen Eindruck, so dass die Plausibilität der erzeugten Bewegungen einfach geprüft werden kann.

Für den Dreifußgang werden zwei Beingruppen benötigt. Die Beine einer Beingruppe werden dabei jeweils wechselnd angehoben. Wird der Stützfaktor $\beta$=0,5 gewählt, so ist die Überlappung zwischen den Stützphasen der beiden Beingruppen minimal, das heißt, dass sich die Beingruppen beim Stützen abwechseln. Abbildung 5.48 zeigt das zu erwartende Laufmuster für die zwei Beingruppen des Dreifußganges. Dargestellt ist der zeitliche Verlauf der vertika-

len Positionskomponente. Der Untergrund dient dabei als Bezugswert und ist im Diagramm identisch mit der Nulllinie. Wechselt ein Bein in die Transferphase, so wird es bis zur Höhe *h* angehoben, wird in dieser Höhe über den neuen Aufsetzpunkt bewegt und dann abgesenkt. Stehen die soeben neu platzierten Beine wieder auf dem Boden, so werden die Beine der anderen Beingruppe angehoben. Dieser Zyklus wiederholt sich abwechselnd für jede Beingruppe.

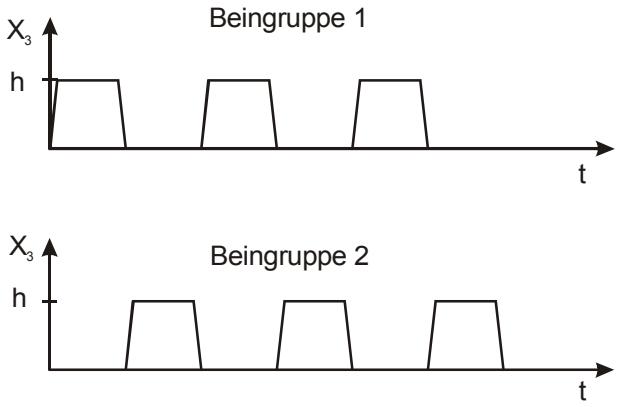



Abbildung 5.48: Zu erwartendes Laufmuster für jede Beingruppe

In einem Laufexperiment soll dieses Verhalten gezeigt werden. Dazu werden die mit Hilfe der Gelenkwinkelpotentiometer gemessenen Gelenkpositionen mittels der direkten Kinematik in den kartesischen Raum überführt, so dass sich die Fußposition in kartesischen Koordinaten darstellen lässt. Im Laufexperiment wird dem Roboter ein Laufkommando gegeben, so dass er sich aus der Ruheposition in Bewegung setzt. Zunächst wird für eine Beingruppe nur ein verkürzter Schritt erzeugt, da sich alle Beine zu Beginn in der Neutralposition befinden. Danach müssen alle folgenden Schritte von gleicher Dauer sein, es stellt sich also ein gleichmäßiges Laufmuster ein. Die Laufbewegungen während des Experimentes werden in beschriebener Weise aufgezeichnet.

Abbildung 5.49 zeigt den zeitlichen Verlauf der vertikalen Fußpositionen für alle Beine, wie sie während des Laufexperimentes aufgezeichnet worden sind. Die Beine 1, 3 und 5 gehören zu einer Beingruppe, während die Beine 2, 4 und 6 zur zweiten Beingruppe gehören. Es ist zu sehen, dass sich für beide Beingruppen jeweils gleichartige Bewegungsmuster ergeben. Nach einem anfänglich verkürzten Schritt für Beine 1, 3 und 5 stellt sich ein stabiles Laufmuster ein, das für beide Beingruppen alternierend ist. Der anfänglich verkürzte Schritt entsteht dadurch, dass sich die Beine des Roboters zu Beginn in einer neutralen Position befinden. Aus dieser Position heraus ist nur ein Schritt mit halber Länge möglich. Aus den Diagrammen ist noch nicht eindeutig ersichtlich, ob zu jedem Zeitpunkt tatsächlich mindestens drei Stützbeine zur Verfügung stehen. Um dies zu prüfen, müssen die Zeiten gegenübergestellt werden, in denen sich die Beine in der Stützphase befinden. Dazu werden während des gleichen Experimentes die Zeiten festgehalten, in denen sich ein Bein in der Stützphase befindet.

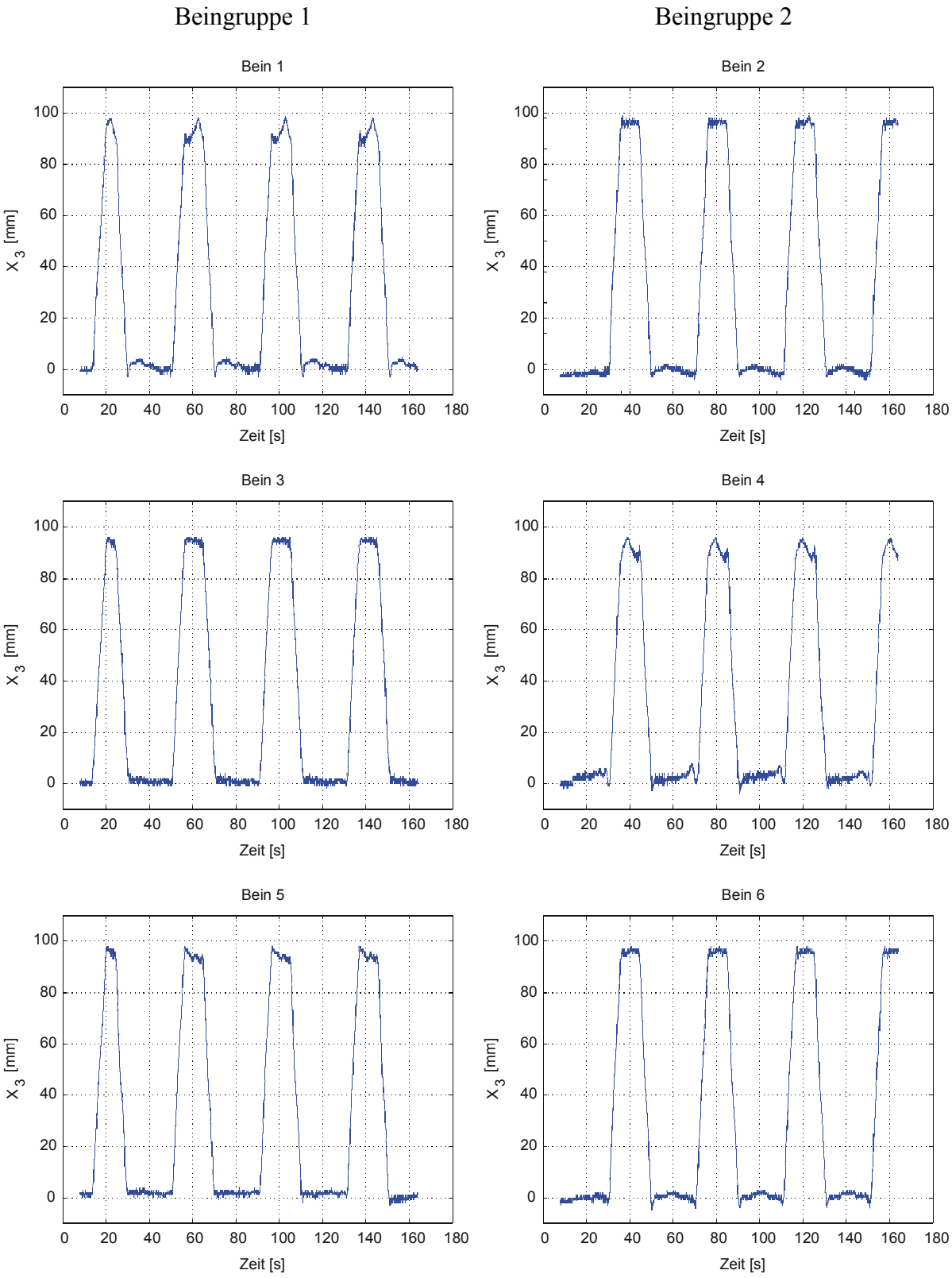



Abbildung 5.49: Zeitlicher Verlauf der vertikalen Fußpositionen beim Dreifußgang Experiment mit dem Dreifußgang

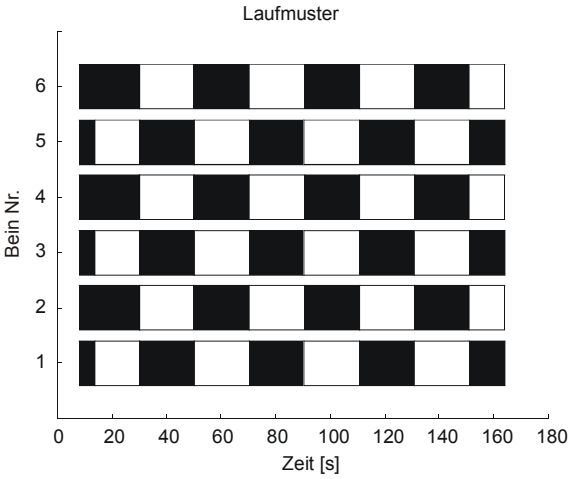



Abbildung 5.50: Laufmuster während des Experimentes

Abbildung 5.50 zeigt das entstandene Laufmuster während des Experimentes. Die schwarzen Bereiche kennzeichnen, dass das betreffende Bein als Stützbein genutzt wird, die hellen Bereiche kennzeichnen ein angehobenes Bein. Im Diagramm ist deutlich zu erkennen, dass zu jedem Zeitpunkt mindestens drei Beine zum Stützen genutzt werden. Zu Beginn des dargestellten Zeitraumes dienen alle Beine zum Stützen, da sich alle Beine in der Neutralposition befinden.

#### 5.3.1.3 Omnidirektionales Laufen

Bisher wurde das korrekte Erzeugen der Lauftrajektorie und des Laufmusters überprüft. Die Abbildungsvorschriften erlauben jedoch auch, die Lauftrajektorien räumlich so zu orientieren, dass sich der Roboter in eine gewünschte Richtung fortbewegt. Zur Prüfung der Richtungssteuerung wird der Parameter für die Laufrichtung aus Gleichung (5.29) in gewünschter Weise vorgegeben. Die Laufrichtung kann nicht zu jedem beliebigen Zeitpunkt gewechselt werden. Der Wechsel der Laufrichtung kann nur dann erfolgen, wenn bestimmte Konfigurationen der Beine vorhanden sind. Das ist dann der Fall, wenn sich die Beine in bestimmten neutralen Punkten befinden, die für Trajektoriern verschiedener Laufrichtung gemeinsam sind. Diese Punkte werden Umschaltpunkte genannt. Die Veränderung der Laufrichtung in diesen Umschaltpunkten wird durch einen Zustandsautomaten getriggert, der in Abschnitt 5.2.4.5 beschrieben ist. Im Experiment soll zunächst eine Laufrichtung vorgegeben werden, die dann nach mehreren Schritten beliebig geändert wird. Die gemessenen Fußtrajektorien und der jeweils vorgegebene Vektor der Laufgeschwindigkeit werden aufgezeichnet.

Nach Gleichung (5.44) lässt sich der Vektor $\vec{R}_0^{(i)}$ zu einem Fuß im Weltkoordinatensystem durch die Addition des Vektors $\vec{R}_0$ zum Zentralkoordinatensystem und des Vektors $\vec{R}^{(i)}$ vom Ursprung des Zentralkoordinatensystems zum Fuß bestimmen. $\vec{R}^{(i)}$ wiederum lässt sich durch Addition des Vektors $\vec{r}_0^{(i)}$ zum Beinkoordinatensystem und dem Vektor $\vec{r}^{(i)}$ im Beinkoordinatensystem zum Fuß bestimmen. Zusammengesetzt ergibt sich

$$\vec{R}_0^{(i)} = \vec{R}_0 + \vec{r}_0^{(i)} + \vec{r}^{(i)} \ . \tag{5.152}$$

$\vec{r}_0^{(i)}$ ist eine feste geometrische Größe, da der Körper bei diesem Experiment nicht bewegt wird[1]. Der zeitliche Verlauf von $\vec{r}^{(i)}$ beschreibt die Fußtrajektorie und wird während des Ausführens der Laufbewegung aufgezeichnet. Der noch fehlende Vektor $\vec{R}_0$ kennzeichnet die aktuelle Position des Roboters, die sich entsprechend der Fortbewegung ändert

$$\vec{R}_0(t) = \vec{R}_0(t_0) + \int_{t_0}^{t} \left( \frac{\vec{R}_0(t)}{dt} \right) dt \quad . \tag{5.153}$$

Durch Differenzieren von Gleichung (5.5) nach der Zeit erhält man

$$\frac{\vec{R}_0^{(i)}}{dt} = \frac{\vec{R}_0}{dt} + \frac{\vec{R}^{(i)}}{dt} \quad . \tag{5.154}$$

Da die Fußpunkte der Stützbeine während der Stützphase ortsfest auf dem Untergrund sind, ist ihre Geschwindigkeit gegenüber dem Weltkoordinatensystem Null. Somit ergibt sich

$$\frac{\vec{R}_0}{dt} = -\frac{\vec{R}^{(i)}}{dt} \quad . \tag{5.155}$$

Voraussetzung dafür ist, dass sich der Körper geradlinig bewegt, d.h. er sich nicht dreht. Das ist laut Voraussetzung für das richtungsunabhängige Design gegeben und bestätigt Gleichung (5.55). Somit wird durch die Vorgabe der Stützgeschwindigkeit der Füße und der Laufrichtung auch unmittelbar die resultierende Laufrichtung bestimmt, die alternativ auch durch die Geschwindigkeitskomponenten in vektorieller Form gegeben werden kann. Unter der Annahme, dass die Stützgeschwindigkeiten nahezu ideal realisiert werden, kann die Fortbewegungsgeschwindigkeit gleich der vorgegebenen Geschwindigkeit gesetzt werden. Der Roboter gibt hierzu die verwendeten Geschwindigkeitskomponenten zurück. Somit kann Gleichung (5.153) mit Hilfe der vorgegebenen Geschwindigkeit ausgedrückt werden

$$\vec{R}_0(t) = \vec{R}_0(t_0) + \int_{t_0}^{t} (\vec{v}(t)) dt \quad . \tag{5.156}$$

Geringfügige Fehler bei der Realisierung der Bewegungen können hier vernachlässigt werden, da an dieser Stelle nur die Korrektheit der Laufbewegungen zur Realisierung einer gewünschten Laufrichtung geprüft werden sollen.

Da Geschwindigkeiten und Beinpositionen zu diskreten Zeitpunkten gemessen werden, kann der Aufenthaltsort des Roboters zu jedem dieser Zeitpunkte bestimmt werden. Die Messungen erfolgen mit der Periodendauer *T*, die auch maßgeblich für die Steuerung des Roboters in der Hauptsteuerungsebene ist. Die Position des Roboters zu den Messzeitpunkten kann mit

$$\vec{R}_0(t_k) = \vec{R}_0(t_0) + T \cdot \sum_{i=0}^{k} \vec{v}(t_i) \tag{5.157}$$

angegeben werden. Unter Verwendung der Gleichungen (3.1) und (5.157) lässt sich die Position der Fußpunkte mit

[1] Gemeint ist hier die Körperbewegung im Sinne von Abschnitt 5.2.4.3

$$\vec{R}_0^{(i)}(t_k) = \vec{r}_0^{(i)} + \vec{r}^{(i)}(t_k) + \vec{R}_0(t_0) + T \cdot \sum_{i=0}^{k} \vec{v}(t_i) \tag{5.158}$$

bestimmen.

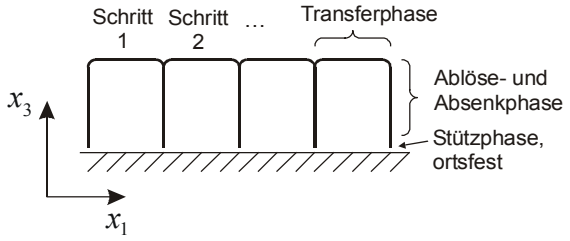



Abbildung 5.51: Lauftrajektorie im Weltkoordinatensystem

Der Laufzyklus ist aus mehreren Phasen zusammengesetzt. Dies sind Stützphase, Ablösephase, Transferphase und Absenkphase. Während der Stützphase ist der Fußpunkt eines Beines ortsfest. In der Ablöse- und Absenkphase werden zwei Geschwindigkeitskomponenten überlagert. Das sind die horizontale Geschwindigkeitskomponente, die identisch ist mit der in der Stützphase und eine vertikalen Komponente. Da Körperbewegung und Stützgeschwindigkeit nach Gleichung (5.155) genau entgegengesetzt sind, kompensieren sich diese im Bezug auf das Weltkoordinatensystem. Abbildung 5.51 zeigt die prinzipielle Form einer solchen Trajektorie. Die Stützphase wird zu einem einzigen Punkt und stellt aufgabengemäß den Kontakt zum Untergrund her. Ablöse- und Absenkphase sind vertikale Bewegungen. Sie würden beim Einsinken in einen weichen Untergrund keine seitliche Kräfte erzeugen, so dass auf den Untergrund nur eine vertikale Kraft ausgeübt wird. Die Transferphase sorgt für den Transport des Fußes über eine neue Aufsetzposition.

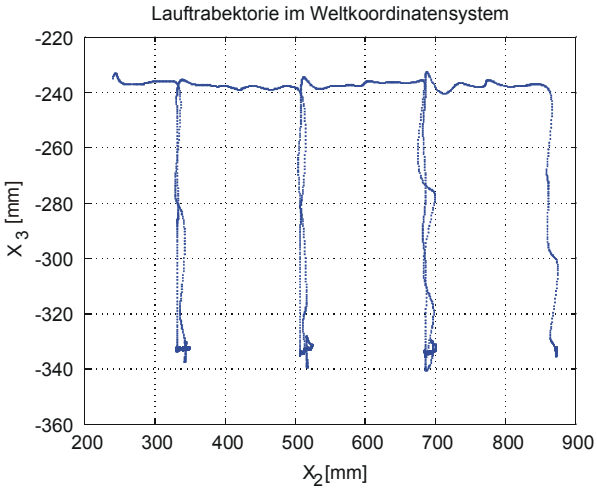



Abbildung 5.52: Aufgezeichnete Lauftrajektorie eines Fußes im Weltkoordinatensystem

Abbildung 5.52 zeigt die mit Hilfe von Gleichung (5.129) ermittelte Lauftrajektorie für ein Bein. Die durch gewünschte Überlagerungen entstandene Trajektorienform ist deutlich zu erkennen und entspricht somit der aus theoretischen Überlegungen entstandenen Trajektorienform. Auch in dieser Darstellung sind wieder die bereits diskutierten Ungenauigkeiten in der Realisierung der Trajektorie zu erkennen. Das wesentliche Merkmal, das vertikale Anheben und Absenken wird jedoch in der Funktion bei kleineren Abweichungen nicht beeinträchtigt.

Die Genauigkeit in der Transferphase ist ohnehin von untergeordneter Bedeutung, da sich das Bein frei bewegen kann. In der Stützphase, wo der Bodenkontakt wichtig ist, sind horizontale Bewegungen unerwünscht, da sie Querkräfte im Boden erzeugen. Im Diagramm ist zu sehen, dass hier offensichtlich kleine Bewegungen auftreten, so dass Querkräfte entstehen. Diese sollten minimiert werden.

Nachdem die Eigenschaften der Lauftrajektorien im Weltkoordinatensystem aufgezeigt wurden, können diese nun zur Prüfung der Ergebnisse beim omnidirektionalen Laufen herangezogen werden.

Zum Prüfen der korrekten Ausführung der Laufbewegungen beim omnidirektionalen Laufen wird folgendes Experiment durchgeführt. Der Roboter läuft zunächst in eine Richtung. Nach einigen Schritten wird ein Richtungswechsel vorgegeben. Wie in Abschnitt 5.2.4.5 beschrieben, wird die Richtungsänderung erst nach Erreichen eines Umschaltpunktes wirksam. Dieser Umschaltpunkt ist den Trajektorien für beide Laufrichtungen gemeinsam. Der Übergang kann hier also problemlos erfolgen. Im Experiment sollte das bei einem Dreifußgang dadurch deutlich werden, dass bei den sich in der Transferphase befindenden Beinen ein Richtungswechsel der Trajektorie in der Mitte der Transferphase erfolgt. Die anderen Beine befinden sich zu diesem Zeitpunkt in der Stützphase, so dass die Richtungsänderung in der Trajektorie erst mit Beginn der folgenden Transferphase sichtbar wird.

Abbildung 5.53 zeigt die beim durchgeführten Experiment aufgezeichneten Fußtrajektorien, die mit Hilfe von Gleichung (5.129) ausgewertet worden sind. Zunächst lief der Roboter beginnend bei Position ($X_1$=0, $X_2$=0), wo er im Diagramm dargestellt ist, vier Schritte in Richtung $X_1$. Dann erfolgte das Kommando für den Richtungswechsel und es folgten weitere fünf Schritte mit der neu eingeschlagenen Richtung. Im Diagramm sind die bereits in der zweidimensionalen Darstellung gezeigten Trajektorienmuster zu erkennen. Sie verlaufen jeweils wodurch zwei Beingruppen zu betrachten sind. Der Richtungswechsel erfolgt beim Durchlaufen der neutralen Punkte (Umschaltpunkte) auf der Trajektorie. Diese befinden sich in der

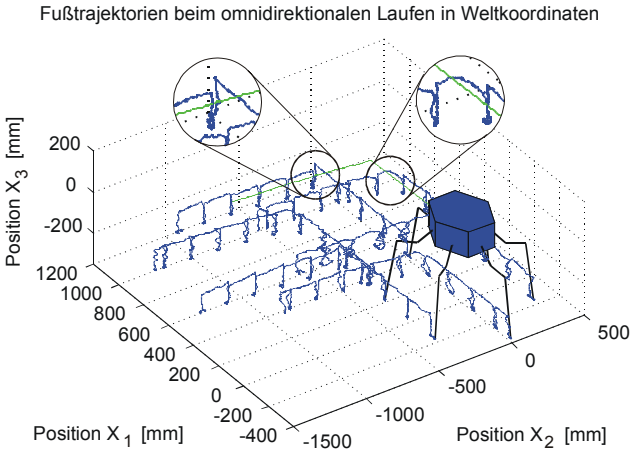



Abbildung 5.53: Aufgezeichnete Fußtrajektorien beim omnidirektionalen Laufen im Weltkoordinatensystem

Mitte der Transferphase und in der Stützphase. In Abbildung 5.53 sind Trajektorienausschnitte mit Richtungswechsel für jede Beingruppe vergrößert dargestellt. Die rechte Vergrößerung zeigt eine Trajektorie, bei der ein Richtungswechsel während der Transferphase stattfand. Es ist zu erkennen, dass der Richtungswechsel ungefähr in der Mitte der Transferphase erfolgte, wo der gemeinsame Punkt der Trajektorien existiert (Umschaltpunkt). Der

Richtungswechsel erfolgt in der vorhergesehenen Art und Weise. Aus der linken Vergrößerung ist zu erkennen, dass nach einem erfolgten Schritt eine Richtungsänderung erfolgte. Auf den Laufzyklus bezogen bedeutet das, dass die Richtungsänderung während der Stützphase stattfand. Da der Fuß während der Stützphase ortsfest gegenüber dem Weltkoordinatensystem ist, lässt sich nicht genau identifizieren, an welcher Stelle der Richtungswechsel stattfand. Für eine weitere Analyse muss die Relativbewegung des Beines gegenüber dem Körper herangezogen werden und ist in Abbildung 5.54 dargestellt. Zu sehen ist jeweils ein Teil der Transfer-, Absenk- und Ablösephase sowie die interessierende Stützphase. In der Mitte der Stützphase hat die Trajektorie einen Knick von ca. 90°, der den Richtungswechsel an der dafür vorgesehenen Stelle dokumentiert.

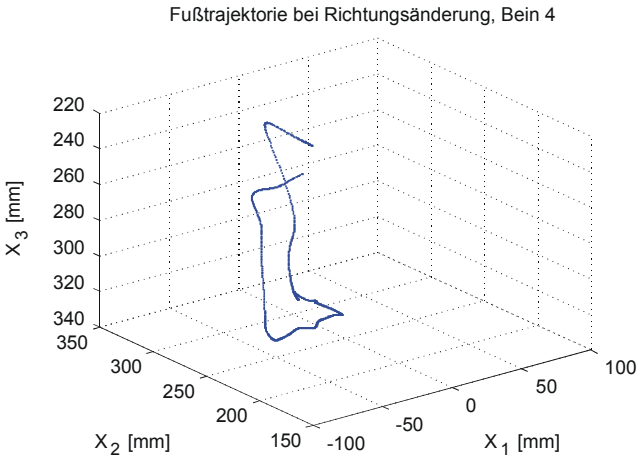



Abbildung 5.54: Fußtrajektorie des Richtungswechsels im Beinkoordinatensystem

### 5.3.2 Erprobung der Körperbewegung

Zur Erprobung der Körperbewegung steht der Roboter in der Grundposition. Es werden dann die verschiedenen Parameter für die Körperposition der Gleichungen (5.69) bis (5.71) variiert und die Positionen beobachtet, die der Roboter einnimmt.

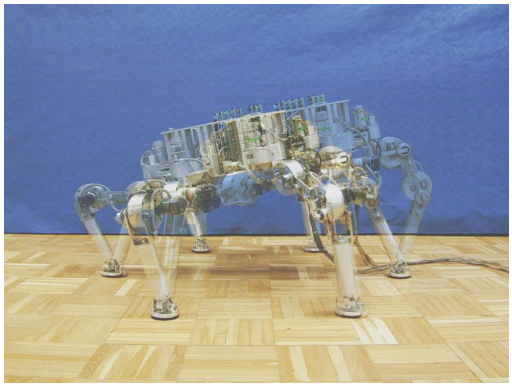


Abbildung 5.55: Beispiel einer Körperbewegung entlang der Körperachsen bei geneigtem Körper

Abbildung 5.55 zeigt ein Beispiel für die Anwendung der Körperbewegung. Dabei wurde eine lineare Körperverschiebung entlang der Körperachsen entsprechend der ersten Teiloperation nach Gleichung (5.69) bei gleichzeitiger Neigung des Körpers entsprechend der zweiten Teil-

operation nach Gleichung (5.70) ausgeführt. Wie zu erkennen ist, bleibt die Körperverschiebung mit Bezug auf die Körperachsen aus der ersten Teiloperation erhalten. Das lässt sich beispielsweise ausnutzen, um die in [Ihme et al. 98] beschriebenen Bohroperationen mit gewünschter Orientierung auszuführen. Im Gegensatz zu den dort beschriebenen Verfahren kann der Körper mit dem hier demonstrierten Algorithmus in allen sechs Freiheitsgraden ausgerichtet werden.

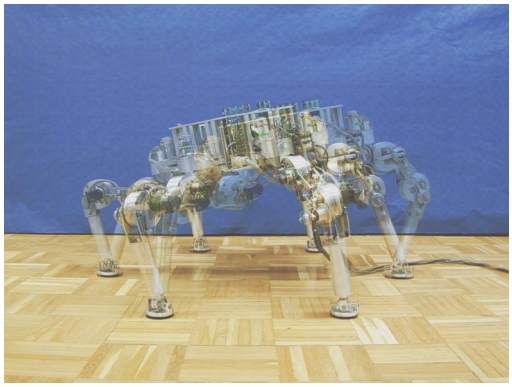


Abbildung 5.56: Beispiel einer Körperbewegung parallel zum Untergrund bei geneigtem Körper

In Abbildung 5.56 hingegen wurde der Körper mit Hilfe der zweiten Teiloperation aus Gleichung (5.70) geneigt und mit Hilfe der dritten Teiloperation aus Gleichung (5.71) parallel zum Untergrund verschoben. Das kann für Manipulationen an externen Objekten oder Handhabungen, zum Beispiel zur Positionierung eines am Roboterkörper befestigten Greifers genutzt werden.

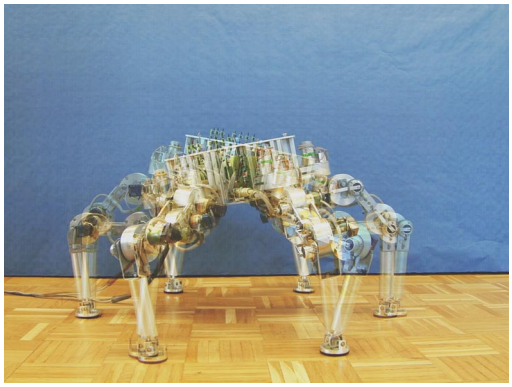


Abbildung 5.57: Beispiel einer rotatorischen Bewegung

Abbildung 5.57 zeigt ein Beispiel einer rotatorischen Körperbewegung durch Variation des Neigungswinkels in der zweiten Teiloperation aus Gleichung (5.70).

### 5.3.3 Kombination von Laufen und Körperbewegung

Zur Erprobung der Kombination von Laufen und Körperbewegung müssen beide Operationen gleichzeitig ausgeführt werden. Im Experiment wird der Körper um einen bestimmten Winkel geneigt und gleichzeitig wird ein Laufkommando gegeben.

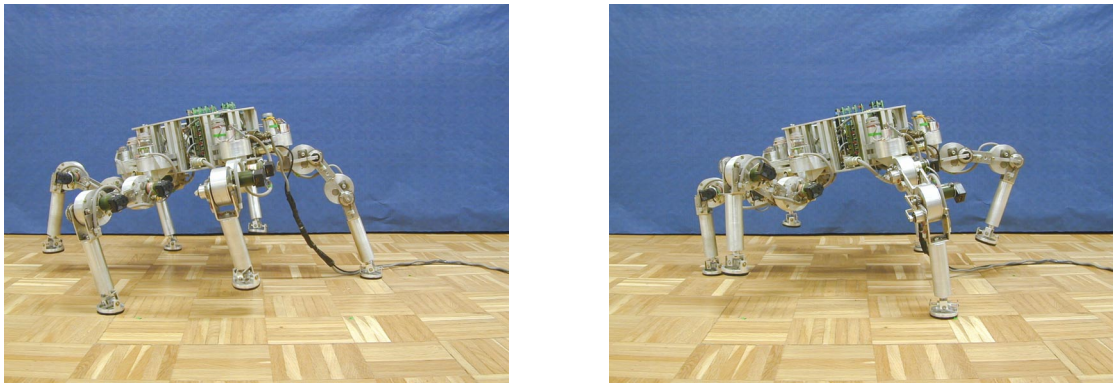


Abbildung 5.58: Beispiele zum Laufen mit geneigtem Körper

Abbildung 5.58 zeigt das Ergebnis der Überlagerung von Körper- und Beinbewegungen. Wie vorgegeben, ist der Körper geneigt, während eine Laufbewegung ausgeführt wird. Dieser Algorithmus lässt sich beispielsweise nutzen, um den Körper auch bei geneigtem Untergrund horizontal zu halten. Damit ist es möglich, einen hoch liegenden Massenschwerpunkt so zu positionieren, dass dessen Projektion sich stets innerhalb des Stabilitätspolygons befindet und nach Möglichkeit eine ausreichende Stabilitätsreserve besteht. Somit kann der Roboter zusätzlich stabilisiert werden. Gleichzeitig kann die Kombination aus Laufbewegung und Körperbewegung dazu genutzt werden, um Instrumente während des Laufens in einer bestimmten Position zu halten oder die Positionierung von Instrumenten über den Arbeitsbereich der Körperbewegung hinaus zu ermöglichen.

### 5.3.4 Erprobung der Kraftwirkungen auf den Roboter

Nachfolgend sollen die Kraftwirkungen untersucht werden, die durch die Interaktion des Roboters mit der Umgebung entstehen und auf die er reagieren kann. Weiterhin soll untersucht werden, wie die aktive Nachgiebigkeit dazu beitragen kann, die unerwünschten Querkräfte beim Laufen abzubauen.

#### 5.3.4.1 Vertikale Kraftwirkungen beim Laufen

Durch die Gewichtskraft des Roboters wirken Kräfte in Richtung der Gravitation auf die stützenden Beine. Diese teilen sich beim Dreifußgang je nach Position auf dem Untergrund in Teilkräfte auf. Eine auf den Fuß wirkende Stützkraft ist um so größer, je näher sich der Massenschwerpunkt des Roboters an den Rändern des Stützpolygons befindet. Durch die Auswertung der vertikalen Kraftkomponenten lässt sich eine Aussage darüber gewinnen, wie groß

die Stabilitätsreserve ist. Auch für das Anpassen und somit das Verteilen der Stützkräfte bei mehr als drei stützenden Beinen ist die Kenntnis über wirkende Stützkräfte wichtig. Zusammen mit der Information über die ideale Kraftverteilung kann die gemessene Kraft zur Realisierung der aktiven Nachgiebigkeit eingesetzt werden.

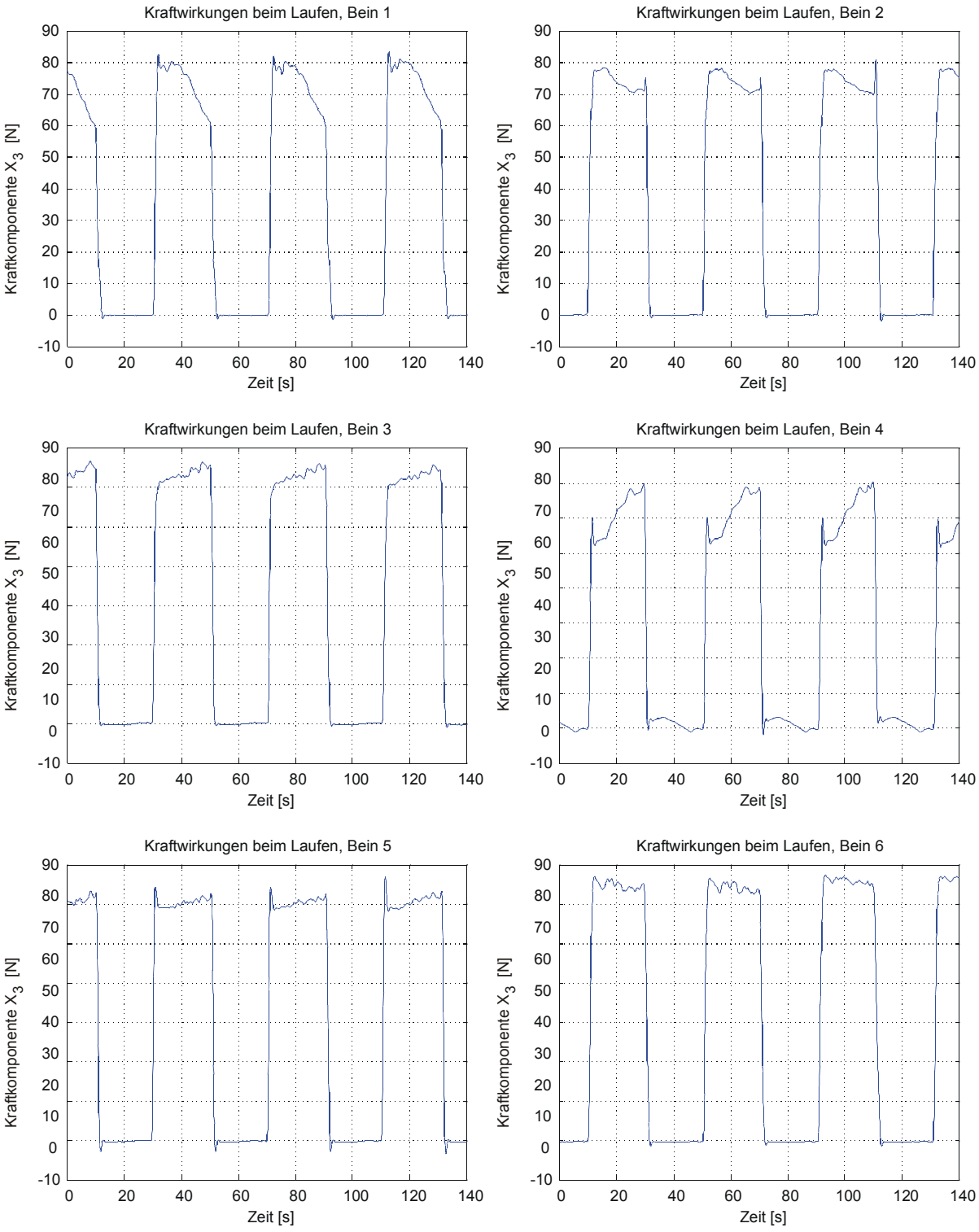



Abbildung 5.59: Vertikale Kraftkomponenten beim Laufen

Wenn die wirkende Stützkraft beim Laufen gemessen wird, so sollte sich die Stützkraft in Abhängigkeit vom Abstand zum projizierten Masseschwerpunkt verändern[2]. Für Stützphasen, bei denen sich der Abstand zwischen projiziertem Masseschwerpunkt und dem Stützpunkt gleichförmig vergrößert oder verringert, sollte die Stützkraft ab- oder zunehmen, falls sich die Konfiguration der Stützfüße nicht verändert.

Zur Messung der beim Laufen auftretenden vertikalen Kräfte wurde ein Laufexperiment durchgeführt, bei dem der Roboter in Richtung Bein 4 lief. Abbildung 5.59 zeigt die beim Laufen gemessenen vertikalen Kraftkomponenten. Dazu wurden die vom Kraftsensor gemessenen Kraftkomponenten in ein dem jeweiligen Beinkoordinatensystem paralleles Koordinatensystem transformiert, um vergleichbare Darstellungen zu erhalten. Da nur die vertikalen Komponenten betrachtet werden, müssen die horizontalen Komponenten nicht in ein gemeinsames Koordinatensystem transformiert werden.

In der Abbildung sind die wirkenden Kräfte während mehrerer Laufzyklen dargestellt. Es ist zu sehen, dass sich die bei den Stützfüßen wirkenden Kräfte im Bereich von 60..80 N bewegen, was etwa einem Drittel der Gewichtskraft des Roboterkörpers in Höhe von ca. 210 N entspricht. Es ist weiterhin die bereits in der Simulation gezeigte Dachform zu sehen. Die wirkende Kraft steigt entweder an oder fällt ab und entspricht damit dem erwarteten Verhalten. Wenn der Roboter in Richtung Bein 4 läuft, so setzt Bein 4 an einem entfernten Punkt auf und nähert sich während der Stützphase dem projizierten Massenschwerpunkt an. Dabei wächst die auf das Bein wirkende Kraft stetig an. Umgekehrt ist es beim Bein 1, das nah am Körper aufsetzt und sich dann von ihm entfernt.

Ein Vergleich mit der berechneten Kraftverteilung aus Abschnitt 5.2.6 zeigt, dass die Kurvenformen mit der berechneten Kraftverteilung qualitativ übereinstimmen. Bei den gemessenen Kräften kann bei einigen Kurven die nichtgleichmäßige Zu- oder Abnahme der Kräfte beobachtet werden, wie sie die simulierte Kraftverteilung zeigt. Die gemessene Kraftverteilung bestätigt die Verwendbarkeit des für die Berechnung der Kraftverteilung genutzten Modells und der zugehörigen Rechnungen.

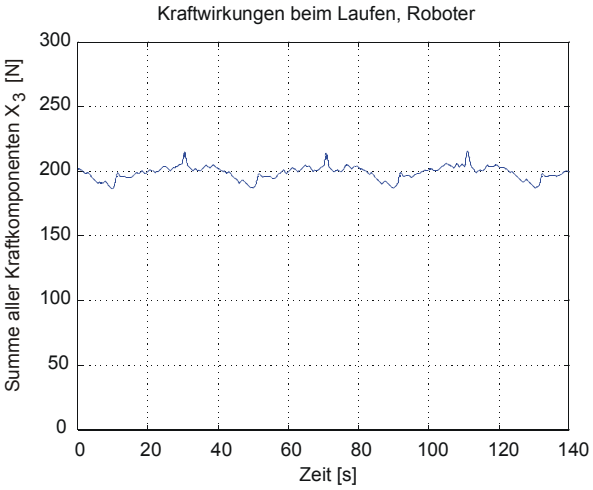



Abbildung 5.60: Summe aller vertikalen Kräfte

Die vertikalen Kräfte müssen zusammen die Gewichtskraft des Roboters ergeben. Abbildung 5.60 zeigt hierzu die für alle Beine summierten Kräfte während des Laufexperimentes. Im

[2] Die Projektion erfolgt in gleicher Weise wie beim Stabilitätspolygon bzw. bei der Berechnung der Kraftverteilung.

Diagramm ist zu erkennen, dass Werte um ca. 200 N ermittelt werden. Das entspricht ungefähr der Gewichtskraft des Roboters, unterliegt aber einer gewissen Schwankung.

#### 5.3.4.2 Horizontale Kraftwirkungen beim Laufen

Zur Untersuchung der horizontalen Kraftwirkungen wurden Experimente mit und ohne aktiver Nachgiebigkeit durchgeführt. Mit diesen Experimenten soll die These geprüft werden, dass durch Implementierung der aktiven Nachgiebigkeit die auftretenden Querkräfte zwischen den Beinen verringert werden können. Die Nachgiebigkeitsparameter werden auf folgende Werte eingestellt: für das Experiment ohne aktiver Nachgiebigkeit

$$k_{FN,X1} = 0\frac{\text{mm}}{\text{N}},\ k_{FN,X2} = 0\frac{\text{mm}}{\text{N}},\ k_{FN,X3} = 0\frac{\text{mm}}{\text{N}} \tag{5.159}$$

und für das Experiment mit aktiver Nachgiebigkeit

$$k_{FN,X1} = 2{,}0\frac{\text{mm}}{\text{N}},\ k_{FN,X2} = 2{,}0\frac{\text{mm}}{\text{N}},\ k_{FN,X3} = 0\frac{\text{mm}}{\text{N}}\ . \tag{5.160}$$

Die auftretenden Querkräfte sollen mit den im Roboter integrierten Kraftsensoren registriert und aufgezeichnet werden. Als Gangart wird der Dreifußgang genutzt. Dadurch treten Querkräfte zwischen den jeweils drei Füßen der Beingruppe auf, die sich in der Stützphase befindet. Der Roboterkörper wird parallel zum Untergrund gehalten, so dass die Beinkoordinatensysteme parallel zum Untergrund sind. Der Gravitationsvektor steht senkrecht zum Untergrund.

Zur Auswertung wurden die vom Kraftsensor gemessenen Kraftkomponenten in ein dem jeweiligen Beinkoordinatensystem paralleles Koordinatensystem transformiert und so gedreht, dass das Koordinatensystem parallel zum Roboterkoordinatensystem ist, um die horizontalen Kraftkomponenten vergleichen zu können.

Die Messergebnisse in Abbildung 5.61 belegen, dass die wirkenden Querkräfte tatsächlich verringert werden können. Es sind die Kraftwirkungen in der horizontalen Ebene ($X_1$, $X_2$) für die aufgezeichneten drei zusammengehörige Beine einer Beingruppe beim Dreifußgang (hier Beine 2, 4 und 6) dargestellt. In der Darstellung sind die Kraftkomponenten parallel zur Roboterkoordinatensystem dargestellt. Die Teilabbildungen a, c und e zeigen die Kraftwirkungen ohne aktive Nachgiebigkeit, Teilabbildungen b, d und f zeigen die Kraftwirkungen mit aktiver Nachgiebigkeit. Man erkennt jeweils eine kleine Punktwolke um den Punkt (0,0), die die fehlenden Querkräfte in der Returnphase anzeigt und eine interessierende große Punktwolke in der Stützphase. Es sind zwei Effekte zu erkennen. Ein Effekt besteht darin, dass mit aktiver Nachgiebigkeit die Punktwolken näher am Punkt (0,0) sind. Das bedeutet, dass die wirkenden Kräfte geringer sind, was die These bestätigt. Ein weiterer Effekt zeigt sich darin, dass die Punktwolken mit aktiver Nachgiebigkeit kompakter sind. Das deutet darauf hin, dass Vibrationen und Schwingungen in der mechanischen Struktur gedämpft werden und somit die Positioniergenauigkeit der Füße durch Kraftführung verbessert wird.

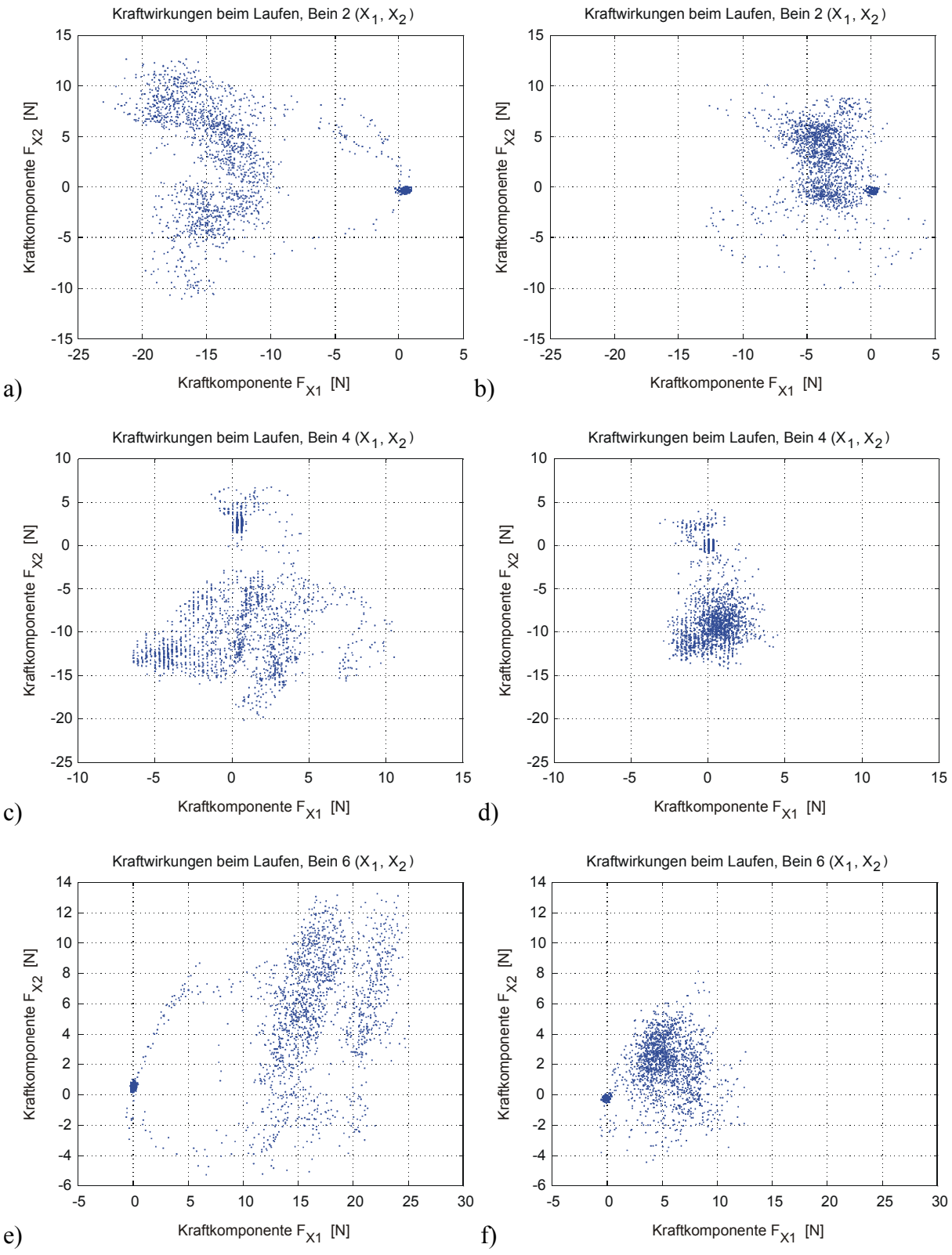



Abbildung 5.61: Kraftwirkungen ohne und mit aktiver Nachgiebigkeit

Zusammenfassend kann festgestellt werden, dass die aktive Nachgiebigkeit einen deutlichen Effekt bei der Verringerung der Querkräfte hat und somit eine einfache und wirkungsvolle Methode für deren Reduzierung ist.

### 5.3.4.3 Kraftwirkungen auf ein einzelnes Bein

Zur weiteren Untersuchung der aktiven Nachgiebigkeit soll festgestellt werden, wie ein einzelnes Bein auf wirkende äußere Kräfte reagiert. Dies dient der Überprüfung, ob das Steuergesetz für die aktive Nachgiebigkeit bei einem Roboterbein die gewünschte lineare Abhängigkeit erzeugt.

Die Nachgiebigkeit der Beine wird mit Hilfe der Nachgiebigkeitsmatrix $\mathbf{K}_{FN}$ festgelegt. Wird die Matrix diagonal besetzt, so wirkt die Nachgiebigkeit in Richtung der wirkenden Kräfte

$$\mathbf{K}_{FN} = \begin{pmatrix} k_{FN,x} & 0 & 0 \\ 0 & k_{FN,y} & 0 \\ 0 & 0 & k_{FN,z} \end{pmatrix} . \tag{5.161}$$

Zur Durchführung des Experimentes mit einem einzelnen Bein werden die Nachgiebigkeitskoeffizienten auf jeweils 3,5 mm/N eingestellt

$$k_{FN,X1} = 3{,}5\frac{\text{mm}}{\text{N}},\ k_{FN,X2} = 3{,}5\frac{\text{mm}}{\text{N}},\ k_{FN,X3} = 3{,}5\frac{\text{mm}}{\text{N}} \ . \tag{5.162}$$

Der Roboterkörper wird räumlich fixiert, z.B. wird er auf einen angepassten Sitzhocker platziert. Demzufolge wirken auf die Kraftsensoren nur noch sehr geringe und zu vernachlässigende Gewichtskräfte, die durch das Eigengewicht der Roboterfüße verursacht werden. Für den Nachweis wird Bein 1 in Richtung der Beinkoordinate $X_2$ mit einer Kraft beaufschlagt. Dabei werden die einwirkende Kraft und die Auslenkung des Fußpunktes registriert.

Für eine Kraft von 20 N sollte folgende Auslenkung entstehen

$$\begin{aligned} \Delta X_2 &= k_{FN,X2} \cdot F_{X2} \\ &= 3{,}5\frac{\text{mm}}{\text{N}} \cdot 20\,\text{N} = 70\,\text{mm} \end{aligned} \ . \tag{5.163}$$

Die Registrierung der Kräfte erfolgt mittels Dynamometer und robotereigener Kraftsensorik. Die Auslenkung wird über die robotereigene Positionssensorik und ein Messlineal registriert. Bei einer über das Dynamometer eingebrachten Kraft von ca. 20 N misst man eine Auslenkung von ca. 70 mm. Das Verhalten entspricht damit der Vorhersage.

Abbildung 5.62 zeigt den Zusammenhang zwischen gemessener Kraft und gemessener Auslenkung. Man erkennt aus den Messwerten den linearen Zusammenhang zwischen wirkender Kraft und Auslenkung. Dies verdeutlicht die eingezeichnete Gerade, die mit Hilfe der Methode der kleinsten Quadrate ermittelt wurde. Der Anstieg dieser Geraden beträgt 3,6 mm/N und liegt damit sehr nahe am eingestellten Parameter. Die Streuung der gemessenen Wertepaare ist mit Positionierungenauigkeiten des Regelungssystems bei der Bewegung und einer Quantisierung durch die Integerarithmetik zu erklären. Die dargestellten Messungen belegen qualitativ das gewünschte Verhalten. Die Streuungen sind akzeptierbar, wenn trotzdem der gewünschte Effekt beim Gesamtsystem Roboter erzielt werden kann. Es kann festgehalten werden, dass ein einzelnes Bein entsprechend dem Steuergesetz bei einer wirkenden Kraft mit einer proportionalen Auslenkung reagiert.

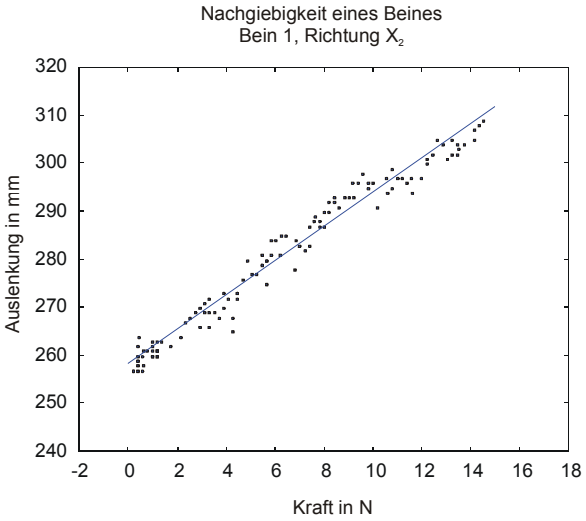



Abbildung 5.62: Verhältnis Kraft und Auslenkung für ein einzelnes Bein

#### 5.3.4.4 Kraftwirkungen auf den Roboter

Nachdem untersucht wurde, wie die aktive Nachgiebigkeit die Querkräfte beim Laufen verringert und wie das konkrete Verhalten eines einzelnen Beines bei einer externen wirkenden Kraft ist, soll abschließend getestet werden, welchen Effekt die aktive Nachgiebigkeit bei einer externen, auf den Roboter wirkenden Kraft zeigt.

Zur Untersuchung dieses Effektes wird der Roboter mit allen sechs Füßen auf den Boden gestellt. Da die Beine symmetrisch um den Körper angeordnet sind, verteilt sich die Gewichtskraft gleichmäßig auf die Füße. Die Nachgiebigkeitsparameter werden für jedes Bein auf folgende Werte gesetzt

$$k_{FN,X1} = 2{,}5\frac{\text{mm}}{\text{N}},\ k_{FN,X2} = 2{,}5\frac{\text{mm}}{\text{N}},\ k_{FN,X3} = 1{,}5\frac{\text{mm}}{\text{N}}. \tag{5.164}$$

Wenn eine Kraft am Roboter zu wirken beginnt, so werden die sechs Kraftsensoren in den Beinen eine entsprechende Teilkraft registrieren. Daraufhin werden entsprechende Korrekturwerte für die Beinpositionen berechnet. Im Experiment soll eine horizontal wirkende Kraft angreifen. Es wird vereinfachend angenommen, dass sich die Kräfte gleichmäßig zu je einem Sechstel auf die Füße verteilen, da die Robotermechanik die Kräfte überträgt. Dadurch entsteht eine gleichartige Reaktion der Beine. Dies hat zur Folge, dass die Verschiebung des Roboterkörpers von allen Beinen gleichartig unterstützt wird.

Greift z.B. eine horizontale Kraft von 50 N an, so ergibt sich mit der angenommenen gleichmäßigen Verteilung der Kräfte folgende Reaktion

$$\begin{aligned}\Delta x &= K_{F.Roboter} \cdot F = \frac{1}{6} K_{F,Bein} \cdot F \\ &= \frac{1}{6} \cdot 2{,}5\frac{\text{mm}}{\text{N}} \cdot 50\text{N} = 20{,}8\ \text{mm}\end{aligned}. \tag{5.165}$$

Im Experiment wird eine auf den Roboterkörper horizontal wirkende Zugkraft ausgeübt, die mit einem Dynamometer registriert wird. Die entstehende Auslenkung wird mit einem Messlineal ermittelt. Nach Einbringen der Zugkraft am Roboterkörper zeigte sich eine Verschiebung des Roboterkörpers von ca. 2 cm. Es wurde beobachtet, dass diese Bewegung durch motorischen Antrieb verursacht wurde.

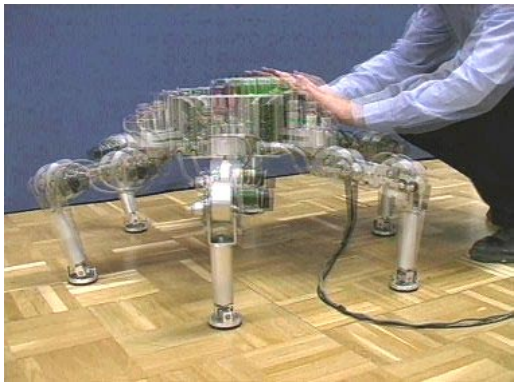


Abbildung 5.63: Horizontale Kraftwirkung bei aktiver Nachgiebigkeit

Weiterhin wird getestet, wie der Roboter auf Kräfte aus anderen Richtungen reagiert. Es ist zu erwarten, dass er wie eine mehrdimensionale mechanische Feder reagiert. Da für jedes Bein alle drei Freiheitsgrade zur Verfügung stehen, kann sich der Roboterkörper entsprechend der festgelegten Parameter mit allen sechs Freiheitsgraden bewegen. Bei einem wirkenden Kräftepaar wird der Roboter folglich mit einer Drehbewegung reagieren. Abbildung 5.63 zeigt die Reaktion des Roboterkörpers. Darin wurden die Bilder ohne und mit maximaler Kraftwirkung überlagert. Dabei ist die horizontale Ausweichbewegung deutlich zu erkennen.

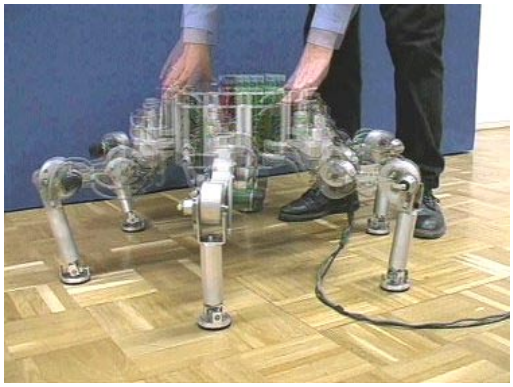


Abbildung 5.64: Vertikale Kraftwirkung bei aktiver Nachgiebigkeit

Bei vertikal einwirkender Kraft erfolgt eine Ausweichbewegung des Roboterkörpers, wie Abbildung 5.64 zeigt. Entsprechend reagiert der Roboter auf ähnlich eingebrachte Kräfte aus verschiedenen Richtungen. Auf diese Weise können auch Verdrehungen des Roboterkörpers erzeugt werden.

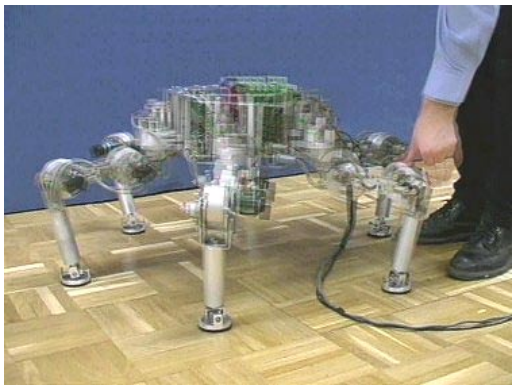


Abbildung 5.65: Kraft auf Unterschenkel bei aktiver Nachgiebigkeit

Abschließend wird die Reaktion auf direkt am Kraftsensor einwirkende Kräfte getestet. Dazu wird eine Kraft entlang der Unterschenkelachse eingebracht und die Reaktion beobachtet. Hier sind allein die auf den Kraftsensor wirkenden Kräfte für eine Reaktion maßgeblich, andere Bewegungsursachen können somit ausgeschlossen werden. Abbildung 5.65 zeigt die Reaktion des Roboters auf eine senkrecht auf den Unterschenkel wirkende Kraft. Die direkt auf den Unterschenkel eingebrachte Kraft wirkt unmittelbar auf den Kraftsensor. Der Nachgiebigkeitsalgorithmus verarbeitet diese Information und erzeugt eine Auslenkung für das betreffende Bein, so dass der Körper gesenkt würde. Damit nimmt die Last auf die anderen Beine zu, die hier ebenfalls mit einer entsprechenden Auslenkung reagieren, so dass sich der Körper wie in der Abbildung gezeigt neigt.

Bei den Experimenten ist noch ein weiterer Effekt zu beobachten. Bei sehr großer Auslenkung und großer vertikaler Nachgiebigkeit beginnt sich der Roboter zu neigen. Dieser Effekt entsteht dadurch, dass der Schwerpunkt des Roboters sich einigen Beinen nähert und sich von anderen entfernt. Damit wirkt auf einige Beine ein größerer Teil der Gewichtskraft, während andere Beine entlastet werden. Diesem Effekt kann durch die in Abschnitt 5.2.6 vorgestellten Berechnungen zur Kraftverteilung entgegengewirkt werden, indem Sollkräfte entsprechend der zu erwartenden Kraftverteilung vorgegeben werden.

Abschließend kann festgehalten werden, dass der Roboter durch Anwendung der aktiven Nachgiebigkeit auch auf externe Kräfte reagiert. Damit kann die aktive Nachgiebigkeit dort eingesetzt werden, wo es auf Interaktion mit externen Objekten ankommt und eine Adaption an externe Kraftwirkungen nötig ist.

## 5.4 Zeitverhalten und Implementierung

Ausgehend vom entwickelten Konzept eines Steuerungssystems soll nachfolgend die Implementierung der vorgestellten Steuerungsalgorithmen für den Laufroboter Katharina auf Basis der verwendeten Hardware dargestellt werden. Dabei muss den durch das mechatronische System vorgegebenen Echtzeitanforderungen genügt werden.

### 5.4.1 Zusammenstellung der Anforderungen

Für die Realisierung der Steuerung muss jeder Controller folgende Funktionen übernehmen:

- Auslesen von Sensoreninformationen bzw. Anforderung von Informationen der untergeordneten Ebene,
- Verarbeitung dieser Informationen und Weitergabe an die übergeordnete Ebene,
- Entgegennahme von Vorgaben der übergeordneten Ebene,
- Verarbeitung von Sensorinformationen und von Steuerinformation der übergeordneten Ebene für auszuführende Aktionen und
- Ausgabe von Informationen für auszuführende Aktionen bzw. von Stellwerten an die untergeordnete Ebene.

Dabei kommt dem Hauptcontroller die Aufgabe zu, die notwendigen Bewegungen zu beschreiben, während die Beincontroller die Bewegungen realisieren. Daraus ergeben sich unterschiedliche Zeitanforderungen.

#### 5.4.1.1 Beincontroller

Die Beincontroller realisieren die Beinbewegungen sowie die aktive Nachgiebigkeit. Dafür sind folgende Informationen zu verarbeiten:

- Regelung der Position der Einzelgelenke in Abhängigkeit von Soll- und Istpositionen nach Gleichungen (5.126) und (5.117),
- Berechnung der kartesischen Raumposition (Istposition) eines Beines mit Hilfe der direkten Kinematik nach Gleichung (5.81),
- Berechnung der beinbezogenen Kraftkomponenten mit Hilfe der Krafttransformation nach Gleichung (5.106),
- Berechnung der kartesischen Sollpositionen der Beine mit aktiver Nachgiebigkeit aus vorgegebener Sollposition, wirkenden Kräften und Sollkräften nach Gleichung (5.125) und
- Berechnung der Gelenksollpositionen aus den kartesischen Sollpositionen mit Hilfe der inversen Kinematik nach Gleichungen (5.85), (5.89), (5.92), (5.96).

Für die Kommunikation, das Auslesen der Sensoren und das Ansteuern der Aktoren sind folgende Funktionen zur Verfügung zu stellen:

- Einlesen der analogen Sensorensignale (3 Achspotentiometer, 3 Kraftkomponenten),
- Ausgabe von Signalen der Puls-Weiten-Modulation zur Ansteuerung der drei Gelenkantriebe,
- Empfangen von Steuerinformationen vom Hauptcontroller,
- Ausgabe von Sensorinformationen an den Hauptcontroller und
- Realisierung des Kommunikationsprotokolls.

Die zeitlichen Anforderungen an die Steuerungsalgorithmen der Beincontroller werden durch die Zeitkonstanten der Regelstrecken und des gesamten mechatronischen Systems bestimmt, wenn es mit der Umwelt interagiert. Bei den Regelstrecken der Gelenkregler lassen sich die Zeitkonstanten beispielsweise durch experimentelle Identifikation bestimmen. Für das gesamte mechatronische System lagen während der Implementierung in den Laufroboter Katharina keine gesicherten Informationen über Zeitkonstanten vor. Da eine kinematische Kette nicht schneller reagieren kann als ihre einzelnen Elemente, sind die Zeitkonstanten der Einzelgelenke maßgeblich. Durch Identifikation der Streckenzeitkonstanten ist bekannt, dass diese im Bereich von 100 ms liegen. Damit die Gelenkregler nach der kontinuierlichen Regelungstheorie entworfen werden können, sollte der kontinuierliche Prozess mit der mindestens fünffachen Tastfrequenz abgetastet werden, als durch die Zeitkonstanten vorgegeben, also mit
20 ms. Nach [Lutz, Wendt 98] und [Franklin et al. 92] wird eine zehnfache Abtastfrequenz für quasikontinuierliche Regelungen empfohlen. Da jedoch stets mehrere Beine auf dem Untergrund stehen, kann dadurch eine Versteifung der gesamten Struktur eintreten, so dass im Gesamtsystem kleinere Zeitkonstanten auftreten können, d.h. eine höhere Eigenfrequenz zu verzeichnen ist. Um diese Effekte beobachten oder kontrollieren zu können, wird eine Abtastzeit von 10 ms angestrebt.

Die Algorithmen für die aktive Nachgiebigkeit verarbeiten Kraftsensorinformationen und generieren daraus Steuerinformationen. Die wirkenden Kräfte können sich entsprechend der vorhandenen Steifheit des Roboters und der Interaktion mit der Umgebung schnell ändern. Es wird daher vorgesehen, die Algorithmen zur aktiven Nachgiebigkeit ebenfalls im Regelungstakt von 10 ms abzuarbeiten.

Zur Abwicklung der Kommunikationsprotokolle muss auf Kommunikationsanforderungen innerhalb der vom Protokoll vorgegebenen Fristen reagiert und alle zeitlichen Vorgaben des Protokolls eingehalten werden. Zur Realisierung der Puls-Weiten-Modulation entstehen weitere zeitliche Anforderungen, deren Einhaltung die Qualität des erzeugten Puls-Weiten-modulierten Signals, d.h. des Tastverhältnisses, beeinflusst. Die Anforderungen hängen von der zu realisierenden Frequenz und der Auflösung der Puls-Weiten-Modulation ab.

#### 5.4.1.2 Hauptcontroller

Der Hauptcontroller übernimmt die Koordination der Beine und realisiert die Körpersteuerung. Neben seiner Aufgabe als zentrale Steuerungsinstanz realisiert er auch die Kommunikation mit der externen Steuerungsebene und den Beincontrollern.

Für die Steuerung und Koordinierung der Beine muss folgender Informationsverarbeitungsprozess realisiert werden:

- Generierung von Bewegungsparametern (Körperbewegung, Schrittparameter) aus Operatorvorgaben und Sensorinformationen, z.B. Schritthöhe, Schrittweite, Laufrichtung,
- Konstruktion des Musterlaufzyklus nach den vorgegebenen Parametern gemäß Gleichungen (5.52), (5.54)-(5.56) und (5.60),
- Interpolation von Zwischenpositionen auf der Mustertrajektorie für jeden Zeitschritt,
- Transformation der jeweiligen Fußpositionen nach Gleichungen (5.66)-(5.68),
- Berechnung der Koordinaten der Beinbefestigungspunkte für die Körperbewegung nach Gleichungen (5.69) bis (5.71),
- Ermittlung der Beinpositionsvektoren nach Gleichungen (5.73) und (5.74),
- Bestimmung der Sollkräfte für die einzelnen Füße (Vorgabe oder Kraftverteilung) und
- Steuerung von Körperbewegungen mit Hilfe von zentralen Sensoren (optional).

Für die Kommunikation, das Auslesen der Sensoren und das Ansteuern der Aktoren sind folgende Funktionen zur Verfügung zu stellen:

- Empfangen von Steuerkommandos der externen Steuerung,
- Senden von Sensor- und Statusinformationen an die externe Steuerung,
- Anforderung von Informationen der Beinsteuerungsebene,
- Senden der Steuerinformationen an die Beincontroller,
- Realisierung des Kommunikationsprotokolls,
- Auslesen der zentralen Sensoren (optional) und
- Ansteuern der zentralen Aktoren mittels Puls-Weiten-Modulation (optional).

Die zeitlichen Anforderungen werden hauptsächlich durch den notwendigen Interpolationstakt für die Beinbewegungen bestimmt.

Für eine maximal angenommene Fußgeschwindigkeit in der Transferphase von 50 mm/s wird gefordert, dass Stützstellen für Abstände von 2 mm berechnet werden. Der geforderte Interpolationstakt ist

$$T_{\text{interp}} = \frac{\Delta x}{v_{\max}} = \frac{2\ \text{mm}}{50\ \text{mm/s}} = 40\ \text{ms}\ . \tag{5.166}$$

Für die zeitlichen Bedingungen zur Realisierung der Kommunikation gilt das Gleiche wie beim Beincontroller. Sollen auch vom Hauptcontroller Sensor-Aktor-Systeme gesteuert werden, beispielsweise ein Kamerakopf, gelten die für die Beincontroller genannten zeitlichen Anforderungen an die Steuerungs- und Regelungsfunktionen entsprechend.

### 5.4.2 Abhängigkeiten

Im Folgenden werden die zur Steuerung des Laufroboters Katharina notwendigen Tasks auf ihre Datenabhängigkeit untereinander untersucht, um diese koordinieren zu können.

#### 5.4.2.1 Beincontroller

Die Sensorinformationen der Beincontroller werden für die Gelenkregelung, die aktive Nachgiebigkeit und zur Weitergabe an höhere Steuerungsebenen benötigt.
Die Gelenkregelungen sind für alle drei Gelenke eines Beines unabhängig. Es werden die aktuellen Sensorinformationen, d. h. die Gelenkwinkel und Sollwerte, zu Steuerinformationen verarbeitet. Der Gelenkregler gibt die Stellwerte als Betrag und Richtung für die Puls-Weiten-Modulation aus. Das Abtasten der Sensorwerte und die Ausgabe der Stellwerte für die Puls-Weiten-Modulation sollte zur Vermeidung von Totzeiten bedingt durch Rechenzeiten möglichst zeitnah erfolgen.

Bei der Erzeugung eines Puls-Weiten-modulierten Signals mit Hilfe der verwendeten Hardware lassen sich zwei Aufgaben unterscheiden. Das ist die Ausgabe des PWM-Signals mit konstantem Tastverhältnis sowie die Änderung des Tastverhältnisses. Bei der Ausgabe eines Signals mit konstantem Tastverhältnis besteht keine Abhängigkeit zu anderen Tasks. Bei der Änderung des Tastverhältnisses besteht eine Abhängigkeit zu den Algorithmen zur Gelenkregelung.

Die Gelenkkoordinaten müssen in kartesische Beinkoordinaten umgewandelt werden. Voraussetzung dafür ist die Kenntnis der aktuellen Gelenkwinkel.

Die Krafttransformation wandelt die Kraftinformationen des Kraftsensors in kartesische Kraftkomponenten um, wozu die Gelenkkoordinaten benötigt werden.

Für die aktive Nachgiebigkeit werden die kartesischen Istkraftkomponenten, die Sollkraftkomponenten und die vom Hauptcontroller übergebene Fußposition verarbeitet. Als Ergebnis erhält man die korrigierte Fußposition.

Bei der inversen Koordinatentransformation wird die korrigierte Fußposition in die drei Gelenksollwerte transformiert.

In Abbildung 5.66 ist zu erkennen, dass zwei wesentliche Abgängigkeitspfade vorhanden sind. Im unteren Pfad ist die Abhängigkeit für den Gelenkregler zu erkennen. Die beiden oberen Pfade sind eine lange Kette von abhängigen Tasks, die durch die Realisierung der aktiven Nachgiebigkeit entstehen. Es kommt dabei zu einer Verknüpfung der in die Beinkoordinatendarstellung transformierten Kraftsensorinformationen und der vom Hauptcontroller vorgegebenen Sollposition und Stützkraft eines Beines zu einer modifizierten Sollposition, die anschließend zu Sollwerten für die Gelenkwinkel transformiert wird. Zusätzlich werden die aktuellen Gelenkpositionen in das Beinkoordinatensystem transformiert. Zusammen mit den transformierten Kraftsensorinformationen ergeben sie einen Vektor aller Sensorwerte. Dieser Vektor wird für die nächsthöhere Ebene bereitgestellt. Da in den beiden oberen Pfaden durchgängig eine Abhängigkeit vorhanden ist, das heißt, die einzelnen Tasks sequentiell gebunden sind, können diese zu einer einzigen Task für die aktive Nachgiebigkeit zusammengefasst werden.

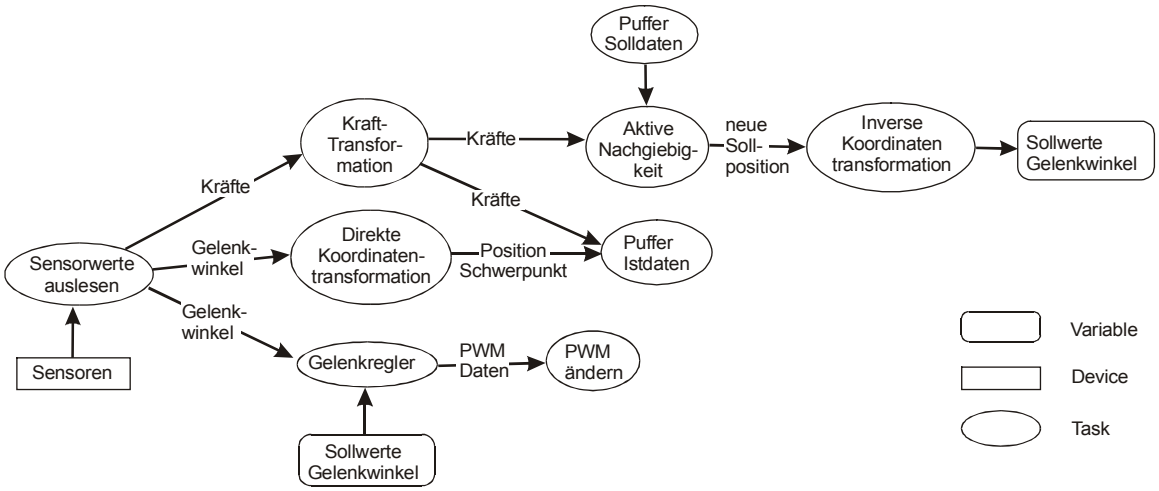



Abbildung 5.66: Abhängigkeitsgraph für die Tasks der Steueralgorithmen eines Beincontrollers

Die Task zur Gelenkregelung und die Task für die aktive Nachgiebigkeit sind unabhängig und können somit getrennt implementiert werden.

Da beide Tasks gemäß Abschnitt 5.4.1.1 mit der gleichen Periode ablaufen sollen, ist in diesem speziellen Fall eine Sequenzialisierung aller Verarbeitungsschritte aus Abbildung 5.66 möglich.

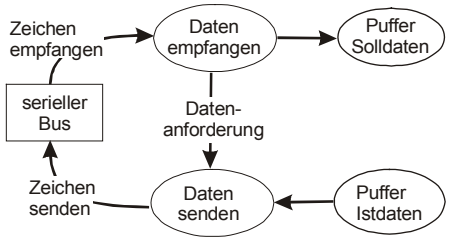



Abbildung 5.67: Abhängigkeitsgraph für die Kommunikation eines Beincontrollers

In Abbildung 5.67 ist der Abhängigkeitsgraph für die Kommunikation eines Beincontrollers dargestellt. Beim Empfang von Daten über den seriellen Bus wird die Empfangstask aktiviert. Diese speichert die empfangenen Daten im Solldatenpuffer. Nach dem Konzept der Master-Slave-Kommunikation wird die Übertragung der Sensordaten vom Hauptcontroller angefordert. Somit wird bei einem Anforderungstelegramm die Sendetask aktiviert.

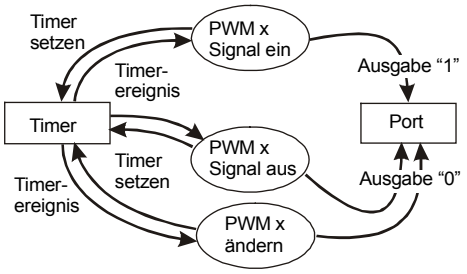



Abbildung 5.68: Abhängigkeitsgraph für die Puls-Weiten-Modulation beim Beincontroller

In Abbildung 5.68 ist der Abhängigkeitsgraph für die Puls-Weiten-Modulation dargestellt. Es findet das in Abschnitt 5.1.3.2 beschriebene Verfahren Anwendung, das mit Hilfe von drei Tasks modelliert werden kann. Zwei Tasks realisieren das Ausgaben von logisch 1 bzw. logisch 0. Diese Tasks werden durch Timer-Ereignisse nach jeweils einer PWM-Periode gestartet. Dazu wird der Timer von den Tasks entsprechend programmiert. Zur Änderung des Tastverhältnisses wird eine dritte Task benötigt, die alternativ zur Task zur Ausgabe von logisch 0 ausgeführt wird. Sie initialisiert das folgende Timer-Ereignis zur Ausgabe von logisch 0 so, dass sich ein neues Verhältnis von Ein- und Ausschaltzeit ergibt. Zusätzlich gibt sie das Drehrichtungssignal aus.

Zusammenfassend gibt es folgende periodische Tasks:

- Gelenkregelung,
- aktive Nachgiebigkeit,
- Puls-Weiten-Modulation einschalten und
- Puls-Weiten-Modulation ausschalten in zwei alternativen Varianten.

Dabei haben Gelenkregelung und aktive Nachgiebigkeit eine Periode von 10 ms, während das Umschalten der Signalpegel der Puls-Weiten-Modulation im Takt ihrer Frequenz auftritt.

Die Kommunikationstasks, d.h. das Senden und Empfangen sind ereignisgesteuert. Eine Periode bzw. die Zeitpunkte der Aktivierung der Kommunikationstasks können nicht ohne weitere Informationen bestimmt werden, da sie wegen des verwendeten Master-Slave-Protokolls vom Hauptcontroller abhängen.

#### 5.4.2.2 Hauptcontroller

Die Steuerparameter für den Hauptcontroller werden nach Eingang eines Steuertelegramms zu Parametern für den Laufzyklus, die Körperbewegungen und die aktive Nachgiebigkeit der Beine verarbeitet.

Für den Laufzyklus wird in Abhängigkeit von der Position auf der Lauftrajektorie entweder der kommende Interpolationsschritt berechnet oder es erfolgt zusätzlich die Verarbeitung der Steuerparameter zu Laufzyklus- und Interpolationsparametern. Des weiteren können zu den Vorgabeparametern auch Sensorinformationen, z.B. für die Schritthöhe verarbeitet werden.

Aus den für den aktuellen Zeitschritt interpolierten Laufzyklluspositionen für jede Beingruppe werden durch Transformationen alle Fußpositionen in Roboterkoordinaten gewonnen. In die Transformationen können Parameter zur Bodenkontakthöhe einfließen.

Für die Körperposition erfolgt entsprechend der vorgegebenen Steuerparameter eine Transformation der Beinbefestigungspunkte, die die Lage des Körpers im Raum beschreiben. Aus den Koordinaten der Fußpositionen und der Beinbefestigungspunkte werden die Beinpositionen ermittelt.

Die Parameter für die aktive Nachgiebigkeit werden anhand der Koordinaten der Masseschwerpunkte von Beinen und Körper bestimmt und daraus die Stützkräfte für die Beine ermittelt.

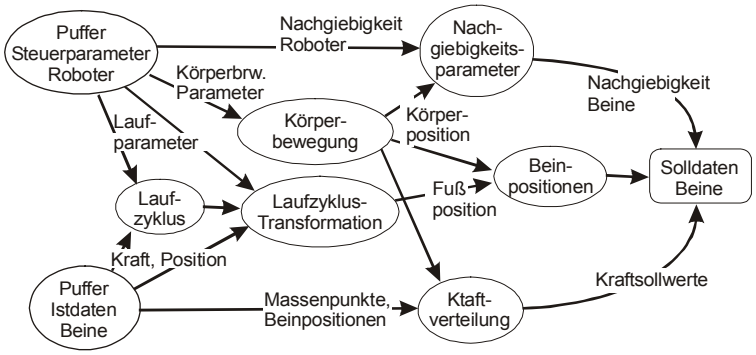



Abbildung 5.69: Abhängigkeitsgraph für die Tasks der Steueralgorithmen beim Hauptcontroller

Abbildung 5.69 zeigt den Abhängigkeitsgraphen für die Haupttask beim Hauptcontroller. Kern der Funktionalität ist das Erzeugen des Laufzyklus und der Körperbewegung. Entsprechend der Anforderungen sollen die Koordinaten der Fußpositionen in den jeweiligen Beinkoordinatensystemen in Zeitabständen von 40 ms erzeugt werden. Diese werden durch die Ergebnisse zur Körperpositionierung und der Laufbewegungen beeinflusst. Wird beispielsweise eine Laufbewegung erzeugt, müssen durch die Trajektorieninterpolation im gleichen Takt Fußpositionen generiert werden. Für eine zu erzeugende Körperbewegung sind die Anforderungen entsprechend. Das heißt, dass bei gleichzeitiger Nutzung von Körperbewegung und Laufen erst beide Ergebnisse vorliegen müssen, um die resultierenden Beinpositionen zu ermitteln. Wird die Körperbewegung sensorbasiert gesteuert, so muss diese fortlaufend berechnet werden.

Wird auch die aktive Nachgiebigkeit genutzt, so müssen die Nachgiebigkeitsparameter für alle Beine in das entsprechende Beinkoordinatensystem transformiert werden. Da die Beine mit dem Körper verbunden sind, hängen die Transformationsparameter von der Körperposition ab. Bei der Nutzung der vertikalen Komponente der aktiven Nachgiebigkeit muss zusätzlich die Kraftverteilung bestimmt werden. Diese hängt von allen Massepunkten und deren Koordinaten ab. Da die Schwerpunkte der Beine von den Beincontrollern berechnet werden, ist zusätzlich der Schwerpunkt des Körpers notwendig, um den Gesamtschwerpunkt und die resultierende Kraftverteilung zu bestimmen.

Sind zusätzliche Sensoren und Aktoren am Roboter vorhanden, so müssen die zusätzlichen Funktionen entsprechend ihrer Aufgabe realisiert werden. Die Aktorenansteuerung kann wie beim Beincontroller entsprechend Abbildung 5.68 ausgeführt werden.

Die Beinpositionen werden periodisch alle 40 ms erzeugt und innerhalb der Periode an die Beincontroller weitergeleitet. Dazu werden die Steuerinformationen über den Bus an die Beincontroller übertragen. Mit der gleichen Periode müssen die Sensorinformationen der Beincontroller mittels Master-Slave-Kommunikation abgefragt werden.

Das Erzeugen der Laufbewegung, der Körperbewegung und der Ableitung der Beinpositionen kann als eine Task zusammengefasst werden, da sie sequenziell gebunden sind. Bei der Nutzung der aktiven Nachgiebigkeit und der Kraftverteilung entstehen weitere sequenzielle Bindungen. Die Aktualisierung der Informationen zur Kraftverteilung ist nicht zwingend mit der gleichen Periode notwendig, wenn sie sich nur geringfügig ändert.

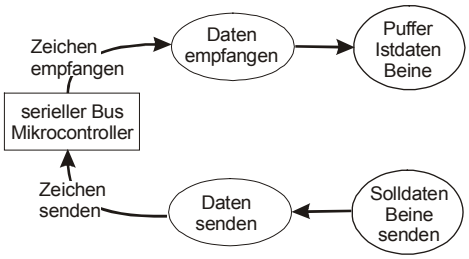



Abbildung 5.70: Abhängigkeitsgraph für die Kommunikation des Hauptcontrollers mit den Beincontrollern

Der Abhängigkeitsgraph der Kommunikation des Hauptcontrollers mit den Beincontrollern ist in Abbildung 5.70 dargestellt. Sie wird periodisch ausgeführt und ist zeitlich an den Interpolationstakt gebunden. Da die Beincontroller nur nach Datenanforderung senden, ist deshalb die Aktivität ihrer Kommunikationstasks ebenfalls vorhersagbar.

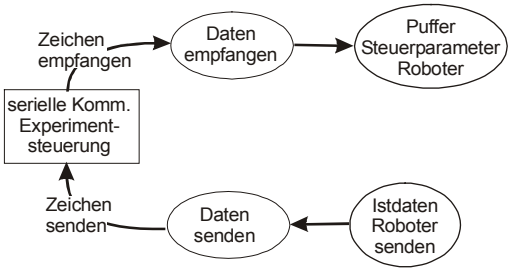



Abbildung 5.71: Abhängigkeitsgraph für die Kommunikation des Hauptcontrollers mit der externen Steuerung

Bei der Kommunikation mit der externen Steuerung, dargestellt in Abbildung 5.71, werden die Daten ebenfalls periodisch gesendet, um die aktuellen Sensorinformationen und die aktuellen Zustände zu beobachten. Der Empfang der Steuerdaten wird jedoch von der Experimentsteuerung initiiert. Es kann jedoch von einer festen und bekannten Periode ausgegangen werden, so dass auch hier die Aktivität der Empfangstask vorhersagbar ist.

### 5.4.3 Kommunikation

Wie bereits in vorhergehenden Abschnitten erwähnt, wird die Kommunikation mit dem Master-Slave-Prinzip realisiert. Das heißt, dass der Hauptcontroller Daten explizit anfordert. Damit wird ein determinierter Datenverkehr erzeugt und die Zeit für die Kommunikation kann modelliert werden.

Die im Laufroboter Katharina implementierte Kommunikation besteht aus den Phasen:

- Anforderung der Sensordaten der Controller und
- Senden der Steuerdaten an die Controller.

Die verwendeten Telegramme bestehen aus einem Telegrammkopf, der eine Telegrammkennung enthält, und einem Telegrammkörper. Die Telegrammkennung ist für jeden Telegrammtyp spezifisch. Dabei wird zwischen Anforderungstelegrammen und Datentele-

grammen unterschieden. Anforderungstelegramme enthalten als einzige Information die Kennung des Controllers, der abgefragt wird. Datentelegramme hingegen enthalten die zu übertragenden Sensor- und Steuerdaten.

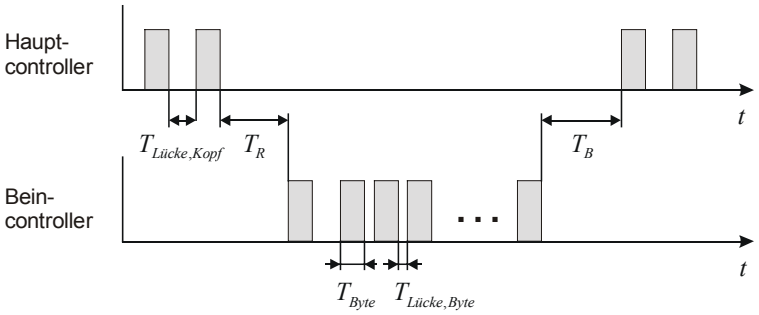



Abbildung 5.72: Zeiten bei der Datenübertragung

Die Übertragungsdauer $T_{Tele}$ für ein Telegramm kann mit

$$T_{Tele} = T_{Kopf} + T_{Daten} \tag{5.167}$$

bestimmt werden, wobei

$$T_{Daten} = n_{Daten} \cdot T_{Byte} + (n-1) \cdot T_{Lücke,Byte} \tag{5.168}$$

$$T_{Kopf} = T_{Byte} + T_{Lücke,Kopf} \tag{5.169}$$

gilt (siehe Abbildung 5.72). Dabei ist $T_{Kopf}$ die Übertragungszeit des Telegrammkopfes und einer folgenden Lücke, die den Controllern Zeit zur Auswertung des Telegrammkopfes gibt. Die darauffolgenden Daten werden kontinuierlich gesendet. Dabei folgt auf jedes übertragene Byte eine kleine Pause, um die Datenbytes voneinander zu trennen und die Entleerung des Empfangspuffers zu ermöglichen.

Für die Anforderung von Sensordaten muss neben der Zeit für die reine Datenübertragung auch die Reaktionszeit $T_R$ auf die Anfrage beachtet werden. Die benötigte Zeit für eine Anfrage mit Antwort ergibt sich zu

$$T_{Anfrage} = 2T_{Kopf} + (n_{Daten} - 1) \cdot (T_{Byte} + T_{Lücke,Byte}) + 2 \cdot T_{Byte} + T_R \ . \tag{5.170}$$

Damit sich zwei Telegramme durch Ausnutzung aller Toleranzen nicht überschneiden können, sollte nach einem Datentelegramm eine Beruhigungszeit $T_B$ eingeplant werden. Die Bedingung für eine Anfrage vorzusehende Zeit ist

$$T_{\max,Anfrage} \geq T_{Anfrage} + T_B \ . \tag{5.171}$$

Sie ist die garantierte Worst-Case-Zeit für die Anforderung der Sonsordaten.

Für das Senden der Stelldaten lässt sich ebenfalls eine Übertragungszeit formulieren. Um die Daten zu übertragen, sind zwei Varianten möglich. Das sind die Adressierung an einen spezifischen Controller und das Senden einer Broadcast-Nachricht. Da alle Controller sämtlichen Datenverkehr mithören können und die Übertragungen unbestätigt sind, ist aus Effizienzgründen das Senden einer Broadcast-Nachricht sinnvoll. Die maximal vorzusehende Zeit für

die Übertragung der Nachricht ergibt sich aus der Zeit für den Telegrammkopf, der Zeit für die Daten und der Beruhigungszeit

$$T_{\max,Stelldaten} \geq T_{Kopf} + n_{Daten} \cdot \left(T_{Byte} + T_{Lücke,Byte}\right) + T_B \,. \tag{5.172}$$

Sie ist ebenfalls eine garantierte Worst-Case-Zeit. Die Summe der Worst-Case-Zeiten für die Übertragung aller Daten darf die Periode des Interpolationstaktes nicht überschreiten. Die Startzeiten für die Datenübertragungen müssen entsprechend geplant werden.

### 5.4.4 Zeitverhalten

Für die Funktion aller Controller ist es wichtig, dass alle Tasks unter den gegebenen zeitlichen Randbedingungen planbar und ausführbar sind. Nur in diesem Fall ist es möglich, die Echtzeitfähigkeit des Steuerungssystems zu gewährleisten.

Für Haupt- und Beincontroller lassen sich die identifizierten Tasks in Haupttasks und ein- und ausgabeorientierte Tasks einordnen. Es ist jeweils eine Haupttask und eine Reihe von ein- und ausgabeorientierten Tasks vorhanden. Die Haupttask ist periodisch. Die ein- und ausgabeorientierten Tasks sind aus Sicht der jeweiligen Mikrocontroller nur zum Teil periodisch.

Für die Puls-Weiten-Modulation gilt, dass bei konstantem Tastverhältnis das Umschalten des Signals von logisch 0 auf 1 und umgekehrt vom Peripheral Transaction Server übernommen wird und somit keine CPU-Ressourcen benötigt. Bei einer Änderung des Tastverhältnisses wird stattdessen ein von der CPU zu verarbeitender Interrupt ausgelöst. Diese Task ist sequenziell an die Haupttask und zeitlich an den folgenden Umschaltvorgang von logisch 1 auf 0 gebunden und tritt somit mit der gleichen Periode wie die Haupttask auf. Bei den Kommunikationstasks ist eine Periodizität nicht primär gegeben. Bei Betrachtung des Gesamtsystems aus Haupt- und Beincontrollern lassen sich auch für die Kommunikation Zusammenhänge formulieren. Da die Kommunikation nach dem Master-Slave-Prinzip organisiert ist, wird sie vom Hauptcontroller initiiert. Sie ist zeitlich an die Periode der Haupttask des Hauptcontrollers gebunden, so dass diese gleichfalls für die Kommunikation der Beincontroller gilt. Für die Kommunikation des Hauptcontrollers mit der externen Steuerung sendet diese periodisch Nachrichten. Von der externen Steuerung wird angenommen, dass sie ebenfalls periodisch Nachrichten an den Hauptcontroller sendet. Somit können auch die Kommunikationstasks als periodische Tasks klassifiziert werden.

Für die Planung der Tasks und die Sicherstellung der zeitlichen Ausführbarkeit stehen verschiedene Methoden zur Verfügung.

Sind mehrere Tasks in einem System aktiv, so müssen ihnen die Betriebsmittel zur Erfüllung ihrer Aufgabe zugewiesen werden. Diese Zuteilung erfolgt von einer zentralen Instanz, dem Scheduler. Er ist dafür verantwortlich, dass Planung und Ausführung die rechtzeitige Fertigstellung der zu bearbeitenden Aufgabe gewährleisten. Tasks repräsentieren also die für den Scheduler sichtbaren Einheiten einer Applikation, welche die Objekte der Planung darstellen. Hiermit wird verdeutlicht, dass die Art und Weise des Scheduling von zentraler Bedeutung für die Bewertung des Verhaltens eines Systems ist. Eine wesentliche Forderung ist die Vorhersage der Einhaltung von zeitlichen Fristen. Damit kann die rechtzeitige Ausführung von Tasks garantiert werden. Dies erfordert eine Machbarkeitsanalyse. Schedulingverfahren wer-

den zunächst nach dem Wann der Machbarkeitsanalyse [Ramamritham, Stankovic 94], [Kaiser, Nett 98] und dann nach ihren Unterschieden hinsichtlich der daraus resultierenden Planung unterteilt.

Bei kalenderbasierten, statischen (offline) Schedulingverfahren wird die Machbarkeitsanalyse für die Menge der auszuführenden Tasks statisch, d.h. a priori durchgeführt. Im positiven Fall wird dabei ein Zeitplan erstellt, in dem exakt festgelegt wird, wann und wie lange Betriebsmittel für die Bearbeitung der Tasks zur Verfügung stehen. Diesem Verfahren liegt das Prinzip der Angemessenheit der Betriebsmittel zugrunde [Lawson 92]. Dies verlangt, dass den Tasks selbst im ungünstigsten Fall alle benötigten Betriebsmittel im erforderlichen Umfang zur Verfügung stehen. Bei diesem Verfahren steht die genaue Vorhersagbarkeit im Vordergrund, ungeachtet des zu betreibenden Aufwandes und der anfallenden Kosten. Außerdem muss eine deterministische Applikationsumgebung vorausgesetzt sein, d.h. alle relevanten Taskparameter, wie Ankunftszeiten, Startzeiten, Perioden und Fristen, müssen ebenfalls a priori bekannt sein.

Bei statischen, prioritätsbasierten Verfahren wird als Resultat der Analyse kein exakter Zeitplan erzeugt, sondern den Tasks werden Prioritäten zugewiesen. Zur Laufzeit sorgt dann der Dispatcher des Betriebssystems dafür, dass zu jedem Zeitpunkt die Tasks mit der höchsten Priorität ausgeführt werden. Bei Einprozessorsystemen kann bei geeigneten Randbedingungen, entsprechender Wahl des Prioritätskriteriums und bei Nichtüberschreitung einer entsprechenden Schranke für die Auslastung des Prozessors die fristgerechte Bearbeitung aller Tasks garantiert werden [Liu, Layland 73].

Einige grundlegende Verfahren sind beispielsweise das Rate-Monotonic Scheduling oder das Earliest-Deadline-Scheduling [Hüsenner 94]. Bei diesen Schedulingverfahren wird davon ausgegangen, dass Tasks periodisch abzuarbeiten sind. Beim Earliest-Deadline-Scheduling (EDF-Scheduling) werden die Taskprioritäten entsprechend der verbleibenden Frist vergeben, in der die Tasks bis zum Ablauf ihrer Periode abzuarbeiten sind. Die Prioritäten werden also dynamisch vergeben. Beim Rate-Monotonic-Scheduling werden die Prioritäten in Abhängigkeit der Periodenzeiten vergeben. Die Task mit der kürzesten Periode erhält die höchste Priorität. Die Prioritäten werden also statisch zugeordnet. Beide Varianten wurden auch für nichtperiodische Tasks erweitert, allerdings ohne garantierte Bearbeitungszeit [Hüsenner 94]. Diese Schedulingalgorithmen setzen ein genaues a priori - Wissen über die zu erwartenden Taskparameter voraus. Im Überlastfall reagieren beispielsweise EDF-Scheduler sehr empfindlich. Dabei tritt ein sogenannter Dominoeffekt auf, bei dem die Überschreitung der Frist einer Task zu einer Fristverletzung etlicher weiterer Tasks führt [Stankovic et al. 98], [Liu 00].

Beim Steuerungssystem des Laufroboters Katharina können Perioden für verschiedene Tasks identifiziert werden. Jedoch liegen Bedingungen vor, die die Nutzung eines online Schedulers im konkreten Fall nicht sinnvoll erscheinen lassen.

Bei allen Controllern kann zwischen der Haupttask und den durch Interrupt aktivierten Tasks unterschieden werden. Bei einer durch einen von der Hardware ausgelösten Interrupt aktivierten Task liegt eine feste Priorisierung vor, die durch den Mikrocontroller gegeben ist. Außerdem ist es wichtig, dass diese Interrupts innerhalb einer bestimmten Latenzzeit reagieren, d.h. ihre Fristen sind nicht an eine Periode gekoppelt. Durch einen zusätzlich aktivierten Scheduler würde die Latenzzeit zudem stark vergrößert werden.

Für die Gelenkregelung ist es wichtig, dass sie periodisch ausgeführt wird. Weiterhin soll zwischen dem Einlesen der Sensordaten und der Ausgabe der Stellinformation eine möglichst geringe Zeit vergehen, um keine zusätzliche Totzeit in das Regelungssystem einzubringen. Es ist also wichtig, dass der Regler periodisch zu den Abtastzeitpunkten und mit relativ kurzer oder zumindest bekannter Verarbeitungszeit arbeitet. Werden beispielsweise Regler und aktive Nachgiebigkeit bei gleicher Periode einem Scheduling unterworfen, das nur die Periode auswertet, so ist nicht klar, in welcher Reihenfolge Regler und aktive Nachgiebigkeit bearbeitet werden. Deshalb ist es im vorliegenden Fall sinnvoll, beide Tasks in der Haupttask zusammenzufassen, so dass eine definierte Bearbeitungsreihenfolge entsteht und die oben genannten Anforderungen erfüllt werden. Damit existiert nur die Haupttask mit einer festen Reihenfolge zu bearbeitender Subtasks. Die Aktivierung der Interrupt-Serviceroutinen erfolgt direkt durch den Prozessor in Abhängigkeit von Ereignissen, so dass ein gesonderter Mechanismus nicht vorgesehen werden muss. In diesem Fall ist die geeignete Lösung ein kalenderbasiertes Verfahren. Dabei wird die Haupttask periodisch aktiviert, die durch die Bearbeitung der Subtasks in der vorgesehenen Reihenfolge die Anforderungen an die Fristen realisiert. Bei dieser Lösung wird ein Online-Scheduler nicht benötigt, dessen Aufwand verglichen mit dem Nutzen in diesem Fall nicht gerechtfertigt scheint.

Um die Ausführbarkeit aller Tasks im vorgegebenen Zeitrahmen zu garantieren, muss für das gewählte Scheduling eine Machbarkeitsanalyse durchgeführt werden. Darin werden sowohl die Haupttask (bzw. die Subtasks) als auch alle Interrupt-Serviceroutinen einbezogen. Dazu müssen deren Perioden $T$ und Ausführungszeiten $C$ bekannt sein. Zur Prüfung der Ausführbarkeit aller Tasks wird der ungünstigste Fall konstruiert. Dieser tritt ein, wenn die Haupttask, die die niedrigste Priorität besitzt, durch sämtliche höher priorisierten Tasks, in diesem Fall alle Interrupt-Serviceroutinen, unterbrochen wird. Der Prozessorauslastungsfaktor $U$ kann mit Hilfe von

$$U = \sum_{i=1}^{n} \frac{C_i^{t_1,t_2}}{\Delta t}. \tag{5.173}$$

bestimmt werden [Hüsenner 94], [Zöbel, Albrecht 95]. Dabei wird für $\Delta t$ die Periode der Haupttask angesetzt. Es werden alle Tasks betrachtet, die maximal während einer Periode gestartet werden können. Ist die Periode größer als die der Haupttask, so wird von einer einmaligen Aktivierung ausgegangen, ansonsten von der Anzahl der innerhalb einer Periode der Haupttask möglichen Aktivierungen.

*Beincontroller*

Die Haupttask eines Beincontrollers kann maximal durch Interrupts zur Neueinstellung der Puls-Weiten-Modulation, einer Kommunikationstask bei Eintreffen eines Datentelegramms und von Anforderungstelegrammen sowie für das Senden der Sensordaten hintereinander unterbrochen werden. Die für das Ausführen des Cyclic Redundancy Check über die vom Hauptcontroller empfangenen Daten benötigte Zeit wird einzeln berechnet, um den Einfluss der Kommunikation auf die Prozessorauslastung zu betrachten.

Die Bedingung für die Ausführbarkeit aller Tasks ist

$$U_{Bein.\max} = \frac{1}{T_{P,Haupt,Bein}} \cdot \left(C_{Haupt} + 3 \cdot C_{PWM} + 4 \cdot C_{Empf.,2a} + C_{Empf.,2b} + C_{Send}\right) \leq 100\% \tag{5.174}$$

Bei der Haupttask mit einer Dauer von $C_{Haupt}$ = 7040 µs, Tasks zur Puls-Weiten-Modulation mit jeweils $C_{PWM}$ = 15 µs, einer Datensendetask mit $C_{Send}$ = 950 µs und Zeiten für den Datenempfang in der Schicht 2a mit $C_{Empf.,2a}$ = 120 µs und in der Schicht 2b $C_{Empf.,2b}$ = 1160 µs beträgt die Prozessorauslastung

$$U_{Bein,\max} = \frac{7040\,\mu s + 3 \cdot 15\,\mu s + 4 \cdot 120\,\mu s + 1160\,\mu s + 950\,\mu s}{10\,ms} = 96{,}75\% \ . \qquad (5.175)$$

Eine minimale Auslastung tritt auf, wenn keine Kommunikation stattfindet, d.h. vom Hauptcontroller keine Kommunikationsaktivität erfolgt. In diesem Fall ist die Auslastung

$$U_{Bein,\max} = \frac{7040\,\mu s + 3 \cdot 15\,\mu s}{10\,ms} = 70{,}85\% \ . \qquad (5.176)$$

Für die angegebenen Zeiten liegen Berechnungen der Ausführungszeiten für die Interrupt-Serviceroutinen und Messungen mit Hilfe des Logikanalysators PA600 [NCI 99] für die übrigen Laufzeiten zugrunde. Der verwendete Prozessor besitz keine Caches und alle Zeiten für Speicherzugriffe können wegen der statischen Speichernutzung als invariant angesehen werden. Aus diesem Grund können diese ermittelten Zeiten als maximale Ausführungszeiten zugrundegelegt werden.

*Hauptcontroller*

Die Haupttask des Hauptcontrollers kann mittels Interrupts durch eine Kommunikationstask bei Eintreffen eines Datentelegramms, zum Senden der Anforderungstelegramme und zum Senden bzw. Empfangen von Daten der externen Steuerung unterbrochen werden. Da die Kommunikation mit den Beincontrollern zeitlich an die Haupttask gebunden ist, besteht hier ein definierter Zusammenhang. Das Senden von Daten an die externe Steuerung erfolgt mit der Periode der Haupttask. Die externe Steuerung sendet ihrerseits periodisch alle 55 ms Daten, so dass die entsprechende Empfangstask maximal einmal pro Periode der Haupttask aktiv wird.

Die Bedingung für die Ausführbarkeit aller Tasks ist

$$U_{Haupt,\max} = \frac{1}{T_{P,Haupt,Haupt}} \cdot \begin{pmatrix} C_{Haupt} + 6 \cdot C_{Send,Anford.} + 6 \cdot C_{Empf,Anford.,2a} \\ + C_{Send,Daten} + C_{Send,ext} + C_{Empf,ext} \end{pmatrix} . \qquad (5.177)$$

Die maximale Prozessorauslastung bei einer Haupttask von $C_{Haupt}$ = 31,5 ms, dem Senden von sechs Anforderungstelegrammen mit $C_{Send,Anford.}$ = 94 µs, dem Empfangen der Daten von den Beincontrollern mit $C_{Empf,Anford.,2a}$ = 120 µs, dem Senden der Broadcastnachricht an die Beincontroller mit $C_{Send,Daten}$ = 2152 µs, dem Senden einer Nachricht an die externe Steuerung mit $C_{Send,ext}$ = 21 µs und dem Empfangen von Daten von der externen Steuerung mit $C_{Empf,ext}$ = 434 µs beträgt

$$U_{Haupt,\max} = \frac{1}{40\,ms} \cdot \begin{pmatrix} 31500\,\mu s + 6 \cdot 94\,\mu s + 6 \cdot 120\,\mu s + \\ 2152\,\mu s + 21\,\mu s + 434\,\mu s \end{pmatrix} = 88{,}48\% \ . \qquad (5.178)$$

Die Zeit zum Ausführen des Cyclic Reduncancy Check über die Datentelegramme von den Beincontrollern ist in der Zeit für den Hauptprozess enthalten.

Abbildung 5.73 zeigt ein Beispiel zur zeitlichen Abfolge von Steuerungszyklen und Kommunikation, die mit Hilfe des Logikanalysators PA600 aufgezeichnet wurde. Im H_Period sind die Startzeitpunkte der Haupttask dargestellt und kennzeichnen deren Periode, die identisch mit dem Interpolationstakt für die Trajektoriengenerierung ist. B_Period zeigt die Periode der Haupttask eines Beincontrollers, B_MainTsk zeigt die Zeit für die vollständige Bearbeitung der Haupttask des Beincontrollers. In H_Comm und B_Comm ist die Datenübertragung des Hauptcontrollers an die Beincontroller bzw. der Beincontroller an den Hauptcontroller dargestellt. Die schwarzen Rechtecke kennzeichnen übertragene Daten.

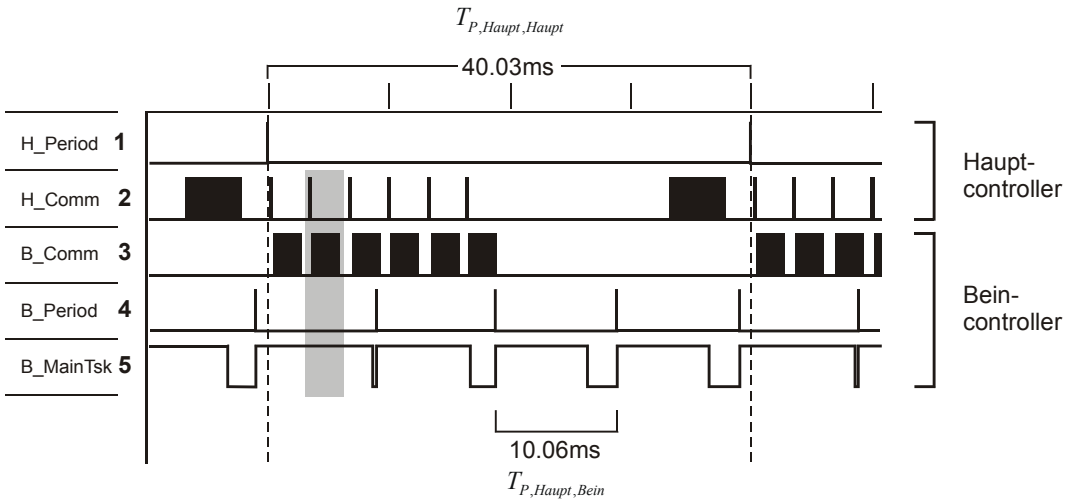



Abbildung 5.73: Zeitverhalten des Steuerungssystems

Aus der Abbildung 5.73 ist aus H_Period und B_Period zu erkennen, dass die Periode der Haupttask des Hauptcontrollers nicht mit der Periode der Haupttask des Beincontrollers synchronisiert ist. H_Period, H_Comm und B_Comm zeigen, dass die vom Hauptcontroller koordinierte Kommunikation nach dem Master-Slave-Prinzip mit der Periode seiner Haupttask synchronisiert ist. Der Hauptcontroller fragt Daten von den Beincontrollern ab, die diese dann senden. Das zeigen die sechs Telegrammfolgen in H_Comm und B_Comm. Mit Hilfe eines Broadcasttelegramms sendet der Hauptcontroller seine Steuerdaten an die Beincontroller, was am großen Telegramm in H_Comm zu erkennen ist.

Trifft ein Anforderungstelegramm beim Beincontroller ein, so wird für dessen Empfang, Auswertung und das Zurücksenden der angeforderten Daten zusätzliche Rechenzeit benötigt. Weitere Rechenzeit wird für das Prüfen eines vorher empfangenen Datentelegramms vom Hauptcontroller benötigt. Das Zusammentreffen dieser Anforderungen kennzeichnet den ungünstigsten Fall, der in Abbildung 5.73 dargestellt ist. Die Datenanfrage des Hauptcontrollers an den Beincontroller ist durch den grauen Bereich markiert. Für Datenanfrage und das Prüfen des vorher empfangenen Datentelegramms wird zusätzliche Rechenzeit benötigt, so dass sich die Bearbeitung der Haupttask verzögert. B_MainTsk zeigt dies anhand der Start- und der Endzeit für die Haupttask, deren Abstand sich während der Anfrage und der Prüfung der vorher empfangenen Steuerdaten vergrößert, jedoch die Periode nicht erreicht.

Es wurde gezeigt, dass das gewählte Verfahren für die Planung der Tasks und der Zuteilung der Ressourcen für Betriebsmittel wie CPU und Kommunikationsbus zu einem vorhersagba-

ren Verhalten führt. Die daraus ableitbare Garantie zur Ausführung aller Tasks ist Grundlage für die Funktion und Stabilität aller Steuerungs- und Regelungsalgorithmen, die Echtzeitanforderungen erfüllen müssen.

### 5.4.5 Implementierung

Bei der Implementierung spielt insbesondere der Aspekt der Bereitstellung von Funktionen eine Rolle, die den Echtzeitbedingungen gerecht werden. In diesem Abschnitt wird auf Fragen der Implementierung eingegangen.

#### 5.4.5.1 Implementierung von Ein- und Ausgabefunktionen

Die Implementierung der Ein- und Ausgabeoperationen ist stets sehr spezifisch an die verwendete Hardware angepasst. Diese Operationen stellen eine abstrakte Schnittstelle für Peripherieoperationen bereit. Beim Laufroboter Katharina sind Ein- und Ausgabefunktionen für die Analog-Digital-Wandlung, die serielle Kommunikation und das Ausgeben der Puls-Weiten-modulierten Signale notwendig. Wegen der Prozessornähe wurden diese zum großen Teil in Assembler mit Hilfe des Makro-Assemblers für die Intel 8096-Familie realisiert [Intel 90].

Das Einlesen der Sensoren erfolgt mit Hilfe des integrierten Analog-Digital-Umsetzers. Da mehrere Kanäle eingelesen werden müssen, muss der Analog-Digital-Wandler und der ebenfalls in den Mikrocontroller integrierte Multiplexer für diese Operationen programmiert werden. Dies wird durch den Peripheral Transaction Server des 80C196 unterstützt, so dass das Einstellen des Multiplexerkanals, das Starten des Analog-Digital-Umsetzers sowie das Auslesen des gewandelten Wertes und das Speichern aller Ergebnisse im RAM nebenläufig bearbeitet werden. Nach Beendigung der kompletten Sensorabfrage stehen sämtliche Ergebnisse im RAM zur weiteren Verarbeitung zur Verfügung. Der Abschluss der Transferoperationen wird durch einen Interrupt signalisiert, der für das Setzen einer Semaphore genutzt wird.

Die Generierung der Puls-Weiten-modulierten Signale erfolgt wie bereits beschrieben mit Hilfe des Event Processor Arrays. Dabei wird ein Timer mit dem Inhalt eines Registers verglichen, das den Umschaltzeitpunkt für das Signal beinhaltet. Das Event Processor Array gibt die Signalveränderung praktisch verzögerungsfrei aus. Das Neuprogrammieren der Vergleichsregister zum Ausgeben von logisch 0 bzw. logisch 1 erfolgt mittels Peripheral Transaction Server, der dazu das Addieren einer Periodendauer übernimmt. Um das Tastverhältnis zu ändern, muss der Zeitpunkt zur Ausgabe von logisch 0 gegenüber dem Zeitpunkt zur Ausgabe von logisch 1 geändert werden. Dazu wird mit Hilfe eines Interrupts, der anstelle des Peripheral Transaction Servers aktiviert wird, der folgende neue Zeitpunkt zum Umschalten auf logisch 0 in das Vergleichsregister geschrieben. Da sich mit Veränderung der Puls-Weiten-Modulation auch die Drehrichtung ändern kann, wird das Drehrichtungsbit ebenfalls gesetzt oder rückgesetzt. Die Aktivitäten des Peripheral Transaction Servers benötigen außer den notwendigen Buszugriffen zum Schreiben und Lesen keine CPU-Zeit. Lediglich das Ausführen des Interrupts benötigt CPU-Zeit, die nur in Zusammenhang mit dem Einstellen eines neuen Tastverhältnisses in Erscheinung tritt.

Für die Kommunikation ist sowohl beim Haupt- und als auch beim Beincontroller die Schicht 2a, d. h. der Medienzugang, mit Hilfe von Interrupts implementiert, die durch die SSIO (Synchronous Serial Input Output Unit des 80196KR) initiiert werden. Es ist so möglich, sowohl das Senden als auch das Empfangen von Daten ereignisgesteuert und nebenläufig zu implementieren. Dabei wird nach jedem empfangenen bzw. gesendeten Byte ein Interrupt ausgelöst, der das Senderegister neu beschreibt bzw. das Empfangsregister auslesen kann. Ferner erfolgt eine Auswertung des Telegrammkopfes, um die Anzahl der noch zu empfangenen Bytes zu bestimmen und bei einer Datenanforderung für die Aktivierung der Sendetask zu sorgen. Die Synchronisation zwischen Sender und Empfänger kommt dadurch zustande, dass auf einen gültigen Telegrammkopf gewartet wird.

Die Schicht 2b wird innerhalb der Haupttask realisiert, wenn die Daten zur Übertragung in den Puffer geschrieben bzw. wenn sie aus dem Puffer gelesen werden. Zur Prüfung auf Übertragungsfehler wird ein CRC [Tanenbaum 96] mit einem Prüfpolynom nach CRC-CCITT eingesetzt. Dazu wurde eine Implementierung aus [Stout 01] genutzt.

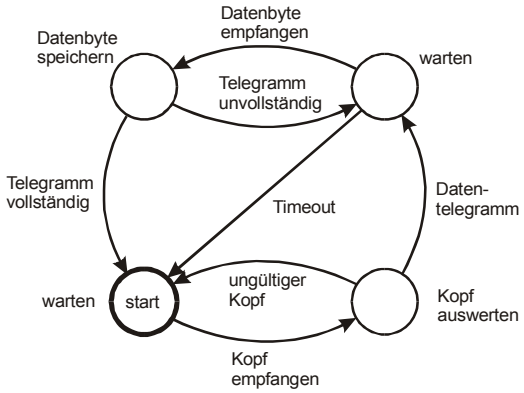



Abbildung 5.74: Zustandsgraph zum Empfangen von Datentelegrammen

Abbildung 5.74 zeigt den zugehörigen Zustandsgraphen zum Empfangen von Datentelegrammen. Der dick markierte Zustand ist der Startzustand. Bei einem empfangenen Byte wird geprüft, ob es sich um einen gültigen Telegrammkopf handelt. Falls nicht, wird in den Anfangszustand zurückgewechselt. Liegt ein gültiger Telegrammkopf vor, kann die zugehörige Anzahl von Datenbytes empfangen werden. Dazu wird auf das folgende Telegrammbyte gewartet, das dann in einem Puffer abgespeichert wird. Dies wird wiederholt, bis alle Daten empfangen worden sind und dann in den Anfangszustand zurückgewechselt. Wird dabei jedoch eine Zeitschranke überschritten, so wird vorzeitig in den Anfangszustand zurückgewechselt.

Bei der Implementierung des Master-Slave-Protokolls muss zusätzlich zum Datenempfang beim Slave bei Vorliegen eines Anforderungstelegramms das Senden eines Datentelegramm an den Master initiiert werden. Um eine kurze Reaktionszeit zu gewährleisten, initiiert die Empfangstask direkt die Sendetask für das Senden der vorbereiteten Daten an den Hauptcontroller. Dazu wird der Zustandsgraph für das Auswerten der Anforderungstelegramme erweitert. Abbildung 5.75 zeigt den erweiterten Zustandsgraphen. Trifft ein Anforderungstelegramm ein, so wird dies beim Prüfen des Telegrammkopfes erkannt und es wird das folgende Datenbyte als Adresse des Controllers für die angeforderten Daten interpretiert. Ist diese Adresse identisch mit der eigenen, so wird das Senden eines Datentelegramms initiiert. Danach wird wieder in den Startzustand zurückgekehrt.

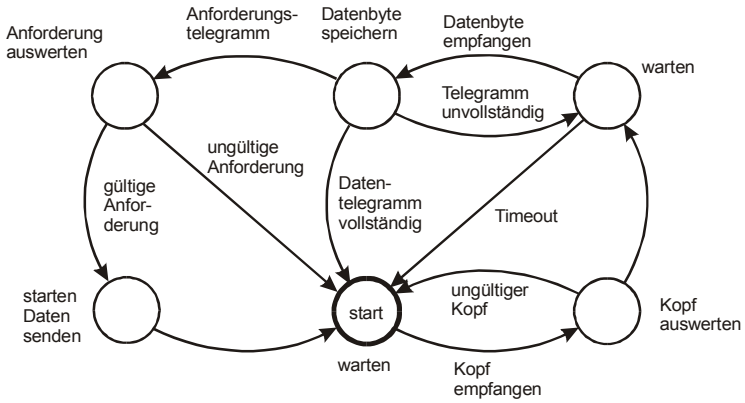



Abbildung 5.75: Erweiterter Zustandsgraph für Anforderungstelegramme

Der Zustandsgraph zum Senden eines Telegramms ist in Abbildung 5.76 dargestellt, welcher für Haupt- und Beincontroller gilt. Zunächst wird ein Telegrammkopf gesendet, gefolgt von einer Pause. Danach wird jeweils ein Byte, gefolgt von einer Pause gesendet. Ist das gesamte Telegramm übertragen, so wird wieder in den Ausgangszustand zurückgekehrt.

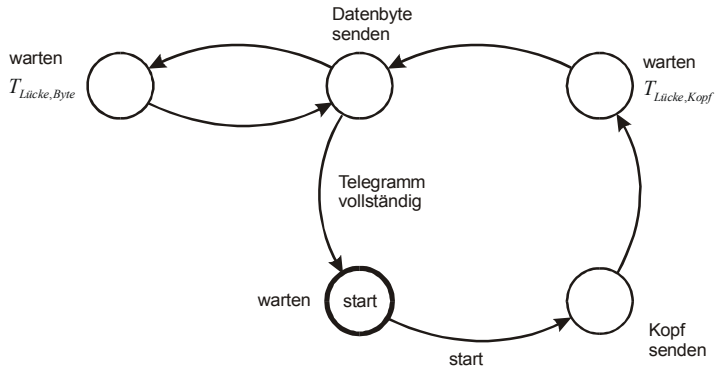



Abbildung 5.76: Zustandsgraph zum Senden eines Telegramms

### 5.4.5.2 Implementierung der Steuerungsalgorithmen

Die Steuerungsalgorithmen müssen in einer vorgegebenen Zeit die benötigten Ergebnisse liefern. Aufgrund der begrenzten Rechenleistung in einem eingebetteten System müssen, um dies zu erreichen, Kompromisse zugunsten der Reaktionsgeschwindigkeit eingegangen werden. Kompromisse sind beispielsweise bei der Implementierung der arithmetischen Funktionen und der Modularität der Rechenalgorithmen notwendig. Für die arithmetischen Operationen können entweder die Gleitkommaarithmetik oder die Festkommaarithmetik genutzt werden. Der Vorteil der Gleitkommaarithmetik besteht in der einfachen Implementierbarkeit der arithmetischen Operationen. Der große Nachteil besteht in der benötigten Rechenzeit, falls keine Gleitkommaunterstützung im Prozessor vorhanden ist und die Funktionen durch eine Softwarebibliothek bereitgestellt werden. Bei den meisten Mikrocontrollern ist keine Gleitkommarecheneinheit vorhanden. Bei der Implementierung der Festkommaarithmetik besteht generell das Problem der Skalierung. Es muss abgeschätzt werden, in welchem Bereich die zu erwartenden Ergebnisse liegen, damit die Ergebnisse darstellbar sind und eine ausreichende Genauigkeit besitzen. Die Grundrechenarten werden in der Regel direkt durch den Prozessor unterstützt. Rechnungen mit doppelter Genauigkeit, zum Beispiel mit 32 Bit,

werden entweder direkt durch den Prozessor oder durch Compilerbibliotheken unterstützt. Besondere Probleme treten auf, wenn beispielsweise transzendente Funktionen berechnet werden müssen.

Durch Vermeidung doppelter Berechnungen kann eine Beschleunigung der arithmetischen Operationen erfolgen. Zum Beispiel sind bei der Berechnung der direkten Kinematik Transformation für die drei Koordinaten entsprechend der Gleichung (5.81) zu implementieren. In dieser Gleichung werden mehrfach die gleichen trigonometrischen Funktionen genutzt. Diese werden ebenfalls in Gleichung (5.106) für die Krafttransformation verwandt. Es liegt nahe, identische Teiloperationen zusammenzufassen und die gesamte Berechnung dadurch zu beschleunigen. Der Preis dafür ist, dass beide Transformationen nicht mehr getrennt behandelt werden können, so dass eine einzige Transformationsfunktion entsteht. Eine weitere Methode zur Beschleunigung der arithmetischen Operationen ist, die Genauigkeit zu reduzieren.

Ein weiterer Weg zur Reduzierung der Rechenzeit besteht darin, Funktionswerte in einer Tabelle abzulegen. Die Funktion wird dabei auf Stützstellen reduziert und Zwischenwerte gegebenenfalls interpoliert.

Beim Laufroboter Katharina wurden die arithmetischen Operationen in der ersten Version mit Hilfe einer Floating Point Bibliothek des iC-96-Compilers [Intel 91a], [Intel 91b] realisiert. Für die Haupttasks von Bein- und Hauptcontroller ergaben sich Rechenzeiten von bis zu 100 ms. Um die Rechenzeit für die Transformationen zu verringern, wurde in einer zweiten Version eine Festkommaarithmetik implementiert. Zusätzlich wurden die trigonometrischen Funktionen mit Hilfe von Tabellenfunktionen realisiert.

*Realisierung der trigonometrischen Funktionen mittels Tabellenfunktionen*

Die Realisierung von trigonometrischen Funktionen mittels Tabellen benötigt wenig Rechenzeit, aber sehr viel Speicherplatz. Eine Tabelle benötigt pro Stützstelle ein korrespondierendes Wertepaar. Bei der Implementierung einer Tabelle, die das Wertepaar explizit enthält, werden

$$n_{Byte} = 2 \cdot w \cdot n_{Tab} \tag{5.179}$$

Bytes benötigt. Dabei ist $w$ die Wortbreite in Bytes und $n_{Tab}$ die Anzahl der Tabelleneinträge. Für eine Tabelle mit 2 Byte langen Wörtern und 2048 Tabelleneinträgen für Argument und Funktionswert würden 8 kByte benötigt. Zum Suchen eines Funktionswertes muss zunächst ein Argumentwert in der Tabelle gefunden werden, um den korrespondierenden Funktionswert zu erhalten. Das Verfahren funktioniert auch mit der Umkehrfunktion, falls diese streng monoton ist und sich eine eineindeutige Abbildung aller Wertepaare ergibt. Zwischenwerte können mit Hilfe einer Interpolation ermittelt werden.

Um das Suchen des Argumentwertes zu beschleunigen, sollte dieser direkt aus dem Funktionsargument hervorgehen. Dazu werden die Argumentwerte der Tabelle äquidistant gewählt. Die Tabelle enthält einen minimalen Argumentwert $x_{Tab,\min}$ und einen maximalen Argumentwert $x_{Tab,\max}$. Die Auflösung bzw. die Distanz zwischen den Argumentwerten wird mit

$$\Delta x_{Tab} = \frac{x_{Tab,\max} - x_{Tab,\min}}{n_{Tab} - 1} \tag{5.180}$$

bestimmt. Damit lässt sich aus dem Argumentwert der Tabellenindex $x_{Tab}$ nach der Funktion

$$x_{Tab} = \text{int}\left( \frac{x_{Arg} - x_{\min}}{\Delta x_{Tab}} + 0{,}5 \right) \tag{5.181}$$

bestimmen.

Falls eine Implementierung auf Integerarithmetik erfolgt, kann Gleichung (5.181) unter Einhaltung weiterer Bedingungen vereinfacht werden. Ist die Auflösung der Zahlendarstellung identisch mit der Auflösung der Tabelle, kann die Skalierung entfallen und Gleichung (5.181) vereinfacht sich zu

$$x_{Tab} = x_{Arg} - x_{\min} \, . \tag{5.182}$$

Bei trigonometrischen Funktionen lassen sich Symmetrien ausnutzen, so dass stets ein mit Null beginnender Argumentbereich gefunden werden kann. Es gilt

$$x_{\min} = 0 \text{ und } x_{Tab} = x_{Arg} \, . \tag{5.183}$$

Damit entspricht der Argumentwert genau dem Tabellenindex. Der Funktionswert kann direkt mittels eines indizierten Tabellenzugriffs

```
y_Tab = Funk_Tab(x_Arg)
```

gefunden werden.

Beim Laufroboter Katharina besitzen die Gelenkwinkelsensoren eine Auflösung von 10 Bit für einen Bereich von ca. 330°. Die Gelenkwinkel werden auf einen Wertebereich von ±4096 abgebildet. Durch zyklische und symmetrische Eigenschaften der Sinusfunktion lässt sich dieser auf einen Argumentbereich von 0 .. ½ π reduzieren. Die Implementierung erfolgt durch folgende Anweisungen:

```
α := α mod 2π
wenn α < 0
       dann   wenn α > -π/2   dann   α := -π - α
              y_Tab = -sin_Tab(-α)
       sonst  wenn α > π/2    dann   α := π/2 - α
              y_Tab = sin_Tab(α)
```

Der Kosinus lässt sich mit $\cos(\alpha) = \sin(\alpha + \pi/2)$ auf den Sinus zurückführen.

Für die Umkehrfunktionen lässt sich eine weitere Tabelle nutzen. Die Tabelle enthält nur Werte in einem monotonen Bereich der Umkehrfunktion, so dass sich eine eindeutige Abbildung ergibt. Die Implementierung erfolgt durch:

```
wenn y < 0
       dann   α := -asin_Tab(-y)
       sonst  α := asin_Tab(y)
```

Die Umkehrfunktion des Kosinus lässt sich durch Phasenverschiebung der Ergebnisse um den Winkel $\pi/2$ realisieren.

Die Tangensfunktion hat eine Periode von $\pi$. Durch Punktsymmetrie um Null lässt sich der Argumentbereich auf $0..\pi/2$ reduzieren. In diesem Bereich ist die Funktion streng monoton steigend. In den Gleichungen (5.85) und (5.92) wird der Arcustangens eines Quotienten benötigt. Daher ist sicherzustellen, dass der Dividend nicht zu Null wird, da dann der Quotient gegen unendlich strebt und damit der Arcustangens nicht bestimmbar ist. Die Arcustangensfunktion kann mit Hilfe von zwei Argumenten realisiert werden. Dadurch lassen sich Fälle für alle vier Quadranten, also in einem Winkelbereich von $-\pi..+\pi$, unterscheiden. Der Argumentbereich kann auf den ersten Quadranten zurückgeführt werden, da die Zuordnung zu Quadranten mittels Vorzeichenauswertung erfolgt. Da

$$\tan(\alpha) = \cot\left(\frac{\pi}{2} - \alpha\right) = \left[\tan\left(\frac{\pi}{2} - \alpha\right)\right]^{-1} . \tag{5.184}$$

gilt, lässt sich der Argumentbereich weiter auf Werte im ersten Oktanten reduzieren, indem ein reziprokes Argument gebildet wird, wobei bei der Zuordnung der Argumentwerte unterschieden wird

$$\alpha = \begin{cases} \arctan(y/z) & \text{falls} \quad y \le z \\ \pi/2 - \arctan(z/y) & \text{falls} \quad y > z \end{cases} . \tag{5.185}$$

Damit ist sichergestellt, dass die Arcustangensfunktion stets bestimmbar ist. Der Arcustangens kann dann über eine Tabellenabfrage ermittelt werden, indem die Relation der Argumente mit

```
wenn y ≤ z
       dann   α := atan2_Tab(y/z)
       sonst  α := π/2 - atan2_Tab(z/y)
```

ausgewertet werden und in Abhängigkeit davon der Funktionswert ermittelt wird. Der Quotient wird dabei auf die Anzahl der Tabelleneinträge skaliert.

Alle vorgestellten Algorithmen zur Berechnung der trigonometrischen Funktionen besitzen den Aufwand O(1) und ist konstant.

*Quadratwurzel*

Zur Berechnung von Quadratwurzeln, die z.B. in Gleichung (5.96) vorkommen, wurde ein Algorithmus zur Nährungslösung auf der Basis von binomischen Formeln implementiert. Speziell wurde ein Algorithmus zum Berechnen einer Quadratwurzel aus einem 32 Bit un-

signed long integer implementiert, der auf einem im Snippets-Archiv veröffentlichten Algorithmus [Stout 01] basiert. Eine ähnliche Lösung wurde auch in [Turkowski 94] vorgestellt. Die Implementierung sieht wie folgt aus:

```
Ergebnisbits = 16
Rest = 0
Akkumulator = 0

Wiederhole für Anzahl der Ergebnisbits
       Verschiebe Rest 2 Bit nach links
       kopiere die beiden höchstwertigen Bits des Radikanden an die unterste
         beiden Bits des Restes
       Verschiebe Akkumulator 1 Bit nach links
       Verschiebe Radikand um 2 Bit nach links
       Testprodukt := (Verschiebe Akkumulator 1 Bit nach links) +1
       Wenn (Rest ≥ Testprodukt)
              dann   subtrahiere Testprodukt vom Rest
                     erhöhe Akkumulator um 1
```

Bei jedem Schleifendurchlauf wird eine Stelle des Ergebnisses im Akkumulator geschätzt. Dieser kann entweder Eins oder Null annehmen, so dass nach jeder Schätzung ein neues Ergebnisbit vorliegt.

Da die Quadratwurzel ohne Nachkommastellen maximal die Hälfte der Stellen des Radikanden besitzt, liegt bei einem 32 Bit Integeroperanden nach 16 Durchläufen ein Ergebnis vor. Der Aufwand für diesen Algorithmus ist von der Ordnung O(n), wobei n von der Anzahl der gewünschten Stellen des Ergebnisses abhängig ist. Da der Radikand stets die gleiche Anzahl von Stellen hat, kann für den implementierten Algorithmus eine maximale Rechenzeit angegeben werden.

# 6 Technische Operationen

Auf die Prinzipien und Methoden zur Steuerung des Roboters wurde in den vorangegangenen Kapiteln eingegangen. Im nachfolgenden Kapitel soll an Beispielen gezeigt werden, wie mit den vorgestellten Steuerungsalgorithmen technische Operationen realisiert werden können.

## 6.1 Interaktion mit der Umgebung

Ein Laufroboter kann neben der eigentlichen Fortbewegung mittels seiner Beine auch Operationen in seiner Umgebung ausführen. Hierbei ist es von Interesse, wie und zu welchem Zweck Kräfte auf die externen Objekte eingebracht werden sollen.

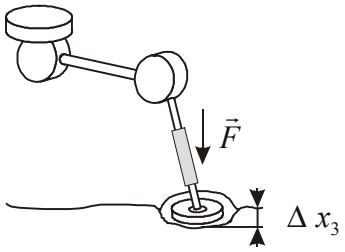



Abbildung 6.1: Test der Bodeneigenschaften

Ein Beispiel zur Interaktion ist eng mit dem Laufen selbst verbunden. Um die Trittfestigkeit des Bodens zu testen, wird ein Bein, das die Tragfähigkeit und die Nachgiebigkeit des Bodens testen soll, als Bevameter genutzt. Wie in Abbildung 6.1 dargestellt, wird eine Kraft auf den Untergrund eingebracht und die Verschiebung des Fußes registriert. Dabei kann die Kraft größer gewählt werden, als sie später für das Stützen des Roboters notwendig ist, da die Gewichtskraft des Roboters auf mehrere Beine verteilt werden kann. Aus dem zeitlichen Verlauf von Kraft und Fußposition wird dann der Zusammenhang zwischen beiden Größen analysiert. Bei einem weichen Untergrund ist die Einsinktiefe größer als bei einem harten Untergrund. Falls sich der Untergrund elastisch verformt, so werden die Wertepaare von Kraft und Verschiebung des Fußes auf einer Geraden liegen. Entsteht jedoch eine plastische Verformung, so werden die Wertepaare von Kraft und Verschiebung nicht auf dieser Geraden liegen, sondern es wird eine Differenz in der Position zwischen erstem Bodenkontakt und letzten Bodenkontakt bleiben. Das Prinzip der Untersuchung der Bodeneigenschaften mit Hilfe der Beine wurde bereits in [Sinha, Bajcsy 92] und [Gorinevsky, Schneider 90] beschrieben.

Die Kenntnis über die Bodeneigenschaften lässt sich dazu nutzen, die Standstabilität besser einzuschätzen und zu verbessern. Beispielsweise kann an besonders nachgiebigen Stellen des Bodens nur eine geringe Stützkraft eingebracht werden, während an Stellen mit festem Boden eine größere Stützkraft eingebracht werden kann.

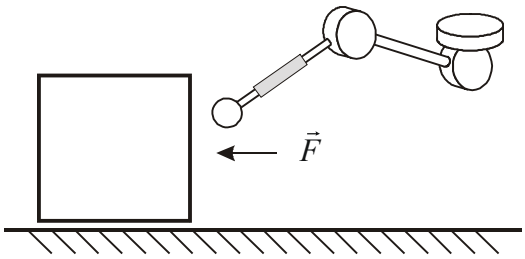



Abbildung 6.2: Manipulation mit Hilfe eines Beines

Die Manipulation ist eine andere Möglichkeit, mit der Umgebung zu interagieren. Hierbei werden die Beine des Roboters als Manipulator eingesetzt, z.B. um Objekte abzutasten oder zu verschieben. Abbildung 6.2 zeigt hierzu eine typische Situation. Diese Operationen können in eine Suchphase und eine Manipulationsphase aufgeteilt werden. In der Suchphase ist es wichtig, ein Bein in eine bestimmte Richtung zu bewegen und mit einer bestimmten Kraft einen Kontakt mit einem externen Objekt herzustellen. Das kann zum Beispiel mit Hilfe eines Dämpfungsreglers

$$\dot{\vec{X}}^{(i)} = \mathbf{K}_{F2} \cdot \left( \vec{F}_{soll}^{(i)} - \vec{F}^{(i)} \right) \tag{6.1}$$

erfolgen. Dabei ist $\dot{\vec{X}}^{(i)}$ der Geschwindigkeitsvektor des Fußes, der die Bewegung zum Objekt realisiert. Diese Bewegung erfolgt so lange, bis sich ein Gleichgewicht zwischen dem Vektor der Sollkräfte $\vec{F}_{soll}^{(i)}$ und den aktuell wirkenden Kräften $\vec{F}^{(i)}$ erreicht ist. Dabei ist die Bewegungsrichtung von der Anpassungsmatrix $\mathbf{K}_{F2}$ und der Kräftedifferenz abhängig. Der Kontakt wird auch aufrecht erhalten, wenn sich das Objekt oder der Roboter bewegt. Nach Herstellung eines Kontaktes kann die Oberfläche abgetastet werden, indem eine zur Oberfläche tangentiale Geschwindigkeitskomponente vorgegeben wird. Eine andere Möglichkeit besteht darin, die Sollkraft schrittweise zu erhöhen, bis die Reibungskraft des Objektes gegenüber dem Untergrund überwunden ist. Diese Interaktion kann auch mit einem auf den Roboter montierten Greifarm durchgeführt werden. In diesem Fall müssen die Kräfte entweder im Handgelenk am Greifer oder mit Hilfe der Kraftsensoren in den Beinen ermittelt werden. Abbildung 6.3 stellt eine solche Situation dar.

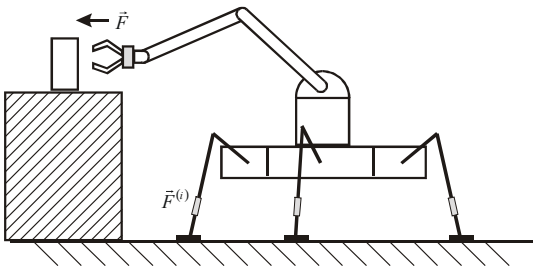



Abbildung 6.3: Manipulation mit Hilfe eines Greifarms

## 6.2 Bohroperationen

Durch die Fähigkeit, dass der Roboter Körperbewegungen ausführen kann sind eine Reihe von wichtigen Anwendungen möglich. Dazu zählen auszuführende Arbeiten in schwer zugänglichen Bereichen, wie das Schließen von Ventilen in havarierten Umgebungen, die Entnahme von Proben oder die Montage von Bauteilen.

Die Flexibilität für den Einsatz wird durch den frei beweglichen Körper gegeben oder verbessert. Am Roboterkörper selbst können die für die auszuführenden Operationen benötigten Werkzeuge oder Instrumente befestigt werden. Mit Hilfe von Körperbewegungen eines Laufroboters können diese positioniert werden.

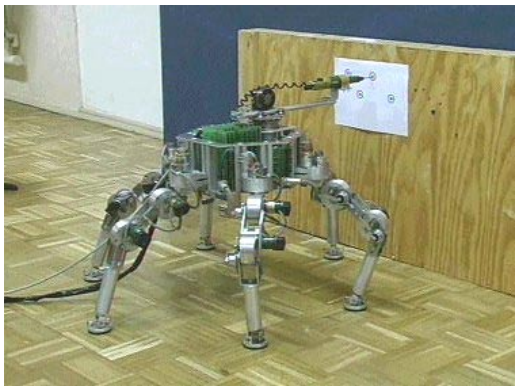


Abbildung 6.4: Das Bohren mit dem Laufroboter Katharina

Am Laufroboter Katharina wurde für das Ausführen von Bohrungen eine Bohrmaschine befestigt. Zur Demonstration des Bohrens wurde ein Experiment durchgeführt, das Abbildung 6.4 wiedergibt. Der Laufroboter Katharina läuft an eine senkrechte stehende Holzwand, positioniert die Bohrmaschine an einem vorgegebenen Punkt und bohrt anschließend ein Loch parallel zum Untergrund. Die Körperbewegungen des Roboters werden mit Hilfe eines Joysticks der externen Steuerung gesteuert. Während des gesamten Experiments werden die Fußkräfte gemessen und aufgezeichnet.

Dabei wirkt die während des Bohrens angreifende Kraft auf den Roboter und wird über dessen Mechanik auf die Füße übertragen. Ausgehend von den Gleichgewichtsbedingungen

$$\sum F_{X_1}^{(i)} = F_{X_1}^{(B)} \tag{6.2}$$

$$\sum F_{X_2}^{(i)} = F_{X_2}^{(B)} \tag{6.3}$$

$$\sum F_{X_3}^{(i)} - F_g = F_{X_3}^{(B)} \tag{6.4}$$

kann man die interessierenden Bohrkräfte ermitteln, die auf den rechten Seiten in den Gleichungen (6.2) bis (6.4) stehen. Auf den linken Seiten stehen die gemessenen Kräfte und die bekannte Gewichtskraft des Roboters.

Abbildung 6.5 zeigt die während des Experiments in Bohrrichtung wirkende Bohrkraft. Es ist zu erkennen, dass nach etwa 6 Sekunden der erste Kontakt mit der Wand erfolgt und der Bohrvorgang beginnt. In den ersten 4 Sekunden des Bohrvorganges steigt die Bohrkraft auf ca. 35 N an und bleibt dann während des Bohrvorganges im Bereich von 30 bis 40 N. Nach

ca. 23 s Gesamtzeit ist die Holzwand durchbohrt und die Bohrkraft fällt sehr schnell ab, da von der Holzwand keine Gegenkraft mehr erzeugt wird.

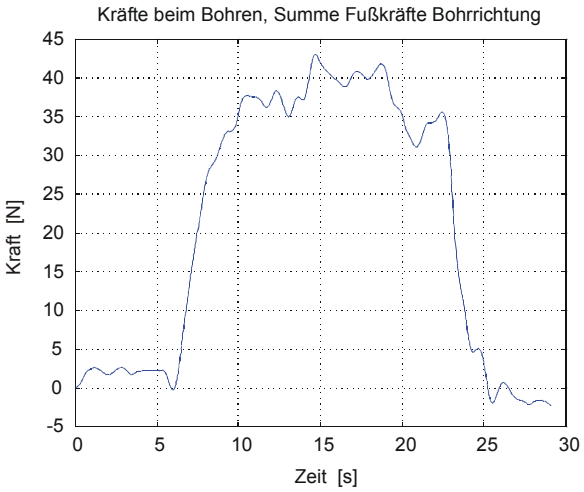



Abbildung 6.5: Bohrkräfte

Da es möglich ist, eine von außen auf den Roboter wirkende Kraft mit Hilfe der Kraftsensoren in den Füßen zu messen, liegt es nahe, den Bohrvorgang selbst mit Hilfe der gemessenen Kräfte zu steuern. Der Bohrvorgang kann in eine Suchphase, eine Bohrphase und eine Rückstellphase aufgeteilt werden. In der Suchphase erfolgt die Positionierung der Bohrmaschine an die gewünschte Bohrstelle mit Hilfe von Körperbewegungen. Für die Bewegung des Bohrers entlang der Bohrachse bis an die Wand heran wird ein Dämpfungsregler nach Gleichung (6.1) benutzt. Die Bewegung der Querachsen wird beispielsweise mit Hilfe eines Kamerakopfes und einem Algorithmus zum Visual Servoing gesteuert, so dass die Bohrachse zum Zielpunkt ausgerichtet wird. Ist der Kontakt erfolgt, so beginnt die Bohrphase. Die Bohroperation wird durch einen Dämpfungsregler für alle drei Koordinatenachsen gesteuert. In Bohrrichtung erfolgt die Vorgabe der gewünschten Bohrkraft als Sollkraft. Um die Kräfte für die Querrichtungen zu minimieren, wird für die entsprechenden Kraftkomponenten die Sollkraft auf Null gesetzt. Ist die Bohrmaschine parallel zur Körperachse $X_1$, dann ist das Steuergesetz für die Bohrphase

$$\dot{\vec{X}} = \mathbf{K}_{F2} \cdot \left( \begin{pmatrix} F_{soll}^{B} \\ 0 \\ 0 \end{pmatrix} - {}^{K}\vec{F} \right). \tag{6.5}$$

Dabei ist $\dot{\vec{X}}$ die Bewegung des Roboterkörpers entlang seiner Körperachsen und bestimmt die Parameter zur Körperbewegung nach Gleichung (5.69).

Da die Bohrmaschine fest am Roboterkörper befestigt ist und die Bohrbewegung mit Hilfe des Roboterkörpers realisiert wird, müssen die wirkenden Kräfte im Roboterkoordinatensystem dargestellt werden. Dies ist bereits dadurch realisiert, dass die Beinkoordinatensysteme einen festen Bezug zum Roboterkoordinatensystem besitzen. Dadurch ist nur eine Transformation mit festen Parametern notwendig. Die Richtung der Gewichtskraft in Gleichung (6.4) lässt sich mit verschiedenen Methoden bestimmen. Ist a priori bekannt, dass der Roboter auf einer horizontalen Fläche steht, so lässt sich die Richtung anhand der Neigung des Roboter-

körpers bestimmen. Ist das nicht der Fall, so kann die Richtung der Gewichtskraft durch Analyse der wirkenden Stützkräfte ohne Interaktion mit externen Objekten bestimmt werden, da die Stützkräfte die einzig wirkenden Kräfte sind. Eine weitere Möglichkeit ist die Bestimmung mit Hilfe eines Neigungssensors. In jedem Fall kann die Richtung der Gewichtskraft bestimmt und mit Hilfe zweier Winkel $\alpha_{x1,g}$ und $\alpha_{x2,g}$ angegeben werden. Die Richtung der montierten Bohrmaschine ist durch den Winkel $\alpha_{X1,B}$ bestimmt. Das Steuergesetz lautet

$$\dot{\vec{X}} = \mathbf{K}_{F2} \cdot \begin{pmatrix} \mathrm{Rot}\left({}^{K}X_3, \alpha_{X_3,B}\right) \cdot \left(F_{soll}^{(B)} \quad 0 \quad 0\right)^{\mathrm{T}} \\ -\left(\sum \vec{F}^{(i)} - \mathrm{Rot}\left({}^{K}X_1, \alpha_{X_1,g}\right) \cdot \mathrm{Rot}\left({}^{K}X_2, \alpha_{X_2,g}\right) \cdot \left(0 \quad 0 \quad F_g\right)^{\mathrm{T}}\right) \end{pmatrix}. \tag{6.6}$$

Dabei wird die Sollkraft $F_{soll}^{(B)}$ mit Hilfe der Rotation um den Winkel $\alpha_{X1,B}$ in Übereinstimmung mit der Bohrrichtung gebracht. Die Bohrkraft wird über die Summe der wirkenden Fußkräfte abzüglich der mittels Rotation bestimmten Gewichtskraftkomponenten bestimmt. Damit ist das kraftgeführte Bohren in verschiedenen Richtungen und mit Körperneigung möglich.

In der dritten Phase, der Rückstellphase erfolgt eine Bewegung in inverser Richtung. Dabei ist die Bewegung entlang der Bohrachse nicht kraftgeführt, während für die Querachsen weiterhin die Dämpfungsregelung aktiv ist, damit die Querkräfte minimiert werden.

Der Unterschied dieser Methode gegenüber der in [Schmucker et al. 96a] vorgestellten Methode besteht darin, dass bei der hier vorgestellten Methode die Korrekturbewegung basierend auf dem Kräftegleichgewicht berechnet wird. Bei der in [Schmucker et al. 96a] vorgestellten Methode wird mittels eines am Werkzeug angebrachten Kraftsensors das auf den Roboter wirkende Moment ermittelt. Der Dämpfungsregler verarbeitet die gemessenen Quermomente und setzt diese in kartesische Bewegungen um. Der Vorteil dabei ist, dass durch einen angepasster Kraftsensor empfindlichere Messungen möglich sind. Der Nachteil jedoch ist der Aufwand für den Kraftsensor und dass die Steuerungsgleichungen nur Bewegungen entlang kartesischer Achsen bereitstellen.

Der Vorteil der oben entworfenen Methode gegenüber der in [Schmucker et al. 96a] dargestellten Methode ist, dass sie für verschiedene Werkzeuge und Instrumente ohne einen speziellen Kraftsensor genutzt werden kann.

Nicht immer ist die räumliche Lage eines Objektes, in das ein Loch zu bohren ist, bekannt. Um dennoch ein Loch senkrecht zur seiner Oberfläche bohren zu können, muss deren Normale bestimmt werden. Mit Hilfe von Bildverarbeitungsmethoden lassen sich Zielpunkte auf einer Oberfläche anhand von Merkmalen ermitteln. Jedoch ist die Bestimmung der räumlichen Ausrichtung einer gleichmäßig eingefärbten Oberfläche sehr schwierig. Eine Lösung besteht darin, dass drei Punkte auf der Oberfläche taktil bestimmt werden. Mit Hilfe der Ebenengleichung

$$E: \quad \vec{x} = \vec{p}_1 + r \cdot (\vec{p}_2 - \vec{p}_1) + s \cdot (\vec{p}_3 - \vec{p}_1) \text{ mit } r, s \in R, \tag{6.7}$$

lässt sich durch drei Punkte $\vec{p}_1$, $\vec{p}_2$ und $\vec{p}_3$, die nicht auf einer Geraden liegen dürfen, der Normalenvektor

$$\vec{n} = (\vec{p}_2 - \vec{p}_1) \times (\vec{p}_3 - \vec{p}_1) \tag{6.8}$$

bestimmen, der senkrecht auf der Oberfläche des Objektes steht und somit die gesuchte Bohrrichtung beschreibt. Um diese drei Punkte zu bestimmen, ist es nur notwendig, drei Punkte auf der Oberfläche zu finden, das heißt zu ertasten. Dafür wird die oben beschriebene Suchphase mehrfach genutzt. Zwischen den Suchphasen wird der Roboterkörper jeweils parallel zur Tastrichtung verschoben, zum Beispiel einmal entlang der vertikalen Achse und einmal entlang der horizontalen Achse, so dass die drei zu ertastenden Punkte eine Ebene aufspannen.

Zum Bohren des Loches wird der Roboterkörper so ausgerichtet, dass die Bohrachse parallel zum Normalenvektor ist. Die Parameter für die erforderliche Neigung lassen sich mit

$$\alpha_{K,X_2} = \operatorname{atan}\left(\frac{n_{X_3}}{n_{X_1}}\right) \tag{6.9}$$

$$\alpha_{K,X_3} = \operatorname{atan}\left(\frac{n_{X_2}}{n_{X_1}}\right) \tag{6.10}$$

bestimmen. Dieses Verfahren erweitert dien in [Ihme et al. 98] für zwei Dimensionen vorgestellte Lösung auf drei Dimensionen.

## 6.3 Montageoperationen

Das Einsetzen von Teilen in eine Öffnung ist ein Beispiel für Montageoperationen. Da die aktive Nutzung von Informationen über wirkende Kräfte zur Steuerung von Montageoperationen Vorteile gegenüber passiven Methoden aufweist [Brussel 91], können die am Laufroboter vorhandenen Kraftsensoren zur Ausführung der Montageoperationen genutzt werden. Prinzipien zur Nutzung von Kraftinformationen für Montageoperationen wurden beispielsweise in [Whitney 87] beschrieben. Ein Verfahren zum Einsetzen einer Röhre in eine trichterförmige Öffnung in [Schneider et al. 95] zeigt, dass sich mit Hilfe einfacher überlagerter Bewegungen auch komplexe Operationen durchführen lassen.

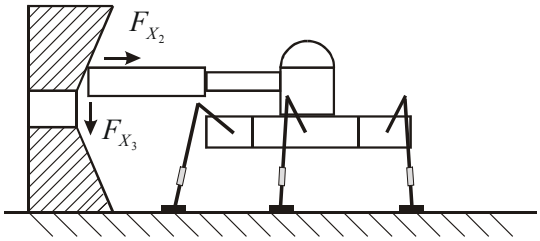



Abbildung 6.6: Beispiel einer Montageoperation

Abbildung 6.6 zeigt ein Beispiel für Montageoperationen. Dabei besteht die Aufgabe, das einzusetzende Teil entlang der trichterförmigen Oberfläche zu führen und es dann in die Öffnung einzusetzen. Diese Aufgabe kann in eine Suchphase, eine Führungsphase und eine Einsetzphase unterteilt werden. Die Suchphase wird mit Hilfe der aktiven Dämpfung realisiert, wobei das einzusetzende Teil an die Wand der trichterförmigen Öffnung herangeführt wird. Bei der Führungsphase werden mehrere Bewegungen überlagert. Diese sind eine Bewegung in Richtung Öffnung und eine Ausweichbewegung in die Querrichtungen. In Längsrichtung

wird eine Bewegungsgeschwindigkeit vorgegeben. Durch die trichterförmige Oberfläche entstehen beim Bewegen in Richtung Öffnung Querkräfte. Daher wird in Querrichtung eine Bewegung nach dem Prinzip der aktiven Dämpfung nach Gleichung (6.1) erzeugt. Dadurch entsteht eine Ausweichbewegung quer zur Vorschubrichtung. Wenn die Querkräfte verschwinden, beginnt die Einsetzphase.

## 6.4 Positionierung eines auf einem Laufroboter befestigten Messinstrument

Die Fähigkeit Körperbewegungen mit einem Laufroboter auszuführen und diese mit seinen Laufbewegungen zu kombinieren, eröffnet die Möglichkeit, Messinstrumente räumlich zu positionieren. Die in den vorangegangenen Abschnitten gezeigten Methoden gelten dabei auch für Messinstrumente, die an einem bestimmten Ort platziert werden müssen. Durch den gezielten Einsatz von Körperbewegungen können diese ganz ohne bzw. nur mit wenigen zusätzlichen Freiheitsgraden positioniert werden. Dadurch wird die Komplexität der Steuerung und die Anzahl der mitzuführenden Hilfsmittel verringert. Ein Beispiel hierzu zeigt Abbildung 6.7. Sie zeigt den Laufroboter Katharina mit einem Kamerakopf. Dieser ist nur mit drei vertikalen Freiheitsgraden ausgestattet ist. Daher kann keine Nickbewegung ausgeführt werden. Der fehlende horizontale Freiheitsgrad kann, wie in der Abbildung demonstriert wird, durch Neigen des Körpers ersetzt werden. Der positive Effekt neben der Vereinfachung ist, dass durch die weggefallenen Freiheitsgrade auch eine Masseeinsparung erreicht wird. Weniger zu transportierende Masse hat einen geringeren Energiebedarf zur Folge, wodurch sich die Autonomie des Roboters verbessert.

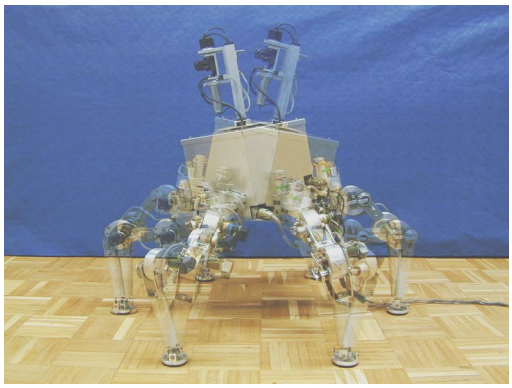


Abbildung 6.7: Beispiel einer Positionierung von Messinstrumenten

# 7 Weiterführende Arbeiten

In den vorangegangenen Kapiteln wurden eine Reihe von Möglichkeiten zur Steuerung von sechsbeinigen Laufrobotern vorgestellt, die komplexe mechatronische Systeme darstellen. Am Beispiel des Laufroboters Katharina wurde die konkrete Realisierung eines solchen mechatronischen Systems dargestellt. Mit Katharina konnten die Prinzipien und Methoden zur Beherrschung solcher komplexen Systeme getestet und die Ausführbarkeit komplexer Operationen, beispielsweise der Überlagerung von Lauf- und Körperbewegung, demonstriert werden.

Bei der vorliegenden Lösung für die Steuerung von Laufrobotern wurden nur einfache Laufmuster genutzt und Sensoren verwendet, die unmittelbar für das Laufen wichtig sind. Bei weiterführenden Arbeiten sollen diese Einschränkungen nicht mehr gelten. Das neue Steuerungssystem soll auch mit unterschiedlichen Bodeneigenschaften, z.B. mit unebenen Gelände, mit Hindernissen und nachgiebigen Untergrund umgehen können. Dazu ist die Erfassung der näheren Umgebung notwendig, um diese Informationen zur Planung der Laufbewegungen verwenden zu können. Mit einer verbesserten Bewegungsregelung soll es möglich sein, die Laufbewegungen noch schneller und genauer auszuführen. Dadurch entstehen neue Probleme bei der Informationsverarbeitung, die mit Hilfe eines verbesserten hierarchischen Steuerungssystem gelöst werden können. Dies gilt besonders vor dem Hindergrund, das solche Steuerungssysteme immer Echtzeitsysteme darstellen.

Die weiterführenden Arbeiten lassen sich zwei wesentlichen Bereichen zuordnen, die aber nicht losgelöst voneinander betrachtet werden dürfen. Ein Bereich ist mit der Planung und Ausführung von Bewegungen und Tasks verbunden, die mit Hilfe eines hierarchischen Steuerungssystems realisiert werden. Der zweite Bereich steht in Zusammenhang mit einem Sensor-Aktor-System, wo die Gewinnung und Verarbeitung von Sensorinformationen im Vordergrund stehen. Nachfolgend wird näher auf diese beiden Bereiche eingegangen.

## 7.1 Scheduling und Steuerungsarchitektur

Eine wichtige Fragestellung bei der Steuerung von Laufrobotern ist, nach welchen Regeln diese erfolgt. Dabei stehen Fragen nach den Steuergesetzen und formalen Zusammenhängen im Vordergrund. Eine weitere wichtige Frage ist, wie diese formalen Zusammenhänge in einem realen, echtzeitfähigen Steuerungssystem umgesetzt werden können. Dabei treten Fragen der Steuerungsarchitektur und des Scheduling in den Vordergrund.

Beide Aspekte sind sehr stark miteinander verknüpft. In einem hierarchischen Robotersteuerungssystem bestehen in einer unteren Ebene stärkere zeitliche Restriktionen als in einer höheren Steuerungsebene. Bei einem zweistufigen hierarchischen Modell ist beispielsweise auf der unteren Ebene entscheidend, wie die Tasks die geplanten Bewegungen umsetzen. Dazu muss der Erfolg oder Nichterfolg der Bewegungen überwacht werden. Die für die Ausführung der Bewegungen verantwortlichen Tasks, beispielsweise Trajektorieninterpolation, Erfassung von Sensorwerten, Koordinatentransformationen oder Regelungstasks, unterliegen zeitlichen Anforderungen, die sie zu erfüllen haben. Auf dieser Ebene kann man von einem Mikroscheduling sprechen, da es bezüglich der Zeit eine sehr feine Auflösung gibt. In der höheren Steuerungsebene steht die Frage nach den Bewegungen im Vordergrund, die als nächstes

ausgeführt werden sollen. Dabei ist in Abhängigkeit von der Umgebung des Roboters nicht gesichert, dass die geplanten Bewegungen auch zum Erfolg führen. Die Bewegungsplanung hat einen wesentlich größeren Zeithorizont als die Taskplanung auf der unteren Ebene. Deshalb kann man diese Bewegungsplanung auch als Makroscheduling bezeichnen. Beide Ebenen sind voneinander abhängig. Das Makroscheduling gibt die zu realisierenden Bewegungen vor und das Mikroscheduling ist für die Ausführungstasks verantwortlich. Sollte die Ausführung nicht korrekt erfolgen können, so muss eventuell eine neue Bewegung geplant werden. Die Einhaltung der vorgegebenen Parameter muss von einer Monitoringkomponente überwacht werden, wie in Abbildung 7.1 am Beispiel von zwei Steuerungsebenen dargestellt ist. Nachfolgend wird näher auf die Steuerungsarchitektur und das Scheduling eingegangen.

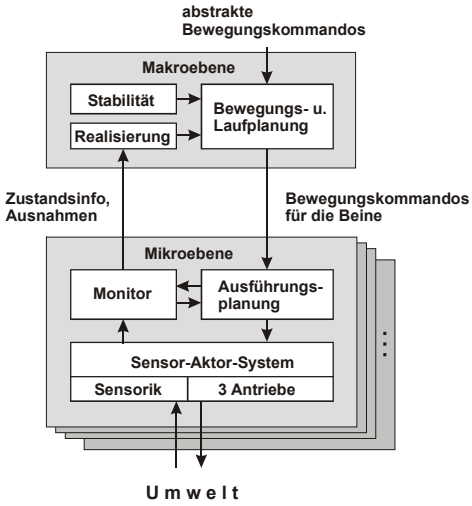



Abbildung 7.1: Planungsebenen

### 7.1.1 Steuerungsarchitektur

Bei der Realisierung einer Steuerungsarchitektur muss die Frage geklärt werden, wie die zu implementierenden Steuergesetze in einer Architektur zu organisieren sind. Zur Realisierung der Steuerung für den Laufroboter Katharina wurde ein hierarchisches Modell genutzt. Im neuen Steuerungssystem erlangen diese Abstraktionsmöglichkeiten noch mehr Bedeutung. Beispielsweise kann das Problem der komplizierten Form des Arbeitsbereiches eines Beines bei der Trajektorienplanung auf die Beinebene verlagert werden, da der Arbeitsbereich in direktem Zusammenhang mit der Beinkinematik steht. Das Bein bietet der Planungsebene einen Service an, die maximale Länge einer geradlinigen Trajektorie zu bestimmen, um die größtmögliche Schrittlänge zu nutzen [Ihme, Deutscher 01]. Dadurch ist es möglich, die Bewegungsmöglichkeiten der Beine besser zu nutzen.

Beim Design der Steuerungsarchitektur wird bei der Modellierung der funktionalen Zusammenhänge gleichzeitig festgelegt, wie die Interaktion mit der Umgebung des Roboters erfolgt bzw. welche physikalischen Größen gesteuert werden sollen. Hierbei diktieren die Eigenfrequenzen der gesteuerten Systeme, deren Trägheit oder andere Systemzeitkonstanten die benötigte Reaktionszeit des Steuerungssystems. Weitere Restriktionen bezüglich der Verarbeitungszeit sind applikationsspezifischer Natur, beispielsweise der erhöhte Bedarf an Sensormesswerten während der aktiven Exploration von Bodeneigenschaften oder bei der Interaktion mit externen Objekten. Diese zeitlichen Anforderungen sind Grundlage für das Design des

Echtzeitsteuerungssystems. Dazu müssen existierende Rahmen- und Entwurfsumgebungen, wie beispielsweise die Modular Controller Architecture MCA2 [Scholl et al. 00], um Komponenten erweitert werden, die den Strukturelementen, wie Ebenen, Module und Kommunikation, Parameter zuordnen, die im laufenden System Vorgaben für das Scheduling darstellen. Dabei kann es sich sowohl um Zeitvorgaben als auch um alternative Tasks handeln, auf die in bestimmten Situationen zurückgegriffen werden kann.

### 7.1.2 Laufmuster- und Bewegungsplanung

Bei der vorliegenden Realisierung des Steuerungssystems für den Laufroboter Katharina standen Methoden im Vordergrund, um außer dem Laufen auch technische Operationen mit Hilfe des Laufroboters zu ermöglichen. Dabei war die Möglichkeit der Erzeugung definierter und vorhersagbarer Bewegungen von Interesse. Im neuen Steuerungssystem sollen definierte Laufmuster auch in Abhängigkeit von den Umgebungsbedingungen erzeugt werden, die trotz ihrer Unregelmäßigkeit eine sichere Fortbewegung ermöglichen. Dabei muss sowohl auf das Bodenrelief reagiert werden als auch auf Hindernisse, die eine Neuplanung von Bewegungen erfordern. Abbildung 7.2 zeigt ein Beispiel für eine Neuplanung von Trajektorien für ein einzelnes Bein. Beim Abfahren der vorgesehenen Trajektorie erfolgt eine Kollision mit einem Hindernis. Als Reaktion darauf wird eine Neuplanung der Trajektorie durchgeführt, die dann nicht mit dem Hindernis kollidiert.

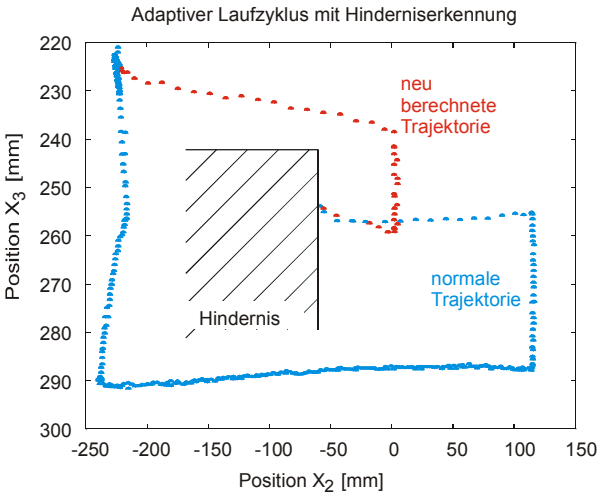



Abbildung 7.2: Beispiel einer Trajektorienneuplanung für ein Einzelbein

Eine wesentliche Komponente zur Lösung dieser Aufgabe ist ein Laufplaner, der zu jedem Zeitpunkt eine ausreichende Stabilitätsreserve gewährleistet. Das heißt, es muss ein ausreichend großer Abstand des projizierten Massenschwerpunktes zu den Rändern des Stabilitätspolygons vorhanden sein. Dabei spielt die Reaktion auf Hindernisse, wie im Beispiel der Abbildung 7.2, ebenso eine Rolle wie die benutzbaren Stützpunkte am Boden. Es existieren verschiedene Ansätze, um sogenanntes freies Laufen zu ermöglichen [Kumar, Waldron 88], [Halme et al. 97], [Cruse et al. 98], [Yoneda et al. 97]. Da in einem möglichen Zielszenario eine sichere Fortbewegung sehr wichtig ist, soll ein vorhersagbares Verhalten erzeugt werden, das in jedem Fall die Stabilität sichert und im Bedarfsfall extern durch Teleoperation gesteuert werden kann. Das ist insbesondere im Hinblick auf technische Operationen notwendig, die nicht vorhergesagt werden können.

Zur Gewährleistung der statischen Stabilität kann das Stabilitätsproblem als Ressourcen- und Aufgabenplanungsproblem aufgefasst werden. Die zu sichernde Ressource ist das Stabilitätspolygon und der Abstand des projizierten Masseschwerpunktes zu dessen Rand. Dazu wird eine bestimmte Anzahl von Stützbeinen in einer bestimmten Konfiguration benötigt. Umgekehrt betrachtet, kann die Ressource „Beintransfer“ nur dann belegt werden, wenn sie für bestimmte Beine frei ist. Dazu könnten die Beine entsprechend einer Dringlichkeit, wenn sie beispielsweise am Rand des Arbeitsraumes angelangt sind, eine entsprechende Priorität erhalten. Zur Lösung dieses Problems können Schedulingmethoden auf ihre Eignung für diese Problemstellung untersucht werden.

Wie gezeigt wurde, sollte die Schrittweite unter Einhaltung der Stabilitätsbedingungen möglichst groß gewählt werden, um ein schnelles Laufen zu ermöglichen. Die größte mögliche Schrittweite kann durch Untersuchung der maximalen Schrittweiten aller beteiligten Beine unter Beachtung des Arbeitsraumes herausgefunden werden [Ihme, Deutscher 01]. Die kleinste mögliche Schrittweite kann dann für alle beteiligten Beine der Beingruppe genutzt werden.
Da die Ausführung der geplanten Bewegungen nicht a priori vorausgesetzt werden kann, muss deren Realisierung überwacht werden. Eine Bewegung kann beispielsweise nicht erfolgreich sein, wenn sie nicht in der vorgesehenen Zeit realisiert wird, d.h. innere Ursachen vorliegen oder ein Hindernis berührt wird, d.h. äußere Ursachen vorliegen. Bei den inneren Ursachen, die mit dem Mikroscheduling in Verbindung stehen, kann eine nicht erfolgreiche Realisierung durch eine Monitoringkomponente festgestellt werden, siehe Abbildung 7.1. Externe Ursachen müssen mit Hilfe der Sensoren des Roboters erfasst werden. Sowohl innere als auch äußere Ursachen können zu einer Neuplanung von Bewegungen führen.

### 7.1.3 Taskplanung

In der untersten Steuerungsebene sind die zeitlichen Restriktionen besonders stark, denn diese werden unter anderem durch die elektrischen und mechanischen Zeitkonstanten der Roboters bestimmt. Wie gezeigt wurde, führen die Anforderungen an die Reaktionszeiten und die echtzeitgerechte Informationsverarbeitung bereits bei sehr einfachen Ausgabenstellungen zu einer wechselnden CPU-Last. Beim neuen Steuerungssystem kann nicht vorausgesetzt werden, dass sich alle Steuerungstasks wie beim Laufroboter Katharina zu einer einzigen Haupttask zusammenfassen lassen. Bei komplexeren Algorithmen ist es wahrscheinlich, dass sich die zeitlichen Anforderungen der einzelnen Tasks unterscheiden, beispielsweise bei der Messwertverarbeitung und bei den Regelungsalgorithmen. Mit einer steigenden Anzahl von Tasks wird der Einsatz von Schedulingmethoden notwendig, die die Ausführung der Tasks entsprechend den zeitlichen Anforderungen sicherstellen.

Tabelle 7.1: Einteilung von Schedulingverfahren nach Planung und Laufzeitanalyse

| Schedulingverfahren | Planung | Laufzeitanalyse |
|---|---|---|
| Statisch | offline | offline |
| Dynamisch, nicht adaptiv | online | offline |
| Dynamisch, adaptiv | online | online |

Dabei ist die wichtige Frage zu klären, wie die Planung der Tasks erfolgt und wann die Parameter für die Machbarkeitsanalyse der Planung gewonnen werden. Tabelle 7.1gibt dazu einen

Überblick. Bei statischen, offline Schedulingverfahren wird die Analyse und die Bestimmung der Parameter offline durchgeführt. Das ist beispielsweise beim Laufroboter Katharina der Fall, wo ein kalenderbasiertes Verfahren genutzt wird. Wie bei der Analyse des Zeitverhaltens beim Laufroboter Katharina deutlich wird, ergeben sich bei statischen Schedulingverfahren ergeben sich zwei wichtige Probleme:

- zur Machbarkeitsanalyse wird eine maximale Ausführungszeit angenommen (Worst Case Execution Time, WCET) und
- die Taskplanung lässt sich im Nachhinein nicht an die aktuellen Bedingungen anpassen.

Das Nutzen der WCET für die Machbarkeitsanalyse hat den Nachteil, dass der konstruierte Fall unter bestimmten Bedingungen erheblich von der tatsächlichen Ausführungszeit abweichen kann. Dieses Phänomen tritt verstärkt bei moderneren Prozessoren auf, wo Techniken wie Pipelining oder hierarchische Speichersysteme, beispielsweise Caches, eingesetzt werden. Eine Garantie für eine Ausführung kann nur dann gegeben werden, wenn auch der schlechteste Fall, also beispielsweise ohne Caches, berücksichtigt wird. Durch dieses sehr konservative Vorgehen wird sehr viel Performance des Rechners verschenkt. Zu Problemen bei der Bestimmung der maximalen Ausführungszeit wird in [Kopetz 97], [Krishna, Shin 97], [Kim et al. 00] ausführlich eingegangen.

Wird durch eine zu optimistische Annahme der WCET eine Frist verletzt, so kann das für das Scheduling erhebliche Konsequenzen haben. Bei EDF-Schedulern kann das zum sogenannten Dominoeffekt führen. Sehr konservative Annahmen zur WCET sind für statische Schedulingverfahren sehr wichtig. Die Anwendung von statischen Schedulingverfahren ist auf Grund der sehr restriktiven Annahmen nur in speziellen, in sich geschlossenen Applikationsumgebungen möglich. Sie sind besonders geeignet für das Scheduling von harten Tasks und bei Vorhandensein von sicherheitskritischen Aspekten sogar zwingend [Kaiser, Nett 98].

Bei autonomen Systemen, so auch bei Laufmaschinen, besteht eine Forderung darin, die verfügbaren Ressourcen so effizient wie möglich zu nutzen. Eine mögliche Lösung bietet sich durch fehlertolerante statische Schedulingverfahren. Bei solchen fehlertoleranten Verfahren wird eine funktionale Redundanz bei der Planung der Tasks ausgenutzt [Kaiser, Nett 98]. Dabei wird für die im Normalfall auszuführende Task von einer erwarteten Ausführungszeit (Expected Case Execution Time, ECET) ausgegangen [Gergeleit, Streich 96]. Kann die Frist für die Task nicht eingehalten werden, stehen verschiedene Methoden zur Verfügung, diese Fehlersituation zu behandeln. Bei der Methode der ungenauen oder schrittweisen Berechnung (imprecise calculation [Liu et al. 94], incremental tasks [Stankovic, Ramamritham 92] oder iterative tasks [Nett et al. 98]) wird in kurzer Zeit ein erstes Ergebnis berechnet, das dann schrittweise verbessert wird. Die Task kann so nach Vorliegen des ersten Ergebnisses auch vor ihrer vollständigen Ausführung abgebrochen werden, so dass in jedem Fall ein verwertbares Ergebnis vorliegt. Eine weitere Möglichkeit besteht in der Anwendung des Task Pair Scheduling, das hauptsächlich für dynamische Schedulungverfahren entwickelt wurde [Streich 95]. Dabei wird eine Task in zwei Subtasks aufgeteilt, wobei nur die fristgerechte Ausführung der Subtask, welche die minimal notwendige Funktionalität implementiert, garantiert wird. Für diese Subtask ist eine maximale Ausführungszeit bekannt, die in jedem Fall zur Verfügung gestellt werden kann. Für die im Normalfall auszuführende Task kann eine erwartete Ausführungszeit angenommen werden. Eine Implementierung dieses Prinzips wurde bei der GMD-Snake vorgenommen [Nett, Streich 97]. Eine dritte Möglichkeit besteht darin, dass insbesondere bei periodischen Tasks die Einhaltung der Fristen nur für eine minimale Anzahl von Instanzen dieser Task garantiert wird. Ein Verfahren dazu wird in [Bernat, Burns 97] beschrieben.

Fehlertolerantes Scheduling bietet für eine neue Steuerung von Laufrobotern bereits entscheidende Vorteile. Es kann insgesamt eine höhere Performance bei einer gesicherten Funktion des Gesamtsystems erzielt werden. Zur Realisierung bieten das Task Pair Scheduling und das Verfahren mit der Garantie einer minimalen Anzahl von Taskinstanzen verwendbare Ansatzpunkte. Beim Task Pair Scheduling besteht der Effekt bei der Ausnahmetask darin, dass lediglich Systemfunktionen oder, wenn aus applikationsspezifischer Sicht gegeben, eine sehr eingeschränkte Funktionalität implementiert wird. Das Verfahren ist beispielsweise für modellbasierte Regler geeignet, bei denen in der Haupttask das modellbasierte, in der Regel rechenaufwendige, Regelungsverfahren implementiert wird. Die Ausnahmetask enthält einen sehr einfachen Regelalgorithmus, der mit einer minimalen Anzahl von Rechenschritten auskommt (z.B. P-Regler) und damit ein sinnvolles Verhalten des gesteuerten Systems garantiert. Die zweite Variante kann ebenfalls für Reglertasks angewandt werden. Nach dem Abtasttheorem von Shannon [Shannon 93] kann eine Information, wie auch das Verhalten eines Systems, dann beschrieben werden, wenn die Abtastrate mindestens doppelt so groß wie die höchste vorkommende Frequenz ist. Wenn die Abtastfrequenz nur geringfügig größer als die Bandbreite des Systems ist, so können sich die Stellgrößen sehr sprunghaft ändern. Außerdem ist zu beachten, dass eine Zustandsänderung des Systems nicht zwischen zwei Abtastungen erkannt werden kann, so dass sich eine maximale Verzögerung der ersten Reaktion von einem Abtastschritt ergeben kann. Die Abtastfrequenz wird deshalb entsprechend den Anforderungen im Bereich der 6-40fachen Bandbreite gewählt. In realen technischen Systemen wird das abzutastende Signal durch einen Anti-Aliasing-Filter bei einer maximalen Frequenz begrenzt, so dass sich die Bereiche der Spiegelfrequenzen im Frequenzbereich im abgetasteten Signal nicht überlappen bzw. keinen Effekt zeigen. Die Grenzfrequenz des Filters wird so gewählt, dass sie möglichst dicht am Frequenzband des abzutastenden Signals liegen, die Phasendifferenz jedoch noch vertretbar ist. Die Abtastfrequenz wird so gewählt, dass der Filter bei dieser Frequenz ausreichend dämpft, sodass auch eine ausreichende Trennung der Spiegelfrequenzen erzielt wird. Die Abtastfrequenz sollte im Bereich des 10-30fachen der Systembandbreite liegen. Für konservative Designs wird bis zum 50fachen der Systembandbreite vorgeschlagen [Franklin et al. 92]. Da die Abtastfrequenz offensichtlich in Grenzen variierbar ist, können die Instanzen der Reglertask in gegebener Situation so weit reduziert werden, bis ein gewähltes Kriterium gerade noch erfüllt ist. Mit der Reduktion der Abtastfrequenz werden eine geringere Dämpfung des Anti-Aliasing-Filters, größere Sprünge des Stellsignals und größere Ungenauigkeiten des digitalen Reglers gegenüber einem gleichwertigen kontinuierlichen Regler in Kauf genommen. Zu beachten ist ferner, dass die Abtastfrequenz bzw. die Zeit zwischen den Abtastzeitpunkten einen Einfluss auf die nachfolgenden Signalverarbeitungsalgorithmen bzw. die Regelalgorithmen haben. Die Abtastzeit kann in solchen Fällen oft als zusätzlicher Parameter in den Algorithmus aufgenommen werden und muss der Task bekannt sein.

Beim hier betrachteten Applikationsszenarium ist jedoch nicht davon auszugehen, dass die Taskparameter im Voraus bekannt oder konstant sind. Der Grund dafür liegt in den beim Einsatz des Roboters nicht im Voraus bekannten Umgebungsbedingungen. Beispielsweise ist nicht vorhersehbar, welche technischen Operationen an welchen Objekten durchgeführt werden müssen. Bei der Umschaltung der Reglerstruktur der Gelenkregler zwischen Stützphase, Returnphase und technischen Operationen verändern sich die Taskausführungszeiten in Abhängigkeit vom gewählten Algorithmus. Bei der aktiven Exploration zur Bestimmung von Bodeneigenschaften kommt es darauf an, dass in schneller Folge Messwerte von Position und Kraft während des Aufsetzens eines Beins auf den Boden gewonnen werden. Dafür kann es notwendig sein, dass bei begrenzten Ressourcen des Beincontrollers eine Neukonfiguration der Tasks vorgenommen werden muss.

Je nach Aufgabenstellung können Inhalt und Parameter der auszuführenden Tasks variieren. Es besteht also die Notwendigkeit, eine Taskplanung auch online durchführen zu können. Dazu sind dynamische Schedulingverfahren notwendig. Dabei wird die Machbarkeitsanalyse durch einen Akzeptanztest ersetzt. Eine Garantie für die Ausführung einer Task kann nur dann gegeben werden, wenn der Akzeptanztest erfolgreich ist. Dabei wird überprüft, ob die ankommende neue Task zusätzlich zu den bisher ausgeführten Tasks einplanbar ist und ob eine Garantie für deren rechtzeitige Fertigstellung gegeben werden kann. Da der Akzeptanztest auch zum negativen Resultat führen kann, ist die Ausführung einer Task nicht a priori gegeben, was in Bezug auf sicherheitskritische Tasks beachtet werden muss. Ein Beispiel für dynamisches Scheduling ist der Spring Kernel [Stankovic, Ramamritham 92]. Tabelle 7.2 gibt einen Überblick zur Einordnung der Schedulingverfahren mit Bezug auf den Akzeptanztest.

Tabelle 7.2: Einordnung verschiedener Schedulingverfahren

| | Statisches Scheduling | Dynamisches Scheduling | |
|---|---|---|---|
| | | Nicht adaptiv | Adaptiv |
| Ohne Garantie bzw. Akzeptanztest | keine Echtzeit, nur best effort | | |
| Mit Garantie bzw. Akzeptanztest | z.B. Rate Monotonic | z.B. Spring Kernel | Nicht sinnvoll |
| Mit Garantie bzw. Akzeptanztest und Fehlertoleranz | z.B. kalenderbasiert + Task Pair Scheduling | z.B. EDF + Task Pair Scheduling | Fehlertoleranz erforderlich, z.B. TAFT |

Auch beim dynamischen Schedulingverfahren ist die maximale Ausführungszeit von großer Bedeutung und muss für die auszuführenden Tasks bekannt sein. Aus den dargestellten Gründen führt die Verwendung der maximalen Ausführungszeit nicht zu einer effizienten Nutzung der CPU. Realistische Schätzungen lassen sich hingegen durch ein Monitoring ermitteln [Gergeleit et al. 99]. Scheduler mit dieser Fähigkeit heißen adaptiv. Da die Schätzung der erwarteten Ausführungszeit naturgemäß nicht exakt ist, muss auch im Fall einer falschen Schätzung und der daraus resultierenden Fristverletzung eine Fehlerbehandling implementiert werden. Dies kann durch das bereits vorgestellte Task Pair Scheduling erfolgen, das Bestandteil des TAFT-Schedulers ist (Time Aware Fault Tolerant) [Nett et al. 98], [Nett, Gergeleit 97] . Mit Hilfe der für das TAFT-Scheduling notwendige Monitoringkomponente ist es möglich, den aktuellen Auslastungszustand zu bewerten und dies auf einer höheren Steuerungsebene zur Planung zu nutzen. Damit ist es beispielsweise möglich, realistische Voraussagen zur maximalen Frequenz von Messungen während der Phase der aktiven Exploration zu gewinnen.

Wird die untere Steuerungsebene mit einem adaptiven Scheduler ausgestattet, so wird es möglich, die Funktion der unteren Steuerungsebene an die konkreten Anforderungen anzupassen, ohne dass der laufende Betrieb unterbrochen werden muss. Dazu sind neben, den die eigentlichen Steuerungsfunktionen beinhaltenden Tasks, folgende Funktionselemente notwendig:

- adaptiver Taskscheduler,
- Monitoringkomponente zur Bestimmung der Taskausführungszeiten,
- Kommunikation zur höheren Steuerungsebene,
- Funktionen zur Manipulation von Taskparametern und
- Funktionen zum Laden neuer Tasks.

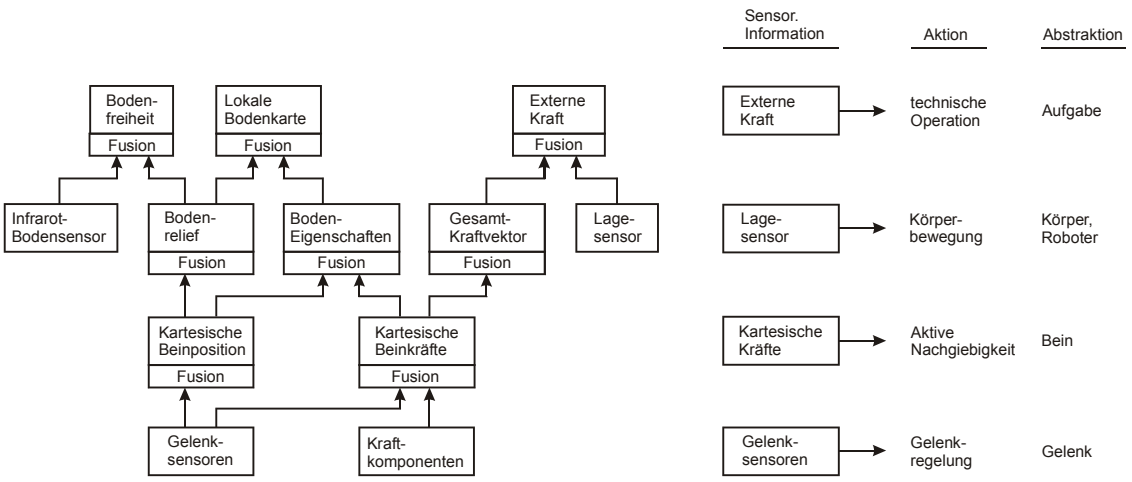
Die neue Funktion der unteren Steuerungsebene hat auch Einfluss auf die Kommunikation mit den höheren Steuerungsebenen. Erfolgt die Kommunikation der beiden Ebenen über einen Bus, so ist auch die Kommunikationsressource Bus nicht mehr a priori planbar, da sich die Kommunikationsanforderungen mit den Tasks und den Taskparametern ändern. Die online Planung muss den Kommunikationsbus bei der Planung mit berücksichtigen.

## 7.2 Sensor-Aktor-System

In einem Sensor-Aktor-System bilden die Komponenten Sensoren, Informationsverarbeitung und Aktoransteuerung eine Einheit. Im neuen Steuerungssystem sollen mehr Sensorinformationen genutzt werden, um mehr Informationen über die Umgebung des Roboters zu gewinnen. Dabei ist wichtig, wie die Informationen mit Hilfe eines geeigneten Interfaces in den Rechner gelangen und wie aus den Sensorsignalen abstrakte Umgebungsinformationen gewonnen werden. Für mobile Roboter ist eine leistungsfähige Bewegungsregelung wichtig, um die Steuerkommandos in adäquate Bewegungen umzusetzen. Dazu soll die Bewegungsregelung verbessert werden. Auf diese Punkte wird nachfolgend näher eingegangen.

### 7.2.1 Sensorinformationsverarbeitung

Bei Laufrobotern ist die Komplexität der Interaktion mit der Umgebung und die dafür notwendige Anzahl von Sensoren besonders groß. Beim Laufroboter Katharina wurden nur Gelenksensoren und Kraftsensoren verwendet, da diese unmittelbar für den Laufprozess wichtig sind und Möglichkeiten zur Interaktion mit der Umgebung eröffnen. Die Beispiele zur aktiven Nachgiebigkeit oder zur Ermittlung der Bohrkräfte mit Hilfe der Fußkraftsensoren zeigen, dass der Sensorinformationsverarbeitung eine große Bedeutung zukommt. Weiterhin ist es nützlich, zusätzliche externe und interne Sensoren, zum Beispiel Neigungssensoren und Distanzsensoren, einzusetzen. Um eine solche große Komplexität zu bewältigen, müssen die anfallenden Sensorinformationen geeignet vorverarbeitet und fusioniert werden. Durch Fusion mehrerer Sensoren lassen sich komplexe Informationen gewinnen. Dabei kann zwischen dem Zusammenführen mehrerer gleichartiger und verschiedenartiger Sensorinformationen unterschieden werden [Luo, Kay 89]. Die neue, fusionierte Sensorinformation kann wieder als ein neuer, virtueller Sensor betrachtet werden. Generell gilt, dass die Sensorinformationen in den unteren Ebenen mehr Signalcharakter tragen. Dabei ist auch der zeitliche Verlauf von Interesse. Mit zunehmender Zusammenführung einzelner Sensorinformationen (Sensorfusion), erhalten diese mehr Symbolcharakter. Während in den unteren Ebenen einfache Informationen in kurzer Zeit zu verarbeiten sind, d.h. ein Sensordatenstrom vorliegt, sind in höheren Ebenen komplexe Informationen in größeren Zeitabständen zu verarbeiten. Dieser Zusammenhang hat auch Einfluss auf das im konkreten Fall notwendige Scheduling.




a) Fusionsschema b) Zuordnung zu Hierarchien

Abbildung 7.3: Sensorfusion und komplexe Operationen

Durch Zusammenführen gleichartiger Sensorinformationen können die Aussagequalität und Aussagesicherheit verbessert werden. Ein Beispiel hierzu ist die Bestimmung der externen Kraft mit Hilfe mehrerer Kraftsensoren. Dazu zeigt Abbildung 7.3a ein Fusionsschema für einen Laufroboter, bei dem Sensorinformationen verarbeitet werden, die in unmittelbarem Zusammenhang mit dem Laufen und der Körperbewegung stehen. Aus den Beinpositionen und den Kraftkomponenten werden die Beinkraftvektoren gewonnen. Die Beinkraftvektoren der sechs Beine werden mit Hilfe des Robotermodells, das heißt dem kinematischen Zusammenhang zwischen den Beinkoordinatensystemen, interpretiert und zu einem Gesamtkraftvektor fusioniert. Da sich die störenden Querkräfte zwischen den Beinen wegen actio gleich reactio gegenseitig aufheben, bleibt der Gesamtkraftvektor, der auf den Roboter wirkt, übrig. Somit wird die Aussagesicherheit mit Hilfe von sechs Kraftsensoren erhöht. Der Gesamtkraftvektor besteht aus dem Gewichtskraftvektor und dem Vektor der externen Kraft. Mit Hilfe eines Lagesensors kann die Richtung der bekannten Gewichtskraft bestimmt werden, so dass der externe Kraftvektor bestimmt werden kann. Bei diesem Prozess sind verschiedene Sensoren beteiligt, so dass aus einfachen Sensorinformationen komplexe Sensorinformationen abgeleitet werden können. Abbildung 7.3b zeigt, wie die gewonnenen Sensorinformationen unterschiedlicher Abstraktionsstufe zu entsprechend komplexen Operationen genutzt werden können.

Abbildung 7.4 zeigt ein Beispiel eines Fusionsschemas zur Nutzung von Sensoren eines Roboters zu seiner Lokalisierung in der Umgebung. Kennzeichnend für dieses Fusionsschema ist, dass sich die Sensorinformationen in starkem Maße ergänzen. Jeder der verwendeten Sensoren hat spezielle Vorteile, mit denen die Nachteile anderer Sensoren ausgeglichen werden können. Ein Beispiel hierzu sind die Odometrie und Entfernungssensoren, d.h. Laser- und Ultraschallsensoren. Da einige Beine des Roboters stets Bodenkontakt haben, kann zu jedem Zeitpunkt eine Roboterposition in Bezug zu einer Referenzposition bestimmt werden. Der Nachteil der Odometrie ist, dass die Unsicherheit der Information mit dem zurückgelegten Weg zunimmt. Eine Entfernungsmessung zu markanten Punkten oder Objekten liefert stets Ergebnisse mit einer bestimmten Sicherheit. Der Nachteil an diesem Verfahren ist, dass es in der Regel nicht kontinuierlich arbeitet. Die Kombination beider Verfahren, das heißt die Fusion ihrer Sensorinformationen, ermöglicht, die kontinuierliche Positionsbestimmung der Odometrie mit der Genauigkeit der Entfernungsmessung zu verbinden.

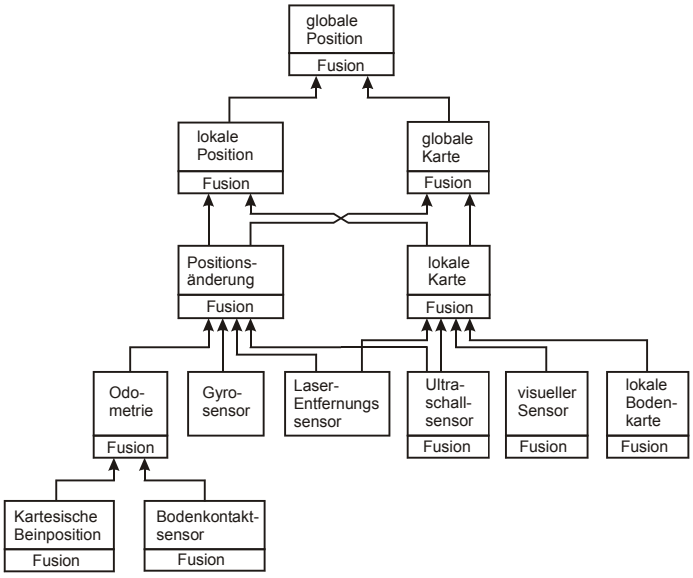



Abbildung 7.4: Lokalisierung beim Laufroboter

Gegenstand fortführender Arbeiten ist es, die Methoden der Sensorfusion unter Echtzeitbedingungen zu untersuchen und geeignete Verfahren zu ermitteln, die für die Anwendung bei Laufrobotern geeignet sind.

### 7.2.2 Sensor-Aktor-Interface und Rechnerarchitektur

Bei der Realisierung einer Rechnerarchitektur für ein komplexes mechatronisches System ist es von großer Bedeutung, wie eine Kopplung der Sensoren und Aktoren mit den Recheneinheiten erfolgen kann. Dabei treten für jeden Sensor und Aktor neue Anforderungen in Erscheinung. Die Suche nach einem geeigneten Interface lässt die Wahl in der Regel auf einen Mikrocontroller fallen, da diese bereits geeignete Interfaces besitzen. Dies ist bei vielen Laufrobotern der Fall, beispielsweise bei Lauron II [Cordes et al. 97], Max (TUM Walking Machine) [Pfeiffer et al. 95] und Katharina [Schmucker et al. 96b]. Bei neuen Robotern, etwa Lauron II, wird oft ein Rechnersystem aus Mikrocontrollern, die im wesentlichen einfache Funktionen zur Gelenkregelung ausführen, und einer Hochleistungs-CPU, die komplexere Funktionen berechnet, genutzt. Zur Kommunikation zwischen dem zentralen Rechner und den Mikrocontrollern wird oft ein CAN-Bus verwendet, da er inzwischen in viele Mikrocontroller, beispielsweise dem C167 [Infineon 00], integriert worden ist. Damit vereinfacht sich ein nicht unwesentlicher Teil der Implementierungsarbeit. Diese Rechnerarchitektur stößt dann an ihre Grenzen, wenn mehr Sensoren, beispielsweise für eine verbesserte Bewegungsregelung, benötigt werden oder Regelungsschleifen mit einem hohen Regelungstakt über den CAN-Bus geschlossen werden. Das ist beispielsweise bei der Kraftregelung im kartesischen Raum oder beim dynamisch stabilen Laufen der Fall. Der Laufroboter Johnnie der TU München [Gienger et al. 01] wurde deshalb mit mehreren CAN-Bussen ausgestattet.

Der Nachteil der auf Mikrocontroller basierten Interfaces ist, dass sie sehr speziell sind. Es entsteht daher eine starke Bindung an einen bestimmten Mikrocontroller. Dadurch wird ein Systemwechsel, beispielsweise aus Performancegründen, erschwert. Da Sensor-Aktor-Interfaces naturgemäß sehr stark an die verwendete Hardware gebunden sind, ist eine Lösung mit

einem skalierbaren und konfigurierbaren Interface sehr vorteilhaft, das auch Funktionen zur Signalvorberarbeitung bietet.

Die Lösung besteht darin, das Sensor-Aktor-Interface von der CPU zu trennen und das Interface konfigurierbar zu gestalten. Dadurch entsteht bei der Wahl der Rechnerarchitektur eine größere Flexibilität, da das Interface als Auswahlkriterium wegfällt. Somit kann die Rechenleistung an den tatsächlichen dezentralen Bedarf besser angepasst werden. Das Sensor-Aktor-Interface lässt sich mit Hilfe eines FPGA konfigurierbar gestalten, so dass die Anpassung an eine konkrete Aufgabenstellung möglich ist. Abbildung 7.5 zeigt hierzu eine mögliche Konfiguration.

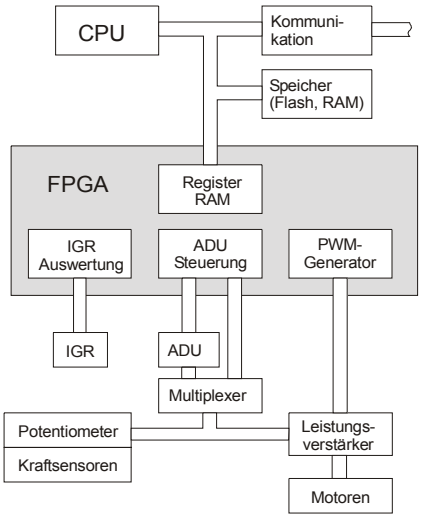



Abbildung 7.5: Sensor-Aktor-Interface mit FPGA

Dieser Lösungsansatz mit FPGA hat insbesondere nachfolgende Vorteile:

- Durch in den FPGA integrierbare Funktionen zur zeitgesteuerten Sensorabfrage, Signalvorverarbeitung (Ansteuerung ADU, Zähler) und Signalformung (z.B. PWM) kann die CPU von zeitkritischen Peripherieoperationen entlastet werden. Diese müssten sonst mit Hilfe von Interrupts und harten Tasks implementiert werden. Es ist möglich, die Peripherieoperationen direkt durch die CPU zu initiieren oder eine Zeitsteuerung zu verwenden. Die Sensorinformationen werden zwischengespeichert, so dass die Anforderungen an die Tasks zum Einlesen der Sensorwerte weniger streng sind.

- FPGA's, z.B. der Xilinx Spartan-Serie, lassen sich mit Hilfe von hierarchisch aufbaubaren Logikplänen, Zustandsautomaten und VHDL programmieren. Einzelne Module, z.B. Generatoren für die Puls-Weiten-Modulation, lassen sich wie Software wiederverwenden und entsprechend den Anforderungen zu komplexen Peripheriecontrollern skalieren. Durch die Programmierung der Logikfunktionen ist es möglich, auch nach erfolgtem Hardwareaufbau Veränderungen vorzunehmen. Das Interface ist nicht an eine spezielle CPU gebunden.

- In den FPGA lässt sich ein Interface integrieren, auf das wie auf RAM zugegriffen werden kann, so dass sich Ein- und Ausgaben auf der Basis von Memory-mapped I/O realisieren lassen. Dadurch entfallen Kommunikationsprotokolle und durch den parallelen Zugriff wird eine hohe Übertragungsgeschwindigkeit erreicht.

### 7.2.3 Bewegungsregelung

Die Bewegungsregelung ist bei mobilen Robotern Voraussetzung für die Umsetzung der Steuerkommandos in definierte Bewegungen. Sie ist auf einer der untersten Ebenen angesiedelt und verarbeitet Bewegungsvorgaben, wie Position und Geschwindigkeit in Abhängigkeit von Sensorinformationen zu Stellkommandos. Die Bewegungsregelung nimmt damit direkten Einfluss auf das physikalische System Roboter, dessen Eigenschaften sich in den Methoden und Parametern der Regelalgorithmen widerspiegeln.

Beim Laufroboter Katharina erfolgen die Bewegungen relativ langsam, so dass dynamische Effekte nicht in Erscheinung treten. Damit ist es möglich, alle Bewegungen rein kinematisch zu beschreiben. Bei der vorliegenden Realisierung eines Steuerungssystems wurde deshalb auf einen einschleifigen Lageregelkreis zurückgegriffen, der den Vorteil einer einfachen Realisierbarkeit besitzt, was ein wichtiger Aspekt ist. Der Nachteil dieser Lösung ist, dass die Genauigkeit bei schnelleren Bewegungen abnimmt.

Zur Verbesserung der Bewegungseigenschaften, insbesondere bei Körperbewegungen und kraftgeführten Bewegungen, muss zumindest ein Geschwindigkeitsregelkreis unterlagert werden. Der Geschwindigkeitsregelkreis benötigt einen nochmals unterlagerten Momentenregelkreis [Lutz, Wendt 98]. Für den Geschwindigkeitsregler ist die Erfassung der Drehzahl notwendig, die beispielsweise mit Hilfe eines Inkrementalgebers gemessen werden kann. Für den Momentenregelkreis muss das erzeugte Motormoment gemessen werden. Da dieses proportional zum Ankerstrom ist, kann der Momentenregelkreis mit Hilfe eines Stromreglers realisiert werden.

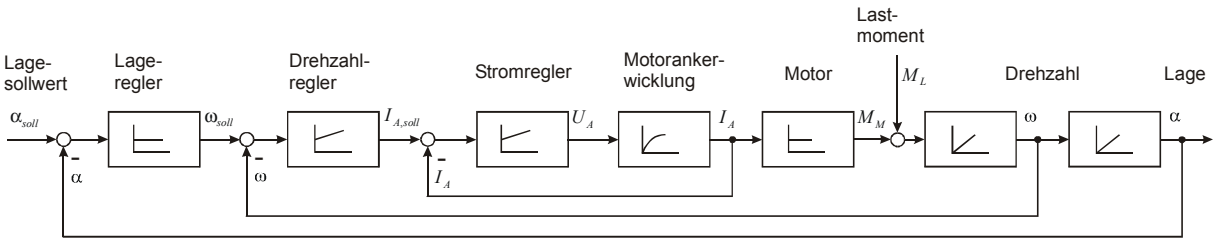



Abbildung 7.6: Mehrschleifiger Regler für die Gelenkregelung

Abbildung 7.6 zeigt einen mehrschleifigen Regelkreis mit unterlagertem Geschwindigkeits- und Stromregler. Zur Erfassung der Messgrößen für Position, Drehzahl und Strom sowie zur Ausgabe des Puls-Weiten-modulierten Signals zur Motoransteuerung kann das oben beschriebene Sensor-Aktor-Interface genutzt werden. Durch die unterschiedlichen Zeitkonstanten in den drei Regelschleifen ist es notwendig, die Perioden der Reglertasks für jede Regelschleife den vorliegenden Zeitkonstanten anzupassen.

Gegenstand weiterführender Arbeiten ist es, die Bewegungsregelung für kleine mobile Roboter, insbesondere Laufroboter, zu verbessern. Dazu ist es notwendig, Lösungen für integrierte Steuerungssysteme, wie sie bei autonomen Systemen notwendig sind, zu entwickeln und zu untersuchen. Dabei werden die Sensoren und Aktoren, die elektronischen Komponenten und die Steuerungssoftware als eine Einheit aufgefasst. So ist es möglich, ein systematisches Gesamtkonzept zu entwickeln.

# 8 Zusammenfassung

Eine Vielzahl von Anwendungen von Laufrobotern ist technischer Natur. Sie erfordern ein deterministisches Verhalten des Laufroboters, um die zu erwartenden Bewegungen vorhersagen zu können. Dieses Ziel kann mit Hilfe eines technisch orientierten Steuerungsansatzes erreicht werden. Für den Einsatz für technische Anwendungen spielt die Autonomie eine wichtige Rolle. Deshalb ist es wichtig, dass das Steuerungssystem in den Roboter integriert wird, wodurch sich eine Reihe von Beschränkungen ergeben, beispielsweise durch begrenzte Rechenleistung. In dieser Arbeit werden Methoden zur Steuerung von sechsbeinigen Laufrobotern aufgezeigt, die in Verbindung mit einem geeigneten Steuerungssystem deterministische Bewegungen als Basis für technische Operationen bereitstellen können. Die wichtigsten Methoden sind die Ableitung von individuellen Bewegungen durch Transformation von zentral erzeugten Bewegungsmustern sowie die Superposition von mehreren einfachen Bewegungen zu komplexen Bewegungen. Dieses Prinzip gilt sowohl für die Bewegungserzeugung als auch zur Ausführung kraftgeführter Bewegungen. Daraus resultieren einfache Lösungen, die in ein eingebettetes Steuerungssystem implementiert werden können. Die Steuerungsalgorithmen müssen dabei an die vorhandenen mechatronischen Komponenten angepasst sein und die Steuerung des Roboters in Echtzeit erlauben. Das heißt, dass sichergestellt wird, dass alle Steuerinformationen innerhalb einer bestimmten Zeit vorliegen. Mit Hilfe des Laufroboters Katharina konnte die Leistungsfähigkeit der entwickelten Steuerungsalgorithmen demonstriert werden. Es konnte u.a. gezeigt werden, dass Lauf- und Körperbewegungen kombiniert werden können, dass kraftgeführte Bewegungen möglich sind und dass technische Operationen, wie z.B. Bohren ausgeführt werden können. Anhand der erfolgreichen Implementierung der Steuerungsprinzipien in den Laufroboter Katharina konnte deren Anwendbarkeit für Roboter der 20 kg-Klasse demonstriert werden.

Zum Erreichen der Zielstellung wurde zunächst der Frage nachgegangen, wie ein geeignetes Steuerungssystem für ein Laufroboter entworfen werden kann. Dazu bieten sich verschiedene Methoden an: Bottom-Up-Entwurf und Top-Down-Entwurf. Es wurde gezeigt, dass diese Entwurfsmethoden nicht immer ein geeignetes Ergebnis hervorbringen, da entweder der globale Zusammenhang nicht ausreichend betrachtet wird oder der Entwurf keine geeignete Lösung für das konkrete mechatronische System hervorbringt. Bessere Ergebnisse können mit einem kombiniertem Ansatz erzielt werden, um die Vorteile beider Verfahren, die Beachtung der globalen Zusammenhänge und des konkreten mechatronischen Systems, zu nutzen. Das Ergebnis der konkreten Betrachtungen ist ein hierarchisches Steuerungskonzept.

Die Rechnerarchitektur muss die benötigte Rechenleistung bereitstellen und die Anbindung von Sensoren und Aktoren muss in geeigneter Weise unterstützt werden. Dabei müssen Einschränkungen bezüglich des verfügbaren Volumens für die Recheneinheit und der begrenzt zur Verfügung stehenden Energie beachtet werden. Für die Steuerung sechsbeiniger Roboter lassen sich drei Varianten herausstellen, die das hierarchische Steuerungskonzept unterstützen.

Für den Laufroboter Katharina wurde eine hierarchische und verteilte Rechnerarchitektur ausgewählt. Durch die Verteilung besteht die Möglichkeit, nebenläufige Steuerungsfunktionen auf verschiedenen Mikrocontrollern zu realisieren. Durch das Konzept der Verteilung besteht die Notwendigkeit der Kommunikation. Die Kommunikation nach dem Master-Slave-Prinzip stellt bei dem Steuerungssystem eine geeignete Variante bezüglich Protokollaufwand

und zeitlicher Determiniertheit dar. Für sicherheitskritische Funktionen ist es wichtig, dass die verwendeten Sensoren Absolutinformationen liefern.
Die Hardware des Steuerungssystems muss folgende Aufgaben erfüllen:

- Erfassung und Auswertung der Messgrößen,
- Berechnung der Steuerinformationen und
- Ausgabe der Stellsignale an die Aktoren.

Für die Integration des Steuerungssystems in den Roboter muss die Hardware sehr kompakt realisiert werden, was durch die Nutzung von Mikrocontrollern möglich ist, da in ihnen Peripheriekomponenten integriert sind, die die Auswertung von Sensorsignalen und das Erzeugen von Steuersignalen unterstützen. Dadurch ist es beispielsweise möglich, Puls-Weiten-modulierte Signale zur direkten Motoransteuerung zu erzeugen, die eine energieeffiziente und räumlich kompakte Motoransteuerung erlauben. Die dadurch entstehende nichtlineare Verhalten kann mit Hilfe von Kennliniengliedern kompensiert werden. Durch den kompakten Aufbau des Steuerungssystems besteht die Notwendigkeit, Maßnahmen zur Unterdrückung bzw. Verhinderung eines störenden Einflusses der Leistungssignale auf Messsignale zu ergreifen.

Zur Steuerung des Roboters wird ein hierarchisches Steuerungssystem eingesetzt. In der Hauptsteuerungsebene erfolgt die Koordinierung aller Bewegungen, während die Beinsteuerungsebene deren Realisierung übernimmt. Von der Hauptsteuerungsebene wird mit Hilfe eines geeigneten Laufmusters sichergestellt, dass die Stabilität des Roboters zu jedem Zeitpunkt gewährleistet ist. Das wird mit einem zentralen Laufmustergenerator erzielt. Die für die technischen Operationen notwendigen Körperbewegungen werden unabhängig von der Laufbewegung erzeugt. Mittels Transformation werden die Fußkoordinaten für die Laufbewegung und die Beinkoordinaten für die Körperbewegung bestimmt. Davon lassen sich die Beinpositionen ableiten, die von der Beinsteuerungsebene realisiert werden. In der Beinsteuerungsebene werden die vorgegebnen Bewegungen durch dezentrale Gelenkregler realisiert. Durch direkte und inverse kinematische Transformationen ist es möglich, auf der Beinebene kraftgeführte Bewegungen zu realisieren, was am Beispiel der aktiven Nachgiebigkeit demonstriert wird.

Anhand der Implementierung der Steuerungsalgorithmen wurde gezeigt, wie mit Hilfe einer Abhängigkeitsanalyse ein Taskmodell erstellt werden kann, das Grundlage für die Implementierung eines Echtzeitsystems ist. Am Beispiel von auf Festkommaarithmetik basierten Algorithmen wurde gezeigt, wie die Steuerungsalgorithmen effizient und zeitlich vorhersagbar implementiert werden können. Im vorliegenden Fall ließ sich ein kalenderbasiertes Schedulingmodell anwenden. Mittels Fallbetrachtung und Laufzeitbestimmungen konnte die Echtzeitfähigkeit der Implementierung gezeigt werden.

Die Grenzen des Steuerungssystems werden mit neuen Funktionen schnell überschritten. Auf Grundlage der gewonnenen Erkenntnisse werden deshalb Ansätze für weiterführende Arbeiten aufgezeigt. Die Steuerungsarchitektur wird beispielsweise komplexer werden und nur mit geeigneten Entwurfsmethoden beherrschbar sein. Bei zunehmender Zahl von Funktionen, die auf modernen Rechnerarchitekturen bereitgestellt werden, werden Schedulingmethoden benötigt, die mit Hilfe von statistischen Laufzeitanalysen ein echtzeitfähiges Gesamtverhalten erzeugen können. Grundlage dafür ist die Nutzung von funktionalen Redundanzen, abbrechbarer Algorithmen und die Ausnutzung systemtheoretischer Grenzen.

Bei Sensor-Aktor-Systemen steigt die Anzahl der zu verarbeitenden Sensorinformationen, mit deren Hilfe komplexe Informationen über die Umwelt gewonnen werden. Dabei spielen sowohl Aspekte der Sensorinterface eine Rolle als auch die Gewinnung abgeleiteter Informationen mittels Sensorfusion.
Die Realisierung von Laufrobotern und deren Steuerung kann nur durch die Integration von Kenntnissen und Methoden unterschiedlicher Teildisziplinen erfolgreich sein. Die Lösung der entstehenden Probleme, die auch durch die Integration der Teildisziplinen bedingt sind, hat positive Rückwirkungen auf die beteiligten Disziplinen. Die Realisierung von Laufrobotern wird deshalb auch weiterhin eine Herausforderung bleiben.

## Literaturverzeichnis

[Analog 95]
AD620 Data Sheet. Rev. E. Analog Devices, Inc., 1999. Document no. C1599c–0–7/ 99. http://www.analog.com/pdf/AD620_e.pdf

[Baginski 98]
Baginski, A.; Müller, M.: InterBus. Grundlagen und Praxis. Hütig Verlag, Heidelberg, 1998. ISBN 3-7785-2471-2

[Bares et al. 89]
Bares, J.; Hebert, M.; Kanade, T.; Krotkov, E.; Mitchell, T.; Simmons, R.; Whittaker, W.L.: Ambler: An Autonomous Rover for Planetary Exploration. IEEE Computer, Vol. 22, No. 6, June, 1989, pp. 18-26.

[Baudoin, Colon 98]
Baudoin, Y, Colon, E.: Humanitarian Demining: Robotics through the (BE)HUDEM Project. In: Workshop on Autonomous Walking 1998: Theory and Practical Realisation of Walking Machines (WAL ’98), June 25, 1998, Magdeburg, Germany. pp. 53-58

[Bekker 56]
Bekker, M. G.: Theory of Land Locomotion. – The Mechanics of Vehicle Mobility. University of Michigan Press, Ann Arbor, 1956

[Bender 93]
Bender, K.: PROFIBUS. The Fieldbus for Industrial Automation. Carl Hanser Verlag, 1993. ISBN 13-012691-8

[Bernat, Burns 97]
Bernat, G.; Burns, A.: Combining $\binom{n}{m}$-Hard deadlines and Dual Priority Scheduling. Proceedings of the 18th IEEE Real-Time Systems Symposium (RTSS’97), San Francisco, 1997, pp. 46-57

[Bernstein 88]
Bernstein, N. A.: Bewegungsphysiologie. Johann Ambrosius Barth Verlag, Leipzig, 1988

[Brussel 91]
Brussel, H. v.: Sensor Based Robot Control. In: Space Course on Low Earth Orbit Transportation and Orbital System. 18.2.-8.3.1991, RWTH Aachen, Vol. 3, pp. 71-1...71-27

[Buehler 01]
Buehler, M.: RePaC Design and Control – Cheap and Fast Autonomous Runners. In: Berns, K.; Dillmann, R. (eds.): Proceedings of the Fourth International Conference on Climbing and Walking Robots (CLAWAR 2001) – From Biology to Industrial Applications. 24-26 September 2001, Karlsruhe, Germany. Professional Engineering Publishing, Bury St Edmunds (u.a.), UK, 2001. pp. 579-585. ISBN 1-86058-265-2

[Byrd, DeVries 90]
Byrd, J. S.; DeVries, K. R.: A Six-Legged Telerobot for Nuclear Applications Development. International Journal of Robotics Research. Vol. 9, No. 2, April 1990. pp. 43-52

[Cordes et al. 93]
Cordes, S., Berns, K., and Dillmann, R.: Steuerungsarchitektur der sechsbeinigen Laufmaschine LAURON. In Autonome Mobile Systeme, 9. Fachgespräch an der Technischen Universität München, Seiten 205-213. Lehrstuhl für Steuerungs- und Regelungstechnik, Günther Schmidt, 1993

[Cordes et al. 97]
Cordes, S., Berns, K., Leppänen, I.: Sensor Components of the Six-Legged Walking Machine Lauron II. In: Proceedings of the 8th International Conference on Advanced Robotics (ICAR'97). Monterey, California, USA. July 7-9, 1997

[Cruse et al. 95]
Cruse H.; Bartling, Ch.; Dreifert, M.; Schmitz, J.; Brunn, D. E.; Dean, J.; Kindermann, T.: Walking: a complex behaviour controlled by simple systems. Adaptive Behaviour, 1995, v. 3, 385-418.

[Cruse et al. 98]
Cruse, H., Kindermann, Th., Schumm, M., Dean, J., Schmitz, J.: Walknet - a biologically inspired network to control six-legged walking. In: Neural Networks, 11 (7-8) 1998, pp. 1435-1447

[Dean 91]
Dean, J.: A model of leg coordination in the stick insect, Carausius morosus. II. Description of the kinematic model and simulation of normal step patterns. Biological Cybernetics. Vol. 64, pp. 403-411, Springer-Verlag, 1991.

[Dewey 00]
Dewey, R.: Full Bridge PWM Motor Driver 3951. Datasheet 29319.4. Allegro Microsystems Inc, 2000. http://www.allegromicro.com/datafile/3951.pdf

[Dillmann, Huck 91]
Dillmann, R.; Huck, M.: Informationsverarbeitung in der Robotik. Springer-Verlag, Berlin u.a., 1991. ISBN 3-540-53036-3

[Eltze et al. 92]
Eltze, J.; Weidemann, H.-J.; Pfeiffer, F.: Walking Machines for Space. In: Proceedings of the 1st International Symposium on Missions, Technologies and Design of Planetary Vehicles. Toulouse, France, Sept. 28 - 30, 1992. pp. 377 - 384

[Ferrell 95]
Ferrell, C.: A comparison of three insect-inspired locomotion controllers. Robotics and Autonomous systems 16 (1995) 135-159

[Föllinger 90]
Föllinger, O.: Regelungstechnik. Einführung in die Methoden und ihre Anwendung. Hütig-Verlag, Heidelberg, 1990. ISBN 3-7785-1808-9

[Franklin et al. 92]
Franklin, G. F.; Powell, J. D.; Workman, M. L.: Digital Control of Dynamic Systems. In Serie: Addison-Wesley series in electrical and computer engineering: control engineering.; Reading, Mass. u.a., Addison-Wesley, 1992, ISBN 0-201-11938-2

[Frik et al. 98]
Frik, M.; Guddat, M.; Losch, D.; Karataş, M.: Terrain Adaptive Control of the Walking Machine TARRY II. In: Proceeddings of the European Mechanics Colloquium, Euromech 375 - Biology and Technology of Walking. March 23-25, 1998, Munich, Germany. pp. 108--115

[Frik et al. 99]
Frik, M.; Guddat, M.; Karataş, M.; Losch, D.: A novel approach to autonomous control of walking machines. In: Proceedings of the 2nd International Conference on Climbing and Walking Robots, 13-15 September, Portsmouth, UK. Professional Engineering Publishing Limited, Bury St. Edmunds 1999, pp. 333-342.

[Fujita, Kageyama 97]
Fujita, M.; Kageyama, K.: An Open Architecture for Robot Entertainment. In: Proceedings of the First International Conference on Autonomous Agents. Marina del Rey, CA, USA, February 5-8, 1997

[Gantmacher 86]
Gantmacher, F. R: Matrizentheorie. Springer-Verlag, 1986. ISBN 3-540-16582-7

[Gao, Song 90]
Gao, X.C.; Song, S.M.: Stiffness matrix method for foot force distribution of walking vehicles. 1990 IEEE International Conference on Robotics and Automation, vol. 3. pp. 1470 – 1475

[Gergeleit et al. 99]
Gergeleit, M.; Nett, E.; Fitzner, J.: On-line Prediction of Execution Times - A Basis for Adaptive Scheduling. In: Fourth International Workshop on Object-oriented Real-time Dependable Systems, Santa Barbara, California, February 1999, ISBN 0-7695-0101-X, pp. 186 - 194

[Gergeleit, Streich 96]
Gergeleit, M.; Streich, H.: TaskPair-Scheduling with Optimistic Case Execution Times - An Example for an Adaptive Real-Time System. In: Second International Workshop on Object-oriented Real-time Dependable Systems, Laguna Beach, CA., IEEE Computer Society Press, 1996

[Gienger et al. 01]
Gienger, M.; Löffler, K.; Pfeiffer, F.: Design and sensor system of a biped robot. In: Berns, K.; Dillmann, R. (eds.): Proceedings of the Fourth International Conference on Climbing and Walking Robots (CLAWAR 2001) - From Biology to Industrial Applications. 24-26 September 2001, Karlsruhe, Germany. Professional Engineering Publishing, Bury St Edmunds (u.a.), UK, 2001. pp. 205-212. ISBN 1-86058-265-2

[Giménez et al. 98]
Giménez, A.; Balaguer, C.; Pastor, J.M.; Padrón, V.M.; Abderrahim, M.: Design and Path-Planning for a Climbing Robot Able to Travel Along 3D Metallic Structures. In: Proceedings of the First International Symposium on Mobile, Climbing and Walking Robots 1998 (CLAWAR ’98). Brussels, Belgium, November 25-27, 1998, S.161-165

[Gorinevsky et al. 97]
Gorinevsky, D. M.; Formalsky, A. M.; Schneider, A. Y.: Force Control of Robotics Systems. CRC Press, 1997. ISBN 0-8493-2671-0

[Gorinevsky, Schneider 90]
Gorinevsky, D. M.; Schneider, A. Yu: Force Control in Locomotion of legged Vehicles over Rigid and Soft Surfaces. International Journal of Robotics Research, Vol. 9, No. 2, April 1990. pp. 4-23

[Gross et al. 86]
Gross, D.; Hauger, W.; Schnell, W.: Technische Mechanik. Band 1: Statik. Heidelberger Taschenbücher Band 225. Springer-Verlag, 1986. ISBN 3-540-11706-7

[Graf et al. 98]
Graf, R.; Vierling, R.; Dillmann, R.: A flexible controller for a Stewart platform. International Conference on Conventional and Knowledge-Based Intelligent Electronic Systems (KES ’98), Adelaide, Australia, 21-23 April 1998

[Gurfinkel et al. 81]
Gurfinkel, V. S.; Gurfinkel, E. V.; Shneider, A. Yu.; Devjanin, E. A.; Lensky, A. V.; Shtilman, L. G.: Walking Robot with Supervisory Control. Mechanism and Machine Theory. Vol. 16, pp. 31-36

[Halme et al. 97]
Halme, A.; Salmi, S.; Leppänen, I.: Control and stabilisation of the semi-dynamical motion of a heavy six-legged walking machine. 8th international conference on advanced robotics (ICAR’97), Monterey, CA, 1997. Monterey 1997, pp. 44-47.

[Halme, Hartikainen 96]
Halme, A.; Hartikainen, K.: Walking machine technology – designing the control system of an advanced six-legged machine. In: Gray, J. O.; Caldwell, D. G. (Editors): Advanced robotics and intelligent machines. The Institution of Electrical Engineers, London, 1996. ISBN 0-85296-853-1

[Haug 93]
Haug, A.; Haug, F.: Angewandte elektrische Messtechnik. Vieweg Verlag, 1993. ISBN 3-528-14567-6

[Hauptmann 91]
Hauptmann, P.: Sensoren Prinzipien und Anwendungen.Carl Hanser Verlag München Wien, 1991. ISBN 3-446-16073-6

[Heimann et al. 98]
Heimann, B.; Gerth, W.; Popp, K.: Mechatronik: eine Einführung in die Komponenten zur Synthese und die Methoden zur Analyse mechatronischer Systeme. Fachbuchverlag Leipzig im Carl Hanser Verlag. München, Wien, 1998. ISBN 3-446-18719-7

[Hertzsch 86]
Hertzsch, A.: CMOS-Schaltkreisliste. Militärverlag, Berlin, 1986. ISBN 3-327-00103-0

[Hiller et al. 99]
Hiller, M.; Müller, J.; Roll, U.; Schneider, M.; Schröter, D.; Torlo, M.; Ward, D.: Design and realization of the anthropomorphically legged and wheeled Duisburg robot ALDURO. In: Proceedings of the 10th World Congress on the Theory of Machines and Mechanisms. IFToMM, University of Oulu, Finnland, June 20-24, 1999.

[Hirose, Kato 98]
Hirose, S.; Kato, K.: Quadruped Walking Robot to Perform Mine Detection and Removal Task. In: Proceedings of the First International Symposium on Mobile, Climbing and Walking Robots 1998 (CLAWAR '98). Brussels, Belgium, November 25-27, 1998, pp. 71-76

[Hüsenner 94]
Hüsenner, T.: Entwurf komplexer Echtzeitsysteme: state of the art. In Reihe: Balzert H. (Herausgeber): Angewandte Informatik, Band 11. BI Wissenschaftsverlag, Mannheim (u.a.), 1994. ISBN 3-411-16441-7

[Ihme, Deutscher 01]
Ihme, T.; Deutscher, M.: Design and Control Aspects for Six-legged Walking Robots to Realize Adaptation to the Environment. In: Berns, K.; Dillmann, R. (eds.): Proceedings of the Fourth International Conference on Climbing and Walking Robots (CLAWAR 2001) - From Biology to Industrial Applications. 24-26 September 2001, Karlsruhe, Germany. Professional Engineering Publishing, Bury St Edmunds (u.a.), UK, 2001. pp. 627-634. ISBN 1-86058-265-2

[Ihme 00a]
Ihme, T.: Realisierung eines verteilten Steuerungssystems für einen Laufroboter. In: Ch. Döschner (Hrsg.): Verteilte Automatisierung - Modelle und Methoden für Entwurf, Engineering und Instrumentierung. Magdeburg, 22. bis 23. März 2000. Fachtagung 2000. ISBN 3-929757-30-3, S. 171 – 177

[Ihme 00b]
Ihme, T.: Kombination von Schreitzyklen und Körperbewegung bei einem sechsbeinigen Laufroboter. Robotik 2000: Leistungsstand - Anwendungen - Visionen - Trends; Tagung Berlin, 29. und 30. Juni 2000. Fachtagung Robotik 2000. Düsseldorf: VDI-Verl., 2000, S. 515-522. ISBN: 03-18-091552-8

[Ihme et al. 98]
Ihme, T.; Schneider, A.; Schmucker, U.: The Use of a Six-Legged Walking Robot as an Adaptive Platform. In: Workshop on Autonomous Walking 1998: Theory and Practical Realisation of Walking Machines (WAL'98), June 25, 1998, Magdeburg, Germany. pp. 45-51

[Ilg 01]
Ilg, W.: Eine bioligisch motivierte adaptive Bewegungssteuerung für eine vierbeinige Laufmaschine. Akademische Verlagsgesellschaft, Berlin, 2001

[Ilg, Berns 95]
Ilg, W; Berns, K: A learning architecture based on Reinforcement Learning for adaptive control of the walking machine LAURON, Robotics and Autonomous Systems, vol. 15, pp. 321-334, 1995

[Infineon 00]
C167CR Derivatives: 16-Bit Single Chip Microcontroller. User's Manual, V. 3.1. Infineon Technologies AG, 2000.
http://www.infineon.com/cmc_upload/0/000/008/066/c167cr_um_v31.pdf

[Intel 90]
MCS-96 Macro Assembler User's Guide for DOS Systems. Intel Corporation, 1990. Order Number 122350.

[Intel 91a]
iC-96 Compiler User's Guide for DOS Systems. Intel Corporation, 1991. Order Number 481195-004.

[Intel 91b]
8096 Floating-Point Library Supplement. Intel Corporation, 1991. Order Number 122365-002.

[Intel 95]
8XC196Kx, 8XC196Jx, 87C196CA Microcontroller Family User's Manual. Intel Corporation, June 1995. Doc. No. 272258-002
ftp://download.intel.com/design/mcs96/manuals/27225802.pdf

[Intel 98a]
87C196KR, 87C196JV, 87C196JT, 87C196JR, and 87C196CA Advanced 16-Bit CHMOS Microcontrollers. Datasheet. Intel Corporation, April 1998. Doc. No 270827-007
ftp://download.intel.com/design/auto/mcs96/datashts/27082707.pdf

[Intel 98b]
87C196KR Specification Update. Intel Corporation, December, 1998. Doc. No. 272861-004
ftp://download.intel.com/design/auto/mcs96/specupdt/27286104.pdf

[Kaiser, Nett 98]
Kaiser, J.; Nett, E.: Echtzeitverhalten in dynamischen, verteilten Systemen, GI Informatik Spektrum 21(6), Dez. 1998, S. 356-365

[Kim et al. 00]
Kim, K.H.; Liu, J.; Kim, M. H.: Deadline Handling in Real-Time Distributed Objects. Proceedings of the IEEE CS 3rd International Symposium on Object-oriented Real-time distributed Computing, ISORC 2000, Newport Beach, CA, March 2000, pp.7-15

[Kirchner 98]
Kirchner, F.: Q-Learning of complex behaviours on a six-legged walking machine. Robotics an Autonomous Systems, vol. 25, 1998, pp. 253-262

[Kitano et al. 99]
Kitano, H.; Tadokoro, S.; Noda, I.; Matsubara, H.; Takahashi, T.; Shinjou, A.; Shimada, S.: RoboCup-Rescue: Search and Rescue for Large Scale Disasters as a Domain for Multi-Agent Research. In: Proceedings of the 1999 IEEE International Conference on Systems, Man, and Cybernetics (SMC '99). October 12- 15 1999, Tokyo. ISBN 0-7803-5731-0

[Kizono et al. 89]
Kizono, J. A.; Iwasaki, M.; Nemoto, T.; Asakura, A.: Development on Walking Robot for Underwater Inspection. In: Waldron, K. J. (ed.): Proceedings of the 4th International Conference on Advanced Robotics, Columbus, Ohio, June 13-15, 1989

[Klein, Wahawisan 82]
Klein, C. A.; Wahawisan, W.: Use of a Multiprocessor for Control of a Robotic System. International Journal of Robotics Research, Vol. 1, No. 2, Summer 1982

[Kopetz 97]
Kopetz, H.: Real-Time Systems: Design Principles for Distributed Embedded Applications. In Serie: The Kluwer International Series in Engineering and Computer Science; SECS 395 Real Time Systems. Kluwer Academic Publishers, Boston u.a., 1997. ISBN 0-7923-9894-7

[Krishna, Shin 97]
Krishna, C. M.; Shin, K. G.: Real-Time Systems. In Serie: McGraw-Hill Series in Computer Science, 1997. ISBN 0-07-057043-4

[Krotkov et al. 91]
Krotkov, E.; Bares, J.; Kanade, T.; Mitchell, T.; Simmons, R.; Whittaker, W.L.: Ambler: a six-legged planetary rover. In: Fifth International Conference on Advanced Robotics, 1991, Robots in Unstructured Environments (ICAR '91), Vol. 1, June, 1991, pp. 717 - 722.

[Kühn 86]
Kühn, E.: Handbuch TTL- und CMOS-Schaltkreise. Verlag Technik, Berlin 1986.

[Kumar, Waldron 88]
Kumar, V.; Waldron, K. J.: Gait analysis for walking machines for omnidirectional locomotionon on uneven terrain. In: A. Morecki, G. Bianchi, K. Jaworek (eds.): Proceedings of the 7th CISM-IFToMM Symposium on Theory and Practice of Robots and Manipulators, RoManSy'88, Udine, Italy, Sept. 12 - 15, 1988. Hermes, Paris 1990. pp. 37-62

[Lawrenz 99]
Lawrenz, W. (Herausgeber): Controller Area Network: CAN; Grundlagen und Praxis. Hütig Verlag, 1999

[Lawson 92]
Lawson, H. W.: CyClone - An Approach to the Engineering of Resource Adequate Cyclic Real-Time Systems, Real-Time Systems, Vol. 4 (1992), No. 1, pp. 55-84

[Lehtinen 94]
Lehtinen, H.: Force based motion control of a walking machine. Espoo 1994, Technical Research Centre of Finland, VTT Publications 179. 150 p.

[Liu 00]
Liu, J. W. S.: Real-Time Systems. Prentice Hall, New Jersey, 2000. ISBN 0-13-099651-3

[Liu et al. 94]
Liu, J. W. S.; Shih, W.-K.; Lin, K.-J.; Bettati, R.; Chung, J.-Y.: Imprecise Computations. Proceedings of the IEEE, vol. 82, no. 1, 1994, pp. 83-94

[Liu, Layland 73]
Liu, C. L.; Layland, J. W.: Scheduling algorithms for multiprogramming in a hard real time environment. Journal of the ACM, Vol. 20, No. 1, pp. 46-61, January 1973.

[Luo, Kay 89]
Luo, R.C.; Kay, M.G.: Multisensor Integration and Fusion in Intelligent Systems. In: IEEE Transactions on Systems, Man, and Cybernetics, Vol. 19, No. 5, September/October 1989, pp.901-931

[Lutz, Wendt 98]
Lutz, H.; Wendt, W.: Taschenbuch der Regelungstechnik. Harri Deutsch Verlag, Frankfurt a. M., 1998. ISBN 3.8171-1552-0

[Martin-Alvarez et al. 94]
Martin-Alvarez, A.; de Peuter, W.; Gonzalez de Santos, P.; Armada, M. A.: A Survey of Locomotion Concepts for Planetary Exploration Rovers. In: 3rd ESA Workshop on Advanced Space Technologies for Robot Applications (ASTRA 1994). ESTEC, Noordwijk, The Netherlands, April 27-28, 1994. Document No. ESA WPP-075

[Martin-Alvarez et al. 96]
Martin-Alvarez, A.; Hillebrand, J.; De Peuter, W.; Putz, P.; Matthyssen, A.; de Weerd, J.F.: Walking Robots for Planetary Exploration Missions. In: Jamshidi, M.; Pin, F. G. (Eds.): Proceedings of the 2nd World Aitomation Congress. Vol. 3: Robotic and Manufacturing Systems. Montpellier, France, May 28-30, 1996. TSI Press, Albuquerque, NM, 1996. pp. 7-14. ISBN 1-88933-500-2

[Mason 81]
Mason, M. T.: Compliance and Force Control for Computer Controlled Manipulators. IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-11, No. 6, June 1981. pp. 418-432

[Maxon 00]
Maxon Hauptkatalog 2000, Maxon motor GmbH

[Moore, Buehler 01]
Moore, E. Z.; Buehler, M.: Stable Stair Climbing in a Simple Hexapod Robot. In: Berns, K.; Dillmann, R. (eds.): Proceedings of the Fourth International Conference on Climbing and Walking Robots: From Biology to Industrial Applications. 24.-26. September 2001, Karlsruhe, Germany. Professional Engineering Publishing, Bury St Edmunds (u.a.), 2000. pp. 603-609. ISBN 1-86058-365-2

[Morecki 97]
Morecki, A.: Modelling and Simulation of Human and Walking Robot Locomotion. In: Morecki, A.; Waldron, K. (eds.): Human and machine locomotion. In Serie Courses and lectures, Nr. 375. Springer, Wien, 1997. pp. 1-78. ISBN: 3-211-82905-9

[Morecki, Waldron 97]
Morecki, A.; Waldron, K. (eds.): Human and machine locomotion. Materials presented during the Advanced School "Modelling and Simulation of Human and Walking Robots Locomotion", Udine, Italy, July 8 - 12, 1996. In Schriftenreihe: Courses and lectures / International Centre for Mechanical Sciences. Springer, Wien, 1997. ISBN: 3-211-82905-9

[Müller et al. 98]
Müller, J.; Schneider, M.; Hiller, M.: Modelling and Simulation of the Large-Scale Hydraulically Driven ALDURO. In: Workshop on Autonomous Walking 1998: Theory and Practical Realisation of Walking Machines (WAL'98), June 25, 1998, Magdeburg, Germany. pp. 83-90

[NCI 99]
Using the NCI logic analyzer: Logic analyzer manual for the PA600 PC-based logic analyzer. NCI, Huntsville, 1999

[Nett et al. 98]
Nett, E.; Gergeleit, M.; Mock, M.: An Adaptive Approach to Object-Oriented Real-Time Computing. In: Proceedings of the IEEE Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'98), Kyoto, Japan, 20-22 April 1998.

[Nett, Gergeleit 97]
Nett, E.; Gergeleit, M.: Preserving Real-Time Behaviour in Dynamic Distributed Systems, IASTED International Conference on Intelligent Information Systems, The Bahamas, 8.-10. 12. 1997

[Nett, Streich 97]
Nett, E., Streich, H.: The GMD-Snake - Real-Time Scheduling of a Flexible Robot Application at run-time; Klaus Ecker and Andrei Tchernykh (Hrsg.): Proceedings of the International Workshop on Parallel Computation and Scheduling, 1997; Ensenada, Mexico, pp. 13-16.

[Neumann 90]
Neumann, P.: Kommunikationssysteme in der Automatisierungstechnik. In Schriftenreihe: Woschni et al.: Reihe Automatisierungstechnik, Band 242. Verlag Technik, Berlin, 1990

[Oberschelp, Vossen 00]
Oberschelp, W.; Vossen, G.: Rechneraufbau und Rechnerst
rukturen. Oldenbourg Wissenschaftsverlag. München, Wien, 2000. ISBN 3-486-25340-9

[Orin 82]
Orin, D. E.: Supervisory Control of a Multilegged Robot. International Journal of Robotics Research, Vol. 1, No. 1, Spring 1982. pp. 79-91

[Pfeiffer et al. 95]
Pfeiffer, F.; Eltze, J, Weidemann, H.: Six-legged technical walking considering biological principles. Robotics and Autonomous Systems 14 (1995) pp. 223-232. ISBN: 3-211-82905-9

[Pfeiffer et al. 97]
Pfeiffer, F.; Rossmann, T.; Steuer, J.: Theory and Practice of Machine Walking. In: Morecki, A.; Waldron, K. (eds.): Human and machine locomotion. In Series Courses and lectures, Nr. 375. Springer, Wien, 1997. pp. 231-282

[Phoenix 98]
Phoenix Contact (Herausgeber): Grundkurs Sensor/Aktor-Feldbustechnuik. Vogel Buchverlag. Würzburg, 1998. ISBN 3-8023-1764-5

[Preumont 94]
Preumont, A.: An investigation of the kinematic control of a six-legged walking robot. Mechatronics. Vol. 4, No. 8, 1994, pp. 821-829

[Pugh et al. 90]
Pugh, D. R.; Ribble, E. A.; Vohnout, V. J.; Bihari, T. E.; Walliser, T. M.; Patterson, M. R.; Waldron, K. J.: Technical Description of the Adaptive Suspension Vehicle. International Journal of Robotics Research. Vol. 9, No. 2, April 1990

[Quinn et al. 01]
Quinn, R. D.; Nelson, G. M.; Bachmann, R. J.; Kingsley, D. A.; Offi, J. T.; Ritzmann, R. E.: Insect Designs to Improve Robot Mobility. In: Berns, K.; Dillmann, R. (eds.): Proceedings of the Fourth International Conference on Climbing and Walking Robots (CLAWAR 2001) – From Biology to Industrial Applications. 24-26 September 2001, Karlsruhe, Germany. Professional Engineering Publishing, Bury St Edmunds (u.a.), UK, 2001. pp. 69-76. ISBN 1-86058-265-2

[Raibert 86a]
Raibert, M. H.: Legged Robots. Communications of the ACM. Vol. 29, No. 6, 1986, pp. 499-514

[Raibert 86b]
Raibert, M. H.: Legged Robots that Balance. In Serie: MIT Press Series in artificial intelligence. Cambridge (u.a.), MIT Press, 1986. ISBN 0-262-18117-7

[Raibert, Craig 81]
Raibert, M. H.; Craig, J. J.: Hybrid Position/Force Control of Manipulators. Transactions of the ASME: Journal of Dynamic Systems, Measurement and Control. Vol. 102, June 1981. pp. 126-133

[Ramamritham, Stankovic 94]
Ramamritham, K.; Stankovic, J. A.: Scheduling Algorithms and Operating Systems Support for Real-Time Systems. Proceedings of the IEEE, Vol. 82, No. 1, Jan. 1994, pp. 55-67

[Regan 99]
Regan, T.: Application Note 694 A DMOS 3A, 55V, H-Bridge: The LMD18200. National Semiconductor Corporation, 1999. http://www.national.com/an/AN/AN-694.pdf

[Reinisch 82]
Reinisch, K.: Analyse und Synthese kontinuierlicher Steuerungssysteme. Verlag Technik, Berlin, 1982

[Riefenstahl 00]
Riefenstahl, U.: Elektrische Antriebstechnik. In Schriftenreihe: Leitfaden der Elektrotechnik. Stuttgart [u.a.]: Teubner, 2000. ISBN 3-519-06429-4

[Robo 01]
The RoboCup Federation: RoboCup-Rescue Robotics and Infrastructure Evaluation. Methods. http://www.robocup.org/games/36.html

[Rosheim 94]
Rosheim, M. E.: Robot Evolution: The Development of Anthrobotics. John Wiles & Sons, Inc., New York, Chichester u.a., 1994. ISBN 0-471-02622-0

[rti 99]
ControlShell: The Tool for Intelligent Control. A White Paper by Real-Time Innovations, Inc., http://www.rti.com/products/controlshell/CSMessageOct99.pdf, September 1999

[Rygg 1893]
Rygg, L. A.: Mechanical Horse. US Patent No. 491,927, Class 280/1.181; 472/99

[Schmucker et al. 94]
Schmucker, U; Schneider, A.; Ihme, T.: Sechsbeiniger Schreitroboter mit Kraftrückführung - Konzeption und erste Ergebnisse, 1. Brandenburger Workshop Mechatronik, Brandenburg, November 1994, SS. 45-53.

[Schmucker et al. 96a]
Schmucker, U.; Schneider, A.; Ihme, T.: Hexagonal walking vehicle with force sensing capability, ISMCR '96, Brussels, 9-11 May 1996, Proceedings, pp. 354-359

[Schmucker et al. 96b]
Schmucker U., Schneider A., Ihme T.: Sechsbeiniger kraftadaptiver Schreitroboter. In: 2. Brandenburger Workshop Mechatronik, Tagungsband. 26.-27. September 1996

[Schmucker et al. 00]
Schmucker, U; Schneider, A.; Ihme, T.: Multisensorielle Verfahren zur Bewegungssteuerung sechsbeiniger Schreitroboter. Abschlußbericht für den Projektzeitraum von Oktober 1997 bis September 1999

[Schneider et al. 95]
Schneider, A.; Schmucker, U.; Ihme, T.; Devjanin, E.; Savitsky, K.: Force control in locomotion of legged vehicle and body movement for mounting operations, Ninth World Congress on the Theory of Mechanics and Mechanisms, Milano, Italy, August 29 - September 2 1995, Proceedings, v.3, pp. 2363-2367.

[Schneider et al. 96]
Schneider, A.; Schmucker, U.; Ihme, T.; Devjanin, E.; Savitsky, K.: Force Control in Locomotion of Legged Vehicle and for Service Operations, 2nd World Automation Congress, Montpellier, France, 27-30 May 1996.

[Scholl et al. 00]
Scholl, K.-U.; Kepplin, V.; Albiez, J.; Dillmann, R.: Developing Robot Prototypes with an Expandable Modular Controller Architecture. Proceedings of the International Conference on Intelligent Autonomous Systems, Venice 2000, pp 67-74

[Schrüfer 95]
Schrüfer, E.: Elektrische Meßtechnik. Messung elektrischer und nichtelektrischer Größen. Carl Hanser Verlag, Wien, München, 1995. ISBN 3-446-17955-0

[Sciavicco, Siciliano 00]
Sciavicco, L.; Siciliano, B.: Modelling and Control of Robot Manipulators. (Advanced Textbooks in Control and Signal Processing). London [u.a.] : Springer, 2000. ISBN 1-85233-221-2

[Seifart 87]
Seifart, M.: Analoge Schaltungen. Verlag Technik, Berlin, 1987. ISBN 3-341-00091-7

[Shannon 93]
Shannon, C. E.: Communication in the Presence of Noise. In: Cloane, N. J. A.; Wyner, A. D. (Eds.): Claude Elwood Shannon: Collected Papers. Piscataway, NJ. IEEE Press, 1993, pp. 160-172. ISBN 0-7803-0434-9

[Shutter et al. 98]
De Shutter, J.; Bruyninckx, H; Żhu, W; Spong, M. W.: Force Control: A Bird's Eye of View. In: Control problems in robotics and automation. Lecture notes in control and information sciences; 230. London [u.a.]: Springer, 1998

[Sinha, Bajcsy 92]
Sinha, P. R.; Bajcsy, R. K.: Robotic Exploration of Surfaces and its Application to Legged Locomotion. Proceedings of the 1992 IEEE International Conference on Robotics and Automation. Nice, France. May, 1992. pp. 221-226

[Spenneberg, Kirchner 00]
Spenneberg, D.; Kirchner, F: Omnidirectional Walking in an Eight Legged Robot. In: Proceedings of the 2nd International Symposium on Robotics and Automation (ISRA2000), Monterrey, N.L. , Mexico, 10.11. - 12.11.2000

[Stankovic et al. 98]
Stankovic, J. A.; Spuri, M.; Ramamritham, K.; Buttazzo, G. C.: Deadline Scheduling for Real Time Systems. In Serie: The Kluwer International Series in Engineering and Computer Science; SECS 460 Real Time Systems. Kluwer Academic Publishers, Boston u.a., 1998. ISBN 0-7923-8269-2

[Stankovic, Ramamritham 92]
Stankovic, J. A.; Ramamritham, K.: The Spring Kernel: A New Paradigm for Real-Time Systems. In: Stankovic, J. A.; Ramamritham, K. (Eds.): Advances in Real-Time Systems. IEEE Computer Society Press, Los Alamitos, California, USA, 1992. pp. 247-257. ISBN 0-8186-3792-7

[Streich 95]
Streich, H.: TaskPair-Scheduling: An Approach for Dynamic Real-Time Systems, Int. Journal of Mini & Microcomputers, Vol. 17, No. 2, pp 77-83, 1995

[Stroppe 88]
Stroppe, H.: Physik für Studenten der Natur- und Technikwissenschaften. Fachbuchverlag, Leipzig, 1988. ISBN 3-343-00182-1

[Stout 01]
Stout, B.: The SNIPPETS C/C++ Source Code Collection. 2001.
http://www.snippets.org/snippets/home.php3

[Tanenbaum 96]
Tanenbaum, A. S.: Computer networks. Prentice Hall, UK, 1996. ISBN 0-13-394248-1

[TI 98]
SN75179B DIFFERENTIAL DRIVER AND RECEIVER PAIR. Datasheet SLLS003E – October 1985 – revised June 1998. Texas Instruments Incorporated, 1999.
http://www-s.ti.com/sc/psheets/slls003e/slls003e.pdf

[TI 99]
SN65176B, SN75176B Differential Bus Transceivers. Datasheet SLLS101B – July 1985 – revised June 1999. Texas Instruments Incorporated, 1999.
http://www-s.ti.com/sc/psheets/slls101b/slls101b.pdf

[Tietze, Schenk 99]
Tietze, U., Schenk, Ch.: Halbleiter-Schaltungstechnik. Springer-Verlag, 1999. ISBN 3-540-64192-0

[Todd 85]
Todd, D. J.: Walking machines: an introduction to legged robots. Kogan Page, London, 1985. ISBN 0-85038-932-1

[Turkowski 94]
Turkowski, K.: Fixed Point Square Root. Apple Technical Report No. 96, October 1994

[Unbehauen 89]
Unbehauen, H.: Regelungstechnik I: Klassische Verfahren zur Analyse und Synthese linearer kontinuierlicher Regelungssysteme. 1989

[Volpe, Khosla 95]
Volpe, R.; Khosla, P.: The Equivalence of Second Order Impedance Control and Proportional Gain Explicit Force Control. International Journal of Robotics Research, Vol. 14, No. 6, December, 1995. pp. 574-589

[Waldron, McGhee 86]
Waldron, K. J.; McGhee, R. B.: The Adaptive Suspension Vehicle. IEEE Control Systems Magazine. 1986. pp. 7-12

[Whitney 87]
Whitney, D. E.: Historical Perspective and State of the Art in Robot Force Control. International Journal of Robotics Research. Vol. 6 No. 1, 1987. pp. 3-14

[Wloka 92]
Wloka, D. W.: Robotersysteme. 1: Technische Grundlagen. Berlin [u.a.] : Springer, 1992, 271 S., ISBN 3-540-54739-8

[Xu et al. 94]
Xu, Y.; Brown, B.; Aoki, S.; Kanade, T.: Mobility and manipulation of a light-weight space robot. In: Robotics and Autonomous Systems 13, 1994, S.1-12

[Yoneda et al. 97]
Yoneda, K.; Suzuki, K.; Kanayama, Y; Takahashi, H.; Akizono, J.: Gait and Foot Trajectory Planning for Versatile Motion of a Six-Legged Robot. Journal of Robotic Systems. Vol. 14, No. 2, 1997, pp. 121-133

[Yoneda 01]
Yoneda, K.: Design of Non-Bio-Mimetic Walker with Fewer Actuators. In: Berns, K.; Dillmann, R. (eds.): Proceedings of the Fourth International Conference on Climbing and Walking Robots (CLAWAR 2001) – From Biology to Industrial Applications. 24-26 September 2001, Karlsruhe, Germany. Professional Engineering Publishing, Bury St Edmunds (u.a.), UK, 2001. pp. 115-126. ISBN 1-86058-265-2

[Zielinska 01]
Zielinska, T.: Synthesis of Control System – Gait Implementation Problems. In: Berns, K.; Dillmann, R. (eds.): Proceedings of the Fourth International Conference on Climbing and Walking Robots (CLAWAR 2001) – From Biology to Industrial Applications. 24-26 September 2001, Karlsruhe, Germany. Professional Engineering Publishing, Bury St Edmunds (u.a.), UK, 2001. pp. 489-496. ISBN 1-86058-265-2

[Zöbel, Albrecht 95]
Zöbel, D.; Albrecht, W.: Echtzeitsysteme: Grundlagen und Techniken. International Thomson Publishing, Bonn (u.a.), 1995. ISBN 3-8266-0150-5

## Verzeichnis der Abbildungen

## Verzeichnis der Tabellen

## Lebenslauf

### Personalien

| | |
|---|---|
| Name: | Thomas Ihme |
| Geburtstag: | 21.11.1968 |
| Geburtsort: | Magdeburg |
| Familienstand: | verheiratet, 1 Kind |

### Bildungsweg

| | |
|---|---|
| 09/75 – 08/85 | Polytechnische Oberschule, Magdeburg<br>Abschluss: 10. Klasse |
| 08/85 – 10/87 | Lehre zum Elektronikfacharbeiter bei den Magdeburger Armaturenwerken<br>Abschluss: Facharbeiter |
| 09/88 – 08/90 | Vorkurs für junge Facharbeiter zur Erlangung der Hochschulreife an der Ingenieurschule Berlin<br>Abschluss: Hochschulreife |
| 09/89 – 08/90 | Studium der Automatisierungstechnik an der Ingenieurschule Berlin |
| 09/90 – 11/94 | Studium der Elektrotechnik an der Otto-von-Guericke-Universität Magdeburg<br>Abschluss: Diplom-Ingenieur |
| 11/00 – 03/02 | Doktorand an der Fakultät für Informatik der Otto-von-Guericke-Universität Magdeburg |

### Berufstätigkeit

| | |
|---|---|
| 11/87 – 08/88 | Magdeburger Armaturenwerke<br>Elektronikfacharbeiter im Bereich Produktentwicklung |
| 12/94 – 12/97 | Tätigkeit für das Fraunhofer Institut für Fabrikbetrieb und –automatisierung Magdeburg, teilweise in Kooperation mit der Otto-von-Guericke-Universität Magdeburg<br>Wissenschaftlicher Mitarbeiter in der Abteilung Automatisierung |
| 01/98 – 08/99 | Otto-von-Guericke-Universität Magdeburg, Fakultät für Maschinenbau, Institut für Arbeitswissenschaften, Fabrikautomatisierung und Fabrikbetrieb<br>Wissenschaftlicher Mitarbeiter im Rahmen des DFG-Schwerpunktprogramms „Autonomes Laufen“ |
| seit 09/99 | Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik, Institut für Verteilte Systeme, Arbeitsgruppe Echtzeitsysteme und Kommunikation<br>Wissenschaftlicher Mitarbeiter |