Michael Jahn

Ein Beispiel zur Entwicklung kooperierender mobiler Roboter

Ein Beispiel zur Entwicklung kooperierender mobiler Roboter

Konstruktives Design und Steuerungsentwurf

Von Michael Jahn

Universitätsverlag Ilmenau
2010

Impressum

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

Dieser Publikation liegt die Arbeit mit dem Titel „Ein Beitrag zur Entwicklung kooperierender mobiler Roboter“ [urn:nbn:de:gbv:ilm1-2009000169] zu Grunde, die der Fakultät für Maschinenbau der Technischen Universität Ilmenau als Dissertation vorgelegen hat.

Tag der Einreichung:	12. Januar 2009
1. Gutachter:	Univ.-Prof. Dr.-Ing. habil. Klaus Zimmermann (Technische Universität Ilmenau)
2. Gutachter:	Univ.-Prof. Dr.-Ing. habil. Mathias Weiß (Technische Universität Ilmenau)
3. Gutachter:	Prof. Dr. sc. techn. Harald Loose (Fachhochschule Brandenburg)
Tag der Verteidigung:	20. Mai 2009

Technische Universität Ilmenau/Universitätsbibliothek
Universitätsverlag Ilmenau
Postfach 10 05 65
98684 Ilmenau
www.tu-ilmenau.de/universitaetsverlag

Herstellung und Auslieferung
Verlagshaus Monsenstein und Vannerdat OHG
Am Hawerkamp 31
48155 Münster
www.mv-verlag.de

ISBN 978-3-939473-73-2 (Druckausgabe)
urn:nbn:de:gbv:ilm1-2010100068

Titelfoto: Barbara Neumann | Erfurt

"Don't be afraid to take a big step if one is indicated.
You can't cross a chasm in two small jumps."
~David Lloyd George~
(ehem. britischer Premierminister)

Kurzfassung der Dissertation

In dieser Arbeit werden die wesentlichen Ergebnisse zur Konstruktion des RoboCup-Small-Size-Roboters LUKAS und des Aufbaus der koordinierenden Mastersoftware XBase vorgestellt. Für den Roboter wurden technische Lösungen gefunden, die einsatztaugliche Fahrleistungen durch einen verminderten Materialeinsatz und einen hohen Anteil eigener Fertigung erzielt.

Die erreichte Kompaktheit und das notwendige Masse-Leistungs-Verhältnis beruhen im Wesentlichen auf der orthogonalen Achskonfiguration. Spezielle Material- und Strukturlösungen für Leichtbau und Bauraumnutzung finden sich im Polyamid-Chassis, im motorintegrierenden Dribbler und im energieeffizienten Schussmechanismus. Im Einsatz sind omnidirektionale Räder mit solider Rundlauf-, Traktions- und Stabilitätseigenschaft. Robust und kompakt integriert die spezielle elektronische Hauptplatine alle benötigten Steuerungskomponenten. Die galvanische Trennung der Steuer- und Lastseite garantiert Betriebssicherheit und längere Laufzeit.

Der dreirädrige Roboter setzt ein hierarchisches Konstruktionskonzept um, welches hohe Zuverlässigkeit, Flexibilität und Robustheit in verstärktem Maße durch technische Funktionsintegration einer minimalen Anzahl von Bauteilen realisiert.

Nach Analyse der kinematischen und dynamischen Zusammenhänge konnte für den Roboter eine Reglerarchitektur entwickelt werden, die unter Berücksichtigung von Radschlupf und Schwerpunktverlagerung die omnidirektionale Manövrierbarkeit und Bewegungspräzision gestattet. Die lokale Positionsregelung des Roboters ist ein wesentliches Element des Controllerprogramms und zugleich Basis des Konzeptes einer bahngeführten Positionierung. Die spezielle Bahnsteuerung, u.a. nach dem Prinzip Lissajousscher Figuren, erlaubt die halbautomatische Verhaltenssteuerung des Roboters und eine Entlastung der Kommunikationskanäle.

Die weiterentwickelte Mastersoftware XBase verbindet eine modifizierten Bildverarbeitung zur Detektierung der Roboter als mobile Feldobjekte und

eine anwenderfreundliche Mensch-Maschine-Schnittstelle mit dem Instrument zur Roboterkoordinierung. Stabile Bildraten und geringer Berechnungsaufwand werden durch die Verwendung einer Look-Up-Tabelle zur Maximum-Likelihood-Farbklassifizierung, Bildsegmentierung mit Zeilenkoinzidenzverfahren und ressourcenschonende Operationen garantiert. Die Genauigkeit der akquirierten Daten steigerte sich gegenüber der Vorläuferversion durch den Beitrag eines Butterfly-Robotertrikots.
Ein geeigneter Trainingseinstieg wurde mit verschiedenen Formationsvarianten zur koordinierten Kontrolle einer Robotergruppe gefunden. Der strukturelle Aufbau der Formation wird durch einen Datensatz polarer Bindungsparameter beschrieben. Das Modell gestattet durch individuelle Freigabe von Freiheitsgraden einen Übergang vom starren zum flexiblen Verband. Anhand der Beispielformationen Block, Reihe und Rotte ist die Flexibilität und Wandlungsfähigkeit dargestellt. Das System XBase kann sowohl als Multi-Roboter-System unter der Kontrolle einer Verhaltenssteuerung, wie auch als verteiltes Multi-Agenten-System mit mehreren parallelen Verhaltenssystemen fungieren.

Abstract of the Thesis

This work presents the main outcomes of the design of LUKAS, a robot in the RoboCup Small Size League, and of the development of a structure for the coordinating software, XBase. Technical solutions were found, enabling the robot to perform competitively because use of material was reduced, and, of that material, a high proportion was made by the author and his team.

The fact that both compactness and the necessary mass-power ratio were achieved is mainly due to the configuration: three wheels and an orthogonal axle. The polyamide chassis, the backspin engine integrated into the dribbler, and the energy-efficient goal shooting mechanism, all make use of special ideas for material and structure, so that the three wheeled robot is light in weight and makes good use of constricted space. It has omni-directional Stanford wheels which guarantee smooth running, traction and stability on a steady basis. Particular care was taken in the development of a special electronic mainboard providing robust integration of all the active control components. Operational reliability and extended operational life are ensured by galvanic separation of the control and the motor sides. The design is basically sandwich construction.

This is a means of using the integration of technical functions and a minimal number of components to ensure high reliability, flexibility and robustness.

It was possible to develop control architecture for the robot on the basis of kinematic and dynamic mathematical analysis so that omni-directional manoeuvring and precise positioning are permitted because wheel slippage and changes in the centre of gravity are registered. In the robot's controller program, the local positioning regulation system plays an important role, at the same time providing the basic scheme for path-controlled positioning. This path-controlled positioning relies in part on the principle of Lissajous figures, permits semiautomatic control of the robot behaviour and relieves the communication channels.

The master software, XBase, has been developed to combine a user-friendly man-machine interface with a toolbox, to enable all the robots' systems to be coordinated and moving objects in the field to be detected by customised image processing. A look-up table with maximum-likelihood colour classification is one of the means by which stable picture rates and economical computation are achieved. Others are picture segmentation with line coincidence procedure and resource-protective operations. In the course of development, the accuracy of the data captured was increased over that of the initial version by incorporating a butterfly-shaped patch on the robot team's jersey.

For a group of robots, suitable training exercises have been devised using different formations so that they can be controlled in a coordinated way. The structure of the formation is determined by a data set containing the polar bonds. This model for the formation permits transition, by individual release of parameters, from a rigid to a flexible network. The flexibility and adaptability of the robot group is demonstrated using three types of formation, bloc, line and rout.

The XBase system devised will run either as a multi-robot system managed by a single behaviour control system, or as a multi-agent combination managed by several parallel behaviour controls.

Vorwort

Die vorliegende Dissertation entstand an der Fakultät für Maschinenbau der Technischen Universität Ilmenau in Zusammenarbeit mit dem Fachgebiet Technische Mechanik und dem Fachgebiet Rechneranwendung im Maschinenbau.

Herrn Univ.-Prof. Dr.-Ing. habil. Klaus Zimmermann danke ich für die wohlwollende Betreuung und die Förderung, welche eine Realisierung dieser Arbeit erst ermöglichten. Seine Erfahrung und Anregung waren ein Ansporn und große Hilfestellung.
Herrn Univ.-Prof. Dr.-Ing. habil. Mathias Weiß und Frau Dr.-Ing. Marion Braunschweig bin ich gleichsam ausgesprochen dankbar für ihre investierte Zeit, die freundliche und wertvolle Unterstützung sowie die kreative Freiheit, die sie mir als Koordinator und Betreuer des Robo-Cup-Projektes gewährten.

Für die fruchtbare Zusammenarbeit danke ich sehr Dipl.-Ing.(FH) Hans-Peter Walkling, Dipl.-Ing. Peter Landsmann, Dipl.-Ing. Steffen Lerm, Dipl.-Ing. Christoph Ußfeller, Dr. Thorsten Rieß, Dipl.-Ing. Tilman Wimmer und Dipl.-Ing. Dennis Füchsel, welche alle mit Ideen und Eigeninitiative wesentliche Bereiche des Projektes mitgestalteten, bereicherten und vorantrieben.

Gleichfalls gilt mein Dank Dr.-Ing. Erik Gerlach und Dr.-Ing. Dipl.-Math. Carsten Behn für ihre fachlichen Beratung und die eingehende Durchsicht des Manuskriptes.
Meine Arbeit am Promotionsvorhaben wurde vor allem im Rahmen der Landesgraduiertenförderung mit Mitteln des Freistaates Thüringen gefördert. Für die finanzielle Unabhängigkeit und Entlastung innerhalb des Förderzeitraumes von Oktober 2005 bis November 2007 bin ich außerordentlich zu Dank verpflichtet.

Dipl-Chem. Burkhard Jahn und Dr. rer. pol. Enrico Schöbel sei für den Erfahrungsaustausch im Verlauf der Promotion, Dipl.-Trophologin Ricarda Knetsch und M.A. Gilda Barthel für das gründliche Korrekturlesen des Manuskriptes gedankt. Die Arbeit konnte auch nicht durch die Zuwendung und das Verständnis aus dem Kreis der Familie entstehen. Ich bedanke mich bei meiner Ehefrau, meinen Eltern und Großeltern.

Gewidmet sei die Arbeit meinem Sohn Lukas. Der in dem Zeitraum seine ersten Schritte wagte, in dem sich auch für einen kleinen Roboter neue Wege der Mobilität eröffneten.

Ilmenau, den 12.01.2009 Michael Jahn

Inhaltsverzeichnis

Verwendete Symbole und Begriffsbestimmung

Symbole und Variablen

Im Sinne einer verbesserten Übersichtlichkeit und Zuordnung sind nachfolgend die Symbole und Variablen unter der Abbildung bzw. Gleichung genannt, mit der sie inhaltlich im Zusammenhang stehen. Aufgrund des interdisziplinären Charakters dieser Arbeit zwischen den Gebieten der Technischer Mechanik, Elektrotechnik, Regelungstechnik und Informatik wurde mehrfach die Bezeichnung und Symbolik gewählt, wie sie auch in Veröffentlichungen des thematischen Umfeldes der mobilen Robotik und des RoboCup zu finden ist [80].

[nach Abb. 3.15 Systembeschreibung zur kinematischen Analyse, Seite 58]

ω_i - Winkelgeschwindigkeit Rad i, i=1,2,3
φ - Rotationswinkel
$\vec{e}_{X0}, \vec{e}_{Y0}$ - raumfestes Koordinatensystem
$\overrightarrow{E_x}, \overrightarrow{E_y}$ - körperfestes Koordinatensystem
a, b - Radabstände zum lokalen Ursprung
r - Radradius
$\mathbf{X}, \mathbf{x}$ - Vektoren der Position und Orientierung
$\boldsymbol{\omega}$ - Vektor der Radwinkelgeschwindigkeiten

[nach Abb. 3.17 Elektromechanisches Modell eines Antriebsstranges, Seite 63]

u - Eingangsspannung eines Motors
ω_m - Motorwinkelgeschwindigkeit
e - induzierte Gegenspannung
$\ddot{u}$ - Getriebeübersetzung
i_a - Ankerstrom
ω - Radwinkelgeschwindigkeit

R_a - Ankerwiderstand

m_b - Beschleunigungsmoment

L_a - Ankerinduktivität

m_L - Lastmoment

k_e, k_m - Motorkonstanten

m_a - Antriebsmoment

k_d - Dämpfungskonstante des Motors

m_d - Dämpfungsmoment

J_g - Trägheitsmoment von Anker und Last

f_r - Betrag der Radantriebkraft

v_b - Bodengeschwindigkeit des Rades

r - Radradius

v_r - Radumfangsgeschwindigkeit

[nach Abb. 3.18 Radkräfte und Lage des Schwerpunktes, Seite 65]

F_{ri} - Radantriebskraft

F_{zi} - Radaufstandskraft

f_{ri} - Betrag der Radantriebskraft

f_{zi} - Betrag der Radaufstandskraft

S - Roboterschwerpunkt

g - Schwerkraftbeschleunigung

M - Robotermasse

J - Masseträgheitsmoment des Roboters

F_S - Betrag der resultierenden Kraft an S

M_{SZ} - Betrag des resultierenden Drehmoments an S

$\mathbf{v_b}$ - Vektoren der Bodengeschwindigkeit

$\mathbf{u}$ - Vektor der Motorspannungen

$\mathbf{v_r}$ - Vektoren der Umfangsgeschwindigkeit

$\mathbf{f_r}$ - Vektor der Radantriebskräfte

[nach Abb. 3.20 Messergebnisse praktischer Versuche und Schlupfkurve vom Prototyp, Seite 69]

$\mu_{\max}$ - Haftreibungsgrenze

f_{Ci} - Betrag der Coulomb'schen Reibkraft eines Rades

$\mathbf{f}_Z$ - Vektor der Radaufstandskräfte

$\mathbf{f}_C$ - Vektor der Coulomb'schen Reibkräfte

$\boldsymbol{\mu}$ - Vektor der Rad-Haftreibungskoeffizienten

$\mathbf{E}$ - Einheitsmatrix

s - Radschlupf

[nach Formel (4.1) Seite 97]

$d(\vec{x}, \vec{y})$ - Mahalanobis-Distanz

S^{-1} - Inverse Kovarianzmatrix der Farbklasse

$\vec{x}$ - RGB-Farbraum-Tripel

$\vec{y}$ - Schwerpunkt-Tripel der Farbklasse

[nach Formel (4.3) Seite 111]

SP - Formations-Koordinatenschwerpunkt

X_i - Robotermittelpunkt

x, y - Koordinaten-Variablen

i - Zählvariable

Begriffsbestimmung

Die Verwendung diverser Begriffe entstand aus den entwickelten Gegebenheiten des Projektes, weshalb für die wesentlichen Grundbegriffe zur Beschreibung der Systemeigenschaften eine eindeutige Zuordnung nachfolgt.

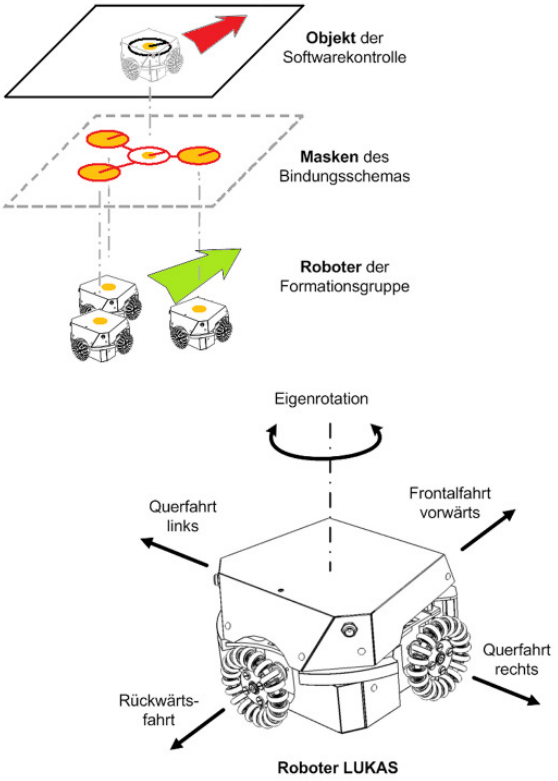


Abb. Grundbegriffe der Hard- und Software

Abkürzungsverzeichnis

CAD	Computer Aided Design
CAN	Controller Area Network
CCD	Charged Coupled Device
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DSS	Dynamics-Safety-Search
ERRT	Execution-Extended-Rapidly-Exploring-Random-Tree
FIFA	Fédération Internationale de Football Association
FPGA	Field Programmable Gate Array
GPS	Global Positioning System
GUI	Graphical User Interface
HSI	Hue-Saturation-Intensity
ID	Identitätsnummer
IR	InfraRot
KI	Künstliche Intelligenz
MFC	Microsoft Foundation Classes
MOSFET	Metall-Oxid-Halbleiter-Feldeffekttransistor
NTSC	National Television Systems Committee
PAL	Phase Alternating Line
PID	Proportional-Integral-Derivative
PWM	Pulsweitenmodulation
RAW	Rohdatenformat bei Digitalkameras
RGB	Rot-Grün-Blau
RLE	Run-Length-Encoding
SNAP	Scaleable Node Address Protocol
SRAM	Static Random Access Memory
SSL	Small-Size-League
STP	Skills-Tactics-Plans
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WLAN	Wireless Local Area Network
YUV	Helligkeits-Farbdifferenz-Konzept

1 Einleitung

1.1 Motivation

Medien aller Art müssen fließen, damit Systeme und Anlagen fähig sind, zu agieren. Koordinierte Aufnahme, Umsatz und Weitergabe von Material, Energie oder Information sind Grundlage zur zweckdienlichen Erfüllung von Aufgaben. Neben den klassischen Naturwissenschaften befassen sich im Computerzeitalter auch die Wissenschaften Mathematik, Informatik und Fachrichtungen wie Biomechanik, Bioinformatik und Bionik mit den Gesetzen und evolutionären Lösungen der belebten Natur. Im Blickpunkt steht nicht zuletzt auch das kollektive Schwarmverhalten von Organismen. Erkenntnisse auf diesem Gebiet sind nötig, um z.B. produktionstechnische Fließprozesse zu optimieren, zeitintensive Blockaden in der Infrastruktur moderner Großstädte zu vermeiden oder kollektive autonome Maschinenstrukturen aufzubauen.
Die Verfügbarkeit leistungsfähiger Computer-, Aktor- und Sensortechnik ermöglicht in der Robotik die technische Realisierung komplexer Vorlagen und Verhaltensweisen aus der natürlichen Umwelt. Zur Lösung definierter Schwarmaufgaben können reale Prozesse abstakt technisch nachempfunden werden, wenn sie die Methoden und Prinzipien der Künstlichen Intelligenz (KI) nutzen [1]. Auf Basis von Anwendung agentenbasierter Modelle in Computersimulationen schreiten methodische Nachbildungen und das Verständnis der Kooperationsmechanismen voran [2, 3]. Der Vormarsch interagierender maschineller Assistenten unter dem Oberbegriff 'Roboter' ist kaum mehr aufzuhalten [4].
Stationäre Gruppen von kooperierenden Taktstraßen-Robotern verrichten schon heute flexible Handhabungs- und Produktionsaufgaben. Der Durchbruch der Teamarbeit im mobilen Robotikbereich ist ebenfalls zu erwarten, weil die Zukunft ohnehin im Zeichen des intelligenten Roboterservice gesehen wird [5] und im privaten wie im öffentlichen Sektor gleichermaßen ein Interesse an effektiven kooperierenden Technologien besteht.

Im Fokus dieser Arbeit steht in diesem Sinne der Aufbau eines aktiven Basissystems, in dem sich individuelle Roboterobjekte kollektiv bewegen und handeln. Ballspielenden Robotern oder steuerbaren amorphen Schwarmstrukturen als künstliche Flüssigkeit sind eine technische Herausforderung. Durch das Zusammenführen und die Kombination des Wissens ist die Robotik als interdisziplinäres Forschungsfeld seit ihrer Entstehung eine Quelle neuer Visionen und macht die Mitarbeit in diesem Bereich der Technik sehr vielfältig.

1.2 Überblick

In der Arbeit werden Konstruktion und Inbetriebnahme des Roboters LUKAS mit dem Schwerpunkt Design und Ausbau der zentral koordinierenden Software XBase vorgestellt.
Einleitend wird ein allgemeiner Blick auf die Zielrichtung und die Historie innerhalb des Projektes geworfen, um dem Leser das Umfeld dieser Arbeit zu verdeutlichen. Unter diesem Aspekt ist auch das nachfolgende Kapitel zum Stand der Technik zu verstehen. In der Kurzübersicht zur Robotik, der Forschung zur Künstlichen Intelligenz und des RoboCup werden die für das allgemeine Vorwissen relevanten Informationen dargestellt. Auf eine umfassende, allgemeine Aufzählung und Gegenüberstellung technischer Besonderheiten der RoboCup-Small-Size-League (SSL) wird verzichtet. Um den Leser jedoch einen knappen, aber vertiefenden Blick auf aktuelle Systeme zu geben, stehen stellvertretend die Teams der CMDragons und der FU-Fighters im Vordergrund. Die dreiteilige Systematik zur Analyse der Hardware-, Bildverarbeitungs- und Verhaltens-Komponenten dient weiterhin als Leitfaden in der Beschreibung des eigenen Systems.
Den Hauptteil der Arbeit leitet das Kapitel über den Roboter LUKAS ein. Umfassend sind die speziellen Richtlinien und die daraus entwickelten Lösungen der technischen Komponenten dargestellt. Chronologisch folgt die Ausarbeitung den Betrachtungen zur kinetischen und dynamischen Modellierung.

Aufbauend auf diesen mathematischen Zusammenhängen wird danach der Entwurf der roboterinternen Steuerungssoftware RoboCon mit seinen besonderen Merkmalen beschrieben. Im zweiten Teil folgt einleitend der Überblick des modularen Softwareprogramms XBase. Anforderungen und Merkmale der zentralen Anlagensteuerung werden dargelegt, um eine vollständige Betrachtungsweise zu vermitteln. Bezüglich Bildverarbeitung und Verhaltenssteuerung legen verschiedene Abschnitte die Neuerungen und eigenen Ansätze dar. Abschließend steht ein Ausblick mit verstärktem Augenmerk auf weitere Entwicklungsmöglichkeiten kooperierender Roboter und Robotermodule anhand semi-flexibler bzw. amorpher Formationssteuerung.

1.3 Zielrichtung und Konzept

„Der RoboCup[1] gilt als Benchmark der KI-Forschung“ ist eine oft zitierte Formulierung, welche nicht nur auf die Simulationsligen zutrifft [6].
Seit der Wettkampf, beginnend mit den Small-Size-Robotern, in realer physischer Interaktion ausgetragen wird und die Ligen die wichtigsten Fähigkeitsbereiche intelligenter Serviceroboter abdecken, kann der RoboCup ebenso als ein Vertreter der zivilen Servicerobotik angesehen werden.
Hard- und Software bilden eine Einheit, deren Symbiose die gegenseitige Weiterentwicklung fördert. Der permanente Fortschritt in den mechatronischen Komponenten und der Leistungsfähigkeit der verwendeten technischen Systeme bietet dem Publikum und den Mitwirkenden des Wettkampfes zusätzliche Attraktivität.
Als Open-Source-Projekt ist es im RoboCup grundsätzlich erwünscht, das Know-How führender Teams zur Orientierung und Anwendung zu übernehmen. Aktuelle Teilnehmerkonfigurationen sind häufig das Ergebnis neuester individueller Entwicklungen unter Angleichung und Einarbeitung technischer Entwicklungen freigegebener Siegerplattformen.

[1] http://www.robocup.org/

Das Ilmenauer Experimentalsystem strebt eine angepasste Umsetzung aktueller Techniken nach dem Vorbild der RoboCup Small-Size-League an. Mit differenziertem Hardwareaufwand werden die Herausforderungen zum Einstieg in die praktische mobile Robotik absolviert [7, 8]. Aus dem Anspruch der Individualität und dem Ehrgeiz der eigenen Qualifikation kam hier die Möglichkeit der Übernahme von direkten Vorlagen oder des Kaufes von Fertigrobotern jedoch nicht in Frage. Bewusst wurden grundsätzlich eigene Ansätze und Lösungswege verfolgt [9]. Die Zielstellung dieser Dissertation ist somit im Gesamtprojekt gegeben und eröffnet ein breites Spektrum der zu bearbeitenden Themenfelder in den beiden Schwerpunkten:

- **Roboterhardware**
  Konstruktion und Inbetriebnahme. Ein neuer, robuster, leichter und omnidirektionaler Roboter soll nach Kriterien der Small-Size-League den praktischen Anforderungen des Feldeinsatzes gewachsen sein. Der Einsatz soll nicht nur als elementarer Ballspieler im homogenen Team, sondern in Erweiterung auch mit universellen Verwendungsmöglichkeiten als modifizierte, heterogene Plattform und Träger anderer Aktoren und Sensoren im Einzel- oder Formationsbetrieb erfolgen.

- **Anlagensoftware**
  Design und funktionale Erweiterung. Eine echtzeittreue Multi-Roboter-Kontrollstruktur soll die Fähigkeiten der Roboterobjekte auf dem Spielfeld unterstützen. Dies beinhaltet im Grundstadium die Kontrolle freier Formationsbewegungen und darauf aufbauend die Umsetzung teambasierter Ballspiele. Während das stationäre Masterprogramm der globalen Überwachung die Modulebenen Bildverarbeitung, Objektverwaltung, Verhaltensteuerung und Kommunikation umfasst, ist die mobile Software für die lokale Bahnsteuerung und Motorregelung der Roboter zuständig.

Beide Hauptbereiche werden vornehmlich aus der Perspektive des angewandten Maschinenbaus betrachtet und tragen deshalb verstärkt praktischen, funktionalen Charakter.
Einzelne automatische und halbautomatische Verhaltensweisen wurden schon in der aktuellen Softwarestruktur integriert, so dass die Funktionalität der Roboterobjekte, die Leistungsfähigkeit und die reibungslose Zusammenarbeit aller Komponenten für Trainings- und Entwicklungszwecke verfügbar sind. Strategische Intelligenz und vertiefende Lernverfahren können aufbauend implementiert werden und sind der Inhalt weiterführender Entwicklungen des Ballspieles, jedoch nicht Teil dieser Arbeit.

1.4 Historie des Projektes

Das Fachgebiet Rechneranwendung im Maschinenbau und das Fachgebiet Technische Mechanik der Fakultät für Maschinenbau an der TU Ilmenau betreuen gemeinsam seit acht Jahren eine Laboranlage in Adaptation der RoboCup-Small-Size-League.
Für das im Sommer 2006 gegründete Ilmenauer KickElan-Team wurde durch die jüngste Zusammenarbeit ein neues Multi-Roboter-System bereitgestellt. Das System basiert auf eigener Entwicklung der Software, sowie Roboterhardware und ist im aktuellen Stadium in der Lage, Formationsbewegungen mit mehreren Robotern und einfaches Ballspiel auszuführen.
Zu den wesentlichen Systemkomponenten der stationären Anlage gehört das Spielfeld mit einer darauf ausgerichteten Kamera, als auch ein Zentralcomputer inklusive Framegrabber und Funkmodem [10]. Mittels der Mastersteuerung XBase ist die interaktive Positionierung am Kamerabild, das Abfahren von Pfadlinien und eine Torschussaktion mit Robotern der alten Generation ausführbar.
Neben der Entwicklung und Optimierung des Roboters LUKAS bestand das Bemühen in der Leistungssteigerung der Programmplattform XBase, der Handsteuerung Comtrol sowie der Controllersoftware RoboCon.

Nachdem grundlegende Mechanismen zur Kontrolle der einzelnen Verhaltensaktionen im Ballspiel erstellt und erprobt waren, konnte damit begonnen werden, die Architektur für flexible Formationsbewegungen zu erweitern. Zu diesem Zeitpunkt stand für praktische Erprobungen jedoch nur ein Prototyp der neuen Roboterserie zur Verfügung, so dass eine zusätzliche Simulatorebene im Hauptprogramm XBase integrierte wurde, die ersatzweise als Trainingsparcours, Entwicklungsumgebung für Formationskontrolle und Ballanwendungen bereitsteht.

Im Vordergrund stand vorerst die Umsetzung des geplanten Bewegungsverhaltens auf dem realen Spielfeld. Dazu wurden alle relevanten Systemmodule einer prinzipiellen und leistungssteigernden Überarbeitung unterzogen. Die neuen Routinen des Moduls zur Bildverarbeitung und Objekterkennung sind deutlich robuster gestaltet und in ihrer Abarbeitung beschleunigt. Im Zuge dessen entstand auch das Konzept zur bahngesteuerten Positionierung des Roboters mit Hilfe eines abrufbaren Spurkataloges, der als besondere Neuerung die Anwendung einer Bahnregelung auf der Basis Lissajousscher Figuren enthält.

Die Möglichkeiten dieser Robotersteuerung werden durch die gekoppelte Reglerstruktur eines Motorreglers und eines Dynamikreglers direkt vom Roboter selbst unterstützt. Über die motor- oder schlupfgeregelte Roboterkontrolle ist somit die präzise Einhaltung der kinematischen Vorgaben gewährleistest. Mit wachsender Erfahrung konnte der Roboter LUKAS den Ansprüchen iterativ in Material und Form angepasst werden.

2 Stand der Technik

2.1 Robotik

Mit immer preiswerteren Robotern kann es sich der 'moderne' Mensch leisten, mehr und mehr ungeliebte Aufgaben abzugeben. Neue Haushaltsroboter erobern bereits Domänen der Hand-Rasenmäher und Hand-Staubsauger. Es verwundert kaum, dass Firmen wie iRobot[2] nicht nur auf dem militärischen und wissenschaftlichen, sondern auch auf dem zivilen Markt mit ihren Produkten erfolgreich sind.

Der Stand der Technik kann durchaus an dem Spektrum der Bewegungsarten abgelesen werden, welche mobile Roboter schon beherrschen. Im Folgenden sind einige Projekte aufgezählt, die zum Teil auch in der Öffentlichkeit eine gewisse Popularität erreichten. Diese beispielhafte Auflistung in Tab. 2.1 dient lediglich der Darstellung des Umfeldes, in dem sich die wissenschaftliche Forschung aktuell bewegt.

Robotersysteme sind die modernen Werkzeuge des Menschen. Durch mobile Roboter erweitern sich der Freiheitsgrad und der Aktionsradius menschlichen Denkens und Handelns. (Halb-)autonome Roboter erkunden das Sonnensystem, die strategische Lage und übernehmen Servicedienste. Es gibt nur noch wenige biologische Nischen, in die noch nicht künstliche Akteure vorgedrungen sind. Biologisch inspirierte künstliche Fortbewegungsspezialisten wie der Wasserläufer Strider sind ebenso vertreten wie Generalisten in Form der modularen Robotersysteme M-TRAN und Superbot. Analog der biologischen Artenvielfalt fallen der systematische Überblick und umso mehr die Einführung einer Ordnung nicht leicht. Mit hybriden Systemen, die mehr als eine Bewegungsform beherrschen, verschwinden die Grenzen und neuen speziellen Techniken fehlt die klassische Zuordnung. Dazu zählt z.B. auch der Schleimpilzroboter von Tsuda et al. [11] als organisch-anorganische Hybridform. Die

[2] http://www.irobotstore.com (Stand 06/2008)

aktuellen Entwicklungsprogramme durchziehen selbst alle technischen Größendimensionen von Nano[3], Micro[4], Mini[5] bis Makro[6].

Nach anderen Kriterien können Robotikprojekte auch durch die Art der Sensorik bzw. Aktorik, der Art ihrer Mensch-Maschine-Schnittstelle, der Art ihrer Anwendungsaufgaben, der Art ihrer Architektur, Organisation, Kontrolle oder der Art der Umgebung klassifiziert werden. Sofern es sich allgemein um softwaregesteuerte Technologie handelt, ist eine automatische Unterstützung der Bearbeitung möglich. Je nach Fokus gehören damit auch die zahllosen künstlichen Assistenten im öffentlichen und privaten Softwarebereich in den Bereich der Robotik. Nicht unerwähnt soll die Vielfalt der Roboterwettbewerbe bleiben, die besonders in Japan, Deutschland und den USA für medienwirksame Unterhaltung und Umsätze sorgen und bewusst auch dem Nichtrobotiker einen Einblick auf den aktuellen Forschungsstand geben. Die Tab. 2.2 enthält eine aktuelle Auswahl der bedeutenderen Veranstaltungen.

Der Fortschritt existierender Robotik-Organismen und KI spiegelt sich selbst im bisher stetig wachsenden Ligasystem des RoboCup wieder. Jede Sparte ist der Optimierung einer definierten Fähigkeit gewidmet.

Weil einzelne Ligen des RoboCup, wie z.B. Small-Size und Mid-Size, mitunter synergetisch aufeinander wirken, wird es zukünftig Verschmelzungen geben. Letztendlich wird die Erschaffung eines praxistauglichen humanoiden Roboters angestrebt. Als einer der bekannteren Roboterwettbewerbe will auch der RoboCup in der Gesellschaft neues Interesse und Potential wecken. Für den entsprechenden Einstieg in die anderen neun Ligen des RoboCup verfolgt die Junior-League bewusst dieses Ziel.

[3] http://chemistry.fas.nyu.edu/object/nadriancseeman.html (Stand 06/2008)
[4] http://www.mein.nagoya-u.ac.jp/index.html
[5] http://www.eco-be.com, 'Eco-Be!' Höhe: 25mm, Gewicht: ca. 20g
[6] http://www.ri.cmu.edu/projects/project_362.html, 'Ambler' Höhe: 6m, Gewicht: ca. 2t

Bewegungsart	
Projekt Internetseite bzw. Literaturquelle	**Abbildung** Bild jeweils von der Quelle des Projekts

Weltraumflug / 6-Radantrieb

Galileo
http://www.jpl.nasa.gov
Mars Rover Sprit
http://marsrovers.jpl.nasa.gov/home/

Starrflügelflug

UAV Global Hawk
http://www.northropgrumman.com/
UCAV Barracuda
www.airpower.at/news06/0511_eads-uav/


Drehflügelflug

Autonomous Flying Vehicle (AFV)
http://control.mae.cornell.edu/purwin/PhDWork
USC Autonomous Flying Vehicle
http://www-robotics.usc.edu/~avatar/


4-Radantrieb / Kettenantrieb

Stanley
http://cs.stanford.edu/group/roadrunner/stanley
MF3
http://www.khgmbh.de/fern3.php3

Omni-3-Radantrieb / Omni-4-Radantrieb

Parsian-Robotic
http://www.parsianrobotic.com/
Airtrax
http://www.airtrax.com/


1-Radantrieb / Kugelantrieb

Gyrover
http://www.ri.cmu.edu/projects/project_102.html
Rotundus / www.rotundus.se
Ballbot
http://www.msl.ri.cmu.edu/projects/

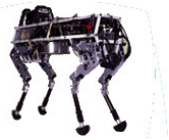

Fortsetzung auf der nächsten Seite

8-beiniges Gehen / 6-beiniges Gehen

Skorpion-Roboter
http://ais.gmd.de/BAR/SCORPION/
LAURON 4c
www.fzi.de/ids/projekte.php?id=293


4-beiniges Gehen / 3-beiniges Gehen

BigDog
www.bostondynamics.com/content/sec.php
STriDER
www.me.vt.edu/romela/RoMeLa/Research.html

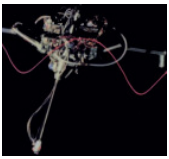



2-beiniges Gehen

P3 & ASIMO
http://www.honda-robots.com
Bruno
http://www.dribblers.de/


1-beiniges Springen / 1-beiniges Hüpfen

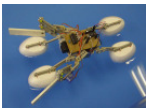
Toyota Monopode 2
http://robotgossip.blogspot.com/2006/09/jumping-robot-leg.html
Pogo Robot
[12]

Klettern / Paddeln / Wasserlaufen

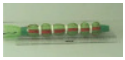
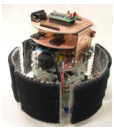
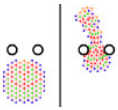
RiSE, RHex
www.bostondynamics.com/content/sec.php
Strider
www.engadget.com/2006/06/16/





Peristaltisches Kriechen / Amöbisches Kriechen

Wormlike Robot
http://www.tu-ilmenau.de/tm/
SlimeBot
www.tu-ilmenau.de/fakmb/Slimebot-A-Modular.4077.0.html

Fortsetzung auf der nächsten Seite

Undulatorisch Kriechen

ACM-R5
www.robot.mes.titech.ac.jp/robot/snake/acm-r5/acm-r5_e.html

Schwimmen / Tauchen

Robotfish
http://cswww.essex.ac.uk/Research/roboticfish/
RoboTunaII
http://web.mit.edu/towtank/www/Tuna/tuna.html

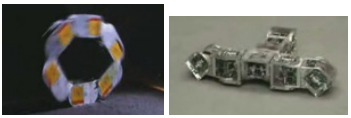
Überschlagen

Platonic Beast
www.cs.ubc.ca/spider/pai/history.htm

Multimodale Lokomotion

SuperBot

http://www.isi.edu/robots



M-TRAN

http://unit.aist.go.jp/is/dsysd/mtran3/

Swarmanoid (S-Bot)

http://www.swarm-bots.org

Tensegrity Robot

[13]

Tab. 2.1 Auswahl mobiler Roboter nach Art der Lokomotorik (Stand 06/2008)

Wettkampf / Internetseite	**Kennzeichen**
RoboGames http://www.robogames.net/	Jedermann-Roboter-Olympiade mit ca. 70 Disziplinen
Aerial Robotics Vehicle Competition http://avdil.gtri.gatech.edu/AUVS/	Wettkampfmissionen für autonome Flugobjekte
Ground Robotic Vehicle Competition http://www.darpa.mil/grandchallenge/	Wettkampfstreckenfahrt für autonome Fahrzeuge
Underwater Robotic Vehicle Comp. http://www.auvsi.org/competitions/water.cfm	Unterwasseroperationen für autonome Objekte
International METU Robotics Days http://www.roboticsdays.org/	Kreativer Roboter-Workshop mit Wettkämpfen
IEEE Micromouse competition http://www.micromouseonline.com/	Wettstreit der schnellsten Labyrinthmäuse
Annual fire-fighting home robot contest http://www.trincoll.edu/events/robot/	Internationaler Vergleich bester Feuerlöschroboter
Duke Annual Robo-Climb Competition http://robotics.pratt.duke.edu/roboclimb/	Die funktionale Herausforderung für Kletterroboter
RoboCup http://www.robocup.org/	Teamwettkampf mit Schwerpunkt Fußball und Rettung
Federation of Int. Robot-Soccer Association http://www.fira.net/	Teamwettkampf mit Schwerpunkt Fußball

Tab. 2.2 Bedeutende Wettbewerbe autonomer mobiler Roboter (03/2008)

2.2 Modulare Robotersysteme

Im Zusammenhang mit kooperierenden Systemen und im Hinblick auf zukünftige Bestrebungen sollen noch die amorphen Roboter (engl.: amorphous and soft robotics) etwas genauer betrachtet werden. Deren Forschungsrichtung ist noch sehr jung. In den Veröffentlichungen ist auch von formverändernden (engl.: shape changing) [14], selbst aufbauenden (engl.: self-assembling), selbst replizierenden (engl.: self-replicating) [15], selbst reproduzierenden (engl.: self-reproducing) [16] oder häufig von rekonfigurierenden

(engl.: reconfigurable) Robotern die Rede [17-19]. Ob offene Strukturen aus der Schwarmforschung oder mechatronische Einheiten mit einer abgeschlossenen Zahl von verknüpften Modulen (engl.: cluster), gemeinsam ist ihnen ihre Transformations- und Anpassungsfähigkeit. Die Fähigkeit einer modularisierten Einheit, sich flexibel verschiedener Fortbewegungsarten zu bedienen, erweitert die Einsatzmöglichkeiten mobiler Roboter auf vielfältige Weise, weil eine komplexe, strukturierte Umwelt nicht zwangsläufig als Menge zu meidender Hindernisse zu betrachten ist. Mit der Anzahl der Module steigt die Anzahl der Konstellationen möglicher Strukturen, weshalb hochmodulare Systeme den Vorteil haben, sich besser an die Umgebung anzupassen. Ebenso mannigfaltig wie die Bewegungsformen gestalten sich die Module selbst. Tab. 2.1 zeigt eine Auswahl vom homogenen, unverknüpften Slimebot der Ebene [17] über das heterogene System Swarmanoid bis zu den räumlichen, gekoppelten Strukturen des Superbot, M-TRAN und Tensegrity Robot.

Vielfältig sind auch die speziellen Methoden der internen Kontrolle. Je nach Aufgabenstellung und Kooperation der Module werden indes Methoden mit zentral gesteuerter, geplanter morphologischer Veränderung [20], lernende Schwarmagenten [21] oder dezentrale Konfigurationssysteme bevorzugt [22]. Sensorgesteuerte reaktive Systeme mit limitierten Elementfähigkeiten bauen auf der selbstorganisierenden Wirkung und der emergenten Eigenschaft des Gesamtsystems auf [15, 17]. Aus den Eigenschaften der Einzelelemente, ihrer Wechselwirkung miteinander und mit der Umwelt erwächst ein komplexes Gesamtverhalten (engl.: swarm intelligence). Der Vorteil gegenüber Robotern formfester, untrennbarer Struktur liegt in der höheren Fehlertoleranz im Falle technischer Probleme [23]. Defekte Module werden aus der Struktur ausgeschlossen, während das wiederhergestellte Gesamtsystem arbeitfähig bleibt [14]. Besonders leichte modulare Roboterstrukturen sind durch Tensegrity-Module erreichbar [13, 24]. Den Begriff tensegrity, als Kunstwort aus tensile und integrity, prägte Buckminster Fuller [25, 26]. Tensegrity-Module eignen sich dazu, einen beweglichen Organismus zu kreieren, der seine Flexibilität ohne mechanische Drehgelenke erreicht [27].

2.3 Klassische und verhaltensbasierte Künstliche Intelligenz

Die Verbindung von KI-Forschung und Robotik ist eine Symbiose zum beiderseitigen Vorteil. Die klassische KI in der Informatik, die sich typischerweise anhand des Schachproblems verdeutlichen lässt, ist in ihrer funktionsorientierten, deliberativen Architektur, durch ihre festen Grundsätze und symbolischen, planenden Strukturen nicht in der Lage gewesen, die neuen Aufgaben und technischen Herausforderungen in unbestimmten Umwelten mit intelligenten Maschinen zufriedenstellend zu lösen. GOFAI (Good Old Fashioned Artificial Intelligence [28]) geht davon aus, dass alle Auswirkungen einer Aktion in dieser Welt schon bekannt sein sollen, bevor die Aktion überhaupt stattfindet [29, 30].
Als Vertreter der Nouvelle-AI, d.h. der neuen verhaltensbasierten KI, versuchten Arkin [31], Brooks [32] und Mackworth [33] mit ihren reaktiven, verhaltensbasierten Ansätzen den Paradigmenwechsel in der KI-Forschung durchzusetzen, um die notwendige enge Kopplung von Wahrnehmung (engl.: perception), Denken (engl.: reasoning) und Handeln (engl.: action) zu erreichen. Roboter werden intelligenter agieren, wenn sie auf höherer Instanz mit begrifflicher und logischer Intelligenz nach Art der GOFAI zu handeln lernen, nachdem sie reflex- und verhaltensbasierten Architekturen schon vollständig beherrschen. Softwarearchitekturen zur Realisierung autonomen Verhaltens lassen sich daher in die grundsätzlichen Ansätze einteilen [34, 35]:

- **Funktionsorientierte Architektur (deliberativ)**
  (Auf Basis eines erstellten Weltmodells erfolgt nach der Planung die gesteuerte Aktion)
- **Verhaltensbasierte Architektur (reaktiv)**
  (Sensorische Signale bewirken nach festen Regeln definierte Verhaltensreaktionen)

- **Hybride Architektur (reaktiv/deliberativ)**
  (Hierarchisches Modell mit reaktiver Unter-, Vermittlungs- und planender Oberschicht)
- **Kognitive Architektur**
  (Lernfähigkeit und Selbstmodellierung des Roboters zur Erweiterung des eigenen Verhaltens)

Diese Methoden werden in der Praxis in Abwandlungen und Kombinationen verwendet. Da sich Robotik in der komplexen physischen Welt bewegt und diese nie vollständig vorhersehbar sein wird, stellt sie den idealen Partner für die Weiterentwicklung der KI-Forschung dar. Roboterwettbewerbe aller Art wetteifern auch aus diesem Grund in der Entwicklung hybrider und kognitiver Architekturen für die künstliche Annäherung an den Entwicklungsstand mobiler Lebewesen.

2.4 RoboCup Small-Size-League 2008

Wie in Abschnitt 2.1 schon erwähnt, ist das Small-Size-Konzept in eine RoboCup-Ordnung eingebettet, die sich hauptsächlich durch konstruktive Vorgaben und spieltechnische Richtlinien voneinander unterscheiden. Wie der Prinzipskizze in Abb. 2.1 zu entnehmen ist, setzt sich das gesamte System aus Modulen der Bildverarbeitung, KI-Spielplanung, Funkübertragung und den Feldspielern zusammen. Alljährlich messen, erproben und vergleichen internationale Forschergruppen ihre neuesten Systeme und Algorithmen. Für ein erfolgreiches Team ist es Voraussetzung, ein Gesamtsystem zu besitzen, dessen optimierte und hocheffiziente Soft- und Hardwarekomponenten sich zu einer robusten funktionalen Einheit ergänzen. Es ist ein Zeichen des Fortschrittes, dass sich das Regelwerk der Small-Size seit 1997 immer mehr den Regeln der Fédération Internationale de Football Association (FIFA) angenähert hat. So gestalten gelbe und rote Karte, Einwurf, Eckstoß, direkter

und indirekter Freistoß, Strafstoß und Elfmeter auch ohne menschlichen Eingriff einen abwechslungsreichen Spielablauf.
Aus der offiziellen Regelbeschreibung[7] für die Saison 2008 geht hervor:

- Ballspiel um Trefferpunkte mit zwei Teams mit zu je fünf Robotern
- Spiel mit orangenem Golfball auf grüner Teppichfläche (ca. 35m²)
- Roboter im maximalen Zylinderbauraum: max. Ø180mm x 150mm Höhe
- farbiges Teamtrikot auf der Oberseite für die bildgestützten Identifizierung
- (externe) autonome Verhaltenssteuerung der Feldspieler
- Spielzeit 2 x 15min.

Das Grundprinzip zur Kontrolle von mehreren konkurrierenden und kooperierenden Spielagenten in einer dynamischen Welt gilt auch in den anderen ballspielenden Ligen der Mid-Size, Humanoid bzw. Legged League.
Obwohl die Hauptmerkmale der Small-Size schon genannt wurden, ist ein vertiefender Blick auf individuelle Annäherungen und die Betrachtung grundlegender Komponenten der einzelnen Systembereiche für das Verständnis und die Möglichkeiten sinnvoll. Die Klassen der selbstgebauten Roboter zählen nicht ohne Grund zu den schwierigsten. Alle Komponenten und Parameter werden im Rahmen des Reglements selbst bestimmt. Aufgrund des wettbewerbsbasierten Konkurrenzdruckes haben sich die Small-Size-Teams seit jeher die Möglichkeiten der fortgeschrittenen Robotertechnologie zunutze gemacht. Das Open-Source-Prinzip des RoboCup erleichtert hierfür den Aufbau eines Spielsystems und vermeidet, dass die Leistungsunterschiede der Lösungsansätze zwischen Siegern und Verlierern allzu gravierend ausfallen[8].

[7] http://www.robocup.org/regulations/4.html
[8] Quellen z.B. unter http://sourceforge.net

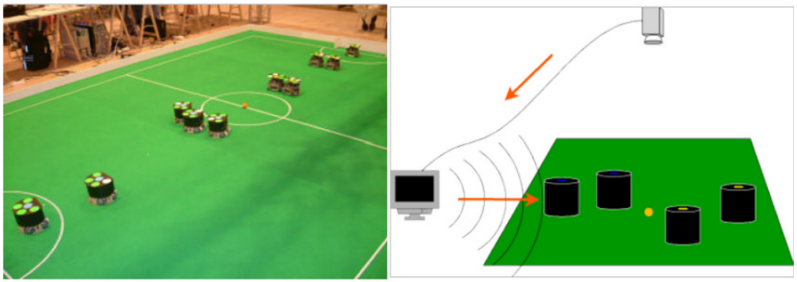
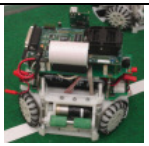
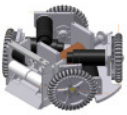


Abb. 2.1 Feldansicht und prinzipielles Schema (Bildquelle: www.RoboCup.org)

2.5 Roboterkonstruktionen der Small-Size-League

2.5.1 Allgemeine Merkmale

Die Mehrheit der Arbeitsgruppen verwendet aktuell vierrädrige omnidirektionale Roboter mit Pulsweitenmodulation (PWM) angesteuerten Gleichstrommotoren, deren kinematische Bahnsteuerung auf Basis der Geschwindigkeitskontrolle erfolgt (siehe Tab. 2.3). Die Teams profitieren untereinander von neuen Methoden bzw. Konstruktionslösungen und verknüpfen sie mit eigenen Entwicklungen, so dass die erwünschte Vielfalt erhalten bleibt. Von den RoboCup-Mannschaften, die bisher an Small-Size-Meisterschaften teilnahmen, zählen die Arbeitsgruppen der FU-Fighters und CMDragons mit zu den erfolgreichsten. Sie übernehmen eine Vorbildfunktion für Ein- und Aufsteiger und gehören zu den Mannschaften, die ihren eigenen Grundprinzipien und Problemlösung weitgehend treu geblieben sind. Bezüglich Roboter, Software und Ausrüstung sollen sie nachfolgend genauer vorgestellt werden.

Team (Nationalität)	Internetseite
Bild des Roboters	• Antrieb / Ansteuerung des Roboters • Chassis / Elektronik-Layout / Robotermasse • Feldsensor / Methode der Bildverarbeitung • Methode der Verhaltenssteuerung
FU-Fighters (Deutschland)	http://www.fu-fighters.de
	• 4x brush / Geschwindigkeitsvorgaben • Aluminium / Monoboard / ~2,0kg • 2x Kamera / Adaptive Farbkarten • Reaktive verhaltensbasierte Kontrolle mit Potentialfeld- und A*-Pfadplanung
CMDragons (USA)	http://www.cs.cmu.edu/~robosoccer/small
	• 4x brushless / Geschwindigkeitsvorgaben • Aluminium / Monoboard / ~2,0kg • 2x Kamera / Schwellwert-basierte Look-Up Tabelle • Hybride (STP) Kontrolle mit ERRT-Pfadplanung
Parsian (Iran)	http://www.parsianrobotic.ir
	• 3x brush / Geschwindigkeitsvorgaben • Aluminium + Kunststoff / Monoboard / ~1,5kg • 2x Kamera / Schwellwert-basierte Look-Up Tabelle • Hybride Kontrolle
Eagle Knights (Mexiko)	http://robotica.itam.mx
	• 4x brush / Geschwindigkeitsvorgaben • Kunststoff / Multiboard / ~2,0kg • 2x Kamera / Schwellwert-basierte Look-Up Tabelle • Hybride Kontrolle mit geometrischer Pfadplanung durch Baumstrukturen
B-Smart (Deutschland)	http://www.b-smart.de
	• 4x brush / Geschwindigkeitsvorgaben • Aluminium / Multiboard / ~1,6kg • 2x Kamera / Schwellwert-basierte Look-Up Tabelle • Hybride Kontrolle mit RRT-Pfadplanung
Skuba (Thailand)	http://iml.cpe.ku.ac.th/skuba
	• 4x brushless / Geschwindigkeitsvorgaben • Aluminium / Multiboard / ~2,8kg • 2x Kamera / Schwellwert-basierte Look-Up Tabelle • Hybride Kontrolle mit Potentialfeld-Pfadplanung

Tab. 2.3 Systemmerkmale ausgewählter RoboCup-Small-Size-Teams 2007

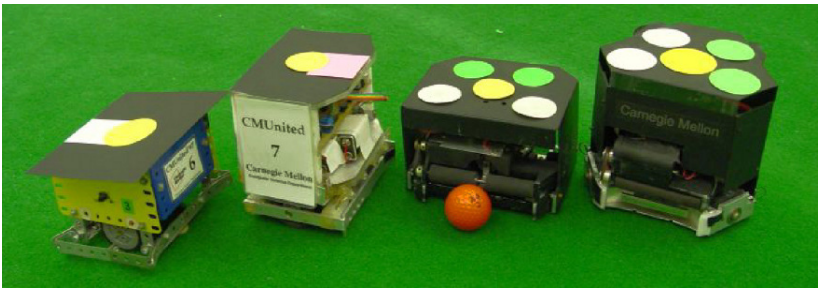


Abb. 2.2 Carnegie Mellon Roboter der Jahre 1997, 1998, 2001 und 2002 [36]

2.5.2 CMDragons Roboter

Seit ihrem Start im Jahr 1997 haben die Roboter des CMDragons Teams der Carnegie Mellon University ihre Fähigkeiten erheblich verbessert (Abb. 2.2). Das Modell 2007 (Abb. 2.3) wird von Bruce et al. [37] als ein holonomer Roboter beschrieben, der bei Geschwindigkeiten bis 4m/s eine maximale Beschleunigung von 3 bis 6m/s² erreicht.. Die vier omnidirektionalen Räder werden von bürstenlosen 30W-Gleichstrommotoren angetrieben, deren Quadratur-Encoder für einen akkuraten Radlauf und die Geschwindigkeitsüberwachung sorgen. Radschlupf wird verhindert, indem eine fest definierte Antriebskraft (bzw. Beschleunigung) nicht überschritten werden darf.

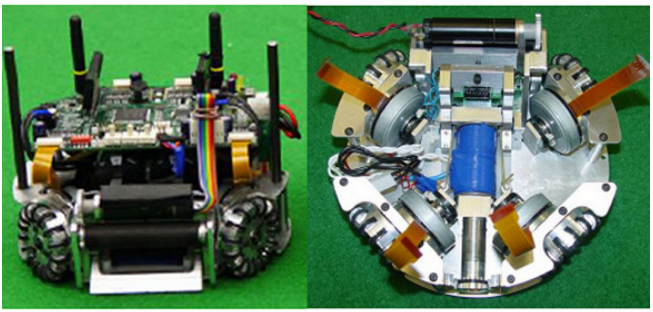


Abb. 2.3 Innenansichten des CMDragons Small-Size Roboters 2006 [36, 37]

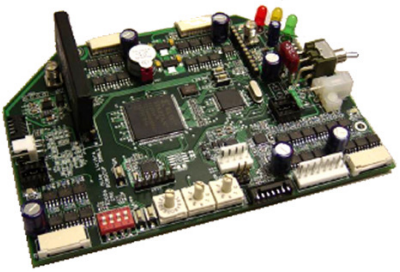


Abb. 2.4 Mainboard des CMDragons Small-Size-Roboters 2006 [36]

Der Schussmechanismus erreicht Ballgeschwindigkeiten bis zu 15m/s durch eine, mit 200V-Hochspannungsimpulsen aktivierte, eigens gefertigte Magnetspule. Zusätzliches Passspiel ist durch einen Chip-Kick-Hochschuss (bis 4,5m Flugreichweite) möglich. Die Ballkontrolle wird von einer aktiven, gummibeschichteten Dribblerrolle unterstützt, die sich auf einem schwenkbaren Dämpfer zur Verbesserung des Ballaufschlages befindet.
Für eine verbesserte Eigenkontrolle greift der Roboter auf ein integriertes Winkelgeschwindigkeits-Gyroskop zurück. Das Gyroskop ist über einen zweiten Controller in die Fahrsteuerung eingebunden und wird zum genaueren Positionieren genutzt [36]. So kann der Roboter, der sein internes Koordinatensystem permanent mit externen Daten des Bildverarbeitungssystems kalibriert, auch einige Sekunden autonom navigieren. Die gesamte Positionierung ist dadurch sehr robust. Die Roboterelektronik in Abb. 2.4 dominiert eine zentrale ARM7-CPU (Taktfrequenz 58MHz), die mit einem Xilinx Spartan2 FPGA (Field Programmable Gate Array) zusammenarbeitet. Der ARM7-Kern übernimmt die RS232-Funkkommunikation, die Berechnungen der Geschwindigkeits-PD-Regelung und überwacht weitere Onboard-Systeme. Das FPGA kontrolliert die Quadratur-Decoder, die PWM-Generierung der bürstenlosen Motoren, die Drehfelder und die serielle Kommunikation mit anderen Onboard-Komponenten. Da auch rechenintensive Operationen auf den FPGA verlagert sind, ist die ARM7-CPU frei für relativ komplexe Steueraufgaben. Neben dem kompakten, robusten Mainboard sind im Roboter noch separat ein Step-Up-Kickerboard und ein IR-Ball-Sensor untergebracht.

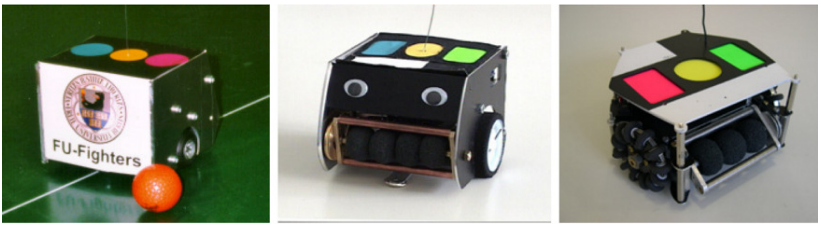


Abb. 2.5 FU-Fighters Roboter der Jahre 1998, 1999 und 2003 [39]

2.5.3 FU-Fighters Roboter

Die FU-Fighters setzten sich mehrfach mit ihren schnellen, leichten Robotern in der Small-Size-League durch (Abb. 2.5). Die Erfahrung mit den Vorgängern schlug sich wiederholt in verbesserten Neuentwicklungen nieder. Das Modell, mit dem die FU-Fighter 2005 die Weltmeisterschaft in Osaka gewannen, soll hier kurz vorgestellt werden (Abb. 2.6) [38, 39]. Der Roboter basiert auf einem selbstgebauten Aluminiumrahmen in der Konstruktion eines vierrädrigen omnidirektionalen Antriebes und erreicht Fahrgeschwindigkeiten bis 3m/s und Schussgeschwindigkeiten bis 10m/s. Der Roboter im Computer Aided Design (CAD) ist in Abb. 2.6 zu sehen. Daneben liegt dieser Small-Size-Roboter zerlegt. Darin befindet sich mittig das elektronische Mainboard mit Funkmodul und allen angeschlossenen Sensoren und Aktoren. Oben liegen die Lithium-Akkumulatoren, die Antriebsmotoren mit den Ritzeln des Stirnradgetriebes und rechts daneben die zwei Elektromagnete für Hoch- und Flachschuss. Rechts von der Elektronikplatine sind die Kondensatoren, Schalter und die Ladeelektronik für die Schussspulen zu sehen. Unten rechts im Bild ist das Fahrgestell und links daneben die Dribblerwalze mit dem dazugehörigen Motor. Das kleine rote Objekt mittig ist das Gyroskop. Links neben dem Mainboard liegt der Vorbau des Roboters mit eingebauter Lichtschranke. Die Elektronik basiert auf einem 8MHz-Mikrocontroller der HCS12-Baureihe von Freescale. Wie oben beschrieben sind an das Mainboard die Sensoren (acht Tickzähler für die Motordrehrichtungen, eine Lichtschranke, ein Gyroskop)

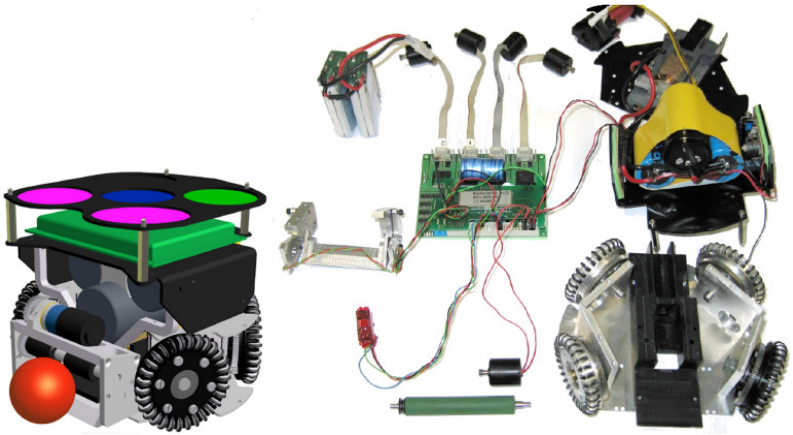


Abb. 2.6 FU-Fighters Roboter 2005 als CAD-Modell und in Komp. zerlegt [38, 40]

und Aktoren (vier H-Brücken für die PWM-Ansteuerung der Gleichstrommotoren, zwei Schussmagneten, ein Dribbler) angeschlossen [38]. Die serielle Funkschnittstelle eines Roboters arbeitet mit 868 und 914MHz, oder alternativ im kabellosen Netzwerk (WLAN) oder via Bluetooth. Die Funkbefehle umfassen Vorgaben zu Fahrgeschwindigkeiten in X,Y-Richtung, Drehgeschwindigkeit, Aktivierung der Dribblerwalze oder der zwei Schussmagneten mit Angabe einer Schussstärke durch die Begrenzung der Auslösezeit. Das Steuerprogramm des Mikrocontrollers hat eine primär interruptgestützte Struktur. Bevor die eigentliche Verarbeitung auf höherer Ebene innerhalb der Programmhauptschleife umgesetzt wird, erfolgt die Registrierung der Interrupts. Auslöser der Interrupts sind der serielle Empfang (Befehle) oder Versand (Statusrückgabe), die Lichtschranke und der Echtzeittakt. Alle 4ms berechnen die drei PID-Regler der Geschwindigkeitsteuerung die Korrekturen zur Vor-, Seit- und Drehbewegung. Die Reglerparameter sind automatisch mit Bestärkenden Lernen (engl.: reinforcement learning) aus gefahrenen Testbahnen optimiert. Mit dem direkten Messen der Drehgeschwindigkeit durch das Gyroskop besteht die Möglichkeit einer besseren Detektierung des Schlupfes. Zur Schlupfvermeidung wird auch hier die Beschleunigung der Räder begrenzt.

2.6 Bildverarbeitung und Objekterkennung

2.6.1 Allgemeine Verfahren

Autonome Robotersysteme des RoboCup besitzen eine sensorgestützte Pfadplanung, die das Modul zur aktiven Robotersteuerung über die Umgebung und den erforderlichen Bewegungsablauf instruiert. Nach dem Reglement der Small-Size-League ist es erlaubt, die sensorische Erfassung der Feldlage aus globaler Perspektive zu betreiben und die dafür nötigen höheren Ebenen der Softwaresteuerung im externen Computersystem berechnen zu lassen.
Die Bildverarbeitung umfasst im Allgemeinen auch die Objekterkennung und hat die Aufgabe, die optisch erfassten Kamerabilder in strukturierte numerische Daten zu transformieren. Die über farbige Trikotflächen kodierten Roboter müssen jeweils in ihren Positions- und Bewegungsparametern erkannt werden, um sie exakt mit der Verhaltenskontrolle anzusteuern. Bevorzugt werden momentan hochauflösende Fire-Wire-Kameras mit Bildwiederholraten bis zu 60Hz[9]. Die Detektierungsgeschwindigkeit der aktuellen Situation bestimmt somit die Reaktionsgeschwindigkeit der Steuerung, weshalb die Bearbeitung der Softwarealgorithmen für einen Wettkampfvorteil mit minimaler Verzögerung und möglichst synchron zur Bildfrequenz zu erfolgen hat. Ein Zyklus der Bildverarbeitung und Objekterkennung kann nach Tab. 2.4 verallgemeinert dargestellt werden. Arbeitet man mit gespeichertem Vorwissen, muss nicht in jedem neuen Kamerabild der gesamte Rahmen durchsucht werden. Ein Tracker-Modul verfolgt erkannte Objekte und begrenzt damit den Berechnungsaufwand auf einen kleinen Bildausschnitt.
Anschließend kann ein Softwaremodul zur abschätzenden Vorhersage der gewonnenen Objektparameter zur Kompensation der Latenzzeit in der Regelungsschleife genutzt werden. Eine Gesamtlösung der Bildverarbeitung kann

[9] Aktuelle Übersicht von Fire-Wire-Kameras unter
http://damien.douxchamps.net/ieee1394/cameras/

Bearbeitungsschritt
(0. Maschinelles Lernen) – (verfahrensabhängige Einordnung dieses Schrittes) Von zu untersuchenden Farbklassen werden Lerndatensätze aufgenommen. Mit Hilfe dieser Lerndatensätze wird ein Klassifikator trainiert.
1. Farbklassifikation Aus dem Eingangsbild mit Pixel-Farbcodierung wird unter Zuhilfenahme der trainierten Referenz ein klassifiziertes Bild erzeugt.
2. Bildsegmentierung Im klassifizierten Bildausschnitt wird nach zusammenhängenden Regionen gleicher Farbklasse gesucht. Geometrische Strukturen können dadurch erkannt und abstrahiert werden.
3. Trikoterkennung Strukturen des segmentierten Bildes können mit vordefinierten Trikotmasken verglichen und zu gefundenen Objekten zusammengesetzt werden.
4. Objektinitialisierung Aktuelle Objektparameter zu Identität, Teamzugehörigkeit, Spielfeldposition, Orientierung und Geschwindigkeit werden ermittelt.

Tab. 2.4 Einzelschritte der Bildverarbeitung und Objekterkennung

aus diversen Methoden für jeden der in Tab. 2.4 genannten Einzelschritte zusammengesetzt sein. So beeinflusst der jeweilige Verwendungszweck schon die Wahl des verwendeten Farbraumes. Die kubischen RGB- und YUV- oder der zylindrische HSI-Farbraum sind in Anwendungen der Computer- und Videotechnik sehr verbreitet [41].

Der maschinelle Lernvorgang kann entweder manuell (engl.: supervised learning) oder automatisch (engl.: unsupervised learning) erfolgen. Angewandte Verfahren zur automatischen Gewinnung klassifizierter Referenzdaten sind z.B. die Clusteranalyse eines Testbildes [42] oder die Verwendung adaptiver Farbkarten [40].

Für die automatische Klassifikation farbiger Flächen sind einfache Verfahren, wie die Schwellwertklassifikation oder komplexere Verfahren wie die Maximum-Likelihood-Klassi-fikation, sowie die Klassifikation mit Support-Vector-Machines in Gebrauch. Die Schwellwertklassifikation ermöglicht die eindeutige Zuordnung von Farbraumtripeln mit Hilfe einer statischen Look-Up-Tabelle

[43]. Mittels der statistischen Maximum-Likelihood-Klassifikation ist es möglich, einen Farbraumtripel dem Vertreter der ähnlichsten Farbklasse zuzuordnen [44]. Die Methode der Support-Vector-Machines begrenzt die Klassen derart, dass zwischen den Vertretern der Klassen ein möglichst breiter Trennbereich verbleibt [45].

Auch bei den Verfahren zur Bildsegmentierung gibt es vielfältige Ansätze. Grundsätzlich werden pixel-, kanten- und regionenorientierte Verfahren unterschieden. Außerdem unterscheidet man modellbasierte Verfahren, bei denen man von einer bestimmten Form der Objekte ausgeht sowie texturbasierte Verfahren, bei denen auch eine innere homogene Struktur der Objekte berücksichtigt wird [43]. Die Grenzen zwischen den Segmentierungsmethoden sind nicht eindeutig, weil mitunter verschiedene Verfahren kombiniert werden, um bessere Ergebnisse zu erzielen. Im Zusammenhang mit der Beschreibung der eigenen Bildverarbeitung ist an dieser Stelle das pixelorientierte Zeilenkoinzidenzverfahren zu nennen [46].

Der Aufwand zur Ermittlung der Objektparameter ist nicht zuletzt von der Struktur der verwendeten Robotertrikots abhängig. Freie Bibliotheken wie z.B. ICE[10], VIGRA[11], CXImage[12] oder CImg[13] können die Bildanalyse mit vielen gesammelten Funktionen zur Bildbearbeitung unterstützen. Allerdings sind auch innerhalb des RoboCup auf den Anwendungsfall spezialisierte Module frei verfügbar. Repräsentativ werden nachfolgend wiederum die teamspezifischen Varianten der FU-Fighters und der CMDragons erläutert.

[10] http://www.inf-cv.uni-jena.de/ice/ice.html (Stand 06/2008)
[11] http://kogs-www.informatik.uni-hamburg.de/~koethe/vigra/
[12] http://www.xdp.it/cximage.htm
[13] http://cimg.sourceforge.net/

2.6.2 CMDragons Vision

Das Modul CMVision2 (siehe Abb. 2.7) ist ausschließlich für die Low-Level Farbklassifizierung, Bildsegmentierung und Regionsanalyse zuständig [47].
Wegen des Helligkeitsproblems von Licht und Schatten auf RGB-Farbwerten und der Bedingung eines dreidimensionalen Farbmodells wird das YUV-Format verwendet, das aus einem Helligkeits-Wert Y (Luminanz) und zwei Farbanteilswerten U und V (Chrominanz) aufgebaut ist. Die Information zur Mitgliedschaft einer Farbklasse wird in dieser Darstellung des Farbmodells diskret in den drei Dimensionen gespeichert. Mitgliedsvolumen klassifizierter Farbtripel können diverse geometrische Formen (Kegel, Ellipsoid) haben. Es wird die schnellste Form der Schwellwert-Farbklassifizierung mittels UND-Verknüpfung angewendet:

$$Pixel\_in\_class = YClass[Y] \cap UClass[U] \cap VClass[V];$$

Der resultierende binäre Wert gibt an, ob der Pixel zur Farbklasse gehört oder nicht. Diese Vorgehensweise ist deshalb vorteilhaft, weil ein Pixel nur einer Klasse angehören kann. Als Referenz steht eine Look-Up-Tabelle bereit, die es mit der Verwendung von 32-Bit-Werten ermöglicht, bitcodiert 32 Farbklassen im dreidimensionalen Farbmodell zu verschlüsseln.
Im klassifizierten Bild werden mittels Run-Length-Encoding (RLE) analog dem Zeilenkoinzidenzverfahren benachbarte horizontale Pixellinien vertikal zueinander in Bezug gebracht und dadurch klassifizierte Pixelgruppen zueinander verknüpft [47]. In einem weiteren Bilddurchlauf werden zusammenhängende Regionen inkremental aus dem verlinkten RLE-Bild aufaddiert, wodurch ihre Farbsortierung und Nummerierung in einer Tabelle ermöglicht wird. Getrennte Regionen einer identischen Farbklasse werden mit einem Regions-Merging-Verfahren vereinigt.
Mit dem NTSC Videosignal (640x240 Pixel, 60Hz) wird auf einem 2.4GHz-Computer, bei einer Berechnungszeit von ca. 1ms ein Durchsatz von ca. 1033 Bilder/s erreicht. Zur Codierung der Position, des Winkels und der ID des Roboters hat sich in der Small-Size-League das Prinzip der Farbsegmentierung

von Markern durchgesetzt. Robotertrikots dieses Prinzips sind fehlerresistenter und in kürzerer Zeit zu prüfen als geometrische Muster oder Barcodes.
Aufbauend auf dem CMVision2-Modul wertet das High-Level-Vision-System die Farb-Marker des CMDragons Butterfly-Tikots zur Berechnung der Position, Orientierung und Identifizierungsnummer (ID) des Roboters aus [48]. Die Zentrumspunkte der Marker werden iterativ bestimmt und ergeben durch Addition der gekreuzten Hilfsvektoren die Orientierung des Roboters. Asymmetrische Muster sind von Vorteil, da somit die Orientierung eindeutig ist und die Marker-Farbe für die Codierung der Roboter-ID zur Verfügung steht. Das High-Level-Vision-System enthält ebenfalls eine Kamera-Geometrie-Kalibrierung und eine Farbschwellwert-Kalibrierung.

2.6.3 FU-Fighters Vision

Das Modul FU-Vision ermöglicht die kontinuierliche Verarbeitung von Bildströmen zweier globaler Fire-Wire-Kameras an einem Mastercomputer mit separaten Controllern [40].
Dadurch, dass die Bilder im RAW-Format mit geringerer Datenrate übertragen werden, d.h. entsprechend des Bayer-Mosaiks der CCD-Kamera pro Pixel nur einen Farbanteil zählt, erfolgt im Rechner zuerst eine Umwandlung in RGB-Werte (Demosaicing). Es werden jeweils vier benachbarte Pixel zu einem zusammengefasst, wobei Rot und Blau direkt übernommen und die beiden Grünen diagonal interpoliert werden.
Die Bildauswertung wird beschleunigt, wenn nur Teile des Bildes für die Objektverfolgung betrachtet werden. Eine lokale Suche wird nur dort durchgeführt, wo die nächste Position eines Feldobjektes anhand Vorwissens seiner letzten Position und Geschwindigkeit vorhergesagt ist. Schlägt diese fehl und der Suchrahmen ist schon bis zur maximalen Größe expandiert, wird die globale Suche (der fusionierten Kamerabilder) aufgerufen. Für eine stabile Bildrate ist aber der Aufruf der rechenintensiven globalen Suche durch definierte Bedingungen begrenzt.

Anstatt zur Farbklasssifizierung eine fixierte Look-Up-Tabelle zu nutzen, die für jede Farbe die ortsunabhängige, globale Toleranzreferenz speichert, wird in der FU-Vision der Gebrauch von Farbkarten forciert, um sich möglichst auch in Echtzeit den lokalen Beleuchtungsänderungen anzupassen [40]. Farbkarten ermöglichen die Verwendung vieler genauerer ortsabhängiger Referenzen. In einem Abbild des Spielfeldes sind sowohl die Farbe in RGB als auch die Größe in Pixeln der Markerreferenz positionstreu gespeichert.
Gerade in einem mit harten Schatten sehr ungleichmäßig beleuchteten Feld und bei langsamen zeitlichen Veränderungen der Beleuchtung sind die besonderen Eigenschaften von adaptiven Farbkarten von Vorteil. Ist der Euklidische Abstand zwischen der Farbe des Feldobjekts und der Referenz im RGB-Farbraum hinreichend klein (Schwellwert), erfolgt die Klassifizierung des Markers entsprechend der gespeicherten Zuordnung [40].
Die farbspezifischen Segmentierer arbeiten in der Regel im HSI-Farbraum. Obwohl berechnungsintensiver als im RGB-Farbraum, variieren in diesem Fall die Distanzen des Farb- und Sättigungs-Kanals nicht so stark. Die Ergebnisse sind mit den verwendeten gesättigten Trikotfarben noch sicherer. Für alle segmentierten Pixel werden der gemeinsame Schwerpunkt, der durchschnittliche Farbwert, die Anzahl der gefundenen Pixel (d.h. die Größe des Markers) und die Kompaktheit gemessen. Diese Werte bilden die Qualität des gefundenen Markers. Bei erfolgreicher Markersuche wird auch die Farbkartenreferenz an den aktuellen Zustand dieser Position angepasst (lokale Adaptation). Aus den qualitativ sortierten Markerlisten der Farbkarten werden die Kombinationen bestimmt, die der geometrischen Anordnung des Robotermodells entsprechen. Die Mannschaft nutzte seit 2004 ein Robotertrikot, das aus drei radialen Markern mit zwei möglichen Farben besteht und acht Identitäten erlaubt (siehe Abb. 2.6).
Zusätzlich wurde eine Vielzahl verschiedener Gegnertrikots implementiert, so dass sich unterschiedlichste Roboter anderer Teams mit höherer Präzision als nur anhand des Teammarkers verfolgen lassen. Für den Anlernprozess genügt es, jede Farbe an einer Stelle einmal per Mausklick zu initialisieren. Die restliche Karte wird automatisch erlernt, während sich der Roboter über das Feld bewegt. Wegen der Torgefährlichkeit von Hochschüssen ist im FU-Vision-

System eine Erkennung von Hochschüssen integriert [49]. Dabei wird der genaue räumliche Parabelflug des Balls mit Auftreffpunkt und Auftreffzeit berechnet. Das System arbeitet auch mit Aufhängepositionen, bei denen die Kameras nicht perfekt zentral und senkrecht über dem Feld hängen, weil die exakten Koordinaten aus der Bildinformation zurückgerechnet werden.

2.7 Methoden der reaktiven Verhaltenssteuerung und Pfadplanung

2.7.1 Allgemeine Verfahren

Wie aus Tab. 2.3 ersichtlich und im Abschnitt 2.3 angedeutet, haben sich die hierarchisch reaktiven Ansätze der Verhaltensteuerung nach Art der Nouvelle AI für dynamische Umgebungen als vorteilhaft erwiesen. Die Kombination mit statistischen und planenden Methoden ergänzen und optimieren hierbei die reaktiven Basismodelle. Beispielsweise spielen die Gewichtung von konkurrierenden Verhaltensmodellen der Strategie oder die Pfadplanung im Detail der Bahnsteuerung eine wichtige Rolle. Da auch hier die Variationsbreite verwendeter Verfahren sehr groß ist, sind die Übergänge der verschiedenen Systeme mitunter fließend. Eine allgemeine Systematik im Sinne der reaktiven und hybriden Systeme nach Arkin [31] ist insofern erschwert und auch entwicklungsbedingt nicht erforderlich.
Methoden der Bahnsteuerung dienen der Kollisionsvermeidung und der Wegoptimierung. Im Allgemeinen nutzt man reaktive Ansätze, die eine nahezu echtzeittreue Anwendung erlauben. Sehr verbreitet ist die Potentialfeldmethode, bei der die Hindernisse mit künstlichen Potentialfeldern belegt werden. Dadurch wirken auf das zu steuernde Objekt abstoßende Kräfte, während der Zielpunkt eine Anziehungskraft ausübt. Der lokale Gradient im Potentialfeld weist den Weg durch das Potentialgebirge [50]. Eine Verbesserung stellen harmonische Dipol-Potentiale dar [51]. Als mögliche Alternative sind mitunter auch Fuzzy-Systeme im Einsatz. In Anwendung definierter Fallrichtlinien

können diese auf Signale mit einer dem menschlichen Verhalten entsprechenden Aktion reagieren [52].
Während einfache geometrische Ausweichbahnen keine optimale Route liefern, sind die deliberativen Verfahren des Rapidly-Exploring-Random-Tree (RRT) [53] und der A*-Algorithmus [54] anhand statischer Szenarien dafür geeignet. Die notwendige Eigenschaft einer geringen Berechnungszeit ermöglicht ihre dynamische Anwendung innerhalb einer echtzeittreuen Regelung. Das RRT-Verfahren liefert eine Start-Ziel-Route anhand einer stochastisch gewachsenen Baumstruktur. Erreicht ein Ast den Toleranzbereich des Zielpunktes, kann der Pfad rückwärtig zur Wurzel aufgebaut werden. Parameterwahl und zusätzliche Bedingungen erhöhen die Effizienz des Verfahrens. Der A*-Algorithmus, als Erweiterung des graphentheoretischen Dijkstra-Algorithmus, verwendet eine Heuristik, um zielgerichtet in einer Menge von Knotenpunkten die effektivsten Kombinationen zwischen Start- und Zielpunkt zu untersuchen. Gefunden wird der kürzeste Weg bei günstiger Kostengewichtung von Knotenpunkten.

2.7.2 CMDragons Verhaltenssystem

Einen Überblick über die Struktur der verwendeten Robotersteuerung gibt Abb. 2.7. Der Programmteil des Servers bearbeitet das (Global-)Vision-System, den Tracker und die Kommunikation mit den Robotern. Die Gewinnung momentaner Objektinformationen durch die beiden Vision-Module der Bildverarbeitung und Objekterkennung wurde unter Abschnitt 2.6.2 beschrieben. Der Tracker des Servers wendet eine erweiterte Wahrscheinlichkeitsmethode nach dem Prinzip des Kalman-Bucy-Filters an, um genaue Positions- und Geschwindigkeitsaussagen von Ball und Robotern zu schätzen. Die Anwendung eines speziellen Tracking-Verfahrens nach Kim et al. [55] ermöglicht die Detektierung von hohen Ballschüssen.
Per UDP-Socket sind zwei weitere wichtige Klienten am Server angekoppelt. Einerseits berechnet das Programm die Verhaltensweisen der Roboter im Ball-

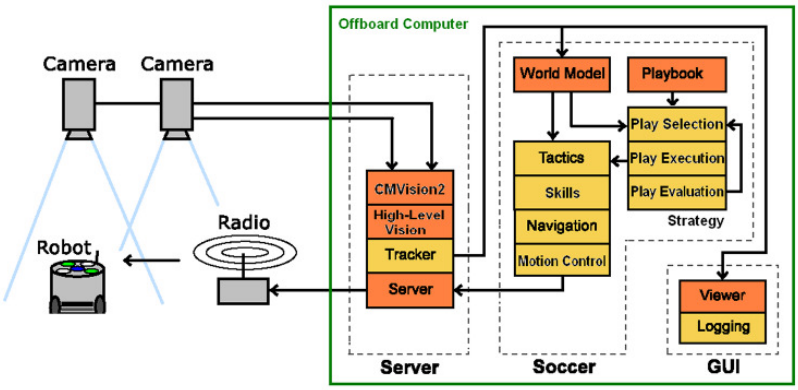


Abb. 2.7 Architektur des CMDragons Multi-Roboter-Systems 2006 [36, 37]

spiel bzw. überwacht die Roboternavigation, andererseits beobachtet man über die grafische Benutzeroberfläche GUI das gesamte System.

Im Programmteil Soccer interpretiert das World Model den eingehenden Tracking-Status, um Situationszustände zu extrahieren (z.B. Information zum Ballbesitz). Es dient als Datenbank der letzten Sekunden des gesamten Geschehens. Dies erlaubt dem Rest des Systems Zugriff auf den aktuellen Stand, die Abfrage der jüngsten Vergangenheit sowie auch Voraussagen zum künftigen Zustand durch den Kalman-Filter.

Das Programm basiert auf der hybriden Skills-Tactics-Plans-Verhaltensstruktur (STP) [56], die im Gegensatz zur allgemeinen Reaktions-Methodik eher für einzelne Situationen bestimmt und optimiert ist. Die höchste Ebene des Verhaltens bildet die Strategieschicht. Sie ist ein adaptives Strategiemodul, welches das Teamverhalten auch nach der eigenen Leistung aus dem Verlauf des Spieles anpasst. Diese reaktive Verhaltenssteuerung wird hierbei durch die Auswahl eines optimalen Spielverhaltens nach dynamischer Gewichtung umgesetzt [57].

Unterhalb der Strategieschicht steht eine Auswahl definierter Tactics (wie z.B. Angreifer, Torwart, Verteidiger) zu Verfügung. Tactics können wiederum auf Skills (Kompetenzen) zugreifen. Tactics und Skills sind mit der praktischen

Umsetzung der Roboteraktionen im Rahmen des definierten Play-Verhaltens beauftragt.
Für die robuste, effiziente Roboternavigation wird eine zufallsbasierte Pfadplanung verwendet. Der Execution-Extended RRT (ERRT), als Erweiterung der Rapidly-Exploring-Random-Tree (RRT) Konzeptfamilie von Kuffner und LaValle [53], erlaubt einen Kompromiss zwischen Pfadeffizienz- und Pfadlängenoptimierung. Unter der Verwendung von abgelegten Pfaden für eine Planung mit Vorwissen (Waypoint-Cache) wird der Algorithmus noch beschleunigt [58] und erreicht im aktuellen System 1-2ms Planungszeit, indem der ERRT-Planer die Dynamik des Spieles ignoriert. Gepaart mit dem Dynamics-Safety-Search-Algorithmus (DSS) wird ebenfalls die Kollisionssicherheit sichergestellt.
Die Anwendung und Erweiterung der DSS-Methode nach Brock [59] ermöglicht eine kollisionsfreie Kontrolle von Single-Agent- und koordinierten Multi-Agent-Systemen. Die DSS-Methode hält zugleich die Kollisionssicherheit durch ein komplettes Geschwindigkeits- und Beschleunigungsmodell des Roboters aufrecht [60]. In zweiter unterlagerter Schicht erfolgt die Bewegungssteuerung der Roboter mittels nahezu optimaler trapezförmiger Geschwindigkeitsprofile (Bang-Bang-Control) [56].

2.7.3 FU-Fighters Verhaltenssystem

Abschnitt 2.6.3 beschrieb die Generierung der Objektinformationen. Die Latenzzeit des Systems, die bis zur Umsetzung des gewünschten Verhaltens vergeht, wurde experimentell mittels sinusförmiger Steuersequenz gemessen und durch Vorhersage mit Hilfe automatischen Lernens eines neuronalen Netzes oder eines linearen Modells aktiv umgangen [61]. Beide Modelle nehmen als Eingaben die aktuellen kinematischen Zustandsparameter des Roboters und liefern als Ausgaben die zu erwartende relative Position und Orientierung nach 130ms.
Damit stehen dem Verhaltenssystem Eingangsdaten zur Verfügung, die echtzeittreu und realitätstreu von den Robotern ausgeführt werden können.

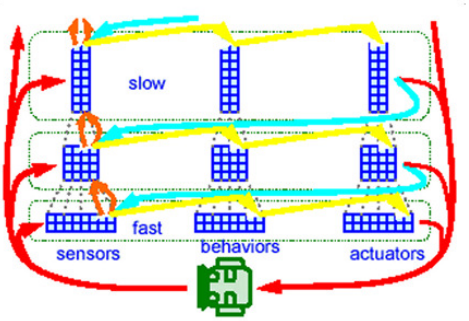


Abb. 2.8 Reaktiver Programmrahmen des FU-Fighter-Multi-Roboter-Systems [38]

Die Verhaltenskontrolle besteht aus einer Struktur von Sensoren, Verhaltensmustern und Aktoren (Abb. 2.8). Sie orientiert sich an der Subsumption-Architektur von Brooks [62], welche die Verhaltensmuster in Module unterteilt und dem Dual-Dynamics-Ansatz, welcher die Verhaltensmuster mit einer Aktivierungs- und einer Zieldynamik ausstattet [63].
Die Interaktion mit der Umgebung stimuliert in dem komplexen, vernetzten und hierarchischen Entscheidungsprozess die (Re-)Aktionen der Roboter. In Sensorenwerten gespeicherte Informationen sind z.B. Ist-Abstände, Ist-Positionen, Ist-Geschwindigkeitsvektoren, Ist-Beschleunigungen, aber auch der Ist-Winkel zum Ball, das beste Schussziel im gegnerischen Tor, die geschätzte Zeit und Position, bei der ein Roboter den Ball abfangen kann oder die aktuelle Spielsituation wie Elfmeter oder Freistoß.
Einzelne Verhaltensmuster sind für bestimmte zielführende Tätigkeiten des Roboters zuständig, wie z.B. Fahr-Hinter-Ball, Ausweichen, Anlauf oder Schuss. Aktorwerte sind wiederum z.B. Soll-Positionen oder Soll-Geschwindigkeiten.
In der Ebenenstruktur gibt es schnelle und langsamere Sensoren und zwar in dem Sinne, dass die Taktpausen ihrer periodischen Berechnung unterschiedlich lang sind. Schnelle Sensoren werden durch jedes neue Videobild erneuert und dienen als Eingang schneller Verhaltensmuster, wie z.B. Tormann, dessen Sensor die Ballbewegung mit 60Hz verfolgt. Bewegt sich der Ball in Richtung

des Tores, so wird das Stop-Verhalten aktiviert, das den Ball abfängt. Ist dies nicht der Fall, so versucht der Tormann sein Tor durch eigenes Positionieren gegen Angriffe zu schützen.
Ein bewegungsunfähiger, gefangener Roboter kann beispielsweise mit einem Sensorwert detektiert werden, der permanent die reale Befehlsumsetzung misst. Sinkt die Befehlsumsetzung unter einen Grenzwert, so wird als Reaktion das Verhalten zur schnellen Eigenrotation ausgelöst, um den Roboter aus seiner Zwangslage zu befreien.
Die Verhaltenssteuerung durch automatische Sensorüberwachung ermöglicht die parallele Aktivierung mehrerer Verhalten gleichzeitig. Verfolgt z.B. ein Roboter den Ball, aber ein Sensor detektiert eine bevorstehende Kollision, so wird die Hindernisvermeidung ebenfalls angesteuert. Das Ergebnis ist eine Fahrtrichtung, die proportional aus der gewichteten Summe beider Verhaltens-Trajektorien hervorgeht.
Verhalten können sich auch untereinander regulieren, weil der Aktorwert eines komplexeren Verhaltensmusters nicht unbedingt physisch umgesetzt werden muss, sondern als Zielvorgabe und Sensorwert eines primitiveren Verhaltensmusters dienen kann.
Der Spieler, der den Ball holen soll, hemmt das Ballsuchverhalten seiner Teammitspieler. Ist ihm das Ballholen selbst unmöglich, weil er blockiert wurde, so endet die Hemmung und startet ein Verhalten, mit dem der nächstgelegene Roboter diese Aufgabe übernimmt. Im ursprünglichen Ansatz existierte kein explizites Weltmodell, was gleichzeitig aber sehr schnelle Reaktionen ermöglichte.
Die Integration einer Pfadplanung verbessert die strategische Entscheidung zwischen konkurrierenden Verhaltensmustern, wodurch das implementierte Verhalten zwar nicht mehr rein reaktiv, aber doch durch die reaktive Komponente dominiert ist. Die Pfadplanung arbeitet mit einer optimierten Potentialfeldmethode, bei der mit dem A*-Algorithmus von der momentanen Position des Roboters zur Zielposition der effektivste Pfad gesucht wird [64].

2.8 Roboter- und Systemvoraussetzungen an der TU Ilmenau

Die technische Ausrüstung der stationären Anlage wird in den nachfolgenden Kapiteln im Zusammenhang der eigenen Neuerungen näher erläutert. Das Laborsystem basierte auf früheren Robotergenerationen der Jahre 2001 und 2003, die sich durch die Realisierung eines federbasierten Schussmechanismus und eines Differentialantriebes charakterisieren lassen. Das Etablieren neuer Tendenzen, wie sie die beiden vorgestellten Teams der Small-Size-League repräsentieren, prägte die Notwendigkeit zur zeitgemäßen Entwicklung des Roboters LUKAS.
Eigene Konstruktionskriterien, z.B. für omnidirektionale Räder, einen effektiven Schussmechanismus oder die Gestaltung eines leichten, leistungsstarken und kompakten Roboters mit einer robusten Fahrdynamik durch einen besonders niedrigen Schwerpunkt, ließen das Replizieren fremder Roboter nicht zu. Als Basis erweiterter Betrachtungen über Anwendungen mit kooperierenden Robotern oder eines modularen Robotersystems unterscheiden sich zudem die technischen Bedingungen und Anforderungen.

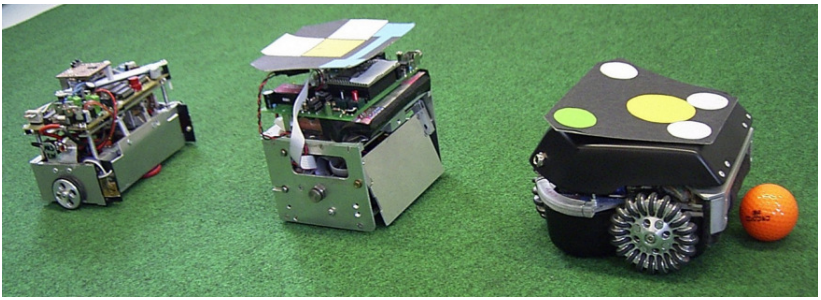


Abb. 3.1 KickElan-Roboter der Jahre 2001, 2003 und 2007

3 Roboter LUKAS

3.1 Konzeptrichtlinien

Innerhalb der offiziellen RoboCup-Regeln für konstruktive Gestaltungsmerkmale (siehe Abschnitt 2.4) ist viel Freiraum gegeben, die eigenen Vorstellungen eines Roboters zu verwirklichen. Abb. 3.1 zeigt die Eigenentwicklungen der Ilmenauer Fakultät für Maschinenbau, wobei das Modell 2007 auf dieser Arbeit beruht. In den folgenden Abschnitten sollen die praktischen und theoretischen Grundlagen sowie charakteristischen Merkmale der aktuellen Generation im Detail beschrieben werden. Erst die Erneuerung der Roboterhardware erschloss auch Alternativen im Bereich der Steuerung und Anwendung. Gleichwohl, wie die Planung durch Recherchen geprägt ist, steht die Auswertung der eigenen Erfahrungen und die technische Machbarkeit im Vordergrund. Das endgültige Produkt hat im Allgemeinen die Qualität, welche die geforderten Kriterien, Schwerpunkte und konkreten Zielvorgaben vorsehen, unter deren Prämisse es geplant und umgesetzt wurde.
Motiviert durch den konstruktiven Wettbewerb und die Möglichkeiten des technischen Materials stellen sich Maximalforderungen, die erst im Entwicklungsprozess zu praktikablen Lösungen führen. Über den objektiv messbaren Erfolg einer Konstruktion entscheidet schon das verwendete Grundprinzip.

Im Wesentlichen tragen aber subjektive Entscheidungen dazu bei, die Stärken einer Konstruktion zu fördern und ihre Schwächen zu mildern. Zum Verständnis des Roboters LUKAS sind hier die zugrunde liegenden konstruktiven Charakteristika wie folgt vorgestellt.

- **Grundforderungen zur technischen Umsetzung des Roboters**

- allgemeinen Baumaße der RoboCup-Small-Size-League
- Funktionsmasse kleiner 1,5kg und Geschwindigkeit größer 2m/s; Beschleunigung größer 4m/s² mit Fokus auf die frontale Hauptachse
- symmetrischer Schwerpunkt mit niedriger Schwerpunkthöhe
- Fähigkeit zur Ballhandhabung mittels Schuss- und Dribblermechanismus
- freie allseitige Manövrierfähigkeit mittels omnidirektionalen Antriebssystems
- Integration von Odometrie für Geschwindigkeitsregelung und bahngesteuerte Positionierfähigkeit
- Modularität und Austauschbarkeit der elektrischen Komponenten für den Fall der Erweiterung oder des Umbaus der Konstruktion

Die mögliche Funktionsintegration einzelner Roboterkomponenten vermindert die gesamte Anzahl der Teile sowie die Masse. Im Laufe der Realisierung und Erprobung konnten Forderungen wiederholt verschärft und eingegrenzt werden. Die Erfahrungen im praktischen Umgang mit dem Roboter sind unabdingbare Voraussetzung und Quelle notwendiger Korrekturen auf dem Weg zum optimalen Ergebnis. Richtlinienkataloge sind somit auch ein Ausdruck des technischen Entwicklungsstandes. In Erweiterung der Grundforderungen gelten hierzu noch die folgenden Konstruktionsrichtlinien. Sie geben den Einzelkomponenten die spezifischen Empfehlungen:

- **Elektrische Aktoren**

- Verwendung belastbarer Motoren mit Leistungsressourcen für geforderte Beschleunigung und Robustheit

- odometrische Erfassung der Bewegung durch hochauflösende Impulsgeber
- Einsatz eines robusten, kräftigen und zielgenauen Schussmechanismus
- Ballkontrolle mittels rotatorischen Dribblermechanismus
- Mechanismen mit wenigen, justagefreien mechanischen Elementen in minimalen, kompakten Bauräumen

- **Steuerelektronik**
- modulare Demontage der elektrischen Komponenten durch Steckersystem
- Verwendung eines leichten, betriebssicheren Mainboardsystems für einen niedrigen Schwerpunkt
- Mainboard mit allen elektronischen Komponenten unter galvanischer Trennung der Steuerseite vom Leistungsteil, sowie Reserveschnittstellen für weitere Elemente

- **Elektrische Energiespeicher**
- Nutzung ungiftiger, leichter Akkumulatoren mit hoher Spannungsebene
- außenliegende Anordnung für erleichterte Handhabung im Havariefall
- mögliche Umrüstung eines Schnellwechselsystems vorsehen

- **Konstruktive Strukturelemente**
- Struktursymmetrie und Gewichtsbalance für optimales Fahrverhalten
- montagefreundliches Tragwerk mit zentralem Basisbaustein als Innenrahmen für den erleichterten Zugriff und die Montierbarkeit aller Komponenten
- Sandwich-Modulbauweise ohne mechanische Spannungen unter den Komponenten
- kreisorientierte Grundform für Rotationsfreiheit und optimale Bauraumnutzung
- Verminderung elektrischer Fehlergefahr und Gewicht durch isolierenden Kunststoff

- fixierter Trikotträger für geringe Lagefehler der Bildverarbeitung
- Aufbau mit genügend Bodenfreiheit (4mm) über rauen und welligen Untergrund
- Einsatz laufruhiger omnidirektionaler Räder mit robusten Kranzrädchen
- Räder und Radaufhängung uniform, frei zugänglich und leicht wechselbar
- Zwangfreiheit des Antriebssystems durch dreirädrigen Stand
- orthogonaler Achswinkel erleichtert konstruktiven Aufbau und maximiert Vortrieb[14]

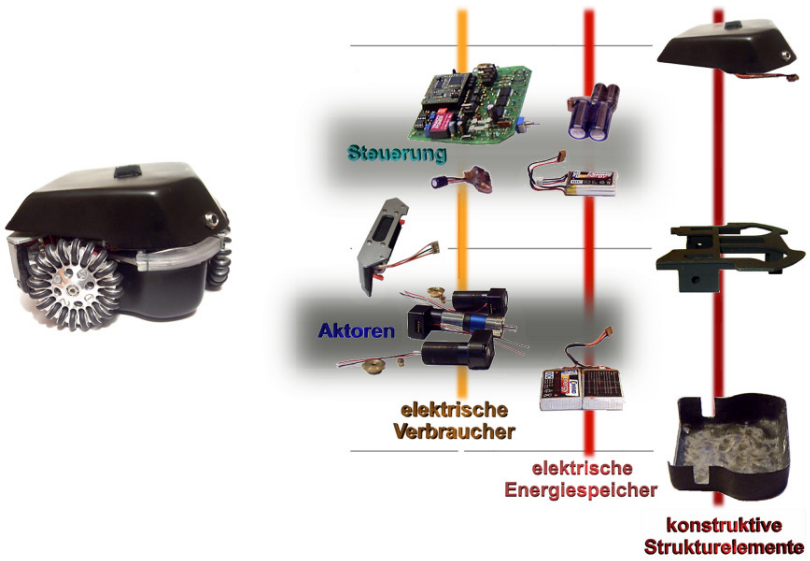


Abb. 3.2 Kategorisierung der Komponenten in der Ebenenstruktur

[14] Das bedeutet, dass es für eine maximale Frontalgeschwindigkeit empfehlenswert ist, die Vorspur in Frontalrichtung parallel auf 0° zu setzen und einen Winkel zur Querachse von 90° anzuwenden. Räder, die in Laufrichtung mit großem Vorspurwinkel gegeneinander arbeiten, summieren ihre vektoriellen Kraftkomponenten für die frontale Fahrtrichtung (Kräfteparallelogramm). Dies würde eine geringere Endgeschwindigkeit ergeben.

3.2 Technische Realisierung

Mit dem Roboter LUKAS ist eine omnidirektionale Fahrmaschine entstanden, deren aktive Hauptelemente die Antriebs- und Schusseinheiten bilden [65]. Es wurde eine Grundstruktur gewählt, die es erlaubt, alle Komponenten mit der höchsten Packungsdichte anzuordnen. Wie in Abb. 3.2 zu sehen ist, gliedert sich die Struktur in zwei Horizontalebenen für Steuerung und Aktoren. Rahmen und Gehäuseschalen fixieren, stabilisieren und schützen die funktionalen Elemente. Die vertikale Anordnung der Komponenten folgt im Wesentlichen den Grundforderungen zu passiver Sicherheit, Schwerpunktabsenkung und Bauraumkomprimierung. Der schwere Leistungsakkumulator ist dadurch auch sicher von der Elektronik entfernt auf der Unterseite angeordnet. Aus materiellen Erwägungen und zur Gewichtsersparnis wurde der Aufbau eines dreirädrigen Antriebes favorisiert, wobei der typische Radstand mit 120° Achswinkeln in die orthogonale Achslage abgewandelt wurde. Begründet durch die dichte Nutzung des Bauraums, der Anwendung leistungsstarker Antriebsmotoren und der Unterbringung je eines ausreichend dimensionierten Schuss- und Dribblermechanismus ergaben sich auch Vereinfachungen und Vorteile mit der Umsetzung einer orthogonalen Achslage. Die Verknüpfung einer hohen frontalen Geschwindigkeit und Positionierungsgenauigkeit mit dem gleichzeitigen Freiheitsgrad der Querfahrt entspricht damit der Erweiterung eines Differentialantriebes um eine orthogonale Radachse. Der entkoppelte, ebene Antrieb basiert auf drei geregelten Gleichstrommotoren mit kombinierten Impulsgebern. Der innere Zentralrahmen trägt alle Aktoren und bildet den Kern der modularen Architektur.
Die Verwendung von Polyamid als leichtes Trägermaterial ist eine weitere Verbesserung [65]. Der Schwerpunkt liegt ideal mittig ausbalanciert im Modell. Die ca. 33mm Bodenhöhe des Schwerpunktes kann erreicht werden, weil alle gewichtigen Elemente die niedrigstmögliche Einbaulage erhielten. Die eigens neu entwickelte Hauptplatine auf der leicht zugänglichen Oberseite des Modells umfasst alle elektronischen Steuer- und Lastelemente zur Überwachung

der Signalverarbeitung, der Regelung der drei Fahrmotoren sowie aller übrigen Sensoren und Aktoren. Weitere technische Leistungsdaten können aus Abb. 3.3 entnommen werden.

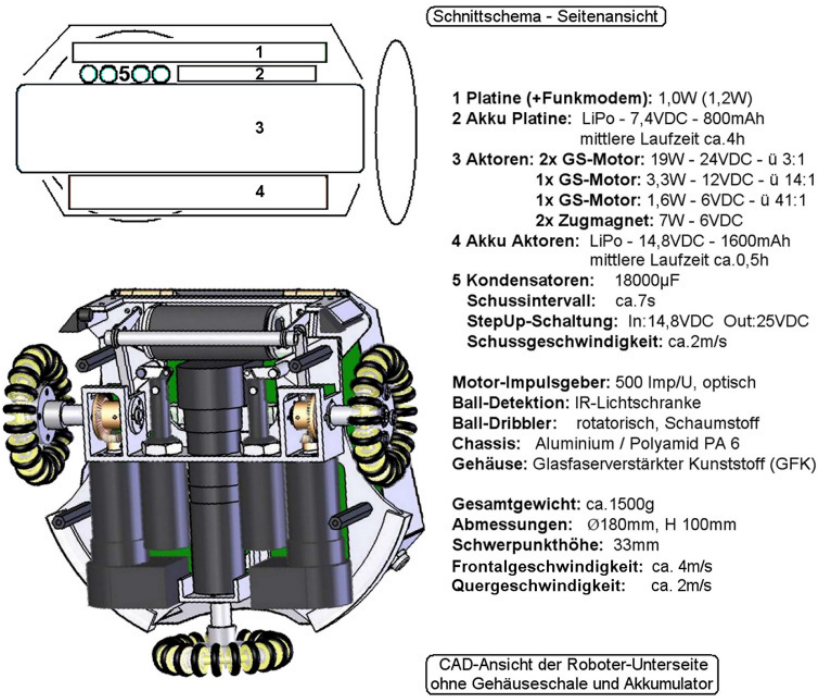


Abb. 3.3 Konstruktionstechnische Daten des Roboters LUKAS

Anders als die im Abschnitt 2.5 vorgestellten Konstruktionen mit Bodenplatte und äußerem Rahmen beruht das Modell LUKAS auf dem Sandwich-Grundprinzip. Trotz Platzreserven sind die verwendeten Akkumulatorkapazitäten so bemessen, dass eine volle Spielzeit gefahren werden kann. Das Ziel ist die Fusion von Geschwindigkeit, Schusskraft, Flexibilität und Ausdauer. Das bedeutet jedoch nicht, dass alle Leistungsoptionen schon erschöpft sind. Die Hauptpriorität hatte der Aufbau eines stabilen, funktionsfähigen Roboter-

systems. Verglichen mit anderen Robotermodellen der Liga ist der Roboter LUKAS nicht ausschließlich auf Hochleistungen ausgerichtet und hat in der heutigen Konfiguration noch ungenutztes Geschwindigkeits- und Schusspotential. Die Torgefährlichkeit kann mit dem Austausch der Antriebsmotoren durch Typen mit kleinerer Nennspannung und der vielfachen Steigerung der Schussspannung noch wesentlich erhöht werden. Eine großzügigere Motordimensionierung bietet nicht nur zusätzliche Kraftreserven, sondern auch Schutz gegen Überhitzung im Dauerlastbetrieb und die Gewähr von Robustheit bzw. Langlebigkeit. Obwohl sich der Roboter in diesem Aspekt von denen anderer Teams unterscheidet, verfügt er anhand der technischen Daten über eine vergleichbare Laufzeit, Agilität und Dynamik.

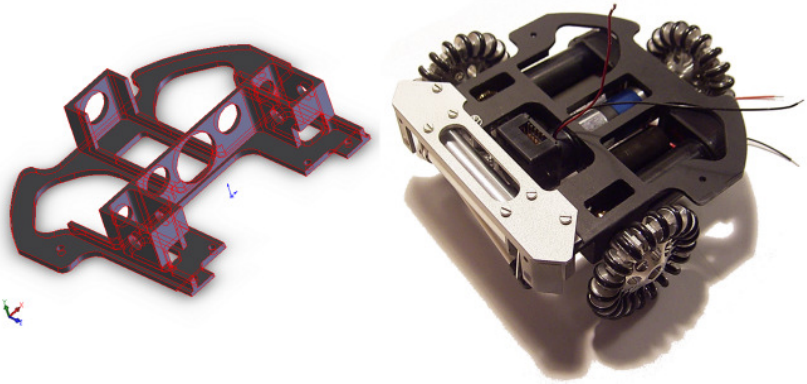


Abb. 3.4 PA6-Chassis als CAD-Modul und in Bestückung der kompletten Aktorik

3.2.1 Chassis

Das zentrale Polyamid-Chassis sichert Robustheit im Einklang mit kompaktem Leichtbau ($m = 44g$). Wie aus Abb. 3.4 ersichtlich ist, stellt das Chassis den inneren Zusammenhalt zwischen Antriebsmotoren, Radwellen, Schussmechanismus und Dribblermechanismus her. Um Kollisionsstöße aufnehmen und

weiterleiten zu können, wurde ein stabiles Bauteil gefordert, das nach dem Prinzip der direkten und kurzen Kraftleitung konstruiert ist [66]. Der montagefreundliche modulare Aufbau ist strukturcharakteristisch und gewährt eine sicherheitsrelevante Trennung von Mechanik, Leistungsakkumulator und elektronischen Elementen.
Die Raumaufteilung legte somit die Grundform fest, in die sich der Zentralrahmen einpassen muss. Obwohl spielarme Direktantriebe die bessere Wahl darstellen, war der Einbau der zwei Kegelradgetriebeboxen, als Kompromiss zwischen minimalem Bauraum und maximalen Antriebskräften, erforderlich. Kegelradgetriebe sind jedoch sehr empfindlich gegenüber Achsabstandsänderungen und müssen wesentlich enger lagetoleriert werden als Stirnräder [67]. Um eine exakte Lage der Kegelräder zu erreichen, ist deren axiale Position, die Lagerung der Wellen und die Steifigkeit des Gehäuses zu gewährleisten. Die notwendige Unveränderlichkeit der Lage wird durch das Prinzip der gemeinsamen Fertigung erreicht. Alternativ zum Aluminium ist als Rahmenmaterial Polyamid (PA6) im Einsatz und überzeugt mit den positiven Eigenschaften:

- Schall- und Vibrationsdämpfung im integrierten Getriebe
- Kurzschlussschutz von Platine und Akkus durch Isolation bei Kontaktfehlern
- Leichtbau
- Steifigkeit und Bruchsicherheit der konstruktive funktionsrelevanten Teile, sowie Verwindungsstabilität im vertikalen Verbund
- günstige Herstellung durch Fräsbearbeitung

Die meisten technisch bedeutsamen Polyamide zeichnen sich durch geringes Gewicht, eine hohe Festigkeit, Steifigkeit und Zähigkeit aus, besitzen eine gute Chemikalienbeständigkeit und Verarbeitbarkeit und sind bis zu 150°C formstabil [68, 69]. Das geringe Gesamtgewicht von ca. 1,5kg hat somit nicht nur seine Ursache in der Verwendung von Lithium-Polymer-Akkumulatoren und Glasfaser-Gehäuseschalen.

3.2.2 Schussmechanismus

Trotz potentieller Alternativen werden magnetische Schussmechanismen aufgrund der flexiblen Anordnung des Energiespeichers (elektr. Kondensatoren), sowie der softwareseitigen Kraftdosierung, Schussleistung und Robustheit favorisiert. Magnetische Impulsgeber erfordern einen geringen Ansteuerungsaufwand und sind im Ruhezustand frei von Kräften.
Im Modell LUKAS beruht der Mechanismus auf zwei Zugmagneten und ist an die Platzbedingungen im Chassis angepasst. Die beiden Magneten bewirken bei gleichzeitiger Aktivierung eine frontale Schlagrichtung und im Einzelbetrieb eine seitliche Ablenkung. Da es sich nur um Kurzzeitbetrieb handelt, können die Schussmagneten für ein gutes Masse-Leistungs-Verhältnis mit einem Vielfachen ihrer Nennspannung betrieben werden, um die nötige Kraft zu entwickeln. Vorausgesetzt, die durchschnittlich zugeführte Leistung überschreitet nicht die Nennleistung, kann ein Magnet dieser Serie mit höherer Leistung und kleineren Tastverhältnissen betrieben werden, so wie es in den Herstellerkennlinien in Abb. 3.5 ersichtlich ist.

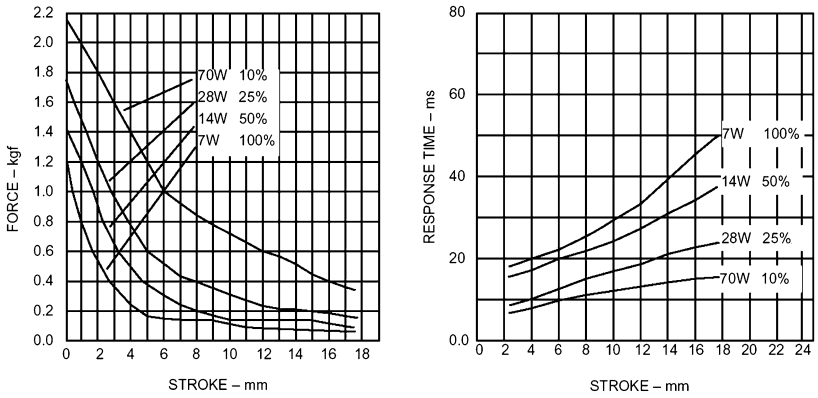


Abb. 3.5 Leistungskennlinien des BlackKnight 121-Magneten [71]

Einen 6V-Magneten der Nennleistung $P_{Nenn} = 7W$ mit zehnfacher Impulsleistung $P_g = 70W$ zu betreiben, erlaubt nur einen Tastgrad $g = 0{,}1$, d.h. 10%. Die nötige Spannung $U_g \approx 19V$ errechnet sich aus Gleichung (3.1) [70].

$$P_g = \frac{(U_g)^2}{R} = \frac{P_{Nenn}}{g} = \left(\frac{U_{Nenn}}{\sqrt{g}}\right)^2 \cdot \frac{1}{R} \qquad (3.1)$$

Wobei die ermittelte Fläche unter den entsprechenden Kraft-Hub-Kennlinien des BlackKnight 121-Magneten [71] der Hubarbeit W_{mech} entspricht, die bis zum Endpunkt des Hubes an den Mechanismus abgegeben wird. Die Hersteller-Kennlinien in Abb. 3.5 beschreiben jedoch nur das statische Kraft-Hub-Verhalten. Weil sich der Anker bei einem Schuss allerdings sehr schnell bewegt und somit zusätzlich mechanische Energie und Energie zum Aufbau des Magnetfeldes verloren geht, wird ein Wirkungsgrad von $\eta = 0{,}2$ angenommen. Überträgt man die verbleibende Arbeit von zwei Magneten mechanisch in die kinetische Energie eines Golfballes der Masse $m = 46g$, so besteht die Abhängigkeit zwischen Hubenergie und eingespeister Spannung, wie sie in Abb. 3.6 für einen jeweiligen Hubweg von $\delta = 12mm$ demonstriert ist.
Im Spannungsbereich bis $160VDC$ [71] erreicht der Ball nach Gleichung (3.2) die Schussgeschwindigkeit, wie sie das rechte Geschwindigkeits-Spannungs-Diagramm in Abb. zeigt.

$$2 \cdot W_{mech} \cdot \eta = E_{kin} = \frac{m \cdot v^2}{2} \quad \rightarrow \quad v = \sqrt{\frac{4 \cdot W_{mech} \cdot \eta}{m}} \qquad (3.2)$$

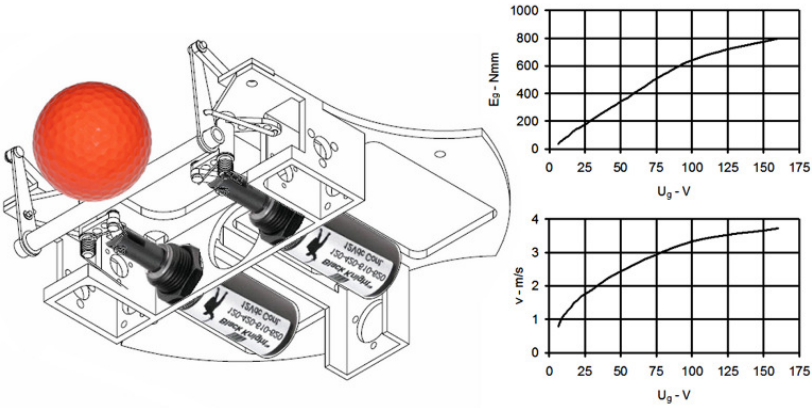


Abb. 3.6 Hubarbeit- und Ballgeschwindigkeitsabhängigkeit bei einem Hubweg von 12mm

Das Abfallen der Kurven ist auf die Sättigung des Magnetkreises zurückzuführen. Eine Steigerung der zugeführten elektrischen Leistung bewirkt zunehmend weniger Gewinn der Hubarbeit. In ähnlichem Verhältnis stehen die Höhe der Impulsspannung und die Schussgeschwindigkeit des Balles zueinander. Die maximale Startgeschwindigkeit des Balles liegt mit dem Mechanismus bei ca. $3{,}7\,m/s$. Aufgrund ihrer elektronischen und mechanischen Robustheit und Bauraumeffizienz ist die konservative Variante mit 25VDC Eingangsspannung und $1{,}8\,m/s$ Ballgeschwindigkeit im Roboter realisiert.

Die mechanische Energieübertragung basiert auf einem Kurbelhebel, dessen Arbeitsbereich der Kurbeldrehung $\varphi = 90°$ betragen sollte. Ideal ist, nach Abb. 3.7, ein Kurbelhebel mit einem Öffnungswinkel $\alpha = 90°$, der das vom Winkel abhängige Übersetzungsverhältnis $i(\varphi)$ in Form einer Tangensfunktion umsetzt.

$$i(\varphi) = \frac{s_{ab}}{s_{an}} = \frac{\sin(\varphi)}{\cos(\varphi)} = \tan(\varphi) \qquad (3.3)$$

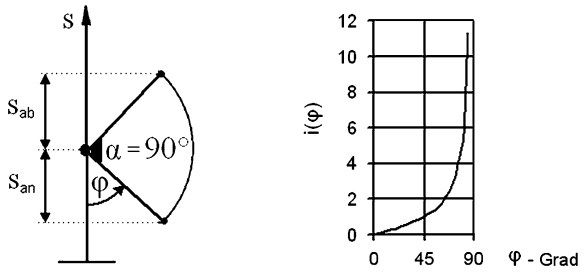


Abb. 3.7 Schubübersetzung der rechtwinkligen Kurbel

In Erinnerung an das Hebelwurfprinzip der Blide [72], einer mittelalterlichen Steinschleuder mit beweglichem Gegengewicht, hat das Schubkurbelgetriebe im Roboter nicht nur die Aufgabe, dem Ball die größtmögliche Startgeschwindigkeit zu geben, sondern auch den nichtlinearen Kraftverlauf der Magneten in eine gleichmäßige Beschleunigung am Ball zu wandeln. Somit verfügt der Schussapparat neben der elektrischen Leistungssteigerung auch über eine verbesserte Übersetzung des Energietransportes, welche am Ende der Ballbeschleunigung eine minimale kinetische Energie in den beweglichen Teilen der Mechanik verbleiben lässt. Magnetanker und Hebel kommen ohne Anschlag zum Stillstand.

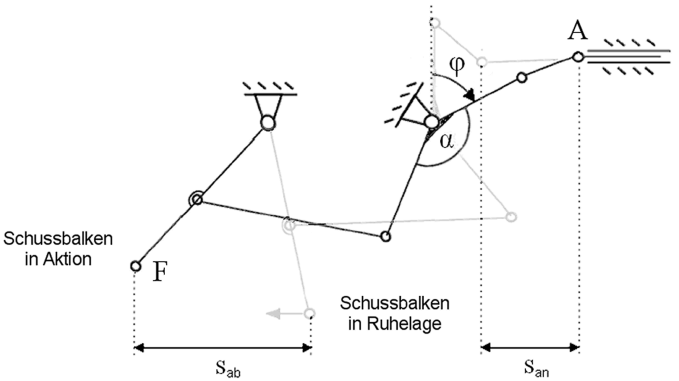


Abb. 3.8 Schussmechanismus im entspannten und angezogenen Zustand [70]

Die zu Beginn kleine Übersetzung soll am Ende der Beschleunigungsphase unendlich groß werden. In Abb. 3.8 ist dargestellt, dass der Öffnungswinkel des Kurbelhebels bauraumbedingt mit $\alpha = 129°$ integriert ist. Als charakteristische Größen erfüllen jedoch die resultierende Übersetzung und die Ankergeschwindigkeit die gestellten Erwartungen und weisen das für die Rückgewinnung der kinetischen Energie des Ankers benötigte Verhalten auf. Abb. 3.9 zeigt im linken Teil den Kurvenverlauf des Gesamtübersetzungsverhältnisses $|i_{ges}|$ der Wegänderungen von Punkt F zu Punkt A.

Beginnend mit $|i_{ges}| \approx 1{,}2$ verändert es sich im Anzug zum Totpunkt $\varphi = 66{,}7°$ zu $|i_{ges}| \to \infty$. Der Anker hingegen verliert im letzten Drittel der Anzugsphase, trotz der stärksten Magnetkräfte, seinen Impuls bis zum Stillstand, so wie es im Geschwindigkeits-Zeit-Diagramm der Abb. 3.9 dargestellt ist. Voraussetzungen zur Berechnung dieser Kurve sind die Randbedingungen der Schussschwinge, mit den Geschwindigkeiten $v_0 = 0$, $v_{end} = 1{,}8\,m/s$, der Anzugsdauer von $\Delta t = 30ms$ und der Annahme einer konstanten Beschleunigung des Balles [70].

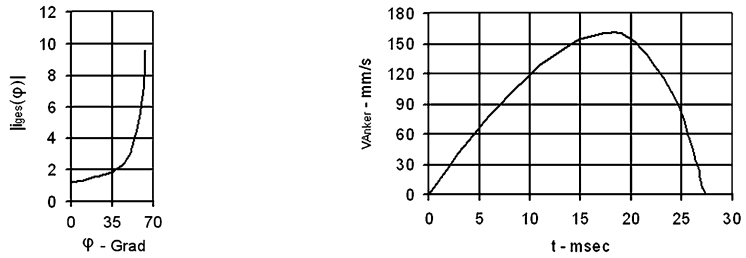


Abb. 3.9 Kurvenverläufe der Gesamtübersetzung und der Ankergeschwindigkeit

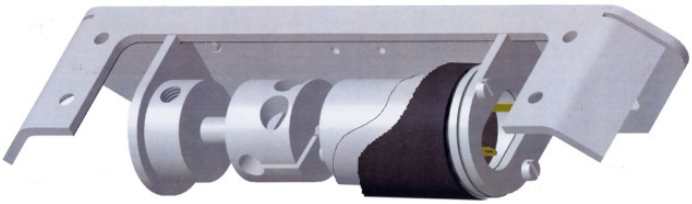


Abb. 3.10 Schnittansicht der Dribblerbaugruppe

3.2.3 Dribblermechanismus

Für den erwünschten Hafteffekt eines mitzunehmenden Balles sorgt die rückwärtsdrehende Schaumstoffwalze des Dribblers. Der Sensor der Infrarot-Lichtschranke (IR) und die Mitnehmerrolle des Dribblers besitzen zueinander einen definierten Abstand, der eine sichere Ballerkennung gewährleistet. Die Funktion einer automatisch ausgelösten Schuss- oder Dribbleraktion auf der Interruptebene der Robotersteuerung beweist im Feldeinsatz einen erfolgreichen Ballumgang. Balldetektierung und Dribblermechanismus bilden somit eine kompakte Baugruppe, die auf dem Chassis aufsitzt. Gemäß der besonderen Stoßbelastung wurde an der Roboterfront ein stabiler Schutzrahmen aus Aluminium verwendet. Dieser Träger ist mittels Distanzbolzen und Dämpferelementen derart fixiert, dass lediglich die Höhenlage der Einheit über der Fahrbahn justiert werden muss, um den Ball in geeigneter Weise zu führen.
Die Schnittdarstellung des Rotationszylinders in Abb. 3.10 offenbart die interne Nutzung des Raumes. Nach dem Vorbild eines Rohrantriebes sind Getriebemotor, Kupplung und Lagerungen axial hintereinander untergebracht, so dass die Abtriebswelle des Getriebes durch wenige Verbindungselemente direkt mit der Mitnehmerrolle verbunden ist und damit für eine ausreichend hohe Laufruhe, Kraftübertragung und Spielarmut sorgt.

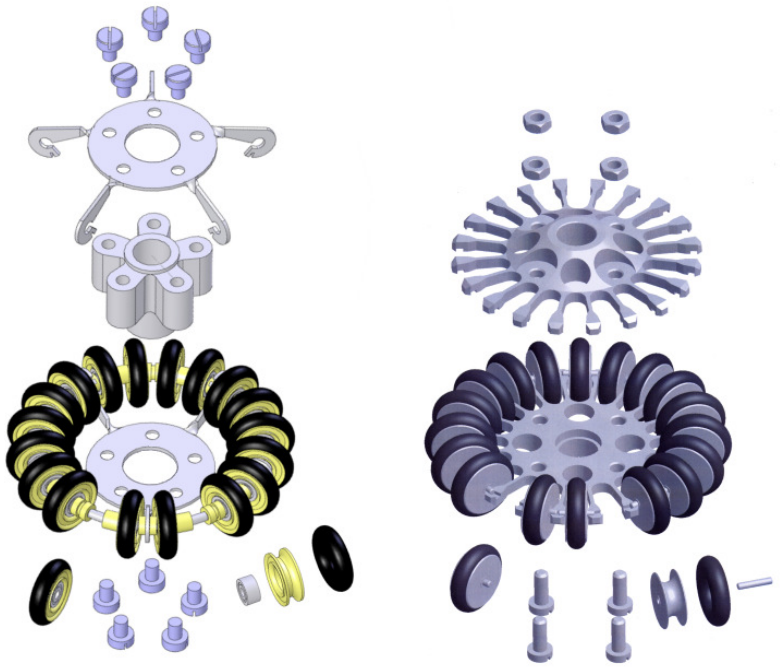


Abb. 3.11 Explosionsdarstellung der Radstrukturen JAHN/2006 und WALKLING/2007

3.2.4 Radkonstruktion

Basierend auf dem Prinzip des omnidirektionalen Grabowiecki-Rades (US-Patent von 1919) [73] garantieren die für den Roboter entwickelten 20-teiligen Räder die geforderten Eigenschaften. Die Bewegungsqualität und Dynamik holonomer Fahrwerke ist nicht nur abhängig von der Ausrichtung der Antriebswellen, sondern ebenso bestimmt durch die Konstruktionsweise der Laufräder. Nach anfänglicher Erprobung käuflicher Großserienkomponenten fiel aufgrund ungenügender Eignung die Entscheidung für den Einsatz einer eigenen Modifikation. Leichtlauf, Laufruhe, Traktion und Festigkeit können

durch die Gestaltung von Material und Form den Leistungsbedürfnissen besser angepasst werden. Der erste eigene Entwurf war eine gebogene, der zweite eine gefräste Felgenvariante für die omnidirektionalen Räder, links und rechts dargestellt in Abb. 3.11. Der Prototyp JAHN/2006 (Ø 64mm, Tiefe 16mm) hat durch eingesetzte Miniaturkugellager einen sehr leichten, spielfreien Querlauf, einen geringen Fertigungsaufwand und in der Materialpaarung Kunststoffnabe/Stahlfelge eine Masse von nur 39g. Der frühe Einsatz dieser Räder ermöglichte die für die Ausarbeitung der robotertypischen Steuermethoden erforderliche Präzision und Stabilität des Antriebes. Mechanische Belastung, verlustarme Querfahrt und die Realisierung der Bahnsteuerung erfüllten die gestellten Erwartungen auf ebener Teppichoberfläche.
Es zeigte sich aber auch, dass die Räder im kompromisslosen Feldeinsatz mitunter heftige seitliche Kollisionsstöße kompensieren müssen. Mit gleicher Radzahl und gleicher Abmessungen beruht alternativ der robustere Radtyp WALKLING/2007 auf dem Prinzip symmetrischer Halbschalen. Er überzeugt mit einer höheren Festigkeit gegen seitliche Stoßbelastung, mit einer geringeren Unwucht und einen sehr guten Rundlauf. Die bewährten O-Ringe aus Fluor-Kautschuk befinden sich hier auf gleitgelagerten Kranzradfelgen, weil mit dem Vorteil des Leichtlaufs von Kugellagern nicht die damit verbundenen Kosten zu rechtfertigen waren.
Mit zwanzig Kranzrädchen (engl.: sub-wheel) pro Omni-Rad liegen diese Entwürfe im Mittelfeld gebräuchlicher Diskretisierungen. Schäden an der Lauf-Gummierung wurden bisher nicht festgestellt. Je größer die Teilung gewählt wird, desto ruhiger und exakter ist das Lauf- und Positionierverhalten in Längsrichtung, da jederzeit mindestens 2 Kranzrädchen Bodenkontakt besitzen sollten. Dies ist besonders für die exakte Steuerung vierrädriger Roboter wichtig, um nicht zufällig ein Rad ohne tragenden Bodenkontakt zu erhalten. Aber auch die Reibung in Querrichtung, Verschleiß, Belastung und Ausfallwahrscheinlichkeit der kleineren Radstrukturen nimmt zu und muss in der Auswahl geeigneter Materialen berücksichtigt werden.
Anhand der vorgestellten Roboter anderer Mannschaften lassen sich noch zwei Tendenzen in den Antriebssystemen aufzeigen (siehe Tab. 2.3). Die Funktionsintegration der Antriebsräder mit umlaufendem Stirnrad spart die

Verwendung eines motorseitigen Vorschaltgetriebes. Die leichten, unterdimensionierten Antriebsmotoren erbringen die geforderte Leistung mittels PWM-kontrollierter Überlastung. Andererseits setzt man auf die Anwendung sehr belastbarer bürstenloser Motoren. Die elektronischen Anforderungen zur genauen Regelung und Positionierung sind hingegen anspruchsvoller.

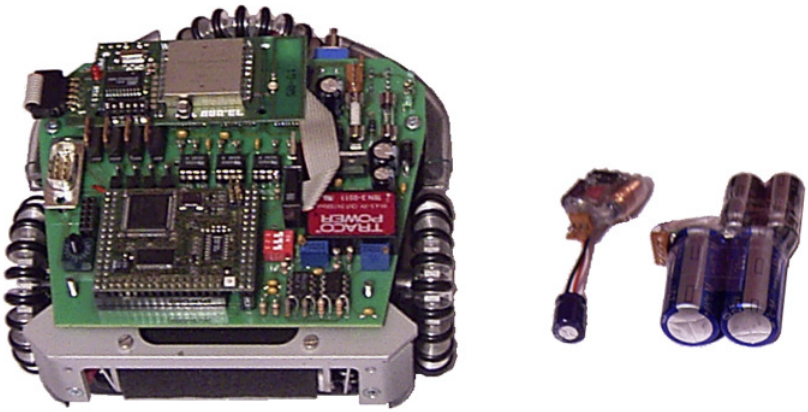


Abb. 3.12 Hauptplatine des Roboters LUKAS mit nebenliegender Schusselektronik

3.2.5 Elektronische Komponenten

Das Mainboard zeichnet sich durch bewährte Robustheit und Kompaktheit aus und enthält alle notwendigen elektronischen Komponenten unter galvanischer Trennung der Steuerseite von der Lastseite.
Auf Grund eines niedrigen Schwerpunktes, der Stabilität und Zuverlässigkeit wird eine flache zentrale Hauptplatine favorisiert. Das speziell gefertigte Board, links in Abb. 3.12 dargestellt, entstand in Zusammenarbeit mit industriellen Partnern. Die separate Anordnung des Controllermoduls, des Funkmodems und der Schusselektronik hat den Vorteil, im Fehlerfall oder für Nachrüstungen die prägnantesten Komponenten flexibel austauschen zu

können. Alle externen Aktoren und Sensoren sind über sichere Steckverbinder angekoppelt. Den Kern des TQM164C Controllermoduls bildet, neben dem 256KByte Flash Speicher und dem 256 KByte SRAM, der Infineon C164 Mikrocontroller [74]. Mit einer Taktfrequenz von 20MHz bearbeitet der Controller alle Hauptaufgaben der seriellen Kommunikation, Odometrie, Generierung der PWM, sowie die Positionsregelung und Verhaltenssteuerung des Roboters. Ausgeführt werden die Steuersignale durch diverse Power-MOSFET und drei H-Brücken MC33186DH, die lastseitig die Motorströme polrichtig schalten. Für die Unabhängigkeit der Signalverarbeitung von den Spannungsschwankungen der Motoren werden Steuer- und Lastseite der Elektronik, galvanisch getrennt, durch einen jeweils eigenen Akkumulator betrieben. Auf der Platine sind neben der elektrischen Spannungsversorgung auch Jumper-, Schalt- und Steckverbindungen zum Einrichten der individuellen Roboterkonfiguration enthalten. In Erweiterung der vorhandenen Balldetektierung und Motor-Encoder können noch freie Controller-Eingänge und die integrierte CAN-Schnittstelle mit Abstands-, Lage- oder Beschleunigungs-Sensoren aufgerüstet werden. Alternativ ist das (über die RS232-Schnittstelle kommunizierende) Funkmodem gegen ein Bluetooth-Modul oder bidirektionale Funksysteme anderer Frequenzbänder austauschbar. Die Integrität der Daten sichert momentan das Sendemodul WIZ-434-SML-IA/5V der Firma Aurel mit 434MHz. Es besitzt eine gebäudeinnere Reichweite von ca. 15m [75]. Die ausgelagerte Schusselektronik, zu der der Step-Up-Converter und die zugehörende Kondensatorbatterie zählen (Abb. 3.12, rechts), ist zusammen mit der gesamten Verkabelung und dem kleineren steuerseitigen Akkumulator im Zwischenraum von Platine und Chassis eingeordnet. Soll die Ballschussgeschwindigkeit noch erhöht werden, so kann diese Einheit leicht gegen eine Schaltung höherer Schussspannung ausgetauscht werden.

Die Bordspannung von $U_{in} = 14{,}8V$ erhöht in Abb. 3.13 der kleine, leistungsschwache Step-Up-Konverter MC 34063 A auf eine Kondensatorladespannung von $U_{out} = 25V$.

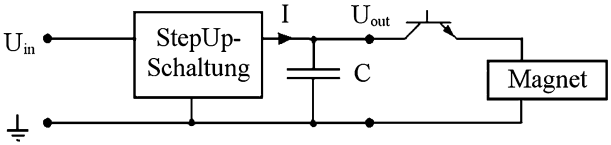


Abb. 3.13 Step-Up-Konverter in typischer Einsatzweise

Die im Kondensatorblock gespeicherte Energie steht nach abgeschlossenem Ladevorgang den daran geschalteten Schussmagneten zur Verfügung. Ein impulsartig ausgelöster magnetischer Schuss durch vielfache Nennspannung ist im Vergleich zum Nennbetrieb des Magneten wesentlich leistungsfähiger (siehe Abb. 3.6). Hierzu ergibt sich aus Gleichung (3.4) in Anknüpfung der Kennwerte aus Abschnitt 3.2.2 eine für den Kondensator, bei $U_g = 25V$, notwendige Mindestkapazität zu $C = 960\mu F$:

$$C = \frac{2(2 \cdot W_{mech})}{U_g^{\,2}} \qquad (3.4)$$

Ein Kondensator dieser Kapazität enthält allerdings lediglich eine der mechanischen Nutzenergie äquivalente Ladung, um den Ball zu beschleunigen. Die Energie zum Aufbau des Magnetfeldes und zur Kompensierung auftretender Wärmeverluste ist hier noch nicht inbegriffen. Das Diagramm der Hubarbeit (Abb. 3.6) veranschaulichte zudem den Zusammenhang durch den in Sättigung geratenden Magnetkreis. Je weiter die Impulsspannung U_g gesteigert wird, umso weniger entspricht die Hubarbeit W_{mech} dieser Steigerung. Die Impulsleistung P_g erhöht sich jedoch mit quadratischem Verhalten zur nutzbaren Energie. Aus praktischer Erfahrung zeigt sich bei der Verwendung von zu geringen Kapazitäten ein frühzeitiger Zusammenbruch der Spannung innerhalb des $30ms$ dauernden Kurzschlussimpulses. In Anwendung einer

Kapazität von $C = 18000\mu F$ wird die erwartete Ballgeschwindigkeit von $v = 1{,}8\,m/s$ erreicht, wobei die Kondensator-Restspannung nicht unter $14{,}3V$ sinkt und die Step-Up-Schaltung somit eine Schussrate von ca. $7s$ erlaubt [70]. Der gewählte Tastgrad, die Belastbarkeit des Magneten und des Hebelmechanismus bieten noch Reserven für eine höhere Schussrate und Impulsspannung (Abschnitt 3.2.2). Ein Konzept zur Rückgewinnung der Magnetfeldenergie liegt nach Abb. 3.14 vor und kann im zukünftigen Ausbau höherer Impulsspannungen helfen, die Effizienz des Systems zu verbessern. Nach dem Abschalten der speisenden Impulsspannung erzwingt die wirkende Induktionsspannung ein Weiterfließen des Stromes, wodurch sich bis zum Abklingen des vorhandenen Magnetfeldes der Schusskondensator wieder auflädt.

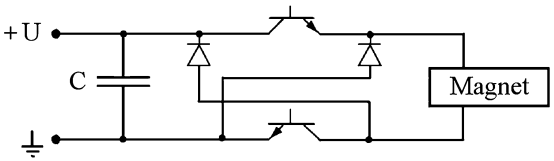


Abb. 3.14 Schaltung zur Rückgewinnung der Magnetfeldenergie

3.3 Kinematik

3.3.1 Modell

Die momentane Roboterposition ergibt sich bei radgetriebenen Plattformen durch aktive Ansteuerung der Räder mit den Winkelgeschwindigkeiten $\omega_i(t), i \in \{1,2,3\}$. Die Mobilität und Beweglichkeit omnidirektionaler Radsätze bewirkt hierbei die unmittelbare Veränderungsmöglichkeit der Lage ohne Rangieraufwand, so dass Translation und Rotation unabhängig voneinander ablaufen können [76]. Die Radanordnung ist hierbei verantwortlich für den notwendigen algorithmischen Berechnungsaufwand konkreter Fahrmanöver. Diesbezüglich haben allradgetriebene omnidirektionale Fahrzeuge ohne aktives Lenksystem auf ebenen Oberflächen vereinfachende Vorteile gegenüber anderen Fahrwerkskonfigurationen [77-79].
Roboter der Small-Size-League werden lokal angesteuert, aber global überwacht und dirigiert. Somit ist es notwendig, den Bewegungszustand des Roboters einerseits aus der Ansicht des stehenden und andererseits aus der Sicht des mitbewegten Beobachters zu beschreiben und zu transformieren. Das interne Roboter-Koordinatensystem $\left(\vec{E}_X, \vec{E}_Y\right)$ befindet sich auf Bodenhöhe im Schnittpunkt der Radachsen und ist der Achslage entsprechend ausgerichtet (Abb. 3.15). Die konzentrische Rotation des Roboters wird im raumfesten System $\left(\vec{e}_{X0}, \vec{e}_{Y0}\right)$ deckungsgleich wahrgenommen.

Zur Analyse dieser kinematischen Beziehungen sind Matrizen und Spaltenvektoren definiert. Die Lagevektoren $\mathbf{X}$ und $\mathbf{x}$ stellen die momentane Position und Orientierung des Roboters aus der Sicht der Koordinatensysteme $\left(\vec{e}_{X0}, \vec{e}_{Y0}\right)$ und $\left(\vec{E}_X, \vec{E}_Y\right)$ dar. Die Winkelgeschwindigkeiten des Radsatzes werden durch den Vektor $\boldsymbol{\omega}$ beschrieben.

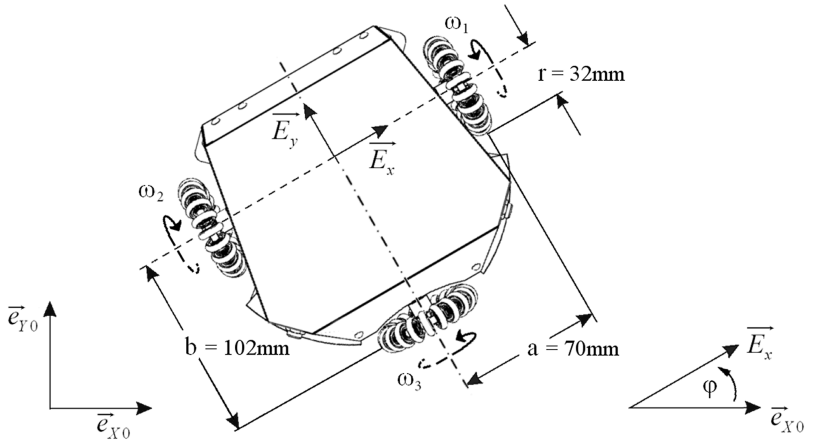


Abb. 3.15 Systembeschreibung zur kinematischen Analyse

$$\mathbf{X} = \begin{pmatrix} |\, X(t) \cdot \vec{e}_{X0} \,| \\ |\, Y(t) \cdot \vec{e}_{Y0} \,| \\ \varphi(t) \end{pmatrix}, \mathbf{x} = \begin{pmatrix} |\, x(t) \cdot \vec{E}_X \,| \\ |\, y(t) \cdot \vec{E}_Y \,| \\ \varphi(t) \end{pmatrix}, \quad \boldsymbol{\omega} = \begin{pmatrix} \omega_1(t) \\ \omega_2(t) \\ \omega_3(t) \end{pmatrix} \tag{3.5}$$

Das kinematische Modell beruht auf der Rollbedingung der Räder $(\mathbf{v_b} = r\boldsymbol{\omega})$, der Annahme konstanter Reibverhältnisse und spielfreier Übersetzungen. Die Transformation via Drehmatrix $\mathbf{R}$ lautet zwischen globaler, raumfester Kameraperspektive und lokalem, körperfestem Bezug:

$$\mathbf{X} = \mathbf{R}\mathbf{x} \Leftrightarrow \mathbf{x} = \mathbf{R}^{-1}\mathbf{X} \tag{3.6}$$

mit

$$\mathbf{R} = \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix}, \mathbf{R}^{-1} = \begin{pmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Angepasst an die erforderliche Regelung stehen Radwinkel-, Translations- bzw. Rotationsgeschwindigkeiten und davon abgeleitete Beschleunigungen sowie deren Formulierung in den Koordinatensystemen im Vordergrund. Position und Orientierung im Inertialsystem resultieren aus der zeitlichen Integration der Bewegungsgrößen.

Unter Vorgabe der Radgrößen sind für Lokalisierung und Navigation die Gleichungen der direkten Kinematik zutreffend. Die konstruktive Radanordnung bedingt durch die Radabstände a, b und den Radradius r (siehe Abb. 3.15) die charakteristische Matrix $\mathbf{C}$, mit der sich in beiden Systemen folgende Geschwindigkeiten und Beschleunigungen ergeben:

$$\dot{\mathbf{X}} = \mathbf{D}r\boldsymbol{\omega} \tag{3.7}$$

$$\ddot{\mathbf{X}} = \dot{\mathbf{D}}r\boldsymbol{\omega} + \mathbf{D}r\dot{\boldsymbol{\omega}} \tag{3.8}$$

$$\dot{\mathbf{x}} = \mathbf{C}r\boldsymbol{\omega} \tag{3.9}$$

$$\ddot{\mathbf{x}} = \mathbf{C}r\dot{\boldsymbol{\omega}} \tag{3.10}$$

wobei $\mathbf{C} = \frac{1}{2a}\begin{pmatrix} -b & b & 2a \\ a & a & 0 \\ 1 & -1 & 0 \end{pmatrix}$ und

$$\mathbf{D} = \mathbf{R}\mathbf{C} = \frac{1}{2a}\begin{pmatrix} (-b\cos\varphi - a\sin\varphi) & (b\cos\varphi - a\sin\varphi) & 2a\cos\varphi \\ (-b\sin\varphi + a\cos\varphi) & (b\sin\varphi + a\cos\varphi) & 2a\sin\varphi \\ 1 & -1 & 0 \end{pmatrix}$$

$$\dot{\mathbf{D}} = \frac{\dot{\varphi}}{2a}\begin{pmatrix} (b\sin\varphi - a\cos\varphi) & (-b\sin\varphi - a\cos\varphi) & -2a\sin\varphi \\ (-b\cos\varphi - a\sin\varphi) & (b\cos\varphi - a\sin\varphi) & 2a\cos\varphi \\ 0 & 0 & 0 \end{pmatrix}$$

Häufiger im Gebrauch sind andererseits die Gleichungen der inversen Kinematik unter Vorgabe des robotereigenen Bewegungszustandes. Sie werden für Richtungsfahrten, Positionierung und geregelte Fahrmanöver herangezogen.

Radgeschwindigkeiten und Radbeschleunigungen stellen sich dabei in den beiden Koordinatensystemen wie folgt dar:

$$r\boldsymbol{\omega} = \mathbf{D}^{-1}\dot{\mathbf{X}}\,,\ \mathbf{D}^{-1} = \mathbf{C}^{-1}\mathbf{R}^{-1} = \begin{pmatrix} -\sin\varphi & \cos\varphi & a \\ -\sin\varphi & \cos\varphi & -a \\ \cos\varphi & \sin\varphi & b \end{pmatrix} \qquad (3.11)$$

$$r\dot{\boldsymbol{\omega}} = \dot{\mathbf{D}}^{-1}\dot{\mathbf{X}} + \mathbf{D}^{-1}\ddot{\mathbf{X}}\,,\quad \dot{\mathbf{D}}^{-1} = \dot{\varphi}\begin{pmatrix} -\cos\varphi & -\sin\varphi & 0 \\ -\cos\varphi & -\sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \end{pmatrix} \qquad (3.12)$$

$$r\boldsymbol{\omega} = \mathbf{C}^{-1}\dot{\mathbf{x}}\ , \qquad \mathbf{C}^{-1} = \begin{pmatrix} 0 & 1 & a \\ 0 & 1 & -a \\ 1 & 0 & b \end{pmatrix} \qquad (3.13)$$

$$r\dot{\boldsymbol{\omega}} = \mathbf{C}^{-1}\ddot{\mathbf{x}} \qquad (3.14)$$

3.3.2 Trajektorien

Als Konsequenz des vorherigen Abschnittes kann man anhand einfacher geometrischer Bahnkurven das mechanische Fahrverhalten des Roboters demonstrieren. Die obigen Gleichungen der direkten Kinematik gestatten zwei Lösungswege. Folgen die Radgeschwindigkeiten zeitabhängigen Funktionen, empfiehlt sich der Weg über den Vektor $\dot{\mathbf{X}}$ der Gleichung (3.7). Der Lösungsweg über den Vektor $\dot{\mathbf{x}}$ der Gleichung (3.9) erfordert weniger Aufwand, wenn zur zweckmäßigen Vereinfachung konstante Raddrehzahlen vorausgesetzt werden. Ein konstanter Betrag der Geschwindigkeit des Roboters bedingt allerdings keine konstante vektorielle Geschwindigkeit, sofern er sich nicht geradlinig bewegt. Um die Bewegungsspur des Roboters sichtbar zu machen, erfolgt mit der aktiven Drehmatrix $\mathbf{R}$ die Transformation in das raumfeste System sowie die anschließende Integration. Mit den Anfangsbe-

dingungen $X(t_0)=0$, $Y(t_0)=0$ und $t_0=0$ folgt der Roboter im Weltsystem dem Bahnverlauf:

$$\mathbf{X}=\frac{1}{\dot{\varphi}}\begin{pmatrix}\sin(\dot{\varphi}\cdot t) & \cos(\dot{\varphi}\cdot t) & 0\\ -\cos(\dot{\varphi}\cdot t) & \sin(\dot{\varphi}\cdot t) & 0\\ 0 & 0 & \dot{\varphi}\cdot t\end{pmatrix}\dot{\mathbf{x}}+\frac{1}{\dot{\varphi}}\begin{pmatrix}-\dot{y}\\ \dot{x}\\ 0\end{pmatrix}, \text{ mit} \qquad (3.15)$$

$$\dot{\mathbf{x}}=(\dot{x},\dot{y},\dot{\varphi})^T=konst.$$

Dies gilt nur für kreisförmige Bahnverläufe und ist für $\dot{\varphi}=0$, d.h. $\omega_1=\omega_2$, nicht definiert. Trotzdem kann dieser Grenzfall für Translationsbetrachtungen in Abb. 3.16 beliebig angenähert werden. Die Grafik beschreibt fünf Parametervariationen der drei Radgeschwindigkeiten und ihre Auswirkungen auf den Bahnverlauf des Roboters. Ungeachtet der grundlegenden Fähigkeit zur Translation, Rotation und Bogenfahrt nach Art des Differentialantriebes zeigen die Kurven A, B und E auch die erweiterte omnidirektionale Manövrierfähigkeit.

Zeitintervall: t [0sec , 20sec]
Radgeschwindigkeiten: [m/sec]

	$r\omega_1$	$r\omega_2$	$r\omega_3$
Ⓐ	0,00001	0	-0,025
Ⓑ	0,0250001	0,025	-0,025
Ⓒ	-0,025	0,025	-0,025
Ⓓ	-0,025	0,025	0
Ⓔ	-0,025	0,025	0,07

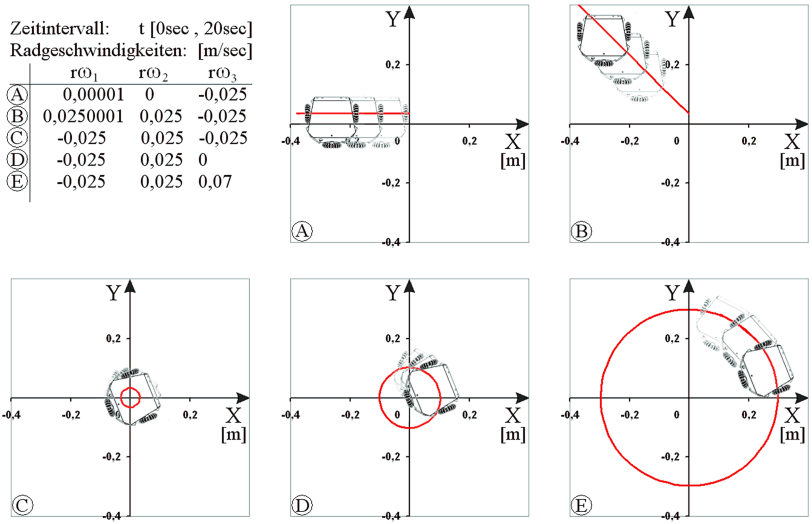


Abb. 3.16 Bahnverläufe konstanter Raddrehzahlen

3.4 Dynamik

Sofern keine Einschränkung durch Schlupf auftritt, gelangt der Roboter vom aktuellen Standpunkt auf einer beliebig geformten Trajektorie zum anvisierten Zielpunkt. Weil durchdrehende Räder zu unkontrollierten Bewegungen führen, setzt dies jedoch immer die Einhaltung der Rollbedingung an den Rädern voraus. Im Bereich maximaler Geschwindigkeiten und Beschleunigungen genügt deswegen eine radbasierte Lageregelung des Roboters LUKAS nicht den Anforderungen. Massenträgheit und Kraftübertragung der gesamten Einheit erhalten im Bewegungsverhalten eine verstärkte Bedeutung und können nur mit der Betrachtung des dynamischen Modells erfasst werden. Zur Anpassung eines optimalen Reglerbetriebes beschreibt das Modell den Zusammenhang zwischen den elektrischen Eingangspannungen der Motoren und den daraus resultierenden Rad- bzw. Robotergeschwindigkeiten.

3.4.1 Modellierung der Antriebsstränge

Bevor sich die elektrische Akkumulatorladung in mechanische Roboterbewegung umsetzt, durchläuft sie mehrfache Energiewandlung. Die Modellierung eines Antriebsstranges hat das Ziel, die Verknüpfung der motorseitigen Eingangsspannung mit der antreibenden Radkraft herzuleiten. Neben dem Ersatzschaltbild des Ankers im permanenterregten Gleichstrommotor zeigt Abb. 3.17 symbolisch die Übertragungskette zur angekoppelten Radlast, wie sie im Roboter Anwendung findet. Das Zahnradgetriebe überträgt mit der Getriebeübersetzung ü. Die Energiewandlung zwischen den beiden Teilsystemen beruht auf dem elektrodynamischen Gesetz, demzufolge die elektrisch stromführenden Leiterbahnen im Magnetfeld mechanische Lorentzkraft bewirken können. Beim Gleichstrommotor basiert dies auf Differentialgleichungen, die sich für den Ankerstromkreis aus dem 2. Kirchhoffschen Gesetz und für alle rotierenden Teile aus dem Drehimpulserhaltungssatz der Mechanik ableiten.

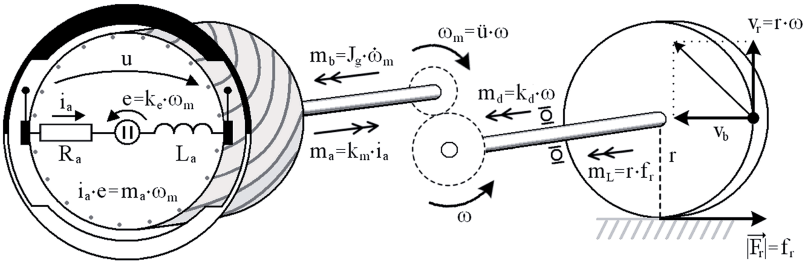


Abb. 3.17 Elektromechanisches Modell eines Antriebsstranges

In einem konstanten Erregerfeld gilt:

$$L_a \frac{di_a}{dt} + R_a i_a = u - k_e \omega_m \qquad (3.16)$$

$$J_g \frac{d\omega_m}{dt} = k_m i_a - k_d \omega_m - \frac{r}{\ddot{u}} f_r \qquad (3.17)$$

Die Dynamik des elektrischen Kreises soll nicht weiter betrachtet werden, weil die Zeitkonstante des Motors im Gegensatz zur mechanischen Zeitkonstante sehr klein ist. Der Ankerstrom i_a ergibt sich zu:

$$i_a = \frac{1}{R_a}(u - k_e \omega_m) \text{, unter der Annahme } \frac{di_a}{dt} = 0 \qquad (3.18)$$

J_g entspricht, unter der Voraussetzung eines spielfreien Getriebes der Übersetzung $\ddot{u}$, der Summe von Rotorträgheitsmoment J_m und dem Trägheitsmoment der Masse m_r eines zylindrischen Radkörpers.

$$J_g = J_m + \frac{1}{\ddot{u}^2}\left(\frac{1}{2} m_r r^2\right) \qquad (3.19)$$

Für die Darstellung des prinzipiellen Vorgehens genügt an dieser Stelle die Annahme identischer Motoren. Die Sonderstellung der Querachse im Roboter LUKAS wird erst im erweiterten Modell berücksichtigt (Abschnitt 3.4.3).
Die Gleichungen (3.17), (3.18) und (3.19) ergänzen sich somit zum jeweiligen dynamischen Modell der drei Antriebsstränge:

$$J_g \dot{\omega}_i + k_d \omega_i + \frac{r f_{ri}}{\ddot{u}^2} = \frac{k_m}{R_a \ddot{u}} u_i - \frac{k_m k_e}{R_a} \omega_i \qquad (3.20)$$

und $\omega_{mi} = \ddot{u}\omega_i$, $i \in \{1,2,3\}$

Mit den substituierten Koeffizienten

$$\mathrm{K_U} = \frac{k_m \ddot{u}}{R_a r}, \mathrm{K_J} = \frac{J_g \ddot{u}^2}{r^2}, \mathrm{K_\Omega} = \frac{\ddot{u}^2}{r^2}\left(\frac{k_m k_e}{R_a} + k_d\right)$$

sowie unter Verwendung der Vektoren

$$\mathbf{f_r} = \begin{pmatrix} f_{r1} & f_{r2} & f_{r3} \end{pmatrix}^T \text{ und } \mathbf{u} = \begin{pmatrix} u_1 & u_2 & u_3 \end{pmatrix}^T$$

entsprechen die drei Bewegungsdifferentialgleichungen dem Ausdruck:

$$\mathbf{f_r} = \mathrm{K_U}\mathbf{u} - \mathrm{K_\Omega} r\boldsymbol{\omega} - \mathrm{K_J} r\dot{\boldsymbol{\omega}} \qquad (3.21)$$

Die am Rad wirkende Antriebskraft wird hier durch den radeigenen Bewegungszustand und die Motorspannung beschrieben, wobei für das schlupffrei abrollende Rad die Voraussetzung $v_r = v_b = r \cdot \omega$ gilt.

3.4.2 Eingeschränktes Basismodell

Ausgehend von der Annahme idealer Rollbedingungen kann nach dem Impulssatz eine Vorschrift entwickelt werden, die der Berechnung des dynamischen Bewegungszustandes dient [80]. In Anwendung des bekannten Designs nach Abb. 3.15 soll nachfolgend für eine räumliche Betrachtung der Dynamik die Anordnung der Radkräfte und die Lage des Schwerpunktes, wie in Abb. 3.18 dargestellt, eingeführt werden. Der Schwerpunkt $S(0,0,r)$ liegt beim eingeschränkten Basismodell lotrecht über dem lokalen Koordinatenursprung in der Höhe der Radachse.

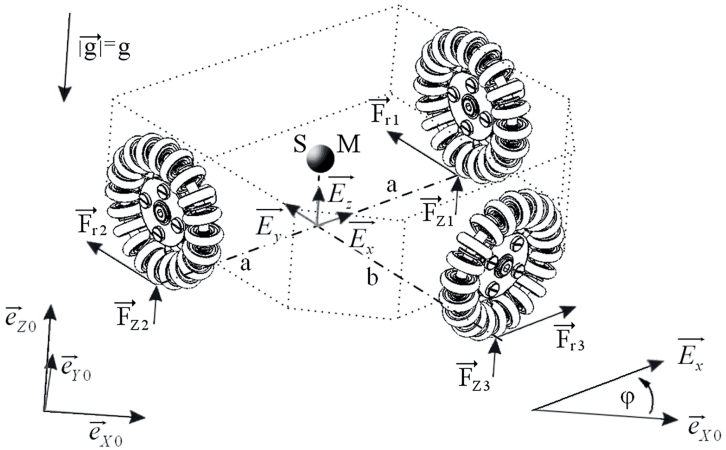


Abb. 3.18 Radkräfte und Lage des Schwerpunktes

Für die Betrachtung der Impulserhaltung werden im raumfesten Bezugssystem der am körperfesten Schwerpunkt S angreifenden Kraftbetrag F_S sowie der wirkende Drehmomentbetrag M_{SZ} herangezogen. Sie resultieren aus Sum-

mation bzw. Hebelwirkung der antreibenden Kraftvektoren $\overrightarrow{F_{r1}}, \overrightarrow{F_{r2}}, \overrightarrow{F_{r3}}$ und können in Komponentenschreibweise auch als Ausdruck deren Beträge f_{r1}, f_{r2}, f_{r3} dargestellt werden. Die Richtungen der positiven Betragsgrößen dieser Antriebskraftvektoren gehen aus Abb. 3.18 hervor. Treibt man ein Rad entgegengesetzt an, muss mit dem negativen Betrag gerechnet werden. Für die Masse des Roboters steht das Symbol M :

$$F_S = \left| \begin{pmatrix} M\ddot{x} \\ M\ddot{y} \end{pmatrix} \right| = \sqrt{(f_{r3})^2 + (f_{r1} + f_{r2})^2} \qquad (3.22)$$

$$M_{SZ} = J\ddot{\varphi} = +af_{r1} - af_{r2} + bf_{r3} \qquad (3.23)$$

Der am Schwerpunkt S angreifende Drehmomentbetrag M_{SZ} bleibt aufgrund deckungsgleicher z-Achslage in der Transformation vom lokalen ins globale Koordinatensystem identisch. Der Roboter wird in der vereinfachten Annahme als homogener Vollzylinder betrachtet, wobei sein Massenträgheitsmoment um die vertikale Rotationsachse mit $J = \frac{1}{2}Ma^2$ bestimmt ist. In Anwendung der Impulserhaltung führt dies zur Darstellung der Roboterbeschleunigung im stationären globalen Koordinatensystem:

$$\ddot{\mathbf{X}} = \mathbf{R} \begin{pmatrix} \frac{1}{M} f_{r3} \\ \frac{1}{M}(f_{r1} + f_{r2}) \\ \frac{1}{J} M_{SZ} \end{pmatrix} = \mathbf{C_W f_r} \qquad (3.24)$$

mit $\mathbf{C_W} = \frac{1}{M} \begin{pmatrix} -\sin\varphi & -\sin\varphi & \cos\varphi \\ \cos\varphi & \cos\varphi & \sin\varphi \\ 2a^{-1} & -2a^{-1} & 2ba^{-2} \end{pmatrix}$

Dem Vorbild der direkten Kinematik auf globaler Ebene nach Gleichung (3.8) folgt die Abhängigkeit der Radkräfte zu den Bewegungsgrößen der Räder damit:

$$\mathbf{C_W f_r} = \dot{\mathbf{D}} r\boldsymbol{\omega} + \mathbf{D} r\dot{\boldsymbol{\omega}} \qquad (3.25)$$

Ergänzt man diesen kinetischen Zusammenhang (3.25) mit der Gleichung (3.21) aus der Vorbetrachtung der Antriebstränge, so erwächst ein abstrahiertes dynamisches Bewegungsmodell der Radgeschwindigkeiten, mit dem regelbaren Motorspannungsvektor $\mathbf{u}$ als Eingangsgröße:

$$\mathbf{C_W K_U u} = (\dot{\mathbf{D}} + \mathbf{C_W K_\Omega}) r\boldsymbol{\omega} + (\mathbf{D} + \mathbf{C_W K_J}) r\dot{\boldsymbol{\omega}} \qquad (3.26)$$

Mittels der inversen kinematischen Herleitungen aus Abschnitt 3.3.1 lässt sich diese Modellgleichung auch zur Berechnung der Robotergeschwindigkeit im lokalen und globalen System in Beziehung setzen.

$$\mathbf{C_W K_U u} = (\dot{\mathbf{D}} + \mathbf{C_W K_\Omega})\mathbf{C^{-1}}\dot{\mathbf{x}} + (\mathbf{D} + \mathbf{C_W K_J})\mathbf{C^{-1}}\ddot{\mathbf{x}} \qquad (3.27)$$

$$\mathbf{C_W K_U u} = ((\dot{\mathbf{D}} + \mathbf{C_W K_\Omega})\mathbf{D^{-1}} + (\mathbf{D} + \mathbf{C_w K_J})\dot{\mathbf{D}}^{-1})\dot{\mathbf{X}} + (\mathbf{D} + \mathbf{C_W K_J})\mathbf{D^{-1}}\ddot{\mathbf{X}} \qquad (3.28)$$

Unter wiederholtem Hinweis auf die idealisierten Voraussetzungen lassen sich diese Ausführungen im Strukturbild der Abb. 3.19 veranschaulichen. Das Basismodell eignet sich gut für prinzipielle Simulationen, ist jedoch für den Entwurf und die Dimensionierung der internen Reglerstruktur noch zu ungenau. Die Erfassung weiterer physikalischer Gesetzmäßigkeiten ist notwendig, um für den Roboters LUKAS eine verbesserte Modellierung zu erhalten.

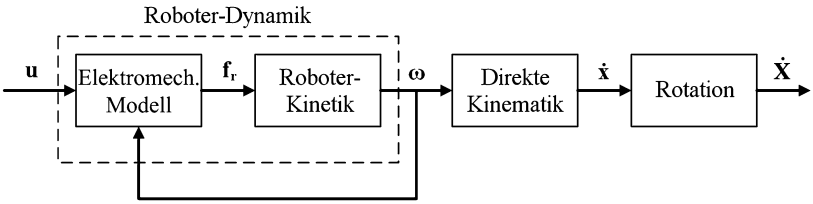


Abb. 3.19 Strukturbild des eingeschränkten dynamischen Basismodells des Roboters

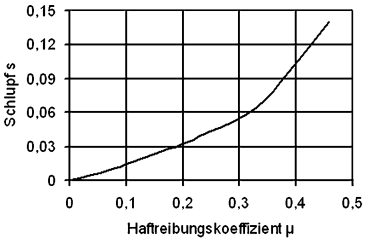
3.4.3 Erweitertes Simulationsmodell

Unter Nutzung der vom Basismodell bereitgestellten Struktur und der eingeführten Symbolik wurde das dynamische Modell mit realistischer Annäherung verbessert [81]. Aufgrund der Berücksichtigung der realen Schwerpunktlage, spezifischer Antriebe sowie des Reibungs- und Schlupfverhaltens der Räder, wie es Abb. 3.20 darlegt, können die Grenzübergänge in undefinierte Bewegungszustände bestimmt und mit der angepassten Kontrolle des Fahrtreglers stabilisiert werden. Die modifizierte Struktur des erweiterten Simulationsmodells ist in Abb. 3.21 ersichtlich. Die drei Antriebsstränge des Roboters weisen differenzierte Motor- und Getriebeparameter auf, weshalb zu den Restriktionen des Basismodells, die ausgeweitet wurden, auch die Überarbeitung des elektromechanischen Modells gehört. Die Anwendung erfordert radspezifische Koeffizienten und die Antriebskraftminderung durch Coulomb'sche Reibkräfte.

$$\mathbf{f_r} = \mathbf{E}\begin{pmatrix} K_{U1} \\ K_{U2} \\ K_{U3} \end{pmatrix}\mathbf{u} - \mathbf{E}\begin{pmatrix} K_{\Omega 1} \\ K_{\Omega 2} \\ K_{\Omega 3} \end{pmatrix}\mathbf{v_r} - \mathbf{E}\begin{pmatrix} K_{J1} \\ K_{J2} \\ K_{J3} \end{pmatrix}\dot{\mathbf{v}}_\mathbf{r} - \mathbf{f_C} \qquad (3.29)$$

$$\mathbf{v_r} = r\boldsymbol{\omega}\,,\ \mathbf{f_C} = \begin{pmatrix} \mathrm{sgn}(\omega_1) f_{C1} \\ \mathrm{sgn}(\omega_2) f_{C2} \\ \mathrm{sgn}(\omega_3) f_{C3} \end{pmatrix}$$

Der reale Abrollvorgang ist durch die Verformung der Gummiringe mit einem effektiven Vortriebsverlust verbunden. Die Struktur (siehe Abb. 3.21) zeichnet sich deshalb besonders durch den Einschub von Schlupfanalyse und Schwerpunktverlagerung in die Kopplung von elektromechanischem Antriebsmodell und Roboterkinetik aus.



Messergebnisse praktischer Versuche: Prototyp LUKAS

- lokale Schwerpunktlage [mm]: $S(-0{,}45 \quad -27{,}89 \quad 33{,}58)$
- Haftreibungsgrenze: $\mu_{max} = 0{,}9571$
- Coulomb'sche Reibkraft [N]: $\mathbf{f}_C = \begin{pmatrix} 0{,}418 \\ 0{,}418 \\ 0{,}982 \end{pmatrix}$

Abb. 3.20 Messergebnisse praktischer Versuche und Schlupfkurve vom Prototyp

Durch Überwachung von Umfangs- und Bodengeschwindigkeiten der Räder und der Berücksichtigung der auftretenden Schwerpunktverlagerung kann die wirksame Antriebskraft bei auftretendem Schlupf ermittelt werden.

Als Bodengeschwindigkeit v_b wird die Projektion der Geschwindigkeit des Radmittelpunktes auf die Kraftübertragungsrichtung des Rades bezeichnet. Wie aus Abb. 3.17 am Antriebsrad erkennbar ist, ergibt sich aus der Eigenrotation die Radumfangsgeschwindigkeit v_r. Über beide Geschwindigkeitsparameter eines Rades errechnet sich richtungsabhängig der jeweilige Radschlupf s, d.h. entweder für auftretenden Antriebs- oder Bremsschlupf.

$$s = \frac{v_r - v_b}{\max(|v_r|, |v_b|)} \qquad (3.30)$$

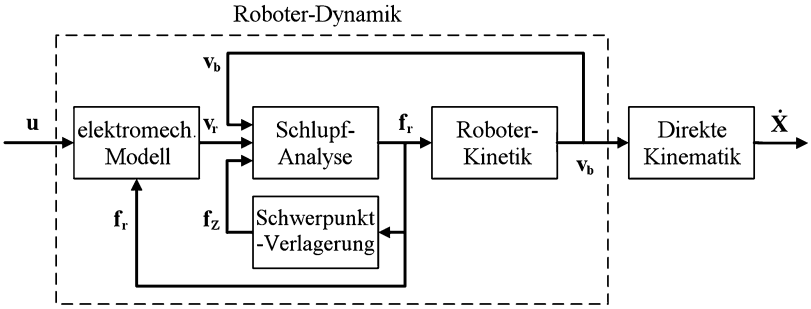


Abb. 3.21 Strukturbild des erweiterten dynamischen Simulationsmodells des Roboters

Der Betrag der Radantriebskraft f_r, der kinetisch für den realen Vortrieb des Roboters verantwortlich ist, entsteht im Gegensatz zum Basismodell als Produkt des dem Schlupfwert zugeordneten Radkraftverhältnisses $\mu(s)$ mit der Aufstandskraftbetrag f_Z (siehe Abb. 3.18).

$$f_r = \mu(s) \cdot f_Z \qquad (3.31)$$

Der modellierte Radschlupf wird nach der in Abb. 3.20 gezeigten Radschlupfkurve in das zugehörige Radkraftverhältnis $\mu(s)$ überführt und bei Überschreitung der Haftreibungsgrenze $\mu_{\max}$ durch Kontrolle der Antriebskraft beschränkt. Zunehmende Antriebskräfte sind deshalb nur mit steigendem Radschlupf umzusetzen, sofern die Aufstandskräfte dabei nicht zu stark abnehmen. Dass die Aufstandskräfte bei der Beschleunigung des Roboters nicht konstant bleiben, ist begründet in dessen realer Schwerpunktverlagerung. Mit einem starren Chassis führt das um den Schwerpunkt $S(s_X, s_Y, s_Z)$ ausgewertete Drehmomentengleichgewicht zu:

$$\mathbf{f_Z} = \begin{pmatrix} f_{Z1} \\ f_{Z2} \\ f_{Z3} \end{pmatrix} = \mathbf{F}\mathbf{f_r} + \mathbf{G} \qquad (3.32)$$

$$\mathbf{F} = \frac{s_Z}{2ab}\begin{pmatrix} -a & -a & -b \\ -a & -a & b \\ 2a & 2a & 0 \end{pmatrix}, \quad \mathbf{G} = \frac{Mg}{2ab}\begin{pmatrix} ab + bs_X + as_Y \\ ab - bs_X + as_Y \\ -2as_Y \end{pmatrix}$$

Auch der Schwerpunkt des Roboters ist in seinen lokalen Ursprungskoordinaten anzugeben und allgemein mit $S(s_X, s_Y, s_Z)$ zu bezeichnen (siehe Abb. 3.20).

Diese verallgemeinerte Angabe bedingt die Änderungen des wirkenden Drehmoments M_{SZ} zu $\tilde{M}_{SZ}$:

$$\tilde{M}_{SZ} = (a - s_X) f_{r1} + (-a - s_X) f_{r2} + (b + s_Y) f_{r3} \qquad (3.33)$$

und damit die Änderung der Impulsgleichungen:

$$\ddot{\mathbf{X}} = \mathbf{R} \begin{pmatrix} \frac{1}{M} F_{SX} \\ \frac{1}{M} F_{SY} \\ \frac{1}{J} \tilde{M}_{SZ} \end{pmatrix} = \tilde{\mathbf{C}}_{\mathbf{W}} \mathbf{f}_{\mathbf{r}} \qquad (3.34)$$

$$\tilde{\mathbf{C}}_{\mathbf{W}} = \frac{1}{M} \begin{pmatrix} -\sin\varphi & -\sin\varphi & \cos\varphi \\ \cos\varphi & \cos\varphi & \sin\varphi \\ 2a^{-2}(a - s_X) & -2a^{-2}(a + s_X) & 2a^{-2}(b + s_Y) \end{pmatrix}$$

Um die Differentialgleichungen des dynamischen Gesamtmodells zu erhalten, setzt man zuerst den Aufstandskraftvektor nach Gleichung (3.32) in die Gleichung (3.31) der Schlupfanalyse.

$$\mathbf{f}_{\mathbf{r}} = \boldsymbol{\mu}(\mathbf{F}\mathbf{f}_{\mathbf{r}} + \mathbf{G}) \;\Leftrightarrow\; \mathbf{f}_{\mathbf{r}}(\mathbf{E} - \boldsymbol{\mu}\mathbf{F}) = \boldsymbol{\mu}\mathbf{G} \text{ mit } \boldsymbol{\mu} = \begin{pmatrix} \mu(s_1) \\ \mu(s_2) \\ \mu(s_3) \end{pmatrix} \qquad (3.35)$$

Diese Gleichung wird im Folgenden einerseits mit der Roboterkinetik nach dem Vorbild von Gleichung (3.25) zur Differentialgleichung der Bodengeschwindigkeit v_b und andererseits mit dem elektromechanischen Modell nach Gleichung (3.29) zur Differentialgleichung der Radumfangsgeschwindigkeiten v_r verbunden.

$$\tilde{\mathbf{C}}_{\mathbf{W}} \boldsymbol{\mu}\mathbf{G} = (\dot{\mathbf{D}}\mathbf{v}_{\mathbf{b}} + \mathbf{D}\dot{\mathbf{v}}_{\mathbf{b}})(\mathbf{E} - \boldsymbol{\mu}\mathbf{F}) \qquad (3.36)$$

$$\left(\mathbf{K_U u} - \mathbf{K_\Omega v_r} - \mathbf{K_J}\dot{\mathbf{v}}_\mathbf{r} - \mathbf{f_C}\right)\left(\mathbf{E} - \boldsymbol{\mu}\mathbf{F}\right) = \boldsymbol{\mu}\mathbf{G}\text{, mit} \qquad (3.37)$$

$$\mathbf{K_U} = \begin{pmatrix} K_{U1} & 0 & 0 \\ 0 & K_{U2} & 0 \\ 0 & 0 & K_{U3} \end{pmatrix},\ \mathbf{K_\Omega} = \begin{pmatrix} K_{\Omega 1} & 0 & 0 \\ 0 & K_{\Omega 2} & 0 \\ 0 & 0 & K_{\Omega 3} \end{pmatrix},\ \mathbf{K_J} = \begin{pmatrix} K_{J1} & 0 & 0 \\ 0 & K_{J2} & 0 \\ 0 & 0 & K_{J3} \end{pmatrix}$$

An der Bewegungsdifferentialgleichung ist nach dem Einsetzen der Gleichung (3.37) in die Gleichung (3.36) ersichtlich, dass sich das erweiterte Modell für den Fall schlupffreier Rollbewegung mit $\mathbf{v_b} = \mathbf{v_r} = r\boldsymbol{\omega}$ analog dem eingeschränkten Basismodell nach Gleichung (3.26) verhält:

$$\tilde{\mathbf{C}}_\mathbf{W}\mathbf{K_U u} = (\dot{\mathbf{D}} + \tilde{\mathbf{C}}_\mathbf{W}\mathbf{K_\Omega})r\boldsymbol{\omega} + (\mathbf{D} + \tilde{\mathbf{C}}_\mathbf{W}\mathbf{K_J})r\dot{\boldsymbol{\omega}} + \tilde{\mathbf{C}}_\mathbf{W}\mathbf{f_C} \qquad (3.38)$$

Damit ist auch der lineare Zusammenhang gegeben, der im Fahrtregler zur Sollvorgabe der Motorspannungen $\mathbf{u}$ für konstante Endgeschwindigkeiten $\boldsymbol{\omega}_f$ der Räder angewendet werden kann:

$$\tilde{\mathbf{C}}_\mathbf{W}\mathbf{K_U u} = (\dot{\mathbf{D}} + \tilde{\mathbf{C}}_\mathbf{W}\mathbf{K_\Omega})r\boldsymbol{\omega}_f + \tilde{\mathbf{C}}_\mathbf{W}\mathbf{f_C}\text{, mit } \dot{\boldsymbol{\omega}}_f = 0 \qquad (3.39)$$

Es ist zu beachten, dass die Verschiebung des Schwerpunktes auch in den Gleichungen der Kinematik berücksichtigt wird, weil das lokale Koordinatensystem für eine fehlerfreie Berechnung im Drehzentrum stehen muss. Die Korrektur von $\mathbf{C}$ nach $\tilde{\mathbf{C}}$ bedingt auch Änderungen der mit ihr verknüpften Matrizen.

$$\tilde{\mathbf{C}} = \frac{1}{2(a^2 - s_X{}^2)}\begin{pmatrix} -(b+s_Y)(a+s_X) & (b+s_Y)(a-s_X) & 2(a^2 - s_X{}^2) \\ (a^2 - s_X{}^2) & (a^2 - s_X{}^2) & 0 \\ (a+s_X) & -(a-s_X) & 0 \end{pmatrix} \qquad (3.40)$$

Eingebettet ist dieses Modell in eine MATLAB®/SIMULINK®-Struktur und steht zur zeitsparenden virtuellen Optimierung von Regleralgorithmen bzw. Reglerparametern bereit [81].

3.5 Steuerungsstruktur RoboCon

Den Roboter LUKAS verwaltet das spezifizierte Controllerprogramm RoboCon auf lokaler Ebene. Es hat die Aufgabe, dem PC der Spielsteuerung die Funktionalität des Roboters durch die freie Wahl des Assistenzgrades im Spektrum von ungeregelter Motorsteuerung bis zum automatischen ausgeführten Fahrmanöver zu erschließen. Aus den Kombinationen der Kontrollebenen mit der dualen Reglerstruktur eröffnet es die Möglichkeit direkter Positionierung und erleichterter adaptiver Bahnsteuerung (Abb. 3.22).

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11
Ziel		Befehl		Datenwerte						CRC	
Quelle			Parameter								

Tab. 3.1 Datenstruktur eines Sendepakets

3.5.1 Kommunikation und Befehlsverarbeitung

Die serielle, asynchrone Kommunikation findet gegenwärtig über ein Bluetooth- oder 433 MHz-Funkmodul statt. Befehlsdaten werden dem Mikrocontroller ähnlich der SNAP-Codierung (Scaleable Node Address Protocol)[15] zur Verfügung gestellt. Controllerintern besteht kein Unterschied gegenüber der seriellen kabelgebundenen RS232-Kommunikation zum Computer der Mastersteuerung. Für kurze Sendepakete nach Tab. 3.1 wurde ein schlichtes Netzwerkprotokoll angestrebt. Ein minimaler Datenstrom vergrößert den Spielraum zur Steuerung eines Roboterteams über die begrenzte Bandbreite des Kanals. Vorteilhaft ist im Roboter dadurch auch die zwischenzeitliche Pufferung des empfangenen Befehlsstromes bis zur Abarbeitung. Während die Steuerung des Bewegungszustandes eine permanente Übertragung erfordert, genügt zur Umsetzung lagebasierter Anweisungsfolgen eine vereinzelte,

[15] www.hth.com

temporäre Abfolge von Befehlen. Die definierte Befehlspriorität entscheidet über die zeitliche Eingliederung der im Befehlspuffer gelisteten Anordnungen (Abb. 3.22).

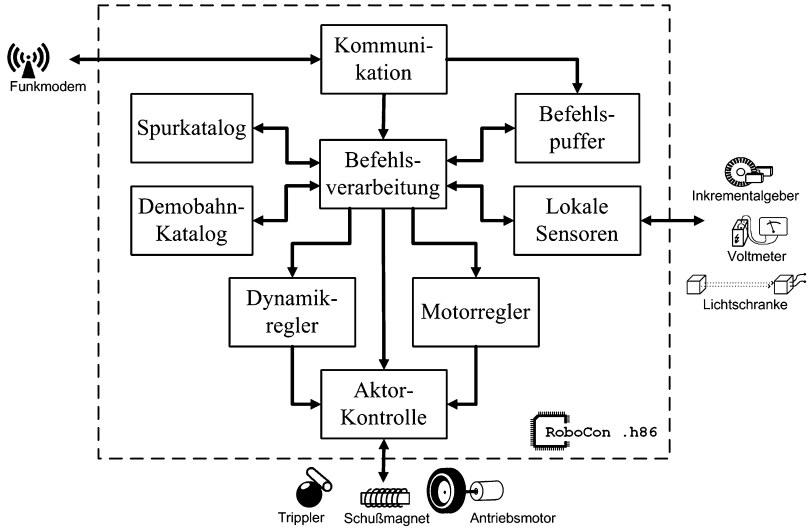


Abb. 3.22 Struktur der Controllersoftware

3.5.2 Kontrollebenen

Als hierarchische Rangfolge stehen die verschiedenen Ebenen der Antriebsüberwachung zur Verfügung (Abb. 3.23). Die steigende Eigenkontrolle, bzw. Abstraktion der Steuerungsvorgaben entlastet die zentrale Spielsteuerung von der permanenten Ansteuerung der Roboter. Höhere Ebenen beruhen im Allgemeinen auf der Funktionalität der ihnen untergeordneten Schicht, wobei die duale Regelungsstruktur für spezifische Variationen genutzt wird. Die Spannungssteuerung ist die niedrigste kontrollierbare Ebene, in welcher die zu generierende Motorspannung der Antriebe direkt aus dem Stapelbefehl entnommen und ungeregelt als PWM-Signal am Leistungssteller umgesetzt wird.

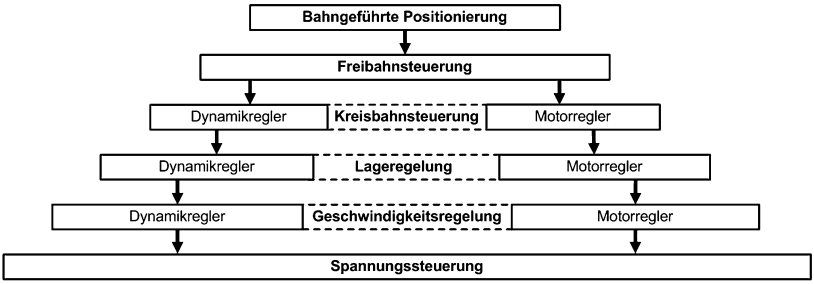


Abb. 3.23 Kontrollebenen der Controllersoftware

Darauf aufbauend bildet die Geschwindigkeitsregelung die niedrigste Ebene mit Regelungscharakter. Die beiden Regelungsarten erlauben zwei Varianten der Parametervorgabe. Im Modus des Motorreglers werden die Geschwindigkeiten der einzelnen Räder separat kontrolliert, wobei ein aktiver Drehmomentausgleich am Roboter eine asymmetrische Korrektur der Sollvorgaben erfordert. Der Dynamikregler überwacht die Geschwindigkeitskomponenten des Roboters im lokalen Koordinatensystem. Die Entstehung einer ungewollten Drehbewegung wird hier durch die Entkopplung der Ansteuerung verhindert. Ein störendes Drehmoment am Roboter kompensiert die Berücksichtigung entsprechender Gegenkräfte. Beide Regler unterstützen auch die übergeordneten Ebenen der Lageregelung und Kreisbahnsteuerung. Die Lageregelung hat die Aufgabe, den Roboter aus dem Blickwickel seiner lokalen Koordinaten in eine neue Position zu fahren. Dabei sind in den Befehlswerten die Sollwege für Frontal- sowie Querfahrt und der Sollwinkel relativ zur aktuellen Position hinterlegt. Als Kreisbahnsteuerung ist eine der Lageregelung überlagerte Ortsteuerung bezeichnet, deren aus Schwingungen generierte Bahnen (Lissajous-Figuren) automatisch abgefahren werden. Für genau definierte omnidirektionale Sollbahnen in frontaler Ausrichtung sind auf diese Weise nur wenige Befehlsparameter zu übertragen. Die obereren Ebenen der Freibahnsteuerung und der bahngeführten Positionierung greifen auf das Verkettungsschema von Wegsegmenten zurück. Strecken-, Bogen- und Ellipsenelemente und deren Kombinationen können nach individuellen Maßvorgaben und Ausrichtungen vom Roboter selbständig automatisch abgefahren werden.

3.5.3 Motorregler

Die Koordinierung des Roboters anhand der kinematischen Beziehungen wird durch die Lage- und Geschwindigkeitsregelung einzelner Antriebsmotoren angeboten. Die dynamische Rückwirkung des Roboters muss durch den abgestimmten Motorregler kompensiert werden. Die direkte Radkontrolle kann besonders für die Kreisbahnsteuerung in frontaler Orientierung genutzt werden, da bedingt durch die orthogonale Achslage keine kartesischen Transformationen notwendig sind. Die Harmonisierung der einzelnen Lageregler erlaubt die Anwendung separater Translation und Rotation. Die Positionierung über gemischte Bahnbewegungen erfordert jedoch die Zerlegung und Einzelabarbeitung von Streckensegmenten.
Die Struktur dieser Regelung entspricht einer zweistufigen Kaskade (Abb. 3.24). Dem Lageregelkreis ist eine Geschwindigkeitsregelung unterlagert. Durch Moduswahl können die beiden Regler von außen nach innen deaktiviert werden und ermöglichen so ein externes Vorgeben der Führungsgröße für die höchste verbleibende Schicht. Eine Eigenschaft dieser Struktur ist damit der geschichtete Aufbau, auf welchen durch Befehle der Spielsteuerung separat zugegriffen werden kann. Die charakteristische Eigenschaft beruht dennoch auf der Radbezogenheit. Alle Parameter, Messgrößen, Soll- und Grenzwerte sind in einer Struktur gespeichert, die einem Rad zugeordnet ist. Ein Feld dieser drei Strukturen speichert damit sämtliche Informationen über den Zustand des Roboters. Direkte Positionierungsaufgaben sind nach der Methode von Ziegler und Nichols [82] mit vergleichsweise kleinem algorithmischen Aufwand bezüglich der Regelstrecke zu bewerkstelligen.
Empfangene Befehle werden in radbezogene Sollwerte zerlegt. Beispielsweise transformiert die Befehlsverarbeitung die roboterbezogenen Lagevorgaben in die konkreten Wegvorgaben eines Rades. Durch den Vergleich mit dem absoluten Messwert des jeweiligen Motorinkrementalgebers ergibt sich proportional zur Regelabweichung die notwendige Radgeschwindigkeit. Die Geschwindigkeitsregelung vergleicht die Soll- und Istwerte des jeweiligen Rades, woraus sich über ein PI-Glied mit fester Begrenzung die Pulsbreite berechnet,

mit der ein Rad angesteuert wird. Schon bei der Optimierung des gesamten Reglers zeichneten sich für den Roboter LUKAS einige Sonderfälle durch dynamisch verursachte Effekte ab (siehe folgender Abschnitt 3.5.4), die in ihrer Wirkung mit diesem Ansatz nur indirekt kompensiert werden können.

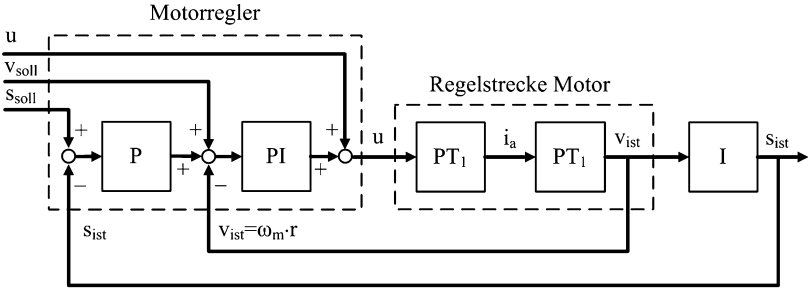
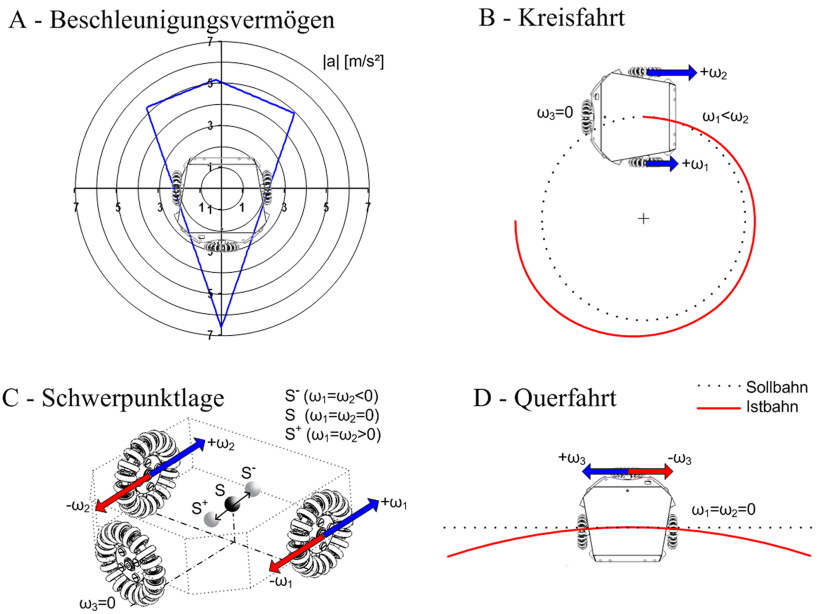


Abb. 3.24 Kaskadenstruktur der radbasierten Regelung

3.5.4 Fahrdynamische Merkmale

Die reale, robotertypische Fahrdynamik wird neben den physikalischen Gesetzmäßigkeiten durch fertigungstechnische Toleranzen und konstruktive Gestaltung beeinflusst. Diese Einflüsse, Grenzen und Abweichungen wirken auf die Regelung bzw. Verhaltenskontrolle ein. Sie können innerhalb eines erweiterten, entkoppelten Dynamikreglers berücksichtigt und kompensiert werden, um den Roboter im Rahmen der Möglichkeiten und seiner Leistungsfähigkeit zu manövrieren.

In Tab. 3.2 sind vier Attribute bezeichnet, die für eine ungenaue Positionierung und Bahnführung verantwortlich sind und damit für eine präzise Lageregelung des Roboters LUKAS beachtet werden müssen. Während die Fälle A und B konstruktiv einkalkuliert sind, haben die Fälle C und D besonders störenden Charakter und gewinnen bei höheren Beschleunigungen an Bedeutung.



	Merkmal	Ursache	Korrekturmaßnahme
A	**Asymmetrische Beschleunigung**	Asymmetrische Dimensionierung des dreirädrigen Antriebes	Nutzen des vollen Potentials der Frontalfahrt durch entsprechende Orientierung in der Pfadplanung
B	**Kreisdrift (quer)**	Die orthogonale Achslage verhindert Seitenführung entgegen der Fliehkraft	Lokale Korrektur durch Dynamikregler mit Lageüberwachung
C	**Schwerpunktverlagerung, Radschlupf**	Physikalische Gesetzmäßigkeiten	Lokale Korrektur durch Dynamikregler mit Drehimpulssatz und Schlupfkurve
D	**Verzogene Querfahrt**	Spiel zwischen Kranzrad und Felge des Omnirades, Fehlen des diametralen vierten Rades	Lokale Korrektur durch Dynamikregler mit Gegenmoment

Tab. 3.2 Fahrdynamische Merkmale, Ursachen und Korrekturmaßnahmen

3.5.5 Dynamikregler

Der entkoppelte Dynamikregler beruht auf dem inversen Modell des komplexeren dynamischen Systems, welches in Abschnitt 3.4.3 vorgestellt wurde. Der Regler gewährleistet über die Kontrolle der Radkräfte insbesondere die schlupfgeregelte Bewegung des Roboters, die Orientierung und Positionierung [81]. Die Struktur lässt sich anhand der Abb. 3.25 und den bekannten Grundlagen wie folgt erläutern:
Der Dynamikregler ist analog dem Motorregler in drei Ebenen unterteilt, regelt jedoch die kinematischen Größen des Roboterschwerpunktes. Der Lageregler nutzt ein polares Koordinatensystem und besitzt für eine bestmögliche Positionierzeit die Charakteristik einer Wurzelfunktion. Das Positionierproblem kann dadurch auf eine eindimensionale Regelung zur Vorgabe eines Geschwindigkeitsbetrages vereinfacht werden. Nachfolgend arbeitet der PI-Geschwindigkeitsregler auf Basis der lokalen kartesischen Koordinaten des Steuervektors $\dot{\mathbf{x}}$ und gibt die kartesische Sollbeschleunigung $\ddot{\mathbf{x}}$ des Roboters vor. Die geforderte lokale Sollbeschleunigung wird über die bekannten Parameter der inversen Impulsgleichungen (3.3.4) in Radkräfte transformiert.

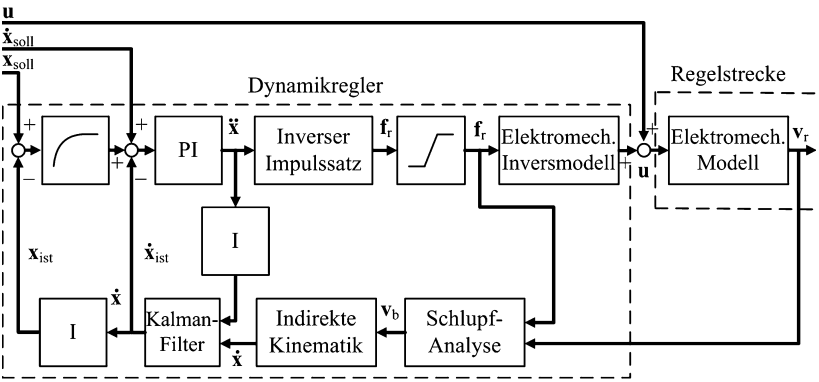


Abb. 3.25 Prinzipielles Strukturbild der entkoppelten Dynamikregelung

Das Ziel ist hierbei, die Antriebsmotoren so anzusteuern, dass der Roboter die Sollbeschleunigung im Rahmen seiner Möglichkeiten exakt ausführt. Übersteigt die Vorgabe die Leistungsfähigkeit oder die Grenzen der Haftreibung, so müssen die Radkräfte derart verhältnismäßig korrigiert werden, dass sich die Bewegungsrichtung nicht verändert. Höchste Priorität hat die Erhaltung der Spurtreue, auch wenn die Bewegung langsamer als erwünscht ausgeführt wird. Über das inverse elektromechanische Modell der Motoren werden die notwendigen Spannungen für den stabilen Fahrzustand (d.h. $\dot{\mathbf{v}}_\mathbf{r} = 0$) nach Gleichung (3.2.9) berechnet. Weil die Aufstandskräfte $\mathbf{f}_\mathbf{Z}$ als Funktion der Antriebskräfte $\mathbf{f}_\mathbf{r}$ darstellbar sind, können mit den bekannten Daten und den Gleichungen der Schlupfberechnung (3.3.0), (3.3.1) und (3.3.2) die tatsächlichen Beträge der Bodengeschwindigkeiten $\mathbf{v}_\mathbf{b}$ des Roboters ermittelt werden. Für die Stabilisierung der unabhängig geregelten Längs-, Quer-, und Drehbewegung des Roboters ist die Datenaufbereitung und das Speichern der aktuellen Roboterposition und -geschwindigkeit in separaten raumfesten kartesischen Variablen $\mathbf{x}_\mathbf{ist}, \dot{\mathbf{x}}_\mathbf{ist}$ erforderlich. Der zwischengeschaltete gewichtete Geschwindigkeitsschätzer verbessert die Qualität der Analyse für eine verlässlichere Ortung.

3.5.6 Bahnsteuerung nach dem Prinzip Lissajousscher Figuren

Ein Körper, der gleichzeitig zwei Teilbewegungen ausführt, bewegt sich resultierend in Richtung der Summe ihrer geometrischen Addition. Teilbewegungen von senkrecht aufeinanderstehenden Sinusschwingungen bilden Formen, die nach dem Physiker Jules Antoine Lissajous benannt sind [83]. Die Bahnsteuerung nach dem Prinzip Lissajousscher Figuren ist ein Verfahren, welches diese resultierenden Formen aus den orthogonal zueinander oszillierenden Vektorkomponenten nutzt [84]. Im Gegensatz zur Linearisierung von Trajektorien sind die Informationen über den Streckenverlauf von Bogenfahrten auf noch kürzere, abstraktere Weise zu erfassen.

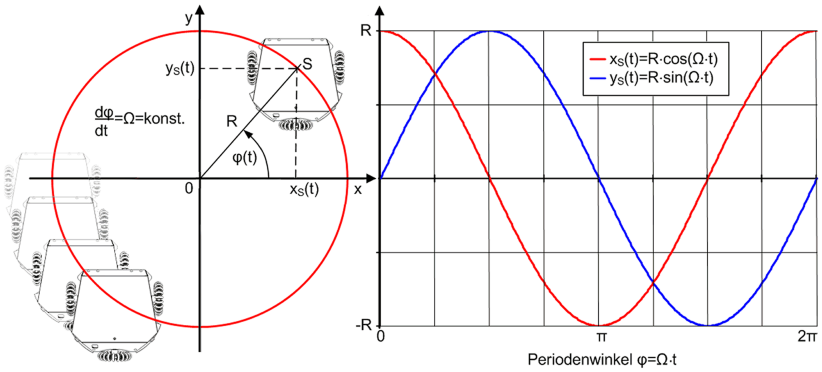


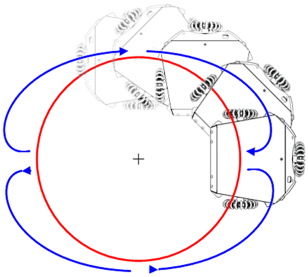
Abb. 3.26 Winkelfunktionen zur Steuerung einer Lissajousschen Kreisfigur

Die Genauigkeit der Bahnführung ist ebenso von der Präzision des überwachenden Lagereglers abhängig.

In Abb. 3.26 sind der resultierende kreisförmige Bahnverlauf des Roboters mit konstanter Orientierung (ohne Eigenrotation) und die zugehörigen Partialschwingungen der Roboterkoordinaten und Radwinkelgeschwindigkeiten aufgezeigt. Diese Bahnfigur wird vom Roboter auf der Grundlage eines einzigen Startbefehls selbständig ausgeführt. Die Teilbewegungen beschreiben zwei Sinusschwingungen gleicher Schwingungsdauer, gleicher Amplitude und mit neunzig Grad Phasendifferenz.

Für den Roboter, welcher selbst als Pendel eines ebenen harmonischen Schwingers zu betrachten ist, erleichtert sich somit deutlich die Koordinierung der Bewegungskurven, so dass die Umsetzung äquivalenter Bahnverläufe keine rechenintensive analytische Lösung der kinematischen Bewegungsgleichung erfordert.

Die eindeutigen Parametervorgaben der Schwingung (Frequenz, Amplitude, Phasenwinkel, Laufzeit) formen die resultierende Bahn, auf der sich der Roboter bewegt, so dass sich diese Art der Bahnsteuerung mit wenig Datentransfer und Berechnungsaufwand auch für die lokale Verarbeitung auf der Ebene des Mikrocontrollers eignet.



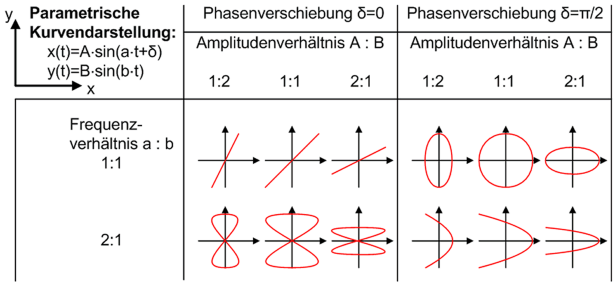


Abb. 3.27 Beispiele Lissajousscher Figuren und Pirouettenfigur

Da oft nur ein Intervall innerhalb einer Periode ausgeführt wird, spielt die Geschlossenheit einer Kurve durch ein rationales Frequenzverhältnis eine untergeordnete Rolle. Das Amplituden- und Frequenzverhältnis sowie die zeitliche Ansteuerung der Schwingungsparameter sind im Rahmen der Wertebereiche frei wählbar. Der Algorithmus ist grundsätzlich für die Umsetzung fixierter Parameter bestimmt, obwohl zeitlich veränderliche Vorgaben auch echtzeittreu übernommen werden. Lissajoussche Figuren enthalten ein breites Spektrum verwertbarer Bahnsegmente, welche sich durch geeignete Parameterwahl flexibel in den ebenen Abmessungen verformen lassen (Abb. 3.27). Das Arbeitsprinzip des roboterinternen Oszillators bietet in Verbindung mit verketteten Bahnsegmenten eine einfache Lösung für nahezu beliebig geformte Trajektorien.

Der Algorithmus des Frequenzgenerators erzeugt proportional zu den trigonometrischen Amplitudenwerten die absoluten kartesischen Ortsvorgaben, welche durch die nachfolgende Regelung ausgeführt werden (Tab. 3.3).
Der Reglerbetrieb startet mit zyklisch veränderten Sollvorgaben. Überschreitet im Zeitregime die jeweilige Taktzahl die zugewiesene Dauer, so beginnt eine neue Periode. Die Steuerung ist dann abgeschlossen, wenn die angeforderte Anzahl an Perioden der langsameren Schwingung durchlaufen wurde. Die realisierte Bahngeschwindigkeit des gesteuerten Roboters bestimmt sich durch die Initialisierung der beiden Schwingungsfrequenzen.
In Kombination mit der frei überlagerten Rotation des Dynamikreglers erweitert sich die lokal kontrollierte Manövrierbarkeit wesentlich. Dies kann besonders an der Bahn des Pirouettenkreises demonstriert werden. Der Roboter rotiert bei der Fahrt eines Vollkreises fortwährend um seine eigene Hochachse, ohne dabei die Orientierung zu verlieren (Abb. 3.27).

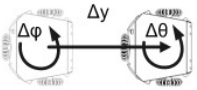
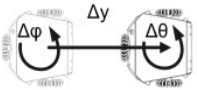
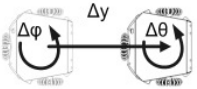
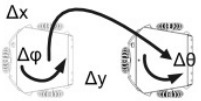
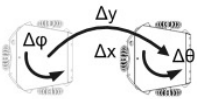
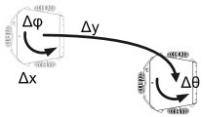
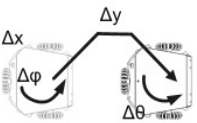
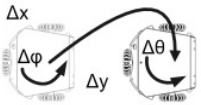
```
Prozedur: Kreisbahnsteuerung
Zweck: Algorithmus zur Abbildung ebener Bahnkurven auf der Grundlage
       von zwei orthogonalen Auslenkungen zeitlich getakteter Winkel-
       funktionen
Parameter: Periode, XTakt, YTakt, XMinTakt, YMinTakt, XMaxTakt,
           YMaxTakt, XDauer, YDauer, XAmpl, YAmpl, Xglobal, Yglobal,
           XStart, YStart, Phase
01. Wenn Regelzyklus > 0, Dann dekrementiere Regelzyklus
02. Sonst
03.     Solange Periode > 0 wiederhole
04.        Wenn XTakt > XMaxTakt, Dann inkrementiere XTakt
05.        Sonst Setze XTakt = XMinTakt
06.              Falls YDauer <= XDauer, Dann dekrementiere Periode
07.        Wenn YTakt > YMaxTakt, Dann inkrementiere YTakt
08.        Sonst Setze YTakt = YMinTakt
09.              Wenn XDauer < YDauer, Dann dekrementiere Periode
10.       Setze Xglobal = XAmpl * Cosinus(2*Pi*XTakt/XDauer + Phase)
                          - XStart
11.       Setze Yglobal = YAmpl * Sinus(2*Pi*YTakt/YDauer) - YStart
12.       Setze Regelzyklus = 20msec
Ergebnis: Xglobal, Yglobal amplitudentreu nachgeregelt
```

Tab. 3.3 Algorithmus der Kreisbahnsteuerung in Pseudocode

3.5.7 Bahngeführte Positionierung

Die bahngeführte Positionierung kann als oberste Befehlsschnittstelle zwischen Roboter und externer Teamkoordinierung bzw. Pfadplanung des Zentralrechners genutzt werden. Im konsequenten Stil einer Top-Down-Struktur bleiben die Details der Robotersteuerung den schnelleren Regelzyklen der untergeordneten Schichten überlassen. Die Organisation und Überwachung der externen zentralen Systemebenen ist damit unabhängig von den verwendeten Modellen der Robotergeneration. Standardverhaltensweisen und Manöver werden innerhalb eines Spurkataloges definiert, dessen Elemente über standardisierte Dimensionsparameter dem aktuellen Bedarfsfall anzupassen sind. Die bahngeführte Positionierung umgeht durch den geringeren Datentransfer die entwicklungs- und anlagenspezifischen Engpässe innerhalb des Systems. Diese lagen zum Zeitpunkt der Entstehung in der hohen Bildverzögerung und der geringen Bandbreite der Funkübertragung. Aufgrund dessen liegt der Schwerpunkt auf der Minimierung der externen Befehlsrate und der erhöhten Eigenkontrolle des mobilen Roboters. Eine bloße Linearisierung von Bahnkurven in Form von direkt angesteuerten Wegpunkten [85] erfordert jedoch mit hoher Diskretisierung ebenfalls hohe Datenraten. Die Ansteuerung einer nicht linear zu erreichenden Zielposition ist daher von Eingaben zur konkreten Robotersteuerung so zu reduzieren, dass mit minimalen Befehlspaketen die Zielkoordinaten, die Form der Bahn und die gewünschte Orientierung eindeutig bestimmt sind. Tab. 3.4 listet den Befehlssatz des entworfenen Spurkataloges auf, dessen Bahnformen die Elementarbewegungen aus der unterlagerten Ebenen kombinieren und integrieren. Analog der Manövergrammatik von Moore und Flann [86] stellen Abfolgen von Zielpunkten die geplanten Pfadlinien dar, wobei die entstehenden Segmente den definierten Musterelementen zugeordnet werden. Andere Mannschaften reduzieren die Befehlsebene zum Roboter auf permanente Geschwindigkeitsvorgaben mit einem hohen Datenfluss und verarbeiten die Lage- und Bahnregelung mit optimierten Algorithmen in der zentralen Software der Mastersteuerung [80, 87]. In der Anwendung des entkoppelten Lagereglers ist es gleichwertig, ob die

Positions- bzw. Bahnsteuerung auf lokaler oder globaler Ebene umgesetzt wird. Unter permanenter Bildüberwachung werden dynamische Aktionen flexibel korrigiert und mit sofortiger Neuorientierung der Zielvorgaben vom Roboter ausgeführt.

Form	Bezeichnung	Basiselement	Parameter
	**Ball Andribbeln**	Drehstrecke, Dribbeln mit Lichtschranke	Länge, Richtung, Endorientierung relativ zur Startposition
	**Ball Zielschuss**	Drehstrecke, Schuss durch Lichtschranke	Länge, Richtung, Endorientierung relativ zur Startposition
	**Gerade**	Drehstrecke	Länge, Richtung, Endorientierung relativ zur Startposition
	**Frühes Ausweichen**	Lissajous-Segment	Länge, Richtung, Startorientierung, Breite, Endorientierung s.o.
	**Kreisbogen**	Kreissegment	Länge, Richtung, Startorientierung, Breite, Endorientierung s.o.
	**Ellipsenbogen**	Ellipsensegment	Länge, Richtung, Startorientierung, Breite, Endorientierung s.o.
	**Langes Ausweichen**	Streckenfolge	Länge, Richtung, Startorientierung, Breite, Endorientierung s.o.
	**Spätes Ausweichen**	Lissajous-Segment	Länge, Richtung, Startorientierung, Breite, Endorientierung s.o.

Tab. 3.4 Spurkatalog der bahngeführten Positionierung

4 Entwurf der Mastersoftware XBase

4.1 Aufbau der Experimentalplattform

Bevor auf Details der Programmstruktur eingegangen wird, sind die Systembedingungen aufzuzeigen, unter denen die Software arbeitet. Analog dem Regelwerk der Small-Size-League darf eine zentrale Spielsteuerung installiert sein, sofern sie autonom agiert. Das Spielgeschehen auf dem Feld hat aufgrund der leistungsstarken Roboter ein hohes Tempo, so dass eine Steuerung aus globaler Perspektive preiswerter und effektiver ist. Analog der allgemeinen Prinzipdarstellung des Systemaufbaus in der SSL (siehe Abschnitt 2.4) beinhaltet Abb. 4.1 die grafische Darstellung und technische Daten der speziellen Systemhardware. Der zentrale Computer übernimmt mit dem internen Framegrabber und dem angeschlossenen Funkmodem die Verarbeitung der Kamerabilder und die teamübergreifende Roboterkontrolle. Das Programm des XBase-Servers ist durch Treiber und System-Bibliotheken an die Kamera und das Microsoft-Betriebssystem gekoppelt. Demgegenüber können die Client-Programme hardwareunabhängig im Netzwerk arbeiten. Das System kann aktuelle Feldinformationen in Abhängigkeit der Bildwiederholrate der Kamera mit bis zu 50Hz generieren. Das Ansprechen der Roboter erfolgt vom PC-Funkmodem im Einsatzfall unidirektional (engl.: pushing), um die Zykluszeiten im Sendeumlauf aller zu steuernder Roboter gering zu halten (ca. 3ms). Im Roboter LUKAS arbeitet die Regelschleife des Mikrocontrollers mit 1,6ms am schnellsten und garantiert somit die Stabilität der gesamten Kaskadenregelung. Zum Zwecke der Statusüberwachung ist die bidirektionale Sendemethode des Polling vorgesehen.

4.2 Anforderungen und Eigenschaften

Die Grundzüge des zentralen Masterprogramms legte schon eine Vorarbeit fest [10].

Im Rahmen der aktuellen Arbeit wurden Bildverarbeitung, Objekterkennung und Verhaltenssteuerung überarbeitet und ausgebaut [65]. Da es sich um einen Entwicklungsrahmen handelt, sind Funktionselemente enthalten oder in Planung (z.B. Formationssteuerung), die sich von einem Wettkampsystem der RoboCup-Small-Size-League unterscheiden. Als Überblick der Softwarearchitektur von XBase sind die wesentlichen strukturellen Eigenschaften und Charakteristika wie folgt vorgestellt.

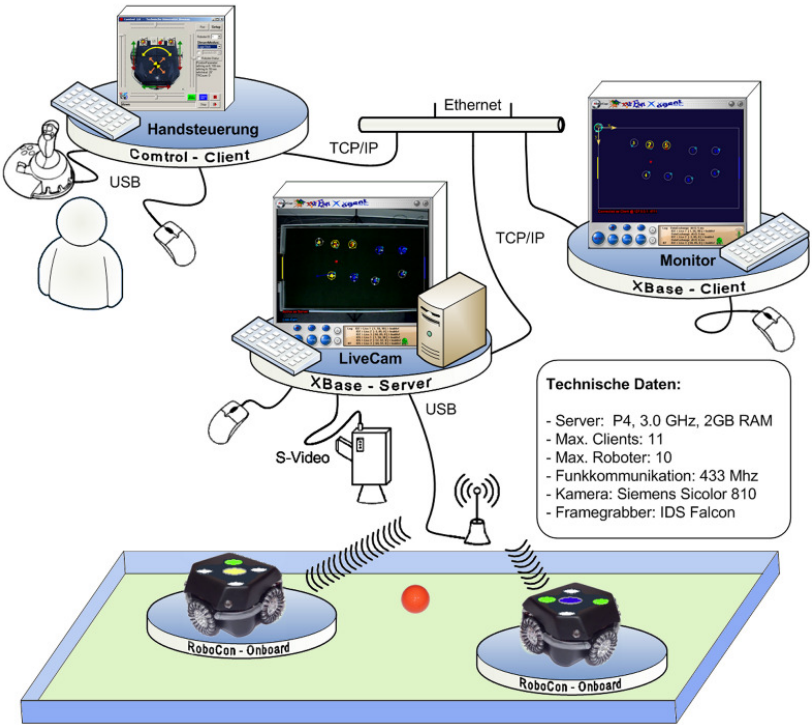


Abb. 4.1 Technische Übersicht des Multi-Roboter-Systems

- **Bildverarbeitung**

- echtzeittreue 100% Erkennungsgüte bis 2m/s Objektgeschwindigkeit (80% bis 4m/s)
- optimale Ortung und Objekt-Stabilisierung
- Maximum-Likelihood-Farbklassifikation auf Basis der Mahalanobis-Distanz
- Bitmapformat 768*288 Pixel, minimierter Halbbildeinzug mit 50fps (noninterlaced)
- Gleitkommaberechnung für hohe Genauigkeit
- Bildsegmentierung mit Zeilenkoinzidenzverfahren
- Kalibrierung für wechselnde Lichtbedingungen mittels überwachten Lernens
- Trikotdesign für leichte Identifizierung und Orientierungsanalyse

- **Hauptprogramm / Bedienoberfläche**

- interaktive Programmoberfläche zur Kontrolle von Robotern oder Robotergruppen
- intuitive Systembedienung, Analyse und Menüführung
- modulare Funktionsstruktur innerhalb eines übersichtlichen Dialog-rahmens
- zentrale Systemsteuerung und Funktionsintegration der Bildverarbeitung, Objekterkennung, Verhaltenssteuerung, Servers/Client und des Simulators
- multifunktionale Nutzung des Programmrahmens im Server-, Client- oder Simulatorbetrieb innerhalb einer verteilten Struktur
- Bereitstellen von Schnittstellen zum Benutzer, zu den Robotern, der Kamera sowie weiteren Anwendungsprogrammen (z.B. Konsolen, Agenten, Monitore)
- Client-Server-Betrieb zur Realisierung von Single-Player bis Multi-Team-Play
- Informationsdisplays für systeminterne Detailüberwachung

- **Verhaltenssystem**

- hybride Verhaltensarchitektur für echtzeittreue, autonome Roboterkoordination mit Wahl der Szenarien und Verhaltensmuster je nach Anwenderkriterien
- Beherrschen von Pfadverfolgung, Kollisionsvermeidung und Positionierung innerhalb situationsabhängiger Aktionen
- Verhaltenssteuerung von Formationsverbänden mehrerer Roboter
- Regie automatischer Aktionsverteilung und Beherrschen von Ballanwendungen

4.3 Modulare Systemarchitektur

Nachdem der Feldroboter LUKAS und seine funkübertragene Anbindung beschrieben wurden, ist es erforderlich, sich den verbleibenden globalen Systemmodulen der Bildverarbeitung und der Verhaltenssteuerung zuzuwenden. Geschaffen wurde ein Programmrahmen, der einer autonomen Spielsteuerung als Basis dient. Nach Abb. 4.2 verknüpfen sich die einzelnen Module des Programms nahezu sternförmig um die Zentralkartei der Objektverwaltung. Der momentane Programmrahmen besteht aus sieben parallel arbeitenden Modulen, von denen jedes Modul im Quellcode der Software mindestens einen eigenen Thread repräsentiert.
Beginnend mit einer robusten manuellen Steuerung wurden der systematische Aufbau und die Verbesserung einzelner autonomer Elemente betrieben. Zum gegenwärtigen Zeitpunkt sind die wesentlichen Teile fertig gestellt, obwohl die Verhaltenssteuerung noch fragmentarisch und eine Vorhersage sowie das Schiedsrichtermodul in Planung sind. Ein Kalman-Filter ist für eine präzisere Objektvorhersage im verwendeten Steuermodus der roboterinternen Lagekoordinierung noch entbehrlich. Sollte eine Verhaltenskontrolle auf Basis zu übertragender Fahrtgeschwindigkeiten einer externen Lageregelung übernommen werden, ist es sinnvoll, dieses Vorhersagemodul zu integrieren, um die Latenzzeit im Regelkreis zu kompensieren [61].

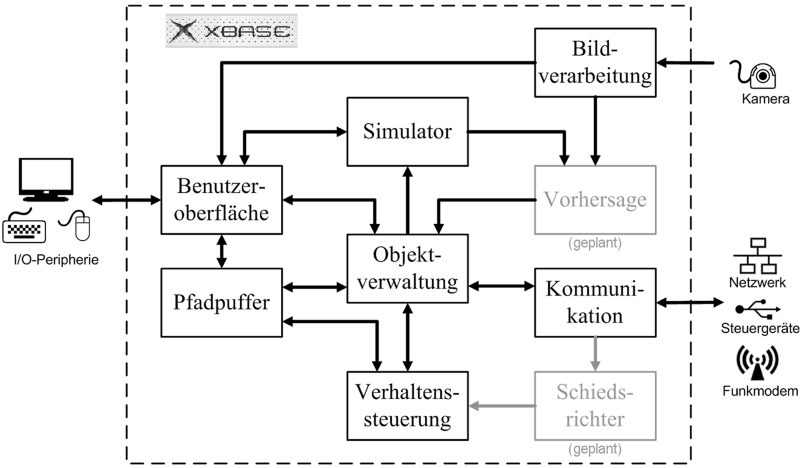


Abb. 4.2 Modulstruktur der Mastersoftware

Die aktuellen Aufgaben der Module lassen sich wie folgt stichpunktartig beschreiben:

- **Bildverarbeitung**
  Bereitstellen von Informationen (Position, Orientierung, Kennung, Geschwindigkeitsvektor) der durch das Kamerabild sichtbaren farbmarkierten Objekte des Spielfeldes
- **Pfadpuffer**
  Bereitstellen von Pfadinformationen, d.h. Messpunkte und Relativbeziehungen im aktuellen und vergangenen Bahnverlauf, zukünftig Pfade und Objektkonstellationen
- **Objektverwaltung**
  zentrale Schnittstelle bzw. Zwischenspeicher zur Organisation der aktuellen Daten und Parameter der erkannten, sichtbaren Feldobjekte (maximal 15 Roboter und ein Ball)

- **Verhaltenssteuerung**
  Planung und Kontrolle von Verhaltensweisen (Aktionen) einzelner Roboter sowie Überwachung und Ansteuerung der Roboteraktionen im gesamten Verhalten des Teams (Regie)
- **Kommunikation**
  Verwaltung des automatischen Datentransfers (TCP/IP-Socket, COM-Port) innerhalb der verteilten Struktur entsprechend dem Betriebsfall zwischen Server, Klienten, Robotern oder dem Simulator
- **Simulator**
  Ersatz der realen exekutiven Feldobjekte und der Bilderfassung durch ein virtuelles Weltmodell des Spielfeldes; Test- und Analyseumgebung der Verhaltenssteuerung; Monitor und Anwenderumgebung im client- oder kameralosen Betriebsfall
- **Benutzeroberfläche (**Abb. 4.3**)**
  Setup und Bedienschnittstelle des Programms für die Funktionalität der verfügbaren Verhaltenssteuerung in diversen Betriebsfällen (LiveCam-Server, Remote-Client, Simulator)

4.4 Benutzeroberfläche

Die grafische Benutzeroberfläche folgt dem Konzept der Funktionsintegration (engl.: all-in-one) und hat die Aufgabe, eine schnelle und nutzerfreundliche Anwendung des gesamten Systems zu ermöglichen. Hervorzuheben ist der Schwerpunkt der interaktiven Oberfläche des Spielfeldfensters. Viele Funktionen können direkt an den eingeblendeten Roboterobjekten ohne Listenmenü per Drag&Drop-Maussteuerung ausgeführt werden (z.B. Roboter selektieren, gruppieren, positionieren, Pfadlinien zeichnen und verfolgen). Des Weiteren steht ein flexibles Menü für direkten Zugriff auf Kontrolloptionen der Verhaltensteuerung und Feldeigenschaften bereit.
Abb. 4.3 zeigt auch das Display für die Überwachung von internen Daten- und Befehlsströmen sowie Objekt- und Systemparametern (z.B. Zeitregime, Regie,

Roboteraktionen, Port-Logger, Geräteinformation, Objektkartei oder Bildverarbeitungsgüte). Systemeinstellungen über Server, Kommunikation und Bilderfassung finden sich im separaten Setup. Aus der Programmoberfläche heraus können automatisch parallel vernetzte Client-Anwendungen gestartet werden (Comtrol-Handsteuerung, XBase-Client).

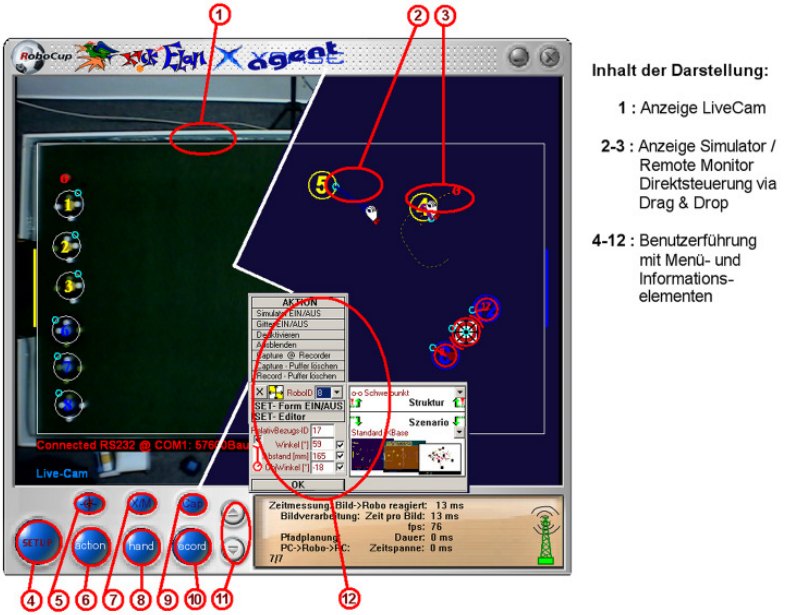


Abb. 4.3 Merkmale der Benutzeroberfläche

4.5 Betriebsmodi

Die Funktionalität der Verhaltenssteuerung steht dem interaktiven Overlay der Programmoberfläche nicht nur in der Einblendung verschiedener Szenariendarstellungen, sondern auch bei der visuellen Überwachung im Live-Bild der Kamera oder der Umschaltung in andere Betriebsmodi zur Verfügung.

Unter der Rahmenstruktur der Oberfläche (Abb. 4.3) sind diverse Betriebsmodi vereinigt, die sich den lokalen Gegebenheiten anpassen können. Quelle der Objektgenerierung können sein:

- LiveCam-Bildverarbeitung des Framegrabbers
- Cient-Monitor vom XBase-Server (Remote-Betrieb)
- Simulation der Feldobjekte

Mehrere XBase-Programme können somit eine vernetzte Struktur aufbauen, um verteilte Aufgaben zu übernehmen (siehe Abb. 4.1). Die Variabilität dieser Struktur bestimmt durch die sich vervielfachende XBase-Funktionalität, ob die Roboter durch ein Single- oder Multi-Agentensystems koordiniert werden. Es ergeben sich nach Tab. 4.1 Betriebsmodi, in denen das Verhaltenssystem für die Steuerung der Objekte genutzt werden kann.

Nach Wooldridge [88] ist ein Agent als ein abgeschlossenes Computersystem definiert, das in einer bestimmten Umgebung arbeitet und in der Lage ist, darin flexibel und autonom zu agieren, um seine vorgegebenen Ziele zu erreichen.

Im Gegensatz zur Systemstruktur der Mid-Size-League, bei der jeder Roboter in seinem Verhalten durch einen eigenen Agenten repräsentiert wird, handelt es sich in der XBase-Grundstruktur um ein Single-Agent-System mit einer Multi-Roboterkontrolle. Der Agent verkörpert hier die oberste Ebene der Verhaltenssteuerung. Die Vorteile der globalen Weltperspektive können mit einem übergeordneten globalen Agenten direkt genutzt werden. Multi-Agenten-Systeme müssen erst eine gemeinsam operierende Plattform auf Basis gegenseitiger Kommunikation aufbauen. Innerhalb der übergeordneten verteilten Netzwerkstruktur kann jedoch auch XBase als Multi-Agenten-System agieren, um gemeinsam mit mehreren Teams von Robotern ein Ziel zu erreichen.

Netzwerkstatus	*Live-Cam*	*Simulator*	*Remote Monitor*
Client	**-**	**-**	**x**
Server	**x**	**x**	**-**
Offline	**x**	**x**	**-**

Zeichenerklärung: **X** Betriebsmodus verfügbar

Tab. 4.1 Übersicht der Betriebsmodi

4.6 Bildverarbeitung und Objekterkennung

4.6.1 Technische Grundlagen

Die Wahrnehmung der Umwelt ist für alle Roboter eine fundamentale Notwendigkeit. Wenngleich viele Arten von Sensoren eingesetzt werden, ist die optische Bilderfassung dabei eine der mächtigsten und sogleich preiswertesten. Die Kamera kann im Bild Informationen aus Farbe und Form liefern. Der Umfang der Auswertung ist dadurch auch von der Kodierung der Information abhängig. In den nachfolgenden Abschnitten soll auf die Realisierung der eigenen Bildverarbeitung eingegangen werden.
Eckdaten und Fakten zur Leistungsfähigkeit wurden bereits in Abschnitt 4.2 aufgeführt. Die allgemeine Vorgehensweise des Verfahrens verdeutlicht Abb. 4.4. Im Setup von XBase befinden sich Konfigurationsmöglichkeiten, um die Bildauswertung durch Anlernen von Farbklassen (TeachColors) oder des Ladens von Referenzparametern den herrschenden Lichtbedingungen anzupassen [89].
Die Bildverarbeitung des Systems arbeitet auf der Grundlage einer Farbkamera Siemens Sicolor 810. Diese enthält als Wandlerelement einen CCD-Chip und liefert ein PAL-konformes Ausgangssignal an die PCI-Frame-Grabber-Karte. Bei einer Messfrequenz von 50Hz werden 752 x 288 Bildpunkte als (noninterlaced) Halbbild aufgelöst. Die Integrationszeit des CCD–Chips ist für eine schnelle Erfassung mittels Shutter auf 1/120s eingestellt. Weißlichtabgleich und Gamma-Korrektur können automatisch oder manuell über die Software erfolgen. Damit das Spielfeld abgebildet werden kann, ist die Kamera in ca. 2m Höhe angebracht. Bei der Frame-Grabber-Karte handelt es sich um eine Karte vom Typ Falcon der Firma IDS, welche das anliegende Signal in Echtzeit digitalisiert.
Für den kontinuierlichen Einzug von Bildern wird der RAM-Speicher, der die Bilddaten aufnimmt, als dreiteiliger Ringpuffer organisiert. Im laufenden Betrieb werden die digitalisierten Kamerasignale fortlaufend in den jeweils aktuellen Speicherbereich geschrieben. Ist nach 20ms ein komplettes Halbbild

eingezogen, schaltet der Ringzeiger auf den nächsten Ringpufferbereich weiter. Der Vorteil des Mechanismus ist, dass nach einer weiteren Zykluszeit das nächste Bild schon digitalisiert im nächsten Puffersegment zur Auswertung bereitsteht.

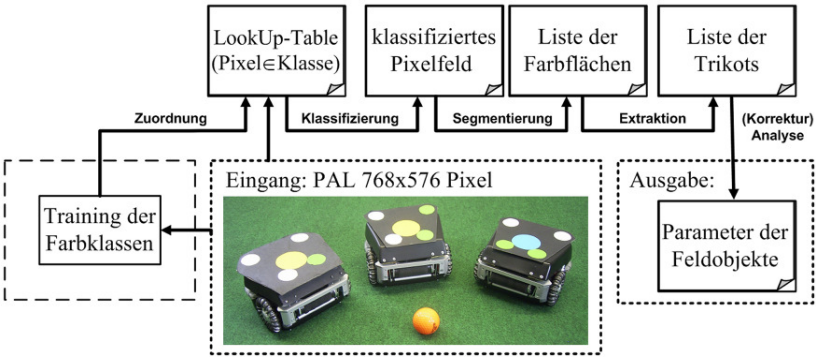


Abb. 4.4 Schematischer Ablauf der Bildverarbeitung

4.6.2 Maschinelles Lernen

Das maschinelle Lernen wird mit dem systemeigenen Anwenderprogramm TeachColors durchgeführt. Manuell wird im Live-Bild der Kamera ein Gebiet ausgewählt, welches als Vertreter der markierten Farbklasse zu trainieren ist. Nachdem für jede Farbklasse typische Vertreter gewählt und angelernt wurden, ist der Datensatz als Datei exportierbar. Da ein Maximum-Likelihood-Klassifikator Verwendung findet, wird für jede der 7 Farbklassen jeweils der Schwerpunkt und die inverse Kovarianzmatrix S^{-1} abgespeichert.
Der Look-Up-Table späterer Auswertung wird ebenfalls mit TeachColors aufgestellt. Hierbei wird für jeden Pixel, d.h. RGB-Tribel $\vec{x}$ des Farbraumes, die Mahalanobis-Distanz zu jedem Schwerpunkt-Tripel $\vec{y}$ der Farbklassen berechnet. In dem Look-Up-Table wird für jeden Pixel diejenige Farbklasse

gespeichert, welche den geringsten Abstand zum untersuchten Pixel besitzt. Der Berechnung der Mahalanobis-Distanz d liegt folgende Formel zugrunde:

$$d(\vec{x},\vec{y}) = \sqrt{(\vec{x}-\vec{y})^T \cdot S^{-1} \cdot (\vec{x}-\vec{y})} \qquad (4.1)$$

4.6.3 Farbklassifikation und Bildsegmentierung

Die Pixel des Live-Bildes werden im relevanten Bildbereich zeilenweise durchlaufen, wobei die Farbkanalwerte der Pixel als Eingangsgröße für den Look-Up-Table dienen. Anhand des Tripels im RGB-Farbraum kann durch einen Zugriff auf den entsprechenden Eintrag im Look-Up-Table die Farbklasse bestimmt werden. Das Ergebnis der Klassifikation ist ein dem Eingangbild äquivalentes, zweidimensionales Array, welches die Farbklasse der Pixel im Eingangsbild codiert (Abb. 4.5). Anhand des klassifizierten Arrays wird des Weiteren eine Farbsegmentierung nach einem modifizierten Zeilenkoinzidenzverfahren durchgeführt [89]. Das Verfahren tastet wiederum zeilenweise das Eingangsarray ab, wobei jeder Pixel anhand seiner Farbklasse und der seiner Nachbarn einem Label zugeordnet wird. Bei der Verwendung einer N_4 Nachbarschaft (Von-Neumann-Nachbarschaft) ist der Labelindex für den untersuchten Pixel vom westlichen und vom nördlichen Pixel abhängig. Die Labelindizes werden im Ausgangsarray gespeichert. Besitzen zwei benachbarte Pixel die gleiche Farbklasse, so erhalten sie einen identischen Labelindex und gehören damit zum gleichen Farbsegment. Das Verfahren ist imstande, das klassifizierte Fünf-Farben-Bild in zwei Bilddurchläufen zu segmentieren, zu verschmelzen und ein Segment-Array zu beschreiben, das zusammenhängende Regionen gleicher Farbklasse mit nur vier Variablen beschreibt (x,y-Koordinaten des Segmentschwerpunktes, Anzahl der Segment-Pixel und Farbklasse).

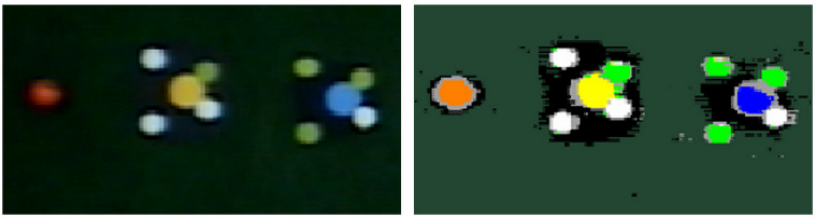


Abb. 4.5 Ausschnitt Live-Bild (links) und zugehörendes klassifiziertes Bild (rechts)

4.6.4 Trikoterkennung

Das Zusammensetzen eines kompletten Robotertrikots erfolgt durch Zusammenstellen gültiger Farbsegmente des Segment-Arrays. Werden beim Array-Durchlauf Segmente der Teammarker Gelb oder Blau gefunden, stellen sie somit das Zentrum eines Robotertrikots dar. Ein erneuter Durchlauf des Segment-Arrays liefert gegebenenfalls Regionen, die durch eine Abstandsmessung vom Teammarker und mit der entsprechenden Weiß- oder Grün-Kennung den nahen oder fernen Markern zuzuordnen sind. Das symmetrische Butterfly-Trikot mit der deutlichen Farbgebung vermindert das Auftreten von Fehlzuordnungen. Fehlerhaft erkannte Marker müssen durch Sortieren oder Rekonstruieren im Rahmen der Möglichkeiten kompensiert werden. Für eine weiterführende Identifizierung sind im Roboter-Array daraufhin Trikot-Einträge gespeichert, welche die fünf nötigen Indizes von Farbsegmenten des Teammarkers sowie der umliegenden Bitmarker beinhalten (Abb. 4.6).

4.6.5 Objektinitialisierung und Objektverfolgung

Die Zuordnung eines Roboters zu einem Objekt des Bildes berechnet sich aus den fünf Markersegmenten der Trikot-Einträge. Der Schwerpunkt des Masterkreises wird zum Standort des Roboters. Hat ein Bitmarker die Farbe Grün, zählt er als 1, ist er weiß, dann wird er als 0 eingeordnet. Die Identitätskennung des Roboters errechnet sich mit der Formel:

$$ID = 2^0 \cdot Bit.0 + 2^1 \cdot Bit.1 + 2^2 \cdot Bit.2 + 2^3 \cdot Bit.3 \qquad (4.2)$$

Durch Addition der beiden diagonalen Vektoren, die sich aus den gegenüberliegenden Vorder- und Hintermarkern ergeben, kann die Orientierung des Roboters am Summenvektor abgelesen werden (Abb. 4.6). Liegt bei der Analyse der Trikotmodelle ein Fehlerfall vor, der nicht rekonstruierbar ist, so werden der Orientierungswinkel und die ID-Kennung aus dem vorherigen Bild übernommen. Dies sorgt dafür, dass trotz möglicher Fehler die Roboter möglichst lange erkannt werden. Erst nachdem die Objektverfolgung (engl.: tracker) und eine Rekonstruktion misslingen, wird die Information eines unvollständig erkannten Trikots verworfen. Die zusätzlichen Roboterparameter der Geschwindigkeit und Beschleunigung werden ebenfalls mit Vorwissen des vergangenen Bildes berechnet und gemeinsam der Objektverwaltung in XBase zugeführt.

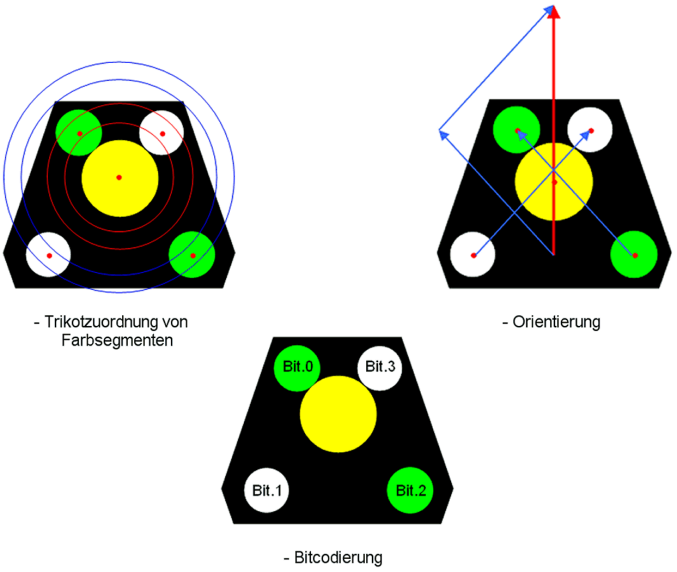


Abb. 4.6 Rekonstruktion, Identifizierung und Ausrichtung des Trikots

4.7 Verhaltenssteuerung

4.7.1 Grundstruktur

Eckdaten, die das Verhaltenssystem bis dato zu erfüllen hat, wurden schon im Abschnitt 4.2 genannt. Diese Arbeit erhebt nicht den Anspruch, eine RoboCup-KI zu entwerfen. Für das Erstellen einer größeren Vielfalt benötigter Verhaltensaktionen eines RoboCup-Spiels legt diese Arbeit die ersten Grundlagen. Die Entwicklung eines einsatzfähigen Trainers und eines Strategieagenten, der sich der Regiekoordinierung des übergeordneten Teamverhaltens widmet, ist Inhalt weiterführender Arbeiten.
Hauptziel dieser Arbeit ist das Etablieren einer eigenen Plattform, die die Grundelemente zur Koordinierung mehrerer Roboter vereinigt. Wie es schon die verschiedenen Ansätze im Abschnitt 2.7 zeigen, gehören dazu im Allgemeinen eine hierarchische Ebenenstruktur mit untergeordneter Pfadplanung bzw. Kollisionsüberwachung. Diese Komponenten lassen sich auch im Grundgerüst von XBase identifizieren (Abb. 4.7).

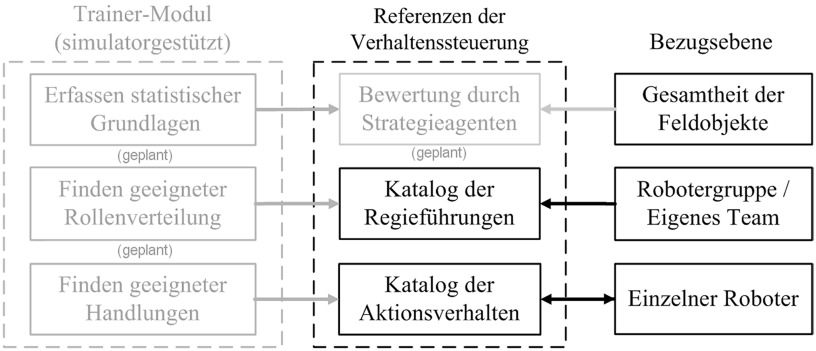


Abb. 4.7 Hierarchie der Verhaltenssteuerung

Nach dem Vorbild der Kaskadenregelung arbeitet der Thread des Strategieagenten mit dem längsten Zeitintervall und verkörpert die höchste der drei

Hierarchieebenen des Verhaltenssystems (siehe Abb. 4.7). Planender Agent, vermittelnde Regie und handelnde Aktion ergänzen sich zu einer hybriden Verhaltensstruktur. Jede Ebene verfügt über eine Wissensbasis in Form hinterlegter Funktionen, welche manuell oder unter gegebenen Bedingungen durch die benachbarte höher liegende Ebene automatisch aktiviert werden.
Dieses diskrete Schema dient dem unproblematischen Hinterlegen neuer Fähigkeiten durch festes Programmieren oder systematisches Anlernen. Die manuelle bzw. automatische Analyse von gestellten Spielsequenzen durch einen geplanten Trainer soll die Mächtigkeit und Flexibilität der Steuerung unterstützen.
Der den Spielverlauf beobachtende Strategieagent hat künftig die Aufgabe, eine günstige Art des Teamverhaltens (Angriffsaufstellung, Verteidigungsaufstellung etc.) aus der Auswahl hinterlegter Funktionsmuster der Regie zu wählen.

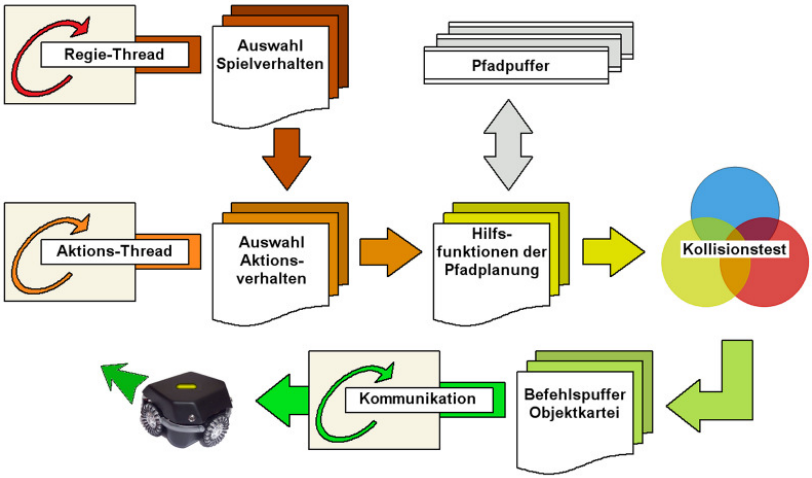


Abb. 4.8 Arbeitsprinzip zur Verhaltenssteuerung eines Roboters

Die Arbeitsweise der im Programm parallel arbeitenden Threads der Regie- und Aktionsmodule soll genauer dargestellt werden. Der zyklische Programm-

ablauf der Regie organisiert die entsprechende Auswahl einzelner Roboterverhalten (Formationsteilnehmer, Torwart, Ballschuss, Deckung etc.) und signalisiert unter Zeit- oder Situationsbedingungen die notwendigen Aktionsparameter in der Roboterkartei (Abb. 4.8).
Nachfolgend durchsuchen die Funktionen der Aktionsebene periodisch alle zugehörigen Parameter der Roboterverwaltung und überwachen im Falle der Aktivierung die Umsetzung der Aktion. Aktionen besitzen das interne Schema:

(1) Konkretisierung der Eingangsparameter (Team, Ziel, Abstandsmaße etc.)

(2) Planung der Ziellage des Roboters unter Berücksichtigung der Feldsituation

(3) (De-)Aktivierung von Sonderfunktionen (Schuss, Dribbler, Lichtschranke)

(4) Korrektur des aktuellen Verhaltens durch Absetzen von Befehlspaketen

Das Vorgehen mündet in einer systembedingten Transformation, wobei die zuständige Funktion für sämtliche zu steuernde Objekte aus den Zieldaten des Weltsystems die jeweiligen Kommunikations-Datenpakete der lokalen Ansteuerung berechnet. Gleichzeitig werden hierbei die Zieldaten eines jeden einzelnen Roboters einem Kollisionstest unterzogen und gegebenenfalls derart korrigiert, dass eine gefahrlose Bewegung in die Nähe des Zielpunktes möglich ist. Registriert das Kommunikationsmodul zu sendende Befehlsfolgen, gelangen sie automatisch zum Roboter. Reagiert dieser in der geforderten Weise, ergibt sich eine neue Feldsituation, die nach erfolgter visueller Erfassung vom Verhaltenssystem neu bewertet wird. Das Programm XBase ist mit C++ in der Entwicklungsumgebung Microsoft Visual Studio 2003 unter Nutzung der Microsoft Foundation Classes (MFC) erstellt.
Das System ist aufgrund der verfügbaren Verhaltensweisen in der Lage, Roboter kooperativ an einer gemeinsamen Formation teilnehmen oder Ballaktionen ausführen zu lassen. Weitere Abschnitte gehen vertiefter auf die geschichtet aufbauende Funktionsstruktur der Kollisionsvermeidung, Pfadverfolgung, Formations- und Ballanwendungen ein.

4.7.2 Kollisionsvermeidung

Auf unterster Ebene der Verhaltenssteuerung ist die Funktion der Kollisionsüberwachung integriert. Sie baut in direkter Nutzung der bahngeführten Positionierung (siehe Abschnitt 3.5.7) auf den roboterinternen Kontrollebenen auf. Der Roboter ist selbst für das präzise Erreichen der Zielposition verantwortlich, wodurch Kontroll- und Kommunikationskapazitäten auf höherer Ebene eingespart werden. Grundprinzip der Zielpunktplanung einer Aktion mit aktivierter Kollisionsüberwachung ist die geradlinige Wegsteuerung vom aktuellen Standort des Roboters. Neue Zielpunkte liegen üblicherweise im Nahbereich der Ist-Position, so dass Wegstrecken bis maximal ein Meter auch eine überschaubare reaktive Kollisionsvermeidung bedeuten. Die Berechnung der Zielposition durch das Aktionsmodul berücksichtigt diese Bedingung wie auch die Prüfung der Hindernisfreiheit im unmittelbaren Zielbereich und die passende Einordnung des Balles als Hindernis oder Angriffsobjekt.
Detektiert die Prüffunktion für eine kollisionsfreie Fahrt ein Hindernis auf der Bewegungslinie zum Zielpunkt (Abb. 4.9), so wird dieses mit ausreichendem Sicherheitsabstand umfahren, indem unter Beibehalten der Zielkoordinaten die Art des betreffenden Bahnelementes aus dem Spurkatalog angepasst wird. Statt der Kodierung eines Streckenelements wird die Kodierung eines geeigneten Bogenelements in die Befehlsfolge eingesetzt (siehe Tab. 3.4).

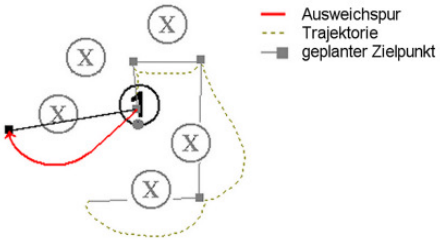


Abb. 4.9 Kollisionsvermeidung in Anwendung adaptiver Spurelemente

Liegt ein Hindernis im Bereich des Ausweichpfades, verfällt die aktuelle Zielposition und das Aktionsmodul erhält die Aufgabe, neue Zieldaten zu ermitteln.
Die drei typischen Grundmuster der Bogenelemente (Tab. 3.4) leiten sich, wie es Abb. 4.10 darstellt, aus der willkürlichen Reduktion nichtlinearer Bahnkurven ab. Obwohl nur wenige Arten von Bögen zur Verfügung stehen, ermöglicht die freie Dimensionierung der Längen-, Breiten-, und Orientierungsparameter eine ausreichende Variabilität. Eine Folge von Ziel-Punkten kann somit in der geeigneten Spurkodierung beliebige Bahnverläufe verschlüsseln, denen der Roboter LUKAS im Toleranzrahmen seiner Positionierfähigkeit selbständig folgt.

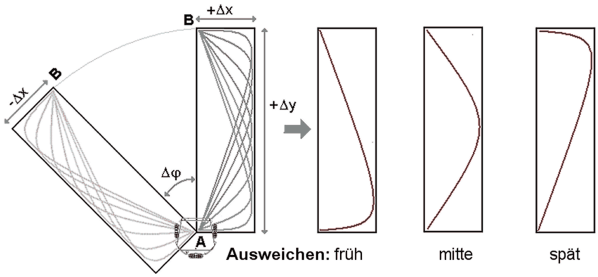


Abb. 4.10 Abstraktion der Bogenmuster

4.7.3 Aktion der Pfadverfolgung

Das Erstellen und Verfolgen von Pfaden ist eine grundlegende Eigenschaft der Spielsteuerung XBase (Abb. 4.11). Übernommen vom beobachtenden Fahrtenschreiber, durch individuelle Benutzereingaben oder durch Berechnungen der Pfadplanung werden die Pfadlinien in Listen von Punktkoordinaten gespeichert. Obwohl der Roboter LUKAS in der Lage ist, selbst Zielpositionen zu finden, präzisiert die permanente externe Positionskontrolle das Fahrverhalten und verhindert frühzeitig ein Summieren von abweichenden Folgefehlern im Kursverlauf.

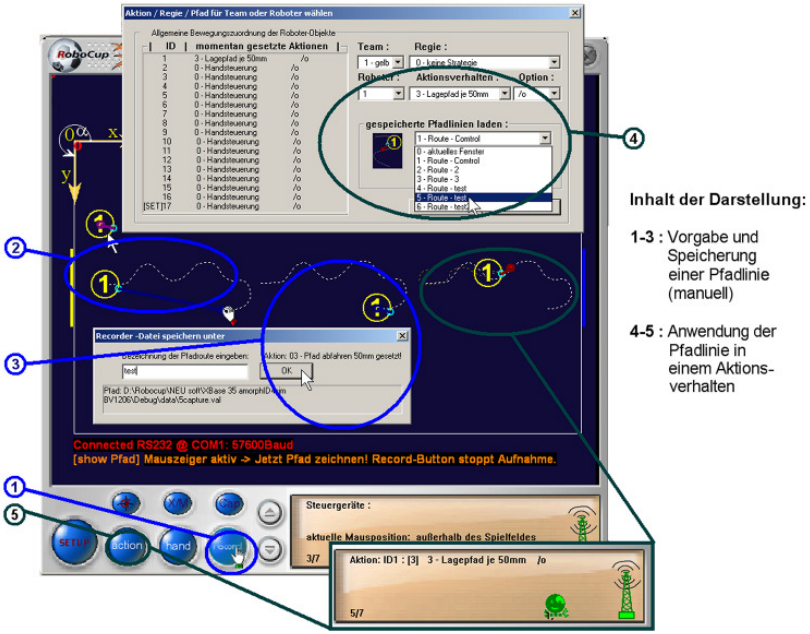


Abb. 4.11 Roboterkontrolle auf Pfadlinien

```
Thread: Aktion eines Roboters(ID-Kennung)
Zweck: Überwachung und Ausführung von Basisverhalten
Parameter: AktionPara, ID
Externer Zugriff: Objektverwaltung, Pfadpuffer

01. Solange Thread aktiv wiederhole
02.     Falls ID berechtigt Und AktionPara = ‚Lagepfad je 50mm'
03.     Dann
04.         initialisiere diskrete Steuerungsparameter
05.         Wenn Istposition innerhalb Wegtoleranz zu Sollposition
06.             Dann suche im Pfadpuffer neue Sollposition
07.                 Wenn Sollposition plausibel
08.                     Dann Sollposition zwischenspeichern
09.         Wenn Zeit- und Lagekriterien des Roboters erfüllt
10.             Dann Sollposition an Roboter übergeben
Ergebnis: Anfahren und Setzen neuer Sollposition
```

Tab. 4.2 Abfolge der Aktion ‚Lagepfad je 50mm' in Pseudocode

Tab. 4.2 kann exemplarisch an einem der anspruchslosen Aktionsverhalten (dem Folgen eines Lagepfades mit Wegmarken je 50mm) den Einblick auf das diskrete Wirkungsprinzip geben.

4.7.4 Aktionen der Formationssteuerung

Um zu einer leistungsfähigen Teamsteuerung zu gelangen, bedarf es des Trainings und der Erfahrung im Umgang mit dem Multi-Roboter-System. Im Rahmen der Verhaltenssteuerung entstanden modifizierbare Aktionen für Formationsszenarien, deren Aufgaben in Test-, Demonstrations- oder Experimentalanwendung begründet sind.

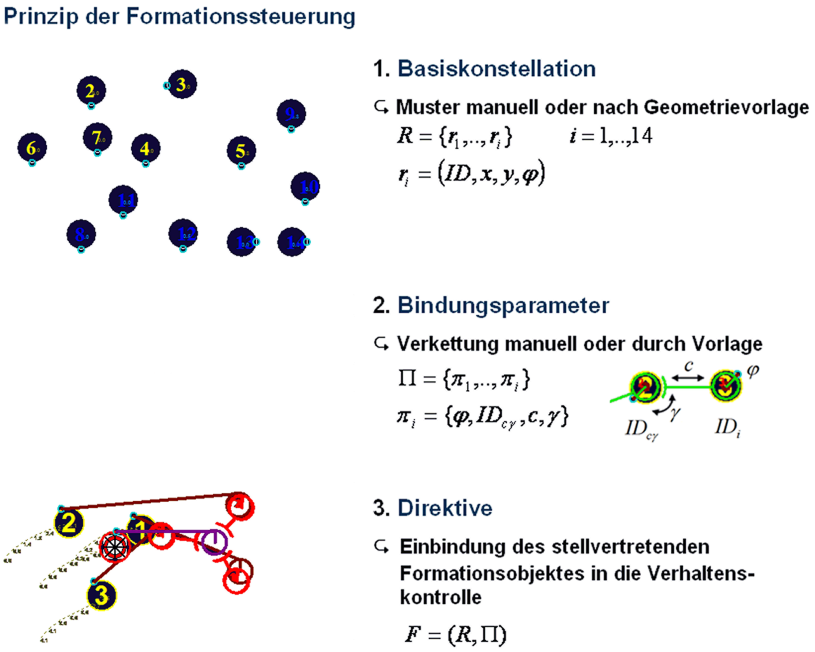


Abb. 4.12 Gruppieren und Positionieren einer Formation

Der Entwurf zur Lagekoordinierung einer Gruppe von Robotern basiert auf der Vorlage eines vernetzten Formationsmusters. Abb. 4.12 beschreibt die prinzipielle Gruppierung einer Menge freier Roboterobjekte im virtuellen Bund des Formationsobjektes (Maske).
Der Anwender verfügt über die genaue Anordnung des Grundmusters und legt zu Beginn die Eigenschaften der Maskenverknüpfung fest. Mit der Bestätigung dieser Gruppierung wird ein virtuelles Roboterobjekt aktiviert, welches manuelle oder automatische Sollvorgaben stellvertretend für die Formationsgruppe entgegennimmt. Der virtuelle Drehpunkt der Gruppenformation liegt nach Wahl des Anwenders entweder auf dem kartesischen Schwerpunkt der teilnehmenden Roboter, im Zentrum des umschreibenden Rahmens oder auf dem Mittelpunkt eines willkürlich festgelegten Teilnehmers. Position und Orientierung des virtuellen Objektes passen sich im Bewegungsprozess dynamisch der realen Konstellation der Roboterobjekte an. Es besitzt eine ID-Kennung wie ein normaler Roboter und befähigt gleichermaßen die freie Positionierung und Verhaltenssteuerung. Zielvorgaben an das virtuelle Formationsobjekt werden automatisch in die entsprechende lokale Verschiebung der Teilnehmer abgeleitet. Die Substitution durch das Formationsobjekt kapselt die Teilnehmer vom direkten Zugriff der Aktionssteuerung ab (Abb. 4.13). Roboter führen hierbei generell das Aktionsverhalten des Formationsteilnehmers aus, welches die korrekte Einhaltung der zugewiesenen Maskenposition sowie die Kollisionsvermeidung überwacht. Der Eingriff auf einzelne Verhaltensaktionen ist dennoch durch die Verwaltung der höheren Regieebene gewährt. Aus der Sicht der Verhaltenssteuerung besteht kein Unterschied zwischen den realen und virtuellen Objekten, so dass im Falle der Erweiterung große Formationen ihrerseits wieder aus hierarchisch gruppierten Formationsobjekten bestehen können (Abb. 4.13).
Das Speichern und Laden von Bindungsparametern der abstrakten Formationsmuster erlaubt in XBase einen erleichterten Umgang und regieüberwachte dynamische Missionswechsel.

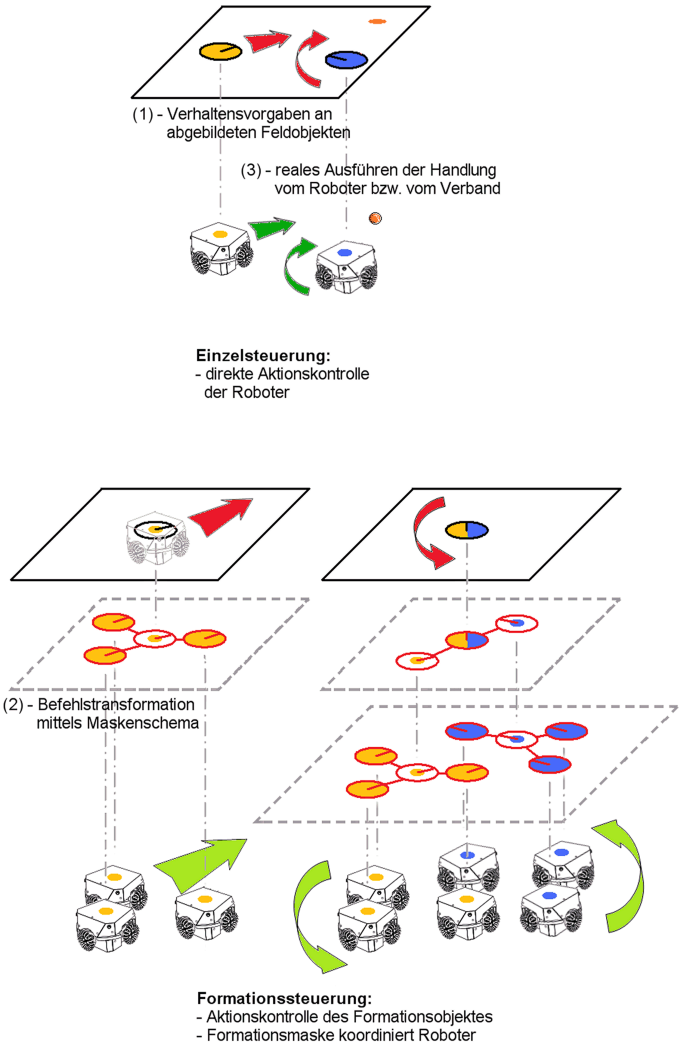


Abb. 4.13 Signalumsetzung der Robotersteuerungen

4.7.5 Modell der polaren Einfachbindung

Die Art des gesamten Verbundes sowie die Lage der Maskenposition relativ zum Bezugsobjekt wird objektspezifisch anhand der polaren Bindungsparameter (Bezugsobjekt, Winkel, Abstand, Orientierung) und ihren jeweiligen Freigaben definiert (Abb. 4.14). Sind einzelne Bindungsparameter nicht durch Vorgaben fixiert, erhöht sich die Flexibilität der Struktur. Die Vernetzung durch relative Einfachbindung bewirkt ebenfalls eine Variabilität in der Gesamtstruktur der Formation, so wie es Tab. 4.3 darstellt. Stern, Strang oder Baumstrukturen sind durch die freie Wahl der Bezugobjekte frei formierbar. Durch eine Doppel- oder Dreifachbindung pro Maske erhält das Baukastensystem noch Potential an stabilen Strukturen höherer Freiheitsgrade. Auch räumliche Polarkoordinaten mit einer zusätzlichen Winkelangabe sind durch die modulare Architektur zu verwalten.

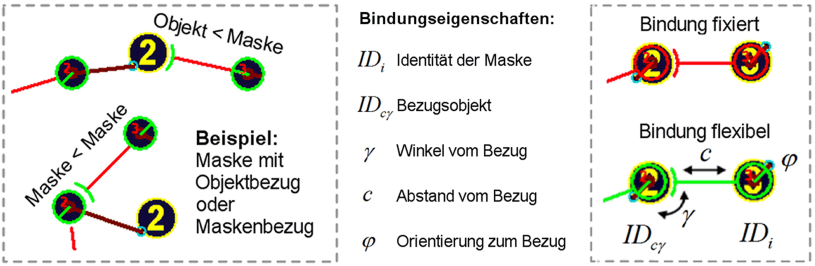
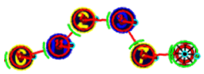
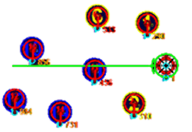


Abb. 4.14 Parameter der polaren Einfachbindung

Obwohl die zwischengelagerte Ebene der Formationsmaske ein Hilfsmittel zur Verteilung und Zuordnung von Informationen ist, stellt sie keine weitere Instanz im Steuerungsablauf dar. Im Vergleich zur direkten Kontrolle von Robotern ist der zusätzliche Aufwand der Transformation gering und wird dezentral im Zuge der ohnehin notwendigen Objektverwaltung berechnet (Tab. 4.4).

Bindungseigenschaft:	Bezug	Winkel	Abstand	Orientierung
$\pi_i = \{\varphi, ID_{SP}, \bar{c}, \bar{\gamma}\}$ 	Schwerpunkt Rahmenzentrum Einzelobjekt	konstant	konstant	konstant
	Formationsvorlage: Block Die Bindung eines Teilnehmers innerhalb der fixierten Formationsstruktur (Maske) kann frei definiert werden. Die Substitution der Gruppenformation durch ein virtuelles Roboterobjekt erleichtert die Handhabung.			
$\pi_i = \{\varphi, ID_{n-1}, \bar{c}, \gamma\}$ 	Einzelobjekt	flexibel	konstant	flexibel
	Formationsvorlage: Reihe Ein Roboterobjekt wird als Kopf der Formation festgelegt. Alle anderen Formationsteilnehmer folgen kettenförmig mit fixiertem Abstand, aber freiem Bindungswinkel der Spur des Kopfobjektes.			
$\pi_i = \{\varphi, ID_{c\gamma}, \Delta c, \Delta\gamma$ 	Schwerpunkt Einzelobjekt	dynamisch	dynamisch	flexibel
	Formationsvorlage: Rotte Die gespeicherte Ordnung der einzelnen Maskenpositionen kann durch Hindernisse temporär verformt und umstrukturiert werden. Es besteht jedoch die permanente Tendenz, in das Urschema zurückzukehren.			

Tab. 4.3 Ausgewählte Formationsstrukturen und deren Bindungsparameter

Um das konkrete Verhalten der Roboterobjekte, das Modell der Einfachbindung und den Umgang mit verschiedenen Konfigurationen der Bindungsparameter deutlicher darzustellen, werden die drei Formationsbeispiele aus Tab. 4.3 im Folgenden näher beschrieben. Der Freiheitsgrad dieser beispielhaften Formationsbindungen erhöht sich vom starren Block über die verfolgende Reihe bis zur unabhängigen Rottenstruktur.

4.7.6 Formationsstrukturen

Die nachfolgenden Bindungsstrukturen bauen in ihrer Funktionalität aufeinander auf. Es sind ausgewählte Entwicklungsbeispiele für die prinzipielle Anwendung des Grundmodells der objektbezogenen relativen Einfachbindung und sie werden ohne weitere Systematik beschrieben. Die Strukturen der relativen Mehrfachbindungen gleichen dieser Verfahrensweise, wenn auch der Berechnungsaufwand zur Lokalisierung der Maske steigt. Wird eine definierte Relativbewegung innerhalb der geordneten Roboter (Masken) gewünscht, so lassen sich mehrere hinterlegte Formationsschemata im sequentiellen Wechsel für eine willkürliche Umwandlung des Formationsverbandes anwenden.

Block (starr)
Die Formationsstruktur des Blockes dient als Beispiel eines starren Maskenmusters. Zwischen 2 und 15 Robotern können durch das System miteinander verknüpft werden. Fünf Roboter mit Einfachbindungen bieten dazu allein schon 5!=120 Bindungsvarianten mit fixierten Parametern. Ist das Maskengebilde nach Abb. 4.12 erfolgreich angelernt und konfiguriert, wird mit ihm in Form eines virtuellen Objektes gearbeitet. Der Koordinatenschwerpunkt der Formationsanordnung SP errechnet sich aus dem arithmetischen Mittel aller Teilnehmerpositionen.

$$SP(x,y) = \frac{1}{i}\left(\sum_{1}^{i} X_i(x,y)\right) \qquad (4.3)$$

Unter Vorgabe der Bindungsparameter aus Tab. 4.3 lässt sich am Pseudocode der Tab. 4.4 die Herleitung jeder Maske verfolgen. Die Zielpunktlage eines jeden Formationsteilnehmers wird anhand der Formationsmaske durch die vormals definierten Bindungsparameter innerhalb der Anordnung berechnet. Abhängig von der Art des gewählten Formationsverbunds orientieren sich die Bezugsobjekte entweder an den realen Objekten der erkannten Roboter oder ihren virtuellen Formationsmasken (siehe links in Abb. 4.14). Die Art des

Verbundes beeinflusst das Bewegungsverhalten in der dynamischen Übergangsphase, ist jedoch für die statische Zielkonstellation irrelevant, wenn reale und virtuelle Bezüge deckungsgleich liegen. Der empfohlene virtuelle Verbund garantiert der starren Blockstruktur ein exakteres Manövrieren. Mit fixierten Bindungsregeln ergeben trigonometrische Berechnungen die absoluten kartesischen Soll-Koordinaten und die Soll-Orientierung der Maske (Tab. 4.4). Dem teilnehmenden Formationsobjekt ist damit eine neue Zielpunktlage zugewiesen, welche die aktivierte Verhaltensaktion des Formationsteilnehmers kollisionsüberwacht am Roboter nachregelt.

Reihe (semi-flexibel)

Die Formationsstruktur der Reihe dient als Vorlage eines kopfgesteuerten hierarchischen Maskenmusters. Lagebasis des Formationsobjektes ist der Roboter, der die Führungsfunktion am Kopfende der Formationskette übernimmt (Tab. 4.3). Er zwingt nachfolgenden Formationsteilnehmern den eigenen Pfadverlauf auf, so dass eine enge Beziehung zum Pfadspeicher besteht. Das Vorgehen zur Herleitung der objektspezifischen Formationsmasken gleicht bis auf wenige Unterschiede dem Block. Berücksichtigt wird die Kopplung des repräsentierenden Formationsobjektes an einen Roboter und die zwanghafte Bahnverfolgung des Formationsobjektes durch nachfolgende Masken (Tab. 4.4). Damit sind die relativen Bezugspositionen, trotz der freigegebenen Flexibilität jeweiliger Bezugswinkel, eindeutig bestimmt und die Formation kann sich als kettenförmige Gelenkstruktur bewegen.

Rotte (semi-amorph)

Die Formationsstruktur der Rotte dient als Beispiel eines kopfgesteuerten amorphen Maskenmusters. Formationsobjekt ist ebenfalls ein Führungsroboter aus der teilnehmenden Objektmenge. Der Herleitung einzelner Maskenpositionen sind zwei Besonderheiten hinzugefügt, die in der gewollten Flexibilität und Verformbarkeit der Anordnung begründet sind. Zur Vermeidung von Selbstblockaden und Sprungeffekten im Maskenaufbau sind hier der Austausch zugeordneter Bindungsparameter zu den Roboterobjekten (Platzwechsel) und die Beschränkung der Maskenposition auf eine konstante nachführen-

```
Prozedur: Herleitung einer Maske innerhalb der Formationsgruppierung
Zweck: Herleiten der aktuellen Zielpunktkoordinaten eines
       Formationsteilnehmers von dessen relativen Bezug
Parameter: FormDefinition, VerbundParameter
Externer Zugriff: Objektverwaltung, Pfadpuffer, MaskenPuffer

        // Bestimmen des Bezuges
01. Wenn VerbundParameter = ‚an Maske gebunden'
02.       Dann initialisiere Maske durch Bezugsmaske
03.       Sonst initialisiere Maske durch Bezugsobjekt
       //Herleiten der Sollposition
04. Wenn FormDefinition = ‚bindungsbasiert'  //z.B. Block
05.     Wenn Bindungsregeln aktiv
06.        Dann Sollposition der Maske trigonometrisch initialisieren
07.           Sonst Sollposition der Maske vom akt. Objekt übernehmen
08. Wenn FormDefinition = ‚bezugsfolgend'   //z.B. Reihe
09.     Wenn Pfadlänge vom Führungsobjekt lang genug
10.         Dann Sollposition der Maske vom Pfad des Bezuges herleiten
11. Wenn FormDefinition = ‚defensiv'  //z.B. Rotte
       //Gradientendrift der Maske
12.    Wenn Abstand der Maske zur ursprünglichen Sollposition zu groß
13.    Dann Drift der Maske zur ursprünglichen Sollposition aktivieren
       //Relative Drift der Maske
14.     Wenn Abstand des Roboters zur Sollposition zu groß
15.        Dann Drift der Maske zum naheliegendsten Teilnehmer
       // Neugruppierung des Bezuges im Nahbereich
16.     Wenn Maske nahe Sollposition Und Gradientendrift aktiv
17.        Wenn weitere Position einer Maske in Driftrichtung unbe-
                setzt
18.          Dann Tausch des Bezuges mit der entfernteren Maske
19.            Drift der eigenen Maske auf die Sollposition zum Bezug
       // Kollisionsvermeidung der driftenden Maske
20. Wenn Objekt als Hindernis im Bewegungsbereich
21.     Dann im Sicherheitsabstand Und auf kürzesten Weg ausweichen
22. Wenn Maske als Hindernis im Bewegungsbereich
23.     Wenn Hindernis-Maske eine höhere Priorität besitzt
24.        Dann im Abstand Und auf kürzesten Weg ausweichen
25.     Wenn eigener Masken-Bezug im gegenseitigen Vergleich weiter
             entfernt
26.        Dann Tausch des Bezuges mit der Hindernis-Maske
       //Rahmen der geschlossenen Formation einhalten
28. Wenn Abstand zur Sollposition zu groß
29.    Dann Maske auf limitierten Abstand zum Bezug positionieren
Ergebnis: neu berechnete Lage der Masken im globalen Koordinatensystem
```

Tab. 4.4 Ableitung einer einfach gebundenen Maske in Pseudocode

de Geschwindigkeit (Drift) zu nennen. Kommt der Formationsverband mit starren Hindernissen in Berührung, so geben betroffene Masken ihre gebundene Anordnung defensiv auf und umfahren es unter dem Kriterium des minimalen Weges. Ist das Flächenangebot wieder ausreichend, so formen die Masken wieder die gespeicherte Ausgangskonstellation. Ziel ist es, die Formation flexibel und geschlossen durch Hindernisräume zu bewegen. Mit den reaktiven Regeln im Mechanismus der einzelnen Maskenableitungen (Tab. 4.4) können Hindernissituationen, wie sie Abb. 4.15 darstellt, durch Unterdrückung von Bindungsparametern selbständig bewältigt werden.

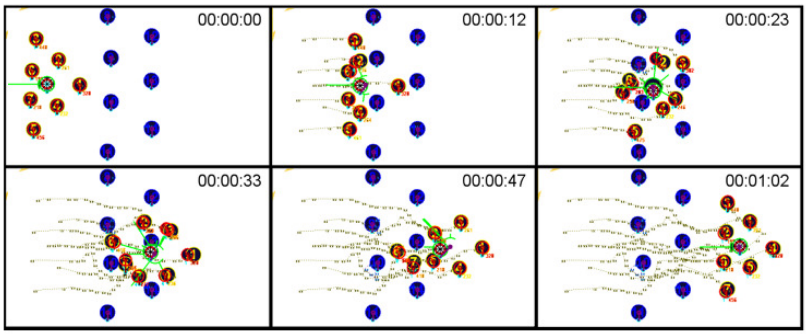


Abb. 4.15 Bildsequenz der Rottenformation im Hindernisraum

4.7.7 Aktionen mit Ballanwendung

Neben den Bemühungen um die Koordination von Robotern innerhalb einer Formation wurden auch die dynamischen Fähigkeiten in der Handhabung des Balles ausgebaut. In Übereinstimmung bekannter Architekturen der Small-Size-League ist die unterste Ebene der Aktionssteuerung in der Lage, mit den Robotern reaktive Verhaltensweisen umzusetzen.
Das Aktionsmodul soll Handlungen beinhalten, deren Funktionen sich übersichtlich automatisieren sowie variabel parametrisieren lassen, um in der gemeinsamen Zusammenarbeit komplexere Verhaltensmuster zu bilden.

Tab. 4.5 beschreibt einige Aktionsbeispiele des auf die wesentliche Sequenz reduzierten XBase-Quellcodes, der sich mit dem Finden der geeigneten Zielposition befasst. Nicht nur die enge Bindung an die Objektverwaltung, mit allen Zustandsvariablen der aktuellen Feldsituation, sondern auch der Zugriff der lagebasierten Steuerung auf einen eigenen Katalog von Pfadfunktionen erleichtert eine flexible Berechnung der Koordinaten.
Die Mehrzahl der Verhaltensweisen definiert sich allein über die Kriterien der Positionierung zum Zielobjekt in Verbindung mit dem Einsatz von Schuss- oder Dribblerautomatik. Nach anschließender Plausibilitätskontrolle werden die Koordinaten generell erst unter Prüfung diverser Toleranzkriterien zur Übertragung an den Roboter freigegeben. Das Setzen von Zielpositionen wurde schon im Abschnitt 4.7.3 für die Aktion der Pfadverfolgung (Tab. 4.2) betrachtet.

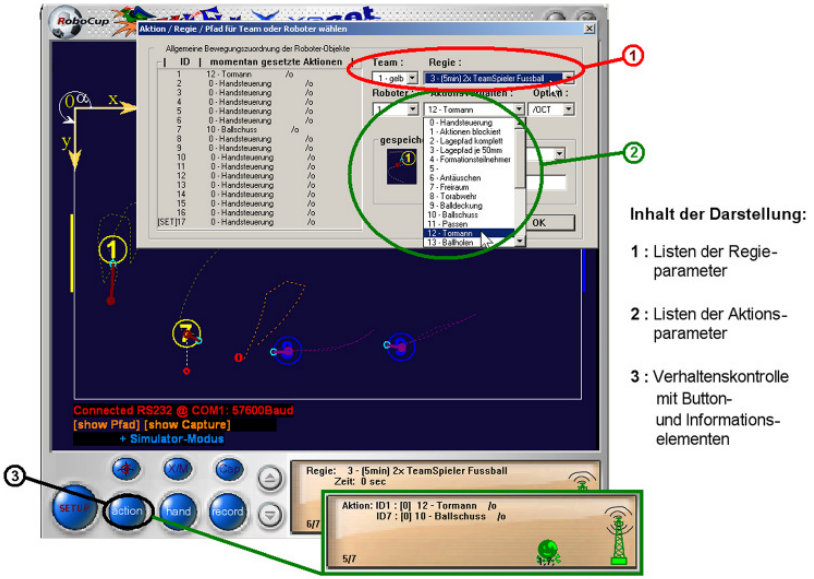


Abb. 4.16 Automatische Verhaltenskontrolle der Aktions- und Regieebene

```
Prozedur-Abschnitt: Hilfspfadplanung der Objektverfolgung
Parameter:  Zielabstand, Zielobjekt, Abwehrrichtung
05. Setze Sollposition in Abwehrrichtung mit Zielabstand vom Zielobj.
```

```
Prozedur-Abschnitt: Hilfspfadplanung der Freiraumsuche
Parameter:  Endpunkte der Abtastlinie, Zielpunkt
05. Solange Rasterpunkt auf der Abtastlinie im Feld  wiederhole
06.     Wenn Linie zwischen Rasterpunkt und Zielpunkt hindernisfrei
07.          Dann Inkrementiere Abtastintervall
08. Wenn Abtastintervall größer als Platzbedarf des Roboters
09.      Dann Setze Sollposition auf Abtastlinie mittig im Intervall
```

```
Prozedur-Abschnitt: Hilfspfadplanung für gezielten Ballschuss
Parameter:  Schusszielpunkt
05. Wenn Ball in Bewegung
06.      Dann Berechne Soll-Abfangpunkt mit linearem Vorhersagemodell
07.       Setze Sollposition mit Orientierung zum Ball auf Abfangpunkt
07. Sonst Setze Sollposition in Schussrichtung zum Ziel hinter Ball
08. Wenn Roboter hinter Ball in Schussrichtung
09.      Dann Aktiviere Lichtschranke für Schussautomatik
10.           Setze Sollposition auf Ballposition
```

```
Prozedur-Abschnitt: Hilfspfadplanung der Passannahme des Balls
Parameter:  Schusszielpunkt
05. Wenn Ball in Bewegung
06.     Dann Berechne nächsten Abfangpunkt mit Schussrichtung zum Ziel
07.      Setze Sollposition in Reflektionsorientierung auf Abfangpunkt
08. Sonst Bleibe Stehen
09. Wenn Ball im Nahbereich
10.      Dann Aktiviere Lichtschranke für Schussautomatik
```

```
Prozedur-Abschnitt: Hilfspfadplanung der Linienverteidigung
Parameter:  Linienendpunkte, Abwehrorientierung, Abwehrobjekt
05. Wenn Abstand zum Abwehrobjekt zu groß
06.     Dann Setze Sollposition mit Abwehrorientierung auf Mittelpunkt
07. Sonst Berechne Abfangpunkt des Abwehrobjektes auf der Linie
08.       Setze Sollposition mit Abwehrorientierung auf Abfangpunkt
```

```
Prozedur-Abschnitt: Hilfspfadplanung zum Ballholen
Parameter:  Demarkationslinie, Zielorientierung
05. Wenn Ball hinter Demarkationslinie
06.     Dann Aktiviere Lichtschranke für Dribblerautomatik
07.       Setze Sollposition mit Dribblerorientierung auf Ballposition
08. Sonst Setze Sollposition mit Zielorientierung und Abstand zum Ball
```

Tab. 4.5 Positions-Suchsequenzen aus Aktionen mit Ballanwendung in Pseudocode

Die gelisteten Aktionen in Tab. 4.5 nutzen dieses Grundschema bis zur Zeile 05. Durch variable Parameter ist beispielsweise die Aktion der Objektverfolgung nicht nur auf den Ball als Zielobjekt festgelegt und kann teamübergreifend angewandt werden. Neben der Menüoberfläche für die manuelle Regie- oder Aktionswahl, wie sie in Abb. 4.16 dargestellt ist, existiert keine hierarchische Ordnung innerhalb der Ebenen. Austausch, Erweiterungen und Neuordnungen sind durch eine Schnittstelle erleichtert. Aktionen der Verhaltenssteuerung werden durch definierte Kennzahlen aufgerufen und mittels Standardparameter initialisiert.

4.7.8 Regieanwendung

Das Aktionsmodul stellt eine Sammlung von Verhaltensweisen bereit, welche am Roboter erst in ihren Kombinationen und mit dem Wirken der übergeordneten Regieebene die typischen Rollenverteilungen (Tormann, Stürmer, Verteidiger etc.) erzeugen. Den Spielregeln entsprechend werden nach Auswertung von Lage- und Bewegungsparametern allen Teamrobotern die situationsrelevanten, logischen Verhaltensmuster ihrer Spielerrollen zugeordnet. Das prinzipielle Vorgehen ist an zwei manuell zu wählenden Demonstrationsbeispielen von Regiespielen ersichtlich (Abb. 4.16).
Die in Tab. 4.5 genannten Aktionen (Objektverfolgung, Freiraumsuche, Ballschuss, Passannahme, Linienverteidigung und Ballholen) bilden einen minimalen Grundstock zur Kontrolle eines 1vs1 Ping-Pong oder eines 2vs2 Schlagabtausches. Die Regieebene wird durch einen eigenen Thread repräsentiert, der zyklisch von allen Teammitgliedern die Aktionen ordert, welche je Roboter sofort in separaten Aktions-Threads verwirklicht werden (Tab. 4.6). Der Tormann eines Ping-Pong-Spiels wendet die Aktionen Linienverteidiger, Ballschuss, Objektfolgen und Ballholen an.
Die diskrete, fallbasierte Programmierung, welche das Verhalten der Roboter in sämtlichen Situationen vorherbestimmt, ist für komplexe Spielsituationen aufwendiger, doch auch mit einer begrenzten Anzahl grundlegender Verhaltensweisen beherrschbar [56].

```
Thread: Regie
Zweck: Überwachung und Ausführung von Teamverhalten, Spiel-
       koordinierung
Parameter: RegiePara
Externer Zugriff: Objektverwaltung, Pfadpuffer

01. Solange Thread aktiv wiederhole
02.    Falls RegiePara = ‚1vs1 Ping-Pong'
03.    Dann Wenn aktiver Spieler auf dem Feld
06.             Dann Initialisiere Kennung, Team und Feldparameter
07.                  Wenn Ball im Gegnerfeld
08.                      Dann Aktiviere Linienverteidigung
09.                  Sonst Wenn Ball im Nahbereich
10.                            Dann Aktiviere Ballschuss
11.                       Sonst Aktiviere Linienverteidigung
12.                       Wenn Ball im Aus
13.                       Dann Wenn Roboter-Ball-Abstand zu groß
14.                                 Dann Aktiviere Ballverfolgen
15.                       Sonst Aktiviere Ballholen
Ergebnis: automatischer Wechsel der Aktionen für Tormannbesetzung
```

Tab. 4.6 Regiekoordinierung ‚1vs1 Ping-Pong' in Pseudocode

Weitere Möglichkeiten bieten z.B. das trainergestützte Anlernen oder die Auswahl der Aktionen nach unscharfer Fuzzy-Logik. Die Qualität eines Spieles hängt jedoch nicht nur vom Zusammenspiel und der Manöverkoordinierung, sondern im Wesentlichen von der Flexibilität, Vielfalt und Dynamik der Roboteraktionen ab.

4.7.Simulator

Der Simulator ist ein Zusatzmodul, welches parallel zur realen Peripherie betrieben werden kann (siehe Abb. 4.2). Ausgehende Kommunikations-Befehlsfolgen zur Ansteuerung eines Zielpunktes aktivieren Berechnungen von Objektkoordinaten anhand einer konstanten Durchschnittsgeschwindigkeit. Das Modul speist diese Daten zur Visualisierung anstelle der Bildverarbeitung in die Objektverwaltung ein (z.B. Abb. 4.16). Auf eine realistisch simulierte physikalische Umgebung wurde bisher verzichtet. Die Hauptaufgabe des

Simulators besteht im kamera- und roboterlosen Testbetrieb für die transparente Analyse der Verhaltenssteuerung. Mit der aktiven Verhaltenskontrolle gestattet der Simulator die Darstellung der Befehls- und Datenströme im manuellen oder animierten Betriebsmodus.
Weil XBase generell auch als Server agieren kann, ist auf diese Weise selbst der Testbetrieb eines verteilten Systems möglich.

5 Zusammenfassung und Ausblick

5.1 Zusammenfassung Roboterkonstruktion LUKAS

Das Kapitel 3 gibt eine Erläuterung der hard- und softwaretechnischen Merkmale des Roboters LUKAS. Der Roboter wurde nach dem Reglement der RoboCup-Small-Size-League entworfen und über das Masterprogramm eines zentralen Steuercomputers funkgesteuert. Ausgehend von eigenen Vorstellungen, Grundforderungen und Richtlinien steht er aber im Gegensatz zu den Modellen anderer RoboCup-Mannschaften. Für den Roboter konnten Lösungen gefunden werden, die spieltaugliche Fahrleistungen durch einen verminderten Materialeinsatz erzielen.
Die Dimensionierung der Komponenten ermöglichte auch unter maximaler Belastung einen sicheren und robusten Betrieb und hielt bezüglich Schuss- und Fahrleistung sowie Gewichtsminimierung noch ungenutztes Potential und Spielraum für zukünftige Erweiterungen bereit. Die erreichte Kompaktheit und das notwendige Masse-Leistungs-Verhältnis beruhen im Wesentlichen auf der orthogonalen Achskonstellation. Die symmetrische Konstruktion besitzt ein geringes Gewicht und einen niedrigen Schwerpunkt. Spezielle Material- und Strukturlösungen für Leichtbau und Bauraumnutzung finden sich im Polyamid-Chassis, im motorintegrierenden Dribbler und im energieeffizienten Schussmechanismus. Die Detektierung des Balles erfolgt mittels Lichtschranke. Im Einsatz sind omnidirektionale Räder mit solider Rundlauf-, Traktions- und Stabilitätseigenschaft. Robust und kompakt integriert die spezielle elektronische Hauptplatine alle benötigten Steuerungskomponenten. Die galvanische Trennung der Steuer- und Lastseite garantiert Betriebssicherheit und längere Laufzeit.
Der Roboter setzt ein hierarchisches Konstruktionskonzept um, welches hohe Zuverlässigkeit, Flexibilität und Robustheit in verstärktem Maße durch technische Funktionsintegration einer minimalen Anzahl von Bauteilen realisiert. Nach Analyse der kinematischen und dynamischen Zusammenhänge konnte für den Roboter eine Reglerarchitektur entwickelt werden, die unter Berück-

sichtigung von Radschlupf und dynamischer Schwerpunktverlagerung die omnidirektionale Manövrierbarkeit und Bewegungspräzision gestattet. Die lokale Positionsreglung ist ein wesentliches Element des Controllerprogramms und zugleich Basis des Konzeptes einer bahngeführten Positionierung. Die spezielle Bahnsteuerung, u.a. nach dem Prinzip Lissajousscher Figuren, erlaubt die halbautomatische Verhaltenssteuerung des Roboters und eine Entlastung der Kommunikationskanäle.
Neben der Fähigkeit des Roboters, Positionierungsaufgaben auf Basis reduzierter Anweisungen selbständig durchzuführen, besteht die Möglichkeit, untergeordnete Befehlsschnittstellen für eine erweiterte externe Kontrolle zu nutzen. Das Programm der zentralen Computersteuerung kann auf diese Weise mit unabhängigen Steuerungs- und Koordinierungsansätzen arbeiten.

5.2 Zusammenfassung Mastersoftware XBase

Im Kapitel 4 werden die Ergebnisse der weiterentwickelten Mastersoftware XBase vorgestellt. Begründet auf einem dialogbasiertem Programmrahmen konnte durch modularen Ausbau ein einsatzfähiges Instrument zur Roboterkoordinierung in Verbindung mit einer anwenderfreundlichen Mensch-Maschine-Schnittstelle entstehen.
Das Programm XBase ist eine kompakte Einheit, in der die benötigten Elemente zur Bearbeitung der Anwendungsaufgaben enthalten sind. Um die zentrale Objektverwaltung wurde eine sternförmige Modulstruktur separater Funktionsbereiche aufgebaut, die den Zugriff, die Erweiterung und die Transparenz der Software erleichtert.
Die optionale Einbindung des Live-Bildes in Verbindung mit dem eingeblendeten Daten-Overlay der Objekterkennung ist ein besonderes Merkmal der Arbeitsoberfläche, welche, um neue Menü- und Displayfunktionen erweitert, die Einzel-, Gruppen- oder Teamsteuerung omnidirektionaler Roboter anhand diverser Aktionsverhalten realisiert. Die Programmoberfläche ermöglicht eine komfortable Bedienung und erleichtert die Handhabung mit Hilfe einer flexiblen Maussteuerung direkt am Objekt. Die im Hintergrund ablaufenden

Systemprozesse können innerhalb diverser Betriebsmodi (Server, Client, Offline) transparent überwacht werden.
Die Integration einer methodisch überarbeiteten Bildverarbeitung und Objekterkennung war eine notwendige Voraussetzung, um schnelle Objekte (bis 4m/s) in verwertbarer Qualität zu detektieren. Stabile Bildraten und geringer Berechnungsaufwand werden durch die Verwendung einer Look-Up-Tabelle der Maximum-Likelihood-Farbklassifizierung, Bildsegmentierung mit Zeilenkoinzidenzverfahren und ressourcenschonende Operationen erreicht. Die Genauigkeit der akquirierten Daten steigerte sich gegenüber der Vorläuferversion durch den Beitrag eines Butterfly-Robotertrikots.
Besonders die dynamischen Verhaltensweisen mit Ballanwendung profitierten in der hybriden Verhaltensteuerung von der Zuverlässigkeit der sensorischen Erfassung.
Ein geeigneter Trainingseinstieg wurde mit verschiedenen Formationsvarianten zur koordinierten Kontrolle einer Robotergruppe gefunden Der strukturelle Aufbau der Formation ist durch einen Datensatz polarer Bindungsparameter beschrieben. Das Modell gestattet durch individuelle Freigabe von Freiheitsgraden einen Übergang vom starren zum flexiblen Verband. Anhand der Beispielformationen Block, Reihe und Rotte ist die Flexibilität und Wandlungsfähigkeit der Struktur dargestellt. Die Substitution der Gruppe durch ein virtuelles Formationsobjekt erlaubt z.B. auch die Pfadverfolgung einer Gruppe analog der Art eines einzelnen Roboters. Weil die Verhaltensteuerung des Systems noch weitgehend fragmentarisch ist, bilden die Aktionen der Formations- und Ballanwendung den tragfähigen Grundstein zum Ausbau einer selbständigen Spielregie- und Agentenkontrolle. Bisherige Beispiele von Aktions- und Spielplanungen mit Ballanwendung dienten dem Zweck, die Funktionsweise der entworfenen Struktur zu beweisen und in realer Umgebung oder im Simulator Erfahrungen zu sammeln.
Die automatische Ansteuerung der Roboter basiert hauptsächlich auf der Anwendung der bahngeführten Positionierung, um die Bandbreite der Funkübertragung auszunutzen.
Das System XBase kann sowohl als Multi-Roboter-System unter der Kontrolle einer Verhaltenssteuerung wie auch als verteiltes Multi-Agenten-System mit

mehreren parallelen Verhaltenssystemen fungieren. Mit geringem Bedienungsaufwand lässt sich im Server-Client-Betrieb ein verteiltes System zur Steuerung auch mehrerer Gruppen oder diverser Einzelobjekte installieren.

5.3 Ausblick

Es wurde zu Beginn gerade auch deshalb auf die vielseitigen Ausprägungen der Robotik eingegangen, weil das bearbeitete Projekt ebenso differenzierte Entwicklungsrichtungen unterstützen kann.
Grundsätzlich liegt noch großes Potential im Ausbau der zentralen Software. Die Komponenten der Verhaltensteuerung, die sich mit der Koordinierung der Ballanwendung befassen, nutzen bisher nicht die volle Leistungsfähigkeit des Systems. Für die Realisierung eines RoboCup-konformen Ballspiels bedarf es deshalb nicht nur der Implementierung eines strategischen Agenten, sondern auch einer grundlegenden Erweiterung bzw. Vervollständigung benötigter Aktionen und einer breiteren Variation von Regiespielen und Aufstellungen. Sie sind erforderlich, um dem koordinierenden Agenten ein minimales Spektrum zwischen aggressiven, moderaten und defensiven Spielen zur Auswahl zu geben.
Trotz der Vorbildfunktion bekannter Mannschaften erfordert die eigene Entwicklungsstrategie die Suche nach spezifischen, innovativen Lösungen mit dem Fokus auf einem günstigen Aufwand-Nutzen-Verhältnis. Das System wird auch zukünftig für die Ingenieursausbildung von Maschinenbaus bzw. Mechatronik eingesetzt. Mittelfristiges Ziel kann gleichwohl ein Verhaltenssystem sein, das zusätzlich zu den Basiselementen ein Schiedsrichtermodul und einen simulatorgestützten Trainer enthält.
Andere Perspektiven bietet der davon unabhängige Ausbau der XBase-Formationssteuerung. Kurzfristig lässt sich das Modell auf eine Zweifachbindung umstellen und die übergeordnete Kontrolle mehrfacher Gruppierungen verbessern. So ließen sich die aktuelle Lokomotions- und Transportsituationen zu ebenen Szenarien erweitern, welche für eine Verrichtung von Tätigkeiten

geeignet sind (z.B. zangenartiges Ergreifen und Handhaben von Gegenständen).

Ergebnisse einer effektiven Formationskontrolle können auf realistische Anwendungen der Praxis übertragen werden. Würde man beispielsweise die globale Bildverarbeitung auf ein GPS (Global Positioning System) bzw. eine Laser-Überwachung und die Funktionalität der Roboter auf unbemannte Feldtraktoren übertragen, könnten Rotten-Formationen im größeren Maßstab vorrücken, so wie es für die riesigen Kornfelder der Ukraine vom Traktoren-Hersteller Fendt als Zukunftsvision gesehen wird[16].

Ein anderes Ziel besteht zukünftig in der automatischen Generierung von missionsabhängig angepassten Formationsstrukturen. Nach getaner Arbeit erfolgt der Formationswechsel zur Reihe und der Traktoren-Tross könnte selbständig zur Basisstation zurückkehren.

Dass besonders diese Art der Formation in Zukunft an Bedeutung gewinnt, zeigt die wachsende Vielzahl an Fahrerassistenzsystemen für automatische Abstands- und Temporegelung (engl.: adaptive cruise control) in aktuellen Fahrzeugen [90]. Assistierende Autos können den Fahrer auf langen Autobahnfahrten komfortabel entlasten und tragen indirekt dazu bei, den Fließprozess durch Verkettung zu homogenisieren. Mit massenhaftem und koordiniertem Einsatz ist somit nicht nur der Stress-, sondern auch der Stauvermeidung gedient. Parallel dazu bieten sich aber auch auf der Seite der eigenen Roboterhardware noch wesentliche Möglichkeiten für Weiter- und Neuentwicklungen. Langfristig zielt dies nicht unbedingt in die Richtung einer noch leichteren, schnelleren und schussstärkeren Roboterversion, sondern vielmehr auf die engere physische Kopplung der kooperierenden Robotermodule.

Die Intensivierung der eigenen Formationsrobotik könnte ebenfalls eine Einstiegschance in die Welt amorpher modularer Robotersysteme eröffnen. Obwohl beispielsweise die radgetriebenen S-Bot-Schwarmroboter aus Tab. 2.1 einen stabilen Greifarm besitzen, der dem verkoppelten System sogar das Treppensteigen erlaubt, sind räumlich symmetrische Trägersysteme für homogene oder nicht homogene modulare Systeme wesentlich vielseitiger.

[16] http://www.stern.de/wirtschaft/news/ unternehmen/Martin-Richenhagen (Stand 06/2008)

Tensegrity-Module sind hierzu eine zweckmäßige Alternative zu herkömmlichen Multi-Mode-Modulen in Gestalt von autonomen Kreuzgelenken (z.B. M_TRAN, siehe Tab. 2.1). Für geschlossene Gruppierungen von Modulen würden z.B. sechsstrebige ikosaedrische Tensegrity-Strukturen in Frage kommen. Diese lassen sich formbedingt als räumlicher Quasikristall anordnen [91], sind dabei jedoch so flexibel, dass sie in Relativbewegung untereinander Verknüpfungen eingehen und auflösen können.

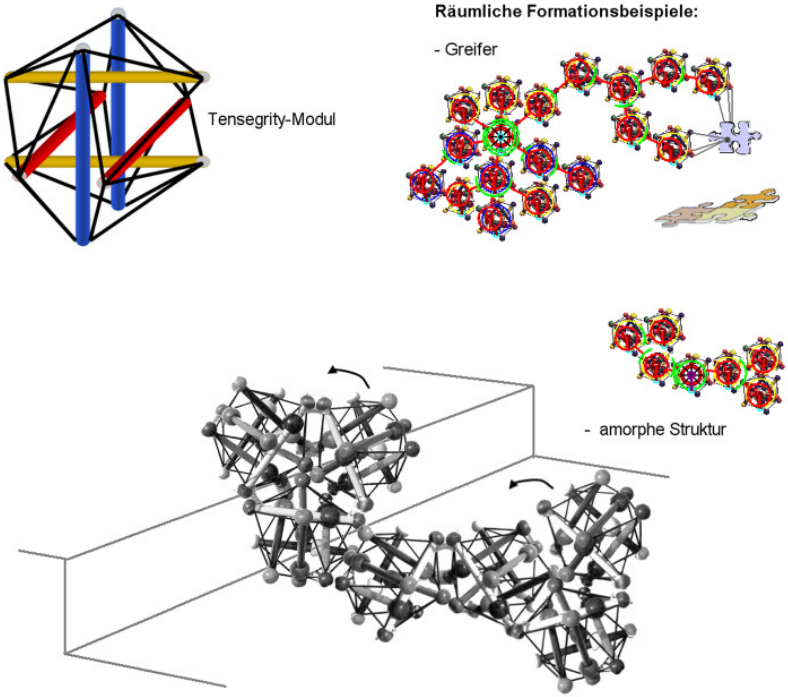


Abb. 5.1 Amorphe Verbundstrukturen ikosaedrischer Tensegrity-Module

Abb. 5.1 zeigt eine Konzeption für prinzipielle Verknüpfungen von polymorphen Modulen auf Basis sechsstrebiger Tensegrity. Die dynamisch verformbaren Grundmodule könnten aus der Bestückung umgeordneter elektronischer Komponenten des Roboters LUKAS hervorgehen, da ihnen schon der energieautarke mobile Einsatz im Verband und die Möglichkeit der funkkommunizierenden, variablen Regelung von drei (Raum-)Achsen gemeinsam ist.
Konstruktion, Koppelmechanismen und die Transformations- bzw. Verhaltenssteuerung einsatzfähiger homogener oder heterogener Konstellationen bieten dementsprechend Potential für weiterführende Untersuchungen auf diesem Gebiet.

Literaturverzeichnis

1. Murphy, R.R., *Introduction to AI Robotics*. 2000: MIT Press.
2. Gilbert, N., Troitzsch, K. G., *Simulation for the Social Scientist*. 2005: McGraw-Hill Publ. Comp.
3. Resnick, M., *Turtles, Termites, and Traffic Jams*. 1997: MIT Press.
4. Randow, G.v., *Roboter-Unsere nächsten Verwandten*. 1998.
5. Lawitzky, G., Buss, M. (Eds.) et al., *Service Roboter*. Schwerpunktthemenheft der Zeitschrift it-Information Technology. 2007, München: Oldenbourg Verlag.
6. Kitano, H., *Research program of RoboCup*. Applied Artificial Intelligence. 1998.
7. Röllich, T., *RoboCup: Gesamtkonzeption, Elektronik und Bilderkennung*. Diplom-arbeit. 2000: TU Ilmenau, Fakultät Maschinenbau.
8. Abraham, S., *Entwicklung eines mobilen Roboters für ein RoboCup Team*. Diplom-arbeit. 2001: TU Ilmenau, Fakultät Maschinenbau.
9. Zimmermann, K., Weiß, M., Jahn, M., Braunschweig, M., *Aufbau eines RoboCup-Teams an der TU Ilmenau*, in *der Vortragsreihe der Gesellschaft für Informatik, Regionalgruppe Deutsches Eck Universität Koblenz* 2007, (unveröffentliches Manuskript).
10. Jahn, M., *Master-Steuerung für ein Team der RoboCup Small-Size-League*. Diplom-arbeit. 2005: TU Ilmenau, Fakultät Maschinenbau.
11. Tsuda, S., Zauner, K.-P., Gunji, Y.-P., *Robot Control with Biological Cells*. Proceedings of Sixth International Workshop on Information Processing in Cells and Tissues. 2005, York: St. William's College. 202-216.
12. Raibert, M., *Legged Robots that Balance*. 1986, Cambridge, MA: MIT-Press.
13. Rieffel, J., Stuk, R., Lipson, H., *Locomotion of a Tensegrity Robot via Dynamically Coupled Modules*. Proceedings of the International Conference on Morphological Computation, 2007.

14. Murata, S., Kurokawa, H., *Self-Reconfigurable Robot: Shape-Changing Cellular Robots Can Exceed Conventional Robot Flexibility*. IEEE Robotics & Automation Magazine 2007.
15. Studer, G., Lipson, H., *Spontaneous emergence of self-replicating, competing cube species in physical cube automata*. GECCO, 2005.
16. Zykov, V., Mytilinaios, E., Adams, B., Lipson, H., *Self-reproducing machines*. Nature, 2005. 435(7038): p. 163-164.
17. Shimizu, M., Ishiguro, A., Kawakatsu, T., *Slimebot: A Modular Robot That Exploits Emergent Phenomena*. IEEE International Conference on Robotics and Automation, 2005: p. 2982- 2987.
18. White, P., Zykov, V., Bongard, J., Lipson, H., *Three Dimensional Stochastic Reconfiguration of Modular Robots*. Proceedings of Robotics Science and Systems. 2005, Cambridge, MA: MIT.
19. Shen, W.-M., Krivokon, M., Chiu, H., Everist, J., Rubenstein, M., Venkatesh, J., *Multimode Locomotion for Reconfigurable Robots*. Autonomous Robots, 2006. 20(2): p. 165-177.
20. Kamimura, A., Kurokawa, H., Yoshida, E., Tomita, K., Kokaji, S., Murata, S., *Distributed Adaptive Locomotion by a Modular Robotic System, M-TRAN II (From Local Adaptation to Global Coodinated Motion using CPG Controllers)*. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2004: p. 2370-2377.
21. Peters, J.F.H., C.; Ramanna, S., *Reinforcement learning in swarms that learn*. IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2005: p. 400-406.
22. Hou, F., Shen, W.-M., *Distributed, Dynamic, and Autonomous Reconfiguration Planning for Chain-Type Self-Reconfigurable Robots*. IEEE Intl. Conf. on Robotics and Automation, 2008.
23. Paul, C., Valero-Cuevas, F.J., *Redundancy in the Control of Robots with Highly Coupled Mechanical Structures*. International Conference on Intelligent Robots and Systems, 2005: p. 802-808.
24. Paul, C., Valero-Cuevas, F. J., Lipson, H., *Design and Control of Tensegrity Robots*. IEEE Transactions on Robotics, 2006. 22(5): p. 944-957.

25. Fuller, R.B., *Synergetics-Explorations in the Geometry of Thinking*. 1975: MacMillan Publishing Co.
26. Baldwin, J., *Bucky works, Buckminster Fuller´s ideas for today*. 1996, New York: Wiley.
27. Paul, C., Lipson, H., Valero-Cuevas, F. J., *Gait Production in a Tensegrity Based Robot*. Proceedings of 12th International Conference on Advanced Robotics (ICAR), 2005: p. 216-222.
28. Haugeland, J., *Artificial Intelligence: The Very Idea*. 1985, Cambridge, MA: MIT Press.
29. McKerrow, P.J., *Introduction to Robotics*. 1991, Sydney: Addison-Wesley Publishing Co.
30. Goldberg, K., Halperin, D., Latombe J.C., Wilson, R.H., *Algorithmic Foundations of Robotics*. 1995, Wellesley, MA: A.K. Peters Ltd.
31. Arkin, R.C., *Behavior Based Robotics*. 1998, Cambridge, MA: MIT-Press.
32. Brooks, R.A., *Elephants don't play chess*. Robotics and Autonomous Systems, 1990. 6: p. 3-15.
33. Mackworth, A.K., *On seeing robots*. Technical Report TR-93-05. 1993: University of British Columbia.
34. Burkhard, H.D., *Einführung in die Agenten-Technologie*, in *it + ti - Informationstechnik und Technische Informatik, Heft 4*. 1998, R. Oldenbourg Verlag.
35. Müller, J., *Kontrollarchitekturen für autonome kooperierende Agenten*. it + ti - Informationstechnik und Technische Informatik, Heft 4. 1998: R. Oldenbourg Verlag.
36. Bruce, J.R., *Real-Time Motion Planning and Safe Navigation in Dynamic Multi-Robot Environments*. PhD thesis. 2006, Pittsburgh, PA: Carnegie Mellon University.
37. Bruce, J., Zickler, S., Licitra, M., Veloso, M., *CMDragons 2007 Team Description*. 2007, Pittsburgh, PA: Carnegie Mellon University.
38. Wiesel, F., *Steuerung und Kontrolle von omnidirektionalen Fussballrobotern*. Diplomarbeit. 2006: Freie Universität Berlin.
39. Gloye, A., *Lernmethoden für autonome mobile Roboter*. Dissertation. 2005: Freie Universität Berlin.

40. Simon, M., *Ein robustes Echtzeit-Vision-System für die Robocup F180 SmallSize Liga*. Diplomarbeit. 2006: Freie Universität Berlin.
41. *12. Workshop Farbbildverarbeitung*. Zeitschriftenreihe des Zentrums für Bild- und Signalverarbeitung (ZBS) e.V. 2006, Ilmenau.
42. Shi, J., Malik, J., *Normalized Cuts and Image Segmentation*. Proc. of IEEE Conf. on Comp. Vision and Pattern Recognition. 1997, Puerto Rico.
43. Jähne, B., *Digitale Bildverarbeitung*. 2005, Berlin, Heidelberg: Springer
44. Fahrmeir, L., Hamerle, A., Tutz, G., *Multivariate statistische Verfahren*. 1996, Berlin: Walter de Gruyter & Co.
45. Christianini, N., Shawe-Taylor, J., *Introduction to Support Vector Machines and other kernel-based learning methods*. 2000: Cambridge University Press.
46. Steinmüller, J., *Bildanalyse: Von der Bildverarbeitung zur räumlichen Interpretation von Bildern*. 2008, Berlin: Springer.
47. Bruce, J., Balch, T., Veloso, M., *Fast and inexpensive color image segmentation for interactive robots*. Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '00), 2000. **3**: p. 2061-2066.
48. Bruce, J., Veloso, M., *Fast and Accurate Vision-Based Pattern Detection and Identification*. Proceedings of ICRA'03, the 2003 IEEE International Conference on Robotics and Automation, 2003.
49. Simon, M., Rojas, P., Tenchio, O., *Parabolic flight recontruction from multiple images from a single camera in general position*. Proceedings of The 10th RoboCup International Symposium, 2006: p. 183-193.
50. Laue, T., Röfer, T., *A Behavior Architecture for Autonomous Mobile Robots Based on potential Field*. 8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences), 2005.
51. Guldner, J., *Lokale Kollisionsvermeidung für mobile Roboter mittels künstlicher harmonischer Dipol-Potentiale*. at - Automatisierungstechnik, 1997. 45(1): p. 24-35.
52. Koch, M., Kuhn, T., Wernstedt, J., *Fuzzy Control*. 1996, München: R. Oldenbourg Verlag GmbH.

53. Kuffner, J., LaValle, S.M., *RRT-Connect: An effcient approach to single-query path planning.* Proceedings of the IEEE International Conference on Robotics and Automation, 2000.
54. Nilsson, N.J., *Principles of Artificial Intelligence.* 1982, Berlin: Springer.
55. Kim, T., Seo, Y., Hong K.-S., *Physics-based 3d position analysis of a soccer ball from monocular image sequences.* Sixth International Conference on Computer Vision, 1998: p. 721-726.
56. Browning, B., Bruce, J., Bowling, M., Veloso, M., *STP: Skills tactics and plans for multi-robot control in adversarial environments.* Journal of System and Control Engineering, 2005.
57. Bowling, M., Browning, B., Veloso, M., *Plays as effective multiagent plans enabling opponent-adaptive play selection.* Proceedings of International Conference on Automated Planning and Scheduling (ICAPS'04), 2004.
58. Bruce, J., Veloso, M., *Real-time randomized path planning for robot navigation.* Proceedings of the IEEE Conference on Intelligent Robots and Systems (IROS), 2002.
59. Brock, O., Khatib, O., *High-speed navigation using the global dynamic window approach.* Proceedings of the IEEE International Conference on Robotics and Automation, 1999.
60. Bruce, J., Veloso, M., *Safe multi-robot navigation within dynamics constraints.* Proceedings of the IEEE, Special Issue on Multi-Robot Systems, 2006.
61. Behnke, S., Egorova, Gloye, A., Rojas, R., Simon, M., *Predicting away robot control latency.* RoboCup, 2003: p. 712-719.
62. Brooks, R.A., *A robust layered control system for a mobile robot.* IEEE Journal of Robotics and Automation, 1986. RA-2: p. 14-23.
63. Jaeger, H., Christaller, T., *Dual dynamics: Designing behavior systems for autonomous robots*, in *Proceedings International Symposium on Artificial Life and Robotics (AROB '97).* 1997: Japan. p. 76-79.
64. Behnke, S., *Local Multiresolution Path Planning.* RoboCup. 2003: Springer.

65. Jahn, M., Weiß, M., Zimmermann, K., Braunschweig, M., Walkling, P., Lerm S., *RoboCup-Player "LUKAS" – an object of interdisciplinary research and a benchmark problem in mechatronics*, in *53. Internationales Wissenschaftliches Kolloquium*. 2008, Technische Universität Ilmenau.
66. Pahl, G., *Konstruktionslehre: Grundlagen erfolgreicher Produktentwicklung*. 2005, Berlin: Springer.
67. Krause, W., *Konstruktionselemente der Feinmechanik*. 2004, München: Hanser.
68. Bottenbruch, L., Binsack, R., *Polyamide, Kunststoff-Handbuch Band 3/4: Technische Thermoplaste*. 1998: Hanser.
69. Schwarz, O., *Kunststoffkunde*. 1988: Vogel Buchverlag.
70. Landsmann, P., *Konstruktion des Schussmechanismus eines Small-Size-Roboters*. Studienarbeit. 2006: TU Ilmenau, Fakultät Maschinenbau.
71. *Datasheet: BlackKnight Solenoids - Series 121 Tubular*. 2004, Newmarket: BLP Components Ltd.
72. Feuerle, M., *Blide – Mange – Trebuchet. Technik, Entwicklung und Wirkung des Wurfgeschützes im Mittelalter. Eine Studie zur mittelalterlichen Innovationsgeschichte*. 2005: GNT-Verlag.
73. Grabowiecki, J., *Vehicle Wheel*. US Patent 1,303,535. 1919.
74. *Minimodule TQM164C Hardware - Manual*. 2000, Wessling: TQ Components GmbH.
75. *Data transmission by the WIZ-434-SML-IA/5V Instruction Manual*. 2002, Modigliana: Aurel S.p.A.
76. Pin, F.G., Killough, S. M., *A new family of omnidirectional and holonomic wheeled platforms for mobile robots*. IEEE Transactions on Robotics and Automation, 1994. 10(2): p. 480-489.
77. Siegwart, R., Nourbakhsh, I.R., *Introduction to Autonomous Mobile Robots*. 2004: MIT-Press.
78. Füchsel, D., *Simulation des Fahrverhaltens von mobilen Robotern*. Diplomarbeit. 2006: TU Ilmenau, Fakultät Maschinenbau.
79. Zimmermann, K., Zeidis, I., Behn, C., *Mechanics of Terrestrial Locomotion: With a Focus on Non-pedal Motion Systems*. 2009, Heidelberg: Springer.

80. Rojas, P., GloyeFörster, A., *Holonic Control of a robot with an omni-directional drive*. KI, 2006. 20(2): p. 12-17.
81. Landsmann, P., *Analyse und Weiterentwicklung der Mikrocontrollersoftware für den Spieler der Robocup Small-Size-League*. Projektarbeit. 2007: TU Ilmenau, Fakultät Maschinenbau.
82. Ziegler, J.G., Nichols, N. B, *Optimum settings for automatic controllers*. Trans. ASME, 1942. 64: p. 759-768.
83. Grimsehl, E., *Lehrbuch der Physik*. 2000, Leipzig: Teubner.
84. Zimmermann, K., Jahn, M., Weiß, M., Braunschweig, M., Rieß, T., *Entwurf einer bahngeführten Positionierung als Basis für die Lokomotion omni-direktionaler Roboter*, in *Autonome Mobile Systeme*. 2007: Kaiserslautern.
85. Liu, Y., Wu, X., Zhu, J. J., Lew, J., *Omni-directional mobile robot controller design by trajectory linearization*. Proceedings of the American Control Conference, 2003: p. 3423-3428
86. Moore, K.L., Flann, N.S., *A Six-Wheeled Omnidirectional Autonomous Mobile Robot*. IEEE Control Systems Magazine 2000. 20(6): p. 53-66.
87. Purwin, O., D'Andrea, R., *Trajectory Generation and Control for Four Wheeled Omnidirectional Vehicles*. Robotics and Autonomous Systems 2006. 54(1): p. 13-22.
88. Wooldridge, M., *Intelligent Agents*. Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, ed. G. Weiss. 1999, Cambridge, MA: MIT-Press. 27-77.
89. Lerm, S., *Entwurf und Implementation einer neuen Version der Bildverarbeitung für die RoboCup Small Size League*. Diplomarbeit. 2008: TU Ilmenau, Fakultät Maschinenbau.
90. Bosch, R., *Autoelektrik Autoelektronik*. 2007, Wiesbaden: Vieweg & Sohn Verlag.
91. Motro, R., *Tensegrity: Structural Systems for the Future* 2006: Elsevier Science & Technology.