Lecture Notes in Control and Information Sciences

422

Krzysztof R. Kozłowski (Ed.)

Robot Motion and Control 2011

Lecture Notes
in Control and Information Sciences 422

Editors: M. Thoma, F. Allgöwer, M. Morari

Krzysztof R. Kozłowski (Ed.)

Robot Motion and Control 2011

Springer

Editor

Krzysztof R. Kozłowski
Poznań University of Technology
Institute of Control and Systems Engineering
ul. Piotrowo 3a
60-965 Poznań
Poland
E-mail: krzysztof.kozlowski@put.poznan.pl

ISBN 978-1-4471-2342-2 e-ISBN 978-1-4471-2343-9

DOI 10.1007/978-1-4471-2343-9

Lecture Notes in Control and Information Sciences ISSN 0170-8643

Library of Congress Control Number: 2011941381

Typeset by Scientific Publishing Services Pvt. Ltd., Chennai, India.

Printed on acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

Preface

The main objective of publishing this collection of papers is to present the most recent results concerning robot motion and control to the robotics community. Thirty one original works have been selected out of 39 papers submitted to the Eighth International Workshop on Robot Motion and Control (RoMoCo'11), Bukowy Dworek, Poland, June 15 to 17, 2011. This Workshop is the eighth in the series of the RoMoCo Workshops (the previous ones were held in 1999, 2001, 2002, 2004, 2005, 2007, and 2009). It is an internationally recognized event, technically co-sponsored by the IEEE Robotics and Automation Society and the IEEE Control Systems Society. The Workshop is organized by the Chair of Control and Systems Engineering of the Poznań University of Technology in Poland. The selected papers have gone through a rigorous review procedure with each receiving three reviews. Based on the reviewers' comments all of the papers were corrected and finally accepted for publication in the *Lecture Notes in Control and Information Sciences* series.

Interest in robot motion and control has remarkably increased over recent years. Novel solutions of complex mechanical systems such as industrial robots, mobile robots, walking robots and their applications are the evidence of significant progress in the area of robotics.

This book consists of seven parts. The first part deals with control and localization of mobile robots. The second part is dedicated to new control algorithms for legged robots. In the third part control of robotic systems is considered. The fourth part is devoted to motion planning and control of nonholonomic systems. The next part deals with new trends in control of flying robots. Motion planning and control of manipulators is the subject of the sixth part. Finally, the last part deals with rehabilitation robotics.

In this year's edition control of various nonholonomic systems seems to be the research area of most interest to the robotics community. We strongly believe that RoMoCo Workshop brings new ideas and recent control techniques which are currently used in research laboratories and in industrial applications.

The book is addressed to Ph.D. students of robotics and automation, informatics, mechatronics, and production engineering systems. It will also be of interest to scientists and researchers working in the aforementioned fields.

I am grateful to the three invited distinguished plenary speakers: Professor Silvère Bonnabel, from Centre de Robotique, Mathématiques et Systèmes, Mines ParisTech, France, Professor Aaron Ames, from the Department of Mechanical Engineering, Texas A&M University, USA, and Professor Dušan M. Stipanović, from the Department of Industrial and Enterprise Systems Engineering and Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, USA, for accepting my invitation and delivering outstanding presentations during the workshop.

I would like to express my deep thanks to Professor Dr. Frank Allgöwer for his support in accepting our proposal and running this project. I am very grateful to Mr K. Romanowski for his suggestions concerning the English of the papers. I am also very grateful to Dr W. Wróblewski for his help and patience and typesetting this book. Finally, I would like to express my thanks to all reviewers, who did very hard work in evaluating all papers which appear in this book. I appreciate their help and patience in communicating through my office with the authors.

Mr O. Jackson and Mr A. Doyle, our editors at Springer, are gratefully acknowledged for their encouragement in pursuing this project.

Poznań, Poland
September, 2011 *Krzysztof R. Kozłowski*

Contents

Part IV: Motion Planning and Control of Nonholonomic Systems

Part VII: Rehabilitation Robotics

List of Contributors

Wojciech Adamski
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: wojciech.adamski@put.poznan.pl

Leszek Ambroziak
Department on Automatics and Robotics, Białystok University of Technology, ul. Wiejska 45 c, 15-351 Białystok, Poland
e-mail: leszek.ambroziak@gmail.com

Aaron D. Ames
Department of Mechanical Engineering, Texas A&M University, 3123 TAMU, College Station, TX 77843-3123, USA
e-mail: aames@tamu.edu

Konrad Banachowicz
Institute of Control and Computation Engineering, Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland

Marek Banaszkiewicz
Space Research Centre, Polish Academy of Sciences, ul. Bartycka 18a, 00-716 Warsaw, Poland
e-mail: marekb@cbk.waw.pl

Dominik Belter
Institute of Control and Information Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: dominik.belter@put.poznan.pl

Yasmina Bestaoui
Laboratoire IBISC, Université d'Evry Val d'Essonne, 91025 Evry Cedex, France
e-mail: bestaoui@iup.univ-evry.fr

Leonardo Bobadilla
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
e-mail: bobadil1@uiuc.edu

Silvère Bonnabel
Centre de Robotique, Mathématiques et Systèmes, Mines ParisTech,
60 Bd Saint-Michel, 75272 Paris cedex 06, France
e-mail: silvere.bonnabel@mines-paristech.fr

Rafał Chojecki
Institute of Automatic Control and Robotics, Warsaw University of Technology,
ul. Św. A. Boboli 8, 02-525 Warsaw, Poland
e-mail: r.chojecki@mchtr.pw.edu.pl

Mateusz Cholewiński
Institute of Computer Engineering, Control, and Robotics, Wrocław University
of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: mateusz.cholewinski@pwr.wroc.pl

Rüdiger Dillmann
FZI Research Center for Information Technology, Haid-und-Neu-Str. 10-14,
D-76131 Karlsruhe, Germany
e-mail: dillmann@fzi.de

Ignacy Dulęba
Institute of Computer Engineering, Control, and Robotics, Wrocław University
of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: ignacy.duleba@pwr.wroc.pl

Didier Dumur
Automatic Control Department, Supeléc Systems Sciences (E3S), Gif-sur-Yvette,
France
e-mail: didier.dumur@supelec.fr

Mirosław Galicki
Faculty of Mechanical Engineering, University of Zielona Góra, Zielona Góra,
Poland and Institute of Medical Statistics, Computer Science and Documentation,
Friedrich Schiller University, Jena, Germany
e-mail: m.galicki@ibem.uz.zgora.pl

Zdzisław Gosiewski
Department on Automatics and Robotics, Białystok University of Technology,
ul. Wiejska 45 c, 15-351 Białystok, Poland
e-mail: gosiewski@pb.bialystok.pl

Katrina Gossman
Department of Computer Science, University of Illinois at Urbana-Champaign,
Urbana, IL 61801, USA
e-mail: kgossma2@uiuc.edu

Grzegorz Granosik
Institute of Automatic Control, Technical University of Łódź,
ul. Stefanowskiego 18/22, 90-924 Łódź, Poland
e-mail: granosik@p.lodz.pl

Yusuke Hayashi
Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya, Aichi 4668555, Japan

Przemysław Herman
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: przemyslaw.herman@put.poznan.pl

Tatsuya Hirano
Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya, Aichi 4668555, Japan

Masahide Ito
Seikei University, 3-3-1 Kichijoji-kitamachi, Musashino-shi, Tokyo 180-8633, Japan
e-mail: masahide_i@st.seikei.ac.jp

Jacek Jagodziński
Institute of Computer Engineering, Control, and Robotics, Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: jacek.jagodzinski@pwr.wroc.pl

Mariusz Janiak
Institute of Computer Engineering, Control, and Robotics, Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: mariusz.janiak@pwr.wroc.pl

Marcin Kaczmarski
Institute of Automatic Control, Technical University of Łódź, ul. Stefanowskiego 18/22, 90-924 Łódź, Poland
e-mail: marcin.kaczmarski@p.lodz.pl

Elie Kahale
Laboratoire IBISC, Université d'Evry Val d'Essonne, 91025 Evry Cedex, France
e-mail: kahale@iup.univ-evry.fr

Iwona Karcz-Dulęba
Institute of Computer Engineering, Control, and Robotics, Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: iwona.duleba@pwr.wroc.pl

Joanna Karpińska
Institute of Computer Engineering, Control and Robotics, Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: joanna.karpinska@pwr.wroc.pl

Thilo Kerscher
FZI Research Center for Information Technology, Haid-und-Neu-Str. 10-14, D-76131 Karlsruhe, Germany
e-mail: kerscher@fzi.de

Marta Kordasz
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: marta.kordasz@doctorate.put.poznan.pl

Tomasz Kornuta
Institute of Control and Computation Engineering, Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: t.kornuta@elka.pw.edu.pl

Krzysztof R. Kozłowski
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: krzysztof.kozlowski@put.poznan.pl

Bartłomiej Krysiak
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: bartlomiej.krysiak@put.poznan.pl

Fabian A. Lara-Molina
Mechanical Engineering School, State University of Campinas, Campinas, SP Brazil
e-mail: lara@fem.unicamp.br

Steven M. LaValle
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
e-mail: lavalle@uiuc.edu

Krystyna Macioszek
Space Research Centre, Polish Academy of Sciences, ul. Bartycka 18a, 00-716 Warsaw, Poland

Rafał Madoński
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: rafal.madonski@doctorate.put.poznan.pl

Alicja Mazur
Institute of Computer Engineering, Control, and Robotics, Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: alicja.mazur@pwr.wroc.pl

Bernd Mädiger
ASTRIUM Space Transportation GmbH, Bremen, Germany
e-mail: bernd.maediger@astrium.eads.net

Jakub Michalski
Materials Engineers Group Ltd., ul. Wołoska 141, 02-507 Warsaw, Poland
e-mail: j.michalski@megroup.pl

Maciej Michałek
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: maciej.michalek@put.poznan.pl

Yoshifumi Morita
Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya, Aichi 4668555, Japan
e-mail: morita@nitech.ac.jp

Rafał Osypiuk
Department of Control Engineering and Robotics, West Pomeranian University of Technology, ul. 26 Kwietnia 10, 71-126 Szczecin, Poland
e-mail: rafal.osypiuk@zut.edu.pl

Dariusz Pazderski
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: dariusz.pazderski@put.poznan.pl

Mateusz Przybyła
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: mateusz.przybyla@doctorate.put.poznan.pl

Michał Pytasz
Institute of Automatic Control, Technical University of Łódź, ul. Stefanowskiego 18/22, 90-924 Łódź, Poland
e-mail: mpytasz@swspiz.pl

Adam Ratajczak
Institute of Computer Engineering, Control, and Robotics, Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: adam.ratajczak@pwr.wroc.pl

João M. Rosario
Mechanical Engineering School, State University of Campinas, Campinas, SP Brazil
e-mail: rosario@fem.unicamp.br

Arne Rönnau
FZI Research Center for Information Technology, Haid-und-Neu-Str. 10-14, D-76131 Karlsruhe, Germany
e-mail: roennau@fzi.de

Imre J. Rudas
Institute of Intelligent Engineering Systems, John von Neumann Faculty of Informatics, Óbuda University, Bécsi út 96/B, H-1034 Budapest, Hungary
e-mail: rudas@uni-obuda.hu

Tomasz Rybus
Space Research Centre, Polish Academy of Sciences, ul. Bartycka 18a, 00-716 Warsaw, Poland
e-mail: trybus@cbk.waw.pl

Kouji Sanaka
Biological Mechanics Laboratory, Aichi, Japan

Jurek Z. Sasiadek
Department of Mechanical and Aerospace Engineering, Carleton University, 1125 Colonel By Drive, Ottawa, ON, K1S 5B6 Canada
e-mail: jsas@connect.carleton.ca

Piotr Sauer
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: piotr.sauer@put.poznan.pl

Karol Seweryn
Space Research Centre, Polish Academy of Sciences, ul. Bartycka 18a, 00-716 Warsaw, Poland
e-mail: kseweryn@cbk.waw.pl

Masaaki Shibata
Seikei University, 3-3-1 Kichijoji-kitamachi, Musashino-shi, Tokyo 180-8633, Japan
e-mail: shibam@st.seikei.ac.jp

Barbara Siemiątkowska
Institute of Automatic Control and Robotics, Warsaw University of Technology, ul. Św. A. Boboli 8, 02-525 Warsaw, Poland
e-mail: bsiem@ippt.gov.pl

Josef Sommer
ASTRIUM Space Transportation GmbH, Bremen, Germany
e-mail: josef.sommer@astrium.eads.net

Dušan M. Stipanović
Department of Industrial and Enterprise Systems Engineering and Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
e-mail: dusan@illinois.edu

Paweł Szulczyński
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: pawel.szulczynski@put.poznan.pl

Keiko Takao
Harvest Medical Welfare College, Kobe, Japan

József K. Tar
Institute of Intelligent Engineering Systems, John von Neumann Faculty of Informatics, Óbuda University, Bécsi út 96/B, H-1034 Budapest, Hungary
e-mail: tar.jozsef@nik.uni-obuda.hu

Krzysztof Tchoń
Institute of Computer Engineering, Control and Robotics, Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: krzysztof.tchon@pwr.wroc.pl

Claire J. Tomlin
Department of Electrical Engineering and Computer Science, University of California at Berkeley, Berkeley, USA
e-mail: tomlin@eecs.berkeley.edu

Piotr Trojanek
Institute of Control and Computation Engineering, Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: p.trojanek@ia.pw.edu.pl

Hiroyuki Ukai
Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya, Aichi 4668555, Japan
e-mail: ukai.hiroyuki@nitech.ac.jp

Steve Ulrich
Department of Mechanical and Aerospace Engineering, Carleton University, 1125 Colonel By Drive, Ottawa, ON, K1S 5B6 Canada
e-mail: sulrich@connect.carleton.ca

Christopher Valicka
Department of Industrial and Enterprise Systems Engineering and Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
e-mail: valicka@illinois.edu

Teréz A. Várkonyi
Applied Informatics Doctoral School, John von Neumann Faculty of Informatics, Óbuda University, Bécsi út 96/B, H-1034 Budapest, Hungary
e-mail: varkonyi.terez@phd.uni-obuda.hu

Krzysztof Walas
Institute of Control and Information Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: krzysztof.walas@put.poznan.pl

Michał Walęcki
Institute of Control and Computation Engineering, Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: mw@mwalecki.pl

Philippe Wenger
Institut de Recherche en Communications et Cybernétique de Nantes, Nantes, France
e-mail: philippe.wenger@irccyn.ec-nantes.fr

Marcin Wielgat
Materials Engineers Group Ltd., ul. Wołoska 141, 02-507 Warsaw, Poland
e-mail: m.wielgat@megroup.pl

Tomasz Winiarski
Institute of Control and Computation Engineering, Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: t.winiarski@ia.pw.edu.pl

Mateusz Wiśniowski
Institute of Automatic Control and Robotics, Warsaw University of Technology, ul. Św. A. Boboli 8, 02-525 Warsaw, Poland
e-mail: wisnio@mchtr.pw.edu.pl

Katarzyna Zadarnowska
Institute of Computer Engineering, Control, and Robotics, Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: katarzyna.zadarnowska@pwr.wroc.pl

Teresa Zielińska
Institute of Aeronautics and Applied Mechanics, Warsaw University of Technology (WUT–IAAM), ul. Nowowiejska 24, 00-665 Warsaw, Poland
e-mail: teresaz@meil.pw.edu.pl

Cezary Zieliński
Institute of Control and Computation Engineering, Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: c.zielinski@elka.pw.edu.pl

Part I
Control and Localization of Mobile Robots

Chapter 1
Symmetries in Observer Design: Review of Some Recent Results and Applications to EKF-based SLAM

Silvère Bonnabel

Abstract. In this paper, we first review the theory of symmetry-preserving observers and we mention some recent results. Then, we apply the theory to Extended Kalman Filter-based Simultaneous Localization and Mapping (EKF SLAM). It allows deriving a new (symmetry-preserving) Extended Kalman Filter for the non-linear SLAM problem that possesses convergence properties. We also prove that a special choice of the noise covariances ensures global exponential convergence.

1.1 Introduction

Symmetries and Lie groups have been widely used for feedback control in robotics, see e.g. [7, 13]. More generally, control of systems possessing symmetries has also been studied for quite a long time, see e.g. [9, 12]. The use of symmetries and Lie groups for observer design is more recent [1, 3]. The main properties of those observers are based on the reduction of the estimation error complexity. When the symmetry group coincides with the state space (observers on Lie groups), the error equation can be particularly simple [4]. This property has been used to derive non-linear observers with (almost) global convergence properties for several localisation problems [11, 4, 15]. Recently [5] established a link between observer design and control of systems on Lie groups by proving a non-linear separation principle on Lie groups.

This paper proposes to recap the main elements of the theory along with some recent results, and to apply it to the domain of Extended Kalman Filter-based Simultaneous Localization and Mapping (EKF SLAM). It is organized as follows: Section 1.2 is a brief recap on linear observers. In Section 1.3 we recap the theory of symmetry-preserving observers [3] and mention some recent results [5, 6].

Silvère Bonnabel
Centre de Robotique, Mathématiques et Systèmes, Mines ParisTech,
60 Bd Saint-Michel, 75272 Paris cedex 06, France
e-mail: silvere.bonnabel@mines-paristech.fr

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 3–15.
springerlink.com

In Section 1.4 we apply it in a straightforward way to EKF SLAM. In Section 1.5 some results for the special case of observers for invariant systems on Lie groups [4] are recalled. In Section 1.6, it is proved that those results can be (surprisingly) applied to EKF SLAM. We derive a simple globally convergent observer for the non-linear problem. We also propose a modified EKF such that the covariance matrix and the gain matrix behave as if the system was linear and time-invariant. Such non-linear convergence guarantees for EKF SLAM are new to the author's knowledge. The author would like to mention and to thank his regular co-authors on the subject of symmetry-preserving observers: Philippe Martin, Pierre Rouchon, and Erwan Salaün.

1.2 Luenberger Observers, Extended Kalman Filter

1.2.1 Observers for Linear Systems

Observers are meant to compute an estimation of the state of a dynamical system from several sensor measurements. Let $x \in \mathbb{R}^n$ denote the state of the system, $u \in \mathbb{R}^m$ be the inputs (a set of m known scalar variables such as controls, constant parameters, etc.). We assume the sensors provide measurements $y \in \mathbb{R}^p$ that can be expressed as a function of the state and the inputs. When the underlying dynamical model is a linear differential equation, and the output is a linear function as well, the system can be written

$$\frac{d}{dt}x = Ax + Bu, \qquad y = Cx + Du. \tag{1.1}$$

A Luenberger observer (or Kalman filter) writes

$$\frac{d}{dt}\hat{x} = A\hat{x} + Bu - L \cdot (C\hat{x} + Du - y), \tag{1.2}$$

where $\hat{x}$ is the estimated state, and L is a gain matrix that can be freely chosen. We can see that the observer consists of a copy of the system dynamics $A\hat{x} + Bu$, plus a correction term $L(C\hat{x} + Du - y)$ that "corrects" the trusted dynamics in function of the discrepancy between the estimated output $\hat{y} = C\hat{x} + Du$ and the measured output y.

One important issue is the choice (or "tuning") of the gain matrix L. The Luenberger observer is based on a choice of a fixed matrix L. In the Kalman filter two positive definite matrices M and N denote the covariance matrices of the state noise and measurement noise, and L relies on a Riccati equation: $L = PC^T N$, where $\frac{d}{dt}P = AP + PA^T + M^{-1} - PC^T NCP$. As M and N have to be defined by the user, they can be viewed as tuning matrices.

In both cases the observer has the form (1.2) with L constant or not. Let $\tilde{x} = \hat{x} - x$ be the estimation error, and let us compute the differential equation satisfied by the error. We have

$$\frac{d}{dt}\tilde{x} = (A + LC)\tilde{x}. \tag{1.3}$$

As the goal of the observer is to find an estimate of x, we want $\tilde{x}$ to go to zero. When the system is observable, one can always find L such that $\tilde{x}$ asymptotically exponentially goes to zero, and the negative real part of the eigenvalues of $A + LC$ can be freely assigned. We can see that the theory is particularly simple as the error equation (1.3) is *autonomous*, i.e. it does not depend on the trajectory followed by the system. In particular, the input term u has vanished in (1.3). The well-known separation principle stems from this fact.

1.2.2 Some Popular Extensions to Nonlinear Systems

Consider a general nonlinear system

$$\frac{d}{dt}x = f(x,u), \qquad y = h(x,u), \tag{1.4}$$

where $x \in \mathscr{X} \subset \mathbb{R}^n$ is the state, $u \in \mathscr{U} \subset \mathbb{R}^m$ the input, and $y \in \mathscr{Y} \subset \mathbb{R}^p$ the output. Mimicking the linear case, a class of popular nonlinear observers writes

$$\frac{d}{dt}\hat{x} = f(\hat{x},u) - L(\hat{x},y,t)\cdot\big(h(\hat{x},u) - y(t)\big), \tag{1.5}$$

where the gain matrix can depend on the variables $\hat{x}, y, t$. The error equation can still be computed, but as the system is nonlinear, it does not necessarily lead to an appropriate gain matrix L. Indeed we have $\frac{d}{dt}\tilde{x} = f(\hat{x},u(t)) - f(x,u(t)) - L(\hat{x},y(t),t)\cdot\big(h(\hat{x},u(t)) - y(t)\big)$. The error equation is no longer autonomous, and the problem of finding L such that $\tilde{x}$ goes asymptotically to zero can not be solved in the general case.

The most popular observer for nonlinear systems is the Extended Kalman Filter (EKF). The principle is to linearize the system around the estimated trajectory, build a Kalman filter for the linear model, and implement it on the nonlinear system. The EKF has the form (1.5), where the gain matrix is computed the following way:

$$A = \frac{\partial f}{\partial x}(\hat{x},u), \qquad C = \frac{\partial h}{\partial x}(\hat{x},u), \tag{1.6}$$

$$L = PC^TN^{-1}, \qquad \frac{d}{dt}P = AP + PA^T + M - PC^TN^{-1}CP. \tag{1.7}$$

The EKF has two main flaws when compared to the KF for time-invariant linear systems. First the linearized system around any trajectory is generally time-varying and the covariance matrix does not tend to a fixed value. Then, when $\hat{x} - x$ is large the linearized error equation can be a very erroneous approximation of the true error equation.

1.3 Symmetry-Preserving Observers

1.3.1 Symmetry Group of a System of Differential Equations

Let G be a group, and M be a set. A group action can be defined on M if to any $g \in G$ one can associate a diffeomorphic transformation $\varphi_g : M \to M$ such that $\varphi_{gh} = \varphi_g \circ \varphi_h$, and $(\varphi_g)^{-1} = \varphi_{g^{-1}}$, i.e., the group multiplication corresponds to the transformation composition, and the reciprocal elements correspond to reciprocal transformations.

Definition 1. *G is a symmetry group of a system of differential equations defined on M if it maps solutions to solutions. In this case we say the system is invariant.*

Definition 2. *A vector field w on M is said invariant if the system $\frac{d}{dt}z = w(z)$ is invariant.*

Definition 3. *A scalar invariant is a function $I : M \to \mathbb{R}$ such that $I(\varphi_g(z)) = I(z)$ for all $g \in G$.*

Salar invariants and invariant vector fields can be built via Cartan's moving frame method [14].

Definition 4. *A moving frame is a function $\gamma : M \to G$ such that $\gamma(\varphi_g(z)) = g \cdot \gamma(z)$ for all g, z.*

Suppose dim $G = r \leq$ dim M. Under some mild assumptions on the action (free, regular) there exists locally a moving frame. The sets $\mathscr{O}_z = \{\varphi_g(z),\ g \in G\}$ are called the group orbits. Let K be a cross-section to the orbits. A moving frame can be built locally via implicit functions theorem as the solution $g = \gamma(z)$ of the equation $\varphi_g(z) = k$, where $k \in \mathscr{O}_z \cap K$. A complete set of functionnaly independent invariants is given by the non-constant components of $\varphi_{\gamma(z)}(z)$. Figure 1.1 illustrates those definitions and the moving frame method.

1.3.2 Symmetry Group of an Observer

Consider the general system (1.4). Consider also the local group of transformations on $\mathscr{X} \times \mathscr{U}$ defined for any x, u, g by

$$\varphi_g(x,u) = \big(\varphi_g(x), \psi_g(u)\big), \tag{1.8}$$

where φ_g and ψ_g correspond to a separate local group of transformations of $\mathscr{X}$ and $\mathscr{U}$.

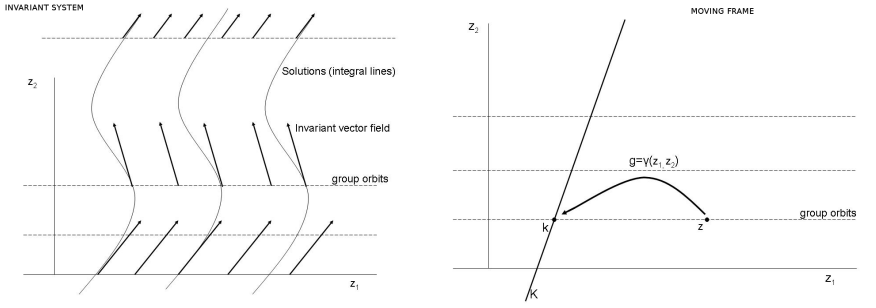


Fig. 1.1 An illustrative example. $M = \mathbb{R}^2$, and the symmetry group is made of horizontal translations. We have $\varphi_g(z_1, z_2) = (z_1 + g, z_2)^T$, where $g \in G = \mathbb{R}$. In local rectifying coordinates, every invariant system can be represented by a similar figure (under mild assumptions on the group action). Left: Invariant system. The symmetry group maps each integral line of the vector field into another integral line. Right: Moving frame method. K is a cross-section to the orbits and $\gamma(z_1, z_2)$ is the group element that maps (z_1, z_2) to K along the orbit. For example if K is the set $\{z_1 \equiv 0\}$, the moving frame is $\gamma(z_1, z_2) = z_1$ and a complete set of invariants is $I(z_1, z_2) = z_2$.

Proposition 1. *The system* $\frac{d}{dt}x = f(x,u)$ *is said invariant if it is invariant to the group action* (1.8).

The group maps solutions to solutions if we have $\frac{d}{dt}X = f(X,U)$, where $(X,U) = (\varphi_g(x), \psi_g(u))$ for all $g \in G$. We understand from this definition that u can denote the control variables as usual, but it also denotes every feature of the environment that makes the system not behave the same way after it has been transformed (via φ_g). The action of ψ_g is meant to allow some features of the environment to be also moved over. We would like the observer to be an invariant system for the *same* symmetry group.

Definition 5. *The observer* (1.5) *is invariant or "symmetry-preserving" if it is an invariant system for the group action* $(\hat{x}, x, u, y) \mapsto \big(\varphi_g(x), \varphi_g(\hat{x}), \psi_g(u), h(\varphi_g(x), \psi_g(u))\big)$.

In this case, the structure of the observer mimicks the nonlinear structure of the system. Let us recall how to build such observers (see [3] for more details). To do so, we need the output to be equivariant:

Definition 6. *The output is equivariant if there exists a group action on the output space (via* ρ_g*) such that* $h(\varphi_g(x), \psi_g(u)) = \rho_g(h(x,u))$ *for all* g, x, u.

We will systematically assume the output is equivariant. Let us define an invariant output error, instead of the usual linear output error $\hat{y} - y$:

Definition 7. *The smooth map* $(\hat{x}, u, y) \mapsto E(\hat{x}, u, y) \in \mathbb{R}^p$ *is an* invariant output error *if*

- $E\big(\varphi_g(\hat{x}),\psi_g(u),\rho_g(y)\big)=E(\hat{x},u,y)$ *for all* $\hat{x},u,y$ *(invariant)*
- *the map* $y\mapsto E(\hat{x},u,y)$ *is invertible for all* $\hat{x},u$ *(output)*
- $E\big(\hat{x},u,h(\hat{x},u)\big)=0$ *for all* $\hat{x},u$ *(error)*

An invariant error is given (locally) by $E(\hat{x},u,y)=\rho_{\gamma(\hat{x},u)}(y)-\rho_{\gamma(\hat{x},u)}(\hat{y})$. Finally, an invariant frame $(w_1,...,w_n)$ on $\mathscr{X}$, which is a set of n linearly point-wise independent invariant vector fields, i.e. $(w_1(x),...,w_n(x))$, is a basis of the tangent space to $\mathscr{X}$ at x. Once again such a frame can be built (locally) via the moving frame method.

Proposition 2. *[3] The system* $\frac{d}{dt}\hat{x}=F(\hat{x},u,y)$ *is an invariant observer for the invariant system* $\frac{d}{dt}x=f(x,u)$ *if and only if:*

$$F(\hat{x},u,y)=f(\hat{x},u)+\sum_{i=1}^{n}\mathscr{L}_i\big(I(\hat{x},u),E(\hat{x},u,y)\big)w_i(\hat{x}), \tag{1.9}$$

where E *is an invariant output error,* $I(\hat{x},u)$ *is a complete set of scalar invariants, the* $\mathscr{L}_i$*'s are smooth functions such that for all* $\hat{x}$*,* $\mathscr{L}_i\big(I(\hat{x},u),0\big)=0$*, and* $(w_1,...,w_n)$ *is an invariant frame.*

The gains $\mathscr{L}_i$ have to be tuned in order to get some convergence properties if possible, and their magnitude should depend on the trade-off between measurement noise and convergence speed. The convergence analysis of the observer often relies on an invariant state-error:

Definition 8. *The smooth map* $(\hat{x},x)\mapsto\eta(\hat{x},x)\in\mathbb{R}^n$ *is an invariant state error if* $\eta(\varphi_g(\hat{x}),\varphi_g(x))=\eta(\hat{x},x)$ *(invariant), the map* $x\mapsto\eta(\hat{x},x)$ *is invertible for all* $\hat{x}$ *(state), and* $\eta(x,x)=0$ *(error).*

1.3.3 An Example: Symmetry-Preserving Observers for Positive Linear Systems

The linear system $\frac{d}{dt}x=Ax,\ y=Cx$ admits scalings $G=\mathbb{R}^*$ as a symmetry group via the group action $\varphi_g(x)=gx$. Every linear observer is obviously an invariant observer. The unit sphere is a cross-section K to the orbits. A moving frame maps the orbits to the sphere and thus writes $\gamma(x)=1/\|x\|\in G$. A complete set of invariants is given locally by $n-1$ independent coordinates of $\varphi_{\gamma(x)}(x)=x/\|x\|$. Let $I(x)\in\mathbb{R}^{n-1}$ be a complete set of independent invariants. $I(x)$ and $\|x\|$ provide alternative coordinates named base and fiber coordinates. Moreover the system has a nice triangular structure in those coordinates. One can prove that $\frac{d}{dt}I(x(t))$ is an invariant function and thus it is necessarily of the form $g(I)$. As a result we have $\frac{d}{dt}I(x)=g(I(x))$ which does not depend on $\|x\|$.

We have thus the following (general) result: if the restriction of the vector field on the cross-section is a contraction, it suffices to define a reduced observer on the orbits, i.e. in our case a norm observer (which means that a scalar output suffices for observability). This is the case for instance when A is a matrix whose coefficients

are stricly positive (according to the Perron-Froebenius theorem). This fact was recently used in [6] to derive invariant asymptotic positive observers for positive linear systems.

1.4 A First Application to EKF SLAM

Simultaneous localisation and mapping (SLAM) addresses the problem of building a map of an environment from a sequence of sensor measurements obtained from a moving robot. A solution to the SLAM problem has been seen for more than twenty years as a "holy grail" in the robotics community since it would be a means to make a robot truly autonomous in an unknown environment. A very well-known approach that appeared in the early 2000's is the EKF SLAM [8]. Its main advantage is to formulate the problem in the form of a state-space model with additive Gaussian noise and to provide convergence properties in the linear case (i.e. straight line motion). Indeed, the key idea is to include the position of the several landmarks (i.e. the map) in the state space. This solution has been gradually replaced by other techniques such as FastSLAM, Graph SLAM etc.

In the framework of EKF SLAM, the problem of estimating online the trajectory of the robot as well as the location of all landmarks without the need for any a priori knowledge of location can be formulated as follows [8]. The vehicle state is defined by the position in the reference frame (earth-fixed frame) $x \in \mathbb{R}^2$ of the centre of the rear axle and the orientation of the vehicle axis θ. The vehicle trusted motion relies on non-holonomic constraints. The landmarks are modeled as points and represented by their position in the reference frame $p_i \in \mathbb{R}^2$, where $1 \leq i \leq N$. $u, v \in \mathbb{R}$ are control inputs. Both vehicle and landmark states are registered in the same frame of reference. In a determistic setting (state noises turned off), the time evolution of the (huge) state vector is

$$\dot{x} = u\, R_\theta e_1, \quad \dot{\theta} = uv, \quad \dot{p}_i = 0, \quad 1 \leq i \leq N, \tag{1.10}$$

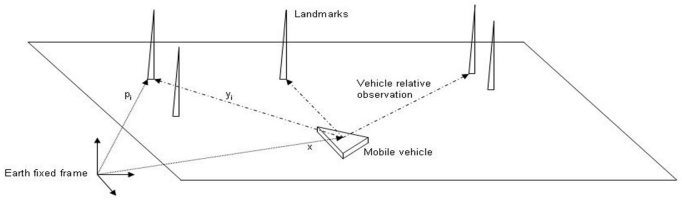


Fig. 1.2 Vehicle taking relative measurements to environmental landmarks

where $e_1 = (1,0)^T$ and R_θ is the rotation matrix of angle θ. Supposing that the data association between the landmarks from one instant to the next is correctly done, the observation model for the i-th landmark (disregarding measurement noise) is its position seen from the vehicle's frame: $z_i = R_{-\theta}(p_i - x)$. The standard EKF SLAM estimator has the form

$$\frac{d}{dt}\hat{x} = uR_{\hat{\theta}}e_1 + R_{\hat{\theta}}(\sum_1^N L_x^k(\hat{z}_k - z_k)), \quad \frac{d}{dt}\hat{\theta} = uv + \sum_1^N L_\theta^k(\hat{z}_k - z_k), \quad \frac{d}{dt}\hat{p}_i = R_{\hat{\theta}}(\sum_1^N L_i^k(\hat{z}_k - z_k)). \tag{1.11}$$

where $\hat{z}_i = R_{-\hat{\theta}}(\hat{p}_i - \hat{x})$ and where the L_i's are the lines of L tuned via the EKF equations (1.6)–(1.7).

Here the group of symmetry of the system corresponds to Galilean invariances, and it is made of rotations and translations of the plane $SE(2)$. Indeed, looking at Fig. 1.2, it is obvious that the equations of motion are the same whether the first horizontal axis of the reference frame is pointing North, or East, or in any direction. For $g = (x_0, \theta_0) \in SE(2)$, the action of the group on the state space is $\varphi_g(x, \theta, p_i) = (R_{\theta_0}x + x_0, \theta + \theta_0, R_{\theta_0}p_i + x_0)$ and $\psi_g(u,v) = u, v$. The output is also unchanged by the group transformation as it is expressed in the vehicle frame and is thus insensitive to rotations and translations of the reference frame. Applying the theory of the last section, the observer above can be "invariantized", yielding the following invariant observer:

$$\frac{d}{dt}\hat{x} = uR_{\hat{\theta}}e_1 + R_{\hat{\theta}}(\sum_1^N L_x^k(\hat{z}_k - z_k)), \quad \frac{d}{dt}\hat{\theta} = uv + \sum_1^N L_\theta^k(\hat{z}_k - z_k), \quad \frac{d}{dt}\hat{p}_i = R_{\hat{\theta}}(\sum_1^N L_i^k(\hat{z}_k - z_k)). \tag{1.12}$$

It is easy to see that the invariant observer is much more meaningful, especially if the $L_i's$ are chosen as constant matrices [3]. Indeed, one could really wonder if it is sensible to correct vectors expressed in the reference frame directly with measurements expressed in the vehicle frame. To be convinced, consider the following simple case: suppose $\hat{\theta} = \theta = \hat{x} = x = 0$ remain fixed. We have $\frac{d}{dt}(\hat{p}_i - p_i) = L_i(\hat{p}_i - p_i)$. Choosing $L_i = -k\ I$ yields $\frac{d}{dt}\|\hat{p}_i - p_i\|^2 = -k\|\hat{p}_i - p_i\|^2$, leading to a correct estimation of landmark p_i. Now suppose that the vehicle has changed its orientation and $\hat{\theta} = \theta = \pi/2$. The output error is now $R_{-\pi/2}(\hat{p}_i - p_i)$. With an observer of the form (1.11) the same choice $L_i = -k\ I$ yields $\frac{d}{dt}\|\hat{p}_i - p_i\| = 0$ and the landmark is not correctly estimated. On the other hand, with (1.12) we have in both cases $\frac{d}{dt}\|\hat{p}_i - p_i\|^2 = -k\|\hat{p}_i - p_i\|^2$, ensuring convergence of $\hat{p}$ towards p.

Constant gains is a special (simple) choice, but the observer gains can also be tuned via Kalman equations. Indeed on can define noises on the linearized invariant error system and tune the L_i's via Kalman equations (see the Invariant EKF method [2]). To sum up, any Luenberger observer or EKF can be invariantized via Eqs. (1.12). This yields in the author's opinion a much more meaningful non-linear observer that is well-adapted to the problem's structure. The invariantized observer

(1.12) is simply a version of (1.11) which is less sensitive to change of coordinates, and even if no proof can support this claim we believe it can only improve the performance of (1.11).

1.5 Particular Case Where the State Space Coincides with Its Symmetry Group

Over the last half decade, invariant observers on Lie groups for low-cost aided inertial navigation have been studied by several teams in the world, [11, 3, 15] to name a few. Several powerful convergence results have been obtained. They are all linked to the special properties of the invariant state error on a Lie group. To recap briefly the construction of invariant observers on Lie groups [4], we assume that the symmetry group G is a matrix group, and that $\mathscr{X} = G$. The system is assumed to be invariant to left multiplications, i.e. $\frac{d}{dt}X = X\Omega(t)$. We have indeed for any $g \in G$ that $\frac{d}{dt}(gX) = (gX)\Omega$. For instance the motion of the vehicle in the considered SLAM problem $\dot{x} = uR_\theta e_1, \dot{\theta} = uv$ can be viewed as a left-invariant system on the Lie group SE(2) via the matrix representation

$$X = \begin{pmatrix} R_\theta & x \\ 0_{1\times 2} & 1 \end{pmatrix}, \quad \Omega = \begin{pmatrix} \omega_x & ue_1 \\ 0_{1\times 2} & 0 \end{pmatrix}, \text{ with } \omega_x = \begin{pmatrix} 0 & -uv \\ uv & 0 \end{pmatrix}.$$

Suppose the output $y = h(X)$ is equivariant, i.e. there exists a group action on the output space such that $h(gX) = \rho_g(X)$. In this case the invariant observer (1.9) can be written intrinsically

$$\frac{d}{dt}\hat{X} = \hat{X}\Omega + \hat{X}L(\rho_{\hat{X}^{-1}}(y)),$$

with $L(e) = 0$, where e is the group identity element. The invariant state error is the natural group difference $\eta = X^{-1}\hat{X}$ and the error equation is

$$\frac{d}{dt}\eta = [\Omega, \eta] + \eta L \circ h(\eta^{-1}).$$

A remarkable fact is that the error equation only depends on η and Ω, whereas the system is non-linear and the error should also depend on $\hat{X}$ (think about the EKF which is based on a linearization around any $\hat{X}$ at each time). Moreover, if $\Omega = cst$, the error equation is clearly *autonomous*. Thus the motion primitives generated by constant Ω are special trajectories called "permanent trajectories". Around such trajectories one can always achieve local convergence (as soon as the linearized system is observable).

It is worth noting this property was recently used to derive a non-linear *separation principle* on Lie groups [5]. It applies to some cart-like underactuated vehicles and some underwater or aerial fully actuated vehicles.

An even more interesting case occurs when the output satisfies right-equivariance, i.e. $h(Xg) = \rho_g(h(X))$. In this case we let the input be $u = \Omega$ and we consider the

action of G by right multiplication, i.e. $\varphi_g(X) = Xg$ and $\psi_g(\Omega) = g^{-1}\Omega g$. The output is equivariant as $h(\varphi_g(X)) = \rho_g \circ h(X)$. The invariant observer associated with this group of symmetry writes $\frac{d}{dt}\hat{X} = \hat{X}\Omega + L(\rho_{\hat{X}^{-1}}(y))\hat{X}$. The invariant state error is $\eta = \hat{X}X^{-1}$ and the error equation is

$$\frac{d}{dt}\eta = \hat{X}\Omega X^{-1} + L(h(\eta^{-1}))\eta - \hat{X}\Omega X^{-1} = L(h(\eta^{-1}))\eta. \tag{1.13}$$

The error equation is completely autonomous! In particular the linearized system around *any* trajectory is the same time-invariant system. Autonomy is the key for numerous powerful convergence results for observers on Lie groups; see e.g. [10, 15, 4].

1.6 A New Result in EKF SLAM

In this section we propose a new non-linear observer for EKF SLAM with guaranteed convergence properties. In the SLAM problem the state space is much bigger than its symmetry group. The orbits have dimension 3 and thus there are $N+1-3+2 = N$ invariants (dimension of the cross-section, see Fig. 1.1). Thus an autonomous error equation seems to be out of reach. Suprinsingly, considering the symmetry group of rotations and translations in the vehicle frame yields such a result. A simple trick makes it obvious. Consider the following matrix representation:

$$X = \begin{pmatrix} R_\theta & x \\ 0_{1\times 2} & 1 \end{pmatrix}, \quad P_i = \begin{pmatrix} R_\theta & p_i \\ 0_{1\times 2} & 1 \end{pmatrix}, \quad \Omega = \begin{pmatrix} \omega_x & ue_1 \\ 0_{1\times 2} & 0 \end{pmatrix}, \quad \Omega_i = \begin{pmatrix} \omega_x & 0 \\ 0_{1\times 2} & 0 \end{pmatrix}.$$

The equations of the system (1.10) can be written $\frac{d}{dt}X = X\Omega$, $\frac{d}{dt}P_i = P_i\Omega_i$, $1 \leq i \leq N$ and the system can be viewed as a left-invariant dynamics system on the (huge) Lie group $G \times \cdots \times G$. Let $\eta_x = \hat{X}X^{-1}, \eta_i = \hat{P}_i P_i^{-1}$ be the invariant state error. The system has the invariant output errors $\tilde{Y}_i = R_{\hat{\theta}}(\hat{z}_i - z_i)$, i.e. $\begin{pmatrix}\tilde{Y}_i & 1\end{pmatrix}^T = (\eta_i - \eta_x)H$ for $1 \leq i \leq N$, where $H = \begin{pmatrix}0_{1\times 2} & 1\end{pmatrix}^T$. Consider the following invariant observer: $\frac{d}{dt}\hat{X} = \hat{X}\Omega + L_X(\tilde{Y}_1, \cdots, \tilde{Y}_N)\hat{X}$, $\frac{d}{dt}\hat{P}_i = \hat{P}_i\Omega_i + L_i(\tilde{Y}_1, \cdots, \tilde{Y}_N)\hat{P}_i$. From (1.13), the (non-linear) error equation is completely autonomous reminding the linear case (1.3). It implies the following global convergence result for the non-linear deterministic system:

Proposition 3. *Consider the SLAM problem (1.10) without noise. The following observer:*

$$\frac{d}{dt}\hat{\theta} = uv, \quad \frac{d}{dt}\hat{x} = uR_{\hat{\theta}}e_1, \quad \frac{d}{dt}\hat{p}_i = k_i\, R_{\hat{\theta}}(\hat{z}_i - z_i)$$

with $k_i > 0$ is such that $\frac{d}{dt}(R_{\hat{\theta}}(\hat{z}_i - z_i)) = -k_i\, R_{\hat{\theta}}(\hat{z}_i - z_i)$, i.e., all the estimation errors $(\hat{z}_i - z_i)$, $1 \leq i \leq N$ converge globally exponentially to zero with rate k_i, which

means the vehicle trajectory and the map are correctly identified. The parameter k_i has to be tuned according to the level of noise associated to landmark i, and vehicle sensors' noise.

If one wants to define noise covariance matrices M,N to tune the observer (and compute an estimation P of the covariance error matrix at each time), it is also possible to define a modified EKF with guaranteed convergence properties:

Proposition 4. *Consider the SLAM problem (1.10). Let $E = (R_{\hat\theta}(\hat z_i - z_i))_{1\le i\le N}$ be the invariant output error. Let e_3 be the vertical axis. Consider the observer*

$$\frac{d}{dt}\hat\theta = uv + \mathscr{L}_\theta(E), \quad \frac{d}{dt}\hat x = uR_{\hat\theta}e_1 + \mathscr{L}_\theta(E)e_3\wedge\hat x + \mathscr{L}_x(E), \quad \frac{d}{dt}\hat p_i = \mathscr{L}_\theta(E)e_3\wedge\hat p + \mathscr{L}_i(E).$$

Let $\eta = (\tilde\theta, \tilde x, \tilde p_1, \cdots, \tilde p_n)$ be the invariant state error where $\tilde\theta = \hat\theta - \theta$, $\tilde x = \hat x - R_{\hat\theta}x$, $\tilde p_i = \hat p_i - R_{\hat\theta}p_i$. The state error equation is autonomous, i.e. $\frac{d}{dt}\eta$ only depends on η. It is thus completely independent of the trajectory and of $u(t), v(t)$. The linearized error equation writes $\frac{d}{dt}\delta\eta = (LC)\delta\eta$, where L can be freely chosen and C is a fixed matrix. As in the usual EKF method, one can define covariance matrices M, N, build a Kalman filter for the linearized system, i.e. tune L via the usual equations (1.7), i.e. $\dot P = M - PC^TN^{-1}CP$, $L = PC^TN^{-1}$, and implement it on the non-linear model. All the convergence results on P and L valid for stationary systems (1.1) with $A = 0$, $B = 0$, $D = 0$ apply.

Simulations (Fig. 1.3) with one landmark and noisy measurements indicate the modified EKF (IEKF) behaves very similarly, or slightly better than the EKF, but the gain matrix tends quickly to a fixed matrix L independently from the trajectory and the inputs u, v. So the Invariant EKF proposed in this paper: 1– is incomparably cheaper computationaly as it relies on a constant matrix L that can be computed offline once and for all (the number of landmarks can thus be much increased); 2– is such that the linearized error system is stable as soon as LC has negative eigenvalues, which is easy to verify.

Remark 1. *The calculations above are valid on $SE(3)$ and could apply to 6 DOF SLAM.*

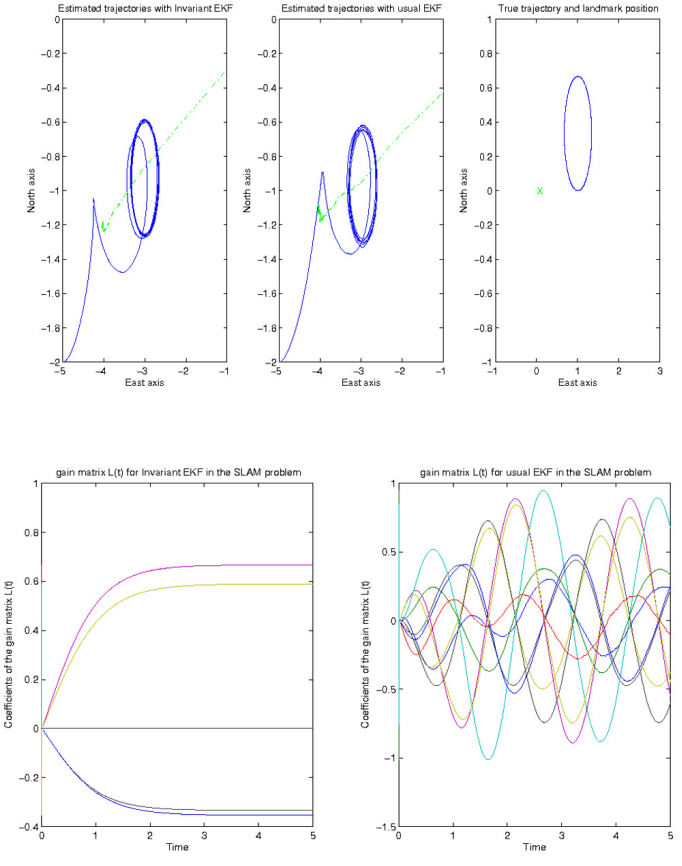


Fig. 1.3 Simulations with one landmark and a car moving over a circular path with a 20% noise. Top: 1– estimated vehicle trajectory (plain line) and landmark position (dashed line) with Invariant EKF, 2– estimation with the usual EKF, 3– true vehicle trajectory (plain line) and landmark position (cross). After a short transient, the trajectory is correctly identified for both observers (up to a rotation-translation). Bottom: 1– coefficients of $L(t)$ over time for Invariant EKF, 2– coefficients of $L(t)$ for EKF. We can see the EKF gain matrix is permanently adapting to the motion of the car (right) whereas its invariant counterpart (left) is directly expressed in well-adapted variables.

References

1. Aghannan, N., Rouchon, P.: On invariant asymptotic observers. In: 41st IEEE Conference on Decision and Control, pp. 1479–1484 (2002)
2. Bonnabel, S., Martin, P., Salaun, E.: Invariant extended kalman filter: Theory and application to a velocity-aided attitude estimation problem. In: IEEE Conference on Decision and Control (2009)
3. Bonnabel, S., Martin, P., Rouchon, P.: Symmetry-preserving observers. IEEE Trans. on Automatic Control 53(11), 2514–2526 (2008)

4. Bonnabel, S., Martin, P., Rouchon, P.: Non-linear symmetry-preserving observers on lie groups. IEEE Trans. on Automatic Control 54(7), 1709–1713 (2009)
5. Bonnabel, S., Martin, P., Rouchon, P., Salaun, E.: A separation principle on lie groups. In: IFAC (available on Arxiv) (2011)
6. Bonnabel, S., Sepulchre, R.: Contraction and observer design on cones. In: Arxiv (2011)
7. Bullo, F., Murray, R.M.: Tracking for fully actuated mechanical systems: A geometric framework. Automatica 35(1), 17–34 (1999)
8. Dissanayake, G., Newman, P., Durrant-Whyte, H.F., Clark, S., Csobra, M.: A solution to the simultaneous localisation and mapping (slam) problem. IEEE Trans. Robot. Automat. 17, 229–241 (2001)
9. Grizzle, J.W., Marcus, S.I.: The structure of nonlinear systems possessing symmetries. IEEE Trans. Automat. Control 30, 248–258 (1985)
10. Lagemann, C., Trumpf, J., Mahony, R.: Gradient-like observers for invariant dynamics on a lie group. IEEE Trans. on Automatic Control 55(2), 367–377 (2010)
11. Mahony, R., Hamel, T., Pflimlin, J.-M.: Nonlinear complementary filters on the special orthogonal group. IEEE-Trans. on Automatic Control 53(5), 1203–1218 (2008)
12. Martin, P., Rouchon, P., Rudolph, J.: Invariant tracking. ESAIM: Control, Optimisation and Calculus of Variations 10, 1–13 (2004)
13. Morin, P., Samson, C.: Practical stabilization of driftless systems on lie groups, the transverse function approach. IEEE Trans. Automat. Control 48, 1493–1508 (2003)
14. Olver, P.J.: Classical Invariant Theory. Cambridge University Press (1999)
15. Vasconcelos, J.F., Cunha, R., Silvestre, C., Oliveira, P.: A nonlinear position and attitude observer on se(3) using landmark measurements. Systems Control Letters 59, 155–166 (2010)

Chapter 2
Constraint Control of Mobile Manipulators

Mirosław Galicki

Abstract. This work offers a solution at the control feedback level of a point-to-point mobile manipulator control task. The task is subject to state equality and/or inequality constraints. Based on the Lyapunov stability theory, a class of asymptotically stable controllers fulfilling the above constraints and generating a singularity-free and collision-free mobile manipulator trajectory is proposed. The problem of singularity and collision avoidance enforcement is solved here based on an exterior penalty function approach, which results in continuous and bounded mobile manipulator controls even near boundaries of obstacles. Numerical simulation results carried out for a mobile manipulator consisting of a nonholonomic unicycle and a holonomic manipulator of two revolute kinematic pairs, operating both in a two-dimensional unconstrained task space and a task space including the obstacles, illustrate the performance of the proposed controllers.

2.1 Introduction

Mobile manipulators have recently been applied to useful practical tasks such as assembly, inspection, spray painting, or part transfer. By combining mobility of the nonholonomic platform with manipulability of the holonomic manipulator, performance characteristics of the mobile manipulator are improved which enable it to accomplish complicated tasks (including point-to-point ones) in complex task spaces including many obstacles. Application of mobile manipulators to the tasks mentioned above complicates their performance because such manipulators provide, in general, non-unique solutions. Consequently, some objective criteria should be specified to solve the mobile manipulator tasks uniquely. Several approaches may

Mirosław Galicki
Faculty of Mechanical Engineering, University of Zielona Góra, Zielona Góra, Poland and Institute of Medical Statistics, Computer Science and Documentation, Friedrich Schiller University, Jena, Germany
e-mail: m.galicki@ibem.uz.zgora.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 17–26.
springerlink.com

be distinguished in such a context. Most of the existing literature deals with the end-effector trajectory tracking using only kinematic equations of the mobile manipulator (see e.g. [1] for an exhaustive overview of the corresponding literature). There are few references dealing with optimal solutions of a point-to-point control problem subject to mobile manipulator dynamic equations and state constraints [2], [3], [4]. Although all the aforementioned algorithms produce optimal solutions, they are not suitable to real-time implementations due to their computational complexity. Therefore, it is natural to attempt alternative techniques for generating the mobile manipulator trajectories in real time [5], [6], [7], [8], [9].

The present work addresses the problem of controlling the mobile manipulator subject to various state and control variable constraints. Kinematic and dynamic parameters of the mobile manipulator can be obtained with sufficient accuracy by means of a calibration and identification technique [10], [11]. Consequently, they are assumed further on to be fully known. The task is to move the mobile manipulator from its arbitrary initial configuration to a fixed desired end-effector location in the task space. When accomplishing this task, the mobile manipulator is required to avoid both holonomic singular configurations (defined formally further on) and collisions with obstacles. According to the author's knowledge, such a general task has not been analyzed before. In [13], [14], the trajectory tracking control problem subject to the aforementioned constraints is solved. The algorithms from [13], [14] require however the pseudo-inverse of the Jacobian matrix, which may introduce numerical instabilities to the control algorithm.

By the fulfilment of a reasonable condition regarding the Jacobian matrix rank, the proposed regulation scheme is shown to be both asymptotically stable and to satisfy the state constraints. Moreover, the controls provided to the mobile manipulator are bounded. The remainder of the paper is organised as follows. Section 2.2 formulates the singularity- and collision-free point-to-point control problem as a constrained optimization task. Section 2.3 sets up a class of controllers solving the mobile manipulator task subject to control and state dependent constraints. Section 2.4 presents computer examples of the task accomplishment for a mobile manipulator consisting of a non-holonomic unicycle and a holonomic manipulator of two revolute kinematic pairs, operating in both an unconstrained two-dimensional task space and a task space with obstacles. Finally, some concluding remarks are drawn in Section 2.5.

2.2 Kinematics and Dynamics of the Mobile Manipulator

Consider a mobile manipulator composed of a non-holonomic platform. It is described by the vector of the location of the platform $x \in R^l$ (platform posture $x_{1,c}\ x_{2,c},\ \theta$, and angles of the driving wheels $\varphi_1,\ \varphi_2$ – see Fig. 2.1, where θ is the orientation angle of the platform with respect to a global coordinate system Ox_1x_2; $x_{1,c},\ x_{2,c}$ stand for coordinates of unicycle centre; W denotes one half of the distance between the platform wheels; $2L$ is the platform length; R stands for the wheel radius; (a, b) denotes the point in a local coordinate frame associated with

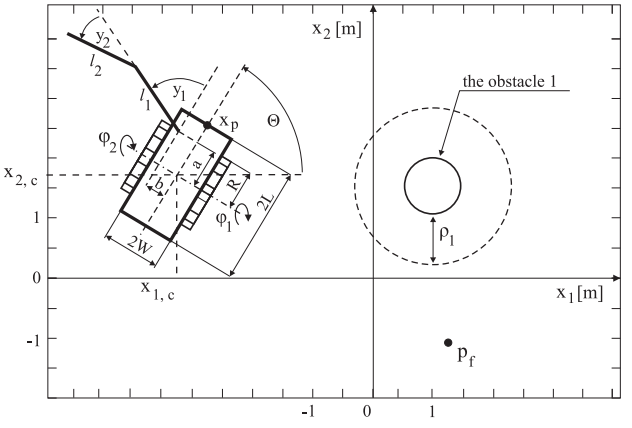


Fig. 2.1 A kinematic scheme of the mobile manipulator and the task to be accomplished

the platform at which the holonomic manipulator base is fastened to the platform), where $l \geq 1$, and joint coordinates $y \in R^n$ of a holonomic manipulator mounted on the platform, where n is the number of its kinematic pairs. The platform motion is usually subject to $1 \leq k < l$ nonholonomic constraints in the so-called Pfaffian form

$$A(x)\dot{x} = 0, \tag{2.1}$$

where $A(x)$ stands for the $k \times l$ matrix of the full rank (that is, $rank(A(x)) = k$), depending on x analytically. Suppose that $Ker(A(x))$ is spanned by vector fields $a_1(x), \ldots, a_{l-k}(x)$. Then, the kinematic constraint (2.1) can be equivalently expressed by an analytic driftless dynamic system

$$\dot{x} = N(x)\alpha\,, \tag{2.2}$$

where $N(x) = [a_1(x), \ldots, a_{l-k}(x)]$, $rank(N(x)) = l - k$, vector $\alpha = (\alpha_1, \ldots, \alpha_{l-k})^T$ denotes some of the quasi-velocities of the platform.

The position and orientation of the end-effector with respect to an absolute coordinate frame is described by a mobile manipulator kinematic equation $f(x,y) = (f_1, \ldots, f_m)^T$, where $f : R^l \times R^n \longrightarrow R^m$ denotes the m-dimensional (in general, nonlinear with respect to x and y) mapping; vector of the generalized coordinates $q = \left(x^T\ y^T\right)^T$ represents the configuration of the mobile manipulator and m stands for the dimension of the task space.

The dynamics of a mobile manipulator described in generalized coordinates is given by the following equation [15]:

$$M^{'}(q)\ddot{q} + F^{'}(q,\ \dot{q}) + [A(x)\ 0_{k\times n}]^T\ \lambda = B^{'}v, \tag{2.3}$$

where $M^{'}(q)$ denotes the $(n+l) \times (n+l)$ inertia matrix; $F^{'}(q,\ \dot{q}) = C^{'}(q,\ \dot{q})\dot{q} + D(q)$; $C^{'}(q,\ \dot{q})\dot{q}$ is the $(n+l)$–dimensional vector representing centrifugal and Coriolis forces; $D(q)$ stands for the $(n+l)$-dimensional vector of generalized gravity

forces; $0_{k\times n}$ denotes the $k\times n$ zero matrix; λ is the vector of Lagrange's multipliers corresponding to non-holonomic constraints (2.1); $B^{'}=\begin{bmatrix} B^{''} & 0 \\ 0 & I_n \end{bmatrix}$; $B^{''}$ stands for the $l\times(l-k)$ matrix describing which location variables of the platform are directly driven by the actuators; I_n denotes the $n\times n$ identity matrix, $R^{n+l-k}\ni v$ is the vector of controls (torques/forces) and $(q^T,\ \dot{q}^T)^T$ denotes the state vector of the mobile manipulator. Combining $\dot{q}$ and $\ddot{q}$ with (2.2) results in the following relations:

$$\dot{q}=Cs,\quad \ddot{q}=C\dot{s}+\dot{C}s, \tag{2.4}$$

where $C=\begin{bmatrix} N(x) & 0 \\ 0 & I_n \end{bmatrix}\in R^{(l+n)\times(l-k+n)}$ and $s=\begin{pmatrix}\alpha\\ \dot{y}\end{pmatrix}$ is the vector of the reduced velocities of the mobile manipulator. Our purpose is to reduce the dimension of dynamic equations (2.3) using dependencies (2.4). Premultiplying (2.3) by C^T, substituting the right-hand side of $\ddot{q}$ from (2.4) into (2.3) and noting that $A(x)N(x)=0$, we obtain dynamic equations in a reduced form

$$M(q)\dot{s}+F(q,\ s)=Bv, \tag{2.5}$$

where $M=C^TM^{'}C,\ F=C^T\left(M^{'}\dot{C}s+F^{'}\right)$, and $B=\begin{bmatrix} N^TB^{''} & 0 \\ 0 & I_n \end{bmatrix}$. From (2.5) it follows that the reduced dynamic equations are fully controllable since the dimension of the vector s equals the number of controls v. Let us note that the $(n+l-k)\times(n+l-k)$ matrix B is invertible since (by assumption) $B^{''}$ has full rank. Hence

$$\dot{s}=M^{-1}(Bv-F). \tag{2.6}$$

Applying the transformation $v=(M^{-1}B)^{-1}(u+M^{-1}F)$, where u is a new control vector, we simplify dynamic equations (2.6) and express them in partial configuration variables as follows:

$$\dot{s}=u. \tag{2.7}$$

Dependence (2.7) presents an input-state linearizable equation of the mobile manipulator. However, form (2.7) is practically cumbersome since it does not directly comprise all the coordinates (coordinates of the vector q) in which the mobile manipulator task (attaining a desirable position and orientation by the end-effector) is usually expressed.

The regulation aim is to design a task space controller which generates control vector u in such a way that the task error $e=(e_1,\ldots,e_m)^T=f(q)-p_f$, where p_f is a fixed desired end-effector location, and the final manipulator reduced velocity tend asymptotically to zero

$$\lim_{t\to\infty}e(t)=0,\qquad \lim_{t\to\infty}s(t)=0. \tag{2.8}$$

Moreover, it is practically important to attain the desired end-effector location with as large as possible manipulability measure $S(y)$ of the holonomic manipulator (introduced by Yoshikawa [12]). If a manipulability value were small, then a

reconfiguration of the whole mobile manipulator at p_f would be necessary to efficiently accomplish the end-goal task (e.g. a part assembly). Thus, the following (state) optimization constraint is imposed on the holonomic part:

$$-S(y) \longrightarrow \min_{y}. \tag{2.9}$$

During the mobile manipulator movement, collision-avoidance (state inequality) constraints resulting from the existence of obstacles in the work space are induced. The general form of these constraints can be written in the following manner:

$$\{c^j(q) > 0\}, \qquad j = 1 : N, \tag{2.10}$$

where c^j denotes either a distance function [16] or an analytic description of an obstacle [17], and N stands for the total number of collision-avoidance constraints. We postulate further on that $q(0)$ together with its small neighbourhood does not cause a collision, i.e., $c^j(q(0)) > 0$. Moreover, functions c^j from (2.8) are assumed to belong to a class of smooth mappings with smooth and bounded derivatives with respect to any q. The next section will present an approach to the solution (provided that it exists) of the control problem (2.8)–(2.10) making use of the Lyapunov stability theory.

2.3 Generating the Control of the Mobile Manipulator

Combining Eqs. (2.4) and (2.7), we obtain a simplified form of the mobile manipulator dynamic equations expressed in all the configuration and state variables

$$\ddot{q} = Cu + \dot{C}s. \tag{2.11}$$

Thus, the mobile manipulator task may now be reformulated as follows: find a control u which moves the mobile manipulator from initial configuration and velocity $q(0)$, $\dot{q}(0)$, such that $e(t) = 0$ and $s(t) = 0$ as $t \to \infty$. Moreover, u should steer the mobile manipulator in such a way as to avoid both holonomic singular configurations and collisions with obstacles.

2.3.1 Constraint-Free Mobile Manipulator Motion

In order to determine mobile manipulator controls, state inequality constraints (2.9), (2.10) are not considered in this section. They will be taken into account in a control scheme proposed in Section 3.2. Let matrix JC, where $J = \frac{\partial e}{\partial q} = \frac{\partial f}{\partial q}$ stands for the $m \times (n+l)$ mobile manipulators' Jacobian matrix, be non-singular during the movement in a region of our interest. It is easy to see that matrix JC becomes non-singular for the mobile manipulator depicted in Fig. 2.1 provided that $a > l_1 + l_2$.

Based on (2.8)–(2.10), a (simple) control law is proposed, as follows:

$$u = -k_0 C^T J^T g(e) - k_1 g(s), \tag{2.12}$$

where k_0 and k_1 stand for positive coefficients (controller gains) which could be replaced by diagonal matrices of positive constants without affecting the stability results obtained further on (this should lead to improved performance); $g(\cdot) = \frac{2}{\pi}\arctan(\cdot)$. The function $g(\cdot)$ is taken over all the coordinates of vectors e and s, respectively. The regulation aim is to provide conditions on controller gains k_0 and k_1 guarantying asymptotic stability of the origin $(e,\ s) = (0,\ 0)$ which is attainable by the control (2.12). Applying the Lyapunov stability theory we derive the following result.

Theorem 2.1. *If JC is the full rank matrix in a (closed) region of the task space and*

$$k_0,\ k_1 > 0 \tag{2.13}$$

then control law (2.12) guarantees asymptotic stability of the origin $(e,\ s) = (0,\ 0)$.

Proof. Consider a Lyapunov function candidate

$$V(e,\ s) = k_0 \int_0^e \langle g(x),\ dx\rangle + \frac{1}{2}\langle s,\ s\rangle, \tag{2.14}$$

where $\langle,\ \rangle$ is the scalar product of the vectors. From the definition of g, it follows that $\langle g(x),\ dx\rangle \geq 0$. Hence, V is positive definite. The time derivative of V is given by $\dot{V} = k_0\langle g(e),\ JCs\rangle + \langle s,\ \dot{s}\rangle$. Substituting $\dot{s} = u$ from the above dependence for the right-hand side of Eq. (2.12), we obtain after simple calculations $\dot{V} = -k_1\langle s,\ g(s)\rangle$. As can be seen, $\dot{V}$ is negative for all $s \neq 0$. Hence, trajectories e, s are bounded. The set $\{(e,\ s):\ \dot{V}(e,\ s) = 0\}$ takes in such a case the following form: $\{(e,\ s):\ s = 0,\ e\ arbitrary\}$. For some solution $s(t) = 0$ for $\forall t \geq 0$, it follows that $\dot{s} = 0 = u$. Based on (2.12), we obtain the following equality: $(JC)^T g(e) = 0$. On account of the assumption regarding the non-singularity of JC, we have $g(e) = 0$. From the definition of g, it follows that e is a unique root of $g(e) = 0$. Consequently, the maximal invariant set is equal to $\{(e,\ s) = (0,\ 0):\ \dot{V}(e,\ s) = 0\}$. Finally, using the La Salle-Yoshizawa invariant theorem [18], we conclude that $(e,\ s)$ tends (asymptotically) to the origin $(0,\ 0)$, as $t \to \infty$. □

Several observations can be made regarding the controller (2.12). First, it is computationally simple. Second, the regulator (2.12) provides bounded controls (the norm of JC is bounded). Finally, as opposed to many algorithms known from the literature, the control law (2.12) does not require any pseudo-inverse of the matrix JC. Although JC could be singular, the controller (2.12) may still work provided that $g(e) \notin Ker\left((JC)^T\right)$.

2.3.2 Constraint Control of the Mobile Manipulator

In this section, we propose a technique which enforces the fulfilment of the condition (2.9). In order to avoid singular holonomic configurations, we propose the following control law:

$$u = -k_0 C^T J^T g(e) - k_1 g(s) + k_2 C^T \frac{\partial S}{\partial q}, \tag{2.15}$$

where k_2 is a positive gain coefficient. Applying a similar argumentation as in Theorem 2.1 and assuming linear independence of the vectors $k_0 C^T J^T g(e)$ and $k_2 C^T \frac{\partial S}{\partial q}$, we can easily prove the following theorem:

Theorem 2.2. *If JC is the full rank matrix in a (closed) region of the task space and*

$$k_0,\ k_1,\ k_2 > 0 \tag{2.16}$$

then control law (2.15) guarantees both asymptotic stability of the origin $(e,\ s) = (0,\ 0)$ and the avoidance of the holonomic singular configurations.

Proof. Defining the Lyapunov function candidate $V_h(e,\ s,\ q) = k_0 \int_0^e \langle g(x),\ dx\rangle + \frac{1}{2}\langle s,\ s\rangle + k_2(S_{\max} - S)$, where $S_{\max}$ denotes the maximal value of the holonomic manipulability measure, it is easy to prove the aforementioned properties of the controller (2.15). □

Let us note that at the equilibrium state, $\left(e,\ s,\ \frac{\partial S}{\partial q}\right) = (0,\ 0,\ 0)$. Hence, function V_h equals zero. In order to additionally involve state inequality constraints (2.10) in the mobile manipulator controller (2.15), we introduce the following exterior penalty function to satisfy collision avoidance constraints (2.10):

$$U(q) = \sum_{j=0}^{N'} c_j E(c^j), \tag{2.17}$$

where $E(c^j) = \left(c^j - \rho_j\right)^2$ for $c^j \le \rho_j$ and $E(c^j) = 0$ otherwise; ρ_j is a user-defined sufficiently small positive number (the extent of the enlargement of the j-th obstacle). In the penalty function approach, ρ_j represents a given threshold value which activates the j-th inequality constraint after exceeding this value; $N' \le N$ is the number of only active constraints (2.10) (i.e. such that $c^j - \rho_j \le 0$); c_j denotes a positive, constant coefficient (the strength of the penalty). Let us note that the configuration set $\{q :\ 0 < c^j \le \rho_j\}$ determines a *safety zone* around the j-th active obstacle, $j = 1 : N'$.

In order to escape from the obstacle interaction zones, we propose the following control law:

$$u = -k_0 C^T J^T g(e) - k_1 g(s) + k_2 C^T \frac{\partial S}{\partial q} - C^T \frac{\partial U}{\partial q}, \tag{2.18}$$

whose asymptotic stability may be proven similarly as in the case of Theorem 2.2, defining now the Lyapunov function candidate as $V_{hc}(e,\ s,\ q) = k_0 \int_0^e \langle g(x),\ dx\rangle + \frac{1}{2}\langle s,\ s\rangle + k_2(S_{\max} - S) + U$ and assuming the linear independence of the vectors $k_0 C^T J^T g(e)$, $k_2 C^T \frac{\partial S}{\partial q}$, and $C^T \frac{\partial U}{\partial q}$.

2.4 Computer Example

Based on the two selected mobile manipulator tasks, this section demonstrates the performance of the controllers given by Eqs. (2.12) and (2.18). For this purpose, a mobile manipulator, schematically shown in Fig. 2.1, is considered. In all numerical simulations SI units are used. The holonomic part (a SCARA type stationary manipulator) with two revolute kinematic pairs ($n = 2$), operating for simplicity of computations in two-dimensional task space ($m = 2$), is mounted on a unicycle. The unicycle is assumed to be physically driven by two wheels of angular velocities $(\dot{\varphi}_1 \ \dot{\varphi}_2)^T$. The vector of the generalized coordinates q takes the form $q = (x_{1,c} \ x_{2,c} \ \theta \ \varphi_1 \ \varphi_2 \ y_1 \ y_2)^T$. Without loss of generality, the simplified form of the mobile manipulator dynamic equations (2.11) is taken for simulation with $s = (\alpha_1 \ \alpha_2 \ \dot{y}_1 \ \dot{y}_2)^T$. The initial configuration and velocity of the mobile manipulator equal $q(0) = (0 \ 2 \ 0 \ 0 \ 0 \ 0 \ \pi/2)^T$, $\dot{q}(0) = (0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0)^T$, respectively. Parameters R and W are equal to $R = 0.2$, $W = 0.15$; $l_1 = l_2 = 0.4$. Controller gains k_i, where $i = 0, \ 1, \ 2$, take the following values: $k_0 = 0.7$, $k_1 = 20$, and $k_2 = 50$. The task is to attain the desired location $p_f = (1.25, \ -1)^T$ by the end-effector.

First, the mobile manipulator motion in a constraint-free task space is considered. The result of a computer simulation for controller (2.12) shows that the holonomic manipulability measure S is equal approximately to 0 in the whole time interval of the simulation.

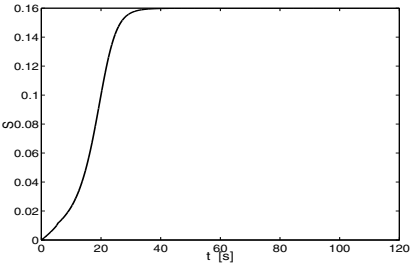


Fig. 2.2 Holonomic manipulability measure S for controller (2.18) in the task space with the obstacle

Next, the mobile manipulator considered in this section is used to solve the same point-to-point control problem as in the first experiment with the same initial configuration and velocity and by the assumption that there is now one obstacle (circle), schematically presented in Fig. 2.1. The boundary of the obstacle takes the form $(x_1 - 1)^2 + (x_2 - 1.5)^2 - 0.45^2 = 0$. In order to accomplish the collision avoidance task, the controller (2.18) is applied in this experiment. The length of nonholonomic platform equals $2L = 0.6$. For simplicity of simulation, we do not take into account the platform wheels in the collision avoidance. Hence, the total number of active collision avoidance constraints N', which are assumed herein to be distance functions, fulfils the inequality $N' \leq 1$ (c^j stands for the distance between the mobile manipulator and the j-th obstacle, $j = 0, \ 1$). In order to calculate numerically

the values of the functions c^j, each link of the holonomic part is discretized into 3 points. The same discretization is carried out for each side of the nonholonomic platform. Following the ideas presented in [14], we introduce for the platform point x_p (see Fig. 2.1) the additional penalty function $U^{'}(x_p) = \sum_{j=1}^{N^{'}} c_j^{'} E(c^j)$, where $c_j^{'}$ is a positive, constant coefficient (the strength of the penalty). The role of the penalty terms U and U' is to ensure the escape of the mobile manipulator from the interaction zones of the circular obstacles. Consequently, the settings include $c_1 = 3 \cdot 10^{-3}$, $c_1^{'} = 9.6 \cdot 10^{-3}$. The threshold value ρ_1 taken for computations is equal to $\rho_1 = 0.8$. Let us note that $U(q(0)) + U^{'}(x_p(0)) > 0$ for chosen ρ_1. The chosen values of c_j and $c_j^{'}$ result in deep penetration of the safety zone but provide mild manipulator movement, which is a desirable property. The results of the computer simulation are presented in Figs. 2.2 and 2.3. From Fig. 2.2 it follows that the controller (2.18) indeed generates singularity-free configurations. Moreover, Fig. 2.3 presents distance d_{mbm-1} between the mobile manipulator and the centre of the obstacle 1 for the controller (2.18). As can be seen from Fig. 2.3, the manipulator collision-freely penetrates the safety zone of the obstacle 1 for t belonging approximately to interval $[0,\ 55]$. Furthermore, we have carried out the simulation for the controller (2.12). The numerical results show that the application of the controller (2.12) would result in a collision for $t \in [0.1,\ 0.4]$.

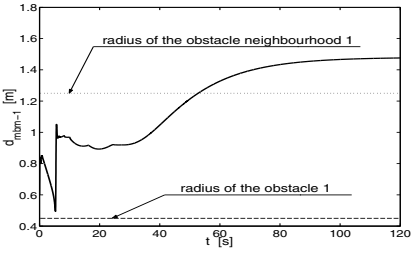


Fig. 2.3 Distance d_{mbm-1} between the mobile manipulator and the centre of the obstacle 1 for the controller (2.18)

2.5 Conclusions

In this paper the point-to-point control task subject to state equality and inequality constraints has been discussed. Using the Lyapunov stability theory, a class of controllers generating the mobile manipulator trajectory fulfilling the state constraints has been derived. The control schemes proposed in this paper generate continuous and bounded controls (even near the boundaries of obstacles), which is a desirable property in on-line control. Moreover, the approach presented forces the mobile manipulator trajectory to escape both from singular configurations and from obstacle neighbourhoods, which makes our control algorithm practically useful. The advantage of using the method proposed is the possibility to implement it in an on-line point-to-point control problem with singularity and collision avoidance. Our future

research will focus on application of the controller (2.18) to singularity and collision avoidance tasks with moving obstacles.

Acknowledgements. This work was supported by the DFG Ga 652/1–1,2.

References

1. Galicki, M.: Inverse kinematics solution to mobile manipulators. Int. J. Robotics Res. 22(12) (2003)
2. Desai, J.P., Kumar, V.: Nonholonomic motion planning for multiple mobile manipulators. In: Proc. IEEE Conf. Robotics Automat., pp. 3409–3414 (1997)
3. Mohri, A., Furuno, S., Iwamura, M., Yamamoto, M.: Sub-optimal planning of mobile manipulator. In: Proc. IEEE Conf. Robotics Automat., pp. 1271–1276 (2001)
4. Huang, Q., Tanie, K., Sugano, S.: Coordinated motion planning for a mobile manipulator considering stability and manipulation. Int. J. Robotics Res. 19(8), 732–742 (2000)
5. Hootsmans, N.A.M., Dubowsky, S.: Large motion control of mobile manipulators including vehicle suspension characteristics. In: Proc. IEEE Conf. Robotics Automat., pp. 2336–2341 (1991)
6. Yamamoto, Y., Yun, X.: Effect of the dynamic interaction on coordinated control of mobile manipulators. IEEE Trans. Robotics Automat. 12(5), 816–824 (1996)
7. Mazur, A.: New approach to designing input-output decoupling controllers for mobile manipulators. Bull. of the Polish Academy of sciences Tech. Sci. 53(1), 31–37 (2005)
8. Tzafestas, C.S., Tzafestas, S.G.: Full-state modelling, motion planning and control of mobile manipulators. Studies in Informatics and Control 10(2) (2001)
9. Padois, V., Fourquet, J.-Y., Chiron, P.: Kinematic and dynamic model-based control of wheeled mobile manipulators: a unified framework for reactive approaches. Robotica, 1–17 (2007)
10. An, C.H., Atkenson, C.G., Hollerbach, J.M.: Model-based control of a robot manipulator. Mit Press, Cambridge (1988)
11. Renders, J.M., Rossignal, E., Becquet, M., Hanus, R.: Kinematic calibration and geometrical parameter identification for robots. IEEE RA 7(6), 721–732 (1991)
12. Yoshikawa, T.: Manipulability of robotic mechanisms. Int. J. Robotics Res. 4(2), 3–9 (1985)
13. Galicki, M.: Task space control of mobile manipulators. Robotica 29, 221–232 (2011)
14. Galicki, M.: Collision-free control of mobile manipulators in a task space. Mechanical Systems and Signal Processing (2010), doi:101016/j.ymssp.2010.11.003
15. Lewis, F.L., Abdallach, C.T., Dawson, D.M.: Control of robotic manipulators. Macmillan, New York (1993)
16. Gilbert, E.G., Johnson, D.W.: Distance functions and their application to robot path planning. IEEE J. Robotics and Automation 1, 21–30 (1985)
17. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile manipulators. Int. J. Robotics Res. 5(1), 90–98 (1986)
18. Krstic, M., Kanellakopoulos, I., Kokotovic, P.: Nonlinear and adaptive control design. J. Wiley and Sons, New York (1995)

Chapter 3
Control Methods for Wheeeler – The Hypermobile Robot

Grzegorz Granosik and Michał Pytasz

Abstract. In this paper we compare two methods for synchronizing movements of segments of the hypermobile robot Wheeeler. Hypermobile robots are a subset of articulated mobile machines and one of the most important problems with controlling them is appropriate coordination of multiple actuators. One of the most popular methods to do this job is the follow-the-leader approach; another possible method is adaptation of the *n*-trailer kinematics. We present some details of the latter approach and then show a few tests realized in a simulation environment. Unlike in other papers, our robot is controlled in the teleoperation mode instead of following a predefined trajectory. Nevertheless, we propose appropriate measures to compare the results of several simulation tests.

3.1 Introduction

Hypermobile robots, which are a subcategory of redundant, articulated body mobile robots, present a very interesting field of research. Common features of hypermobile robots are actuated wheels or tracks, multiple joints, and relatively large length compared to width (or cross-section area). The most significant designs of this kind are [11, 22, 17, 20, 14, 7, 8, 21, 12, 13, 23, 6], while an extended review and comparison of constructions of hypermobile robots can be found in [9]. However, the usefulness and functionality of a robot is gauged by the simplicity of its control system. Especially rescue robots have to be ready for a mission in a short time and the operator has to focus on searching victims instead of struggling with a complicated steering system. Therefore, the most important requirement in designing a hypermobile robot is the need for synchronization of every actuated element during the robot's movements. As we observed from the literature review, most of the

Grzegorz Granosik · Michał Pytasz
Institute of Automatic Control, Technical University of Łódź,
ul. Stefanowskiego 18/22, 90-924 Łódź, Poland
e-mail: granosik@p.lodz.pl, mpytasz@swspiz.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 27–37.
springerlink.com

hypermobile robots presented to date lack autonomy or intuitive teleoperation, or this autonomy is limited to a very specific environment of operation [12, 20]. Although, every robot has some control system but in most cases they employ multi DOF joysticks [17] or sophisticated user interfaces [2]. The main motivation for our Wheeeler project was designing the most intuitive control system for hypermobile robots. We have proposed a computer aided controller and a popular gamepad as the user console [18, 19]. The crucial element of the steering system is the method for synchronization of the behavior of multiple actuators working in the hypermobile robot.

In our first approach the movements of the first segment teleoperated by a human (who directly sets the speed of the wheels and the joint's angle) were copied to the following segments as the robot advanced forward. Therefore, all modules repeat the pattern of the first module – the leader – at the exact same spatial position as the leader module. This approach is well known under the name of follow-the-leader and was introduced to mobile robotics by Choset and Henning [5]. One key problem with this algorithm is measuring the current speed of the robot. To improve the behavior of the propagation algorithm in a rough terrain – such as debris, where wheels can bounce on the surface, we have proposed a basic sensor fusion algorithm [19]. This algorithm assumes that if some of the following segments form a straight line, measurements obtained from these segments (namely encoder and accelerometer readings) should be comparative, therefore they are grouped and the sensors' readings are averaged. In the situation when groups are degraded to one segment and acceleration readings are above threshold we assume larger belief (70%) to encoder readings.

There is, however, another possibility of controlling hypermobile robots – adopting kinematics of a system containing a tractor and n passive trailers, which was applied probably in only one other hypermobile robot – PIKo [6]. In the following section we will present detailed analysis of adaptation of *n*-trailer behavior to a hyper redundant articulated mobile machine. Then simulation tests comparing these two methods will be shown.

3.2 Module Coordination by *n*-trailer Method

The *n*-trailer problem is related to the steering of a truck pulling n trailers linked behind it. Physically it is similar to a wheeled multi-joint robot (schematically shown in Fig. 3.1), but with some important differences. The *n*-trailer system has only one active module, the truck that can change its orientation and velocity, while all the trailers are passive. In contrast, the Wheeeler has several modules with the joints as well as the wheels active. The *n*-trailer model is purely kinematic and is mainly based on the non-holonomic constraints introduced by the wheels that should roll without slippage. Even though the Wheeeler is fully actuated and some dynamic constraints appear, we can consider this robot as a general *n*-trailer system with off-axle hitching but control it in a slightly different way, as explained later in this section.

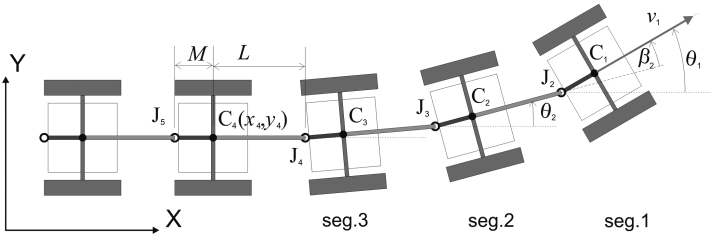


Fig. 3.1 Schematic of the Wheeeler as a general n-trailer system. This top-view shows only active joints (indicated by letter J) working in the horizontal plane

According to the schematic showing a few front segments of the Wheeeler (Fig. 3.1) and following the analysis performed by Altafini [1], we can express the rigid-body constraints as

$$x_{i+1} = x_i - L \cdot \cos\theta_{i+1} - M \cdot \cos\theta_i, \tag{3.1}$$

$$y_{i+1} = y_i - L \cdot \sin\theta_{i+1} - M \cdot \sin\theta_i \tag{3.2}$$

and a non-holonomic constraint by the following equation

$$\dot{x}_i \cdot \sin\theta_{i+1} - \dot{y}_i \cdot \cos\theta_i = 0\,, \tag{3.3}$$

where: x_i and y_i – are the Cartesian coordinates of the point C_i – the center of the axle in the i-th module, θ_i – is the orientation of the i-th module in the base coordinate frame, $\dot{x}_i$ and $\dot{y}_i$ – are the components of v_i – the linear velocity of the i-th segment, L and M – express the respective distances between the C_i and the proceeding (J_i) or the following (J_{i+1}) yaw joints. Using Eqs. (3.1)–(3.3) and a simple relation for the joint angle $\beta_{i+1} = \theta_i - \theta_{i+1}$, we obtain the following recursive equations for the change of the joint angle $\dot{\beta}_{i+1}$ as a function of $\dot{\theta}_i$ and v_i:

$$\dot{\theta}_{i+1} = \frac{v_i \cdot sin\beta_{i+1}}{L} - \frac{M \cdot cos\beta_{i+1} \cdot \dot{\theta}_i}{L}, \tag{3.4}$$

$$\dot{\beta}_{i+1} = \dot{\theta}_i - \dot{\theta}_{i+1} \tag{3.5}$$

and the recursive equation for the linear velocity of module $(i+1)$:

$$v_{i+1} = v_i \cdot cos\beta_{i+1} + M \cdot sin\beta_{i+1} \cdot \dot{\theta}_i, \tag{3.6}$$

all valid for $i = 1 \ldots n-1$, ($n = 7$ in the case of the Wheeeler).

The general n-trailer system has two physical inputs, namely v_1 and $\dot{\theta}_1$, corresponding to translational and steering actions of the car pulling the trailers. All other modules follow this car passively according to Eqs. (3.4)–(3.6). In the case of

a multi-joint mobile robot imitating the behavior of the n-trailer all modules are active. We can set the speeds v_i (for $i = 1 \ldots n$) and the joint angles β_i (for $i = 2 \ldots n$). The only problem is with the steering actions $\dot{\theta}_1$ of the first segment, which has only one axle and no differential drive on the wheels, and therefore cannot change its heading self-supporting. However, we can assume symmetry in the system and follow the explanation of Murugendran et al. [16]: if we set the individual modules' speeds and joint angles according to the calculated values of v_i and $\dot{\beta}_i$, the articulated mobile robot will move as if it actually was being pulled by the virtual truck. Instead of the steering being the result of a steerable axle in the truck, it will emerge as a result of the collective actions of all modules. As a bonus, since the trailers of the n-trailer system behave in a passive manner, this control scheme should result in an equivalent passive behavior in the Wheeeler. This means that in the ideal case, the slip of the wheels will be eliminated. Therefore, using readings from the joystick in a different way than in the follow-the-leader approach, namely as virtual speed v_1 and heading angle $\dot{\theta}_1$, we can calculate joint angles and speeds of wheels for all other modules of Wheeeler according to Eq. (3.5) and (3.6), respectively. This control rule was tested on the model of Wheeeler in the simulation environment as reported in the next section.

3.3 Simulation Tests

In order to test different functionalities of the Wheeeler's control system we prepared a special simulation environment (defined in the program Webots PRO) containing stairs, rubbles, narrow passages, and the robot itself, as shown in Fig. 3.2. We tested the following assistance-features: joint position propagation (horizontal or vertical) using the follow-the-leader approach, vertical joint locking, linear velocity correction (2 algorithms), simple obstacle avoidance based on side distance sensors, predefined robot poses, as reported in [10].

In a special set of tests we tried to compare two proposed methods for coordination of segment movements: follow-the-leader and n-trailer. We ran these tests in the same simulation environment, but the robot was steered along a shorter path, shown in the snapshots in Fig. 3.3. A detailed explanation of twelve tests is presented in Table 3.1. The operator orders the speed of the wheels in the front segment on the level of 11 or 16 cm/s, and controls the turn of the robot according to the methods described earlier. One part of the robot controller (the console) reads information from the gamepad used by the operator to steer the Wheeeler, visualizes data received from the robot, and communicates (using CORBA) with the other part of the controller working directly in the simulation environment [10]. The main controller calculates the current values for the speeds of the wheels and the positions of the joints in all the segments of the Wheeeler using one of the coordination algorithms. These values are sent to the model of the robot in each step of the simulation using appropriate Webots methods. In each step there are also read new values from sensors: wheel encoders, joint rotary sensors, and accelerometers located in every

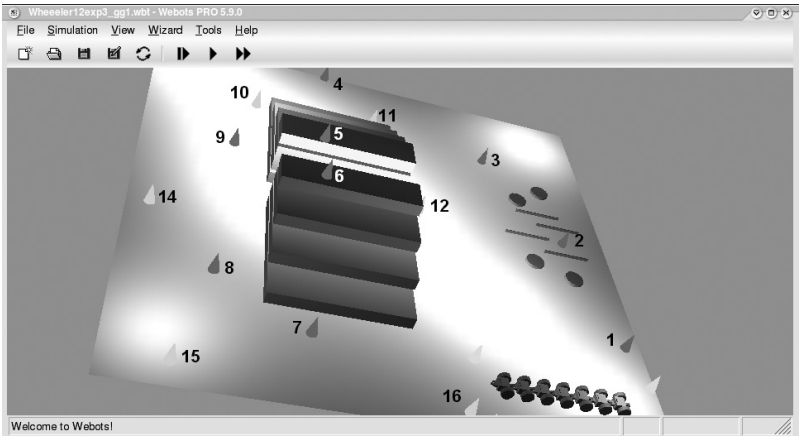


Fig. 3.2 Special test path in the simulation environment: 1-3 rubble, 4-5 staircase going up, 6-7 staircase going down, 7-12 tight turns, 12-14 going through narrow corridor, 15-16 final position

segment. They are used in the control algorithm and they are sent to the console. Additionally, the robot supervisor (running in parallel with the main controller) records the spatial position of the first segment of the robot and the relative orientation of the next segments in every simulation step. These data are used to compare the quality of both coordination methods. In most of the experiments the robot is driven forward; in tests #11 and #12 the robot is driven backward and follows the path clockwise. To check the behavior of the coordination methods on rough terrain in some experiments the robot was driven through the model of the rubble.

Table 3.1 Description of tests comparing different coordination methods: FTL – follow-the-leader, NT – *n*-trailer

Test	Description of the test			
No.	Coordination	Speed [cm/s]	Terrain	Path (see Fig. 3.2)
1	FTL	11	Flat	1-12-13-14-15-16
2	FTL	11	Flat	1-12-13-14-15-16
3	FTL	16	Flat	1-12-13-14-15-16
4	FTL	11	Rubble	1-2-12-13-14-15-16
5	FTL	11	Rubble	1-2-12-13-14-15-16
6	NT	11	Flat	1-12-13-14-15-16
7	NT	11	Flat	1-12-13-14-15-16
8	NT	16	Flat	1-12-13-14-15-16
9	NT	11	Rubble	1-2-12-13-14-15-16
10	NT	11	Rubble	1-2-12-13-14-15-16
11	NT	−11	Flat	16-15-14-13-12-1
12	FTL	−11	Flat	16-15-14-13-12-1

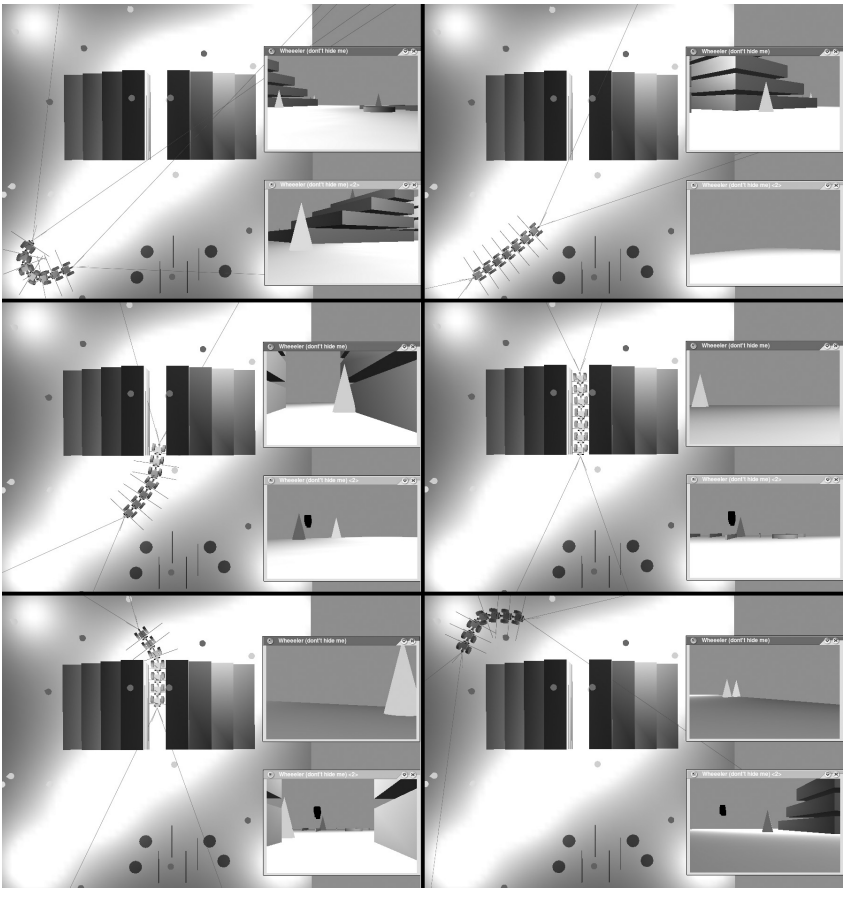


Fig. 3.3 Top-view of the simulation environment – snapshots of the robot path; robot goes counter clockwise: starts from the bottom-left corner of the picture, goes through the passage, turns left, and returns to the starting area; small pictures in the top-left and bottom-left corners in each snapshot contain views from the front and rear camera, respectively – in real application the operator has to control the robot based mainly on similar views

We have to remember that the Wheeeler is intended to be a teleoperator and it does not follow a predefined path but it is always operated by a person, therefore traces of the robot in various tests may differ slightly. Nevertheless, we would like to compare these results by calculating the average deviation between the paths followed by segments 2-7 compared with the path of the first segment. These deviations are measured in each step of the simulation as offset distances (Δ_i) between the path followed by the first segment and the lines traced by the other segments (along the normal to the first line), as shown in Fig. 3.4. Then the mean values along each path (i.e. for segments 2-7) are calculated. To compare the quality of the coordination methods we introduce three indicators: (1) the average

offset distance – Δ and (2) the standard deviation σ_Δ of this value for the entire robot, and (3) the average offset distance related to the length of the path – Lg. The results of the simulations are collected in Table 3.2.

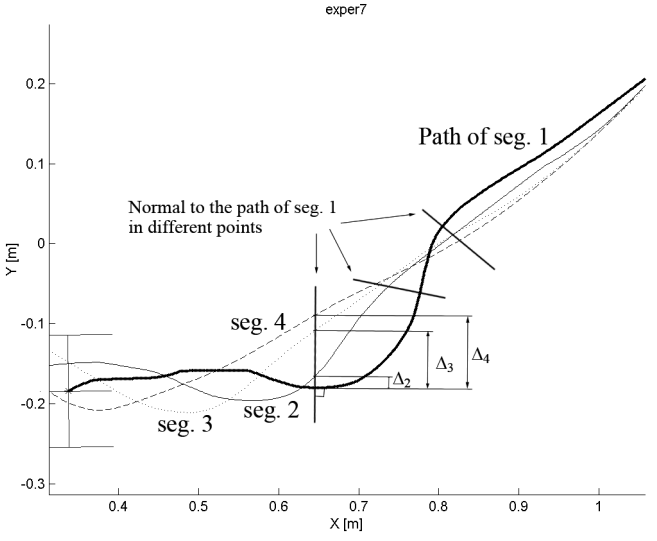


Fig. 3.4 Measuring the offset distance Δ_i for segments i=2,3,4 (part of test #7)

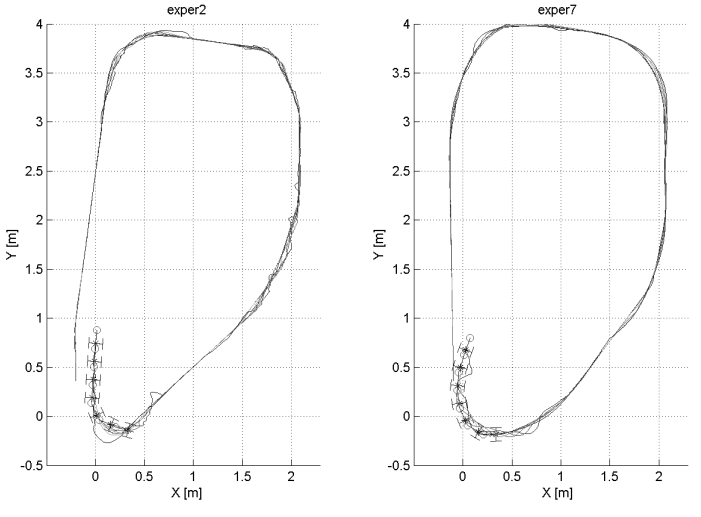


Fig. 3.5 Path of the Wheeeler in test #2 (FTL-method) and test #7 (NT-method)

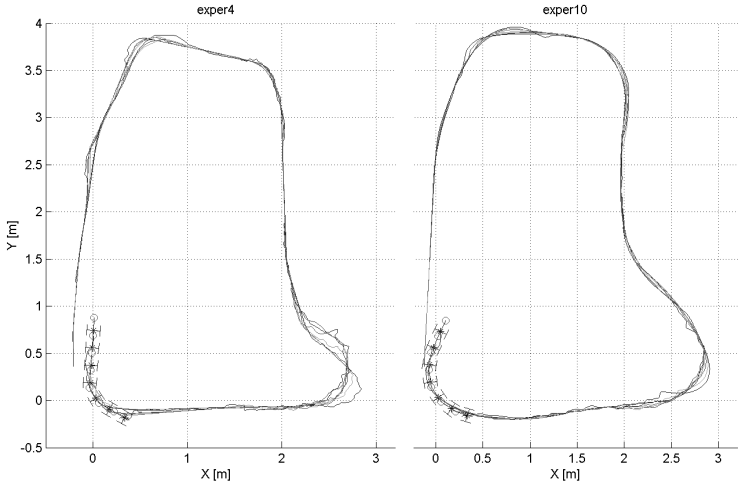


Fig. 3.6 Path of the Wheeeler in test #4 (FTL-method, rubble) and test #10 (NT-method, rubble)

Each part of Figs. 3.5 and 3.6 shows traces of all the Wheeeler's segments presented as color lines on the XY plane. Each pair of the plots is chosen as the comparable example of the tests using the FTL or NT method. In the first pair (tests #2 and #7) we can see the worst case scenario of the runs on the flat terrain with the average speed of 11 cm/s – the robot recorded the largest offset distances Δ and the largest standard deviations σ_Δ (visible on the turns). In the second pair (tests #4 and #10) we show examples of the worst runs through the rubble – we can again observe very large errors on the turn, that was made on the rubble. Much bigger deviations can be seen in the case of FTL coordination. In general, the performed simulations show that:

- The deviation in following the path of the first segment increases with the number of the segment – the further modules (counting from the front one) stray more; from Table 3.2 we can see that the average offset distances for the 7-th segment are about 3 times larger than those measured for the first one,
- The Wheeeler performance under n-trailer control is better than in the case of the follow-the-leader method – the recorded offset distances for the entire robot Δ are smaller by 1 cm in average, while the standard deviations are smaller by 5 cm in average; these numbers give over 40% decrease of the average offset distances and over 70% decrease in the standard deviations.

Table 3.2 Performance of the Wheeeler in tests comparing different coordination methods

Test	Lg	Δ	σ_Δ	Δ/Lg	Average Δ_i for i-th segment [cm]					
No.	[cm]	[cm]	[cm]	$[10^{-3}]$	2	3	4	5	6	7
1	956	2.20	8.64	2.30	1.23	1.60	1.64	2.16	2.59	3.97
2	989	2.39	7.71	2.42	1.37	1.55	1.87	2.38	2.95	4.25
3	978	2.13	4.87	2.17	0.79	1.02	1.40	2.40	2.88	4.26
4	1140	3.15	8.04	2.76	1.27	1.60	2.50	3.67	4.40	5.48
5	1182	2.55	10.28	2.16	1.44	1.81	2.17	2.69	3.13	4.09
6	986	1.33	2.22	1.35	0.72	1.11	1.29	1.37	1.42	2.10
7	1003	1.42	2.98	1.42	0.69	1.00	1.22	1.40	1.57	2.64
8	1005	0.96	1.00	0.96	0.51	0.76	0.95	1.06	1.13	1.37
9	1154	1.79	2.34	1.55	0.82	1.33	1.76	2.13	2.30	2.40
10	1137	1.50	2.22	1.32	0.80	1.02	1.27	1.55	1.79	2.58
11	953	1.62	1.77	1.70	0.79	1.17	1.55	1.83	2.02	2.35
12	945	1.53	2.95	1.62	0.96	1.11	1.26	1.73	1.88	2.23

3.4 Conclusion

Almost all hypermobile robots are prepared to work as teleoperators and usually autonomy is absent or very limited. Although all robots need and have controllers to drive so many actuators they still require human operator to decide on almost any action of the robot. We still did not reach a situation that once global tasks are entered by the user, the robot moves while accomplishing these rescue tasks. What is positive, there are two methods for coordination of movements of several segments of articulated mobile robots, namely: follow-the-leader and n-trailer. Both were applied to the Wheeeler – the model of our hypermobile robot – and tested in the Webots simulation software in teleoperation mode. The same methods were also comprehensively compared by Murugendran et al. [16], who built a dynamic model of PIKo and tested path-following controllers. The follow-the-leader approach has been known for a long time and used in several projects dealing with hyper redundant robots, but proving n-trailer to be a feasible coordination method for hypermobile robot re-opens for new consideration the already proposed approaches to trajectory following like shown in [4, 3, 15].

References

1. Altafini, C.: Some properties of the general n-trailer. Int. Journal of Control 74(4), 409–424 (2001)
2. Baker, J., Borenstein, J.: The Joysnake A Haptic Operator Console for High-Degree-of-Freedom Robots. In: Int. Joint Topical Meeting: Sharing Solutions for Emergencies and Hazardous Environments, Salt Lake City, USA (2006)

3. Bolzern, P., DeSantis, R.M., Locatelli, A., Masciocchi, D.: Path tracking for articulated vehicles with off-axle hitching. IEEE Trans. on Control Systems Technology 6(4), 515–523 (1998)
4. Bushnell, L.G.: An obstacle avoidance algorithm for car pulling trailers with off-axle hitching. In: Proc. of the 34th Conf. on Decision and Control, New Orelans, USA, pp. 3837–3842 (1995)
5. Choset, H., Henning, W.: A follow-the-leader approach to serpentine robot motion planning. ASCE Journal of Aerospace Engineering 12(2), 65–73 (1999)
6. Fjerdingen, S.A., Transeth, A.A., Liljebck, P.: A snake-like robot for internal inspection of complex pipe structures (PIKo). In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, St. Louis, USA, pp. 5665–5671 (2009)
7. Granosik, G., Hansen, M., Borenstein, J.: The OmniTread Serpentine Robot for Industrial Inspection and Surveillance. Industrial Robot: An International Journal, Special Issue on Mobile Robots IR32-2, 139–148 (2005)
8. Granosik, G., Borenstein, J., Hansen, M.G.: Serpentine Robots for Industrial Inspection and Surveillance. In: Huat, L.K. (ed.) Industrial Robotics. Programming, Simulation and Applications, pp. 633–662. pro literatur Verlag, Germany (2007), ISBN 3-86611-286-6
9. Granosik, G.: Hypermobile robots. In: Aschemann, H. (ed.) New Approaches in Automation and Robotics, Austria, pp. 315–332. I-Tech Education and Publishing, Vienna (2008), ISBN 978-3-902613-26-4
10. Granosik, G.: Hypermobile robots: concept, construction and control. Scientific Bulletin of the Technical University of Łódź, No 1088, Wydawnictwo Politechniki Łódzkiej, Łódź (2011), ISSN 0137-4834
11. Hirose, S., Morishima, A.: Design and Control of a Mobile Robot With an Articulated Body. The Int. Journal of Robotics Research 9(2), 99–113 (1990)
12. INSPECTOR SYSTEMS Rainer Hitzel GmbH, http://www.inspector-systems.com (cited November 11, 2010)
13. Kamegawa, T., Yamasaki, T., Igarashi, H., Matsuno, F.: Development of the Snake-like Rescue Robot KOHGA. In: Proc. IEEE Int. Conf. on Robotics and Automation, New Orleans, USA, pp. 5081–5086 (2004)
14. Kimura, H., Hirose, S.: Development of Genbu: Active wheel passive joint articulated mobile robot. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and System, vol. 1, pp. 823–828 (2002)
15. Michałek, M., Kozłowski, K.: VFO Tracking and Set-Point Control for ans Articulated Vehicle with Off-Axle Hitched Trailer. In: Proc. of the European Control Conference, Budapest, Hungary, pp. 266–271 (2009)
16. Murugendran, B., Transeth, A.A., Fjerdingen, S.A.: Modeling and Path-following for a Snake Robot with Active Wheels. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, St. Louis, USA, pp. 3643–3650 (2009)
17. Osuka, K., Kitajima, H.: Development of Mobile Inspection Robot for Rescue Activities: MOIRA. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Las Vegas, USA, pp. 3373–3377 (2003)
18. Pytasz, M., Granosik, G.: Modelling of Wheeeler – hyper mobile robot. In: Tchoń, K. (ed.) Postępy Robotyki. Sterowanie, percepcja i komunikacja, Wydawnictwa Komunikacji i Łc aczno'sci, Warsaw, pp. 275–284 (2006), ISBN: 83-206-1628-X
19. Pytasz, M.: Robot hipermobilny Wheeeler – rozwój algorytmów sterowania. In: Tchoń, K., Zieliński, C. (eds.) Problemy robotyki, Prace Naukowe Politechniki Warszawskiej, Warsaw, vol. 1, pp. 177–184 (2008) (in Polish), ISSN 0137-2343
20. Schempf, H., Mutschler, E., Goltsberg, V., Skoptsov, G., Gavaert, A., Vradis, G.: Explorer: Untethered Real-time Gas Main Assessment Robot System. In: Proc. of Int. Workshop on Advances in Service Robotics, ASER 2003, Bardolino, Italy (2003)

21. Streich, H., Adria, O.: Software approach for the autonomous inspection robot MAKRO. In: Proc. IEEE Int. Conf. on Robotics and Automation, New Orleans, USA, pp. 3411–3416 (2004)
22. Takayama, T., Hirose, S.: Development of Souryu-I connected crawler vehicle for inspection of narrow and winding space. In: 26th Annual Conf. of the IEEE Industrial Electronics Society, IECON, vol. 1, pp. 143–148 (2000)
23. Zhang, H., Wang, W., Deng, Z., Zong, G., Zhang, J.: A Novel Reconfigurable Robot for Urban Search and Rescue. Int. Journal of Advanced Robotic Systems 3(4), 359–366 (2006)

Chapter 4
Tracking Control Strategy for the Standard N-trailer Mobile Robot – A Geometrically Motivated Approach

Maciej Michałek

Abstract. The paper is devoted to a novel feedback control strategy for the standard N-trailer robot kinematics expected to perform a tracking task for feasible reference trajectories. The control method proposed results from solely geometrical features of the vehicle kinematics formulated in a cascaded-form, and especially from the way the velocity components propagate along a vehicle kinematic chain. The control strategy is derived for the original vehicle configuration space not involving any model transformations or approximations. Formal considerations are examined by simulation results of backward tracking maneuvers for a 3-trailer vehicle.

4.1 Introduction

Nonholonomic articulated mobile robots consisting of an active tractor linked to N passive trailers, although fully controllable [8], are especially demanding for control. Difficulties result from three main reasons: the less number of control inputs in comparison to the number of controlled variables in the presence of nonintegrable motion constraints, singularities of the vehicle kinematic model [5], and a vehicle folding effect arising during backward maneuvers leading to the so-called jack-knifing phenomenon. The kinematic structure of the N-trailer vehicle can be treated as an equivalent skeleton of many practical vehicles. Manual maneuvers with articulated vehicles are difficult, even for experienced drivers, thus automation of motion control for such vehicles seems to be desirable in practice [15]. Examples of tracking control laws for multiple-trailer robots can be found in [10, 2] and for vehicles with a single trailer in [7, 11]. The most popular approach to control design relies on auxiliary transformation of the vehicle model into a chained form, which is possible for the standard N-trailer system due to its differential flatness

Maciej Michałek
Chair of Control and Systems Engineering, Poznań University of Technology,
ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: maciej.michalek@put.poznan.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 39–51.
springerlink.com

property, [13, 12, 1, 14]. Since the transformation is only locally well defined, utilization of control solutions based on the chained form approximation may be limiting in applications.

In this paper we present an alternative tracking control strategy for the *standard N-trailer mobile robot* composed of a unicycle-like tractor followed by N passive trailers hooked at a mid-point of the preceding wheel-axle (see Fig. 4.1). The proposed approach, resulting from geometrical arguments, does not involve any state or input transformations; it is formulated in the original configuration space of the controlled vehicle. The resultant control law has a cascaded structure with clearly interpretable components. As a consequence, tunning the controller parameters is simple yielding non-oscillatory vehicle motion in the task space. The concept is formulated using the Vector-Field-Orientation (VFO) control approach for the last vehicle trailer, allowing one to solve the tracking task for the whole set of feasible and persistently exciting reference trajectories (see [4] and also [3]).

4.2 Vehicle Model and the Control Task

The kinematic skeleton of the articulated vehicle under consideration is presented in Fig. 4.1. The vehicle consists of $N+1$ segments: a unicycle-like tractor (the active segment) followed by N semi-trailers (the passive segments) of lengths $L_i > 0,\ i = 1,\ldots,N$, where every trailer is hitched with a rotary joint located exactly on the axle mid-point of the preceding vehicle segment. The configuration of the vehicle can be represented by vector $\boldsymbol{q} = [\beta_1\ \ldots\ \beta_N\ \theta_N\ x_N\ y_N]^T \in \mathbb{R}^{N+3}$ with geometrical interpretation coming from Fig. 4.1. The *guidance point* $P = (x_N, y_N)$ of the vehicle, playing the key role in the tracking control task (defined in Section 4.2), is represented by the position coordinates of the last-trailer posture vector $\overline{\boldsymbol{q}} = [\theta_N\ x_N\ y_N]^T \in \mathbb{R}^3$. The control inputs of the vehicle $\boldsymbol{u}_0 = [\omega_0\ v_0]^T \in \mathbb{R}^2$ are the angular ω_0 and the longitudinal v_0 velocities of the tractor.

Let us formulate the kinematics of the vehicle in a way useful for the subsequent control development. In order to do this, every i-th segment ($i = 0, 1, \ldots, N$) of the vehicle kinematic chain will be described by the unicycle model (cf. Fig. 4.1):

$$\dot{\theta}_i = \omega_i, \qquad \dot{x}_i = v_i \cos\theta_i, \qquad \dot{y}_i = v_i \sin\theta_i. \tag{4.1}$$

The tractor kinematics is obtained by taking $i = 0$ where ω_0 and v_0 are the physical control inputs. For $i = 1, 2, \ldots, N$ the fictitious control inputs ω_i, v_i of the i-th trailer result from two recurrent equations:

$$\omega_i = \frac{1}{L_i} v_{i-1} \sin\beta_i, \qquad v_i = v_{i-1} \cos\beta_i, \tag{4.2}$$

where β_i is the i-th joint angle determined by

$$\beta_i = \theta_{i-1} - \theta_i. \tag{4.3}$$

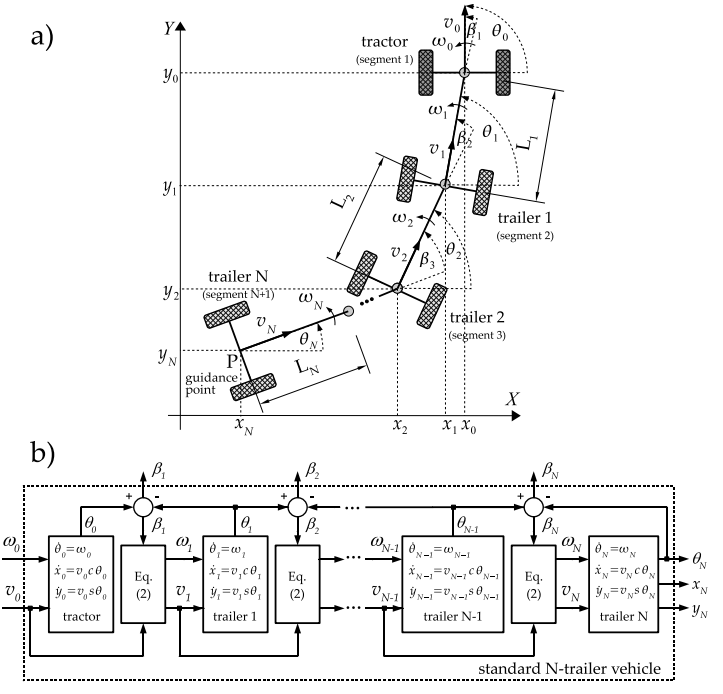


Fig. 4.1 The standard N-trailer vehicle in a global frame (a) and the schematic diagram of the N-trailer vehicle kinematics in a cascaded form with inputs ω_0, v_0 and configuration $\boldsymbol{q}$ (b)

Using Eqs. (4.2)–(4.3) one can describe how the physical inputs ω_0 and v_0 propagate to the i-th trailer along the vehicle kinematic chain. This is explained by the block diagram in Fig. 4.1(b), which represents the general N-trailer vehicle kinematics in the above cascaded-like formulation [12].

Since the guidance point $P = (x_N, y_N)$ has been selected on the last trailer, the control task will be formulated with special attention paid to the last vehicle segment. This selection of the guidance point is theoretically justified, since the coordinates x_N, y_N are the *flat outputs* of the vehicle kinematics [13], and the differential flatness is a key feature implicitly utilized in the subsequent control development.

Let $\boldsymbol{q}_t(\tau) = [\beta_{t1}(\tau)\ \ldots\ \beta_{tN}(\tau)\ \theta_{tN}(\tau)\ x_{tN}(\tau)\ y_{tN}(\tau)]^T \in \mathbb{R}^{N+3}$ denote the reference configuration trajectory of the vehicle. Define the configuration tracking error $\boldsymbol{e}(\tau) = \left[\boldsymbol{e}_\beta^T(\tau)\ \overline{\boldsymbol{e}}^T(\tau)\right]^T \triangleq \boldsymbol{q}_t(\tau) - \boldsymbol{q}(\tau)$ with the joint-angle tracking error component

$$\boldsymbol{e}_\beta(\tau) \triangleq [\beta_{t1}(\tau) - \beta_1(\tau)\ \ldots\ \beta_{tN}(\tau) - \beta_N(\tau)]^T \in \mathbb{R}^N \tag{4.4}$$

and the last-trailer posture tracking error component

$$\overline{\boldsymbol{e}}(\tau) = \begin{bmatrix} e_\theta \\ e_x \\ e_y \end{bmatrix} \triangleq \overline{\boldsymbol{q}}_t(\tau) - \overline{\boldsymbol{q}}(\tau) = \begin{bmatrix} \theta_{tN}(\tau) - \theta_N(\tau) \\ x_{tN}(\tau) - x_N(\tau) \\ y_{tN}(\tau) - y_N(\tau) \end{bmatrix} \in \mathbb{R}^3. \tag{4.5}$$

Assuming that:

A1 the reference trajectory $\boldsymbol{q}_t(\tau)$ is feasible (meets the vehicle kinematics for all $\tau \geq 0$) and is persistently exciting, namely $\dot{x}_{tN}^2(\tau) + \dot{y}_{tN}^2(\tau) \neq 0$ for all $\tau \geq 0$,
A2 all components of the configuration $\boldsymbol{q}$ are measurable,
A3 all the vehicle kinematic parameters L_i are known,

the control problem is to design a feedback control law $\boldsymbol{u}_0 = \boldsymbol{u}_0(\boldsymbol{q}_t(\tau), \boldsymbol{q}(\tau), \cdot)$ which applied to the vehicle kinematics represented by (4.1)–(4.3) makes the error $\boldsymbol{e}(\tau)$ convergent in the sense that:

$$\lim_{\tau \to \infty} \|\overline{\boldsymbol{e}}(\tau)\| \leq \delta_1 \quad \text{and} \quad \lim_{\tau \to \infty} \left\|\boldsymbol{e}_\beta(\tau)\right\| \leq \delta_2, \tag{4.6}$$

with τ denoting the time variable, and $\delta_1, \delta_2 \geq 0$ – the vicinities of zero in the particular tracking error spaces. In the paper two cases of asymptotic ($\delta_1 = \delta_2 = 0$) and *practical* ($\delta_1, \delta_2 > 0$) convergence will be illustrated.

4.3 Feedback Control Strategy

The proposed control strategy for the standard N-trailer robot results from Eqs. (4.2), which describe propagation of the tractor velocities to the particular trailers along the vehicle kinematic chain. In order to explain our concept let us begin from the last vehicle trailer, where the guidance point P is located. We can make a thought experiment where the N-th trailer is separated from the remaining vehicle chain and treated as a unicycle-like vehicle with control inputs ω_N, v_N (cf. Fig. 4.1). Furthermore, let us assume that feedback control functions $\omega_N := \Phi_\omega(\overline{\boldsymbol{e}}(\tau), \cdot)$ and $v_N := \Phi_v(\overline{\boldsymbol{e}}(\tau), \cdot)$ (defined in Section 4.3.2) which ensure asymptotic convergence of the tracking posture error $\overline{\boldsymbol{e}}(\tau)$ for the unicycle model (4.1) with $i := N$ are given.

Formulating the control strategy we need to answer the question how one can force the control signals $\omega_N := \Phi_\omega$ and $v_N := \Phi_v$ in the case when the N-th trailer is not driven directly, but is passive and linked to the whole kinematic chain driven only by the tractor inputs ω_0, v_0. One can answer the question utilizing the fact that the motion of the N-th trailer is a direct consequence of the motion of the $(N-1)$-st trailer. Thus designing the appropriate desired motion for the latter can help forcing the desired control action determined by functions Φ_ω and Φ_v for the N-th trailer. Relations (4.2) reveal how to design the fictitious input v_{N-1} of the $(N-1)$-th trailer and the joint angle β_N and, as a consequence, the design of the fictitious input ω_{N-1} can be devised (it will be formalized in the next subsection). Again, the fictitious inputs v_{N-1}, ω_{N-1} cannot be forced directly, but only through the $(N-2)$-nd vehicle segment. By analogous reasoning for the whole kinematic chain (where the i-th segment influences the motion of the $(i+1)$-st), one can derive the control law for physically available tractor inputs ω_0, v_0, which should accomplish the desired motion of the last trailer (the *guiding* segment).

4.3.1 Control Law Design

Denote by ω_{di} and v_{di} $(i = 1,\ldots,N)$ the desired fictitious inputs which the i-th vehicle segment should be forced with in order to execute the desired motion of the $(i+1)$-st segment. The formulas which determine the desired longitudinal velocity v_{di-1} for the $(i-1)$-st vehicle segment and the desired value for the i-th joint angle can be obtained by combination of relations (4.2), namely:

$$v_{di-1} \triangleq L_i \omega_{di} \sin\beta_i + v_{di} \cos\beta_i, \tag{4.7}$$

$$\beta_{di} \triangleq \mathrm{Atan2c}\,(L_i \omega_{di} \cdot v_{di-1}, v_{di} \cdot v_{di-1}) \in \mathbb{R}, \tag{4.8}$$

where $\mathrm{Atan2c}\,(\cdot,\cdot) : \mathbb{R}\times\mathbb{R} \mapsto \mathbb{R}$ is a continuous version of the four-quadrant function $\mathrm{Atan2}\,(\cdot,\cdot) : \mathbb{R}\times\mathbb{R} \mapsto (-\pi,\pi]$ (it has been introduced to ensure continuity of β_{di} signals[1]), and the term v_{di-1} used in (4.8) determines the appropriate sign of the two function arguments. Whereas Eq. (4.7) determines one of the desired fictitious inputs of the $(i-1)$-st segment, the desired angular fictitious input ω_{di-1} remains to be determined. To do this let us differentiate relation (4.3) and utilize (4.1) to obtain

$$\dot{\beta}_i = \omega_{i-1} - \omega_i =: \nu_i. \tag{4.9}$$

The above relation may suggest a definition of ν_i to make the auxiliary joint angle error

$$e_{di} \triangleq (\beta_{di} - \beta_i) \in \mathbb{R} \tag{4.10}$$

converge to zero. By taking $\nu_i \triangleq k_i e_{di} + \dot{\beta}_{di}$ with $k_i > 0$ and $\dot{\beta}_{di} \equiv d\beta_{di}/d\tau$, one gets the differential equation $\dot{e}_{di} + k_i e_{di} = 0$, which implies the exponential convergence $e_{di}(\tau) \to 0$ as $\tau \to \infty$. According to (4.9), the desired angular velocity for the $(i-1)$-st vehicle segment can be defined as follows:

$$\omega_{di-1} \triangleq \nu_i + \omega_{di} \triangleq k_i e_{di} + \dot{\beta}_{di} + \omega_{di}, \tag{4.11}$$

where $k_i > 0$ is now a control design coefficient, and $\dot{\beta}_{di}$ plays a role of the feedforward component.

Definitions (4.7), (4.8), and (4.11) constitute the so-called i-th Single Control Module (SCM_i) explained by the schematic diagram in Fig. 4.2, where the feedback from the joint angle β_i is denoted. A serial connection of SCM_i blocks allows propagating the computations of the desired velocities between an arbitrary number of vehicle segments. The recurrent relations formulated in (4.7), (4.8), and (4.11) are iterated from $i = N$ to $i = 1$, starting from the last trailer by taking $\omega_{dN} := \Phi_\omega(\overline{\boldsymbol{e}}(\tau),\cdot)$, $v_{dN} := \Phi_v(\overline{\boldsymbol{e}}(\tau),\cdot)$ and finishing on the tractor segment obtaining the control inputs $\omega_0 := \omega_{d0}(\cdot)$, $v_0 := v_{d0}(\cdot)$. The equations of the resultant feedback controller for the standard N-trailer vehicle are formulated as follows:

[1] We refer to [4] for computation details of the $\mathrm{Atan2c}\,(\cdot,\cdot)$ function.

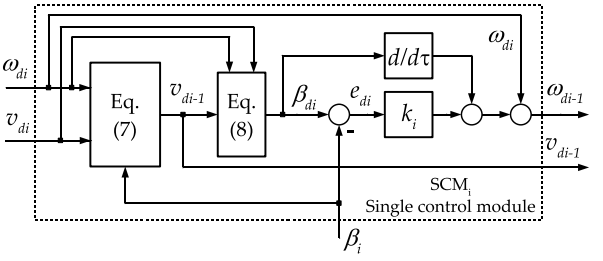


Fig. 4.2 Schema of the i-th Single Control Module (SCM$_i$) with a feedback from joint angle β_i

$$v_{dN} := \Phi_v(\overline{\boldsymbol{e}}(\tau),\cdot) \tag{4.12}$$
$$\omega_{dN} := \Phi_\omega(\overline{\boldsymbol{e}}(\tau),\cdot) \tag{4.13}$$
$$v_{dN-1} := L_N\omega_{dN}\sin\beta_N + v_{dN}\cos\beta_N \tag{4.14}$$
$$\beta_{dN} := \mathrm{Atan2c}\,(L_N\omega_{dN}\cdot v_{dN-1}, v_{dN}\cdot v_{dN-1}) \tag{4.15}$$
$$\omega_{dN-1} := k_N(\beta_{dN}-\beta_N) + \dot\beta_{dN} + \omega_{dN} \tag{4.16}$$
$$\vdots$$
$$v_{d1} := L_2\omega_{d2}\sin\beta_2 + v_{d2}\cos\beta_2 \tag{4.17}$$
$$\beta_{d2} := \mathrm{Atan2c}\,(L_2\omega_{d2}\cdot v_{d1}, v_{d2}\cdot v_{d1}) \tag{4.18}$$
$$\omega_{d1} := k_2(\beta_{d2}-\beta_2) + \dot\beta_{d2} + \omega_{d2} \tag{4.19}$$
$$v_0 = v_{0d} := L_1\omega_{d1}\sin\beta_1 + v_{d1}\cos\beta_1 \tag{4.20}$$
$$\beta_{d1} := \mathrm{Atan2c}\,(L_1\omega_{d1}\cdot v_{d0}, v_{d1}\cdot v_{d0}) \tag{4.21}$$
$$\omega_0 = \omega_{d0} := k_1(\beta_{d1}-\beta_1) + \dot\beta_{d1} + \omega_{d1}. \tag{4.22}$$

Eqs. (4.20) and (4.22) express direct application of the desired velocities computed for the vehicle segment number zero to the control inputs of the vehicle tractor. Figure 4.3 illustrates the structure of the proposed cascaded control system, where the Last-Trailer Posture Tracker (LTPT) block represents the tracking control module designed for the unicycle kinematics of the last trailer. This block computes the feedback control functions $\Phi_\omega(\overline{\boldsymbol{e}}(\tau),\cdot)$ and $\Phi_v(\overline{\boldsymbol{e}}(\tau),\cdot)$ used in Eqs. (4.12)–(4.13). Note that according to the defined control task (cf. Section 4.2) the controlled output of the articulated vehicle is the last trailer posture $\overline{\boldsymbol{q}} \in \mathbb{R}^3$, while the angles β_1 to β_N are the auxiliary outputs used in the SCM$_i$ blocks.

The control strategy presented so far generally does not prevent the vehicle folding effect. This is a consequence of definition (4.8), where the desired angles are the real (unbounded) variables. Since this may be limiting in most practical application, we propose to modify definition (4.7) in order to avoid the folding effect by taking:

$$v_{di-1} \triangleq \sigma\,|L_i\omega_{di}\sin\beta_i + v_{di}\cos\beta_i|\,, \tag{4.23}$$

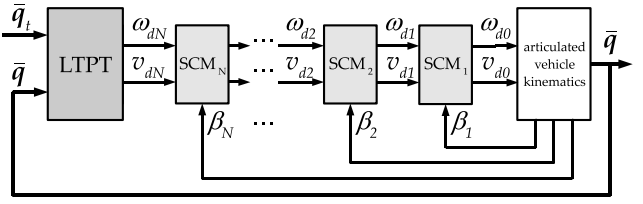


Fig. 4.3 Schema of the proposed cascaded control system for the standard N-trailer vehicle. The controller consists of N Single Control Modules using feedback from joint angles β_i, and the Last-Trailer Posture Tracker (LTPT) block designed for the last trailer treated as the unicycle

where $\sigma \in \{-1,+1\}$ is the decision variable which is inherited from the VFO controller used in the LTPT block and determined in Subsection 4.3.2. By using modification (4.23) and leaving other definitions unchanged, and due to characteristic features of the VFO controller (see [4]), the second argument of Atan2c$(\cdot,\cdot)$ function in (4.8) may now have a constant non-negative sign for almost all time of the vehicle motion. Hence, the image of function Atan2c$(\cdot,\cdot)$ can be limited to the first and fourth quadrant, minimizing the possibility of the folding effect occurence. Summarizing, the only modifications of the controller (4.12)–(4.22) are required for (4.14), (4.17), and (4.20) by replacing them with definition (4.23) using the appropriate indexes.

To accomplish derivation of a feedback controller for the articulated vehicle it remains to determine the feedback functions Φ_v and Φ_ω used by the LTPT block (see (4.12)–(4.13)). Explicit definitions of these functions are given and briefly commented in the next subsection using the original VFO concept.

4.3.2 *Last-Trailer Posture Tracker – The VFO Controller*

The VFO control approach comes from geometrical interpretations related to the unicycle kinematics (4.1). Our selection of the VFO method for the LTPT block is motivated by the specific features of the VFO controller, which guarantee fast and non-oscillatory tracking error convergence in the closed-loop system. Let us briefly recall the equations of the VFO tracker, written for the unicycle model of the last trailer, with short explanation of its particular terms (a more detailed description can be found in [4]).

The VFO controller can be formulated as follows:

$$\Phi_\omega \triangleq k_a e_a + \dot{\theta}_a, \qquad \Phi_v \triangleq h_x \cos\theta_N + h_y \sin\theta_N, \tag{4.24}$$

where Φ_ω is called the *orienting control*, and Φ_v the *pushing control*. The particular terms in the above definitions are determined as follows:

$$h_x = k_p e_x + \dot{x}_{tN}, \qquad h_y = k_p e_y + \dot{y}_{tN}, \qquad e_a = \theta_a - \theta_N, \tag{4.25}$$

$$\theta_a = \mathrm{Atan2c}\left(\sigma \cdot h_y, \sigma \cdot h_x\right), \qquad \dot{\theta}_a = (\dot{h}_y h_x - h_y \dot{h}_x)/(h_x^2 + h_y^2), \tag{4.26}$$

where $k_a, k_p > 0$ are the design coefficients. The decision factor $\sigma \triangleq \mathrm{sgn}(v_{tN}) \in \{-1, +1\}$ (used also in (4.23)) allows the designer to select the desired motion strategy: forward if $v_{tN}(\tau) > 0$ or backward if $v_{tN}(\tau) < 0$, where $v_{tN}(\tau) = \sigma\sqrt{\dot{x}_{tN}^2(\tau) + \dot{y}_{tN}^2(\tau)}$ denotes the reference longitudinal velocity defined along $\overline{\boldsymbol{q}}_t(\tau)$. Note that e_x and e_y are the components of error (4.5). It has been proved in [4] that functions defined by (4.24) guarantee asymptotic convergence of error $\overline{\boldsymbol{e}}(\tau)$ to zero assuming that they are directly forced as inputs to the unicycle-like kinematics.

In the case of an articulated vehicle the functions (4.24) have to be substituted into (4.12) and (4.13) yielding the tracking controller for the N-trailer vehicle.

Remark 4.1. The desired joint angles (4.8) and their time-derivatives are undetermined at the time instants τ_I when the two arguments of function $\mathrm{Atan2c}(\cdot,\cdot)$ are simultaneously equal to zero. We propose to cope with this problem using at τ_I the limit values β_{di}^- and $\dot{\beta}_{di}^-$, where $\beta_{di}^- = \lim_{\tau \to \tau_I^-} \beta_{di}(\tau)$ and $\dot{\beta}_{di}^- = \lim_{\tau \to \tau_I^-} \dot{\beta}_{di}(\tau)$. The explicit formulas of time-derivatives $\dot{\beta}_{di}$ used in Eqs. (4.16), (4.19), and (4.22) may be obtained by formal differentiation of (4.8), which involves the time-derivatives of signals ω_{di} and v_{di}. Since this may cause difficulties in practical implementation, we propose to use instead the so-called *robust exact differentiator* proposed in [9], or approximate terms $\dot{\beta}_{di}$ by their filtered versions $\dot{\beta}_{diF} = \mathscr{L}^{-1}\{s\beta_{di}(s)/(1+sT_F)\}$, which are numerically computable. For tracking the rectilinear or circular reference trajectories terms $\dot{\beta}_{di}$ can be even omitted in control implementation (since now $\dot{\beta}_{ti} \equiv 0$), assuming however sufficiently high values for gains k_i in order to preserve stability of the closed-loop system. The control effectiveness in the latter case will be illustrated by simulation Sim2 in the next section.

4.4 Simulation Results and Discussion

The performance of the closed-loop system with the proposed tracking controller is illustrated by the results of two simulations of backward motion maneuvers: for the *advanced* reference trajectory (Sim1), and for the circular reference trajectory (Sim2). The reference signals $\overline{\boldsymbol{q}}_t(\tau)$ have been generated as a solution of the unicycle model with the reference inputs $\omega_{t3} := 0.15(1 + \sin 0.3\tau)$ rad/s, $v_{t3} := -0.2$ m/s for Sim1, and $\omega_{t3} := 0.15$ rad/s, $v_{t3} := -0.2$ m/s for Sim2, taking the initial configuration for the reference vehicle $\boldsymbol{q}_t(0) = [0\ 0\ 0\ \frac{\pi}{2}\ -1\ 0]^T$. The vehicle initial configurations have been selected as follows: $\boldsymbol{q}(0) = [0\ 0\ 0\ \frac{\pi}{2}\ 0\ 0]^T$ for Sim1, and $\boldsymbol{q}(0) = [0\ 0\ 0\ \pi\ 0\ 0]^T$ for Sim2. The following common numerical values have been selected for both simulations: $L_1 = L_2 = L_3 = 0.25$ m, $k_1 = 50$, $k_2 = 20$, $k_3 = 5$, $k_a = 2$, $k_p = 1$. For Sim1 the feed-forward terms $\dot{\beta}_{di}$, $i = 1,2,3$ have been approximated by $\dot{\beta}_{diF}$ (see Remark 4.1) using the filter time-constant $T_F = 0.05$ s. In the case of Sim2, the simplified control implementation with $\dot{\beta}_{d2,3} := 0$ has been used.

The results of the simulations are presented in Fig. 4.4. For the two simulation tests the relatively demanding initial configurations of the controlled vehicle have been selected in order to show the effectiveness of the proposed control strategy, especially in the context of the vehicle folding effect avoidance.

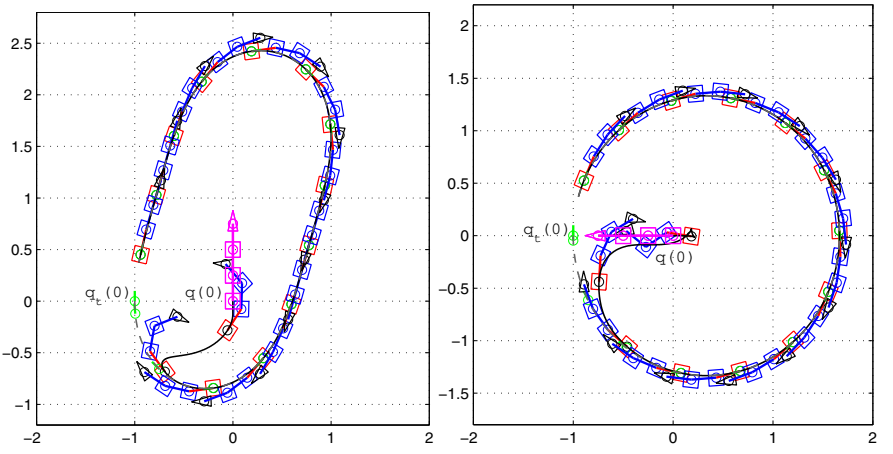


Fig. 4.4 Backward tracking maneuvers in the (x, y) plane (dimensions in [m]): for the advanced reference trajectory (Sim1, left) and for the circular trajectory (Sim2, right). Initial vehicle configuration $\boldsymbol{q}(0)$ and initial reference configuration $\boldsymbol{q}_t(0)$ are denoted in the figure (the former is highlighted in magenta); the last trailer is denoted by the red rectangle, the tractor – by the black triangle; evolution of the reference trailer posture is denoted by the green marks

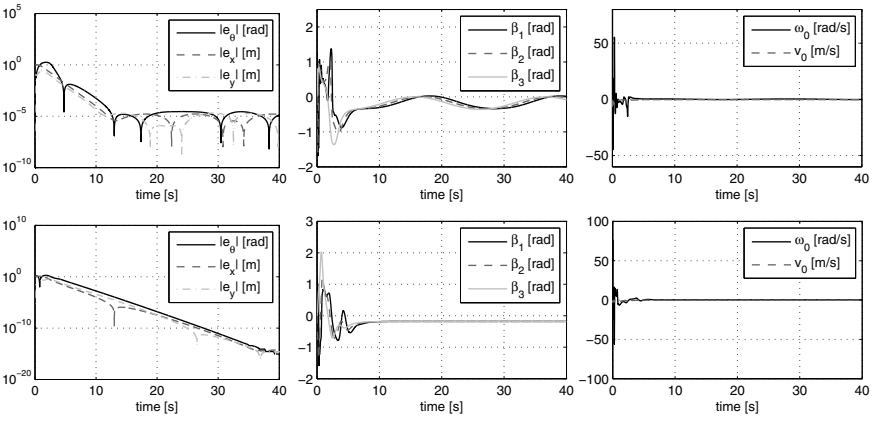


Fig. 4.5 Time plots of the last-trailer posture errors in a logarithmic scale (left), the vehicle joint angles (middle), and the tractor control inputs (right) for simulation Sim1 (top) and Sim2 (bottom)

The reference trajectories selected for simulations differ from each other qualitatively. For the circular trajectory (Sim2) the reference velocities and the reference joint angles remain constant, while for the *advanced* trajectory they are time-varying (this justifies the term *advanced trajectory* for Sim1). Tracking the circular trajectory is inherently easier and it permits the simplified control implementation with $\dot{\beta}_{di} := 0$ simultaneously preserving the asymptotic convergence of errors in (4.6). This is confirmed by the bottom plots in Fig. 4.5, where the last-trailer posture errors converge toward zero and all joint angles β_i converge to the constant steady-state values β_{is} resulting from the formula $|\beta_{is}| = \frac{\pi}{2} - \arctan(r_i/L_i)$, where $r_i^2 = r_t^2 + L_N^2 + \ldots + L_{i+1}^2$ and $r_t = |v_{tN}/\omega_{tN}| > 0$ is the radius of the reference circle.

For simulation Sim1 the feed-forward terms $\dot{\beta}_{di}$ have been approximated by their filtered versions $\dot{\beta}_{diF}$ in order to obtain better tracking precision for time-varying reference signals. However, due to the approximations the error convergence obtained in (4.6) is only the *practical* one with the values of δ_1 and δ_2 depending on the quality of these approximations. This is visible in Fig. 4.5 where the last-trailer posture errors converge to some small envelope near zero. The influence of the quality of the numerical approximations for $\dot{\beta}_{di}$ terms is shown in Fig. 4.6 (left-hand plot), where the evolution of $\|\overline{\boldsymbol{e}}(\tau)\|$ is presented for two scenarios of simulation Sim1: when all the terms $\dot{\beta}_{di}$ are approximated by $\dot{\beta}_{diF}$, and when $\dot{\beta}_{d1}$ is still approximated but $\dot{\beta}_{d2,3}$ are taken as equal to zero (simplified implementation as for Sim2). One can see that in both scenarios stability of the closed-loop system is preserved, but the obtained size of envelope δ_1 is bigger for the simplified control implementation.

In practice the nominal values of L_i used in the recurrence (4.7) (or (4.23)) and (4.8) may be unknown. The robustness of the proposed controller to the parametric uncertainty has been examined by running additional tests for conditions of simulation Sim1 assuming now 5% uncertainty of the vehicle segment lengths. In the first scenario we have used the overestimated parameters by taking in the controller equations the lengths $L_{is} = 1.05 \cdot L_i$. In the second scenario the underestimated parameters $L_{is} = 0.95 \cdot L_i$ have been used. The robustness can be assessed by analyzing

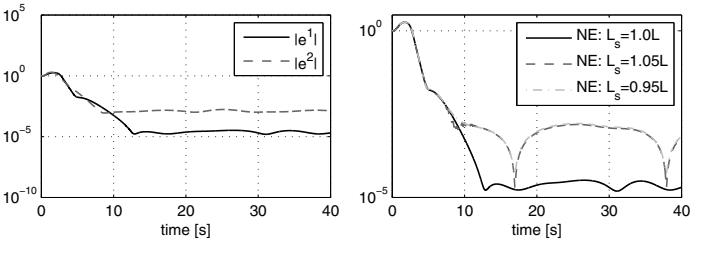


Fig. 4.6 Left: evolution of $\|\overline{\boldsymbol{e}}(\tau)\|$ for two control implementations in Sim1: with terms $\dot{\beta}_{d1,2,3}$ approximated numerically (denoted by $|e^1|$), and with $\dot{\beta}_{d2,3} := 0$ (denoted by $|e^2|$). Right: closed-loop robustness to parametric uncertainty as a plot of $\|\overline{\boldsymbol{e}}(\tau)\|$ for three sets of vehicle segment lengths: $L_{is} = L_i$ (nominal), $L_{is} = 1.05L_i$ (overestimated), and $L_{is} = 0.95L_i$ (underestimated)

the right-hand side plots presented in Fig. 4.6. For the two scenarios stability of the closed-loop system has been preserved. Note that the transient states obtained are virtually indistinguishable in comparison to the nominal case (i.e. when $L_{is} = L_i$).

The control performance obtained in the presence of feedback measurement noise and small offset errors of the trailer hitching point locations can be assessed by analyzing the preliminary results in Fig. 4.7. In this case the 0.01 m hitching offsets have been applied to the robot, and the normally distributed measurement noise with variances $V_{\beta 1,2,3} = 10^{-7}$, $V_{\theta,x,y} = 10^{-6}$ has been added to the feedback signals. During the test the simplified control implementation with $\dot{\beta}_{d2,3} := 0$ was used. Note that sensitivity to the measurement noise increases along the vehicle kinematic chain and is the highest on the tractor side.

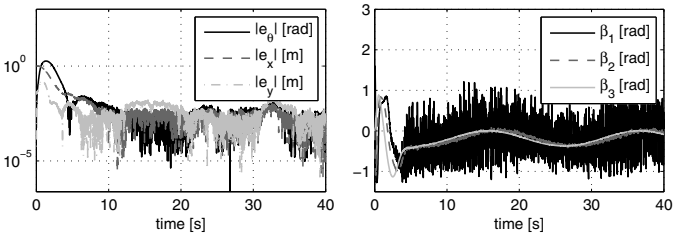


Fig. 4.7 Time plots of the last-trailer posture errors (left) and the joint angles (right) for tracking with small hitching offsets in the vehicle and measurement noise present in feedback

4.5 Final Remarks

In the paper the novel tracking control strategy for the standard N-trailer robot has been presented. The origins of the concept come from geometrical interpretations of the vehicle kinematics formulated in a cascaded form, and also from propagation of velocities along the vehicle kinematic chain. The resultant control system consists of two main components: the serial chain of Single Control Modules with the inner joint-angle feedback loops, and the Last-Trailer Posture Tracker in the outer loop dedicated to the last-trailer segment treated as the unicycle.

A question which naturally arises here concerns the possibility of applying in the LTPT block the feedback controllers other than the VFO one proposed above. Preliminary simulation results obtained by the author (however not presented in this paper) indicate that it may be successful. The necessary and sufficient conditions which the outer-loop controllers should meet to ensure stability and convergence of the closed-loop system have to be investigated yet.

Stability of the proposed closed-loop system remains to be shown. The preliminary analysis conducted so far seems to be promising, since it reveals that the dynamics of the last-trailer posture error (4.5) and the auxiliary joint error $\boldsymbol{e}_d = [e_{d1} \ \ldots \ e_{dN}]^T$ can be written as $\dot{\overline{\boldsymbol{e}}} = \boldsymbol{f}(\overline{\boldsymbol{e}}, \tau) + \boldsymbol{f}_1(\overline{\boldsymbol{e}}, \boldsymbol{e}_d, \tau)$ and $\dot{\boldsymbol{e}}_d = \boldsymbol{A}\boldsymbol{e}_d + \boldsymbol{f}_2(\overline{\boldsymbol{e}}, \boldsymbol{e}_d, \tau)$, where $\boldsymbol{f}_1(\overline{\boldsymbol{e}}, \boldsymbol{e}_d, \tau) = \boldsymbol{G}(\overline{\boldsymbol{q}}_t(\tau) - \overline{\boldsymbol{e}})\boldsymbol{\Gamma}\boldsymbol{e}_{\omega v}(\overline{\boldsymbol{e}}, \boldsymbol{e}_d, \tau)$, $\boldsymbol{G}(\cdot)$ is the

unicycle kinematics matrix, $\boldsymbol{f}_2(\overline{\boldsymbol{e}},\boldsymbol{e}_d,\tau)=\boldsymbol{H}\boldsymbol{e}_{\omega v}(\overline{\boldsymbol{e}},\boldsymbol{e}_d,\tau)$, $\boldsymbol{e}_{\omega v}=[\boldsymbol{e}_\omega^T\ \boldsymbol{e}_v^T]^T$, $\boldsymbol{e}_\omega=[\omega_{d1}-\omega_1\ \ldots\ \omega_{dN}-\omega_N]^T$, $\boldsymbol{e}_v=[v_{d1}-v_1\ \ldots\ v_{dN}-v_N]^T$, $\boldsymbol{A}=\mathrm{diag}\{-k_i\}$ is Hurwitz, and $\boldsymbol{\Gamma}$ and $\boldsymbol{H}$ are the appropriate constant matrices with -1, 0, and $+1$ entries. Furthermore, one can show that $\boldsymbol{f}_2(\boldsymbol{e}_d=\boldsymbol{0},\cdot)=\boldsymbol{0}$, $\boldsymbol{f}_1(\boldsymbol{e}_d=\boldsymbol{0},\cdot)=\boldsymbol{0}$, and the nominal (unperturbed) dynamics $\dot{\overline{\boldsymbol{e}}}=\boldsymbol{f}(\overline{\boldsymbol{e}},\tau)$ is asymptotically stable (see the proof in [4]). We plan to proceed our further analysis using the stability theorems of interconnected systems, [6], showing first the required features of functions $\boldsymbol{f}_1(\cdot)$ and $\boldsymbol{f}_2(\cdot)$. If the convergence for $\overline{\boldsymbol{e}}$ and $\boldsymbol{e}_d$ is shown, the convergence of the joint angle error (4.4) might result from the flatness property of the vehicle and feasibility of the reference trajectory $\boldsymbol{q}_t(\tau)$. Specifically, since $\Phi_v(\overline{\boldsymbol{e}}=\boldsymbol{0},\cdot)=v_{tN}(\tau)$ and $\Phi_\omega(\overline{\boldsymbol{e}}=\boldsymbol{0},\cdot)=\omega_{tN}(\tau)$, then according to (4.14)–(4.21) it could be shown that $(\beta_{di}-\beta_{ti})\to 0$ as $\overline{\boldsymbol{e}}\to\boldsymbol{0}$, which would complete the proof recalling the asymptotic convergence of $\boldsymbol{e}_d$.

Acknowledgements. This work was supported by the Polish scientific fund in years 2010-2012 as the research project No. N N514 087038.

References

1. Aneke, N.P.I., Nijmeijer, H., de Jager, A.G.: Tracking control of second-order chained from systems by cascaded backstepping. Int. J. Robust Nonlinear Control 13, 95–115 (2003)
2. Bullo, F., Murray, R.M.: Experimental comparison of trajectory trackers for a car with trailers. In: IFAC World Congress, pp. 407–412 (1996)
3. De Luca, A., Oriolo, G.: Local incremental planning for nonholonomic mobile robots. In: IEEE Int. Conf. on Robotics and Automation, pp. 104–110 (1994)
4. Michałek, M., Kozłowski, K.: Vector-Field-Orientation feedback control method for a differentially driven vehicle. IEEE Trans. on Control Syst. Technology 18(1), 45–65 (2010)
5. Jean, F.: The car with N trailers: characterisation of the singular configurations. Control, Opt. Calc. Variations 1, 241–266 (1996)
6. Khalil, H.K.: Nonlinear systems, 3rd edn. Prentice-Hall, New Jersey (2002)
7. Kim, D., Oh, J.: Globally asymptotically stable tracking control for a trailer system. Journal of Robotic Systems 19(5), 199–205 (2002)
8. Laumond, J.P.: Controllability of a multibody mobile robot. IEEE Transactions on Robotics and Automation 9(6), 755–763 (1993)
9. Levant, A.: Higher-order sliding modes, differentiation and output-feedback control. Int. J. Control 76(9/10), 924–941 (2003)
10. Morin, P., Samson, C.: Transverse function control of a class of non-invariant driftless systems. Application to vehicles with trailers. In: Proceedings of the 47th IEEE Conference on Decision and Control, Cancun, Mexico, pp. 4312–4319 (2008)
11. Pradalier, C., Usher, K.: Robust trajectory tracking for a reversing tractor trailer. Journal of Field Robotics 25(6-7), 378–399 (2008)

12. Sørdalen, O.J.: Conversion of the kinematics of a car with *n* trailers into a chained form. In: Proc. IEEE Int. Conf. on Robotics and Automation, Atlanta, USA, pp. 382–387 (1993)
13. Rouchon, P., Fliess, M., Levine, J., Martin, P.: Flatness, motion planning and trailer systems. In: Proceedings of the 32nd Conference on Decision and Control, San Antonio, Texas, pp. 2700–2705 (1993)
14. Sørdalen, O.J., Egeland, O.: Exponential stabilization of nonholonomic chained systems. IEEE Transactions on Automatic Control 40(1), 35–49 (1995)
15. Zöbel, D.: Trajectory segmentation for the autonomous control of backward motion for truck and trailer. IEEE Trans. Intell. Transportation Systems 4(2), 59–66 (2003)

Chapter 5
Robust Control of Differentially Driven Mobile Platforms

Alicja Mazur and Mateusz Cholewiński

5.1 Introduction

Wheeled mobile platforms constitute an important group of robotic objects. They can be treated as independent robots or as a transportation part of a composite robotic assembly, for instance mobile manipulators. Depending on the kind of the wheels and the way in which they are fixed to the cart, motion of wheeled mobile platforms can be realized with or without slippage phenomenon. If no slippage effect between the wheels and the surface occurs, then there exists some equation describing forbidden directions of realized velocities of the system. Such a relationship is called the nonholonomic constraint in the platform's motion.

A special kind of wheeled mobile platforms are platforms with more than one axis equipped with fixed wheels. Such vehicles are called differentially driven platforms [1] (due to differential thrust on the wheels at the opposite sides) or skid-steering platforms [4] (due to a skidding effect observed in their behavior). We will use the first name (DDP) because this paper is a kind of polemics with the approach introduced by Caracciolo et al. in [1].

Research activity in the field of modeling and developing control algorithms for DDP has been continued since the 90's. The first attempt to solve this problem was the work [1] of Caracciolo at el., in which some assumption about the instantaneous center of rotation was made. The authors took the stand that there existed some non-integrable relationship between angular velocity of orientation's changes and lateral slipping of the wheels: they assumed that the projection of the x coordinate of the instantaneous center of rotation on the local frame associated with the platform mass center was constant. Such an equation, although derived from the slipping effect, could play a role of a specific nonholonomic constraint. A similar approach to that presented by Caracciolo et al. can be found e.g. in [4].

Alicja Mazur · Mateusz Cholewiński
Institute of Computer Engineering, Control, and Robotics,
Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: {alicja.mazur,mateusz.cholewinski}@pwr.wroc.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 53–64.
springerlink.com

Another approach to the problem of modeling and control of DDP has appeared recently. Many authors, see e.g. [6], [7], have observed that the assumption introduced in [1] is not always fulfilled in practice. They have treated DDP as an underactuated system on dynamic level with non-stationary kinematics (non-stationary velocity constraint).

This point of view can be justified but there are other possible solutions to the problem of modeling and control of DDP. The authors of the paper have been researching virtual force acting on DDP if only the assumption about lack of longitudinal slippage is taken into account. In our opinion it is very hard to judge which method of mathematical description and control is the best one.

In the paper a new control algorithm for solving the trajectory tracking problem for differentially driven mobile platforms is presented. The paper is a polemics with the approach introduced in [1]. First, we want to present a mathematical model of a differentially driven platform with an artificial nonholonomic constraint. Due to the cascaded structure of such a model, the control law is divided into two stages: a kinematic controller (a motion planner on the kinematic level) and a dynamic controller. In opposition to the approach introduced in [1], where some uncertainties in dynamics have very disturbing influence on the behavior of the platform, we want to show simple sliding mode control designed in the second stage, which is robust to such disturbances and preserves asymptotic convergence to the desired trajectory in the presence of unknown platform and terrain parameters. Finally, we want to discuss the robustness of our method of control and compare it with the results presented in [1].

The paper is organized as follows. Section 5.2 illustrates the theoretical design of the mathematical model of the considered objects. In Section 5.3 the control problem is formulated. In Section 5.4 a new control algorithm is designed and its proper action is proved. Moreover, comparison with the approach presented by Caracciolo et al. is done. Section 5.5 contains the simulation results. Section 5.6 presents some conclusions.

5.2 Mathematical Model of Differentially Driven Mobile Platform

Differentially driven wheeled mobile platforms can be considered robotic objects with special nonholonomic constraints. Such an object is usually described by two groups of expressions: equations of constraints (kinematics) and equations of forces or torques acting on the object (dynamics). We want to evoke mathematical model which can be found in many papers, see e.g. [1], [4].

5.2.1 Constraints of Motion

A differentially driven mobile platform is a cart equipped with two or more axes with fixed wheels. In our considerations we will restrict ourselves to the platform with only two axes. Due to the approach introduced in [1], [4], we take the following assumptions:

- the platform moves on a horizontal plane (i.e. gravitational forces neglected),
- the platform wheels have only point contact between the surface and the tires,
- the velocity of the platform is relatively small,
- longitudinal slippage is negligible.

The posture of any mobile platform can be described by the following generalized coordinates

$$q = \begin{pmatrix} x & y & \theta \end{pmatrix}^T, \tag{5.1}$$

where (x, y) are the Cartesian coordinates of the mass center relative to global inertial frame $X_0Y_0Z_0$ and θ is the orientation of the platform (the angle between the X-axes of the local and global frames). The local frame is so mounted in the mass center that the platform moves forward along the local X_p axis, see Fig. 5.1.

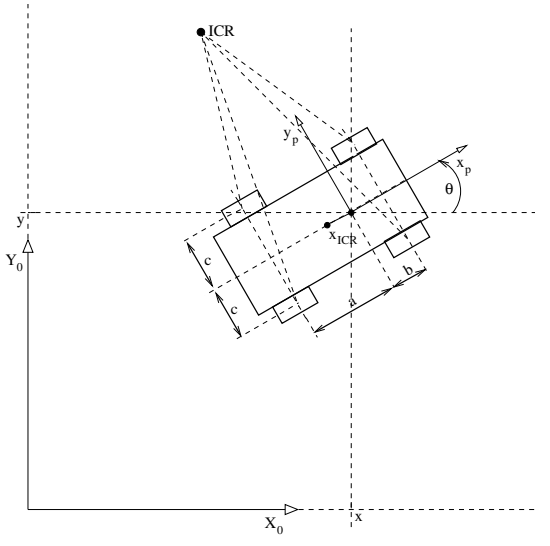


Fig. 5.1 Kinematic structure of differentially driven platform with two axes

Let us define the relationship between velocities v expressed relative to the local frame and global velocities $\dot{q}$ as follows:

$$\dot{q} = Rot(Z,\theta)v = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} v_x \\ v_y \\ \omega \end{pmatrix}, \tag{5.2}$$

where v_x and v_y are the longitudinal and lateral velocities, respectively, and ω denotes the angular velocity of the platform.

If the vehicle moves straightforward then no slippage is present but if the orientation changes ($\omega \neq 0$) then lateral slippage occurs. The equation joining velocities v_y and ω can be expressed as in [1]:

$$v_y + x_{ICR}\omega = 0. \tag{5.3}$$

ICR (instantaneous center of rotation) is a point, due to the Descartes principle, in which a temporary axis of rotation can be located (for vehicle motion without longitudinal slippage). Variable x_{ICR} is a coordinate of the ICR along local axis X_p associated with the platform. It can be computed from (5.2) that

$$v = R^T(Z,\theta)\dot{q} \qquad \longrightarrow \qquad v_y = -\dot{x}\sin\theta + \dot{y}\cos\theta$$

and, after combining with (5.3), where $v_y = -x_{ICR}\omega$, the constraint can be expressed in a Pfaffian form as follows:

$$\begin{bmatrix} -\sin\theta & \cos\theta & x_{ICR} \end{bmatrix} \dot{q} = A(q)\dot{q} = 0. \tag{5.4}$$

Due to (5.4), since the platform velocities $\dot{q}$ are always in the null space of A, it is always possible to find a vector of auxiliary velocities $\eta \in R^2$, such that

$$\dot{q} = G(q)\eta, \tag{5.5}$$

where $G(q)$ is a 3×2 full rank matrix satisfying the relationship $A(q)G(q) = 0$.

A possible form of $G(q)$ can be defined as below:

$$\dot{q} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \\ 0 & -\frac{1}{x_{ICR}} \end{bmatrix} \begin{pmatrix} v_x \\ v_y \end{pmatrix} = G(q)\eta. \tag{5.6}$$

Equation (5.6) plays the role of an artificial nonholonomic constraint for the differentially driven wheeled mobile platform.

5.2.2 *Dynamics of Differentially Driven Mobile Platform*

Because of the nonholonomy of constraint (5.4), to obtain the dynamic model describing the behavior of the mobile platform the d'Alembert principle needs to be used:

$$Q(q)\ddot{q} + C(q,\dot{q})\dot{q} + R(\dot{q}) = A^T(q)\lambda + B(q)u, \tag{5.7}$$

where $Q(q)$ is the inertia matrix of the mobile platform, $C(q,\dot{q})$ is the matrix coming from Coriolis and centrifugal forces, $R(\dot{q})$ is the vector of reactive forces coming from the terrain, $A(q)$ is the matrix of nonholonomic constraints, λ is the Lagrange

multiplier, $B(q)$ is the input matrix, and u is the vector of controls (input signals from the actuators).

The inertia matrix has the form

$$Q(q) = \begin{bmatrix} M & 0 & 0 \\ 0 & M & 0 \\ 0 & 0 & I_z \end{bmatrix}, \tag{5.8}$$

where M denotes the total mass of the platform and I_z is its moment of inertia relative to the Z_p axis. The matrix of Coriolis forces $C = 0$ because $Q(q)$ is constant.

The vector of reactive forces and torques is equal to [4]

$$R(\dot{q}) = \begin{pmatrix} F_s \cos\theta - F_l \sin\theta \\ F_s \sin\theta + F_l \cos\theta \\ M_r \end{pmatrix}, \tag{5.9}$$

with its elements defined below:

$$\begin{aligned} F_s &= \sum_{i=1}^{4} F_{si} = \mu_{si} N_i \text{sign}(v_x), \qquad F_l = \sum_{i=1}^{4} F_{li} = \mu_{li} N_i \text{sign}(v_y) \\ M_r &= b(F_{l2} + F_{l3}) - a(F_{l1} + F_{l4}) + c(F_{s3} + F_{s4} - F_{s1} - F_{s2}). \end{aligned} \tag{5.10}$$

Symbols μ_{si} and μ_{li} denote the dry friction coefficients for the ith wheel in the longitudinal and lateral direction; N_i is the force acting on the ith wheel in the vertical direction, depending on the weight of the platform; a and b are the distances from the platform mass center to both axes of wheels; c is the half of the platform width, see Fig. 1.

Input matrix $B(q)$ can be explicitly calculated as follows:

$$B(q) = \frac{1}{r} \begin{bmatrix} \cos\theta & \cos\theta \\ \sin\theta & \sin\theta \\ -c & c \end{bmatrix}, \tag{5.11}$$

where r is the radius of each wheel. In turn, control signals $u = \left(u_l, u_r\right)^T$ represent the torques generated by the actuators on the left and right side of the platform.

Now we want to express the model of dynamics using auxiliary velocities (5.5) instead of the generalized coordinates of the mobile platform. We compute

$$\ddot{q} = G(q)\dot{\eta} + \dot{G}(q)\eta$$

and eliminate in the model of dynamics the Lagrange multiplier using the condition $G^T A^T = 0$. Substituting $\dot{q}$ and $\ddot{q}$ in (5.7) we get

$$Q^* \dot{\eta} + C^* \eta + R^* = B^* u, \tag{5.12}$$

with the elements of the matrix equation defined in the following way:

$$Q^* = G^T Q G, \quad C^* = G^T Q \dot{G}, \quad R^* = G^T R, \quad B^* = G^T B.$$

5.3 Control Problem Statement

In the paper, our goal is to find a control law guaranteeing trajectory tracking for the differentially driven mobile platform. Our goal is to address the following control problem for such platforms:

> Determine control law u such that a differentially driven mobile platform with fully known dynamics or with some parameter uncertainty in the dynamics follows the desired trajectory.

To design a trajectory tracking controller for the considered mobile platform, it is necessary to observe that a complete mathematical model of the nonholonomic system expressed in auxiliary variables is a cascade consisting of two groups of equations: kinematics and dynamics, see Fig. 5.2. For this reason the structure of the controller is divided into two parts working simultaneously:

- kinematic controller η_r – represents a vector of embedded control inputs, which ensure realization of the task for the kinematics (nonholonomic constraints) if the dynamics are not present. Such a controller generates a 'velocity profile', which can be executed in practice to realize the trajectory tracking for the nonholonomic differentially driven wheeled mobile platform.
- dynamic controller – as a consequence of the cascaded structure of the system model, the system velocities cannot be commanded directly, as assumed in the design of kinematic control signals, and instead they need to be realized as the output of the dynamics driven by u.

It can be observed that we have evoked the backstepping-like algorithm [5] to solve the presented control problem for DDP. Backstepping is a well-known and often used approach to control of cascaded systems, e.g. systems with nonholonomic constraints. Because we have assumed that the artificial constraint introduced in [1] is active, this method of control is justified. In our approach the kinematic level of control should be realized without any disturbances and some uncertainty can occur only at the dynamic level of control.

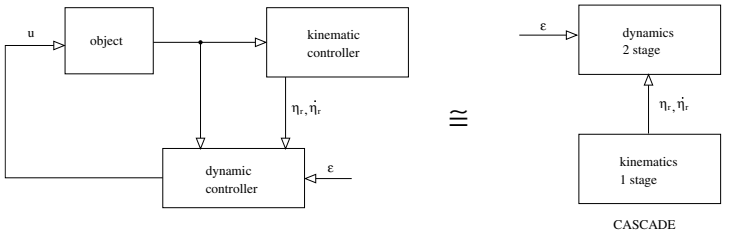


Fig. 5.2 Structure of the proposed control algorithm: cascade with two stages

5.4 Design of Control Algorithm

As mentioned in the previous section, the control algorithm consists of two parts, i.e. the kinematic controller and the dynamic controller. We want to describe both control algorithms necessary to solve the control problem of nonholonomic differentially driven platforms.

5.4.1 Kinematic Control Algorithm for Trajectory Tracking

Due to [1] we will apply the dynamic extension algorithm [2] as the kinematic controller but only for kinematic equation (5.6). First, we choose special outputs, which can be fully linearized and input-output decoupled by dynamic feedback. Next, we extend the state space and introduce some integrators to inputs to give delay to the selected control signals.

We choose as linearizing outputs the position coordinates of some point lying on the x axis at a distance $d_0 = x_{ICR}$ from the origin of the local frame, namely

$$h = \begin{pmatrix} h_1 \\ h_2 \end{pmatrix} = \begin{pmatrix} x + d_0 \cos\theta \\ y + d_0 \sin\theta \end{pmatrix}. \tag{5.13}$$

The task of the platform is to track some desired trajectory $h_{1d}(t), h_{2d}(t)$ defined for the previously selected point.

The extended state space has been chosen as $q_e = (x, y, \theta, \eta_1)^T$ with $\eta_1 = v_x$ and $\eta_2 = v_y$. As new input signals we take $\mu = (\eta_2, w_1)^T$ where $w_1 = \dot{\eta}_1$ is the input of an additional integrator placed before η_1.

We apply the classical method of input-output decoupling and linearization [3] and differentiate Eq. (5.13) until the input signals explicitly appear:

$$\dot{h} = \begin{pmatrix} \dot{h}_1 \\ \dot{h}_2 \end{pmatrix} = \begin{pmatrix} \eta_1 \cos\theta \\ \eta_1 \sin\theta \end{pmatrix}, \tag{5.14}$$

$$\ddot{h} = \begin{pmatrix} \ddot{h}_1 \\ \ddot{h}_2 \end{pmatrix} = \begin{bmatrix} \frac{1}{d_0}\eta_1 \sin\theta & \cos\theta \\ -\frac{1}{d_0}\eta_1 \cos\theta & \sin\theta \end{bmatrix} \begin{pmatrix} \eta_2 \\ w_1 \end{pmatrix} = K_d \mu\,. \tag{5.15}$$

Since the determinant of decoupling matrix K_d is equal to $\det K_d = \eta_1/d_0$, we deduce that the decoupling matrix is nonsingular if the platform longitudinal velocity $\eta_1 = v_x$ is different from 0. Using the following control law:

$$\mu = K_d^{-1}\left(\ddot{h}_d - K_2(\dot{h} - \dot{h}_d) - K_1(h - h_d)\right), \qquad K_1, K_2 > 0, \tag{5.16}$$

we get a linear differential equation of the second order, which is exponentially stable if regulation parameters K_1, K_2 are positive definite matrices.

5.4.2 Dynamic Control Algorithm

At the beginning, we propose a dynamic control algorithm providing asymptotic convergence of all state variables of the mobile platform for full knowledge about the dynamic model of the platform. Next, we extend the previously obtained control law onto parametric uncertainty in dynamics and we get a robust control law for uncertain forces acting on the differentially driven wheeled mobile platform.

5.4.2.1 Full Knowledge about Platform Dynamics

We consider the model of a mobile platform DDP (5.12) expressed in auxiliary velocities. We assume that we know velocities η_r computed due to kinematic control algorithm (5.16), which solve the trajectory tracking problem for the differentially driven mobile platform. Then we propose a control law for fully known dynamics in the form

$$u = (B^*)^{-1}\left\{Q^*\dot{\eta}_r + C^*\eta_r + R^* - K_d e_\eta\right\}, \tag{5.17}$$

where $e_\eta = \eta - \eta_r$, $K_d = K_d^T > 0$. The closed-loop system (5.12)-(5.17) is described by the error equation

$$Q^*\dot{e}_\eta = -K_d e_\eta - C^* e_\eta. \tag{5.18}$$

In order to prove the convergence of the trajectories of the DDP platform to the desired trajectory, we choose the following Lyapunov-like function:

$$V(q, e_\eta) = \frac{1}{2} e_\eta^T Q^*(q) e_\eta. \tag{5.19}$$

Now we calculate the time derivative of V

$$\dot{V} = \frac{1}{2} e_\eta^T \dot{Q}^*(q) e_\eta + e_\eta^T Q^*(q)\dot{e}_\eta$$

and evaluate the time derivative of V along the trajectories of the closed-loop system (5.18). Due to skew-symmetry between the inertia matrix and the matrix of Coriolis forces we calculate as follows:

$$\dot{V} = \frac{1}{2} e_\eta^T \dot{Q}^*(q) e_\eta + e_\eta^T(-K_d e_\eta - C^* e_\eta) = -e_\eta^T K_d e_\eta \leq 0. \tag{5.20}$$

Now from the Yoshizawa-LaSalle theorem [5] we can deduce that every solution of Eq. (5.20) is bounded and converges to the set $N = \{(e_\eta) \mid \dot{V}(e_\eta) = 0$, i.e. velocity error $e_\eta = 0$ is an asymptotically stable equilibrium point. On the other hand, $e_\eta \to 0$ means that the velocity profile is well recorded and the nonholonomic constraints are fulfilled if kinematic control realizes the desired task (i.e. trajectory tracking) without any disturbances. This ends the proof.

5.4.2.2 Parametric Uncertainty in Dynamics

A more interesting situation occurs if some parameters, e.g. interaction forces between the wheels and the terrain, are uncertain. We assume that the form of all the forces defined by (5.12) is known but the coefficients of these forces and torques are not. This means that dynamical model (5.12) can be linearly parameterized in the following way:

$$Q^*\dot{\eta} + C^*\eta + R^* = Y(\dot{\eta}, \eta, q, \dot{q})a = B^*u, \tag{5.21}$$

where Y is a known regression matrix and a is a vector of unknown but constant parameters in dynamics. In such situation we try to use some robust control law using a sliding mode approach, namely

$$\begin{aligned} u &= (B^*)^{-1}\left\{\hat{Q}^*\dot{\eta}_r + \hat{C}^*\eta_r + \hat{R}^* - K_d e_\eta - K\,\mathrm{sign}\, e_\eta\right\} \\ &= (B^*)^{-1}\left\{\, Y(\dot{\eta}_r, \eta_r, q, \dot{q})\hat{a} - K_d e_\eta - K\,\mathrm{sign}\, e_\eta\right\}, \end{aligned} \tag{5.22}$$

where K is some additional positive regulation parameter. In control law (5.22), as opposed to (5.17), all forces and torques acting on the DDP are not known exactly but an approximation of these expressions with the vector of estimated parameters $\hat{a} \in R^p$ is used. These estimates belong to the interval $\hat{a} \in [\, 0, a_{max}\,]$, where $a_{max} \in R^p$ is a known vector of overbounding values for the dynamical parameters. This implies that some approximation error occurs: $\tilde{a} = \hat{a} - a, \quad |\,\tilde{a}\,| \le A < \infty$, which is constant and bounded.

In turn, the closed-loop system (5.12), (5.22) has the modified form

$$Q^*\dot{e}_\eta = Y(\dot{\eta}_r, \eta_r, q, \dot{q})\tilde{a} - K_d e_\eta - C^* e_\eta - K\,\mathrm{sign}\, e_\eta. \tag{5.23}$$

To prove convergence of the algorithm, we choose the same Lyapunov-like function:

$$V(q, e_\eta) = \frac{1}{2} e_\eta^T Q^*(q) e_\eta. \tag{5.24}$$

Now we calculate the time derivative of V and evaluate it along the trajectories of the closed-loop system (5.23). We obtain

$$\begin{aligned} \dot{V} &= \frac{1}{2} e_\eta^T \dot{Q}^*(q) e_\eta + e_\eta^T \left(Y(\dot{\eta}_r, \eta_r, q, \dot{q})\tilde{a} - K_d e_\eta - C^* e_\eta - K\,\mathrm{sign}\, e_\eta\right) \\ &= -e_\eta^T K_d e_\eta - e_\eta^T (K\,\mathrm{sign}\, e_\eta - Y(\dot{\eta}_r, \eta_r, q, \dot{q})\tilde{a}) \\ &= -e_\eta^T K_d e_\eta - \sum_i e_{i\eta}\ \mathrm{sign}\, e_{i\eta} (K - \mathrm{sign}\, e_{i\eta} \sum_{j=1}^{p} Y_{ij}(\dot{\eta}_r, \eta_r, q, \dot{q})\tilde{a}_j) \end{aligned} \tag{5.25}$$

If regulation parameter K is larger than a properly chosen positive number, e.g. $K \ge H + \varepsilon, \;\; \varepsilon > 0$, where $\forall\, i \;|\, \sum_{j=1}^{p} Y_{ij}(\dot{\eta}_r, \eta_r, q, \dot{q})\tilde{a}_j \,| \le H < \infty$, then the following inequality holds:

$$\dot{V} \le -e_\eta^T K_d e_\eta - |\, e_\eta \,| \sqrt{2}\varepsilon \le 0. \tag{5.26}$$

We evoke the Yoshizawa-LaSalle theorem once again and conclude that error e_η converges to 0, and consequently the trajectory tracking is realized in practice. This ends the proof.

5.4.3 *Comments about Robustness of Control Scheme*

As a point of departure we have taken solution to the control problem presented in [1]. Caracciolo et al. have used the dynamic extension algorithm to solve the trajectory tracking problem for DDP. They have made an error because especially for this algorithm, they have treated dynamics as the first step in the cascade, see Fig. 5.3 (the description of the object). This means that any disturbances or uncertainties in dynamics occur at the beginning of the control chain and next they need to be integrated a few times. It could be a source of potential instability of the control loop.

If we compare the control schemes presented in Figs. 5.2 and 5.3 we can observe that possible disturbances in dynamics affect the control law in different ways. In [1] the disturbances have been integrated a few times, see Fig. 5.3, which means that very small steady-state errors go to infinity in time. Only for spline-type trajectories of exactly third order Caracciolo et al. can show that bounded errors occurring on the dynamical level stay bounded.

Such properties of the control loop come from improper use of the dynamic extension algorithm. If this algorithm is used only as kinematic control (with exactly known equations) then any properly chosen robust control law working on the dynamic level, e.g. sliding mode control presented in this paper, can preserve asymptotic convergence of the trajectory tracking error to 0, independent of the type of the tracked trajectory. Only the assumption about the boundness of the desired trajectory is needed.

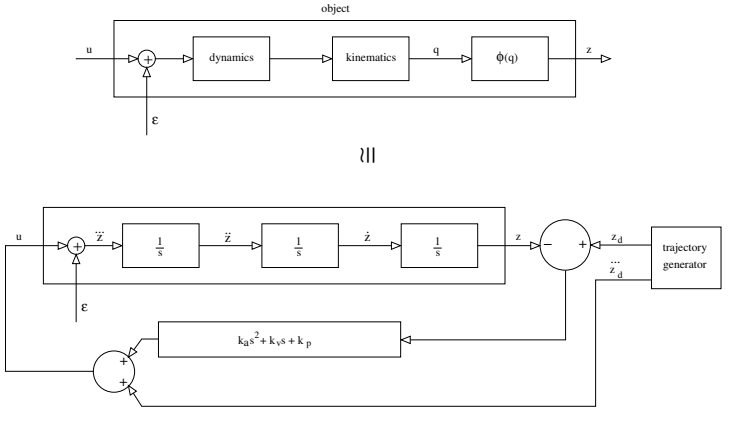


Fig. 5.3 Structure of the control algorithm introduced by Caracciolo et al.

5.5 Simulation Study

The simulations were run with the MATLAB package and the SIMULINK toolbox. As the object of simulations we took a mobile platform equipped with two axes of fixed wheels. The parameters of the platform were equal to: $M = 96$ [kg], $I = 20$ [kg·m^2], $a = 0.37$ [m], $b = 0.55$ [m], $r = 0.2$ [m]. The parameters of terrain were set on value $F_{si} = F_{li} = \mu_i N_i \text{sign}\, v = 1 \cdot \text{sign}\, v$, but for the control law we took the estimated value $F_{si} = F_{li} = \mu_i N_i \text{sign}\, v = 0.5 \cdot \text{sign}\, v$. The same procedure was adopted for the other parameters in dynamics: instead of real parameters we used halves of their values.

The goal of the simulations was to investigate the behavior of the mobile platform with the controller (5.22) proposed in the paper. The desired trajectory of the platform was selected as a circle with radius $R = 10$ [m] and frequency $\omega = 1$ [$\frac{\text{rad}}{s}$]. The parameters of the kinematic controller given by (5.16) were equal to $K_1 = 1$, $K_2 = 10$. In turn, the parameters of the dynamic controller were selected as $K_d = 100$, $K = 10$. Tracking the desired trajectory for the mobile platform is presented in Fig. 5.4. The tracking errors of the Cartesian coordinates for the mobile platform is presented in Fig. 5.5.

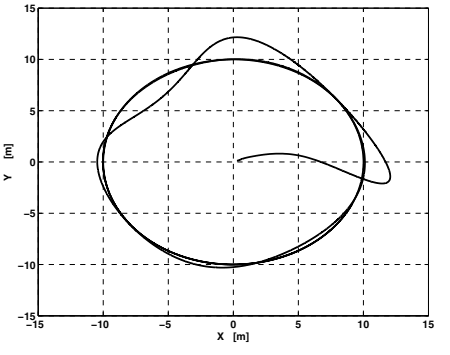
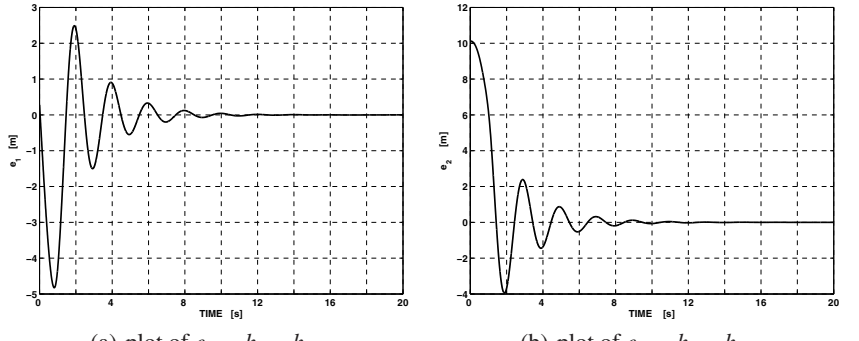


Fig. 5.4 Trajectory of differentially driven platform during tracking a circle



(a) plot of $e_1 = h_1 - h_{1d}$ (b) plot of $e_2 = h_2 - h_{2d}$

Fig. 5.5 Position errors of differentially driven platform during tracking a circle

5.6 Conclusions

This paper presents a solution to the trajectory tracking problem for differentially driven mobile platforms. We assume that the artificial nonholonomic constraint (5.3) is fulfilled in practice. Due to the cascaded structure of the mathematical model of the platform, the control algorithm consists of two levels: kinematic and dynamic. As a kinematic controller we have applied the dynamic extension algorithm, but only to the equations of constraints. A dynamic controller has been introduced as a sliding mode control. Such a solution is robust to parametric uncertainty in dynamics and can work without any adaptation of unknown parameters.

As opposed to the solution introduced in [1], parametric uncertainty in dynamics occurs only in the second part of the cascade (the end of the control chain). This means that disturbances affecting the control system can be compensated and robustly decreased. In [1] uncertainty in dynamics influences the first control level and is integrated three times – this implies that approach to the presented control problem is not robust and should not be applied in practice.

From plots depicted in Fig. 5.4-5.5 we can see that the controller (5.21) introduced in this paper works properly. The dynamic controller (5.22) can be applied for mobile platforms with fully known dynamics and parametric uncertainty in dynamics.

References

1. Caracciolo, L., De Luca, A., Iannitti, S.: Trajectory tracking control of a four-wheel differentially driven mobile robot. In: Proc. IEEE Int. Conf. Rob. Aut., pp. 2632–2638. Michigan, Detroit (1999)
2. Fliess, M., Levine, J., Martin, P., Rouchon, P.: Flatness and defect of nonlinear systems: introductory theory and applications. Int. J. Control 61, 1327–1361 (1995)
3. Isidori, A.: Nonlinear Control Systems, 3rd edn. Springer, London (1995)
4. Kozłowski, K., Pazderski, D.: Practical stabilization of a skid-steering mobile robot – a kinematic-based approach. In: Proc. 3rd Int. Conf. Mechatr., Madras, pp. 520–524 (2006)
5. Krstić, M., Kanellakopoulos, I., Kokotović, P.: Nonlinear and Adaptive Control Design. J. Wiley and Sons, New York (1995)
6. Lewis, A.D.: When is mechanical control system kinematic? In: Proc. 38th Conf. Dec. Contr., Phoenix, Arizona, pp. 1162-1167 (1999)
7. Pazderski, D., Kozłowski, K.: Trajectory tracking of underactuated skid-steering robot. In: Proc. Amer. Contr. Conf., Seattle, WA, pp. 3506–3511 (2008)

Chapter 6
Mobile System for Non Destructive Testing of Weld Joints via Time of Flight Diffraction (TOFD) Technique

Barbara Siemiątkowska, Rafał Chojecki, Mateusz Wiśniowski, Michał Walęcki, Marcin Wielgat, and Jakub Michalski

Abstract. This paper describes research towards the development of a robotic system for automated welded joint testing. The tests are often carried out manually by skilled personnel. Automating the inspection process would reduce errors and associated costs. The system proposed in this paper is based on a mobile robot platform and is designed to carry ultrasonic sensors in order to scan welds for defects. The robot is equipped with a vision system in order to detect the weld position. A fuzzy control system is used in order to control robot motion along the weld.

6.1 Introduction

The TOFD technique (Time of Flight Diffraction) is an ultrasonic technique, especially dedicated for welded joint testing. It offers the possibility of detection, location, and sizing of flaws in welds or parent material. The TOFD technique requires two highly damped, longitudinal wave ultrasonic probes. One is a transmitter and the other is a receiver. The general setup for TOFD is shown in Fig. 6.1.

Since TOFD is a very sensitive technique [1, 2, 7, 11], many factors have an influence on the results. Apart from ultrasonic factors like frequency, resolution,

Barbara Siemiątkowska · Rafał Chojecki · Mateusz Wiśniowski
Institute of Automatic Control and Robotics, Warsaw University of Technology,
ul. Św. A. Boboli 8, 02-525 Warsaw, Poland
e-mail: bsiem@ippt.gov.pl, {r.chojecki,wisnio}@mchtr.pw.edu.pl

Michał Walęcki
Institute of Control and Computation Engineering, Warsaw University of Technology,
ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: mw@mwalecki.pl

Marcin Wielgat · Jakub Michalski
Materials Engineers Group Ltd., ul. Wołoska 141, 02-507 Warsaw, Poland
e-mail: {m.wielgat,j.michalski}@megroup.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 65–74.
springerlink.com

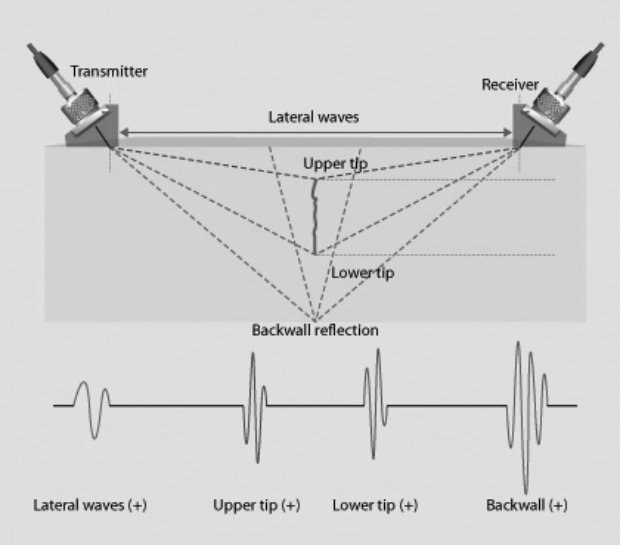


Fig. 6.1 Principle of TOFD technique and the phase sign of four major signals [11]

beam spread and the presence of material noise, also human factors such as mutual position of the probes, simultaneous carrying probes, coupling them and even probes pressing to the material surface could have a great effect on the data obtained. The lack of coupling may cause data loss. Different pressing load of the probes on the standard sample and on the tested material may cause inaccurate flaw measurements. Providing manual motion of the probes also generates many errors. Symmetrical and central positioning of the probes on both sides of the weld is crucial. When the probes are offset from the center of the weld, not all the volume of the weld will be tested and some possible defects could be omitted. During manual scanning all these factors have to be taken under consideration and precisely controlled by an operator, which makes the TOFD technique difficult and not operator-friendly, especially in difficult industrial conditions. These problems could be partially or completely avoided by using specially designed automated scanning tools (robots), where all parameters that normally have to be controlled by an operator are automatically controlled and provided by a robot. Furthermore, using mobile robots equipped with scanning tools allows for control or investigation of welds that are located in places with difficult or dangerous access.

The main objective of the project was to design a modular mobile robot adapted to move on steel vertical surfaces of industrial installations where welded connections occur. The robot is expected to conduct a remote controlled inspection of welded connections with the help of a human operator standing 10 m away from the vehicle or to act autonomously following the weld using a vision system. Visual inspections can be utilized or non destructive testing of welded connections can be used.

(a) (b)

Fig. 6.2 The robot: (a) with additional sensor modules, b) following the weld

6.2 The Magenta Robot – Hardware

The robot (Fig. 6.2) is based on a drive platform equipped with four magnetic wheels. The front wheels are the driving ones while the rear wheels, installed on a self-aligning beam, support the whole construction. The mobile robot consists of the main body, a rear support element and exchangeable sensor modules.

The main body is made of duraluminium. Inside there are two DC motors with encoders and integrated planetary gearboxes as well as the main microprocessor control system, a dual motor controller and two DC/DC converters. The motor encoders measure speed and distance. Turning of the robot is achieved by differentiating the rotational speed of the left and right wheels. Such type of drive enables a 180° turn in one point, which is of great importance while operating on industrial installations.

In order to enable the robot to move vertically along walls made of magnetic materials, special construction of wheels is necessary. Magnetic wheels made of magnetic steel and neodymium magnets are used. The suitable level of magnetic attraction is obtained by providing the necessary number of magnets and steel wheel shields. Apart from controlling the drives, the main microprocessor control system is also responsible for controlling two additional digital servo drives. One servo drive is used for moving the video camera; the other lifts and lowers the scanning head. The camera (water resistant) is installed to a tripod equipped with a bearing, enabling it to rotate within the range of $-15°$ to 45°. A four-bar mechanism is used for the lift system of the scanning head. The mechanism provides high rigidity of the head and enables the head to operate in parallel to the robot's performance. The four-bar mechanism and the servo lever are connected by means of a spring shock absorber, which enables the operator to control the holding down force of the robot. What is more, the heads are mounted on a transverse wishbone, due to which they fit close to the surface being examined. The robot and the control panel are connected with a cable. Functionality and ergonomics were the main criteria when designing the controlling panel. The panel comprises eight function buttons and two LCD displays – a 7" colour one presenting a live view from the camera and another one

(alphanumeric) providing status data of the robot and its modules. The panel is also equipped with an analog joystick.

The magneto robot is equipped with following actuators:

- two DC motors with gear driving the robot's wheels,
- a model servo for setting the tilt of the front rack, on which sonotrodes are mounted,
- a model servo for setting the camera's tilt.

The robot's set of sensor includes:

- two incremental optoelectronic encoders with push-pull type output, mounted on the motors' shafts for reading their position,
- a 3-axis accelerometer to gain robot's inclination.

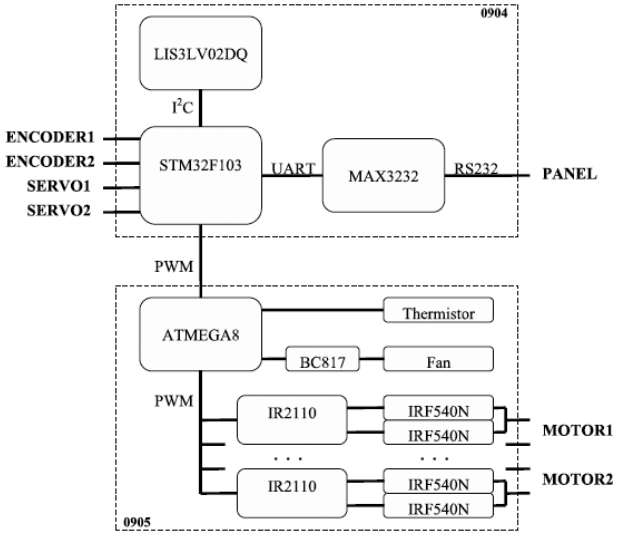


Fig. 6.3 Architecture of the robot driver

Figure 6.3 shows the robot controller's architecture. The controller consists of two separate modules, labelled 0904 and 0905.

0904 is the main control module, based on the ST Microelectronics STM32F103 ARM microcontroller. It is responsible for control of the two model servos, reading the motors' position from the encoders and the robot's inclination from the accelerometer (3-axis ST Microelectronics LIS3LV02DQ, also placed on the module). It also generates signal for power stage that drives the motors. The Maxim MAX3232 converter provides a serial interface for connecting the robot to a computer or the operator's panel. The microcontroller runs two duplicate PID regulators

that control the motors' speed. There is also a third regulator implemented that allows control of the robot's inclination, based on information from the accelerometer.

0905 is a power stage module. It consists of 2 H-bridges built of IRF540N MOS transistors, driven by International Rectifier IR2110 specialized ICs. The module is controlled with the Atmel ATmega8 microcontroller that generates signals for H-bridges, reads their temperature from the thermistor and enables a fan if temperature rises over a set limit. The input value for the 0905 module is the amount of power to be given to motors. Thanks to the ATmega microcontroller the type of input can be digital serial interface, voltage-level signal or pulse width modulated signal.

The robot's operation is controlled manually, using the operator's panel. It is equipped with a set of push-buttons, a 2-axis analog joystick, and a liquid crystal display. The panel's driver is based on the Atmel ATmega16 microcontroller that communicates with the robot through the serial interface using the MAX232 converter.

The robot can operate in one of the following modes:

- Free running mode. If the operator's panel is connected, the user sets the robot's velocity and course with joystick deflection.
- Constant speed mode. The robot's velocity is zero or a constant value set on the operator's panel using push-buttons. The deflection of the joystick controls only the robot's course. Putting the joystick back in the center position makes the robot stop. This mode is convenient for manual operation when the robot is to perform a scan with constant progress.
- Constant inclination on vertical surfaces. Deflecting the joystick in front-back direction causes the robot move forwards or backwards, keeping its inclination. The inclination can be adjusted by deflecting the joystick in left-right direction.
- Horizontal mode and vertical mode. Two special cases of driving with constant inclination on vertical surfaces.

In all the above modes the position of the robot's servomechanisms is set by pushing the corresponding button on the panel and simultaneous deflection of the joystick. The LCD displays the current mode name, the position of the servos, the motors' given and measured velocity, and the robot's inclination.

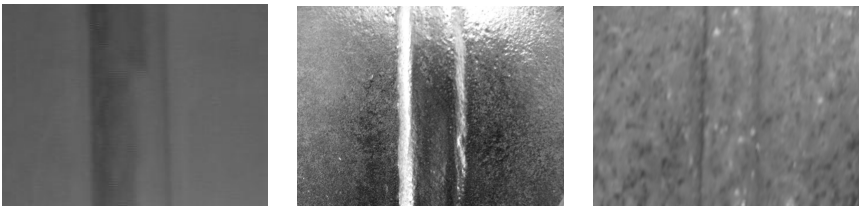
It is also possible to control the robot from a PC in each mode. The information of the encoders' increment, the servo position and the robot's inclination is sent periodically to the computer.

6.3 Vision System

The goal of the control system is to track the weld line and keep the best centered robot position above it. The program is divided into two parts: one part is responsible for detection of the weld line, the other controls the robot motion.

The task of following a weld is similar to the method of following a line and can be viewed as a special case of trajectory tracing [6]. In [5] a visual method which allows following a line by an autonomous mobile robot is described. The approach requires a vision system which detects a line with a high accuracy.

The classical approach to weld detection relies on image segmentation. The pixels are divided into two classes: *weld* and *background*. The segmentation is not easy because welds have different colors and structures. Figure 6.4 presents photos of welds.



(a) (b) (c)

Fig. 6.4 Examples of welds

In our project we decided to use structured light to detect the weld position [9, 10]. The vision system is based on a CCD camera and laser line projectors. The camera is mounted on the top of the robot's frame; its axis is perpendicular to the experiment surface and positioned strictly above the weld quality measurement area (Fig. 6.5). The focus range of the lens and the aperture are adjusted to minimize image blurring (wide depth of field), especially when the surface is not a plane (e.g. cylindrical pipes).

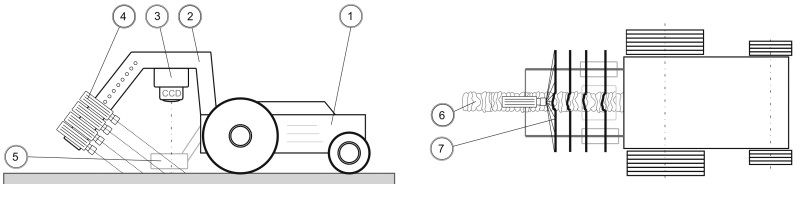


Fig. 6.5 Optical system configuration: 1 – robot, 2 – front frame, 3 – CCD camera, 4 – laser generators, 5 – measurement head, 6 – weld, 7 – laser lines

Laser generators are arranged to project parallel structural lines onto the base surface. Each of the generators is placed on a solid frame in front of the robot. The angle between the lasers and the camera axes is adjustable between $30^\circ - 60^\circ$ to obtain the best recognition of the weld position. The camera and the laser projectors initially chosen are vision range equipment (635 nm), but now infrared range hardware arrangement is being tested. In the actual configuration four laser generators are used as the optimum between analysis complexity and quality of recognition. Each of the parallel projected lines is shifted by $10 - 20$ mm in the direction of forward motion of the robot. This gives a possibility to make a prediction if further movement of the robot and the weld axes still overlap. During the experiments we

have found that slightly unfocused laser lines give better results in image analysis. Sharp and thin curves are more frequently distorted and bent on random material surface defects. An important condition is scene illumination and darkening external sources of light. The camera exposure should be adjusted to constant laser brightness and not to depend on daylight. The solution was a blind screen, cutting off direct sunlight or fluorescent lamp light.

The algorithm of weld detection consists of the following parts:

1. Thresholding using the Otsu algorithm [8] is performed in order to extract the projected lines from the background. Figure 6.6a presents the image and Fig. 6.6b shows the result of the thresholding procedure.
2. Noise reduction
   The flood-fill algorithm [4] is used in order to extract the brightest areas. If the area is small it is removed. It is shown in Fig. 6.6c.
3. The edges of the projected lines are detected (Fig. 6.6d).
4. The edges are analysed and the points where the edges bend are detected. The points indicate the edges of the weld (Fig. 6.6e).
5. Based on the coordinates of the points the equations of the left and right edges of the weld are computed using the regression method [12] (Fig. 6.6f).

Figure 6.6 presents the stages of image processing.

6.4 Control System

The control algorithm consists of communication, decision, and supervision modules. The robot's low-level software (implemented as an on-board microcontroller program) is designed to provide both manual and automatic modes of control. The high-level software runs on a portable PC computer connected to the robot. Communication between the computer and the robot allows instant changes of all motion parameters and reading the current hardware state, required in an automatic control mode. The communication subprogram ensures a direct link to the robot hardware providing movement of the robot and peripheral devices control (lasers, ambient light, servo). The decision subprogram is the most important one and its task is to judge if the measuring head is properly arranged relatively to the weld line. It is not a trivial operation, because the head is permanently attached to the robot frame and the positioning method is realized by robot movement only. The most significant is minimal deviation between the centers of a weld and the ultrasonic head. If the deviation increases, the quality of the scan is poor or could include errors. The program calculates the direction of each next move to compensate this distance. There are several common methods used in this type of control (e.g. the PID algorithm, fuzzy logic). We decided to use a fuzzy logic controller (FLC), taking into consideration the stability and flexibility of the method. The software is designed to run in different environments, varied types and sizes of welds. Rigid PID settings could cause problems and our fuzzy logic based algorithm was designed to avoid sensitivity to those changes. The FLC has also adaptation parameters, which could be set manually or be learned by the program according to the quality of control. An

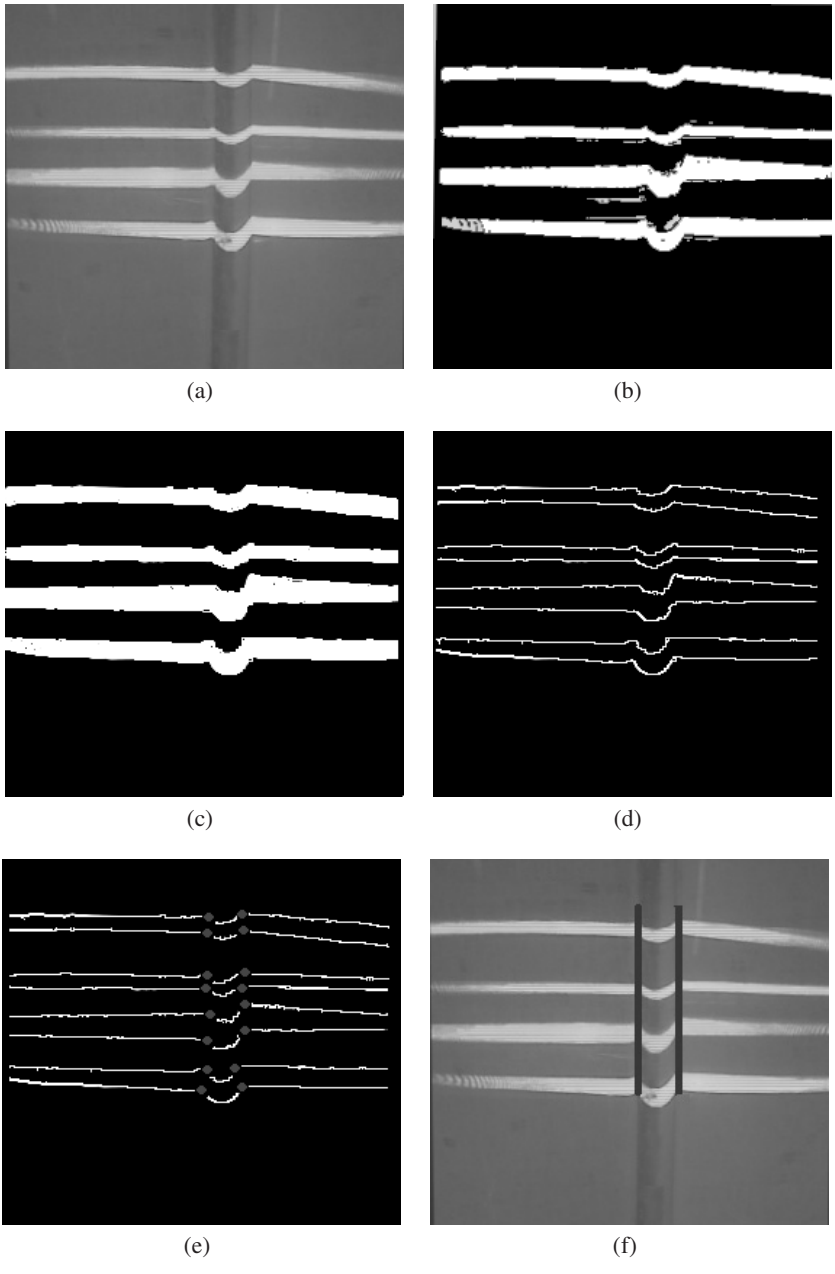


Fig. 6.6 Stages of image processing: (a) the original image of the weld, (b) the image after thresholding, (c) the image after noise reduction, (d) the edges of the strips, (e) the points of the weld edges, (f) the detected edges of the weld

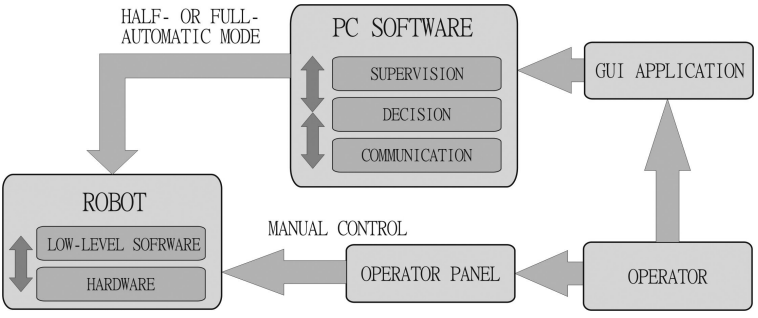


Fig. 6.7 Control modes

additional function is the ability to predict if the weld line will be sidetracking in the next movement. The decision program calculates future deviations of the current forward direction and each of the image-based recognized weld positions. Then it is considered in the FLC which generates a smooth, close-fitting path. Smoothing the velocities of the drives also reduces energy consumption and improves the quality of the measurement. Figure 6.8 presents the experimental results.

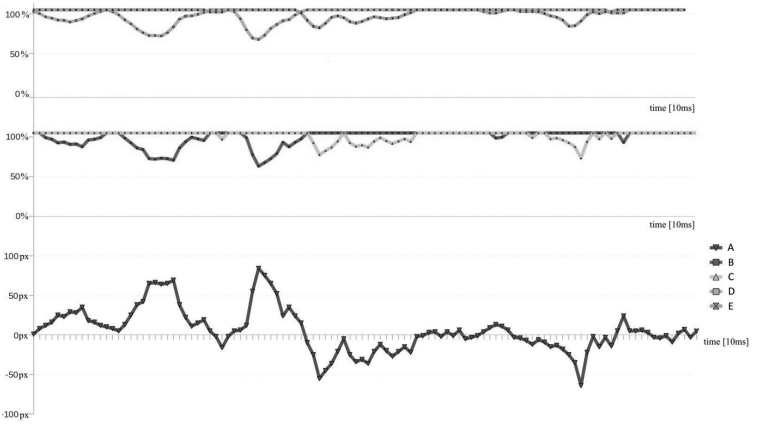


Fig. 6.8 Decision program time-charts: A – weld position deviation [pixels], B – left motor speed, C – right motor speed, D – left motor speed prediction, E – right motor speed prediction

The last part, the supervision program, is simultaneously communicating with the user (robot operator) and automatically performs all system tasks. The operator can manually (e.g. by a joystick) control the robot or work in a semi-automatic mode, in which only the speed and the main directions can be changed (Fig. 6.7). It is a convenient mode when the robot is out of visual reach and computer-aided driving is safest and more precise than full manual control. The third mode is fully automatic and allows the robot to follow the weld without an operator. When an unexpected

situation or spotted danger appears, the robot stops immediately and alerts the user. The supervision program also records the parameters of the weld and generates a report.

6.5 Conclusions

A mobile robot-based automated non-destructive weld joint testing system has been presented. The mobile platform is equipped with a vision system in order to detect the weld position. A fuzzy control system is used in order to control the robot's motion along the weld. Currently the system is undergoing testing in different conditions and with different types of welds.

References

1. Brillon, C., Armitt, T., Dupuistofd, O.: Inspection with Phased Arrays. In: 17th World Conference on Nondestructive Testing (2008)
2. Chen, Y.: The Application of TOFD Technique on the Large Pressure Vessel. In: 17th World Conference on Nondestructive Testing, pp. 25–28 (2008)
3. Gerla, G.: Fuzzy Logic Programming and fuzzy control. Studia Logica 79, 231–254 (2005)
4. Gonzalez, R.C., Woods, R.E., Eddins, S.L.: Digital Image Processing Using Matlab. Pearson Education, 245–255 (2004)
5. Kwolek, B., Kapuściński, T., Wysocki, M.: Vision-based implementation of feedback control of unicycle robots. In: 1st Work. on Robot Motion and Control, pp. 101–106 (1999)
6. Latombe, J.C.: Robot Motion Planning. Kluwer Academic Publishers, Norwell (1991)
7. Olympus, N.D.T.: Advances in Phased Array UltasonicTechniology Applications, Olympus NDT (2007)
8. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Sys., Man., Cyber. 9, 62–66 (1979)
9. Vuylsteke, P., Oosterlinck, A.: Range image acquisition with a single binary-encoded light pattern. PAMI 12(2), 148–164 (1990)
10. Young, M., Beeson, E., Davis, J., Rusinkiewicz, S., Ramamoorthi, R.: Viewpoint-coded structured light. In: CVPR (2007)
11. http://www.olympus-ims.com
12. Barlow, J.L.: Numerical aspects of Solving Linear Least Squares Problems in Rao, C.R. In: Computational Statistics, Handbook of Statistics, 9 (1993)

Chapter 7
Task-Priority Motion Planning of Wheeled Mobile Robots Subject to Slipping

Katarzyna Zadarnowska and Adam Ratajczak

7.1 Introduction

Wheeled mobile robots are most often assumed to be capable of rolling without slipping, and modeled as nonholonomic systems [3, 7, 19]. Such an assumption is far from realistic in practice. Since friction is the major mechanism for generating forces acting on the vehicle's wheels, the problem of modeling and predicting tire friction has always been an area of intense research in the automotive industry [4, 13, 17]. Modeling friction forces exerted at the wheels has been used for various studies [1, 2, 10, 15].

This paper introduces a task-priority motion planning algorithm of wheeled mobile robots for which nonholonomic constraints have been violated during the motion. Its derivation relies on the endogenous configuration space approach [19]. The concept of prioritizing the sub-tasks for redundant manipulators was set forth in [16] and examined in [6]. The endogenous configuration space methodology assumes a control system representation of the mobile robot.

The task-priority motion planning algorithm [21] allows planning the motion along with solving one or more additional tasks. These additional tasks may refer to ensuring a desirable quality of motion, keeping the configuration variables within some limits, avoiding singularities, preventing collisions, etc. In the paper we try to solve the following problem: given a desirable location of the mobile robot in the task space, find a configuration in which the robot reaches the desirable location in a prescribed time horizon with the total quantity of slipping during the motion as small as possible.

The inverse kinematic algorithm based on the inverse of the Jacobian is the most popular method of solving motion planning tasks [3, 14]. Limitations of the methods presented in the abovementioned literature have been discussed in [20]; the

Katarzyna Zadarnowska · Adam Ratajczak
Institute of Computer Engineering, Control, and Robotics,
Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: {katarzyna.zadarnowska,adam.ratajczak}@pwr.wroc.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 75–85.
springerlink.com

inverse Jacobian method is not applicable to mobile robots. Inverse kinematics and dynamics algorithms based on the pseudo inverse of the Jacobian for mobile robots have been examined in [19]. The motion planning problem of mobile robots subject to slipping effects examined using traditional methods have been presented in [1, 11, 5, 18].

A specific contribution of this paper lies in constructing the task-priority motion planning algorithm based on the pseudo inverse of the Jacobian. We show the functionality of the algorithm elaborated within the endogenous configuration space approach. Starting with a description of the robot dynamics, we assume a linear dependence of the traction forces on the slip of the wheels. Then, we incorporate the obtained traction forces into the improved singular perturbation model [1]. We use the stiffness coefficients to describe the character of the slip and its magnitude. To our best knowledge, an application of the endogenous configuration space methodology to the task-priority motion planning problem for mobile robots subject to slipping have not been tackled yet. We will show that the sophisticated nature of the presented algorithm, designed by combining the improved singular perturbation modeling and the endogenous configuration space approach, deals with the motion planning problem along with the slipping minimization. By design, the motion planning algorithm proposed in this paper applies to any control system representation having controllable linear approximation along the control-trajectory pair. The proposed motion planning method provides the open-loop control functions. Nevertheless, there exist some modifications of the method which take into consideration the uncertainties of the model [12].

The paper is composed as follows. Section 7.2 presents an analysis of mobile robotic systems subject to slipping effects. The analysis, patterned on [1], makes use of explicit modeling of the dissipative nature of the interaction forces applied to the system by the external world. Section 7.2 also summarizes basic concepts of the endogenous configuration space approach and introduces the derivation of the task-priority motion planning algorithm. Section 7.3 contains the simulation results related to the application of the presented motion planning algorithm to the Pioneer 2DX mobile robot moving with slipping. The paper is concluded with Section 7.4.

7.2 Basic Concepts

Let us consider a class of wheeled mobile robots where the nonholonomic constraints (e.g. pure rolling, non-slipping conditions at the contact point of each wheel with the ground) are not satisfied. Let $q \in \mathbb{R}^n$ denote the generalized coordinates of the mobile robot. We shall assume that l ($l < n$) velocity constraints $\mathsf{A}(q)\dot{q} = 0$, imposed on the robot motion, can be violated. The violation of the constraints is measured by the norm $\|\mathsf{A}(q)\dot{q}\|$. Using the canonical decomposition of $\mathbb{R}^n = \ker \mathsf{A}(q) \oplus \mathrm{im}\mathsf{A}^{\mathrm{T}}(q)$, quasi-velocities $\eta \in \mathbb{R}^m$, $\mu \in \mathbb{R}^l$, $m + l = n$, are defined, so that the mobile robot kinematics is represented as

$$\dot{q} = G(q)\eta + \mathsf{A}^{\mathrm{T}}(q)\mu\,. \tag{7.1}$$

The columns $g_1(q),\ldots,g_m(q)$ of the matrix $G(q)$ span the null space $\ker \mathsf{A}(q)$, $\eta \in \mathbb{R}^m$ – vector of auxiliary velocities and $\mu \in \mathbb{R}^l$ – vector of slipping velocities providing an information about relative importance of the different violations of constraints. The introduction of quasi-velocities was originally proposed in [1] as a key element of the singular perturbation approach.

The Lagrange equations of the mobile robot dynamics assume the form:

$$\mathsf{Q}(q)\ddot{q} + \mathsf{C}(q,\dot{q}) = F(q) + \mathsf{B}(q)\mathsf{u}\,. \tag{7.2}$$

The vector $F(q)$ denotes the interaction forces (exerted on the system by the external world). In the case when the ideal nonholonomic constraints are satisfied, the vector $F(q)$ defines the forces which, somehow, assume values causing satisfaction of the velocity constraints $F(q) = A^{\mathrm{T}}(q)\lambda$. In practice, predominantly due to various effects such as slipping, scrubbing, sliding, deformability or flexibility of the wheels, it is impossible to provide values of $F(q)$ guaranteeing satisfaction of the nonholonomic constraints. When a violation of the constraints is permitted, these forces have dissipative nature.

Having rewritten the equations of motion (7.2) and with $\ddot{q}$ expressed from (7.1) as $\ddot{q} = \left[G(q)\ \mathsf{A}^{\mathrm{T}}(q)\right]\begin{pmatrix}\dot{\eta}\\ \dot{\mu}\end{pmatrix} + \dot{G}(q)\eta + \dot{\mathsf{A}}^{\mathrm{T}}(q)\mu$, where $\dot{G}(q) = \frac{\partial G(q)}{\partial q}(G(q)\eta + \mathsf{A}^{\mathrm{T}}(q)\mu)$ and $\dot{\mathsf{A}}^{\mathrm{T}}(q) = \frac{\partial \mathsf{A}^{\mathrm{T}}(q)}{\partial q}(G(q)\eta + \mathsf{A}^{\mathrm{T}}(q)\mu)$ and adding an output function characterizing the task of the mobile robot, we obtain an affine control system representation of the kinematics and the dynamics of the mobile robot:

$$\begin{cases} \dot{q} &= G(q)\eta + \mathsf{A}^{\mathrm{T}}(q)\mu \\ \begin{pmatrix}\dot{\eta}\\ \dot{\mu}\end{pmatrix} &= P(q,\eta,\mu) + R(q)\mathsf{u} \\ y &= k(q,\eta,\mu)\,. \end{cases} \tag{7.3}$$

The terms appearing in (7.3) are defined in the following way:

$$\begin{aligned} P(q,\eta,\mu) = \begin{bmatrix} P_1(q,\eta,\mu) \\ P_2(q,\eta,\mu)\end{bmatrix} &= \left[G(q)\ \mathsf{A}^{T}(q)\right]^{-1}\left(-(\dot{G}(q)\eta + \dot{\mathsf{A}}^{T}(q)\mu)\right. \\ &\left. -Q^{-1}(q)\mathsf{C}(q, G(q)\eta + \mathsf{A}^{T}(q)\mu)\right) + H^{-1}(q)\begin{bmatrix}0\\ \mathsf{A}(q)F\end{bmatrix}, \end{aligned} \tag{7.4}$$

$$R(q) = \begin{bmatrix} R_1(q) \\ R_2(q)\end{bmatrix} = \left[G(q)\ \mathsf{A}^{T}(q)\right]^{-1} Q^{-1}(q)\mathsf{B}(q)\,. \tag{7.5}$$

The matrix $H(q) = \begin{bmatrix} G^{\mathrm{T}}(q)\mathsf{Q}(q)G(q) & G^{\mathrm{T}}(q)\mathsf{Q}(q)\mathsf{A}^{\mathrm{T}}(q) \\ \mathsf{A}(q)\mathsf{Q}(q)G(q) & \mathsf{A}(q)\mathsf{Q}(q)\mathsf{A}^{\mathrm{T}}(q)\end{bmatrix}$ is symmetric and positive definite, while the output function $y = k(z)$ may describe the position coordinates or velocities of the mobile robot in its motion plane.

Since the admissible control functions of the mobile robot (7.3) belong to a Hilbert space, they will be assumed Lebesgue square integrable over some time interval, $\mathsf{u}(\cdot) \in L^2_m[0,T]$. They have the sense of forces/torques, and so constitute the *dynamic endogenous configurations* $\mathscr{U} \cong L^2_m[0,T]$ of the mobile robot [20].

Given the system (7.3), we shall study the following motion planning problem for the mobile robot subject to slipping: given a desirable point y_d in the task space, and time horizon $[0,T]$, the motion planning problem consists in defining the control functions $\mathsf{u}_\mathsf{d}(\cdot) \in \mathscr{U}$ such that the system output, starting from the initial y_0 point, reaches the desirable point $\mathsf{T}_{z_0,T}(\mathsf{u}_\mathsf{d}(\cdot)) = y_d$ (sub-task S_1). Moreover, the total quantity of slipping during the motion should be as small as possible (sub-task S_2). These two sub-tasks define a task-priority motion planning problem [21], where the sub-tasks can be written symbolically as $S_1\colon y_0 \xrightarrow{\mathsf{u}_\mathsf{d}(\cdot)} y_d$, $S_2\colon \int_0^T \mu^\mathsf{T}\mu\, Xt \xrightarrow{\mathsf{u}_\mathsf{d}(\cdot)} \min$ and are ordered according to decreasing priorities.

Consider now the proper motion planning (sub-task S_1). Let $z = (q,\eta,\mu) \in \mathbb{R}^{n+m+l}$ denote the state vector of (7.3), and suppose that for a given initial state z_0 and every control $\mathsf{u}(\cdot)$ there exists a state trajectory $z(t) = \varphi_{z_0,t}(\mathsf{u}(\cdot)) = (q(t),\eta(t),\mu(t))$ and an output trajectory $y(t) = k(z(t))$ of system (7.3). In accordance with the endogenous configuration space approach the task map for sub-task S_1 of the mobile robot is identified with the end point map of the control system (7.3), i.e.

$$\mathsf{T}_{1,z_0,T}(\mathsf{u}(\cdot)) = y(T) = k(\varphi_{z_0,T}(\mathsf{u}(\cdot)))\,, \tag{7.6}$$

and computes the system output at T, when driven by $\mathsf{u}(\cdot)$. The Jacobian of the mobile robot can be defined as the derivative of $\mathsf{T}_{1,z_0,T}$ [19]:

$$\mathsf{J}_{1,z_0,T}(\mathsf{u}(\cdot))\mathsf{v}(\cdot) = D\mathsf{T}_{1,z_0,T}(\mathsf{u}(\cdot))\mathsf{v}(\cdot)\,, \qquad \mathsf{v}(\cdot) \in L^2_m[0,T]\,, \tag{7.7}$$

$\mathsf{v}(\cdot) \in \mathscr{U}$. The Jacobian map transforms tangent vectors to the dynamic endogenous configuration space into $\mathbb{R}^3$ and describes how an infinitesimal change in the input force is transmitted into a change of the position and orientation of the mobile robot at T.

In order to compute the Jacobian map at a given configuration $\mathsf{u}(\cdot) \in \mathscr{U}$ we introduce a variational system associated with (7.3):

$$\dot{\xi} = A(t)\xi + B(t)\mathsf{v},\ \zeta = C(t)\xi \tag{7.8}$$

as the linear approximation to (7.3) along $z(t)$, initialized at $\xi_0 = 0$, where

$$A(t) = \begin{bmatrix} \frac{\partial(G(q(t))\eta(t)+\mathsf{A}(q(t))\mu(t))}{\partial q} & G(q(t)) & \mathsf{A}^\mathsf{T}(q(t)) \\ \frac{\partial(P(q(t),\eta(t),\mu(t))+R(q(t))\mathsf{u}(t))}{\partial q} & \frac{\partial P(q(t),\eta(t),\mu(t))}{\partial \eta} & \frac{\partial P(q(t),\eta(t),\mu(t))}{\partial \mu} \end{bmatrix},$$

$$B(t) = \begin{bmatrix} 0 \\ R(q(t)) \end{bmatrix}, \qquad C(t) = \frac{\partial k(z(t))}{\partial z}\,.$$

Taking into account (7.3) and (7.8), we can compute the Jacobian map as the output trajectory $\zeta(T)$ of the variational system (7.8):

$$\mathsf{J}_{1,z_0,T}(\mathsf{u}(\cdot))\mathsf{v}(\cdot) = C(T)\int_0^T \Phi(T,t)B(t)\mathsf{v}(t)\,Xt\,, \tag{7.9}$$

where $\Phi(t,\mathsf{s})$ denotes the fundamental matrix of system (7.8), that satisfies the evolution equation $\frac{\partial \Phi(t,\mathsf{s})}{\partial t} = A(t)\Phi(t,\mathsf{s})$, $\Phi(\mathsf{s},\mathsf{s}) = I_{n+m+l}$. Observe that the Jacobian (7.9) corresponds to the compliance map introduced in [20]. Finally, we will also need the pseudo inverse of the Jacobian

$$\left(\mathsf{J}^{\#}_{1,z_0,T}(\mathsf{u}(\cdot))\zeta\right)(t) = B(t)\Phi^{\mathrm{T}}(T,t)\mathsf{M}^{-1}_{z_0,T}(\mathsf{u}(\cdot))\zeta\,, \tag{7.10}$$

where $\mathsf{M}_{z_0,T}(\mathsf{u}(\cdot)) = C(T)\int_0^T B(t)\Phi(T,t)\Phi^{\mathrm{T}}(T,t)B^{\mathrm{T}}(t)XtC^{\mathrm{T}}(T)$ is the *mobility matrix* of the mobile platform.

Now, let us take into account the sub-task S_2 – slipping minimization. Following the line of reasoning for S_1 we define the task map for S_2 as

$$\mathsf{T}_{2,z_0,T}(\mathsf{u}(\cdot)) = \int_0^T \mu^{\mathrm{T}}(t)\rho\mu(t)Xt\,, \tag{7.11}$$

where $\rho = \mathrm{diag}\{r_1, r_2, \ldots, r_l\} \in \mathbb{R}^{l\times l}$ is a diagonal weight matrix. The differentiation of $\mathsf{T}_{2,z_0,T}$ yields the Jacobian

$$\mathsf{J}_{2,z_0,T}(\mathsf{u}(\cdot))\mathsf{v}(\cdot) = \int_0^T\!\!\int_s^T \mu^{\mathrm{T}}(t)\mathsf{R}\Phi(t,s)B(s)\mathsf{v}(s)Xt\,Xs, \qquad \mathsf{R} = \begin{bmatrix} 0 & \cdots & 0 & r_1 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & r_2 & \cdots & 0 \\ \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & 0 & 0 & \cdots & r_l \end{bmatrix}, \tag{7.12}$$

and $\mathsf{R} \in \mathbb{R}^{l\times n}$. The Jacobian pseudo inverse for $\mathsf{J}_{2,z_0,T}(\mathsf{u}(\cdot))v(\cdot)$ is defined as

$$\left(\mathsf{J}^{\#}_{2,z_0,T}(\mathsf{u}(\cdot))\zeta\right)(t) = \frac{\beta(t)}{\|\beta(t)\|}\zeta\,, \tag{7.13}$$

where $\beta(t) = B^{\mathrm{T}}(t)\int_t^T \Phi^{\mathrm{T}}(s,t)\mathsf{R}^{\mathrm{T}}\mu(s)Xs$ and $\zeta \in \mathbb{R}$.

Having defined the task maps, Jacobians and Jacobian pseudo inverses for both tasks we are ready to define the algorithm.

The motion planning problem may be solved numerically by means of a Jacobian pseudo inverse motion planning algorithm. Let $\mathsf{u}_\vartheta(\cdot) \in \mathscr{U}$, $\vartheta \in \mathbb{R}$ denote a smooth curve in $\mathscr{U}$, passing through an initial configuration $\mathsf{u}_0(\cdot)$. The task space error of the ith sub-task $e_i(\vartheta)$ along this curve should decrease exponentially along with ϑ. In the case of exponential decrease, $\frac{X}{X\vartheta}e_i(\vartheta) = -\gamma_i e_i(\vartheta)$, with decay rate $\gamma_i > 0$, using the Jacobian pseudo inverse operator, we get the algorithm for the ith sub-task:

$$\frac{X\mathsf{u}_\vartheta(\cdot)}{X\vartheta} = -\gamma_i \mathsf{J}^{\#}_{i,z_0,T}(\mathsf{u}(\cdot))e_i(\vartheta) + \mathsf{P}_i(\mathsf{u}(\cdot))\varsigma_i(\cdot)\,, \tag{7.14}$$

where $\mathsf{P}_i(\mathsf{u}(\cdot)) = \mathrm{id}_{\mathscr{U}} - \mathsf{J}^{\#}_{i,z_0,T}(\mathsf{u}(\cdot))\mathsf{J}_{i,z_0,T}(\mathsf{u}(\cdot))$ is the projection of endogenous configuration $\mathscr{U}$ onto $\ker \mathsf{J}_{i,z_0,T}(\mathsf{u}(\cdot))$. Function $\varsigma_i(\cdot)$ denotes the directions in $\mathscr{U}$ and is useful to include into the algorithm the lower priority task. Finally, the dynamic system defining the Jacobian pseudo inverse algorithm with task-priority [21] is as follows:

$$\frac{d\mathsf{u}_\vartheta(\cdot)}{d\vartheta} = -\gamma_1 \mathsf{J}^{\#}_{1,z_0,T}(\mathsf{u}(\cdot))e_1(\vartheta) - \gamma_2 \mathsf{P}_1(\mathsf{u}(\cdot))\mathsf{J}^{\#}_{2,z_0,T}(\mathsf{u}(\cdot))e_2(\vartheta), \tag{7.15}$$

where the task space errors are $e_1(\vartheta) = \mathsf{T}_{z_0,T}(\mathsf{u}_\vartheta(\cdot)) - y_d$, $e_2(\vartheta) = \mathsf{T}_{2,z_0,T}(\mathsf{u}(\cdot)) = \int_0^T \mu^{\mathrm{T}}(t)\rho\mu(t)Xt$. The $+\infty$ limit of the trajectory of (7.15), $\mathsf{u}(t) = \lim_{\vartheta\to+\infty}\mathsf{u}_\vartheta(t)$, provides a solution of the motion planning problem.

Since the dynamic endogenous configuration space $\mathscr{U}$ is infinite-dimensional, to carry out effective computations we shall use a finite parameterization of the control functions $\mathsf{u}(\cdot)$ in (7.3) by truncated orthogonal expansions (Fourier series)

$$\mathsf{u}_{ci}(t) = \sum_{j=0}^{k} c_{i2j-1}\sin j\omega t + c_{i2j}\cos j\omega t, \quad i = 1,2,\ldots,m \quad \omega = \frac{2\pi}{T}, \quad c_{i-1} = 0. \tag{7.16}$$

Subscript c in (7.16) means that the control functions are parameterized. After parameterization, the control function $\mathsf{u} \in \mathscr{U}$ is represented by a vector $c \in \mathbb{R}^s$, $s = m(2k+1)$. A discretization of the motion planning algorithm (7.15) results in changing the control function $c \in \mathbb{R}^s$ iteratively, with iterations indexed by an integer ϑ, i.e.

$$c_{\vartheta+1} = c_\vartheta - \gamma_1 \widetilde{\mathsf{J}}^{\#}_{1,z_0,T}(c_\vartheta)\widetilde{e}_{1,\vartheta} - \gamma_2 \widetilde{\mathsf{P}}_1(c_\vartheta)\widetilde{\mathsf{J}}^{\#}_{2,z_0,T}(c_\vartheta)\widetilde{e}_{2,\vartheta}, \quad \vartheta = 1,2,\ldots, \tag{7.17}$$

where the finite dimensional Jacobian and the corresponding projection are, respectively, $\widetilde{\mathsf{J}}^{\#}_{i,z_0,T}(c) = \widetilde{\mathsf{J}}^{\mathrm{T}}_{i,z_0,T}(c)\left(\widetilde{\mathsf{J}}_{i,z_0,T}(c)\widetilde{\mathsf{J}}^{\mathrm{T}}_{i,z_0,T}(c)\right)^{-1}$ and $\widetilde{\mathsf{P}}_1(c) = I - \widetilde{\mathsf{J}}^{\#}_{1,z_0,T}(c)\widetilde{\mathsf{J}}_{1,z_0,T}(c)$, and the task space errors for both sub-tasks are defined as $\widetilde{e}_{1,\vartheta} = \mathsf{T}_{1,z_0,T}(u_{c_\vartheta}(\cdot)) - y_d$ and $\widetilde{e}_{2,\vartheta} = \mathsf{T}_{2,z_0,T}(u_{c_\vartheta}(\cdot))$.

7.3 Case Study

As an example, let us consider Pioneer 2DX – a differential drive type mobile robot depicted in Fig. 7.1. Let us assume that the wheels may slip longitudinally as well as laterally. In order to preserve dimensional consistence, the generalized platform coordinates are chosen as $q = (x, y, l\theta, r\varphi_1, r\varphi_2)^T$. The meaning of the geometric parameters of the robot along with its coordinates is explained in Fig. 7.1. The

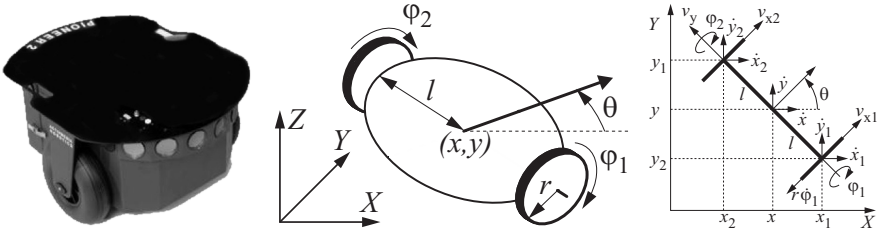


Fig. 7.1 Mobile robot Pioneer 2DX

mathematical model of the kinematics and dynamics of the mobile robot is represented by the control system (7.3) with the following matrices:

$$\mathsf{A}(q)=\begin{bmatrix}\sin\theta & -\cos\theta & 0 & 0 & 0\\ \cos\theta & \sin\theta & 1 & -1 & 0\\ \cos\theta & \sin\theta & -1 & 0 & -1\end{bmatrix},\quad G(q)=\begin{bmatrix}\cos\theta & \sin\theta & 1 & 2 & 0\\ \cos\theta & \sin\theta & -1 & 0 & 2\end{bmatrix}^{\mathrm{T}},\quad \mathsf{B}=\begin{bmatrix}0 & 0 & 0 & 1/r & 0\\ 0 & 0 & 0 & 0 & 1/r\end{bmatrix}^{\mathrm{T}},$$

and $\mathsf{Q}=\operatorname{diag}\left\{m,m,I_\theta/l^2,I_\varphi/r^2,I_\varphi/r^2\right\}$, where m – the robot mass, I_θ – the moment of inertia around the robot vertical axis, and I_φ – the moment of inertia of each wheel. The output function describes the position coordinates and the orientation angle of the robot in its motion plane ($y(z)=y(q)=(x,y,\theta)\in\mathbb{SE}(2)\cong\mathbb{R}^2\times\mathbb{S}^1$).

Now, let us derive a model for the generalized interaction force F. As in [1], we shall adopt a "pseudo-slipping" model [15], according to which the lateral and longitudinal forces applied by the ground to the wheels are proportional, respectively, to the slip angle and the slip coefficient.

Let v_i denote the velocity of the center of the ith wheel which rotates with the angular velocity $\dot\varphi_i$. Let v_{ix} and v_{iy} be, respectively, the longitudinal and lateral components of v_i; then the norm is defined as $\|v_i\|=\sqrt{v_{ix}^2+v_{iy}^2}=\sqrt{(2(\eta_i+\mu_{i+1}))^2+\mu_1^2}$.

The longitudinal traction force f_x and the lateral traction force f_y applied by the ground are characterized as

$$f_x=-Cs=-\frac{C}{\|v\|}(v_x-r\dot\varphi),\qquad f_y=-\frac{D}{\|v\|}v_y,\tag{7.18}$$

where C is a slip stiffness coefficient and D denotes a cornering stiffness coefficient, depending on the nature of the wheel and the ground. Finally, we get the following form of the interaction force acting on the wheel:

$$f=\begin{bmatrix}f_x\\ f_y\end{bmatrix}=-\frac{1}{\|v\|}\begin{bmatrix}C & 0\\ 0 & D\end{bmatrix}\begin{bmatrix}v_x-r\dot\varphi\\ v_y\end{bmatrix}.\tag{7.19}$$

Following the approach presented in [1], we define

$$F_i=-\mathsf{A}^T(q)L_i^T(q)\frac{1}{\|v_i\|}\begin{bmatrix}C_i & 0\\ 0 & D_i\end{bmatrix}L_i(q)\mathsf{A}(q)\dot q,\quad i=\{1,2\},\tag{7.20}$$

where $L_1(q)=\left[\begin{smallmatrix}0 & 1 & 0\\ -1 & 0 & 0\end{smallmatrix}\right]$, $L_2=\left[\begin{smallmatrix}0 & 0 & 1\\ -1 & 0 & 0\end{smallmatrix}\right]$.

Now, let us explain the way of choosing the stiffness coefficients D_i and C_i. In computations we assume that D_i and C_i have the same value for both wheels. The traction force is proportional to the normal force $F_z=mg$ and is equal $f_T=\mu_xF_z$, where μ_x is the friction coefficient describing the type of the road. Its values depend on the tire type, the road conditions, etc. Taking the longitudinal traction force as in (7.18) and comparing $|f_T|=|f_x|$ we obtain $C=\frac{\mu_xF_z}{s}$. We take $s=0.25$. It is the value of the longitudinal slip for which the friction coefficient μ_x is maximum for all road types (see Fig. (2.17) in [8]). Eventually, we assume $D=2C$, meaning that the slip in the lateral direction is twice less than the slip in the longitudinal direction. Generally,

the bigger D and C are, to a higher degree the velocity constraints are taken into consideration, and therefore the less influence of the slip effects we observe.

In order to avoid numerical problems that may appear for small values of $\|v_i\|$ we slightly modify the model by introducing a saturation. In particular, if $\|v_i\| < \delta$ then $\|v_i\|$ is replaced by δ, where δ is a small positive constant. To the needs of computer simulations we assume the following real geometric and dynamic parameters of the mobile platform Pioneer 2DX: $l = 0.163\,\mathrm{m}$, $r = 0.0825\,\mathrm{m}$, $m = 8.67\,\mathrm{kg}$, $I_\theta = 0.256\,\mathrm{kg\,m^2}$, $I_\varphi = 0.02\,\mathrm{kg\,m^2}$ [9]. Additionally, we assume the saturation coefficient $\delta = 10^{-6}$. We take the control parameterization $c \in \mathbb{R}^{34}$, its initial value as $c_0 = (0.2, 0_{1\times 16}, 0.2, 0_{1\times 16})$ and the decay rates $(\gamma_1, \gamma_2) = (0.5, 0.7)$. As the initial state we take $z_0 = (q_0, \eta_0, \mu_0) = (0_{1\times 10}, 10^{-4}, 10^{-4})$, the desirable point as $y_d = (5, 5, l\pi/2)$ and the time horizon $[0, 15]\,\mathrm{s}$. The main sub-task is solved when $\|e_1\|$ drops below 10^{-3}. We create the following simulation scenario: for two highly different friction coefficients (two different types of the road, i.e. dry asphalt $C = 408$ and ice $C = 34$) we try to solve the motion planing problem with the slipping minimization sub-task S_2 ($\gamma_2 = 0.7$) and without this task ($\gamma_2 = 0$). The simulation results are shown in Figs. 7.2–7.3. One can observe that for both road types the longitudinal as well as lateral slips are smaller for the cases with the second task active. Actually, the task map (7.11) is the measure of total slipping. For the dry asphalt the value of $\mathsf{T}_{2,z_0,T}$ is equal 5.08 10^{-4} without S_2 and 2.22 10^{-4} with S_2, hence the profit is

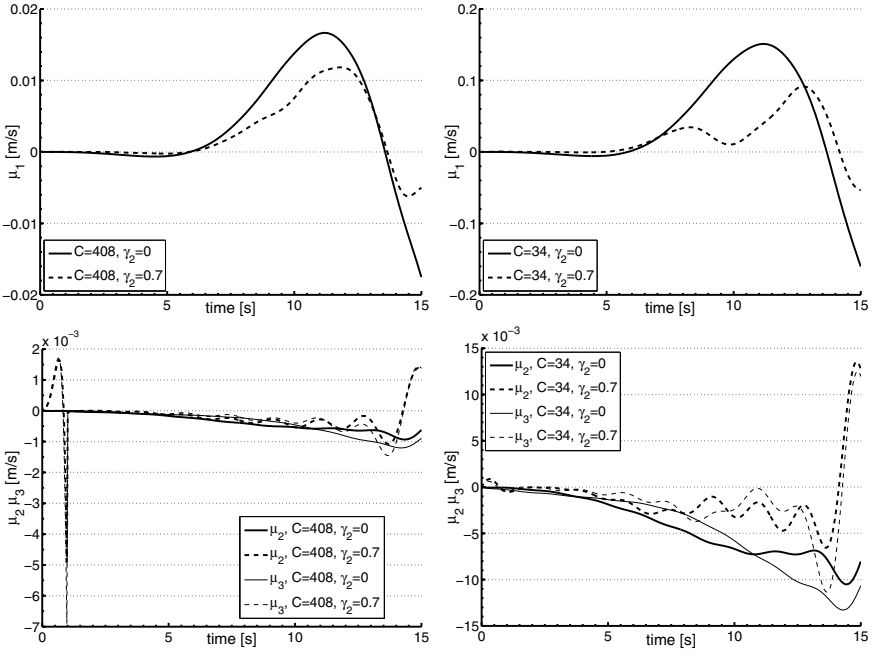


Fig. 7.2 Trajectories of the longitudinal μ_1 and lateral μ_2, μ_3 slipping velocities; without and with the second sub-task

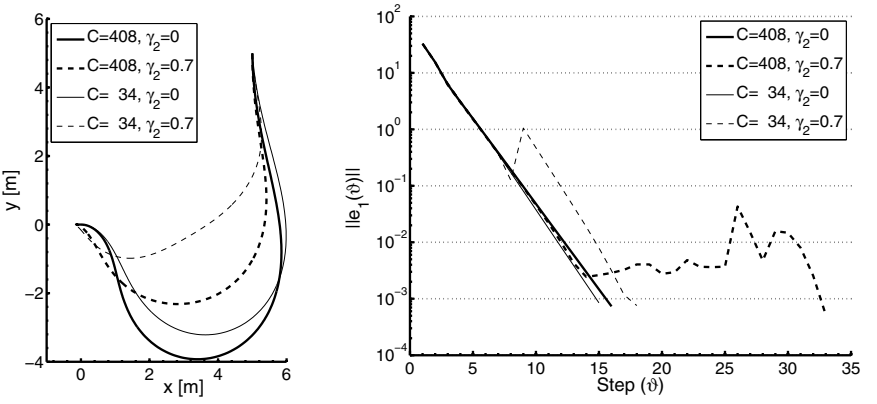


Fig. 7.3 Robot paths (left) and algorithm convergence (right)

small. However, inspecting the $\mathsf{T}_{2,z_0,T}$ values for the ice ($4.38\ 10^{-2}$ without S_2 and $9.20\ 10^{-3}$ with S_2) one can notice that the benefit is bigger. Moreover, a closer examination of Fig. 7.3 (left) shows that the paths corresponding to the cases with S_2 are shorter than without S_2. This means that the velocities must be lower, which has its consequences in a smaller quantity of slipping. As can be seen from Fig. 7.3 (right), the reduction of slipping absorbs more steps of the algorithm.

7.4 Conclusions and Future Work

In the paper we have presented a task-priority motion planning algorithm for a mobile robot subject to slipping, based on the endogenous configuration space approach. We have also derived a complete robot model with traction forces and proposed an algorithm which besides a proper motion planning minimizes the total quantity of slipping. Simulation results show the performance of our application.

In future work we will focus on another representation of the robot dynamics with traction forces. Now, our system (7.3) has two kinds of nonholonomic constraints. The first degree constraint comes from the non-slipping part $G(q)\eta$, and the second order comes from the robot dynamics. It can be shown that the quasi-velocities appearing in (7.1) could be removed from the final system (7.3), and the whole system can be rewritten using the second order equations (7.2). In such a definition of the system the slipping minimization subtask should take the following form: $\int_0^T (A(q)\dot{q})^{\mathrm{T}} A(q)\dot{q}\, Xt$.

Acknowledgements. The research of the first author was supported by the Polish State Committee of Scientific Research under a statutory grant; the work of the second author was supported by the funds for Polish science in the years 2010-2012 as a research project.

References

1. d'Andrea-Novel, B., Campion, G., Bastin, G.: Control of wheeled mobile robots not satisfying ideal velocity constraints: a singular perturbation approach. Int. J. Robust and Nonlinear Control 5, 243–267 (1995)
2. Angelova, A., Matthies, L., Helmick, D.M., Perona, P.: Learning and prediction of slip visual information. Journal of Field Robotics 24(3), 205–231 (2007)
3. Bayle, B., Fourquet, J.-Y., Renaud, M.: Manipulability of wheeled mobile manipulators: application to motion generation. Int. J. Robot. Res. 22(7–8), 565–581 (2003)
4. Canudas de Wit, C., Tsiotras, P., Velenis, E., Basset, M., Gissinger, G.: Dynamic Friction Models for Road/Tire Longitudinal Interaction. Vehicle System Dynamics 39(3), 189–226 (2003)
5. Cheng, P., Frazzoli, E., Kumar, V.: Motion Planning for the Roller Racer with a Sticking/Slipping Switching Model. In: Proc. of the IEEE Conf. on Robotics and Automation, Orlando, FL, pp. 1637–1642 (2006)
6. Chiaverini, S.: Singularity–robust task–priority redundancy resolution for real-time kinematic control of robot manipulators. IEEE Trans. on Robot. and Automat. 13, 398–410 (1997)
7. Cortes Monforte, J.: Geometric, Control and Numerical Aspects of Nonholonomic Systems. Springer, NY (2002)
8. Genta, G.: Motor Vehicle Dynamics: Modeling and Simulation. Series on Advances in Mathematics for Applied Sciences, vol. 43. World Scientific Printers, Singapore (1997)
9. Giergiel, J., et al.: Symboliczne generowanie równań kinematyki mobilnego robota Pioneer – 2DX. Przegląd Mechaniczny, 26–31 (2000) (in Polish)
10. Iagnemma, K., Dubovsky, S.: Traction Control of Wheeled Robotic Vehicles in Rough Terrain with Application to Planetary Rovers. Int. J. of Robotics Research 23(10–11), 1029–1040 (2004)
11. Ishigami, G., Nagatani, K., Yoshida, K.: Path Following Control with Slip Compensation on Loose Soil for Exploration Rover. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robotics and Systems, Beijing, China, pp. 5552–5557 (2006)
12. Jung, S., Wen, J.T.: Nonlinear Model Predictive Control for the Swing–Up of a Rotary Inverted Pendulum. J. of Dyn. Systems, Measurement, and Control 126, 666–673 (2004)
13. Kiencke, U., Nielsen, L.: Automotive Control Systems. Springer, Heidelberg (2000)
14. de Luca, A., Oriolo, G., Giordano, P.R.: Kinematic modelling and redundancy resolution for nonholonomic mobile manipulators. In: Proc. of the IEEE Int. Conf. on Robotics and Automation, Orlando, FL, pp. 1867–1873 (2006)
15. Matsumoto, N., Tomizuka, M.: Vehicle lateral velocity and yaw rate control with two independent control inputs. Transaction of the ASME, J. of Dynamics Systems, Measurement, and Control 114, 606–613 (1992)
16. Nakamura, Y.: Advanced Theoretical Robotics: Redundancy and Optimization. Prentice-Hall, New Jersey (1991)
17. Pacejka, H.B., Sharp, R.S.: Shear Force Developments by Pneumatic Tires in Steady–State Conditions: A Review of Modeling Aspects. Vehicle Systems Dynamics 20, 121–176 (1991)

18. Sidek, N., Sarkar, N.: Dynamic Modeling and Control of Nonholonomic Mobile Robot with Lateral Slip. In: Third Int. Conf. on Systems, pp. 35–40 (2008)
19. Tchoń, K., Jakubiak, J.: Endogenous configuration space approach to mobile manipulators: A derivation and performance assessment of Jacobian inverse kinematics algorithms. Int. J. Control 76(14), 1387–1419 (2003)
20. Zadarnowska, K., Tchoń, K.: A control theory framework for performance evaluation of mobile manipulators. Robotica 25, 703–715 (2007)
21. Ratajczak, A., Karpińska, J., Tchoń, K.: Task–priority motion planning of underactuated systems: an endogenous configuration space approach. Robotica 28, 885–892 (2010), with erratum 28, 943–943

Part II
Control of Legged Robots

Chapter 8
First Steps toward Automatically Generating Bipedal Robotic Walking from Human Data

Aaron D. Ames

Abstract. This paper presents the first steps toward automatically generating robotic walking from human walking data through the use of human-inspired control. By considering experimental human walking data, we discover that certain outputs of the human, computed from the kinematics, display the same "universal" behavior; moreover, these outputs can be described by a remarkably simple class of functions, termed *canonical human walking functions*, with a high degree of accuracy. Utilizing these functions, we consider a 2D bipedal robot with knees, and we construct a control law that drives the outputs of the robot to the outputs of the human. Explicit conditions are derived on the parameters of the canonical human walking functions that guarantee that the zero dynamics surface is partially invariant through impact, i.e., conditions that guarantee *partial hybrid zero dynamics*. These conditions therefore can be used as constraints in an optimization problem that minimizes the distance between the human data and the output of the robot. In addition, we demonstrate through simulation that these conditions automatically generate a stable periodic orbit for which the fixed point can be explicitly computed. Therefore, using only human data, we are able to automatically generate a stable walking gait for a bipedal robot which is as "human-like" as possible.

8.1 Introduction

Obtaining human-like robotic walking has been a long standing, if not always explicitly stated, goal of robotic locomotion. Achieving this goal promises to result in robots able to navigate the myriad of terrains that humans can handle with ease; this would have, for example, important applications to space exploration [4]. Moreover, going beyond purely robotic systems, if one can understand how to make robots

Aaron D. Ames
Department of Mechanical Engineering, Texas A&M University, 3123 TAMU,
College Station, TX 77843-3123, USA
e-mail: aames@tamu.edu

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 89–116.
springerlink.com

walk like humans, this understanding can be used to build robotic assistive and prosthetic devices to aid people with walking impairments and lower extremity amputations walk more efficiently and naturally [29, 8, 22]. Thus, the ability to obtain human-like robotic walking has important and far-reaching ramifications.

The main idea behind this work is that regardless of the complexity present in human walking – hundreds of degrees of freedom coupled with highly nonlinear dynamics and forcing – the essential information needed to understand walking is encoded in simple output functions. That is, we can view the human locomotive system as a "black box" and seek outputs of that system that characterize its behavior in a universally simple fashion or, in other words, attempt to find a low dimensional representation of human walking encoded through outputs described by canonical functions. The goals of this paper are, therefore, two-fold:

1. Determine output functions from human walking data that are accurately described by functions that are canonical, i.e., universal functions of the simplest and most significant form possible.
2. Use these functions to design controllers for a bipedal robot that result in stable walking which is as "close" as possible to human walking.

With the first goal in mind, human data is considered from an experiment where 9 subjects performed straight-line flat-ground walking recorded using motion capture. From the kinematic data associated with this human walking, specific outputs are computed and it is shown that they are accurately described by a very simple class of functions: either a linear function of time, or the time solution to a linear mass-spring-damper system, i.e., a second order linear system response. We therefore achieve the first goal: humans appear to display universally simple behavior when walking, which can be encoded by *canonical human walking functions*. This result provides insight into the basic mechanisms underlying human walking since we conclude that, at the most basic level, the primary outputs associated with locomotion are characterized by a system of linear springs and dampers parameterized in time by the forward walking velocity.

To address the second goal, we consider what we believe to be the simplest bipedal robot that can display "human-like" walking – a 2D robot with knees, modeled as a hybrid system – and we define *human-inspired outputs*: the difference between the outputs of the robot (computed via kinematics) and the human (encoded through canonical walking functions), along with a state-based parameterization of time (achieved through a linear human walking function). Input/output linearization is used to construct an autonomous control law that drives the human-inspired outputs to zero exponentially fast; thus, on the corresponding zero dynamics surface, the outputs of the robot and human are in agreement. We are able to show that, using the human-inspired outputs, we are able to obtain bipedal robotic walking using essentially the same parameters for the human walking functions that were obtained by fitting the data. Yet, due to impacts in the system – which perturb the robot outputs away from the human outputs at foot strike – the resulting walking is reasonably human-like but the velocities are excessively high.

To address the issue of perturbations away from the human outputs at foot strike, we consider the *partial zero dynamics surface* consisting of all of the "relevant" outputs that must be kept invariant through impact, i.e., the relative degree 2 outputs. Conditions are derived – stated only in terms of the parameters of the canonical human walking functions – so that this surface is invariant through impact, i.e., conditions that assure *partial hybrid zero dynamics*. This result is framed as constraints on an optimization problem where the cost function is the sum of residuals squared, resulting in a least squares fit that is as "close" as possible to the human outputs and that ensures invariance of the partial hybrid zero dynamics surface through impact. Even with these constraints, the data for the human outputs is described in a remarkably accurate way by the canonical human walking functions. Moreover, the constraints are constructed in such a way that they automatically produce an initial condition to a stable walking gait for the bipedal robot. We demonstrate this through simulation by obtaining walking for bipedal robots with the mass and length parameters of multiple human subjects. Using only the human walking data, we automatically produce parameters for the human-inspired controller along with a stable walking gait that is as "human-like" as possible.

It is important to note that this is not the first paper that attempts to bridge the gap between human and robotic walking [17, 23, 28, 29], although there have been relatively few studies in this direction when compared to the vast literature on robotic walking and biomechanics (see [9, 10, 11, 33] and [15, 31, 35, 36], to only name a few). Of particular note is [28], which is very much in the same spirit as this paper. In particular, like this paper, that work uses only human parameters and human data to generate human walking, where the least squares fit is used to determine parameters that ensure hybrid zero dynamics. The main difference lies in the output functions and parameterization of time considered. In particular, this paper utilizes canonical human walking functions describing outputs of the human, while [28] considers high degree (9^{th} order) polynomials which are fit directly to the angles of the human over time. This difference is, in the end, a fundamental point of departure. More generally speaking, by looking at outputs of the human that are described by canonical functions intrinsic to walking, the hope is that the fundamental mechanisms underlying human walking can be discovered and exploited to achieve truly human-like bipedal robotic walking.

The structure of this paper is as follows. In Sect. 8.2 we formally introduce hybrid systems with a view toward modeling bipedal robots. The specific robotic model that will be considered throughout this paper is introduced at the end of the section, where the parameters of the robot are obtained from the humans that performed the walking experiment. This experiment is discussed in Sect. 8.3, where the canonical human walking functions are introduced, and it is demonstrated that these functions can accurately fit the human output data. Sect. 8.4 constructs a control law for the bipedal robot based upon the human output functions, and it is shown that this control law can be used to obtain robotic walking – yet the outputs of the robotic walking fail to agree with the human outputs due to the lack of hybrid zero dynamics. In Sect. 8.5 we remedy this problem through the the formulation of an optimization problem that produces the closest fit of the human walking functions

to the human outputs which guarantee partial hybrid zero dynamics. In Sect. 8.6, we show through simulation that the parameters solving the optimization problem automatically produce the initial condition to a stable robotic walking gait. This will be demonstrated on robots with parameters from multiple subjects, in addition to an underactuated robot (with no actuation at the ankle) to demonstrate the extensibility of this approach. Finally, conclusions and future directions are discussed in Sect. 8.7.

8.2 Lagrangian Hybrid Systems and Robotic Model

Hybrid systems are systems that display both continuous and discrete behavior and so bipedal walkers are naturally modeled by systems of this form, with continuous dynamics when the leg is swinging forward and discrete dynamics when the foot strikes the ground resulting in an instantaneous change in velocity. This section, therefore, introduces the basic terminology of hybrid systems, discusses how to build hybrid models of bipedal robots through hybrid Lagrangians, and applies these constructions to the bipedal robot considered in this paper.

Hybrid Systems. We begin by introducing hybrid (control) systems (also referred to as systems with impulsive effects (or systems with impulse effects [13, 14]); for definitions of hybrid systems that consist of more than one domain, we refer the interested reader to [14, 26]. Note that we can consider hybrid systems with one domain because the biped considered will not have feet; if feet are added to the robot, more complex hybrid systems must be considered. For example, all humans appear to display the same universal discrete structure when walking consisting of four discrete domains [7, 30].

Definition 8.1. A *hybrid control system* is a tuple,

$$\mathscr{HC} = (X, U, S, \Delta, f, g),$$

where

- X is the *domain* with $X \subseteq R^n$ a smooth submanifold of the state space $\mathbb{R}^n$,
- $U \subseteq \mathbb{R}^m$ is the set of admissible controls,
- $S \subset X$ is a proper subset of X called the *guard* or *switching surface*,
- $\Delta : S \to X$ is a smooth map called the *reset map*,
- (f, g) is a *control system* on X, i.e., in coordinates: $\dot{x} = f(x) + g(x)u$.

A *hybrid system* is a hybrid control system with $U = \emptyset$, e.g., any applicable feedback controllers have been applied, making the system closed-loop. In this case,

$$\mathscr{H} = (X, S, \Delta, f),$$

where f is a *dynamical system* on $X \subseteq \mathbb{R}^n$, i.e., $\dot{x} = f(x)$.

Periodic Orbits. Due to the discrete behavior present in hybrid systems, solutions to these systems (termed executions) are fundamentally different than solutions to dynamical systems. We will forgo the complexity of introducing executions since in the case of bipedal walking we are only interested in periodic solutions. In particular, in the context of bipedal robots, stable walking gaits correspond to stable periodic orbits. With a view towards periodic orbits in the context of bipedal walking, we will introduce periodic orbits of hybrid systems with fixed points on the guard (for more general definitions, see [14, 32]). Let $\varphi(t, x_0)$ be the solution to $\dot{x} = f(x)$ with initial condition $x_0 \in X$. For $x* \in S$, we say that φ is *periodic* with period $T > 0$ if $\varphi(T, \Delta(x*)) = x*$. A set $\mathcal{O}$ is a *periodic orbit* with *fixed point* x^* if $\mathcal{O} = \{\varphi(t, x^*) : 0 \leq t \leq T\}$ for a periodic solution φ. Associated with a periodic orbit is a Poincaré map [32]. In particular, taking S to be the Poincaré section, one obtains the Poincaré map $P : S \to S$ which is a partial function:

$$P(x) = \varphi(T_I(x), \Delta(x)),$$

where T_I is the *time-to-impact function* [33]. As with smooth dynamical systems, the stability of the Poincaré map determines the stability of the periodic orbit $\mathcal{O}$. In particular, the Poincaré map is (locally) exponentially stable (as a discrete time system $x_{k+1} = P(x_k)$) at the fixed point x^* if and only if the periodic orbit $\mathcal{O}$ is (locally) exponentially stable [19]. Although it is not possible to analytically compute the Poincaré map, it is possible to numerically compute its Jacobian. Thus, if the eigenvalues of the Jacobian have magnitude less than one, the stability of the periodic orbit $\mathcal{O}$ has been numerically verified.

Lagrangian Hybrid Systems. Given a bipedal robot, one naturally has a configuration space, a Lagrangian, and a set of admissible constraints (the swing foot must remain above the ground). This information, inherent to a specific robot, can be encoded formally as a hybrid Lagrangian [5]:

Definition 8.2. A *hybrid Lagrangian* is a tuple:

$$\mathscr{L} = (Q, L, h),$$

where

- Q is the configuration space,
- $L : TQ \to \mathbb{R}$ is a hyperregular Lagrangian,
- $h : Q \to \mathbb{R}$ provides unilateral constraints on the configuration space; it is assumed that $h^{-1}(0)$ is a manifold.

Using a hybrid Lagrangian, we can explicitly construct a hybrid system.

Continuous Dynamics: The Lagrangian of a bipedal robot, $L : TQ \to \mathbb{R}$, can be stated in the form of the kinetic minus potential energy as:

$$L(q, \dot{q}) = \frac{1}{2}\dot{q}^T D(q)\dot{q} - V(q).$$

The Euler-Lagrange equations yield the equations of motion; for robotic systems (see [20]), these are of the form:

$$D(q)\ddot{q} + H(q,\dot{q}) = B(q)u.$$

Converting the equations of motion to a first order ODE yields the affine control system (f,g):

$$f(q,\dot{q}) = \begin{bmatrix} \dot{q} \\ -D^{-1}(q)H(q,\dot{q}) \end{bmatrix}, \qquad g(q) = \begin{bmatrix} \mathbf{0} \\ D^{-1}(q)B(q) \end{bmatrix}. \tag{8.1}$$

Domain and Guard: The domain specifies the allowable configuration of the system as specified by the unilateral constraint function h; for the biped considered in this paper, this function specifies that the non-stance foot must be above the ground. In particular, the domain X is given by:

$$X = \{(q,\dot{q}) \in TQ : h(q) \geq 0\}. \tag{8.2}$$

The guard is just the boundary of the domain with the additional assumption that the unilateral constraint is decreasing:

$$S = \{(q,\dot{q}) \in TQ : h(q) = 0 \text{ and } dh(q)\dot{q} < 0\}, \tag{8.3}$$

where $dh(q)$ is the Jacobian of h at q.

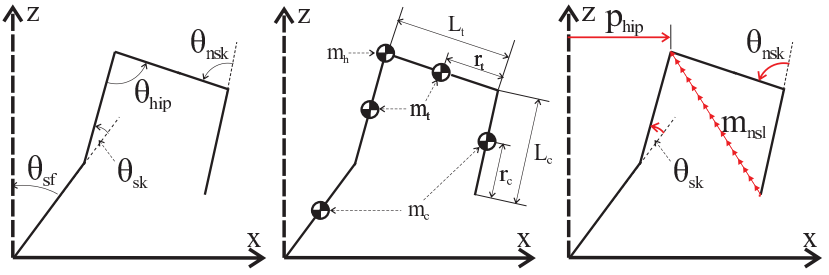
Discrete Dynamics: In order to define the reset map, it is necessary to first augment the configuration space Q. Let p represent the Cartesian position of the stance foot in the x,z plane. The *generalized coordinates* are then written as $q_e = (p_x, p_z, q) \in Q_e = \mathbb{R}^2 \times Q$. Without loss of generality, assume that the values of the extended coordinates are zero throughout the gait, i.e., the stance foot will be fixed at the origin. Therefore, consider the embedding $\iota : Q \to Q_e$ defined as $q \mapsto (0,0,q)$; associated with this embedding is a canonical projection $\pi : Q_e \to Q$.

We employ the impact model described in [16]; that is, plastic rigid-body impacts with impulsive forces are used to simulate impact. In particular, impulsive forces are applied to the swing foot when it contacts the ground. Let $\Upsilon(q_e)$ be the x,z position of the end of the non-stance leg relative to the x,z position of the stance leg (p_x, p_z) and let $J(q_e) = d\Upsilon(q_e)$ be the Jacobian of Υ. The *impact map* gives the post-impact velocity:

$$\dot{q}_e^+ = \mathscr{P}_e(q_e)\dot{q}_e^- = (I - D^{-1}(q_e)J^T(q_e)(J(q_e)D^{-1}(q_e)J^T(q_e))^{-1}J(q_e))\dot{q}_e^- \tag{8.4}$$

with I the identity matrix.

In the bipedal walking literature, under the assumption of symmetric walking, it is common to use a stance/non-stance notation for the legs [13] rather than considering the left and right leg separately; it is more intuitive to think of control design for the legs in the context of stance/non-stance than left/right. To achieve this, the legs must be "swapped" at impact. A coordinate transformation $\mathscr{R}$ (*state relabeling*



(a) configuration space (b) mass/length distribution (c) output functions

Fig. 8.1 Configuration variables (a) and mass/length distribution (b) of the bipedal robot, along with the output functions of the robot considered (c)

procedure) switches the roles of the left and right legs and is included in the reset map:

$$\Delta : S \to X, \qquad \Delta(q,\dot{q}) = \begin{bmatrix} \mathscr{R} & \mathbf{0} \\ \mathbf{0} & \mathscr{R} \end{bmatrix} \begin{bmatrix} q \\ \mathscr{P}(q)\dot{q} \end{bmatrix}, \tag{8.5}$$

where

$$\mathscr{P}(q) = d\pi(\iota(q))\mathscr{P}_e(\iota(q))d\iota(q)$$

with $d\pi$ the Jacobian of the projection π and $d\iota$ the Jacobian of the embedding ι.

Bipedal Robot Model. In this paper, we consider a 2-dimensional bipedal robot with knees as illustrated in Fig. 8.1. The motivation for considering this specific model is that it is simple enough to make the discussion of ideas related to human-inspired control more concise, while being complex enough to display interesting behavior. Using the previous constructions of this section, we explicitly construct a hybrid model.

Hybrid Lagrangian: The 2D kneed biped with knees has four links, yielding a configuration space Q_R with coordinates: $\theta = (\theta_{sf}, \theta_{sk}, \theta_{hip}, \theta_{nsk})^T$, where, as illustrated in Fig. 8.1(a), θ_{sf} is the angle of the stance foot, θ_{sk} is the angle of the stance knee, θ_{hip} is the angle of the hip and θ_{nsk} is the angle of the non-stance (or swing) knee. Combining this choice of coordinates with the mass and length distribution illustrated in Fig. 8.1(b) (with parameters chosen from the table in Fig. 8.2(b)) yields a Lagrangian L_R which depends on the parameters (lengths and masses) of the specific human subject being considered (as discussed in Sect. 8.3). Finally, the unilateral constraint is simply the height of the non-stance foot above the ground: h_R (which again depends on the parameters of the specific subject being considered). The end result is that, given a bipedal robot model with parameters obtained from a specific subject, we obtain a hybrid Lagrangian:

$$\mathscr{L}_R = (Q_R, L_R, h_R). \tag{8.6}$$

Hybrid System Model: From this hybrid Lagrangian we obtain a hybrid system model for the robot with the parameters from a specific subject:

$$\mathscr{HC}_R = (X_R, U_R, S_R, \Delta_R, f_R, g_R), \tag{8.7}$$

where here the domain and guard, X_R and S_R, are given by (8.2) and (8.3), respectively, using the unilateral constraint function h_R, $U_R = \mathbb{R}^4$ since we do not put any restrictions on the set of admissible controls and assume full actuation, and Δ_R is given by (8.5) with relabeling matrix:

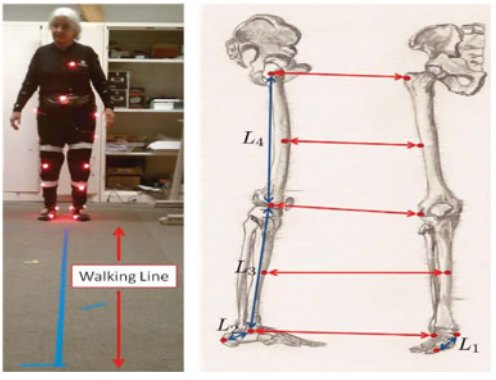
$$\mathscr{R} = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \tag{8.8}$$

which switches the stance and non-stance leg at impact. Note that we assume full actuation because during the course of a human step, most of the time is spent in domains where there is full actuation [7]. Moreover, methods for extending control laws assuming full actuation to domains with underactuation have been discussed in [14, 25, 26]. Also note that the methods discussed in this section can be used to "automatically" generate a wide variety of bipedal robotic models (see [24] for 3 other single-domain models and [7] for the multi-domain case). Finally, to further justify this point, we consider the case of underactuation at the end of Sect. 8.6 and show that we can also obtain walking when the stance ankle, θ_{sf}, is not actuated.

8.3 Canonical Human Walking Functions

This section begins by discussing a human walking experiment. We use the data from this experiment to motivate the introduction of canonical human walking functions. Amazingly, these functions are very simple; they describe the solution to a linear spring-damper system. The use of these functions is justified by fitting them to certain outputs of the human computed from the experimental data, resulting in very high correlation fits. We conclude, therefore, that humans display very simple behavior when the proper outputs are considered.

Human Walking Experiment. Data was collected on 9 subjects using the Phase Space System, which computes the 3D position of 19 LED sensors at 480 frames per second using 12 cameras at 1 millimeter level of accuracy. The cameras were calibrated prior to the experiment and were placed to achieve a 1 millimeter level of accuracy for a space of size 5 by 5 by 5 meters cubed. Six sensors were placed on each leg at the joints; a sensor was placed on each leg at the heel and another at the toe; finally, one sensor was placed at each of the following locations: the sternum, the back behind the sternum and the belly button (as illustrated in Fig. 8.2). Each trial of the experiment required the subject to walk 3 meters along a line drawn on the floor. Each subject performed 12 trials, which constituted a single experiment. 3 female and 6 male subjects with ages ranging between 17 and 77 years, heights



(a) experimental setup

Subject	S_1	S_2	S_3
Sex	M	M	F
Age	22	30	23
Ht. (cm)	169.5	170.0	165.5
Wt. (kg)	90.9	69.1	47.7
m_h (kg)	61.5	46.7	32.3
m_t (kg)	9.1	6.9	4.8
m_c (kg)	5.5	3.3	2.9
L_t (cm)	40.1	43.8	39.2
L_c (cm)	43.5	38.1	37.6

(b) subject parameters

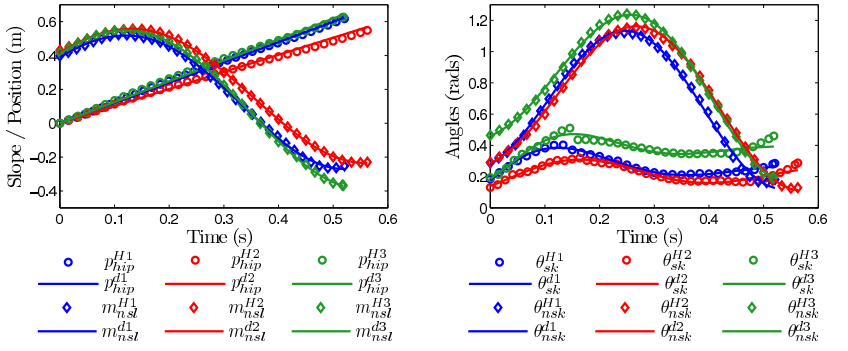
Fig. 8.2 (a) Illustrations of the experimental setup and sensor placement. Each LED sensor was placed at the joints as illustrated with the dots on the right lateral and anterior aspects of the each leg. (b) The physical parameters of the 3 subjects considered in this paper, together with the mapping of those parameters onto the robot being considered (see Fig. 8.1(b))

ranging between 161 and 189 centimeters, and weights ranging between 47.6 and 90.7 kilograms. The data for each individual is rotated so that the walking occurs in the x-direction, and for each subject the 12 walking trials are averaged (after appropriately shifting the data in time). The collected data is available at [2].

In this paper, for the sake of simplicity of exposition, we will consider the data for 3 of these 9 subjects. The parameters for these 3 subjects as they are used on the bipedal robot can be found in the table in Fig. 8.2(b). To map the human parameters that were experimentally determined to the robotic model being considered, the standard mass distribution formula from [35] was used.

Human Outputs and Walking Functions. The fundamental idea behind obtaining robotic walking from human walking data is that, rather than looking at the dynamics of the human, one should look at outputs of the human that represent the walking behavior. By tracking these outputs in a robot, through their representation via canonical functions, the robot will display the same qualitative behavior as the human despite the differences in dynamics. That is, we seek to find a "low-dimensional" representation of human walking. To find this representation, we look for "simple" functions of the kinematics of the human that seem to be representative of walking, termed *canonical human walking functions*.

The specific choice of human outputs to consider depend on the bipedal robot. In particular, and roughly speaking, for each degree of freedom of the robot there must be an associated output. In addition, these outputs must not "compete" with each other, i.e., they must be mutually exclusive; formally speaking, this means that the decoupling matrix associated with these outputs must be non-singular (as will be discussed in Sect. 8.6). Motivated by intuition, and after the consideration of a large



(a) hip position and non-stance leg slope (b) stance and non-stance knee angle, θ_{nsk}

Fig. 8.3 The human output data and the canonical walking function fits for each subject

number of human outputs, the four we have found to be most essential to walking in the case of the robot being considered are:

1. The x-position of the hip, p_{hip},
2. The slope of the non-stance leg, i.e., the tangent of the angle between the z-axis and the line on the non-stance leg connecting the ankle and hip:

$$m_{nsl} = \frac{p^x_{nsf} - p_{\text{hip}}}{p^z_{nsf} - p^z_{\text{hip}}},$$

   where p^z_{hip} is the z-position of the hip and p^x_{nsf}, p^z_{nsf} are the x and z position of the non-stance foot, respectively,
3. The angle of the stance knee, θ_{sk},
4. The angle of the non-stance knee, θ_{nsk} .

These outputs can be seen in Fig. 8.1(c).

Visually inspecting the outputs as computed from the human data for the three subjects, as shown in Fig. 8.3, all of the human outputs appear to be described by two simple functions:

$$\begin{aligned} p_{\text{hip}}: &\quad y_{H,1}(t,v) = vt, \\ m_{nsl},\, \theta_{sk},\, \theta_{nsk}: &\quad y_{H,2}(t,\alpha) = e^{-\alpha_4 t}(\alpha_1 \cos(\alpha_2 t) + \alpha_3 \sin(\alpha_2 t)) + \alpha_5, \end{aligned} \tag{8.9}$$

where $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5)$. Moreover, these functions appear to be *universal* to walking, in that all of the subjects considered in this experiment followed functions of this form for the chosen outputs. Of special interest is the fact that $y_{H,1}$ simply parameterizes time by the forward velocity (walking speed) of the human and $y_{H,2}$ is simply the solution to a linear mass-spring-damper system with constant external forcing [12]. Thus, despite the complexity of the internal dynamics inherent to walking, humans appear to act like a linear spring damper system parameterized by the walking speed.

It is important to note that this universality continues beyond the robotic model considered in this paper. In fact, we have found that this pattern continues for robots with feet [25] and for 3-dimensional robots [24], i.e., the additional human outputs that must be considered for these robots due to the additional degrees of freedom can be described by $y_{H,2}$ with a high degree of accuracy (a high correlation coefficient).

Data-Based Cost Function. Having obtained what we believe to be outputs characterized by functions that appear to be canonical to human walking, it is necessary to determine the specific parameters of these functions that best fit the human data. While this can be done with a myriad of software programs, we describe the procedure to fit these functions in terms of minimizing a cost function; the reason for explicitly stating the cost function is that this same cost function is used when considering hybrid zero dynamics, except that it is subject to constraints.

As a result of the universality of the human walking functions (8.9), we can consider these functions for each of the subjects and each of the outputs specified above: p_{hip}, m_{nsl}, θ_{sk}, and θ_{nsk}. This yields the following four functions that we wish to fit to the data for each subject:

$$\begin{aligned} p^d_{\text{hip}}(t, v_{\text{hip}}) &= y_{H,1}(t, v_{\text{hip}}), & m^d_{nsl}(t, \alpha_{nsl}) &= y_{H,2}(t, \alpha_{nsl}), \\ \theta^d_{sk}(t, \alpha_{sk}) &= y_{H,2}(t, \alpha_{sk}), & \theta^d_{nsk}(t, \alpha_{nsk}) &= y_{H,2}(t, \alpha_{nsk}), \end{aligned} \tag{8.10}$$

where, for example, $\alpha_{nsl} = (\alpha_{nsl,1}, \alpha_{nsl,2}, \alpha_{nsl,3}, \alpha_{nsl,4}, \alpha_{nsl,5})$ in (8.9). The parameters for the four output functions can be combined to yield a single vector of parameters: $\alpha = (v_{\text{hip}}, \alpha_{nsl}, \alpha_{nsk}, \alpha_{sk}) \in \mathbb{R}^{16}$ for each subject. These output functions can be directly compared to the corresponding human data. In particular, from the human walking experiment, we obtain discrete times, $t^H[k]$, and discrete values for the output functions: $p^H_{\text{hip}}[k]$, $m^H_{nsl}[k]$, $\theta^H_{sk}[k]$ and $\theta^H_{nsk}[k]$, for $k \in \{1, \dots, K\} \subset \mathbb{N}$ with K the number of data points.

Consider the following human-data-based cost function:

$$\begin{aligned} &\text{Cost}_{\text{HD}}(\alpha) = \\ &\sum_{k=1}^{K} \Big(\beta_{p_{\text{hip}}} (p^d_{\text{hip}}(t^H[k], v_{\text{hip}}) - p^H_{\text{hip}}[k])^2 + \beta_{m_{nsl}} (m^d_{nsl}(t^H[k], \alpha_{nsl}) - m^H_{nsl}[k])^2 \\ &\qquad + \beta_{\theta_{sk}} (\theta^d_{sk}(t^H[k], \alpha_{sk}) - \theta^H_{sk}[k])^2 + \beta_{\theta_{nsk}} (\theta^d_{nsk}(t^H[k], \alpha_{nsk}) - \theta^H_{nsk}[k])^2 \Big), \end{aligned} \tag{8.11}$$

where each of the weightings, β, are the reciprocal of the difference of the maximum and minimum value of the human data for that output. Therefore, this cost function is simply the weighted sum of the squared residuals. Clearly, this cost function depends on the output data for a specific subject over the course of one step, together with the choice of walking functions (8.10), but it is viewed only as a function of the vector of parameters $\alpha = (v_{\text{hip}}, \alpha_{nsl}, \alpha_{sk}, \alpha_{nsk})$. To determine this vector of parameters, we need only solve the following optimization problem:

$$\alpha^* = \underset{\alpha \in \mathbb{R}^{16}}{\text{argmin}} \, \text{Cost}_{\text{HD}}(\alpha) \tag{8.12}$$

Table 8.1 Table containing parameter values of the canonical human walking functions for the 3 subjects together with the cost and correlations of the fits

$y_{H,1} = vt$ $y_{H,2} = e^{-\alpha_4 t}(\alpha_1 \cos(\alpha_2 t) + \alpha_3 \sin(\alpha_2 t)) + \alpha_5$									
S	Fun.	v	α_1	α_2	α_3	α_4	α_5	Cost	Cor.
1	p_{hip}	1.1969	*	*	*	*	*	0.017	0.9995
	m_{nsl}	*	0.2021	7.8404	0.2248	-0.8070	0.1871		0.9998
	θ_{sk}	*	-0.0828	13.316	0.2073	4.1551	0.2574		0.9933
	θ_{nsk}	*	-0.3809	-10.979	-0.1966	-0.4215	0.6585		0.9990
2	p_{hip}	1.0104	*	*	*	*	*	0.029	0.9992
	m_{nsl}	*	0.2040	7.3406	0.2417	-0.6368	0.2147		0.9999
	θ_{sk}	*	-0.0800	13.379	0.0865	1.6614	0.2198		0.9807
	θ_{nsk}	*	-0.3914	-10.516	-0.1562	-0.5243	0.6789		0.9995
3	p_{hip}	1.2372	*	*	*	*	*	0.057	0.9990
	m_{nsl}	*	0.2618	7.2804	0.2615	-0.6769	0.1490		1.0000
	θ_{sk}	*	-0.1939	16.152	0.0745	4.9945	0.3801		0.9561
	θ_{nsk}	*	-0.3190	10.903	0.1539	-1.0190	0.7790		0.9995

which yields the least squares fit of the data with the canonical walking functions; again, the reason for restating this standard definition is that this same cost function will be used in Sect. 8.5, but subject to specific constraints that ensure hybrid zero dynamics. The parameters given by solving this optimization problem are stated in Table 8.1 along with the cost (8.11) associated with these parameters. The correlations, as given in the same table, show that the fitted walking functions very closely model the human output data, i.e., the chosen human walking functions appear to be, in fact, canonical. Indeed, the coefficients of correlation are all between 0.9561 and 1.000. The accuracy of the fits can be seen in Fig. 8.3.

8.4 Human-Inspired Control

In this section, we construct a human-inspired controller that drives the outputs of the robot to the outputs of the human (as represented by canonical walking functions). Moreover, we are able to make this control law autonomous through a parameterization of time based upon the position of the hip. The end result is a feedback control that yields stable walking when applied to the bipedal robot being considered. The aspects of this walking are discussed – specifically, the fact that the outputs are not left invariant through impact – in order to motivate the introduction of hybrid zero dynamics in the next section.

Output Functions. Based upon the canonical human walking functions, we define the following relative degree one and two outputs for the bipedal robot being considered (see Sect. 8.2 for the robot and [21] for the definition of relative degree):

Relative Degree 1: Recall that the forward position of the hip of the human is described by: $p^d_{\text{hip}}(t, v_{\text{hip}}) = v_{\text{hip}} t$. Therefore, the forward velocity of a human when

walking is approximately constant, and we wish to define an output that will similarly keep the forward velocity of the robot constant. Letting $p^R_{\text{hip}}(\theta)$ be the x-position of the robot (which is computed through forward kinematics), the goal is to drive: $\dot{p}^R_{\text{hip}} \to v_{\text{hip}}$. With this in mind, we define the following actual and desired outputs:

$$y_{a,1}(\theta,\dot{\theta}) = dp^R_{\text{hip}}(\theta)\dot{\theta}, \qquad y_{d,1} = v_{\text{hip}}. \tag{8.13}$$

Note that $y_{a,1}$ will be a relative degree 1 output since it is the output of a mechanical system depending both on position and velocity.

Relative Degree 2: We now consider the remainder of the outputs given in (8.10); in particular, we consider the non-stance leg slope, m_{nsl}, the stance knee angle, θ_{sk} and the non-stance knee angle, θ_{nsk}. The stance slope of the robot can be stated in terms of the angles of the system through the use of kinematics, i.e., we obtain an expression $m^R_{nsl}(\theta)$. Since the goal is for the robot to track the human outputs, we consider the following actual and desired outputs:

$$y_{a,2}(\theta) = \begin{bmatrix} m^R_{nsl}(\theta) \\ \theta_{sk} \\ \theta_{nsk} \end{bmatrix}, \qquad y_{d,2}(t) = \begin{bmatrix} m^d_{nsl}(t,\alpha_{nsl}) \\ \theta^d_{sk}(t,\alpha_{sk}) \\ \theta^d_{nsk}(t,\alpha_{nsk}) \end{bmatrix}. \tag{8.14}$$

Since the actual output is only a function of the configuration variables for a mechanical system, it will be relative degree 2.

Parameterization of Time. The goal is clearly to drive $y_{a,1} \to y_{d,1}$ and $y_{a,2} \to y_{d,2}$. This could be done through standard tracking control laws [21], but the end result would be a time-based, or non-autonomous, control law. This motivates the introduction of a parameterization of time in order to obtain an autonomous control law. This procedure is common in the literature [33, 34], and the parameterization chosen draws inspiration from both those that have been used in the past in the context of bipedal walking and the human data. Using the fact that the forward position of the hip of the human is described by $p^d_{\text{hip}}(t, v_{\text{hip}}) = v_{\text{hip}}t$, for the human: $t \approx \frac{p_{\text{hip}}}{v_{\text{hip}}}$. This motivates the following parameterization of time for the robot:

$$\tau(\theta) = \frac{p^R_{\text{hip}}(\theta) - p^R_{\text{hip}}(\theta^+)}{v_{\text{hip}}}, \tag{8.15}$$

where here $p^R_{\text{hip}}(\theta^+)$ is the position of the hip of the robot at the beginning of a step[1] with θ^+ assumed to be a point where the height of the non-stance foot is zero, i.e., $h_R(\theta^+) = 0$, with h_R the unilateral constraint for the hybrid Lagrangian $\mathscr{L}_R$ associated with the bipedal robot (8.6).

[1] Note that we can assume that the initial position of the human is zero, while this cannot be assumed for the robot since the initial position of the hip will depend on the specific choice of configuration variables for the robot.

Control Law Construction. Using the parameterization of time, we define the following output functions, termed *human-inspired outputs*:

$$\begin{aligned} y_1(\theta,\dot{\theta}) &= y_{a,1}(\theta,\dot{\theta}) - y_{d,1},\\ y_2(\theta) &= y_{a,2}(\theta) - y_{d,2}(\tau(\theta)), \end{aligned} \tag{8.16}$$

which depend on the specific choice of parameters, α, for the human walking functions, where y_1 is an output with relative degree 1 and y_2 is a vector of outputs all with relative degree 2. For the affine control system (f_R, g_R) associated with the robotic model being considered (8.7), we define the following control law:

$$u(\theta,\dot{\theta}) = -A^{-1}(\theta,\dot{\theta})\left(\begin{bmatrix} 0 \\ L_{f_R}^2 y_2(\theta,\dot{\theta}) \end{bmatrix} + \begin{bmatrix} L_{f_R} y_1(\theta,\dot{\theta}) \\ 2\varepsilon L_{f_R} y_2(\theta,\dot{\theta}) \end{bmatrix} + \begin{bmatrix} 2\varepsilon y_1(\theta,\dot{\theta}) \\ \varepsilon^2 y_2(\theta) \end{bmatrix}\right), \tag{8.17}$$

with L the Lie derivative and A the decoupling matrix:

$$A(\theta,\dot{\theta}) = \begin{bmatrix} L_{g_R} y_1(\theta,\dot{\theta}) \\ L_{g_R} L_{f_R} y_2(\theta,\dot{\theta}) \end{bmatrix}. \tag{8.18}$$

Note that the decoupling matrix is non-singular exactly because of the choice of output functions, i.e., as was discussed in Sect. 8.3, care was taken when defining the human outputs so that they were "mutually exclusive." It follows that for a control gain $\varepsilon > 0$, the control law u renders the output exponentially stable [21]. That is, the human-inspired outputs $y_1 \to 0$ and $y_2 \to 0$ exponentially at a rate of ε; in other words, the outputs of the robot will converge to the canonical human walking functions exponentially. In addition, since $y_1 \to 0$, it follows that during the continuous evolution of the system $y_{a,1} = \dot{p}^R_{\text{hip}} \to v_{\text{hip}}$, i.e., the velocity of the hip will converge to the velocity of the human, thus justifying the parameterization of time given in (8.15).

Applying the feedback control law in (8.17) to the hybrid control system modeling the bipedal robot being considered, $\mathscr{HC}_R$ as given in (8.7), yields a hybrid system:

$$\mathscr{H}_R^{(\alpha,\varepsilon)} = (X_R, S_R, \Delta_R, f_R^{(\alpha,\varepsilon)}), \tag{8.19}$$

where, X_R, S_R, and Δ_R are defined as for $\mathscr{HC}_R$, and

$$f_R^{(\alpha,\varepsilon)}(\theta,\dot{\theta}) = f_R(\theta,\dot{\theta}) + g_R(\theta,\dot{\theta})u(\theta,\dot{\theta}),$$

where the dependence of $f_R^{(\alpha,\varepsilon)}$ on the vector of parameters, α, and the control gain for the input/output linearization control law, ε, has been made explicit.

Partial Hybrid Zero Dynamics. In addition to driving the chosen outputs to zero, i.e., driving the outputs of the robot to the outputs of the human, the control law (8.17) also renders the *(full) zero dynamics surface*:

$$\mathbf{Z}_\alpha = \{(\theta,\dot{\theta}) \in TQ : y_1(\theta,\dot{\theta}) = 0,\ y_2(\theta) = \mathbf{0},\ L_{f_R} y_2(\theta,\dot{\theta}) = \mathbf{0}\} \tag{8.20}$$

invariant for the continuous dynamics; here, $\mathbf{0} \in \mathbb{R}^3$ is a vector of zeros and we make the dependence of $\mathbf{Z}_\alpha$ on the set of parameters explicit. This fact is very important for continuous control systems since once the outputs converge to zero, they will stay at zero. The difficulty of ensuring these conditions in the case of hybrid systems will be discussed in Sec. 8.5. In particular, in the case of the robot considered in this paper, we will seek only to enforce the zero dynamics surface related to the relative degree 2 outputs. We refer to this as the *partial zero dynamics surface*, given by:

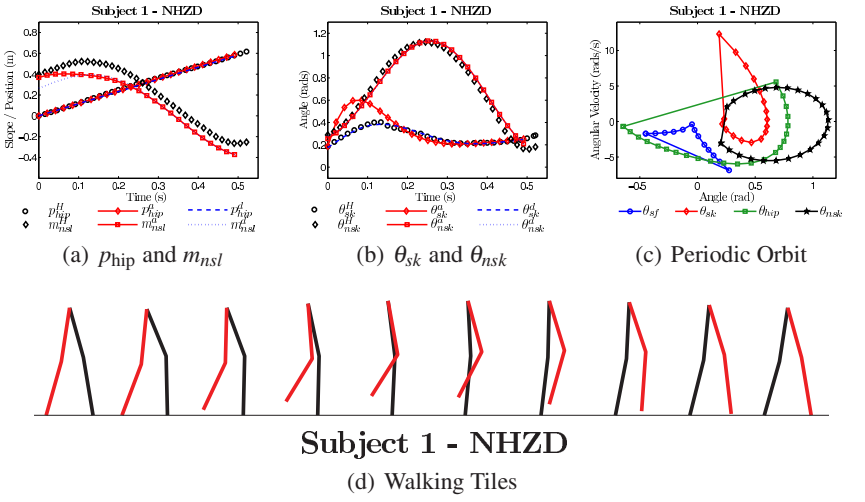
$$\mathbf{PZ}_\alpha = \{(\theta,\dot{\theta}) \in TQ : y_2(\theta) = \mathbf{0},\ L_{f_R} y_2(\theta,\dot{\theta}) = \mathbf{0}\} \tag{8.21}$$

The motivation for considering this surface is that it allows some "freedom" in the movement of the system to account for differences between the robot and human models. Moreover, since the only output that is not included in the partial zero dynamics surface is the output that forces the forward hip velocity to be constant, enforcing partial hybrid zero dynamics simply means that we allow the velocity of the hip to compensate for the shocks in the system due to impact.

As a final note, the partial zero dynamics surface is simply the zero dynamics surface considered in [33]; that is, in that reference and all of the supporting papers, underactuation at the stance ankle is considered (this case will be addressed, in the context of human-inspired control, in Sect. 8.6). It is exactly the stance ankle that we control via the output y_1. Thus, in essence, the partial zero dynamics is just the classical zero dynamics surface considered in the bipedal walking literature. Therefore, if parameters can be determined that render this surface invariant through impact, the system will evolve on a 2-dimensional zero dynamics manifold. The main differences separating this work from existing work is that the shape of the zero dynamics surface – as determined by the human outputs and the parameters of the human walking function – is chosen explicitly from human data.

Robotic Walking without Hybrid Zero Dynamics. To demonstrate that it is possible to obtain human walking with the human-inspired outputs, we manually looked for periodic orbits for the hybrid system $\mathscr{H}_R^{(\alpha,\varepsilon)}$, with α a vector of parameters as "close" as possible to the vector of parameters α^* solving the optimization problem (8.12). In particular, we considered Subject 1 and his corresponding mass and length parameters (Fig. 8.2(b)). We found that by only changing α_5 for m_{nsl} in Table 8.1 from 0.1871 to 0.0623, and picking $\varepsilon = 20$, we were able to obtain a stable periodic orbit, i.e., a stable walking gait, as seen in Fig. 8.4. The stability of the periodic orbit was checked by numerically computing the eigenvalues of the Poincaré map; the magnitude of these eigenvalues and the corresponding fixed point can be found in Fig. 8.7. Finally, a movie of the walking can be found at [1].

The robotic walking shown in Fig. 8.4 allows one to draw some important conclusions. Despite the fact that the parameters α used to obtain the walking are almost identical to the parameters obtained by fitting the canonical output functions to the human data, as seen in Figs. 8.4(a)-(b), the outputs of the robot are dramatically different than those seen in the human data. This is a direct result of the fact that the



(a) p_{hip} and m_{nsl} (b) θ_{sk} and θ_{nsk} (c) Periodic Orbit

(d) Walking Tiles

Fig. 8.4 Walking obtained for Subject 1 without hybrid zero dynamics, with (a)-(b) the actual and desired outputs vs. the human output data over one step, (c) the periodic orbit and (d) the tiles (or snapshots) of the walking gait

zero dynamics surface $\mathbf{Z}_\alpha$ is not invariant through impact, i.e., there are not hybrid zero dynamics. Therefore, at impact, the robot is thrown off the zero dynamics surface, resulting in the parameterization of time, τ, becoming highly nonlinear. Since the other desired functions depend on this parameterization, this changes the shape of these functions in a nonlinear fashion and thus the shape of the zero dynamics surface. Moreover, the fact that the control law tries to drive the outputs to zero means that after impact a spike in both velocity and torque occurs (the angular velocity after impact exceeds $12\frac{rad}{s}$ as seen in Fig. 8.4(c)). The end result is that while this control yields stable walking, it is unrealistic in a physical context. In order to solve this issue, it is clearly necessary to determine conditions that guarantee that the zero dynamics are invariant through impact.

8.5 Partial Hybrid Zero Dynamics for Human-Inspired Outputs

The goal of this section is to find parameters that result in *partial hybrid zero dynamics* (PHZD) while best fitting the human walking data. The main result is that we are able to formulate an optimization problem, *only* depending on the parameters α, that guarantees PHZD. We demonstrate this optimization problem on the human subject data, showing that we are able to simultaneously achieve PHZD and very good fits of the human data with the canonical human walking functions.

Problem Statement: PHZD. The goal of human-inspired PHZD is to find parameters α^* that solve the following constrained optimization problem:

$$\alpha^* = \underset{\alpha \in \mathbb{R}^{16}}{\operatorname{argmin}} \ \text{Cost}_{\text{HD}}(\alpha) \tag{8.22}$$

$$\text{s.t} \quad \Delta_R(S_R \cap \mathbf{Z}_\alpha) \subset \mathbf{PZ}_\alpha \tag{PHZD}$$

with Cost_{HD} the cost given in (8.11), $\mathbf{Z}_\alpha$ the full zero dynamics surface given in (8.20) and $\mathbf{PZ}_\alpha$ the partial zero dynamics surface given in (8.21). This is simply the optimization problem in (8.12) that was used to determine the parameters of the canonical human walking functions that gave the best fit of the human output data, but subject to constraints that ensure PHZD.

The formal goal of this section is to restate (PHZD) in such a way that it can be practically solved; as (8.22) is currently stated, it depends on both the parameters, α, as well as the state of the robot, $(\theta, \dot{\theta})$, due to the inclusion of the (full and partial) zero dynamics surfaces $\mathbf{Z}_\alpha$ and $\mathbf{PZ}_\alpha$ in the constraints.

Position and Velocity from Parameters. To acheive the goal of restating (8.22) in a way that is independent of state variables (position and velocity), we can use the outputs and guard functions to explicitly solve for the configuration of the system $\vartheta(\alpha) \in Q_R$ on the guard in terms of the parameters. In particular, let

$$\vartheta(\alpha) = \theta \quad \text{s.t} \quad \begin{bmatrix} y_2(\mathscr{R}\theta) \\ h_R(\theta) \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}, \tag{8.23}$$

where $\mathscr{R}$ is the relabeling matrix (8.8). Note that $\vartheta(\alpha)$ exists because of the specific structure of the outputs, $y_2(\mathscr{R}\theta)$, chosen. In fact, the reason for considering y_2 at the point $\mathscr{R}\theta$ is because this implies that the configuration at the beginning of the step is $\theta^+ = \mathscr{R}\theta$ and thus $\tau(\mathscr{R}\theta) = 0$ implying that:

$$y_2(\mathscr{R}\theta) = y_{a,2}(\mathscr{R}\theta) - y_{d,2}(0),$$

or there is a solution to (8.23) because of the simple form that y_2 takes at the point $\mathscr{R}\theta$. Also note that since solving for $\vartheta(\alpha)$ is essentially an inverse kinematics problem, the solution obtained exists, but is not necessarily unique. Therefore, a solution must be picked that is "reasonable," i.e., θ_{sf} and θ_{hip} are in $[-\frac{\pi}{2}, 0]$, and that produces the first time when the foot hits the ground, $\tau(\vartheta(\alpha))$.

Using $\vartheta(\alpha)$, we can explicitly solve for velocities $\dot{\vartheta}(\alpha)$. The methodology is to solve for the velocities that are in the full zero dynamics surface $\mathbf{Z}_\alpha \subset \mathbf{PZ}_\alpha$ pre-impact as motivated by the fact that solutions will converge to this surface exponentially during the walking gait; thus, the velocities pre-impact will be sufficiently close to this point even if we only enforce partial hybrid zero dynamics. In particular, let

$$Y(\theta) = \begin{bmatrix} dp^R_{\text{hip}}(\theta) \\ dy_2(\theta) \end{bmatrix}, \tag{8.24}$$

with $dp_{\text{hip}}(\theta)$ and $dy_2(\theta)$ the Jacobian of p_{hip} and y_2, respectively. It follows from (8.13), (8.14), (8.16), and the fact that y_2 is a relative degree 2 output that

$$\begin{bmatrix} y_1(\theta,\dot{\theta}) \\ L_{f_R} y_2(\theta,\dot{\theta}) \end{bmatrix} = Y(\theta)\dot{\theta} - \begin{bmatrix} v_{\text{hip}} \\ \mathbf{0} \end{bmatrix}. \tag{8.25}$$

Therefore, define

$$\dot{\vartheta}(\alpha) = Y^{-1}(\vartheta(\alpha)) \begin{bmatrix} v_{\text{hip}} \\ \mathbf{0} \end{bmatrix}. \tag{8.26}$$

Note that Y is invertible, again because of the specific choice of outputs, i.e., because of the relative degree of the outputs y_1 and y_2.

PHZD Optimization. We now present the first main result of this paper. Using $\vartheta(\alpha)$ and $\dot{\vartheta}(\alpha)$, we can restate the optimization problem (8.22) in terms of only parameters of the system. Moreover, as will be seen Sect. 8.6, by solving the restated optimization problem, we automatically obtain an initial condition corresponding to stable periodic walking.

Theorem 8.1. *The parameters α^* solving the constrained optimization problem:*

$$\alpha^* = \underset{\alpha\in\mathbb{R}^{16}}{\text{argmin}}\ \text{Cost}_{\text{HD}}(\alpha) \tag{8.27}$$

$$\begin{aligned} \text{s.t}\quad & y_2(\vartheta(\alpha)) = \mathbf{0} && \text{(C1)} \\ & dy_2(\mathscr{R}\vartheta(\alpha))\mathscr{R}\mathscr{P}(\mathscr{R}\vartheta(\alpha))\dot{\vartheta}(\alpha) = \mathbf{0} && \text{(C2)} \\ & dh_R(\vartheta(\alpha))\dot{\vartheta}(\alpha) < 0 && \text{(C3)} \end{aligned}$$

yield partial hybrid zero dynamics: $\Delta_R(S_R \cap \mathbf{Z}_{\alpha^*}) \subset \mathbf{PZ}_{\alpha^*}$.

Proof. Let α^* be the solution to the optimization problem (8.27). By (C1) and (8.25), $(\vartheta(\alpha^*), \dot{\vartheta}(\alpha^*)) \in \mathbf{Z}_{\alpha^*}$. Moreover, by (8.23) (specifically, the fact that $h_R(\vartheta(\alpha^*)) = 0$) and (C3), it follows that $(\vartheta(\alpha^*), \dot{\vartheta}(\alpha^*)) \in S_R$. Therefore, $(\vartheta(\alpha^*), \dot{\vartheta}(\alpha^*)) \in S_R \cap \mathbf{Z}_{\alpha^*}$. Now, $\mathbf{Z}_{\alpha^*}$ and S_R intersect transversally since

$$L_{f_R} h_R(\vartheta(\alpha^*), \dot{\vartheta}(\alpha^*)) = dh_R(\vartheta(\alpha^*))\dot{\vartheta}(\alpha^*) < 0$$

by (C3). Since $\mathbf{Z}_{\alpha^*}$ is a 1-dimensional submanifold of X_R, it follows that $S_R \cap \mathbf{Z}_{\alpha^*}$ is a unique point. Therefore, $(\vartheta(\alpha^*), \dot{\vartheta}(\alpha^*)) = S_R \cap \mathbf{Z}_{\alpha^*}$.

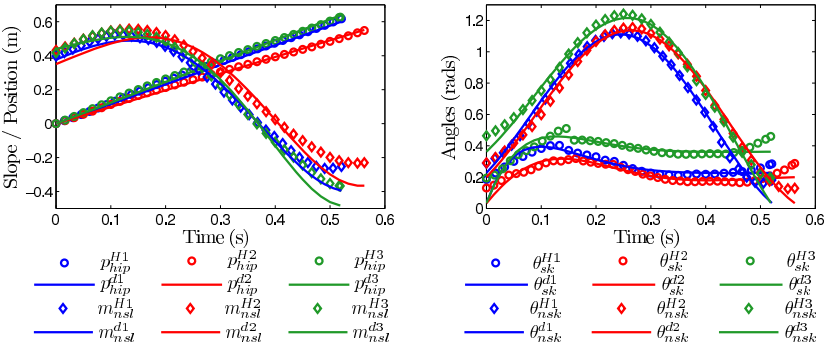
To show that $\Delta_R(S_R \cap \mathbf{Z}_{\alpha^*}) \subset \mathbf{PZ}_{\alpha^*}$ we need only show that $\Delta_R(\vartheta(\alpha^*), \dot{\vartheta}(\alpha^*)) \in \mathbf{PZ}_{\alpha^*}$. Since

$$\Delta_R(\vartheta(\alpha^*), \dot{\vartheta}(\alpha^*)) = \begin{bmatrix} \mathscr{R}\vartheta(\alpha^*) \\ \mathscr{R}\mathscr{P}(\vartheta(\alpha^*))\dot{\vartheta}(\alpha^*) \end{bmatrix}$$

from (8.5), the requirement that $\Delta_R(\vartheta(\alpha^*), \dot{\vartheta}(\alpha^*)) \in \mathbf{PZ}_{\alpha^*}$ is equivalent to the following conditions being satisfied:

$$y_2(\mathscr{R}\vartheta(\alpha^*)) = 0, \tag{8.28}$$

$$L_{f_R} y_2(\mathscr{R}\vartheta(\alpha^*), \mathscr{R}\mathscr{P}(\vartheta(\alpha^*))\dot{\vartheta}(\alpha^*)) = 0. \tag{8.29}$$



(a) hip position and non-stance leg slope (b) stance and non-stance knee angle

Fig. 8.5 The human output data and the canonical walking function fits for each subject obtained by solving the optimization problem in Theorem 8.1

By the definition of $\vartheta(\alpha^*)$, and specifically (8.23), (8.28) is satisfied. Moreover,

$$L_{f_R} y_2 \mathscr{R}\vartheta(\alpha^*), \mathscr{R}\mathscr{P}(\vartheta(\alpha^*))\dot{\vartheta}(\alpha^*) = dy_2(\mathscr{R}\vartheta(\alpha^*))\mathscr{R}\mathscr{P}(\vartheta(\alpha^*))\dot{\vartheta}(\alpha^*)$$

Therefore, (8.29) is satisfied as a result of (C2). Thus we have established that $\Delta_R(S_R \cap \mathbf{Z}_{\alpha^*}) \subset \mathbf{PZ}_{\alpha^*}$. □

Remark on Theorem 8.1. Note that if the goal was to obtain full hybrid zero dynamics: $\Delta_R(S_R \cap \mathbf{Z}_\alpha) \subset \mathbf{Z}_\alpha$, then the theorem would be exactly as stated, except condition (C2) would become:

$$Y(\mathscr{R}\vartheta(\alpha))\mathscr{R}\mathscr{P}(\mathscr{R}\vartheta(\alpha))\dot{\vartheta}(\alpha) = \begin{bmatrix} v_{\text{hip}} \\ \mathbf{0} \end{bmatrix}. \tag{C2'}$$

We also numerically solved the optimization in this case, but while we were able to find a solution, we were unable to accurately fit the human data; specifically, we could fit all the human outputs well except the stance knee (the pattern repeated for PHZD, but to a much smaller degree). We argue that this is due to differences in the model of the robot and the human; thus, constraining the robot to evolve on the 1-dimensional surface, $\mathbf{Z}_\alpha$, is too restrictive to result in "good" robotic walking.

PHZD Optimization Results. By solving the optimization problem in Theorem 1, we are able to determine parameters α^* that automatically guarantee PHZD for the hybrid system $\mathscr{H}_R^{(\alpha^*,\varepsilon)}$ modeling the bipedal robot. In addition to the constraints in Theorem 1, when running this optimization we added the constraint that $\tau(\vartheta(\alpha)) \leq t^H[K]$, with $t^H[K]$ the last time for which there is data for the human, i.e., the duration of one step; this ensures that that canonical human walking functions can be compared to the human output data over the entire step. It is important to note that this optimization does not require us to solve any of the dynamics for $\mathscr{H}_R^{(\alpha,\varepsilon)}$ due to the independence of the conditions (C1)-(C3) on state, i.e., they can be

computed in closed form from the model of the robot. Therefore, the optimization in Theorem 1 can be numerically solved in a matter of seconds[2].

The results of this optimization can be seen in Fig. 8.5, with the specific parameter values given in Table 8.2. In particular, note that with the exception of the stance knee θ_{sk}, all of the correlations are between 0.9850 and 0.9995, indicating that an exceptionally close fit to the human data is possible, even while simultaneously satisfying the PHZD conditions. The stance knee is an exception here; while the fits are still good, they have slightly lower correlations. This is probably due to the fact that the stance knee bears the weight of the robot, and hence is more sensitive to differences between the model of the robot and the human. That being said, it is remarkable how close the canonical human functions can be fit to the human data since the constraints depend on the model of the robot, which obviously has very different dynamics than that of the human.

Table 8.2 Parameter values of the canonical walking functions for the 3 subjects, obtained by solving the optimization problem in Theorem 8.1, together with the cost and correlations of the fits

$y_{H,1} = vt$ $y_{H,2} = e^{-\alpha_4 t}(\alpha_1 \cos(\alpha_2 t) + \alpha_3 \sin(\alpha_2 t)) + \alpha_5$									
S	Fun.	v	α_1	α_2	α_3	α_4	α_5	Cost	Cor.
1	p_{hip}	1.1534	*	*	*	*	*	0.2152	0.9995
	m_{nsl}	*	0.1407	7.7813	0.1332	-2.2899	0.2359		0.9940
	θ_{sk}	*	-0.1942	9.1241	0.8089	11.490	0.2290		0.9185
	θ_{nsk}	*	-0.3248	9.2836	0.4103	-0.2790	0.5517		0.9960
2	p_{hip}	0.9812	*	*	*	*	*	0.3055	0.9992
	m_{nsl}	*	0.0731	8.1450	0.1355	-2.6624	0.2774		0.9850
	θ_{sk}	*	-0.1677	11.852	0.2684	6.3379	0.2010		0.9074
	θ_{nsk}	*	-0.3667	-8.8272	-0.4053	-0.1707	0.5662		0.9959
3	p_{hip}	1.2287	*	*	*	*	*	0.2949	0.9990
	m_{nsl}	*	0.0876	7.9627	0.0552	-4.2023	0.3517		0.9944
	θ_{sk}	*	-0.3283	10.279	0.5051	11.273	0.3638		0.8877
	θ_{nsk}	*	-0.2432	8.6235	0.4105	-0.9881	0.6053		0.9947

8.6 Automatically Generating Stable Robotic Walking

This section demonstrates through simulation that the parameters α^* solving the optimization problem in Theorem 8.1 automatically produce an exponentially stable periodic orbit, i.e., a stable walking gait, for which $(\vartheta(\alpha^*), \dot{\vartheta}(\alpha^*))$ is the fixed point. That is, using the human data, we are able to automatically generate parameters for the controller (8.17) that result in stable walking for which the partial zero

[2] Note that the optimization problem is prone to local minima, so it is likely that there are better fits to the human data than reported.

dynamics are hybrid invariant. We also demonstrate the extensibility of the canonical human walking functions by showing that they also can be used to obtain stable robotic walking in the case of underactuation at the stance ankle. Thus, the ideas presented here are not fundamentally limited to fully actuated systems (unlike other popular methods for producing robotic walking such as controlled symmetries [27] and geometric reduction [6], which build upon principles of passive walking [18] and require full actuation to modify the Lagrangian of the robot).

Walking Gaits from the PHZD Optimization. Before discussing the simulation results of this paper, we justify why robotic walking automatically results from solving the PHZD Optimization in Theorem 8.1. In particular, let α^* be the solution to the optimization problem (8.27) in Theorem 8.1. Then the claim is that there exists an $\overline{\varepsilon} > 0$ such that for all $\varepsilon > \overline{\varepsilon}$ the hybrid system $\mathscr{H}_R^{(\alpha^*,\varepsilon)}$ has a locally exponentially stable periodic orbit $\mathscr{O}_{\alpha^*} \subset \mathbf{PZ}_{\alpha^*}$. Moreover, $(\vartheta(\alpha^*), \dot{\vartheta}(\alpha^*)) \in S_R \cap \mathbf{Z}_{\alpha^*}$ is "sufficiently" close to the fixed point of this periodic orbit and $\tau(\vartheta(\alpha^*)) > 0$ is "sufficiently" close to its period. Informally speaking, this follows from the fact that for *full* hybrid zero dynamics, i.e., $\Delta_R(S_R \cap \mathbf{Z}_\alpha) \subset \mathbf{Z}_\alpha$, these statements can be proven by using the low dimensional stability test of [33] (with the restricted Poincaré map being trivial), coupled with the fact that points in $\mathbf{PZ}_{\alpha^*}$ will converge exponentially fast to the surface $\mathbf{Z}_{\alpha^*}$. We justify these statements further through the use of simulation results.

Robotic Walking with PHZD. To demonstrate how we automatically generate robotic walking from the human data, we do so for each of the subjects considered in this paper. In particular, consider the bipedal robot in Fig. 8.1 with the mass and length parameters specific to each of the three subjects according to Fig. 8.2, from which we construct a hybrid system model, $\mathscr{H}_{R,i}^{(\alpha,\varepsilon)}$ with $i = 1,2,3$, for each according to (8.19) utilizing the human-inspired controllers. Applying Theorem 8.1, we automatically obtain the set of parameters α_i^* such that $\mathscr{H}_{R,i}^{(\alpha_i^*,\varepsilon)}$ is guaranteed to have partial hybrid zero dynamics. Moreover, the explicit functions used to compute this vector of parameters automatically produces the point $(\vartheta(\alpha_i^*), \dot{\vartheta}(\alpha_i^*))$ that, as will be seen, is a fixed point to an exponentially stable periodic orbit. Thus, for each of the subjects, using only their physical parameters and walking data, we automatically and provably obtain stable walking gaits.

The walking gaits that are obtained for each of the subjects through the aforementioned methods can be seen in Fig. 8.6, where the walking is simulated over the course of one step with $\varepsilon = 20$. The specific periodic orbits can be seen in the right column of that figure with the fixed points given in Fig. 8.7. Moreover, stability of the walking is verified by computing the eigenvalues of the Poincaré map; these values can also be found in Fig. 8.7. Note that, unlike the robotic walking in the case where hybrid zero dynamics were not enforced (as discussed in Sect. 8.4), the velocities of the robot are much more reasonable (with a maximum value of approximately $6\frac{rad}{s}$ for Subject 1 as opposed to exceeding $12\frac{rad}{s}$ as in the non-HZD case). This is a very interesting byproduct of the fact that the optimization appears to automatically converge to a fixed point $(\vartheta(\alpha_i^*), \dot{\vartheta}(\alpha_i^*))$ in which the non-stance

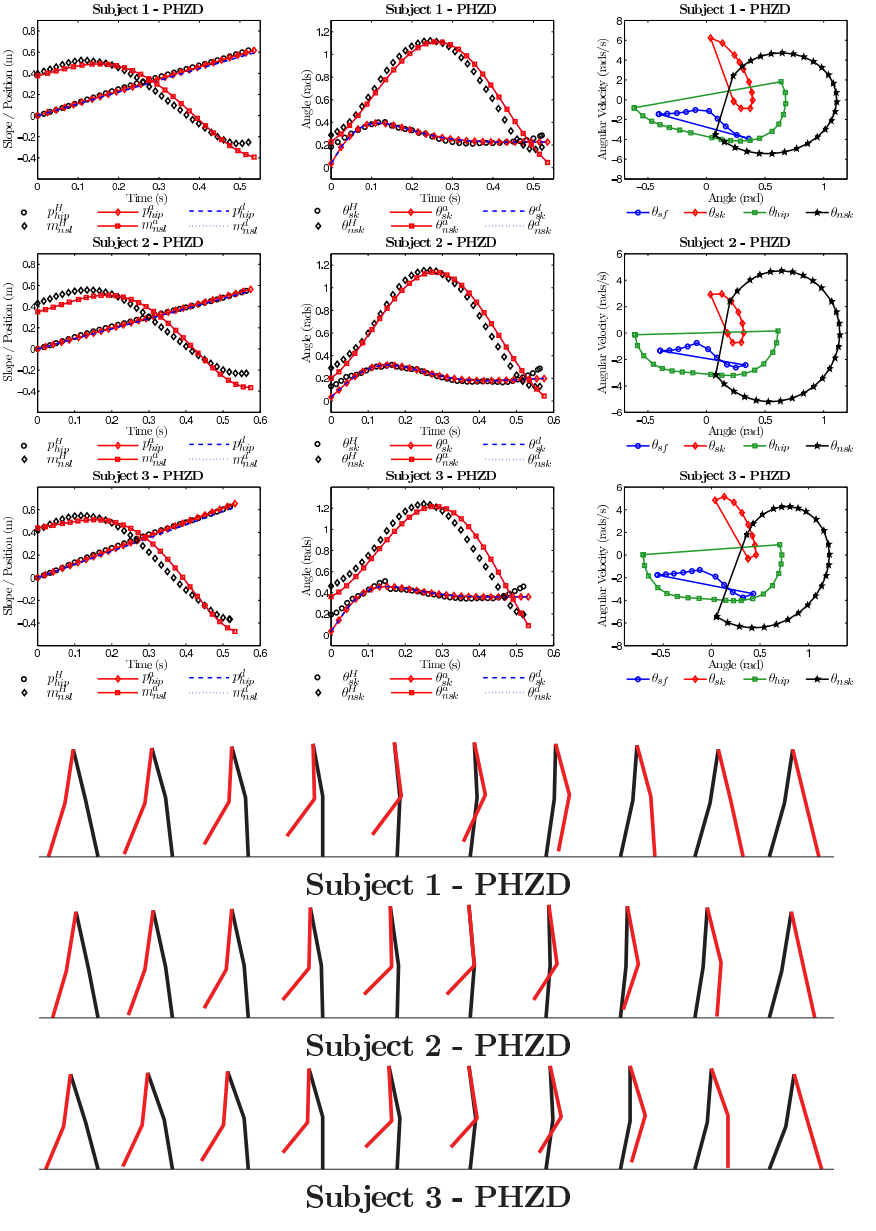
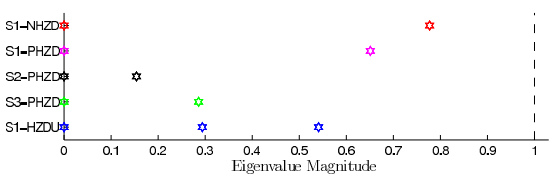


Fig. 8.6 The human data compared to the actual and desired canonical human walking functions for the 3 subjects over one step with the parameters that guarantee PHZD as determined by Theorem 8.1 (left and middle column), along with the corresponding periodic orbits (right column) and tiles of the walking for each subject (bottom)



(a) Eigenvalues

Sub.	s_1	s_1	s_2	s_3	s_1
Type	NHZD	PHZD	PHZD	PHZD	HZDU
θ_{sf}	-0.475	-0.469	-0.444	-0.599	-0.386
θ_{sk}	0.255	0.227	0.199	0.362	0.119
θ_{hip}	-0.681	-0.637	-0.610	-0.703	-0.747
θ_{nsk}	0.187	0.035	0.033	0.036	0.545
$\dot{\theta}_{sf}$	-1.703	-1.492	-1.337	-1.788	-0.620
$\dot{\theta}_{sk}$	0.331	0.020	0.114	0.020	-1.853
$\dot{\theta}_{hip}$	-0.748	-0.472	-0.220	0.462	5.943
$\dot{\theta}_{nsk}$	-2.764	-3.065	-2.556	-5.364	-8.404

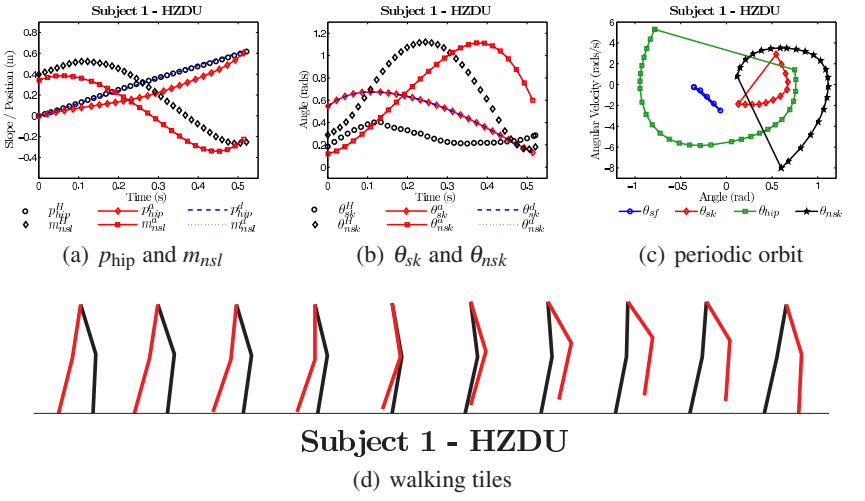
(b) Fixed Points

Fig. 8.7 The magnitude of the eigenvalues (a) associated with the periodic orbits with fixed points (b) for Subject 1 in the non-HZD case, Subject 1-3 in the case of PHZD, and Subject 1 in the underactuated HZD case

leg is essentially locked at foot impact ($\vartheta(\alpha_i^*)_{nsk} \approx 0$), and thus the shocks due to impact are naturally absorbed by the mechanical components of the system rather than having to be compensated for through actuation. This can be visually verified in the tiles of the walking gait in Fig. 8.6 and perhaps, even better, in the movies of the robotic walking obtained (see [1]).

There are some noteworthy aspects of the relationships between the robotic walking obtained and the human data from which it was derived. In particular, as can be seen in the left and middle columns of Fig. 8.6, the actual and desired relative degree 2 outputs agree for all time; thus partial hybrid zero dynamics has been verified through simulation, and the canonical walking function fits produced by Theorem 8.1 (as seen in Fig. 8.5) are the actual outputs of the robot. In addition, while the relative degree 1 output is not invariant through impact, its change is so small that it cannot be seen in the plots. It is interesting to note that the largest deviation from the human outputs is at the beginning and at the end of the step. This makes intuitive sense: the robot, unlike the human, does not have feet, so the largest differences between the two models occur when the feet play a more active role which, in the case of this data, is at the beginning and end of the step (see [25] for a formal justification of this statement). Moreover, the robot and the human react much differently to the impact with the ground, which is again due to the presence of feet and the fact that the robot is rigid while the human is compliant. Thus, the deviations pre- and post-impact are largest due to these differences, and also due to the fact that in the robot we enforce partial hybrid zero dynamics. All that being said, the agreement between the outputs of the robot and the human output data is remarkable. It is, therefore, reasonable to conclude that we have achieved "human-like" robotic walking. Readers are invited to draw their own conclusions by watching the movie of the robotic walking achieved by referring to [1] and related videos at [3].

Underactuated Human-Inspired HZD. To demonstrate the extensability of the methods introduced in this paper, we apply them in the case when the robot being considered is underactuated; specifically, the stance ankle cannot be controlled. This is more "realistic" for the robot being considered due to the fact that it has point feet. As will be seen, this restriction on the actuation of the robot will result in less



(a) p_{hip} and m_{nsl} (b) θ_{sk} and θ_{nsk} (c) periodic orbit

(d) walking tiles

Fig. 8.8 Walking obtained for Subject 1 with hybrid zero dynamics and underactuation at the stance ankle, with (a)-(b) the actual and desired outputs vs. the human output data over one step, (c) the periodic orbit and (d) the tiles (or snapshots) of the walking gait

"human-like" walking, which naturally follows from the fact that humans actively use their stance ankle when walking.

For the sake of brevity, we consider the hybrid system associated with Subject 1, $\mathscr{H}_R^{(\alpha,\varepsilon)}$, where in this case we remove the actuation at the ankle. The control law for this system therefore becomes the control law in (8.17) with the top row removed and with decoupling matrix A as in (8.18) again with the top row removed. Moreover, in this case $\mathbf{Z}_\alpha = \mathbf{P}\mathbf{Z}_\alpha$, i.e., the full and partial hybrid zero dynamics are just the classical hybrid zero dynamics as considered in [33]. Therefore, the optimization problem in Theorem 8.1 ensures hybrid zero dynamics *if* $(\vartheta(\alpha), \dot{\vartheta}(\alpha))$ is an initial condition to a stable periodic orbit. Note that, unlike the case with full actuation, it is necessary to simulate the system to ensure that this is the case; as a result, the optimization is dramatically slower (producing a solution in approximately 12 minutes as opposed to 10 seconds).

The end result of solving the optimization in Theorem 8.1 is a stable periodic orbit with hybrid zero dynamics. The walking that is obtained can be seen in Fig. 8.8, where the outputs of the robot and the human are compared, the periodic orbit is shown, along with tiles of the walking gait. It is interesting to note that the behavior of the walking in the case of underactuated HZD is substantially different than the walking for fully-actuated PHZD. In particular, due to the lack of actuation at the ankle, the robot swings its leg dramatically forward in order to push the center of mass in front of the stance ankle. What is remarkable is that this change in behavior was not hard coded, but rather naturally resulted from the optimization. This provides further evidence for the fact that the human walking functions are, in fact, canonical since they can be used to achieve a variety of walking behaviors.

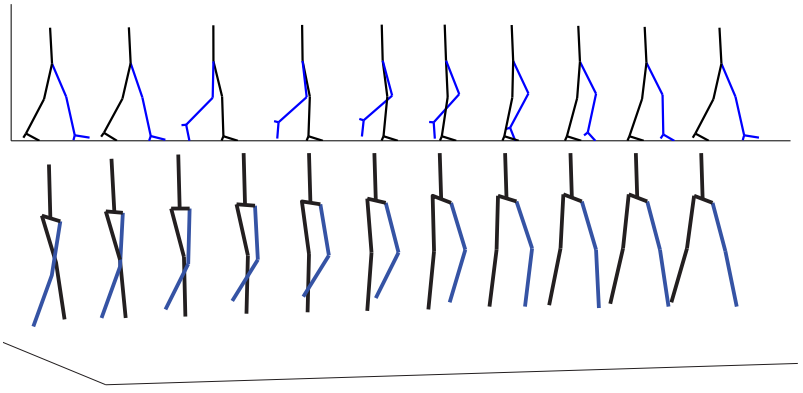


Fig. 8.9 Tiles of the walking gaits obtained through the use of human-inspired outputs for a 2D robot with feet, knees and a torso (top) and a 3D robot with knees and a torso (bottom) utilizing the data for Subject 1

8.7 Conclusions and Future Challenges

This paper presents the first steps toward automatically generating robotic walking from human data. To achieve this goal, the first half of the paper introduces three essential constructions: (1) define human outputs that appear characteristic of walking, (2) determine canonical walking functions describing these outputs, and (3) use these outputs and walking functions, in the form of human-inspired outputs for a bipedal robot, to design a human-inspired controller. We obtain parameters for this control law through an optimization problem that guarantees PHZD and that explicitly produces a fixed point to an exponentially stable periodic orbit contained in the zero dynamics surface. A 2D bipedal robot with knees is considered and these results are applied to experimental walking data for a collection of subjects, allowing us to automatically generate stable robotic walking that is as close to the human data as possible, i.e., walking that is as "human-like" as possible.

While the results of this paper are limited to a simple bipedal robot, human-inspired controllers have been used to achieve robotic walking for 2D bipeds with knees, feet and a torso [25] and 3D bipeds with knees and a torso [24] (illustrated in Fig. 8.9). Yet formal results ensuring hybrid zero dynamics and proving the existence of a stable periodic orbit have yet to be established. The difficulty in extending the results of this paper comes from the discrete phases of walking present in more anthropomorphic bipedal robots due to the behavior of the feet [7]. On each domain human outputs must be defined and canonical walking functions determined in such a way that, using only the human data, parameters for the corresponding human-inspired controllers can be determined so as to guarantee (partial) hybrid zero dynamics throughout the entire step. This points toward a collection of challenging problems, the solutions for which could allow for the automatic generation of human-like robotic walking from human data.

To provide concrete objectives related to the overarching goal of obtaining truly human-like walking for anthropomorphic bipedal robots, as motivated by the ideas related to this paper, we lay out a serious of "challenge problems" for human-inspired bipedal robotic walking:

Problem 1: Obtain *human-inspired hybrid system models* of anthropomorphic bipedal robots.
Problem 2: Determine *outputs* of the human to consider and associated *human walking functions* that are *canonical*.
Problem 3: Use Problem 2 to design *human-inspired controllers* and obtain formal conditions that guarantee hybrid zero dynamics (either full or partial).
Problem 4: Determine the parameters of these controllers *automatically from human data* using no a priori knowledge about the data.
Problem 5: Provably and automatically *produce stable periodic orbits*, i.e., stable walking gaits, that are as "human-like" as possible.

Importantly, the solutions of these problems must go beyond the task of achieving flat-ground straight-line walking, but rather should be automatically extensible to a wide variety of walking behaviors, or *motion primitives*, e.g., walking up and down slopes, stair-climbing, turning left and right, walking on uneven terrain. It is therefore necessary to determine methods for transition between walking motion primitives, formally encoded as solutions to Problems 1-5, so that stability is maintained. Finally, in order to truly demonstrate the validity of specific solutions to the challenge problems, it is necessary to ultimately realize them on physical bipedal robots.

The solutions to these challenge problems would have far-reaching ramifications for our understanding of both robotic and human walking. They would result in bipedal robots able to produce the variety of walking behaviors that allow humans to navigate diverse terrain and environments with ease, resulting in important applications to areas like space exploration. In addition, they would have potentially revolutionary applications to all areas in which humans and robots interact to achieve locomotion – from rehabilitation, prosthetic and robotic assistive devices able to supplement impaired walking to aiding the walking of healthy humans through exoskeleton design.

Acknowledgements. I would like to thank Matthew Powell for his contribution in generating figures and parts of the code used to display the results of this paper, Ryan Sinnet for his contributions to human-inspired control that led to this work, and Ram Vasudevan for many conversations on the role of optimization in automatically generating robotic walking from human data and for helping to run the experiments discussed in this paper. I am also grateful Jessy Grizzle for the many insightful discussions on bipedal walking over the years, and Ruzena Bajcsy for introducing me to the world of experimentation.

This work is supported by NSF grant CNS-0953823 and NHARP award 00512-0184-2009.

References

1. Movie of of the robotic walking obtained via human-inspired hybrid zero dynamics, http://www.youtube.com/watch?v=72hSW44qMgw
2. Website for data set and related papers, http://www.eecs.berkeley.edu/~ramv/HybridWalker
3. YouTube page for AMBER lab, http://www.youtube.com/user/ProfAmes
4. Ambrose, R., Aldridge, H., Askew, R., Burridge, R., Bluethmann, W., Diftler, M., Lovchik, C., Magruder, D., Rehnmark, F.: Robonaut: NASA's space humanoid. IEEE Intelligent Systems and their Applications 15(4), 57–63 (2000)
5. Ames, A.D., Gregg, R., Wendel, E., Sastry, S.: On the geometric reduction of controlled three-dimensional bipedal robotic walkers. In: Lagrangian and Hamiltonian Methods for Nonlinear Control. LNCIS, vol. 366, pp. 183–196. Springer, Heidelberg (2007)
6. Ames, A.D., Sinnet, R.W., Wendel, E.D.B.: Three-dimensional kneed bipedal walking: A hybrid geometric approach. In: Majumdar, R., Tabuada, P. (eds.) HSCC 2009. LNCS, vol. 5469, pp. 16–30. Springer, Heidelberg (2009)
7. Ames, A.D., Vasudevan, R., Bajcsy, R.: Human-data based cost of bipedal robotic walking. In: Hybrid Systems: Computation and Control, Chicago, IL (2011)
8. Au, S.K., Dilworth, P., Herr, H.: An ankle-foot emulation system for the study of human walking biomechanics. In: IEEE Intl. Conf. Robotics and Automation, Orlando, pp. 2939–2945 (2006)
9. Braun, D.J., Goldfarb, M.: A control approach for actuated dynamic walking in bipedal robots. IEEE TRO 25(6), 1292–1303 (2009)
10. Chevallereau, C., Bessonnet, G., Abba, G., Aoustin, Y.: Bipedal Robots: Modeling, Design and Walking Synthesis. Wiley-ISTE, New York (2009)
11. Chevallereau, C., Formal'sky, A., Djoudi, D.: Tracking a joint path for the walk of an underactuated biped. Robotica 22(1), 15–28 (2004)
12. Childs, D.W.: Dynamics in Engineering Practice, 10 edn. CRC Press (2010)
13. Grizzle, J.W., Abba, G., Plestan, F.: Asymptotically stable walking for biped robots: Analysis via systems with impulse effects. IEEE TAC 46(1), 51–64 (2001)
14. Grizzle, J.W., Chevallereau, C., Ames, A.D., Sinnet, R.W.: 3D bipedal robotic walking: models, feedback control, and open problems. In: IFAC Symposium on Nonlinear Control Systems, Bologna (2010)
15. Heller, M.O., Bergmann, G., Deuretzbacher, G., Dürselen, L., Pohl, M., Claes, L., Haas, N.P., Duda, G.N.: Musculo-skeletal loading conditions at the hip during walking and stair climbing. J. of Biomechanics 34(1), 883–893 (2001)
16. Hürmüzlü, Y., Marghitu, D.B.: Rigid body collisions of planar kinematic chains with multiple contact points. Intl. J. of Robotics Research 13(1), 82–92 (1994)
17. Kuo, A.D.: Energetics of actively powered locomotion using the simplest walking model. Journal of Biomechanical Engineering 124, 113–120 (2002)
18. McGeer, T.: Passive dynamic walking. Intl. J. of Robotics Research 9(2), 62–82 (1990)
19. Morris, B., Grizzle, J.: A restricted Poincaré map for determining exponentially stable periodic orbits in systems with impulse effects: Application to bipedal robots. In: IEEE Conf. on Decision and Control, Seville, Spain (2005)
20. Murray, R.M., Li, Z., Sastry, S.S.: A Mathematical Introduction to Robotic Manipulation. CRC Press, Boca Raton (1994)
21. Sastry, S.S.: Nonlinear Systems: Analysis, Stability and Control. Springer, New York (1999)
22. Sauer, P., Kozłowski, K.R., Morita, Y., Ukai, H.: Ankle robot for people with drop foot – case study. In: Kozłowski, K.R. (ed.) Robot Motion and Control 2009. LNCIS, vol. 396, pp. 443–452. Springer, Heidelberg (2009)

23. Schaub, T., Scheint, M., Sobotka, M., Seiberl, W., Buss, M.: Effects of compliant ankles on bipedal locomotion. In: EEE/RSJ International Conference on Intelligent Robots and Systems (2009)
24. Sinnet, R., Powell, M., Jiang, S., Ames, A.D.: Compass gait revisited: A human data perspective with extensions to three dimensions. Submitted for publication, available upon request
25. Sinnet, R., Powell, M., Shah, R., Ames, A.D.: A human-inspired hybrid control approach to bipedal robotic walking. In: 18th IFAC World Congress, Milano, Italy (2011)
26. Sinnet, R.W., Ames, A.D.: 2D bipedal walking with knees and feet: A hybrid control approach. In: 48th IEEE Conference on Decision and Control, Shanghai, P.R. China (2009)
27. Spong, M.W., Bullo, F.: Controlled symmetries and passive walking. IEEE TAC 50(7), 1025–1031 (2005)
28. Srinivasan, S., Raptis, I.A., Westervelt, E.R.: Low-dimensional sagittal plane model of normal human walking. ASME J. of Biomechanical Eng. 130(5) (2008)
29. Srinivasan, S., Westervelt, E., Hansen, A.: A low-dimensional sagittal-plane forward-dynamic model for asymmetric gait and its application to study the gait of transtibial prosthesis users. ASME J. of Biomechanical Eng. 131 (2009)
30. Vasudevan, R., Ames, A.D., Bajcsy, R.: Using persistent homology to determine a human-data based cost for bipedal walking. In: 18th IFAC World Congress, Milano, Italy (2011)
31. Vukobratović, M., Borovac, B., Surla, D., Stokic, D.: Biped Locomotion. Springer, Berlin (1990)
32. Wendel, E., Ames, A.D.: Rank properties of Poincaré maps for hybrid systems with applications to bipedal walking. In: Hybrid Systems: Computation and Control, Stockholm, Sweden (2010)
33. Westervelt, E.R., Grizzle, J.W., Chevallereau, C., Choi, J.H., Morris, B.: Feedback Control of Dynamic Bipedal Robot Locomotion. CRC Press, Boca Raton (2007)
34. Westervelt, E.R., Grizzle, J.W., Koditschek, D.E.: Hybrid zero dynamics of planar biped walkers. IEEE TAC 48(1), 42–56 (2003)
35. Winter, D.A.: Biomechanics and Motor Control of Human Movement, 2nd edn. Wiley Interscience, New York (1990)
36. Zatsiorsky, V.M.: Kinematics of Human Motion, 1 edn. Human Kinetics (1997)

Chapter 9
Dynamic Position/Force Controller of a Four Degree-of-Freedom Robotic Leg

Arne Rönnau, Thilo Kerscher, and Rüdiger Dillmann

Abstract. The leg controllers of a six-legged walking robot have a great influence on the walking capabilities of the robot. A good controller should create a smooth locomotion, minimize mechanical stress and soften collision impacts. This article introduces a model-based position/force controller, which is able to eliminate the negative effects of non-linear disturbances with a computed torque control approach, but also creates very smooth leg movements. An additional force feedback reduces the influences of leg collisions and is able to maintain a desired ground contact force during the stance phase.

9.1 Introduction

Although walking seems so easy, when observing insects or animals, it requires a lot of effort to achieve similar walking capabilities with robots. This is why Nature can still guide developers of walking robots to new innovative designs, mechanical constructions, control mechanisms, and software systems.

The walking performance and the naturalness of the entire locomotion of a walking robot do not only increase its acceptance, but also have an important background. When walking with very smooth leg movements, less energy is lost by collisions with the ground or obstacles. This does not only increase the operational range of an autonomous robot, but also leads to less deterioration of the mechanical components which enhances the robot's lifetime. Because walking robots are designed to operate in rough and unknown areas, it is difficult to create a smooth locomotion at all times. Even with complex environment models and planned foot holds, it is not possible to avoid all leg collisions or prevent leg slippages at all times.

Arne Rönnau · Thilo Kerscher · Rüdiger Dillmann
FZI Research Centre for Information Technology, Haid-und-Neu-Str. 10-14,
D-76131 Karlsruhe, Germany
e-mail: {roennau,kerscher,dillmann}@fzi.de

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 117–126.
springerlink.com

In this article we present a position/force controller for a six-legged walking robot, which is able to reduce the negative effect of leg collisions and create smooth leg movements. The inner position control loop is realized with a PD controller that is extended by a central feed-forward controller using the inverse dynamics of the robotic leg. With this computed torque control approach we are able to compensate the non-linear effects of inertia, centripetal and Coriolis forces, and gravity. The leg trajectories do not only consist of the target position and force of the foot tip, but also its desired velocity and acceleration, which enables us to perform smooth leg movements. The force control loop of the developed controller reduces the effects of collisions and keeps the contact force to the ground at a desired value.

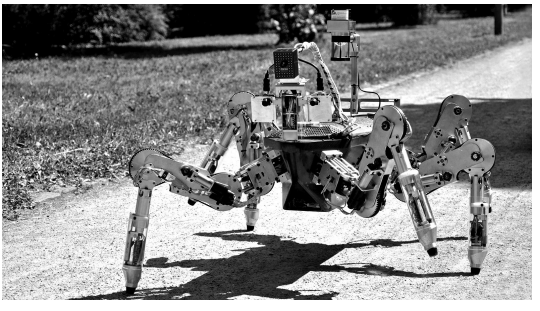


Fig. 9.1 Six-legged walking robot LAURON IVc

In Section 9.2 we shortly present the six-legged walking robot LAURON (see Fig. 9.1). After this short introduction, we give some details on the new four joint leg design. The main focus of this article, the developed model-based hybrid control approach is described in Section 9.4. Some experiments and results can be found in Section 9.5. And finally, Section 9.6 gives a short summary and presents some future work.

9.2 Biologically-Inspired Walking Robot LAURON

After developing some wooden leg prototypes, the first generation of the six-legged walking robot LAURON was presented to the public on the CEBIT, Germany, in the year 1994. The three joints of the first LAURON leg were a simplified approximation of the leg of a stick insect [2] (see Fig. 9.2(a)). A lot of hardware and software improvements have been made since this first generation, but the kinematic structure of the legs has been kept over all the years [5].

The fourth and current generation, LAURON IVc, was improved mechanically compared to its ancestors, but still uses the same approved three joint leg kinematics (shown in Fig. 9.2(b)). LAURON IVc is bigger and heavier than the third generation and reaches a size of about 1.2×1.0 m with a weight of 27 kg, but also has interesting improvements like an extra Vision-PC and spring-damper feet.

With its behaviour-based control system and its six legs, LAURON is able to walk statically stable at all times, which enables it to traverse difficult and rough terrain [3]. LAURON IVc is equipped with several sensor systems like different cameras. These sensor systems are used to gather information about the surrounding environment, which is then used to plan foot holds or inspection tasks. Although LAURON can walk over rough, unstructured terrain and is able to cope with difficult, unknown scenarios, its kinematic structure has some limitations. One reason for these limitations is LAURON IVc's lack of ability to orientate its feet freely towards the ground. With only three joints in each leg LAURON is not able to influence the angle of the contact forces, making it difficult to walk up precipitous slopes and very steep stairways.

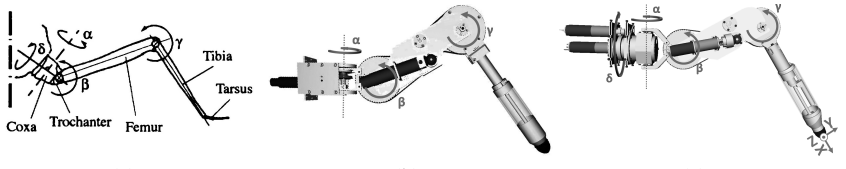


Fig. 9.2 Comparing the kinematic structure of a stick insect leg (a) [8] with the leg of LAURON IVc (b) and the new leg design with four joints and the foot tip coordinate system (c)

9.3 New Leg Design with Four Degrees-of-Freedom

With the aim to increase LAURON's capabilities, we have decided to extend its workspace by adding a fourth joint to the leg design. Taking a closer look at the stick insect (see Fig. 9.2) reveals that the stick insect has an extra joint (δ), which is situated close to the central body. This fourth joint δ, which is mainly used while climbing in the leafs of trees [8], was added to the leg design of LAURON.

The additional joint was designed in a very compact way by using a spur wheel section and its inner space for the second joint α [5]. In the last years the leg design of LAURON IVc has proved to be very robust and reliable. Therefore, the rest of the new leg design was derived from this previous robust design. Details can be found in the illustration of the first leg prototype presented in Fig. 9.2(c).

With the extra joint close to the main body, it is possible to orientate the foot tip towards the ground. But the new design also has some drawbacks. Due to the fourth joint the inverse kinematics is ambiguous, which means that one foot tip position can be reached by multiple joint configurations. By adding the desired orientation of the foot tip as a kinematic constraint, we were able to find a geometrical solution for

the inverse kinematics [4]. The simplicity of this geometrical solution is important to achieve fast computation, which is needed by the leg controller.

9.4 Control Approach

Currently the walking robot LAURON IVc uses joint independent position-based PID controllers with velocity feedback. But these cannot consider the coupling of the joints or compensate influences from non-linear disturbances like gravity. With an additional joint in the new leg design these negative effects are likely to increase. This is why our new control approach takes the joint coupling into account and uses the leg's dynamics to compensate the most important non-linear disturbances. Because the leg position is the most important factor, the key part of the new control system is a position-based controller. However, this controller uses the dynamics of the leg and considers the weight transfer from one leg to the other while walking. Additionally it was extended by a force element, enabling it to keep an optimal contact force to the ground and weaken collision impacts.

9.4.1 Control Structure

The inner feedback control system is a simple PD controller, which controls the leg position in the joint space of the leg. A PD controller has some good characteristics and can be tuned easily. This inner control cycle is extended by a feed-forward control part (Inv. Dynamics, Weight Transfer block), also called computed torque control. The computed torque is based on the inverse dynamics of the leg and enables the controller to eliminate non-linear disturbances. Furthermore, the feed-forward part also considers the weight transfer from one leg to the other while walking. Using the feed-forward torque relieves the inner PD controller, but also requires more complex trajectories with desired joint positions, velocities, and accelerations. These joint space trajectories are calculated from the desired position trajectories in operational space. The force control loop compares the measured forces at the foot tip with the desired force trajectories. Based on the mechanical impedance and the actual force difference a positional adjustments (in operational space) is created in the Force/Compliance block. This adjustment is added to the desired foot tip position trajectories and then passed to the Inverse Kinematics block. After the joint angles have been calculated the joint accelerations and velocities are derived from these angles. For a quick overview, the structure of the developed control approach is illustrated as a block scheme diagram in Fig. 9.3.

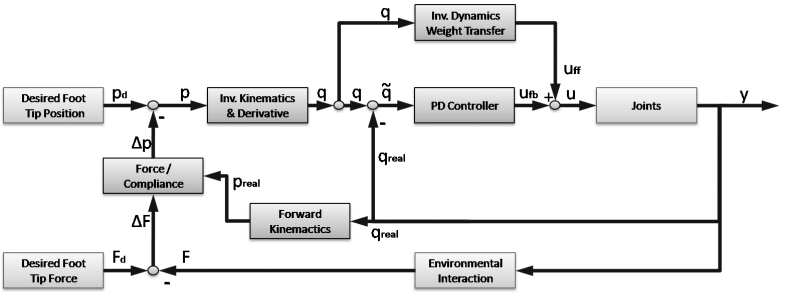


Fig. 9.3 Block scheme of the hybrid position/force control approach

9.4.2 Dynamics

The feed-forward controller is based on the inverse dynamics and therefore we will briefly present important details on the dynamics of the robotic leg. The dynamics of the new leg is modelled with the help of the Euler-Lagrange Equation [6]:

$$\frac{d}{dt}\frac{\partial L}{\partial q_i} - \frac{\partial L}{\partial q_i} = \tau_i, \tag{9.1}$$

where $L = K - P$ is the Lagrangian with $K := kinetic$ and $P := potential\ energy$.

In general the kinetic energy can be estimated with an equation which includes the translational and rotational movement of a rigid object with the mass m:

$$K = \frac{1}{2} m v^T v + \frac{1}{2} \omega^T I \omega. \tag{9.2}$$

The inertia tensor I is a 3×3 matrix with the main inertia tensors I_{XX}, I_{YY}, and I_{ZZ} that describe the rotation around the main coordinate axes of the object. By approximating the new leg with cuboids and cylinders it is possible to determine the main inertia tensors I_i for the centre of mass of each leg element i.

Then with the geometrical Jacobian matrix J it is possible to transform these inertia tensors into the translational ($\dot{p}$) and rotational (ω) velocities of the end effector. The following equations clarify this relationship (where q are the joint angles calculated by the inverse kinematics):

$$J = \begin{pmatrix} J_v \\ J_\omega \end{pmatrix} \qquad \text{with} \qquad \dot{p} = J_v(q)\dot{q} \qquad \text{and} \qquad \omega = J_\omega(q)\dot{q}. \tag{9.3}$$

With the rotational matrices $R_0^i(q)$ it is possible to transform the inertia tensors I_i into the base frame of the leg:

$$I = R_0^i(q) I_i R_0^i(q)^T. \tag{9.4}$$

Together with the masses of the leg elements, we are now able to calculate the total kinetic energy K of the leg:

$$K = \frac{1}{2}\dot{q}^T \left[\sum_{i=1}^{4} \left\{ m_i J_{v_i}(q)^T J_{v_i}(q) + J_{\omega_i}(q)^T R_0^i(q) I_i R_0^i(q)^T J_{\omega_i}(q) \right\} \right] \dot{q}. \tag{9.5}$$

Next, we will determine the total potential energy P. This part of the Lagrangian is described by the equation

$$P = \sum_{i=1}^{4} m_i g^T p_{l_i}, \tag{9.6}$$

where m_i are the masses of the leg elements, g is the gravitational vector and p_{l_i} is the position of the centre of mass of each leg element.

Now we can insert K and P in the Euler-Lagrange Equation and solve this differential equation. The complex result can be reordered and combined to three main parts:

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau. \tag{9.7}$$

$D(q)$ represents the inertia tensors, $C(q,\dot{q})$ includes the centripetal and Coriolis forces and $G(q)$ contains the gravitational forces. But this equation cannot represent the weight transfer from one leg to the other while changing from the stance to swing phase. Hence, the equation was extended by an external force part for the foot tip. With this extension it is possible to model the weight transfer phase and to create sufficient torques in all joints while shifting the weight from one leg to the other. The external contact forces in the foot tip F have to be transformed to joint forces F_J. As before, this is realized by multiplication with the Jacobi matrix. Because we are only regarding the force and not the torques in the foot tip, only the translational part of the Jacobi matrix ${J_v}^T$ is needed:

$$F_J = J_v^T F. \tag{9.8}$$

Then the dynamics equation of the leg can be extended to (with $q :=$ joint angles)

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + J_v^T F = \tau. \tag{9.9}$$

The dynamics of the dc motor are now added to this equation. In literature one can find the dynamics equation for a dc motor ([1]):

$$J_m \ddot{q}_m(t) + B\dot{q}_m(t) = \frac{K_T}{R} V(t) - \tau(t), \tag{9.10}$$

with J_m as the torque of inertia of the rotor, V as the electrical input voltage of the dc motor, B as friction torque, K_T as torque constant, R as the ferrule resistor, $\dot{q}_m(t)$ as the motor's angular rate and $\tau(t)$ as the load torque.

Finally, Equations (9.9) and (9.10) can be put together to the dynamics equation. After reordering and combining the elements we receive

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + B\dot{q} + G(q) + J_v^T F = u_{ff}, \tag{9.11}$$

where u_{ff} are the needed joint torques to compensate the dynamic effects when moving along the desired trajectories. The parameters of the dynamic model like the masses of the leg elements and their centre of mass or the parameters of the dc motors were identified by experiments, CAD-data, or manufacturer data-sheets.

9.4.3 Position/Force Controller

In contrast to the joint positions, the force is controlled in the operational space, therefore it is not easy to combine the two controller parts in one equation. But they can be described very well separately. With the dynamics of the leg in Eq. (9.11) it is possible to formulate the control equations for the position controller in joint space. The PD controller can be modelled with ($\tilde{q} = q - q_{\text{real}}$, $q :=$ desired joint angles)

$$K_P\tilde{q} + K_D\dot{\tilde{q}} = u_{fb}. \tag{9.12}$$

Then the position controller output u is the result of the sum of the feed-forward and feedback control $u = u_{ff} + u_{fb}$:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + B\dot{q} + G(q) + J_v^T F + K_P\tilde{q}(t) + K_D\dot{\tilde{q}} = u. \tag{9.13}$$

The force controller is based on the mechanical impedance, which can be formulated with the equation ($S :=$ Stiffness matrix and $D :=$ Damping matrix):

$$\Delta F = S\Delta p + D\dot{p} \quad \text{with} \quad p := x,y,z \text{ position of foot tip.} \tag{9.14}$$

Then the force controller output, which is a positional correction for the desired positional trajectories in operational space, can be obtained with

$$\Delta p = (\Delta F - D\dot{p})S^{-1}. \tag{9.15}$$

9.5 Experiments and Results

The new position/force controller was first tested in several Matlab simulations. With these simulations it was possible to examine the dynamic effects like the moments of inertia, centripetal and Coriolis forces, and gravitational effects. For example Fig. 9.4(a) shows the torques needed to compensate the effects of the moments of inertia. These torques were determined by reducing Eq. (9.11) to only contain the inertia matrix. In Fig. 9.4(b) the torques needed to compensate all dynamic effects are illustrated.

Before testing the controller on the prototype leg, we decided to examine the joint trajectories and input voltage for the 12 V dc motors. The simulated joint trajectories showed a smooth and harmless motion of the leg and the voltages did not exceed 10 V or show any alarming unsteadiness, which makes it safe for the dc motors.

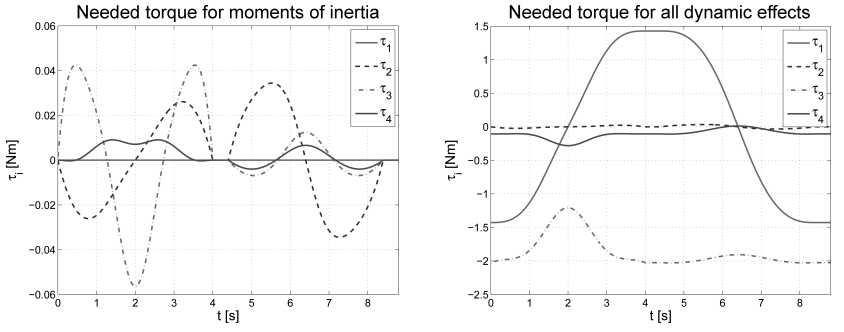


Fig. 9.4 Simulated torques: (a) compensate the effects of moments of inertia and (b) all dynamic effects; $\tau_1 :=$ delta, $\tau_2 :=$ alpha, $\tau_3 :=$ beta, $\tau_4 :=$ gamma joint

After implementing the developed controller in the MCA2 software framework [7] the controller was tested in several experiments. In Fig. 9.5 the real and desired angles for all four joints are illustrated. The trajectories show that the real joint angles have a small steady-state error resulting from the use of a PD controller. But the overall error is not big and the smoothness of the motion was good.

We also examined the force component of the developed leg controller. The desired contact forces were set to zero and the leg trajectory was placed 3 cm above the ground. In this case the leg finished its entire swing and stance movement without touching the ground. After setting up the force trajectories the controller was able to establish and keep ground contact during stance phase (the results are shown in Fig. 9.6).

9.6 Conclusions and Future Work

In this article we have presented a new control approach for a leg of a six-legged walking robot. In contrast to the actual LAURON IVc walking robot this leg has four joints, which increases the operational space significantly. With the help of the inverse dynamics, we are able to compensate almost all non-linear disturbances. The inner position control loop of our controller is based on a PD controller combined with a dynamic feed-forward control. Our developed control approach can easily be transferred to other walking robots after adapting the inverse dynamics and kinematics to the specific robot. Real experiments with the first leg prototype have shown that the PD controller leads to small steady-state errors. But the same experiments have also demonstrated that this new control approach creates very smooth leg trajectories and is able to establish or maintain ground contact reliably with its force feedback. With this advanced control approach we will improve the walking capabilities of the next walking robot generation significantly.

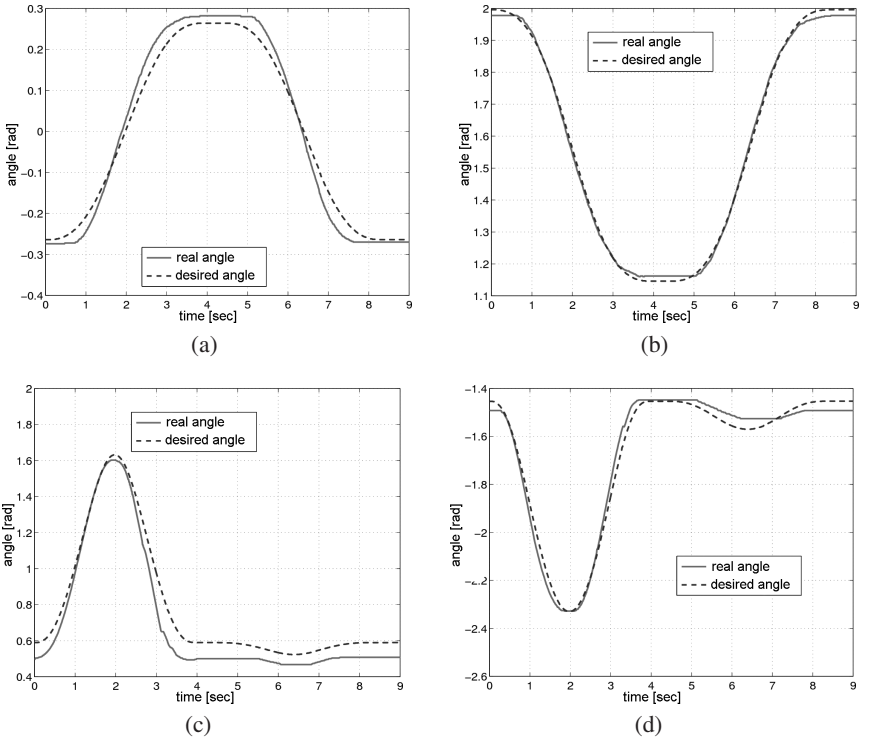


Fig. 9.5 Joint trajectories with steady-state error: (a) delta joint angles q_1, (b) alpha joint angles q_2, (c) beta joint angles q_3, and (d) gamma joint angles q_4

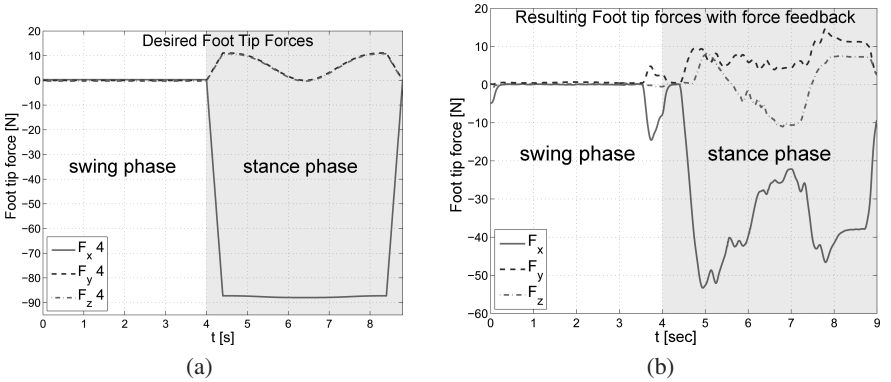


Fig. 9.6 Experiment showing the effectiveness of the force component of the developed controller. Resulting forces were measured with 3D foot force sensor: (a) desired foot tip forces, (b) resulting foot tip forces (foot tip coordinate system shown in Fig. 9.2)

In our future work we will replace the inner PD controller with a PID controller to eliminate the small steady-state errors and investigate the coupling effects between multiple leg controllers on one robot.

References

1. Chiasson, J.: Modeling and High Performance Control of Electric Machines, p. 20. John Wiley & Sons (2005), ISBN 978-0471684497
2. Dillmann, R., Albiez, J., Gassmann, B., et al.: Biologically inspired walking machines: design, control and perception. Phil. Trans. of the Royal Society A: Mathematical, Physical and Engineering Sciences 365(1850), 133–151 (2007)
3. Kerscher, T., Rönnau, A., et al.: Behaviour-based control of a six-legged walking machine LAURON IVc. In: Proceedings of CLAWAR2008, 11th International Conference on Climbing and Walking Robots, Coimbra, Portugal, September 8-10 (2008)
4. Kong, D.-U.: Modellbasierte Hybridregelung eines Laufmaschinenbeins mit vier Bewegungsfreiheitsgraden. Diploma thesis, Forschungszentrum Informatik at the University of Karlsruhe (2010) (in German)
5. Rönnau, A., Kerscher, T., Dillmann, R.: Design and kinematics of a biologically-inspired leg for a six-legged walking machine. In: 3rd IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics, BioRob (2010)
6. Spong, M.W., Hutchinson, S., Vidyasagar, M.: Robot Modeling and Control. John Wiley & Sons (2005), ISBN 978-0471649908
7. Uhl, K., Ziegenmeyer, M.: MCA2 – An Extensible Modular Framework for Robot Control Applications. In: Proceedings of CLAWAR2007, 10th International Conference on Climbing and Walking Robots, Singapore, July 16-18 (2007)
8. Weidemann, H.-J., Eltze, J., Pfeiffer, F.: Leg design based on biological problems. In: Proceedings of the 1993 IEEE International Conference on Robotics and Automation, ICRA (1993)

Chapter 10
Perception-Based Motion Planning for a Walking Robot in Rugged Terrain

Dominik Belter

Abstract. The article presents a path planning algorithm called guided RRT for a six-legged walking robot. The proposed method considers the problem of planning a sequence of elementary motions (steps) and its implementation on the real robot. It takes into account that the robot has limited abilities to perceive the environment. The A* algorithm is used for long horizon planning on a map obtained from the stereo camera data. Then, the RRT-Connect method is used to find a sequence of feasible movements for the body and feet of the robot on a more precise map obtained by using the Hokuyo laser rangefinder. A strategy for path planning is proposed. Experimental results which show efficiency of the algorithm are presented.

10.1 Introduction

Autonomous locomotion of a robot in unknown environment is a fundamental problem in the field of mobile robotics. There is a number of existing solutions to the problem of motion planning for wheeled and legged robots working in indoor environment. Leaving the assumption that the terrain is flat makes the locomotion task significantly more challenging [4]. To find an appropriate path the legged robot should not only avoid obstacles but it also should decide how to climb terrain irregularities. Another problem is caused by leaving the assumption that the robot knows the terrain in advance. During a real mission the robot has to measure the environment while traversing the terrain and then it has to use the environment model to plan further movements.

The presented approach is dedicated to a six-legged walking robot however it can be used on any kind of statically stable walking machine. The practical goal is to use the robot in search and rescue missions in unknown and rough terrain.

Dominik Belter
Institute of Control and Information Engineering, Poznań University of Technology,
ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: dominik.belter@put.poznan.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 127–136.
springerlink.com

10.1.1 Problem Statement

In a real mission the robot does not have a map of the environment and it has to measure the terrain relief. The six-legged Messor robot (Fig. 10.1A) is equipped with two sensors for terrain measurement. Data obtained from these sensors are used to create two partially overlapping elevation maps of the environment. Such a representation of the surroundings is then used to plan movement of the robot.

The first sensor is a 2D laser rangefinder Hokuyo URG-04LX. It is mounted at the front of the robot and tilted down to acquire the profile of the terrain. While the robot is walking a grid-based elevation map is created [3]. The range of the measurements is about 1 m because of the geometrical configuration of the system. This determines the size of the elevation map as 2×2 m (Fig. 10.2A). The robot is located in the center of this map. The map is translated while the robot is walking. The size of a grid cell is 1.5×1.5 cm. It is precise enough to appropriately select footholds and to plan movement of the robot (feet and platform paths). On the other hand the size of the map is not sufficient for motion planning in a longer horizon. An example of the local elevation map obtained by using Hokuyo laser rangefinder is shown in Fig. 10.2B.

To obtain information about the terrain and obstacles which are more distant the robot is equipped with a Videre Design STOC stereo camera. It gives information about the depth of the observed scene and allows creating a global elevation map. This map also moves with the robot, which is located in the center. Its size is set to 10×10 m. The size of the grid cell is 0.1×0.1 m. The low precision of this map is sufficient to roughly plan the path but not to do precise motion planning. An example global elevation map obtained by using stereo camera is shown in Fig. 10.2C.

The two-stage representation of the environment strongly determines the approach to motion planning. The path planner considers not only a 6-dof position of the robot but also its 18-dimensional configuration space (one state for a single joint of the robot's leg). The result of searching in such a huge search space depends on the shape of terrain. The proposed path planning algorithm is based on the grid-based search A* algorithm and a randomized planner – the RRT-Connect algorithm.

Fig. 10.1 Messor robot with Hokuyo laser rangefinder and Videre Design stereo camera (A) and simulated robot (B)

To generate a movement of the walking robot an inclination and distance of the robot's platform to the ground is considered. The robot also selects appropriate footholds and generates paths of its feet to avoid slippages and to preserve a secure posture. The configuration of the robot is also determined by a stability criterion. To check if a transition between two states is possible the robot uses a module for collision detection and determines if the planned configurations are achievable.

10.1.2 State of the Art

A robot can walk over rough terrain without any representation of the environment. The RHex robot was designed to adapt to terrain surface by using an appropriate structure of its legs [4]. Mechanical adaptation to terrain irregularities is effective and reliable. On the other hand such a robot might destroy the terrain which it is walking on. The opposite approach was applied by Kalakrishnan et al. [5]. A hierarchical controller is used to plan and execute the path of the robot. The robot is taught offline how to select footholds. Expert demonstration using a template learning algorithm is used. Then the robot's body path is planned in regions with good footholds. The controller also avoids collisions and optimizes the motion by using a trajectory optimizer.

A combined approach is presented on the BigDog robot [14]. A 2D costmap is created over the terrain. The cost is high for detected objects and its neighborhood. Then an optimal path is searched by using the A* algorithm. The movement execution along the planned path is performed by a reactive controller.

The approach presented here is most similar to the method used on the LittleDog robot [5]. The algorithm implemented on the LittleDog robot finds the best path to the goal by using an elevation map of the environment. Then the motion controller executes the desired path. Simple reactions from force sensors located in the robot's feet and the Inertial Measurement Unit (IMU) are implemented. This makes it possible to compensate inaccuracies of the environment modeling and robot localization errors.

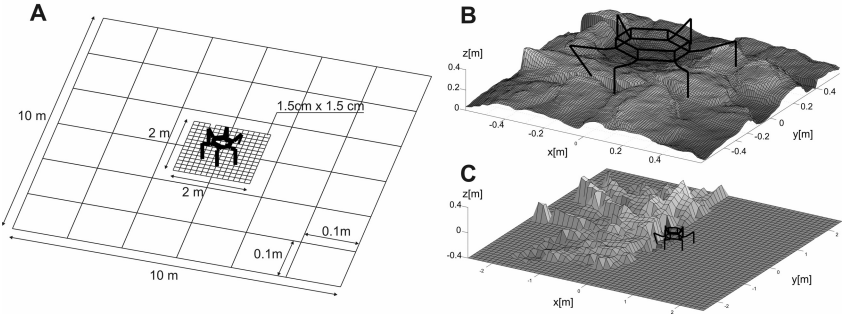


Fig. 10.2 Dimensions of local and global maps (A), example of a local map obtained by using Hokuyo laser rangefinder (B) and a global map obtained by using stereo camera (C)

The majority of path searching methods are based on grid search [9]. Typically, the search space is divided with a finite resolution. The way to overcome the problem of search space division is to use a method known as Probabilistic Roadmap (PRM) [6]. In this method the nodes are randomly located in configuration space. When the graph is created a standard graph search is performed. Another sampling-based motion planning method is the Rapidly-exploring Random Tree [8]. The most important property of this method is the ability to quickly explore the solution space. During searching the algorithm automatically modifies the speed of exploration.

The proposed approach to initial path planning uses coefficients describing the cost of locomotion. A similar method was presented by De Sanctis et al. [12]. The method calculates a weight matrix based on the gradient of each point over the 3D surface, the spherical variance, and the robot limitations. This matrix is used to limit the speed propagation of the Fast Marching wave in order to find the best path.

The method presented here involves two path planning algorithms. They are used for exploring two different grids. The grid obtained from the stereo camera is less precise than the grid obtained by using the Hokuyo rangefinder. A method presented by Kirby et al. [7] also uses a variable grid-based map, whose resolution decreases with the distance from the robot. It allows using an optimal heuristic path planner A* in real-time planning in an environment that has moving obstacles.

10.2 Motion Planning Strategy

The GUIDEDRRT algorithm, which is presented in Fig. 10.3, finds and executes a motion sequence between the start (q_{init}) and the goal (q_{goal}) positions of the robot. First, the ASTARFIND procedure finds the general path A_{path} between the current and the goal positions. Then the RRT-based planner follows this general path with a sequence of elementary motions. The procedure CREATETEMPGOAL finds a node q_{temp} on the A_{path} which is located at rrt_{distance} [m] from the current robot position – slighty farther than the reach of the precise local map. With the local map of size 2×2 m, rrt_{distance} is initially set to 1.2 m. Next, the procedure RRTCONNECT searches for a path between the q_{curr} and q_{temp} nodes.

If the search is successful the robot executes the obtained path, but only to the border of the known local map q_m. Hence, the RRT-based planner does not need to track A_{path} precisely, but rather can approach it asymptotically, making detours if necessary. If no appropriate path to the temporary goal is found, the algorithm allows a bigger detour from A_{path} by increasing $rrt_{distance}$, which moves the temporary goal closer to the global goal (q_{goal}). An increment of 0.2 m is used, which is a distance greater than the cell size in the global map. The RRTCONNECT procedure works on two different maps. If the node is located inside the local map the robot plans its motion precisely. If the node is outside the local map the motion is planned by using a simpler method that roughly determines traversability between two nodes.

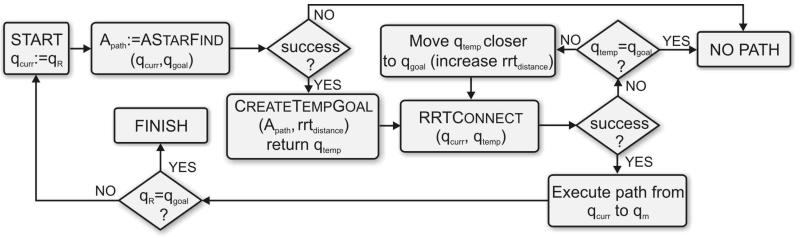


Fig. 10.3 Motion planning strategy – the GUIDEDRRT algorithm

While walking, the robot acquires new data which are stored in the local and the global maps. The whole procedure of path planning (A* and then RRT-Connect) is repeated on the new set of data. The GUIDEDRRT algorithm ends when the robot reaches the goal position or the procedure RRTCONNECT fails during searching for a path between q_{curr} and q_{goal}. The procedures ASTARFIND and RRTCONNECT are further described in detail.

10.2.1 Long Range Planning

The procedure ASTARFIND performs path planning by using the global map obtained from the stereo camera. The A* algorithm is chosen for this purpose. Nodes are located in the center of the grid cells so the distance between two nodes is 0.1 m. It is sufficient for rough path planning. With a map of this resolution A* is still fast enough for our purposes and it allows obtaining a general direction of further motion, i.e. guiding the RRT motion planning. The A* algorithm minimizes the function

$$f(x) = g(x) + h(x). \tag{10.1}$$

The $h(x)$ component is the predicted cost of the path between the position x and the goal node. The Euclidean distance is used to define the $h(x)$ heuristic. The $g(x)$ component is the cost of the path between the initial node and the x node. To compute $g(x)$ a transition cost between the neighboring nodes is defined. To do so three coefficients are used.

The first coefficient c_1 defines the Euclidean distance between the neighboring nodes. It is divided by 0.25 to normalize its value. The greater value of c_1 coefficient the greater cost of transition between the nodes. Example terrain and corresponding c_1 coefficients are shown in Fig. 10.4A and Fig. 10.4B, respectively.

The spherical variance ω [12] is used to define the second coefficient c_2. The grid map is converted to a triangle mesh by using the considered point and its neighbors (the number of triangles n is 8). The vector normal to each triangle on the surface $\overrightarrow{\mathbf{N}_i} = (x_i, y_i, z_i)$ is computed. Then the module R of the vector set $\overrightarrow{\mathbf{N}_i}$ is computed as follows:

$$R = \sqrt{(\sum_{i=0}^{n} x_i)^2 + (\sum_{i=0}^{n} y_i)^2 + (\sum_{i=0}^{n} z_i)^2}. \tag{10.2}$$

The module R is divided by the number of vectors n to normalize the values into a range between 0 and 1. Finally, the spherical variance is computed as follows:

$$c_2 = \omega = 1 - \frac{R}{n}. \tag{10.3}$$

The value of c_2 defines a "roughness" of the terrain. The higher the value the greater difficulties the robot has in finding appropriate footholds. The c_2 coefficient is computed for a destination point while defining the cost of transition between two nodes. The c_2 coefficients for an example terrain are shown in Fig. 10.4C.

To compute the final coefficient c_{final} the kinematic abilities of the robot to traverse between two nodes are also considered. The computation is simplified. The roll, pitch, and yaw angles of the trunk are set to zero and the robot places its feet on the nominal footholds. Finally, if the goal state is acceptable and a transition between the initial and the goal states is possible, the c_{final} coefficient is computed as the mean value of the c_1 and c_2 coefficients. If the transition is not achievable the final cost c_{final} is set to infinity. The final coefficient values are shown in Fig. 10.4D.

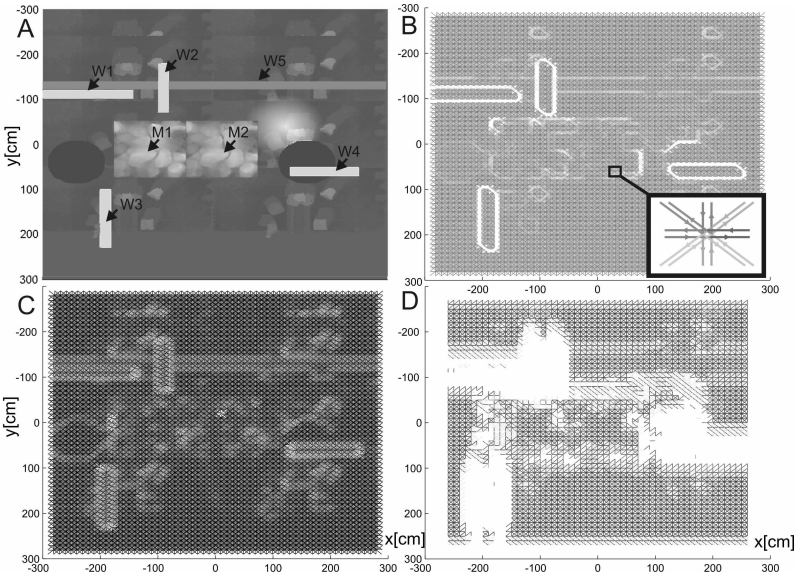


Fig. 10.4 Elevation map of the terrain (A), distance (B), variance (C), and final coefficient (D) (white – high value of coefficient, black – low value of coefficient). The inserted image in subfigure B shows a detail of the c_1 costmap – example transitions with costs between the neighboring cells

10.2.2 Precise Path Planning

To find an appropriate sequence of steps for the walking robot an algorithm which works in a continuous space should be used. Graph methods and A* search are not acceptable. The RRT planner is chosen because of its efficiency with high dimensional problems [13]. There are a number of RRT based planners known from literature. The original RRT algorithm [10] greedily explores the solution space but further versions of this algorithm have improved efficiency and computation time. Some improvements consider a bias of the direction to the goal or Voronoi bias [11]. In this case the property that "pulls" the search tree toward the unexplored areas of the state space is not efficient.

Finally the RRT-Connect [8] is used. In this algorithm two trees are created. The root of the first one is located in the goal position and the second one is located in the initial position. The trees are alternately expanded to random direction and to the direction of the second tree. This strategy makes the algorithm very efficient and feasible for purposes of precise path planning.

The RRT path planner operates on two grid maps with different resolution. If the node is located on global map with low resolution, the algorithm uses simple kinematic checking procedure to verify if a transition between two states is possible. The RRT algorithm allows precise planning motion of the robot on the local, high resolution map (select footholds, feet path planning etc.).

The most important operation in the RRT-Connect algorithm (RRTCONNECT procedure) is the EXTEND operation [1]. The EXTEND procedure tries to add a new node to the existing tree. It creates the robot's configuration at the given (x, y) location on the map. It computes the elevation and inclination of the robot's trunk. The EXTEND procedure also considers the kinematic constraints. It checks if the movement is possible to execute. It verifies the configuration space and also detects collisions. There is also a module for foothold selection [3] and path planning of the robot's feet. All software modules are described in detail in [1].

The foothold selection algorithm uses an analytical relation between the coefficients which are some simple geometric characteristics of the terrain, and the predicted slippage of the robot's feet [3]. The predicted slippage is a slippage which is expected when the robot chooses the considered point of the elevation map. Before the normal operation stage the robot learns unsupervised and offline how to determine the footholds. Then it exerts the obtained experience to predict the slippage and minimize it by choosing proper points from the map. The RRT planner uses the obtained analytical relation to asses potential footholds.

Each point of the planned path should be checked if it is safe and possible to execute. There are three criteria. The first one is the static stability criterion. If the robot is statically stable, then a projection of the center of the robot's mass on the plane is located inside the convex hull formed by the contact points of the legs being in the stance phase. If the robot is statically stable at each position along the path the algorithm checks if each point of the path is inside the robot's workspace. Finally the algorithm uses a CAD model of the robot to check the collision criterion.

10.3 Results

The described motion planning method has been verified in a realistic simulator [2] of the Messor robot (Fig. 10.1B). A specially prepared terrain model has been used. It includes obstacles similar to scattered flagstones, extensive hills with mild slopes, small and pointed hills, depressions and stones. This terrain model includes also a map of a mockup of a rocky terrain acquired using the URG-04LX sensor (M1 and M2 in Fig. 10.4A). Additionally, it includes a non-traversable hill, four non-traversable walls (W1–W4 in Fig. 10.4A) and one traversable wall (W5).

The obtained path is shown in Fig. 10.5A. The algorithm successfully found a path to the goal position. The robot avoided the non-traversable obstacles and also found a way how to deal with the traversable obstacles (mockup and traversable wall) (Fig 10.5B). The problem is solved in two stages – global planning gives a direction to the goal, and then precise planing yields a full plan for the robot's body.

The experiment presented in Fig. 10.5A shows some interesting properties of the algorithm. Region R1 was assessed as traversable by the A* planner. When the robot reached this area it turned out to be too difficult to find an appropriate path. The RRT algorithm found a new path which goes around the non-traversable area. The second interesting situation is related to the R2 region. The A* assessed it as a non-traversable one. The RRT planner found a path across this region and shortened the path. The experiment is shown in detail at `http://lrm.cie.put.poznan.pl/romoco2011.wmv`

Other interesting properties are shown in Fig. 10.6. The left-hand image presents a path found by using only the RRT-Connect algorithm. The right-hand image presents all RRT trees created by the presented algorithm. The original RRT algorithm searches over the whole map, which takes a lot of time. Also the final path is far from optimal. The RRT planning in the proposed approach is focused on exploring areas around the optimal solution given by the A* planner.

To show efficiency of the algorithm a series of three experiments was conducted. The obtained paths are shown in Fig. 10.7A. Although the algorithm is random-based the results are similar. The robot avoids the same untraversable areas, only

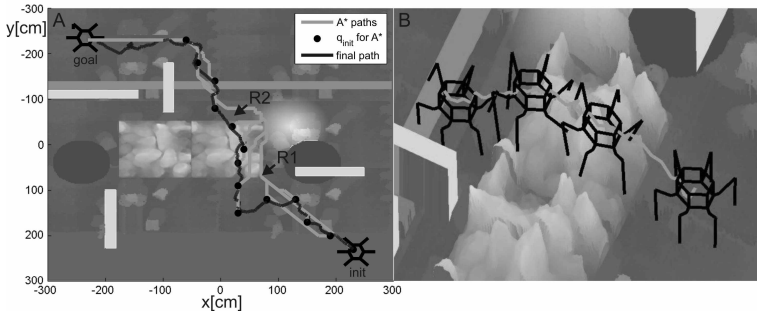


Fig. 10.5 The final path obtained by the guided RRT algorithm (A) and the local path obtained by RRT-connect algorithm (B)

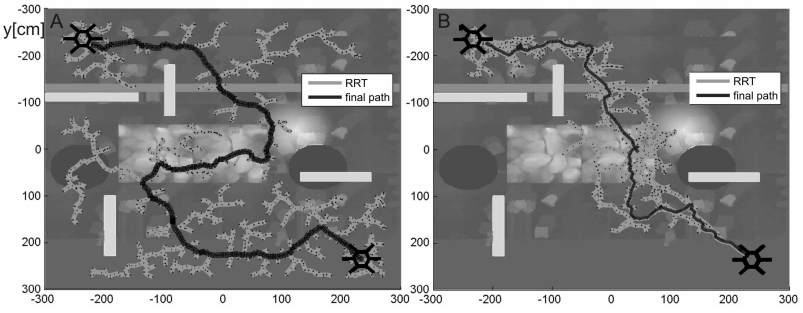


Fig. 10.6 Comparison between RRT-Connect (A) and guided RRT algorithms (B)

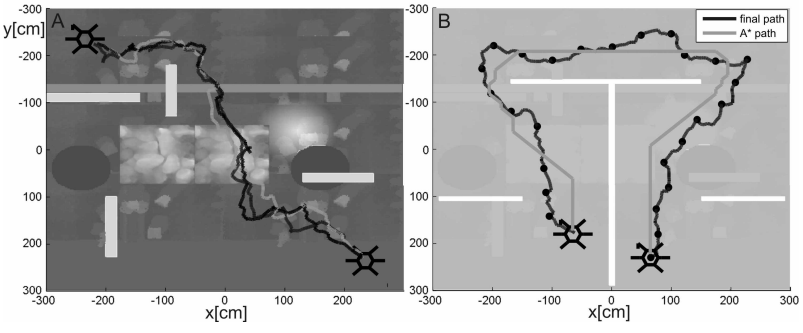


Fig. 10.7 Series of experiments (A) and experiment on a T-map (B)

the local paths are slightly different. Also an experiment on a synthetic map shown in Fig. 10.7B was conducted. This map includes a T-shaped obstacle. Such a map is often used to check if an algorithm can first lead the robot far from the obstacle (the distance to the goal increases), and finally reach the goal. The obtained path is shown in Fig. 10.7B.

10.4 Conclusions and Future Work

The guided RRT algorithm for motion planning of a walking robot is presented. It cooperates with the sensing system of the robot. The measurement system involves a stereo camera and a 2D laser rangefinder. The global map obtained by using the stereo camera allows planning a rough path of the robot whereas the local map allows selecting footholds and computing precise motion of its legs. The combination of A* and RRT-Connect algorithms makes it possible to obtain a feasible path for the walking robot. The algorithm is insensitive to inaccuracy of global path planning. When the precise map is created the robot can go around obstacles or traverse areas which were previously recognized as non-traversable.

The average execution of A* planning is 10 s on a machine with an Intel Core 2 Duo 2,53 GHz processor. The execution of RRT-Connect takes at least 10 s on the

distance of 1.2 m. When the ”roughness“ of the terrain increases the execution time also increases.

The local locomotion with the Hokuyo rangefinder was verified on the real robot Messor. Now the implementation of the whole system with a stereo camera, global and local path planning is being prepared.

Acknowledgements. Dominik Belter is a scholarship holder within the project ”Scholarship support for Ph.D. students specializing in majors strategic for Wielkopolska's development“, Sub-measure 8.2.2 Human Capital Operational Programme, co-financed by the European Union under the European Social Fund.

References

1. Belter, D., Skrzypczyński, P.: Integrated Motion Planning for a Hexapod Robot Walking on Rough Terrain. In: 18th IFAC World Congress (2011) (in print)
2. Belter, D., Skrzypczyński, P.: A Biologically Inspired Approach to Feasible Gait Learning for a Hexapod Robot. Int. Journal of Applied Math. and Comp. Sci. 20(1), 69–84 (2010)
3. Belter, D., Skrzypczyński, P.: Rough Terrain Mapping and Classification for Foothold Selection in a Walking Robot. In: Proc. IEEE Int. Workshop on Safety, Security and Rescue Robotics, CD-ROM, Bremen, Germany (2010)
4. Benoit, M., Rusu, R.B., Sundaresan, A., Hauser, K., Agrawal, M., Latombe, J.C., Beetz, M.: Leaving Flatland: Toward real-time 3D navigation. In: Proc. IEEE Int. Conf. on Robotics and Automation, Kobe, Japan, pp. 3786–3793 (2009)
5. Kalakrishnan, M., Buchli, J., Pastor, P., Mistry, M., Schaal, S.: Fast, robust quadruped locomotion over challenging terrain. In: Proc. IEEE Int. Conf. on Robot. and Autom., Anchorage, USA, pp. 2665–2670 (2010)
6. Kavraki, L.E., Svestka, P., Latombe, J.-C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. & Autom. 12(4), 566–580 (1996)
7. Kirby, R., Simmons, R., Forlizzi, J.: Variable sized grid cells for rapid replanning in dynamics environment. In: Proc. IEEE Int. Conf. on Intelligent Robots and Systems, St. Louis, USA, pp. 4913–4918 (2009)
8. Kuffner Jr., J.J., LaValle S. M.: RRT-connect: An efficient approach to single-query path planning. In: Proc. Int. Conf. on Robot. and Autom., San Francisco, pp. 995–1001 (2000)
9. Latombe, J.-C.: Robot Motion Planning. Kluwer, Norwell (1991)
10. LaValle, S.M., Kuffner, J.J.: Rapidly-exploring random trees: Progress and prospects. In: Donald, B.R., et al. (eds.) Algorithmic and Computational Robotics: New Directions, Wellesley, MA, pp. 293–308 (2001)
11. Lindemann, S.R., LaValle, S.M.: Steps Toward Derandomizing RRTs. In: Proc. 4th Int. Work. on Robot Motion and Control, Poznań, Poland, pp. 271–277 (2004)
12. Sanctis, L., Garrido, S., Moreno, L., Blanco, D.: Outdoor Motion Planning Using Fast Marching. In: Tosun, O., et al. (eds.) Mobile robotics: Solutions and Challenges, Istanbul, Turkey, pp. 1071–1080 (2009)
13. Sanchez, L.A., Zapata, R., Osorio, L.M.: Sampling-Based Motion Planning: A Survey. Computacion y Sistemas 12(1), 5–24 (2008)
14. Wooden, D., Malchano, M., Blankenspoor, K., Howard, A., Rizzi, A., Raibert, M.: Autonomous Navigation for BigDog. In: Proc. IEEE Int. Conf. on Robot. and Autom., Anchorage, USA, pp. 4736–4741 (2010)

Chapter 11
Improving Accuracy of Local Maps with Active Haptic Sensing

Krzysztof Walas

Abstract. Walking robots are supposed to be ubiquitous urban reconnaissance machines. To make them fully operational, accurate local maps of the environment are required. Here the local map is the local representation of the environment (obstacles), for example stairs. There are several perception systems which provide data used to create such representations. In this paper experiments with three of them are described: a Laser Range Finder, a Time of Flight Camera, and a Stereo Camera. The data provided by the above mentioned sensors is not fully reliable. Thus there is a need to augment the perception system with active haptic sensing, understood here as sensing the environment with the legs of the robot. At the beginning of the article the results of the experiments with the sensors mentioned above are described. Then the active sensing concept is presented in detail. At the end concluding remarks are given, followed by future work plans.

11.1 Introduction

Accuracy of local maps plays an important role in motion planning for walking robots. Here the local map is the local representation of the environment (obstacles), for example stairs [9]. Walking robots are supposed to be ubiquitous urban reconnaissance machines. Due to their exceptional mobility they are able to negotiate rough terrains such as debris or to climb obstacles such as curbs and stairs. All these tasks require good perception of the surroundings. The urban type of the environment is not only demanding in the type of the mobility required to negotiate obstacles, but also poses a huge problem for the perception system of the robot. Typical for this environment is presence of large regular surfaces, which in most cases are of uniform colour.

Krzysztof Walas
Institute of Control and Information Engineering, Poznań University of Technology,
ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: krzysztof.walas@put.poznan.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 137–146.
springerlink.com

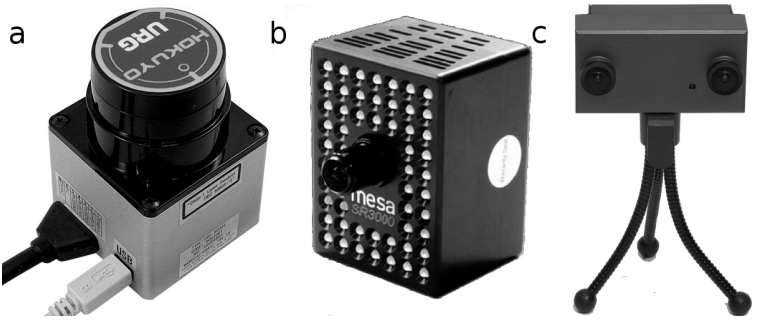


Fig. 11.1 Sensors: (a) Hokuyo URG-04LX, (b) SwissRanger SR-3000, (c) STOC camera

Mobile robots use many multimodal perception systems for building local representations of the environment. However, in the case of walking robots the mass of the used sensors is restricted. The robot has to carry all the devices on its body, and since the robotic legged locomotion is still not energy efficient the sensors have to be lightweight and their number has to be limited. Commonly used sensors are: Laser Range Finders (LRF), Time of Flight cameras (ToF), and Stereo Vision Systems (SVS). An example of a miniature LRF is Hokuyo URG-04LX (Fig. 11.1a). This type of sensor was used in [2] for rough terrain perception. A hybrid solution which comprises explicit visual information with range finder data is the SwissRanger SR-3000 (Fig. 11.1b), which was used in [13] also for rough terrain perception. A structured light system is another possibility to obtain the depth of the scene by using a vision system. Such a system was used for perceiving obstacles for a stair climbing algorithm [1]. Data fusion from an LRF and a structural light system was used to obtain a stair model and was presented in [9]. Information on the depth of the scene could also be gathered with a stereo camera (Fig. 11.1c). This kind of perception used for stair climbing is presented in [5].

According to the experiments presented in the related papers all perception systems described so far have some drawbacks when used in structured urban environment. To overcome the drawbacks of the sensors under discussion and to improve the robustness of map-based motion planning, active haptic sensing is used. The strategy of sensing the environment with the legs of the robot is based on the behaviours observed in nature, as described in [3], where an elevator reflex and a search movement were characterized. The first implementation of active haptic sensing on a robot is described in [7]. Some recent research on this topic is presented in [6], where the leg of a robot was used for haptic terrain classification.

The content of the article is as follows. At the beginning inaccuracies of several perception systems are described. Then the concept of active haptic sensing is presented. At the end concluding remarks are given, followed by future work plans.

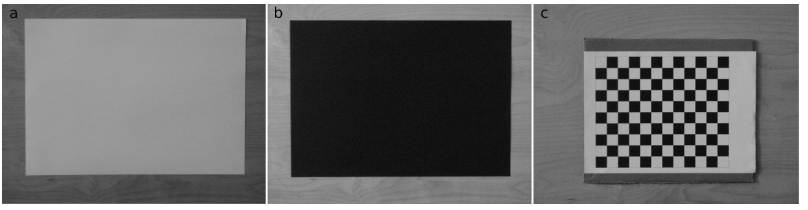


Fig. 11.2 Testing set: (a) white cardboard, (b) black cardboard, (c) chessboard

11.2 Inaccuracies of Walking Robot Perception Systems

The list of perception system for building local representations of the environment is long. However, in the case of a walking robot which is able to carry a limited load the list is rather short. The sensors have to be miniature and lightweight. The devices used in the experiments were LRF, ToF Camera, and Stereo Vision Camera. The mentioned sensors were tested in an urban environment where the surroundings of the robot consist of large surfaces with a uniform colour. This type of the environment is challenging for the mentioned perception systems and reveals their shortcomings. For the experiment three possible scenarios were proposed. The first one is a uniformly painted white cardboard (Fig. 11.2a). The second one is a light absorbent uniformly painted black cardboard (Fig. 11.2b). The last one is a cardboard with a texture which is the black and white chessboard (Fig. 11.2c). The two extremes, black and white, were chosen to show the maximum possible errors while working in the urban environment. The chessboard was used in order to show how the sensors cope with perception of surfaces with patterns. The experiments were performed for each device on the whole testing set.

11.2.1 Laser Range Finder

The sensor used for the experiments was Hokuyo URG-04LX. The characteristics of the device are described in [11]. The colour and the light absorption characteristics of the material greatly influence depth measurements. In the experiment we were trying to check whether distance measurements depend on the colour of the surface and on the distance to the obstacle. The experiment was performed for the distances of 155 mm and 990 mm and its results are presented in Table 11.1 and Fig. 11.3. The measurement error was assumed to have Gaussian distribution according to [11].

Five series of measurements for each distance were performed. For 155 mm experiment each series consisted of 86 samples, and for 990 mm – of 51 samples. For the longer distances (990 mm) the obtained values of errors are less than 7%. The highest error occurs for the black cardboard which absorbs much light. The measurements for the chessboard have the highest standard deviation. Due to the drastic and sharp change of colour virtual edges are observed. The closer to the obstacle the worse. For the distance of 155 mm and the black cardboard the value of error is almost 29%. Even for the white cardboard the error is significant. The measurements

Table 11.1 Distance measurements using Hokuyo URG-04LX

Measurements	Mean value [mm]	Standard dev.[mm]	Error [mm]	Error [%]
white (155 mm)	163.22	3.99	8.22	5.30
black (155 mm)	110.14	3.20	−44.86	−28.94
chess (155 mm)	134.25	14.00	−20.75	−13.39
white (990 mm)	980.10	3.99	−9.90	−1.00
black (990 mm)	926.50	3.23	−63.50	−6.41
chess (990 mm)	948.24	21.80	−41.76	−4.22

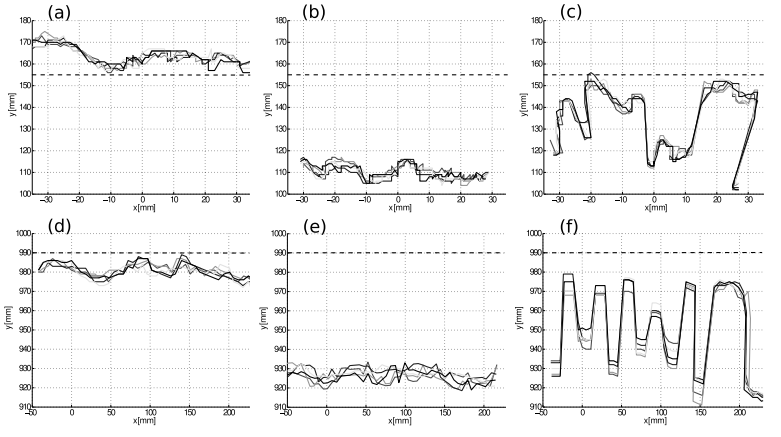


Fig. 11.3 Distance measurements using Hokuyo URG-04LX: from 155 mm (a) white cardboard, (b) black cardboard, (c) chessboard; from 990 mm (d) white cardboard, (e) black cardboard, (f) chessboard. Each line on the graph represents one series of measurements. The dashed line indicates the true value.

from the sensor could be slightly improved by using adaptive gain control for the emitted light according to the power of the reflected light. This is, however, insufficient for the short distance measurements, where the scanner exhibits non-linear behaviour.

11.2.2 Time of Flight Camera

The device used in the experiment was SwissRanger SR-3000. The description of the sensor can be found in [4]. The authors outline problems with accuracy of measurements caused by changes of intensity of light reflected from different surfaces. Our research acknowledges those observations. In our experiment the measurements were performed from the distance of 300 mm and 1040 mm. The results of the measurements are presented in Table 11.2 and Fig. 11.4. The measurement error was assumed to have Gaussian distribution according to [12].

Table 11.2 Distance measurements using SwissRanger SR-3100

Measurements	Mean value [mm]	Standard dev.[mm]	Error [mm]	Error [%]
white (300 mm)	238.21	17.07	-61.79	-20.60
black (300 mm)	262.91	19.44	-37.09	-12.36
chess (300 mm)	240.47	21.15	-59.53	-19.84
white (1040 mm)	1051.62	7.58	11.62	1.12
black (1040 mm)	1059.06	49.78	19.06	1.83
chess (1040 mm)	1018.95	35.75	-21.05	-2.02

For the smaller distance (300 mm) the number of samples is 25000 and for the larger distance (1040 mm) it is 1400. The distance of 300 mm was chosen instead of 155 mm because for the latter it was impossible to perform reliable measurements. The ToF camera and the LRF exploit roughly the same physical phenomenon to obtain the depth value, so the results are similar. Better results are obtained for the larger distances, although for the ToF camera the gain of the emitted light is better adjusted so the differences between the measurements for the black and white cardboards are lower. However, the standard deviation is very high for the black surface. The SwissRanger adapts its gain value according to the power of the reflected signal. This improves the results of the measurements for the different colours, but causes problems when there is a texture, for example the chessboard. This is due to the fact that the gain adjustment is too slow. Thus for the chessboard the measurement error is the highest. Additionally for the short distance measurements the camera has problems with adjusting its gain and the data obtained is not reliable.

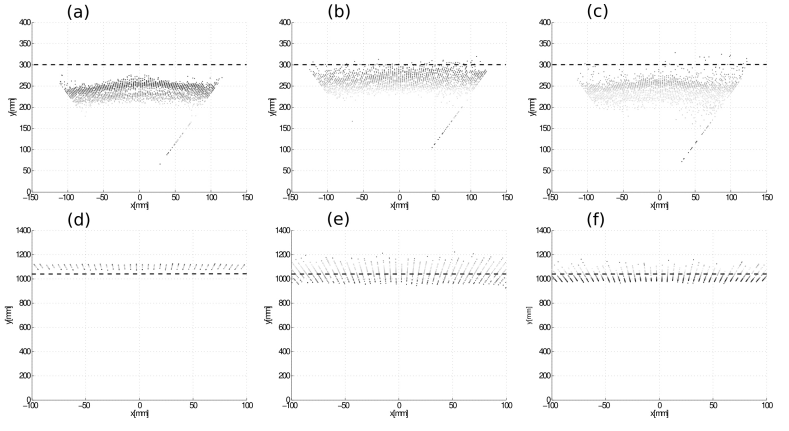


Fig. 11.4 Distance measurements using SwissRanger SR-3100: from 300 mm (a) white cardboard, (b) black cardboard, (c) chessboard; from 1040 mm (d) white cardboard, (e) black cardboard, (f) chessboard. Each graph presents a point cloud seen in the x-y plane. The dashed line indicates true value.

Fig. 11.5 Disparity images obtained using STOC camera (Videre Design) from 1000 mm: (a) white cardboard, (b) black cardboard, (c) chessboard. For the distance of 155 mm there is no information on the pictures while the whole field of view of the camera is covered by the uniform colour cardboard. There are no features for obtaining the disparity map.

11.2.3 Stereo Vision

The sensor used in this experiment is a STOC camera from Videre Design with a 6 cm base line. The depth perception is obtained by using the algorithm developed by Kurt Konolige and described in [8]. The algorithm requires dense features for building the disparity map. However, in the urban environment mainly large uniform surfaces are encountered, for example walls. In our experiment it was impossible to establish the distance to the cardboards with uniform colour, where there were no features for building a disparity map, although some features in the background could be distinguished. Only the chessboard gave some results. The outcome of the experiment is shown in Fig. 11.5.

Each of the investigated sensors has it flaws when tested in the urban environment. There exists one more possibility of scene perception which is the use of a structured light system. It gives accurate results [1]. Its main drawback is that the system is not operational in the presence of sunlight. Clearly, it is possible to use an infrared light source, but it requires mounting an additional camera on the robot, which will be responsible for this mode of sensing. This violates the tight load limit for the walking robot. It is clear that the data obtained from the tested sensors gives a rough map of the surroundings of the robot.

11.3 Active Perception Scheme

While the walking robot has only an approximate map of the environment it is required to augment the measurements with another sensor to make the surroundings model more accurate. The solution to that problem is active sensing, which is understood here as provoking collisions of some parts of the robot body – namely,

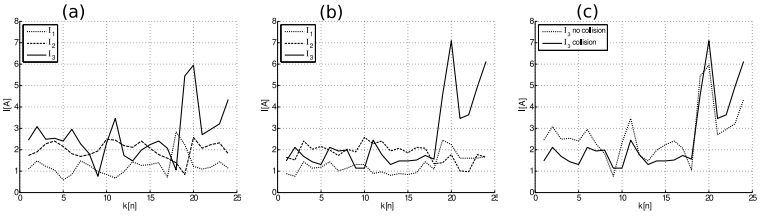


Fig. 11.6 The currents measured in the joints of the front leg of the robot: (a) the gait without a collision (b) gait with a collision (c) the change of the current in the third joint with and without collision. The joint no 1 is the closest to the trunk of the robot.

a probing front leg of the robot – with obstacles. To achieve this goal an efficient method for contact sensing is required. One possibility is to use the sensors which measure the current of the dc motor in each joint of the probing leg.

This approach was tested with the Messor, which is a biologically inspired six-legged walking robot. The trunk has the following dimensions: width 26 cm and length 30.6 cm, while the segments of the leg are: coxa 5.5 cm, femur 16 cm, tibia 23 cm. The machine weights 4.3 kg. The mechanical structure of the robot is shown in Fig. 11.8 and is described in more detail in [15].

For sensing a contact of the robot leg with an obstacle two types of sensors are used. The first one is mounted at the tip of the foot and provides information on the contact with the ground. It is a resistive sensor, where the resistance is decreasing with the increase of the contact force. The other type is current sensors attached to each motor of the leg joints.

The part of the collision sensing system consisting of the latter type need to be described in more detail. In order to obtain an indicator of the collision, a rule based system is applied to the current measurements. The plots of the currents in each joint of the robot leg (the first joint is the closest to the robot body) are shown in Fig. 11.6. As can be seen for currents I_1 and I_2 there is no significant difference between situations when a collision occurred and when it did not. By comparing Figs. 11.6(a) and 11.6(b) the collision with the obstacle is distinguishable when the measurements of I_3 are used. In Fig. 11.6(c) it can be seen that the currents in the third joint when a collision appears and when it does not differ by a value of 1 A (14%). This makes it possible to set a condition on I_3 for the collision detection rule. Nonetheless some information is still missing. The rise in the current value indicates a collision only when it is measured during the swing phase. Collision detection is normally used for behaviour based control. But in this case it is used in a different role – for active haptic sensing of the environment. In this application in the presence of a collision the leg is stopped. According to the kinematics of the leg and the knowledge of the joint angles the distance to the obstacle is calculated.

Now having the information from the external sensors (LRF, SVS) and from active haptic sensing the perception scheme is proposed, see Fig. 11.7. The system is divided into two subsystems: one for map building and one for map refining. Map building is performed with external sensors, when they are working in their

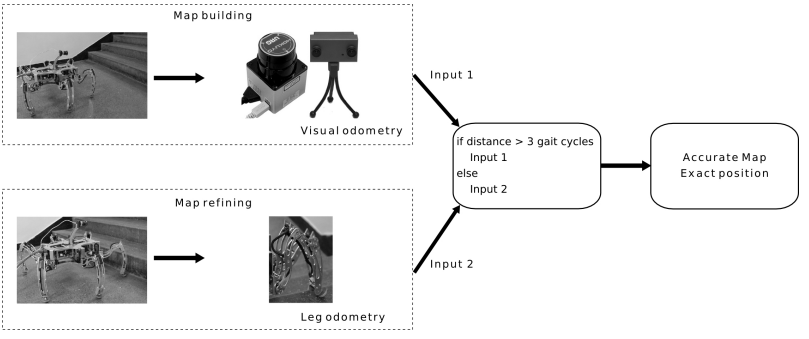


Fig. 11.7 Active perception scheme

accurate range (> 300 mm), whereas map refining is performed with active haptic sensing and is done for short distances to obstacles (≤ 300 mm – approximately 3 gait cycles). The distances were established according to the experiments presented in Section 11.2.

Each of the modules requires good odometry to localize the robot with respect to the obstacle. The value of the relative error of pose estimation ought to be less than 5% to give reliable results. For long range measurements, visual odometry is used. Here the Iterative Closest Point (ICP) method is used on the depth images obtained from a stereo pair. This type of odometry gives good results for obstacles which are far from the robot, as presented in [10]. For the experiments presented in [10] the computed mean percentage errors and corresponding standard deviations of the measurements were $1.9 \pm 2.2\%$ along x and $2.4 \pm 1.8\%$ along y. The experiments were performed on flat ground with an off-road rover Shrimp. However, visual odometry is imprecise when used for obstacles seen from short distances. Therefore for short range measurements odometry based on counting the steps of the robot and the length of the swing phase of the probing leg before hitting an obstacle is used. In conclusion, using two measurement subsystems for different ranges of measurements leads to obtaining an accurate local map and the exact position of the robot.

11.4 Example

One of the most challenging obstacles in the urban environment is stairs. A robust strategy for stairs negotiation requires a precise map of the environment. The strategy presented in [14] uses multisensor perception to achieve this goal [9]. Even though the robot managed to climb the stairs, the structured light system used in the experiment was hard to calibrate and tended to decalibrate from time to time. To make this algorithm more robust to the small imprecisions of the measurement system, the active haptic sensing approach is required. The mentioned small difference means less than 1 cm, but as can be seen in Figs. 11.8(a) and 11.8(b) in the

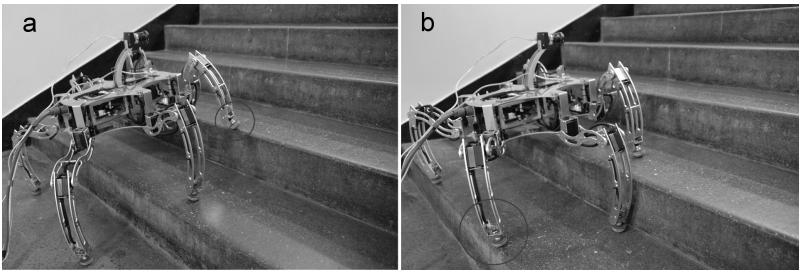


Fig. 11.8 Messor robot climbing stairs: (a) safe phase of the climb – placing front legs on the subsequent step, (b) dangerous phase of the climb – placing hind legs on the subsequent step

tight staircases it signifies failure of the climb. As can be observed in Fig. 11.8(a), when the robot is at the beginning of the climb, a small error in this stage of the strategy may not cause a problem – although inaccuracy accumulates and it may finally cause the lack of space for the hind legs of the robot in the situation presented in Fig. 11.8(b). Active sensing prevents the robot from accumulating the error while performing the climbing strategy.

11.5 Conclusions and Future Work

The article presents the perception systems for sensing local surroundings of a walking robot. As shown in the experiments, each of the tested sensors has its flaws when used in the urban environment. Thus a reliable local representation of the environment is hard to obtain. To make the distance measurements more precise when the robot is close to an obstacle active haptic sensing is used. The robot senses its environment with its front legs and in the case of collision provides information about the position of the obstacle. The active sensing is performed by using contact sensors on the tip of the leg to establish contact with the ground and current sensors in the dc motors of the joints to sense contact with the obstacles. The combination of visual sensing and active haptic sensing allows fast and robust motion of the robot. It exploits advantages of both approaches, namely the speed of visual sensing and the precision of active haptic sensing. The robot uses its legs for sensing only when the data from the other system is not reliable. The proposed approach improves the robustness of the motion strategies for the walking robot.

In the future a thorough investigation of visual odometry systems is required. Moreover, tests of the presented approach in other special cases are needed.

Acknowledgements. The author would like to thank A. Rönnau for performing the experiments with the Time of Flight Camera.

References

1. Albert, A., Suppa, M., Gerth, W.: Detection of stair dimensions for the path planning of a bipedal robot. In: Proc. IEEE Int. Conf. on Adv. Intell. Mechatron., Como, Italy, pp. 1291–1296 (2001)
2. Belter, D., Skrzypczyński, P.: Rough Terrain Mapping and Classification for Foothold Selection in a Walking Robot. In: Proc. IEEE Int. Workshop on Saf. Secur. and Rescue Robotics (SSRR 2010), Bremen, Germany. CD-ROM (2010)
3. Flannigan, W.C., Nelson, G.M., Quinn, R.D.: Locomotion controller for a crab-like robot. In: Proc. IEEE Int. Conf. on Robot. and Aut. (ICRA 1998), Leuven, Belgium, pp. 152–156 (1998)
4. Guomundsson, S.A., Aanaes, H., Larsen, R.: Environmental Effects on Measurement Uncertainties of Time-of-Flight Cameras. In: Proc. Int. Symp. on Signals, Circuits and Syst. (ISSCS 2007), Iasi, Romania, pp. 1–4 (2007)
5. Gutmann, J.-S., Fukuchi, M., Fujita, M.: Stair climbing for humanoid robots using stereo vision. In: Proc. IEEE Int. Conf. on Intell. Robots and Syst. (IROS 2004), Sendai, Japan, pp. 1407–1413 (2004)
6. Hoepflinger, M., Remy, C., Hutter, M., Spinello, L., Siegwart, R.: Haptic Terrain Classification for Legged Robots. In: Proc. IEEE Int. Conf. on Robot. and Aut. (ICRA 2010), Anchorage, Alaska, pp. 2828–2833 (2010)
7. Hoffman, R., Krotkov, E.: Perception of rugged terrain for a walking robot: true confessions and new directions. In: Proc. IEEE/RSJ Int. Workshop on Intell. Robots and Syst. (IROS 1991), Osaka, Japan, pp. 1505–1510 (1991)
8. Konolige, K.: Small vision system: Hardware and implementation. In: Proc. Int. Symp. on Robotics Research, Hayama, Japan, pp. 111–116 (1997)
9. Łabęcki, P., Walas, K.: Multisensor perception for autonomous stair climbing with a six legged robot. In: Proc. Int. Conf. on Climbing and Walk. Robots and the Support Technol. for Mob. Mach. (CLAWAR 2010), Nagoya, Japan, pp. 1013–1020 (2010)
10. Milella, A., Siegwart, R.: Stereo-Based Ego-Motion Estimation Using Pixel Tracking and Iterative Closest Point. In: Proc. IEEE Int. Conf. on Comput. Vis. Syst. (ICVS 2006), Washington, DC, USA, p. 21 (2006)
11. Okubo, Y., Ye, C., Borenstein, J.: Characterization of the hokuyo urg-04lx 2d laser rangefinder for mobile robot obstacle negotiation. In: Unmanned Syst. Technol. XI, SPIE Proc. 7332 (April 2009)
12. Oprisescu, S., Falie, D., Ciuc, M., Buzuloiu, V.: Measurements with ToF Cameras and Their Necessary Corrections. In: Proc. IEEE Int. Symp. on Signals Circuits and Syst. (ISSCS 2007), Iasi, Romania (2007)
13. Roennau, A., Kerscher, T., Ziegenmeyer, M., Zöllner, J.M., Dillmann, R.: Adaptation of a six-legged walking robot to its local environment. In: Kozłowski, K.R. (ed.) Robot Motion and Control 2009. LNCIS, vol. 396, pp. 155–164. Springer, Heidelberg (2009)
14. Walas, K.: Fully parametrized stair climbing strategy for a six-legged walking robot. In: Proc. Int. Conf. on Climb. and Walk. Robots and the Support Technol. for Mob. Mach. (CLAWAR 2010), Nagoya, Japan, pp. 777–784 (2010)
15. Walas, K., Belter, D.: Messor – Versatile Walking Robot for Search and Rescue Missions. J. of Autom., Mob. Robotics & Intell. Sys. 5(2), 28–34 (2011), http://www.jamris.org

Chapter 12
Postural Equilibrium in Two-Legged Locomotion

Teresa Zielińska

Abstract. The method of two-legged gait synthesis for the biped robot is introduced. The problem of dynamic postural equilibration taking into account the role of compliant feet is formulated. The equilibrium conditions are split to feet attachment points and points within the feet-end area. The presented method is important for dynamical motion synthesis taking into account the robot parameters. The method was validated using simulations and experiments.

12.1 Introduction

Much work on multi-legged walking machines has addressed design problems and motion generation principles of statically stable locomotion (i.e. [4, 6]). Control aspects have also been discussed [8], but analysis of dynamical gaits equilibrium conditions considering stabilizing role of the foot is in its initial stages. The legs of multi-legged walking machines have typically 2 or 3 active degrees of freedom. The additional degrees of freedom (if introduced) are passive. The foot compliance is typically obtained using springs. Many multi-legged robots have their feet shaped as balls or as rotating plates – Fig. 12.1a. They are attached to the shank by passive prismatic joints. More complex designs consist of 3 passive DOFs [1] – Fig. 12.1b. The potentiometers are sometimes utilized as sensors for monitoring the joints position. Only recently has attention focused on biped foot design using biological analogies. The biologically inspired foot with three fingers and 2 active DOFs – Fig. 12.1c – is one of such examples [3].

Fig. 12.1d presents the recently developed foot of the Wabian-2R robot. This foot acts to some extent in a similar way to the human foot [2]. In gait synthesis attention is paid to positioning of active joints. The role of a foot and its passive

Teresa Zielińska
Institute of Aeronautics and Applied Mechanics, Warsaw University of Technology (WUT–IAAM), ul. Nowowiejska 24, 00-665 Warsaw, Poland
e-mail: teresaz@meil.pw.edu.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 147–156.
springerlink.com

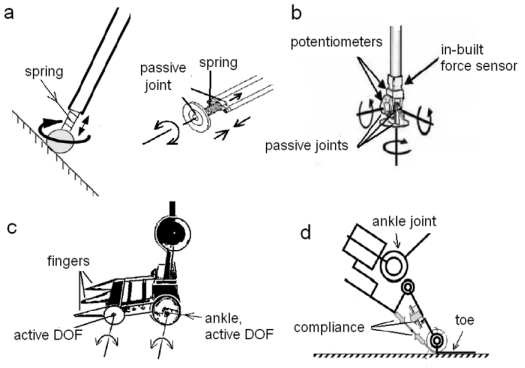


Fig. 12.1 Robot's feet: a) foot shaped as a ball and as a plate, b) leg-end joint with 3 DOF, c) scheme of a foot inspired by animal structure, d) leg-end of Wabian-2R robot [2] inspired by a human foot

DOFs during the walk of a multi-legged machine is often neglected. On the other hand the usefulness of passive joints and springs in walking machines have been confirmed by practical experience. In this manuscript we discuss the role of passive joints in maintaining the postural dynamical stability. The theoretical considerations are supported by simulation; the results were validated by experiments.

12.2 Problem Statement

During normal human gait the support phase takes 60% of the gait period; 40% of that is single support phase (support by one leg only) and 20% – double support phase (support by two legs). It is relatively easy to test postural stability in the single support phase but double support presents problems. Due to that the majority of early bipeds walked with almost no double support. Our aim was to create a robot walking as a human.

The foot and the robot considered in our work is illustrated in Fig. 12.2. The robot has a human leg segments proportion. Foot design was inspired by the property of the human foot, where the bone arcs act as the compliant elements. During the support when the foot is loaded the sagittal arc is compressed along vertical direction, during the take-off the vertical load to the arc is relaxed but the compression force acts along the sole – see Fig. 12.3. To introduce the vertical compliance typical for

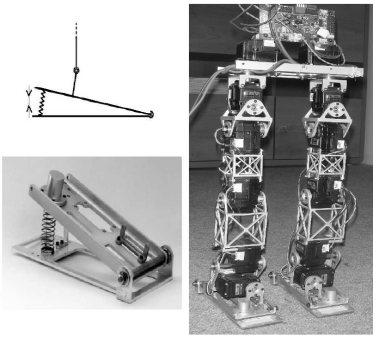


Fig. 12.2 Left: spring-loaded foot, right: robot with spring-loaded foot

a human, we applied a vertical spring mounted next to the ankle joint. In the support phase the spring length changes proportionally to the acting vertical force. This change is small but it influences the postural equilibrium as will be discussed later.

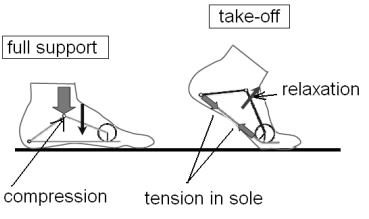


Fig. 12.3 Work of a foot arc in the support phases (according to [2])

Let us assume that $OXYZ$ is the non-moving reference frame, $RXYZ$ is the frame attached to the robot trunk – Fig. 12.4.

The coordinates of the robot mass center ${}^{R}(xyz)_{CG}$ are equal to:

$$ {}^{R}(xyz)_{CG} = \frac{m_o \, {}^{R}(xyz)_o + \sum_i \sum_j m_{ij} \, {}^{R}(xyz)_{ij}}{m}, \tag{12.1} $$

where ${}^{R}(xyz) = \{{}^{R}x, {}^{R}y, {}^{R}z\}$, m_0 – mass of trunk, $m = m_0 + \sum_i \sum_j m_{ij}$ – total mass. The robot prototype (Fig. 12.2) is 0.3295m tall, with the length of the thigh equal to 0.10m and the shank with the foot – 0.145m. The mass of the robot is 1.61kg, with the thigh mass equal to 0.16kg, the shank with the foot – 0.34kg, of which the foot is 0.10kg. The distance between the ground projection of the robot ankle and the rear of the foot is 0.041m. The point mass of the thigh is located below the hip joint at a distance equal to half of the thigh length. The point mass of the shank considered together with the foot is located below the knee at a distance equal to 83.7% of the shank segment length. The legs are about 3.7 times shorter than in a human and the mass proportion of the leg segments is similar to that in a human.

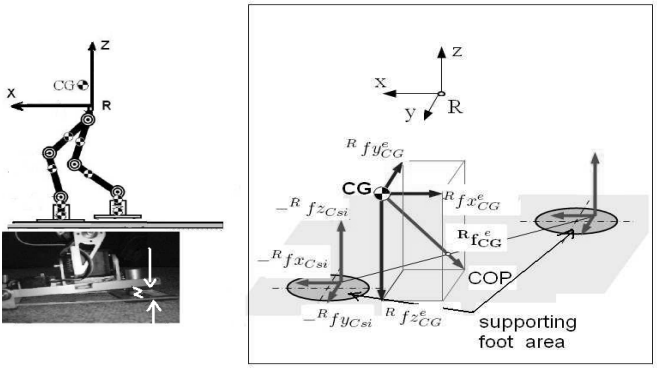


Fig. 12.4 Applied notation

Following the similarities between a human and the robot for the reference gait the human gait pattern was proposed.

12.3 Postural Stability in Single Support Phase

Before implementing the motion pattern in the real robot, postural equilibrium was studied by simulation. For testing the equilibrium in single support phase the ZMP criterion was applied [9]. The details regarding ZMP formulation can be found in many publications – e.g. [5]. The following formula produces coordinates x_{ZMP}, y_{ZMP} of point F_{ZMP}:

$$x_{ZMP} = \frac{\sum_i^n m_i(\ddot{z}_i - g)x_i - \sum_i^n m_i\,\ddot{x}_i\,z_i - \sum_i^n I_i^y\,\ddot{\alpha}_i^y}{\sum_i^n m_i(\ddot{z}_i - g)}, \tag{12.2}$$

$$y_{ZMP} = \frac{\sum_i^n m_i(\ddot{z}_i - g)y_i - \sum_i^n m_i\,\ddot{y}_i\,z_i - \sum_i^n I_i^x\,\ddot{\alpha}_i^x}{\sum_i^n m_i(\ddot{z}_i - g)}, \tag{12.3}$$

where m_i is the mass of the i-th body part, x_i, y_i, z_i are the coordinates of the mass centre of the $i-$th part expressed in the frame attached to the trunk, $\ddot{x}_i$, $\ddot{y}_i$ $\ddot{z}_i$ are accelerations of those points with respect to that frame, I_i^x, I_i^y are the main moments of inertia for the $i-$th body part about X and Y axes, and $\ddot{\alpha}_i^x$, $\ddot{\alpha}_i^y$ are the angular accelerations about those axes, g is the gravity constant.

Based on the obtained ZMP trajectories it was decided that the human gait needed small adjustments for closer similarity between human and robot ZMPs. The ranges of the hip joint trajectories were shifted towards the positive direction reducing the leg-end backward shift at the end of the support phase and increasing the forward shift at the beginning of the support phase. In the knee a small bend during the support phase was added. With this modification the resultant time course of the

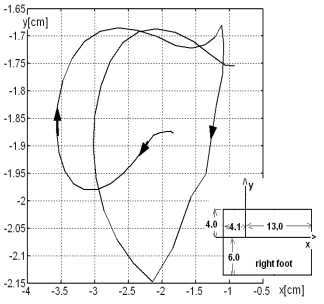


Fig. 12.5 ZMP trajectory for human gait evaluated using human body model with consideration of trunk inclination (the foot frame is shown)

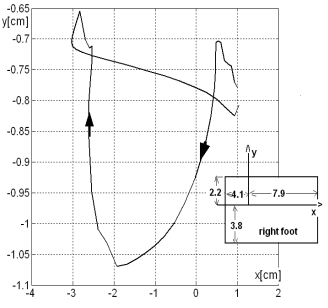


Fig. 12.6 ZMP trajectory for modified robot gait with trunk inclination (the foot frame is shown)

ZMP is similar to that observed in human walk – compare Figs. 12.5 and 12.6. The obtained gait was applied to the prototype and its stable walk confirmed the correctness of the considerations. After successful implementation of the gait the robot's feet were replaced by the modified ones, with in-build springs – Fig. 12.2.

12.4 Stability in Double Support Phase

The postural equilibrium conditions were proposed taking into account the modified foot. It should be noted that due to the closed kinematic chain created by the two supporting legs the ZMP criterion has no meaning for stability evaluation in the double support phase (ZMP is located outside the foot-prints). The lack of unique conditions indicating the stability in the double support phase creates a problem for biped gait synthesis.

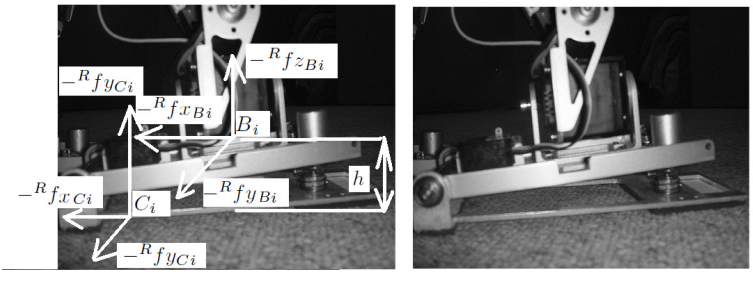


Fig. 12.7 Reaction forces related to the foot, for better illustration on the right the foot is shown

The equilibrium condition formulated for double support phase is based on splitting equilibrium formulas between point B, which is the shank attachment to the foot, and point C, which is the point in the sole where resultant leg-end reaction force vector is applied. By lower script i (e.g. ${}^{R}F_i$) we denote the quantities related to the supporting foot; by adding letter B or C we denote in which point the equilibrium is considered. The force equilibrium conditions are expressed in local frame $RXYZ$:

$$\sum_i {}^{\mathbf{R}}\mathbf{f_i} = \mathbf{F_e} + m({}^{\mathbf{R}}\ddot{\mathbf{r}}_{\mathbf{CG}} + {}^{\mathbf{R}}\mathbf{g}) = \mathbf{F_e} + m_0 {}^{\mathbf{R}}\mathbf{g} + \sum_i \sum_j + m_{ij}({}^{\mathbf{R}}\ddot{\mathbf{r}}_{\mathbf{ij}} + {}^{\mathbf{R}}\mathbf{g}) =$$
$$= \mathbf{F_e} + {}^{\mathbf{R}}\mathbf{f_{CG}} = [{}^{R}fx^{e}_{CG}, {}^{R}fy^{e}_{CG}, {}^{R}fz^{e}_{CG}]^{T}, \tag{12.4}$$

where ${}^{\mathbf{R}}\mathbf{g} = {}^{R}T_O[0,0,-g]^T$, g is the gravity constant, ${}^{R}T_O$ is transformation matrix from $OXYZ$ to $RXYZ$, ${}^{\mathbf{R}}\mathbf{f_{CG}}$ is the resultant force vector acting to the robot mass centre, ${}^{\mathbf{R}}\mathbf{F_i}$=$[{}^{R}fx_i, {}^{R}fy_i, {}^{R}fz_i]^T$ is the force vector exerted by the leg-end – Fig. 12.7. The torque equilibrium conditions in frame $RXYZ$ are described by:

$$\sum_i ({}^{\mathbf{R}}\mathbf{r_i} - {}^{\mathbf{R}}\mathbf{r_{CG}}) \times {}^{\mathbf{R}}\mathbf{f_i} = \sum_i ({}^{\mathbf{R}}\mathbf{r_i} - {}^{\mathbf{R}}\mathbf{r_{CG}}) \times (\mathbf{F_e} + m({}^{\mathbf{R}}\ddot{\mathbf{r}}_{\mathbf{CG}} + {}^{\mathbf{R}}\mathbf{g})) =$$
$$= \sum_i ({}^{\mathbf{R}}\mathbf{r_i} - {}^{\mathbf{R}}\mathbf{r_{CG}}) \times (\mathbf{F_e} + {}^{\mathbf{R}}\mathbf{f_{CG}}) = [-{}^{R}M^{e}_{x}, -{}^{R}M^{e}_{y}, -{}^{R}M^{e}_{z}]^{T}. \tag{12.5}$$

${}^{R}M^{e}_{x}, {}^{R}M^{e}_{y}, {}^{R}M^{e}_{z}$ are the external moments applied to the robot. In our considerations we assume only the rotation along vertical axis Z passing point R, I_{zz} is the main inertia moment around axis Z, and φ_Z is the rotation angle. With the above assumption it is $-{}^{R}M^{e}_{x} = 0$, ${}^{R}M^{e}_{y} = 0$, ${}^{R}M^{e}_{z} = I_{zz}\ddot{\varphi}_z$. Shortening, we denote $\mathbf{F_e} + {}^{\mathbf{R}}\mathbf{f_{CG}}$ by ${}^{\mathbf{R}}\mathbf{f^{e}_{CG}}$.

$$\mathbf{A}\,\mathbf{f} = \mathbf{F}, \tag{12.6}$$

where

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & -{}^{R}z_1 & {}^{R}y_1 & 0 & -{}^{R}z_2 & -{}^{R}z_2 \\ {}^{R}z_1 & 0 & -{}^{R}x_1 & {}^{R}z_2 & 0 & -{}^{R}x_2 \\ -{}^{R}y_1 & {}^{R}x_1 & 0 & -{}^{R}y_2 & {}^{R}x_2 & 0 \end{bmatrix},$$
$$\mathbf{f} = [{}^{R}fx_1, {}^{R}fy_1, {}^{R}fz_1, {}^{R}fx_2, {}^{R}fy_2, {}^{R}fz_2]^T,$$

$$\mathbf{F} = [{}^{R}fx^{e}_{CG}, {}^{R}fy^{e}_{CG}, {}^{R}fz^{e}_{CG}, {}^{R}M_x, {}^{R}M_y, {}^{R}M_z]^T,$$
$${}^{R}M_x = -{}^{R}fy^{e}_{CG}\,{}^{R}z_{CG} + {}^{R}fz^{e}_{CG}\,{}^{R}y_{CG},$$
$${}^{R}M_y = {}^{R}fx^{e}_{CG}\,{}^{R}z_{CG} - {}^{R}fz^{e}_{CG}\,{}^{R}x_{CG},$$
$${}^{R}M_z = -{}^{R}fx^{e}_{CG}\,{}^{R}y_{CG} + {}^{R}fy^{e}_{CG}\,{}^{R}x_{CG} - I_{zz}\ddot{\varphi}_z. \tag{12.7}$$

Matrix $\mathbf{A}$ is singular ($\text{rank}(\mathbf{A}) < 6$), the rank of the extended matrix is 6, which means that the equalities cannot be fulfilled. This means that the equilibrium conditions described by (12.4), (12.5) cannot be fulfilled considering only the fixed points in the leg-ends (e.g. points B). Taking into account the stabilizing role of the feet we split the conditions between points B_i and C_i of the supporting legs. The passive joint in the foot attachment allows rotation around axis Y but not around X. Therefore we can consider that preventing the robot tilt, moment ${}^{R}M_x$ has to be compensated not only by the forces in points C_i in the feet sole but also in points B_i. Neglecting the spring compression we preliminary assume that points B_i are located at a constant distance h from the support plane with points C_i. The forces acting on mounting points B_i are transferred the points C_i, which are the application points of the reaction force vectors $(-{}^{R}fx_{Ci}, -{}^{R}fy_{Ci}, -{}^{R}fz_{Ci})$. Points C_i are translated in plane XY in relation to B_i (Fig. 12.7), which means:

$$\begin{aligned} {}^{R}x_{Ci} &= {}^{R}x_{Bi} + dx_i, \\ {}^{R}y_{Ci} &= {}^{R}y_{Bi} + dy_i. \end{aligned} \tag{12.8}$$

With the constant height of the body CG over points B_i (${}^{R}z_{Bi} = -H$) we evaluate the vertical components of the leg-end forces taking into account appropriate conditions for forces and moments (forces ${}^{R}fz_{B1}$ and moment ${}^{R}M_x$ – the 3rd and 4th equalities from (12.6)):

$$\begin{aligned} {}^{R}fz_{B1} &= \frac{{}^{R}fy^{e}_{CG}({}^{R}z_{CG} - H)}{{}^{R}y_{B2} - {}^{R}y_{B1}} + \frac{({}^{R}fz^{e}_{CG} + 2m_f g)({}^{R}y_{CG} - {}^{R}y_{B2})}{{}^{R}y_{B2} - {}^{R}y_{B1}}, \\ {}^{R}fz_{B2} &= {}^{R}fz^{e}_{CG} + 2m_f\, g\, - {}^{R}fz_{B1}. \end{aligned} \tag{12.9}$$

The foot mass is equal to m_f. Knowing the force ${}^{R}fz_{Bi}$ we evaluate the spring compression dh_i:

$$\begin{aligned} dh_1 &= \frac{|{}^{R}fz_{B1}|}{k_1}, \\ dh_2 &= \frac{|{}^{R}fz_{B2}|}{k_2}, \end{aligned} \tag{12.10}$$

where k_1, k_2 are the spring constants. The heights of points Bi above the support plane are equal to $h_1 = h - dh_1$ and $h_2 = h - dh_2$. ${}^{R}fz_{Ci} = {}^{R}fz_{Bi} - m_f g$ (where m_f is the foot mass), ${}^{R}fx_{Ci} = {}^{R}fx_{Bi}$, ${}^{R}fy_{Ci} = {}^{R}fy_{Bi}$. Now we define ${}^{R}COP$ – the intersection point of $\mathbf{{}^{R}f_{CG}}$ with the supporting plane (Fig. 12.7). This point is also the attachment point of the resultant reaction force vector. During the real (physical) walk it is also equivalent to the Center Of Pressure (COP). Knowing the robot sizes and spring compressions h_1, h_2 it is easy to get the coordinates of COP in the reference frame. In smooth walk (with smooth motion of the trunk) it is expected that ${}^{O}z_{COP} = {}^{R}z_{COP} = -H + geom(h_1, h_2)$. The intersection of vector $\mathbf{{}^{R}f_{CG}}$ with plane ${}^{R}z_{COP}$ has the following coordinates:

$$ {}^{R}x_{COP} = {}^{R}x_{CG} - \frac{{}^{R}fx_{CG}}{{}^{R}fz^{e}_{CG}}(H + geom(h_1,h_2) + {}^{R}z_{CG}), \tag{12.11} $$

$$ {}^{R}y_{COP} = {}^{R}y_{CG} - \frac{{}^{R}fy^{e}_{CG}}{{}^{R}fz_{CG}}(H + geom(h_1,h_2) + {}^{R}z_{CG}). \tag{12.12} $$

In stable posture the moments ${}^{R}M_x$, ${}^{R}M_y$ resulting from the reaction forces and evaluated in the supporting plane towards the reference point *COP* are equal to zero:

$$ {}^{R}Mx_{COP} = ({}^{R}y_{C1} - {}^{R}y_{COP}){}^{R}fz_{C1} + ({}^{R}y_{C2} - {}^{R}y_{COP}){}^{R}fz_{C2} = 0, \tag{12.13} $$

$$ {}^{R}My_{COP} = ({}^{R}x_{C1} - {}^{R}x_{COP}){}^{R}fz_{C1} + ({}^{R}x_{C2} - {}^{R}x_{COP}){}^{R}fz_{C2} = 0. \tag{12.14} $$

We express separately the moments ${}^{R}M^{Bi}_x$, ${}^{R}M^{Bi}_y$ acting on *CG* due to the forces at *Bi*:

$$ \begin{aligned} {}^{R}M^{Bi}_x &= {}^{R}fy_{Bi}\,H + {}^{R}fz_{Bsi}({}^{R}y_{B1} - {}^{R}y_{CG}), \\ {}^{R}M^{Bi}_y &= -{}^{R}fx_{Bsi}\,H - {}^{R}fz_{Bi}({}^{R}x_{B1} - {}^{R}x_{CG}). \end{aligned} \tag{12.15} $$

The moments ${}^{R}M^{Ci}_x$, ${}^{R}M^{Ci}_y$ exerted in *CG* due to the forces at C_i are equal to:

$$ \begin{aligned} {}^{R}M^{Ci}_x &= {}^{R}fy_{Ci}\,(H + h_i) + {}^{R}fz_{Ci}(dy_i + {}^{R}y_{B1} - {}^{R}y_{CG}), \\ {}^{R}M^{Ci}_y &= -{}^{R}fx_{si}\,(H + h_i) - {}^{R}fz_{Ci}(dx_i + {}^{R}x_{B1} - {}^{R}x_{CG}). \end{aligned} \tag{12.16} $$

The localisation of points C_i is such that no additional moments around *X* and *Y* axes are produced:

$$ {}^{R}M^{Bsi}_x = {}^{R}M^{Csi}_x, \qquad {}^{R}M^{Bsi}_y = {}^{R}M^{Csi}_y. \tag{12.17} $$

Substituting (12.15) and (12.16) in (12.17) and rearranging the first equation for each leg separately we obtain:

$$ \begin{aligned} dy_1 &= \frac{-mg(y_{B1} - {}^{R}y_{CG})}{{}^{R}fz_{C1}} - s = a_1 - s, \\ dy_2 &= \frac{-mg(y_{B2} - {}^{R}y_{CG})}{{}^{R}fz_{C2}} - h_1\,{}^{R}fy^{e}_{CG} + s = b_1 + s, \end{aligned} \tag{12.18} $$

where $s = h1\,{}^{R}fy_{C1}$. Remembering (12.8) and substituting dy_i by (12.18) in (12.13) we have:

$$ (a_1 - s)\,{}^{R}fz_{C1} + (b_1 + s)\,{}^{R}fz_{C2} = A, \tag{12.19} $$

where $A = {}^{R}y_{COP}{}^{R}fz^{e}_{CG} - {}^{R}y_{B2}{}^{R}fz_{C2} - {}^{R}y_{B1}{}^{R}fz_{C1}$. From the above it follows that $s = h1\,{}^{R}fy_{C1} = \frac{A - a_1\,{}^{R}fz_{C1} - b_1\,{}^{R}fz_{Cs2}}{{}^{R}fz_{C2} - {}^{R}fz_{C1}}$. Similarly

$$dx_1 = \frac{mg(x_{B1} - {}^R x_{CG})}{{}^R fz_{C1}} - s = c_1 - s_0,$$
$$dx_2 = \frac{mg(x_{B2} - {}^R x_{CG})}{{}^R fz_{C2}} - h_1 {}^R fx^e_{CG} + s_0 = d_1 + s_0, \tag{12.20}$$

where $s_0 = h1\ {}^R fx_{C1}$;

$$(c_1 - s)\ {}^R fz_{C1} + (d_1 + s_0)\ {}^R fz_{C2} = B, \tag{12.21}$$

$B = {}^R x_{B1}\ {}^R fz_{C1} + {}^R x_{B2}\ {}^R fz_{C2} - {}^R M_y - (H - h)\ {}^R fz^e_{CG}$ and based on the second relation from (12.14) $s_0 = h1\ {}^R fy_{C1} = \frac{B - c_1\ {}^R fz_{C1} - d_1\ {}^R fz_{Cs2}}{{}^R fz_{C2} - {}^R fz_{C1}}$.

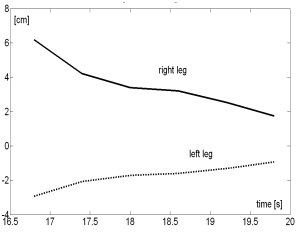


Fig. 12.8 Time courses of dx_1 (right leg) and dx_2 (left leg) during double support phase

Figure 12.8 shows the time courses of dx_1 and dx_2 for the considered robot. The values of dx_i do not go outside (-3cm;7.7cm); the robot posture is stable. To assure postural stability in the double support phase, point C_i has to be located within the foot-print, dy_i and dx_i have to match the foot dimensions. Relations (12.18) and (12.20) form the double support phase stability criterion.

12.5 Conclusion

The presented method of force evaluation decomposes equilibrium conditions taking into account feet attachments and feet-ends. Experimental adjustments of postural stability for the robot prototype by changes of leg-end spring stiffness confirms the correctness of the presented considerations. The compliant gait is summarized in [10]. The relations obtained are useful for design of walking machines. The knowledge of the leg-end force application points based on displacements dx_i, dy_i is important for synthesis of dynamical stable gaits. The knowledge of those displacements is needed to assess whether the foot supporting area will assure postural stability. A change of the spring constant, a change of the foot area, or a change of the leg configuration (change in the position of CG) are the tools which can be used for postural stability adjustment.

Acknowledgements. This work is supported by Warsaw University of Technology Research Program and the Ministry of Scientific Research and Information Technology Grant N N514 297935.

References

1. Garcia, E., Galvez, J.A., Gonzalez de Santos, P.: On Finding the Relevant Dynamics for Model-Based Controlling Walking Robots. Journal of Intelligent and Robotic Systems 37(4), 375–398 (2003)
2. Hashimoto, K., Takezaki, Y., Hattori, K., Kondo, H., Takashima, T., Lim, H.-o., Takanishi, A.: A Study of Function of Foot's Medial Longitudinal Arch Using Biped Humanoid Robot. In: The 2010 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Taiwan, pp. 2206–2211 (2010)
3. Spenneberg, D., Albrecht, M., Backhaus, T., Hilljegerdes, J., Kirchner, F., Strack, A., Schenker, H.: Aramies: A four-legged Climbing and Walking Robot. In: Proc. of 8th Int. Symp. iSAIRAS, Munich, CD ROM (September 2005)
4. Takemura, H., Deguchi, M., Ueda, J., Matsumoto, Y., Ogasawara, T.: Slip-adaptive Walk of Quadruped Robot. Robotics and Autonomous Systems 53, 124–141 (2005)
5. Vukobratovic, M., Borovac, B.: Zero-Moment Point - Thirty Five Years of its Life. Int. J. of Humanoid Robotics 1(1), 157–173 (2004)
6. Zhou, D., Low, K.H., Zielińska, T.: An Efficient Foot-force Distribution Algorithm for Quadruped Walking Robots. Robotica 18, 403–413 (2000)
7. Zielińska, T., Trojnacki, M.: Motion Synthesis of Dynamically Stable Two-legged Gait for a Quadruped Robot. Theoretical Considerations (1), PAR 11, pp. 5–11 (2007) (in Polish)
8. Zielińska, T.: Control and Navigation Aspects of a Group of Walking Robots. In: Robotica, vol. 24, pp. 23–29. Cambridge Univerity Press (2006)
9. Zielińska, T., Chew, C.M., Kryczka, P., Jargilo, T.: Robot Gait Synthesis Using the Scheme of Human Motion Skills Development. Mechanism and Machine Theory 44(3), 541–558 (2009)
10. Zielińska, T., Chmielniak, A.: Biologically Inspired Motion Synthesis of Two Legged Robot. Robotica (in print, 2011)

Part III
Control of Robotic Systems

Chapter 13
Generalized Predictive Control of Parallel Robots

Fabian A. Lara-Molina, João M. Rosario, Didier Dumur, and Philippe Wenger

Abstract. This study addresses the position tracking control application of a parallel robot using predictive control. A Generalized Predictive Control strategy (GPC), which considers the linear dynamic model, is used to enhance the dynamic performance. A realistic simulation of the complete model of the Orthoglide robot is performed on two different trajectories with the purpose of comparing the GPC controller with the classical Computed Torque Control (CTC) in terms of tracking accuracy. The GPC controller shows better performance for high accelerations with uncertain dynamic parameters.

13.1 Introduction

Parallel robots are based on closed-loop chain mechanisms. Due to their mechanical structure, they have some conceptual advantages over serial robots, such as higher stiffness, accuracy, payload-weight ratio and better dynamic performance. However, they have more kinematic and dynamic complexities than serial robots. To reach a high performance in industrial applications, their dynamical potential advantages should be exploited completely. Consequently, it is essential to reduce the time of execution and to increase the accuracy in order to improve the productivity and quality of manipulation and production processes that use parallel robots [11].

Fabian A. Lara-Molina · João M. Rosario
Mechanical Engineering School, State University of Campinas, Campinas, SP Brazil
e-mail: {lara,rosario}@fem.unicamp.br

Didier Dumur
Automatic Control Department, Supelec Systems Sciences (E3S), Gif-sur-Yvette, France
e-mail: didier.dumur@supelec.fr

Philippe Wenger
Institut de Recherche en Communications et Cybernétique de Nantes, Nantes, France
e-mail: philippe.wenger@irccyn.ec-nantes.fr

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 159–169.
springerlink.com

Two factors affect the accuracy of parallel robots. First, passive joints produce kinematic model errors due to clearances and assembly defects [13]. Second, singularities within workspace volume [3] produce a decrease of the stiffness resulting in a lack of accuracy for a given task. Therefore, parallel robots still need improvements in design, modeling and control in order to reach their theoretical capabilities. As seen, many works address modeling and design; nevertheless, there are few works related to parallel robot control.

Mainly two control approaches have been considered for parallel robots in literature: dynamic control, which is based on dynamic models of these robots [9]; and adaptive control, which adjusts the parameters of the system or controller online [11]. Additionally, dynamic control techniques as CTC do not cope very well with modeling errors. They create a perturbation of the error behaviors which may lead to a lack of stability and accuracy [4].

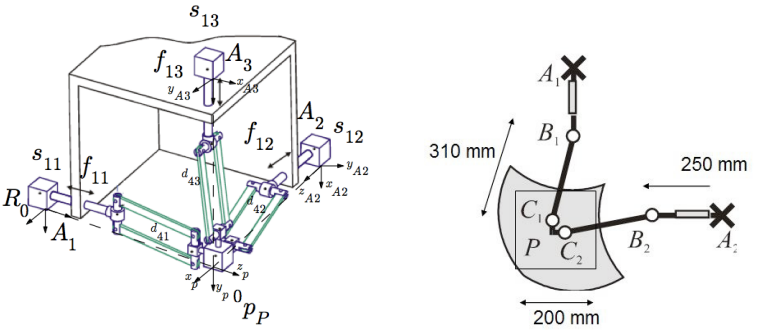
On the other hand, model based predictive control techniques have been applied to parallel robots; Belda et al. [1] designed a GPC controller for path control of redundant parallel robots; Vivas and Poignet [12] applied functional predictive control based on the simplified dynamic model of H4 parallel robot; Duchaine et al. [5] presented a model predictive control law considering the dynamics of the robot. Nevertheless, it is necessary to have robust controllers for model errors and dynamic uncertainties and that guarantee stability and accuracy.

In this study, we use Generalized Predictive Control (GPC) to enhance the dynamic performance of a parallel robot in the position tracking control. Then we compare the GPC performance with the classical robot controller: Computed Torque Control (CTC). First, the dynamic equation of the robot is linearized in order to apply the linear control laws. After that, based on the linear model, we apply GPC and CTC control in each actuator of the parallel robot. Finally, we perform a realistic simulation of the complete model of the Orthoglide robot; thus, the performance of CTC and GPC controllers is evaluated in terms of tracking accuracy using two different workspace trajectories. We evaluate how the tracking accuracy is affected with the dynamic parameters variation.

This study is organized in five sections. Section 13.2 presents the kinematic and dynamic model of the parallel robot. Section 13.3 sumarizes the GPC design procedure. In Section 13.4, the CTC controller is presented. In Section 13.5, simulation results are presented. Finally, we present the conclusion and further work.

13.2 Robot Modeling

Orthoglide is a parallel robot with three translational degrees of freedom (Fig. 13.1). Orthoglide mechanical structure has a movable platform, three prismatic actuators, and three identical kinematic chains PRP_aR (P prismatic, R rotational, P_a parallelogram). The end effector of the parallel robot is the movable platform. The inverse geometric and kinematic models are needed to compute the dynamic equation, hence we present the different relations to obtain these models and matrices [14].



(a) geometric CAD (b) dimensions

Fig. 13.1 Orthoglide robot

The Inverse Geometric Model (IGM) delivers the actuator positions vector $\mathbf{s} = \begin{bmatrix} s_{11} & s_{12} & s_{13} \end{bmatrix}^T$ as function of Cartesian position of the end effector ${}^0\mathbf{p}_P = \begin{bmatrix} x_p & y_p & z_p \end{bmatrix}^T$ [10], thus:

$$\begin{bmatrix} s_{11} \\ s_{12} \\ s_{13} \end{bmatrix} = \begin{bmatrix} z_p \cos(s_{31}) \cos(s_{21}) d_{41} - d_{61} \\ x_p - x_{A2} - \cos(s_{32}) \cos(s_{22}) d_{42} - d_{62} \\ y_p - y_{A3} - \cos(s_{33}) \cos(s_{23}) d_{43} - d_{63} \end{bmatrix}, \tag{13.1}$$

$$s_{31} = \sin^{-1}\left(\frac{-y_p}{d_{41}}\right), \qquad s_{21} = -\left(\sin^{-1}\left(\frac{-x_p}{\cos(s_{31}) d_{41}}\right) + \frac{\pi}{2}\right),$$

$$s_{32} = \sin^{-1}\left(\frac{-z_p + z_{A2}}{d_{42}}\right), \qquad s_{22} = -\left(\sin^{-1}\left(\frac{-y_p + y_{A2}}{\cos(s_{32}) d_{42}}\right) + \frac{\pi}{2}\right)$$

$$s_{33} = \sin^{-1}\left(\frac{-x_p + x_{A3}}{d_{43}}\right), \qquad s_{23} = -\left(\sin^{-1}\left(\frac{-z_p + z_{A3}}{\cos(s_{33}) d_{43}}\right) + \frac{\pi}{2}\right).$$

The Inverse Kinematic Model (IKM) delivers the actuators velocities $\dot{\mathbf{s}}$ as a function of the Cartesian velocity of the end effector ${}^0\mathbf{v}_P = \begin{bmatrix} \dot{x}_P & \dot{y}_P & \dot{z}_P \end{bmatrix}^T$:

$$\dot{\mathbf{s}} = \begin{bmatrix} \dot{s}_{11} & \dot{s}_{12} & \dot{s}_{13} \end{bmatrix}^T = {}^0\mathrm{J}_P^{-1}\, {}^0\mathbf{v}_P, \tag{13.2}$$

where ${}^0\mathrm{J}_P^{-1}$ is the inverse Jacobian matrix of the robot, thus:

$${}^0\mathrm{J}_P^{-1} = \begin{bmatrix} -1/\tan(s_{21}) & \tan(s31)/\sin(s_{21}) & 1 \\ 1 & -1/\tan(s_{22}) & \tan(s_{32})/\sin(s_{22}) \\ \tan(s_{33})/\sin(s_{23}) & 1 & -1/\tan(s_{23}) \end{bmatrix}. \tag{13.3}$$

The second order inverse kinematic model gives the actuators accelerations $\ddot{\mathbf{s}} = \begin{bmatrix} \ddot{s}_{11} & \ddot{s}_{12} & \ddot{s}_{13} \end{bmatrix}^{-T}$ as a function of the Cartesian platform acceleration ${}^0\dot{\mathbf{v}}_P$, and actuator velocities $\dot{\mathbf{s}}$:

$$\ddot{\mathbf{s}} = {}^{0}\mathsf{J}_{P}^{-1}({}^{0}\dot{\mathbf{v}}_{P} - {}^{0}\dot{\mathsf{J}}_{P}\dot{\mathbf{s}}). \tag{13.4}$$

The Cartesian accelerations of the end effector ${}^{0}\dot{\mathbf{v}}_{P}$ can be calculated using the direct dynamic equation of the Orthoglide. It is written in the following form [6]:

$${}^{0}\dot{\mathbf{v}}_{P} = \mathsf{A}_{robot}^{-1}[{}^{0}\mathsf{J}_{P}^{-T}\mathbf{f} - \mathbf{h}_{robot}], \tag{13.5}$$

where: $\mathsf{A}_{robot} = \sum_{i=1}^{3}[\mathsf{A}_{ix}] + \mathsf{I}_3 m_p$ is the total inertia matrix (3×3) of the robot, the inertia of kinematic chains and movable platform. $\mathbf{f} = \begin{bmatrix} f_{11} & f_{12} & f_{13} \end{bmatrix}^T$ are the actuator forces. $\mathbf{h}_{robot} = \sum_{i=1}^{3}[\mathbf{h}_{ix}(\mathbf{s}_i, \dot{\mathbf{s}}_i) - \mathsf{A}_{ix}\,{}^{0}\dot{J}_i\dot{\mathbf{s}}_i] - m_p\mathbf{g}$. $\mathbf{h}_{ix}$ is the Coriolis, gravitation, and centrifuge force vector (3 × 1). A_{ix} is the inertia matrix (3 × 3) of each kinematic chain. m_p is the mass of the movable platform, $\mathbf{g} = \begin{bmatrix} 0 & g & 0 \end{bmatrix}^T$ is the gravity vector.

The dynamic parameters of the robot correspond to masses, inertias and frictions identified in [7]. Thus, the dynamic parameters of the Orthoglide robot arecollected in Table 13.1.

Table 13.1 Essential dynamical parameters of the Orthoglide robot

m_P	1.59 kg	m_{a1}	8.70 kg	F_{v1}	84.66 N/(m/s)	F_{s1}	54.40 N	m_{a2}	8.49 kg
F_{v2}	8.49 N/(m/s)	F_{s2}	80.7283 N	m_{a3}	8.67 kg	F_{v3}	83.7047 N/(m/s)	F_{s3}	54.9723 N

We can express the dynamics equation as function of the actuator forces $\mathbf{f}$ and a vector $\mathbf{k}$ which represents the dynamics parameters of the actuators (dry and viscous frictions and masses), thus:

$${}^{0}\dot{\mathbf{v}}_{P} = f(\mathbf{f}, \mathbf{k}), \tag{13.6}$$

where: $\mathbf{k} = \begin{bmatrix} m_P & F_{v1} & F_{s1} & m_{a1} & F_{v2} & F_{s2} & m_{a2} & F_{v3} & F_{s3} & m_{a3} \end{bmatrix}^T$.

13.3 Generalized Predictive Control

This section presents the principles and briefly describes the formulation of GPC control to introduce the design procedure and implementation on the parallel robot. This control technique was developed by Clarke et al. [2]. In linear GPC theory, the plant is modeled by input/output CARIMA form

$$A(q^{-1})y(t) = B(q^{-1})u(t-1) + \frac{C(q^{-1})\xi(t)}{\Delta(q^{-1})}. \tag{13.7}$$

With $u(t)$, $y(t)$ the plant input and output, $\xi(t)$ a centered Gaussian white noise, and $C(q^{-1})$ models the noise influence. The difference operator $\Delta(q^{-1}) = 1 - q^{-1}$ in the disturbance model helps eliminate the static error by introducing an integral action in the controller. The control signal is obtained by minimization of the quadratic cost function:

$$J = \sum_{j=N_1}^{N_2} [\mathbf{r}(t+j) - \hat{\mathbf{y}}(t+j)]^2 + \lambda \sum_{j=1}^{N_u} \Delta \mathbf{u}(t+j-1)^2, \tag{13.8}$$

where N_1 and N_2 define the output prediction horizons, and N_u defines the control horizon. λ is a control weighting factor, $\mathbf{r}$ the reference value, $\hat{\mathbf{y}}$ the prediction output value, obtained by solving the diophantine equation, and $\mathbf{u}$ the control signal. The receding horizon principle assumes that only the first value of optimal control series resulting from the optimization of Eq. (13.8) is applied, so that for the next step this procedure is repeated. This control strategy leads to a 2-dof RST controller, implemented through a difference equation:

$$S(q^{-1})\Delta(q^{-1})u(t) = -R(q^{-1})y(t) + T(q)r(t)\,. \tag{13.9}$$

Thus, the design has been performed with $C(q^{-1}) = 1$ and N_1, N_2, N_u, λ being adjusted to satisfy the required input-output behavior. The resulting RST controller is showed in Fig. 13.2.

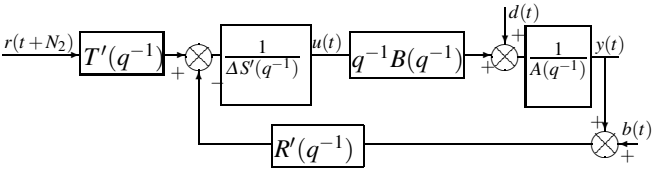


Fig. 13.2 RST form of the GPC controller

13.4 Computed Torque Control (CTC)

In order to apply a linear control strategy, such as CTC and GPC, it is basically required to linearize the non-linear dynamic model of the robot in Eq. (13.5). $\mathsf{A}(\mathbf{s})$ and $\mathsf{H}(\mathbf{s},\dot{\mathbf{s}})$ were defined in Eq. (13.5). Thus, the non-linear dynamic equation of the robot is considered as follows:

$$\mathbf{f} = \mathsf{A}(\mathbf{s})\ddot{\mathbf{s}} + \mathsf{H}(\mathbf{s},\dot{\mathbf{s}})\,. \tag{13.10}$$

The robot equations may be linearized and decoupled by non-linear feedback. $\hat{\mathsf{A}}(\mathbf{s})$ and $\hat{\mathsf{H}}(\mathbf{s},\dot{\mathbf{s}})$ are, respectively, the estimates of $\mathsf{A}(\mathbf{s})$ and $\mathsf{H}(\mathbf{s},\dot{\mathbf{s}})$. Assuming that $\hat{\mathsf{A}}(\mathbf{s}) = \mathsf{A}(\mathbf{s})$ and $\hat{\mathsf{H}}(\mathbf{s},\dot{\mathbf{s}}) = \mathsf{H}(\mathbf{s},\dot{\mathbf{s}})$, the problem is reduced to a linear control on three decoupled double-integrators [8], where

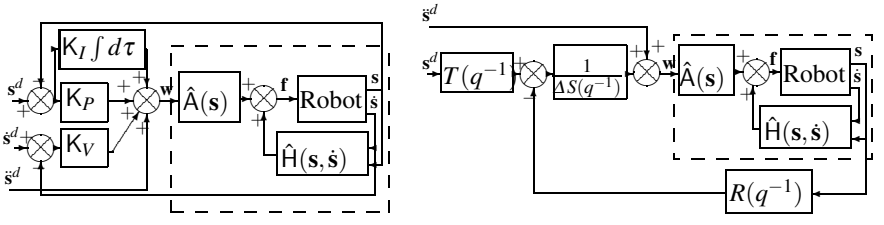
$$\ddot{\mathbf{s}} = \mathbf{w}\,, \tag{13.11}$$

with $\mathbf{w}$ being the new input control vector. This equation corresponds to the inverse dynamics control scheme, where the direct dynamic model of the robot is transformed into a double set of integrators (Fig. 3(a)). Thus, linear control techniques

can then be used to design tracking position controllers, such as the model-based predictive control (CARIMA model of Section 13.3) or a CTC controller. Let us assume that the desired trajectory is specified with the desired position $\mathbf{s}^d$, velocity $\dot{\mathbf{s}}^d$ and acceleration $\ddot{\mathbf{s}}^d$; the CTC control law is given by:

$$\mathbf{w} = \ddot{\mathbf{s}}^d + \mathsf{K}_P(\mathbf{s}^d - \mathbf{s}) + \mathsf{K}_D \frac{d}{dt}(\mathbf{s}^d - \mathbf{s}) + \mathsf{K}_I \int_0^t (\mathbf{s}^d - \mathbf{s}) d\tau \,, \tag{13.12}$$

where K_P , K_D, and K_I are the matrices which contain in the diagonal the controller gains and the other terms are nulls.



(a) block diagram (b) RST form

Fig. 13.3 CTC controller

The controller gains are found in order to have in continuous-time domain the following closed-loop characteristic equation (s is the Laplace variable):

$$(s + \omega_r)(s^2 + 2\xi\,\omega_r s + \omega_r^2) = 0\,. \tag{13.13}$$

The CTC controller is implemented in RST form using the Euler transform with sample period T_e and a filter for the derivative action. Figure 3(b) shows the corresponding controller.

13.5 Simulation and Results

We simulated the dynamic response of the Orthoglide robot using the CTC and GPC controllers in order to establish a performance comparison in terms of tracking accuracy. The controller parameters were set according to the procedure design presented earlier. The control laws were tested in simulation in Matlab/Simulink® environment.

The robot behavior is simulated using the direct dynamic model of the parallel robot in Eq. (13.5). Uncertainties about dynamic parameters, errors in geometric parameters (due to assembly tolerances), Gaussian noise on the sensors due to accuracy in the actuator sensors (mean value: 1 µm and variance $1 \cdot 10^{-13}$) and dynamical parametes $\mathbf{k}$ are included for a realistic simulation (Fig. 13.4).

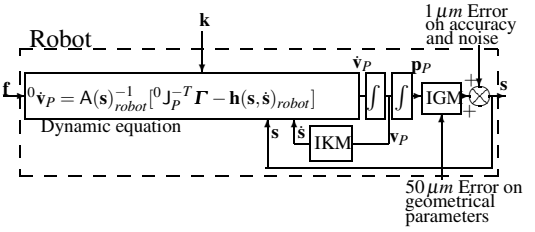


Fig. 13.4 Parallel robot, simulation

The CTC controller is tuned with the following parameters: $\xi = 1$ to guarantee the response without overshot; $\omega_r = 60$ rad/s leading to $k_P = 10800$, $k_D = 180$, $k_I = 216000$, and CTC is implemented in RST form using the Euler transform with sample period $T_e = 2.5$ ms and a filter for the derivative action. In the same way, the GPC controller is tuned with the following parameters: $N_1 = 1$, $N_2 = 10$, $N_u = 1$, and $\lambda = 1 \cdot 10^{-9}$.

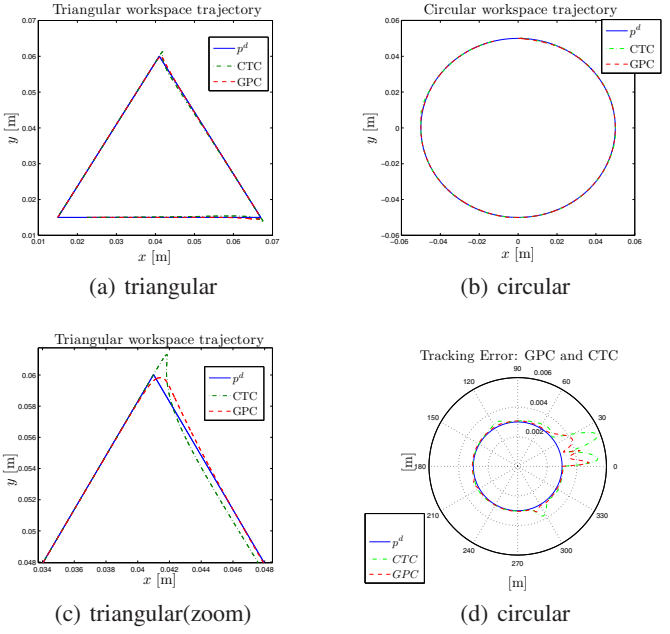
With the purpose of comparing the behavior of the CTC and GPC controllers, two workspace trajectories $\mathbf{p}^d$ on the $x - y$ plane are used: 1) a triangular one (edge length = 50 mm), with a fifth-degree polynomial interpolation; thus it has a smooth joint space trajectory, at the points where the direction of the trajectory changes the initial acceleration is 1 m/s^2 to test the behavior of the controllers (Fig. 5(a)); and 2) a circular one ($\varnothing = 50$ mm, Fig. 5(b)), the initial conditions (position, velocity, and acceleration) of the trajectory in the y axis being different from zero. Figure 13.5 presents the respective workspace trajectories using the CTC and GPC controllers. For these workspace trajectories, the highest acceleration on the end effector is 5 m/s^2.

Initially, we can see that the workspace trajectories are closer to reference using the GPC controller (Fig. 5(d)). The GPC controller improves the tracking of the workspace trajectory, since with this controller the robot softly follows the abrupt changes in direction, due to the anticipative effect of the predictive control (Fig. 5(c)).

In order to establish the total tracking error of the parallel robot over a trajectory, the Root Mean Square Error (RMSE) of the actuators is evaluated:

$$RMSE\left(e(t)\right) = \frac{1}{3}\sum_{k=1}^{3}\sqrt{\sum_{t=1}^{N}\frac{e(t)_k^2}{N}}. \tag{13.14}$$

In Fig. 13.6, the maximum acceleration of the end effector varies from 1 m/s^2 to 5 m/s^2 for the triangular workspace trajectory (Fig. 6(a)) and the circular workspace trajectory (Fig. 6(b)). For both trajectories, when acceleration increases, tracking accuracy decreases because RMSE increases. However, when using the GPC

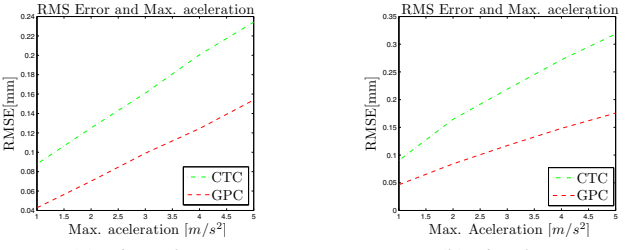


(a) triangular (b) circular

(c) triangular(zoom) (d) circular

Fig. 13.5 Tracking workspace trajectories

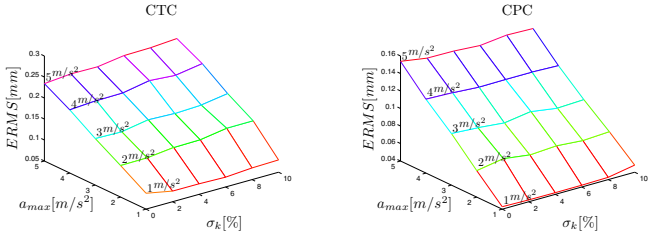
controller, the increase of RMSE is smaller than with the CTC controller; thus GPC offers a better tracking accuracy. Hence, for parallel robots that operate at high accelerations (high dynamics), the GPC controller performs better than the CTC controller according to the increase of acceleration.



(a) triangular (b) circular

Fig. 13.6 RMSE error and maximum acceleration

Finally, we evaluate the robustness of the tracking accuracy of the parallel robot with respect to variations of dynamic parameters of the actuators **k** and acceleration $a_{\max}$ over the triangular workspace trajectory. Figure 13.7 shows RMSE as a function of the maximum acceleration of the end effector $a_{\max}$ and the percentual

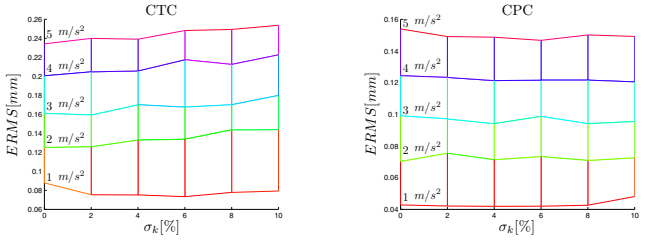


(a) CTC (b) GPC

Fig. 13.7 RMSE, maximum acceleration and parameter variation

variation of dynamic parameters of the actuators of the robot **k** in Eq. (13.6). Each dynamic parameter of **k** varies from the original value with an increment of 10%. This increment is an approximation of the dynamic parameter change due to the operation of the parallel robot or inaccurate parameter identification. We can see that RMSE increases for high accelerations, hence the tracking accuracy decreases for this condition. Even so, the GPC controller performs better than CTC for high accelerations and variations of the dynamic parameters (the scales in Fig. 13.7 are not the same on the vertical axis).

Figure 8(a) shows that when using the CTC controller, the increase of RMSE is proportional to the dynamic parameter increase. However, as illustrated in Fig. 8(b), when using the GPC controller, RMSE remains almost constant to the dynamic parameter variation. Therefore, the GPC controller is more robust to parameter variations because the minimization of the cost function and the integral action of CARIMA model eliminates the offset error due to uncertain parameters. This simple consideration is useful for practical effects.



(a) CTC (b) GPC

Fig. 13.8 RMSE and parameter variation

As we see, the CTC controller is based on model and parameters, its performance depends on an accurate identification of the dynamic parameters; thus, the tracking accuracy over a trajectory decreases if the dynamic parameters vary. Moreover, the GPC controller offers robust behavior, maintaining the tracking accuracy to dynamic parameters variation.

13.6 Conclusion

The simulation position tracking performance of a parallel robot with respect to two trajectories is analyzed for GPC and CTC controllers. In order to apply these linear control laws, the parallel robot was linearized by feedback.

For two trajectories typically used in machining, the simulation results of the complete model of the parallel robot show better performance in terms of tracking accuracy with respect to parameter variation whem using GPC controllers. Thus, the simulations show that the predictive control improves dynamic behavior of the parallel robot in terms of tracking error over a trajectory with high acceleration. Further work will test predictive control techniques for the Orthoglide robot experimentally.

Acknowledgements. The authors gratefully acknowledge the support of Fundação de Amparo à Pesquisa do Estado de São Paulo [Research Support Foundation of the State of São Paulo].

References

1. Belda, K., Böhma, J., Valášek, M.: State-space generalized predictive control for redundant parallel robots. Mechanics Based Design of Structures and Machines 31(3), 413–432 (2003)
2. Clarke, D., Mohtadi, C., Tuffs, P.: Generalized predictive control. part i. the basic algorithm. Automatica 23(2), 137–148 (1987)
3. Dasgupta, B., Mruthyunjaya, T.S.: The Stewart platform manipulator: a review. Mechanism and Machine Theory 35(1), 15–40 (2000)
4. Dombre, E., Khalil, W.: Modeling, Performance Analysis and Control of Robot Manipulators. Wiley (2010)
5. Duchaine, V., Bouchard, S., Gosselin, C.: Computationally efficient predictive robot control. IEEE/ASME Transactions on Mechatronics 12(5), 570–578 (2007)
6. Guegan, S., Khalil, W., Chablat, D., Wenger, P.: Modelisation dynamique d'un robot parallele a 3-DDL: l'Orthoglide. In: Conference Internationale Francophone d'Automatique (2007)
7. Guegan, S., Khalil, W., Lemoine, P.: Identification of the dynamic parameters of the orthoglide. In: IEEE International Conference on robotics and Automation, vol. 3, pp. 3272–3277 (2003)
8. Khalil, W., Etienne, D.: Modélisation identification et commande des robots. Hermes Sciences Publicat. (1999)
9. Paccot, F., Andreff, N., Martinet, P.: A review on the dynamic control of parallel kinematic machines: Theory and experiments. The International Journal of Robotics Research 28(3), 395–416 (2009)
10. Pashkevich, A., Chablat, D., Wenger, P.: Kinematics and workspace analysis of a three-axis parallel manipulator: The orthoglide. Robotica 24(1), 39–49 (2006)

11. Pietsch, I., Krefft, M., Becker, O., Bier, C., Hesselbach, J.: How to Reach the Dynamic Limits of Parallel Robots? An Autonomous Control Approach. IEEE Transactions on Automation Science and Engineering 2(4), 369–390 (2005)
12. Vivas, A., Poignet, P.: Predictive functional control of a parallel robot. Control Engineering Practice 13(7), 863–874 (2005)
13. Wang, J., Masory, O.: On the accuracy of a Stewart platform. i. The effect of manufacturing tolerances. In: IEEE International Conference on Robotics and Automation, vol. 1, pp. 114–120 (1993)
14. Wenger, P., Chablat, D.: Kinematic analysis of a new parallel machine tool: the orthoglide. In: 7th International Symposium on Advances in Robot Kinematics (2000)

Chapter 14
Specification of a Multi-agent Robot-Based Reconfigurable Fixture Control System

Cezary Zieliński, Tomasz Kornuta, Piotr Trojanek,
Tomasz Winiarski, and Michał Walęcki

Abstract. The paper presents a formal specification of the control software of a reconfigurable fixture used for machining thin plates. The fixture is based on relocatable supporting robots. A multi-agent approach to control system structuring is used. The behaviour of agents is defined in terms of finite state automatons, transition functions, and terminal conditions.

14.1 Introduction

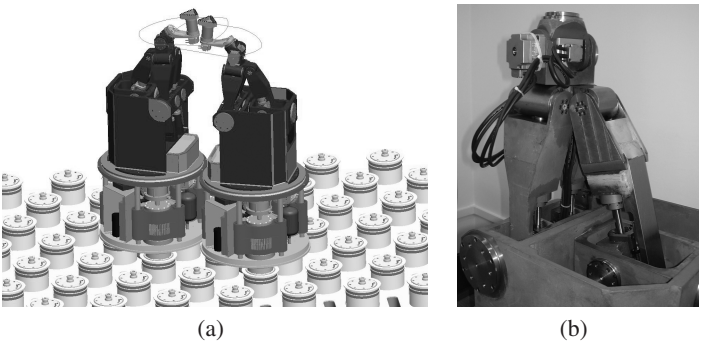
When large-sized thin panels (e.g. parts of airplane fuselage) are machined they need to be fixed rigidly. Universal fixtures have to cope with the diversity and complexity of shapes of those plates. Manual reconfiguration of such fixtures is tedious and time consuming. Thus it is preferable that such fixtures have a self-reconfiguration capability. The number of supports can be reduced if they can relocate themselves during machining of the plate, increasing locally the rigidity of the plate in the vicinity of the place which is currently being machined. This led to the idea of a self-reconfigurable fixture composed of a gang of robots able to relocate themselves under the plate that they have to support [1]. The resulting device is a multi-robot system, thus its control system can assume a multi-agent structure, where the behaviour of each of the agents is specified in terms of transition functions, terminal conditions, selection predicates, and finite state automatons [6, 7]. Such a specification follows the operational semantics approach to the description of programming languages [3]. It should be noted that an overwhelming majority of robot controllers is designed using only intuition and experience of the designer.

Cezary Zieliński · Tomasz Kornuta · Piotr Trojanek · Tomasz Winiarski · Michał Walęcki
Institute of Control and Computation Engineering, Warsaw University of Technology,
ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: {c.zielinski,t.kornuta}@elka.pw.edu.pl,
{p.trojanek,t.winiarski}@ia.pw.edu.pl,
mw@mwalecki.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 171–182.
springerlink.com

The approach presented here has formal roots, thus it can be made algorithm-based to a certain extent, hence the result will be less dependent on the designer's abilities and less prone to errors.

In general functioning of the considered system (Fig. 14.1(a)) can be described as follows. An off-line program, on the basis of CAD geometric data describing the panel, generates a plan of relocation of the robots during machining [4]. Either drilling or milling is performed on the panels. Drilling requires a static configuration of manipulators serving several holes, while milling requires constant relocation of manipulators during this operation. The panels are fixed manually to a few static clamps initially positioning them in the fixture. The supervisory controller uses the plan to control the robots during machining. It also synchronizes the actions of the manipulators and the CNC machine. The robots relocate themselves over a bench, which contains docking elements. The docking elements form a mesh of equilateral triangles. The base of the robot, when supporting the machined panel, is docked to three adjacent docking elements. During transfer to another location it detaches from two of the docking elements and rotates itself around the one that remains attached to the bench. Once the robot is securely locked to the bench, the manipulator (the parallel kinematic machine, PKM (Fig. 14.1(b) [2]), with a serial spherical wrist mounted on top) rises the supporting head to the prescribed location, thus providing support to the machined panel. Initially the head is soft, but once in place it solidifies and vacuum is applied to suck the plate to the head.



(a) (b)

Fig. 14.1 The system (a) agents on the bench (b) PKM (Courtesy of DIMEC, Exechon, ZTS VVÙ)

14.2 General Description of the Controller

Figure 14.2(a) presents the assumed logical structure of the control system. Each robot j is composed of three embodied agents a_{j_1}, a_{j_2}, and a_{j_3}, $j = 1, \ldots, n_a$, where n_a is the number of robots present in the considered system: a_{j_1} uses the manipulator as its effector, a_{j_2} contains the mobile base and the docking elements (three clamps attaching themselves to the pins protruding from the bench) and a_{j_3} is associated with the supporting head and the vacuum producing elements. The control

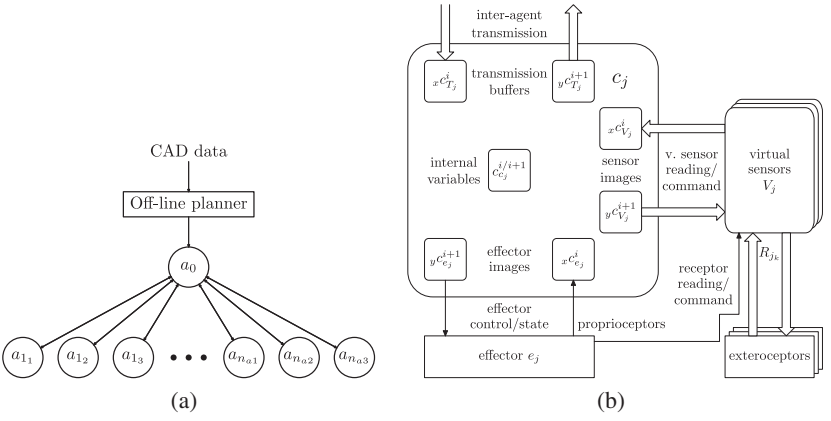


Fig. 14.2 General structure of: (a) the system and (b) an agent

subsystems of those agents are denoted by c_{j_q}, where $q = 1,\ldots,3$. The coordinator is labeled a_0. The agents, in principle, can communicate with each other, however in this case only agent-coordinator communication suffices.

The control subsystem c_j contains data structures representing components of the agent (effector e_j, virtual sensors V_j, and inter-agent transmission buffers T_j) [6, 7]. Those representations are called images. The subsystem (Fig. 14.2(b)) contains:

- ${}_xc_{e_j}$ – input image of the effector (proprioceptive input from the effector),
- ${}_xc_{V_j}$ – input images of the virtual sensors (current virtual sensor readings),
- ${}_xc_{T_j}$ – input of the inter-agent transmission (information obtained from other agents),
- ${}_yc_{e_j}$ – output image of the effector,
- ${}_yc_{V_j}$ – output images of the virtual sensors (commands for the virtual sensors),
- ${}_yc_{T_j}$ – output of the inter-agent transmission (information sent to the other agents),
- c_{c_j} – variables and constants internal to the agent's control subsystem.

The state of the control subsystem changes at certain instants of time. If i denotes the current instant, the next considered instant is denoted by $i+1$. The control subsystem uses ${}_xc_j^i = \langle c_{c_j}^i, {}_xc_{e_j}^i, {}_xc_{V_j}^i, {}_xc_{T_j}^i \rangle$ to produce ${}_yc_j^{i+1} = \langle c_{c_j}^{i+1}, {}_yc_{e_j}^{i+1}, {}_yc_{V_j}^{i+1}, {}_yc_{T_j}^{i+1} \rangle$. For that purpose it uses transition functions:

$$\begin{cases} c_{c_j}^{i+1} &= f_{c_{c_j}}(c_{c_j}^i, {}_xc_{e_j}^i, {}_xc_{V_j}^i, {}_xc_{T_j}^i) \\ {}_yc_{e_j}^{i+1} &= f_{c_{e_j}}(c_{c_j}^i, {}_xc_{e_j}^i, {}_xc_{V_j}^i, {}_xc_{T_j}^i) \\ {}_yc_{V_j}^{i+1} &= f_{c_{V_j}}(c_{c_j}^i, {}_xc_{e_j}^i, {}_xc_{V_j}^i, {}_xc_{T_j}^i) \\ {}_yc_{T_j}^{i+1} &= f_{c_{T_j}}(c_{c_j}^i, {}_xc_{e_j}^i, {}_xc_{V_j}^i, {}_xc_{T_j}^i) \end{cases} \tag{14.1}$$

Formula (14.1), in compact form written as ${}_yc_j^{i+1} = f_{c_j}({}_xc_j^i)$, is a prescription for evolving the state of the system, thus it has to be treated as a program of the agent's behaviour. This function is usually very complex, so it needs to be decomposed into n_f partial functions:

$$ {}_{y}c_j^{i+1} = {}^{m}f'_{c_j}({}_{x}c_j^{i}), \quad m = 1,\ldots,n_f, \tag{14.2} $$

where n_f depends on the number of behaviours that the agent has to exhibit. Each such function governs the operation of the agent for the time defined by events that are detected by a predicate called the terminal condition ${}^{m}f_{\tau_j}$. The satisfaction of the terminal condition stops the repetition of ${}^{m}f'_{c_j}$ computations.

```
e_j ⟼ x c^i_{e_j};  V_j ⟼ x c^i_{V_j};      // Get the initial state – i = i_0
.........
loop
  y c^i_{T0j} ⟼ x c^i_{Tj0};     // Get the command from the coordinator
  // Interpret the command and select appropriate behaviour
  .........
  // Execute the behaviour with the selected transition function and terminal condition
  loop  // The behaviour
    // Check the terminal condition
    if m f_{τ_j}(x c^i_j) = false  then
        y c^{i+1}_j := m f'_{c_j}(x c^i_j);          // Compute the next control subsystem state
        y c^{i+1}_{e_j} ⟼ e_j;  y c^{i+1}_{V_j} ⟼ V_j;  // Transmit the results
        i := i + 1;                                  // Wait for the next iteration
        e_j ⟼ x c^i_{e_j};  V_j ⟼ x c^i_{V_j};       // Determine the current state of the agent
    endif
  endloop  // End of the behaviour
  y c^i_{Tj0} ⟼ x c^i_{T0j};     // Inform the coordinator that the command has been executed
endloop;
```

Fig. 14.3 Pseudocode representing a behaviour of an agent (⟼ represents transfer of data)

The pseudocode presented in Fig. 14.3 governs the behaviour of the system. It reflects the structure of a finite state automaton (FSA), where behaviours are associated with the nodes of the graph representing this automaton. As long as the behaviour is executed, the FSA is in a state represented by a node associated with this behaviour, so the transition function ${}^{m}f'_{c_j}$ of this behaviour is executed iteratively. When the terminal condition ${}^{m}f_{\tau_j}$ is satisfied the FSA changes its state. The next behaviour is chosen by testing predicates associated with the arcs of the graph. The selected m-th behaviour invokes its transition function ${}^{m}f'_{c_j}$ and condition ${}^{m}f_{\tau_j}$.

The agent a_{j_1} controls the j-th manipulator, i.e.: 1) moves the head to a predefined support position, 2) lifts the head until contact with the panel is detected, 3) supports the panel, 4) lowers the head, 5) remains dormant in the folded configuration (in this state its mobile base can relocate it). The agent a_{j_2} controls the j-th mobile base and its locking clamps, i.e.: 1) lifts the clamps, 2) translocates itself to the next pin, 3) lowers the clamps, 4) remains dormant in the locked configuration (in this state the manipulator can move the head or support the panel). The agent a_{j_3} controls the j-th supporting head, i.e.: 1) switches on the vacuum, 2) solidifies the contents of the head, 3) supports the panel, 4) switches off the vacuum, 5) desolidifies the contents of the head, 6) remains dormant. A behaviour of each agent a_{j_q} requires a pair: the transition function ${}^{m}f'_{c_{jq}}$ and the terminal condition ${}^{m}f_{\tau_{jq}}$.

The coordinator a_0 is responsible for: 1) interaction with the operator, 2) reading-in the plan formulated by the off-line program, 3) coordinating the actions of the agents controlling the robots. The transition functions ${}^{m}f'_{c_0}$ and the terminal conditions ${}^{m}f_{\tau_0}$ of the agent a_0 operate only on transmission buffers as their arguments, i.e., ${}_{x}c_{T_{0j_1}}$, ${}_{x}c_{T_{0j_2}}$, and ${}_{x}c_{T_{0j_3}}$. Their results are loaded into output transmission buffers ${}_{y}c_{T_{0j_1}}$, ${}_{y}c_{T_{0j_2}}$, and ${}_{y}c_{T_{0j_3}}$.

The full specification of all agents and their behaviours cannot be accommodated here, so the most elaborate behaviour of agent a_{j_1} has been chosen for presentation. Operational space (Cartesian) straight-line motion of the manipulator end-effector (head) is defined here. The full specification of all agents is contained in [8].

14.3 Cartesian Straight Line Trajectory Generation

First the method of trajectory generation is presented. The algorithm has to take into account not only the required path, but also the capabilities of the control hardware – the motion microcontrollers. The motors are governed by MAXON EPOS2 microcontrollers. The required path is a straight line in Cartesian space. The EPOS2 has several modes of operation. The most appropriate in this case is cubic polynomial change of position with respect to time. Each segment of motion requires three PVT parameters: P – segment terminal position, V – velocity, T – time (duration of motion in that segment). Those parameters pertain to motion of a single motor, so for a 6 dof manipulator 6 motors are controlled.

A two phase trajectory generation algorithm is assumed. A straight line trajectory in operational space is divided into segments assuming parabolic velocity profile (conforming to the capabilities of the EPOS2 microcontrollers) within each segment. The endpoints of the segments are transformed into the configuration space. Those are used to compute the parameters of the configuration space interpolation. Then the feasibility of configuration space velocities and accelerations is checked.

The following notation is assumed. The indices 0, E, and G represent the base, end-effector, and goal Cartesian coordinate frames, respectively. $\mathscr{T}$ is a homogeneous transformation matrix. $\mathscr{M}$ represents a motor position vector; $\mathscr{V}$ and $\mathscr{A}$ are motor velocity and acceleration vectors (all expressed in the configuration space).

On the straight line in Cartesian space the interpolation nodes are established. The homogeneous transformation describing the necessary transition, i.e., ${}^{E}_{G}\mathscr{T} = {}^{0}_{E}\mathscr{T}^{-1}\,{}^{0}_{G}\mathscr{T}$, is computed. Next ${}^{E}_{G}\mathscr{T}$ is transformed into $[{}^{E}_{G}\mathscr{P}_x, {}^{E}_{G}\mathscr{P}_y, {}^{E}_{G}\mathscr{P}_z, {}^{E}_{G}\varphi]^T$, where ${}^{E}_{G}\mathscr{P}_x$, ${}^{E}_{G}\mathscr{P}_y$, and ${}^{E}_{G}\mathscr{P}_z$ are the Cartesian coordinates of the origin of the coordinate frame ${}^{E}_{G}\mathscr{T}$, and ${}^{E}_{G}\varphi$ is the angle of rotation about a certain versor ${}^{0}_{E}\nu$ that transforms the orientation of frame ${}^{0}_{E}\mathscr{T}$ into ${}^{0}_{G}\mathscr{T}$. This versor is constant, only the angle changes.

Trajectory generation consists in generating the nodes of the four functions $\mathscr{P}_x(t)$, $\mathscr{P}_y(t)$, $\mathscr{P}_z(t)$, and $\varphi(t)$. Those nodes need to be time-stamped, thus a velocity profile along the trajectory has to be selected. A parabolic velocity profile is obtained using a cubic polynomial of time as a trajectory. The method of generating each of those functions is the same, so they are represented by a single symbol, $\mathscr{D}(t)$. The

overall duration of motion τ is provided by the planner. Thus position, velocity, and acceleration for $t \in [0, \tau]$ are defined as:

$$\mathscr{D}(t) = {}^{3}wt^3 + {}^{2}wt^2 + {}^{1}wt + {}^{0}w, \quad \frac{\mathrm{d}\mathscr{D}}{\mathrm{d}t} = 3\,{}^{3}wt^2 + 2\,{}^{2}wt + {}^{1}w, \quad \frac{\mathrm{d}^2\mathscr{D}}{\mathrm{d}t^2} = 6\,{}^{3}wt + 2\,{}^{2}w, \tag{14.3}$$

where ${}^{q}w$, $q = 0, \ldots, 3$, are polynomial coefficients. The boundary conditions are:

$$\mathscr{D}(0) = 0, \quad \mathscr{D}(\tau) = \mathscr{D}_\tau, \quad \left.\frac{\mathrm{d}\mathscr{D}(t)}{\mathrm{d}t}\right|_{t=0} = 0, \quad \left.\frac{\mathrm{d}\mathscr{D}(t)}{\mathrm{d}t}\right|_{t=\tau} = 0\,. \tag{14.4}$$

Using (14.4) and (14.3) we produce the coefficients of the cubic polynomial: ${}^{0}w = 0$, ${}^{1}w = 0$, ${}^{2}w = \frac{3\mathscr{D}_\tau}{(\tau)^2}$, ${}^{3}w = -\frac{2\mathscr{D}_\tau}{(\tau)^3}$. The resulting operational space cubic polynomial: $\mathscr{D}(t) = -\frac{2\mathscr{D}_\tau}{(\tau)^3}t^3 + \frac{3\mathscr{D}_\tau}{(\tau)^2}t^2$ is used to find $\mathscr{D}(\tau^k)$, $k = 1, \ldots, n$ (where n is the number of interpolating segments), resulting in ${}^{E}_{E^k}\mathscr{T}$ for $t = \tau^k$. The number n is chosen in such a way that the deviation from the thus obtained Cartesian straight line, due to configuration space interpolation within segments, is acceptable. For the purpose of transformation into the configuration space the homogeneous matrix ${}^{0}_{E^k}\mathscr{T} = {}^{0}_{E}\mathscr{T}\,{}^{E}_{E^k}\mathscr{T}$ is necessary, where ${}^{0}_{E}\mathscr{T}$ is the initial pose of the end-effector with respect to the global reference frame 0, ${}^{E}_{E^k}\mathscr{T}$ is the pose of the end-effector when at node k with respect to the initial end-effector pose. The initial pose, all intermediate poses, and the goal pose have to be transformed from the operational (Cartesian) space into the configuration space by using the inverse kinematic problem solution [9]:

$$\mathscr{M}^k = \mathrm{IKP}({}^{0}_{E^k}\mathscr{T}) = \left[\mathscr{M}^k_1, \mathscr{M}^k_2, \ldots, \mathscr{M}^k_m\right], \tag{14.5}$$

where m is the number of motors driving the manipulator ($m = 6$ in our case). For each motor we need to interpolate between those intermediate poses in the configuration space. Between the nodes $k = 0, \ldots, n$ again cubic polynomial with respect to time is used, hence a cubic spline trajectory results. The motion of each motor within each segment is described by:

$$\mathscr{M}^k_l(t) = {}^{3}w^k_l t^3 + {}^{2}w^k_l t^2 + {}^{1}w^k_l t + {}^{0}w^k_l, \quad k = 1, \ldots, n, \quad l = 1, \ldots, m, \tag{14.6}$$

where $t \in [0, \tau^k]$. τ^k is the i-th segment execution time. At each segment start the time is reset to zero. Thus the velocity and acceleration profiles are:

$$\mathscr{V}^k_l(t) = \frac{\mathrm{d}\mathscr{M}^k_l(t)}{\mathrm{d}t} = 3\,{}^{3}w^k_l t^2 + 2\,{}^{2}w^k_l t + {}^{1}w^k_l, \tag{14.7}$$

$$\mathscr{A}^k_l(t) = \frac{\mathrm{d}^2\mathscr{M}^k_l(t)}{\mathrm{d}t^2} = 6\,{}^{3}w^k_l t + 2\,{}^{2}w^k_l. \tag{14.8}$$

The position of a motor is defined for the nodes starting each segment, hence:

$$\mathscr{M}^k_l(0) = {}_{0}\mathscr{M}^k_l, \quad k = 1, \ldots, n\,. \tag{14.9}$$

The equations describing the terminal positions of the motor for each segment are:

$$\mathscr{M}_l^k(\tau^k) = {}_{\tau}\mathscr{M}_l^k, \quad k = 1,\ldots,n. \tag{14.10}$$

Moreover, the initial and the terminal velocity for the whole trajectory is set to zero:

$$\mathscr{V}_l^1(0) = 0, \quad \mathscr{V}_l^n(\tau^n) = 0. \tag{14.11}$$

The velocities at either side of each node must be equal to each other.

$$\mathscr{V}_l^k(\tau^k) = \mathscr{V}_l^{k+1}(0), \quad k = 1,\ldots,n-1. \tag{14.12}$$

The continuity of accelerations in the intermediate nodes is postulated:

$$\mathscr{A}_l^k(\tau^k) = \mathscr{A}_l^{k+1}(0), \quad k = 1,\ldots,n-1. \tag{14.13}$$

Formulas (14.9)–(14.13) provide $n+n+2+(n-1)+(n-1)$ equations with $4n$ unknowns. Their solution produces coefficients of cubic polynomials (14.6).

For $k = 1,\ldots,n$ using (14.6) and (14.9) we get ${}^0w_l^k = {}_0\mathscr{M}_l^k$. By further using (14.6), (14.10) and introducing ${}_{\Delta}\mathscr{M}_l^k = {}_{\tau}\mathscr{M}_l^k - {}_0\mathscr{M}_l^k$ we get:

$${}^3w_l^k(\tau^k)^3 + {}^2w_l^k(\tau^k)^2 + {}^1w_l^k\tau^k = {}_{\Delta}\mathscr{M}_l^k. \tag{14.14}$$

For $k = 1,\ldots,n-1$ using (14.12) and (14.7) we obtain:

$$3\,{}^3w_l^k(\tau^k)^2 + 2\,{}^2w_l^k\tau^k + {}^1w_l^k = {}^1w_l^{k+1}. \tag{14.15}$$

Equations (14.15) are supplemented by two formulas (14.11):

$$\begin{aligned} \mathscr{V}_l^1(0) &= 3\,{}^3w_l^1 0^2 + 2\,{}^2w_l^1 0 + {}^1w_l^1 = {}^1w_l^1 = 0, \\ \mathscr{V}_l^n(\tau^n) &= 3\,{}^3w_l^n(\tau^n)^2 + 2\,{}^2w_l^n\tau^n + {}^1w_l^n = 0. \end{aligned} \tag{14.16}$$

For $k = 1,\ldots,n-1$, using (14.13) and (14.8), we obtain:

$$6\,{}^3w_l^k\tau^k + 2\,{}^2w_l^k = 2\,{}^2w_l^{k+1}. \tag{14.17}$$

For $k = 1,\ldots,n-1$ equation (14.17) implies:

$${}^3w_l^k = \frac{1}{3\tau^k}\left({}^2w_l^{k+1} - {}^2w_l^k\right). \tag{14.18}$$

To compute ${}^3w_l^n$ we use (14.16).

$${}^3w_l^n = -\frac{1}{3(\tau^n)^2}\left(2\,{}^2w_l^n\tau^n + {}^1w_l^n\right). \tag{14.19}$$

Plugging (14.18) into (14.14) ${}^1w_l^k$ coefficients are computed for $k = 1,\ldots,n-1$:

$$ {}^1w_l^k = \frac{{}_\Delta\mathscr{M}_l^k}{\tau^k} - \frac{\tau^k}{3}\left({}^2w_l^{k+1} + 2\,{}^2w_l^k\right). \tag{14.20} $$

Substituting (14.19) into (14.14), for $k = n$, ${}^nw_l^k$ coefficient is computed:

$$ {}^1w_l^n = \frac{3\,{}_\Delta\mathscr{M}_l^n}{2\tau^n} - \frac{1}{2}\,{}^2w_l^n\tau^n. \tag{14.21} $$

The result (14.21) has to be substituted into (14.19):

$$ {}^3w_l^n = -\frac{1}{2\tau^n}\,{}^2w_l^n - \frac{{}_\Delta\mathscr{M}_l^n}{2(\tau^n)^3}. \tag{14.22} $$

Taking into account the condition for the continuity of velocity (14.15) and inserting into it (14.18) and (14.20) for $k = 1,\ldots,n-2$ we get:

$$ \frac{\tau^{k+1}}{3}\,{}^2w_l^{i+2} + \frac{2}{3}\left(\tau^{k+1}+\tau^k\right){}^2w_l^{k+1} + \frac{\tau^k}{3}\,{}^2w_l^k = \frac{{}_\Delta\mathscr{M}_l^{k+1}}{\tau^{k+1}} - \frac{{}_\Delta\mathscr{M}_l^k}{\tau^k}. \tag{14.23} $$

Taking into account (14.15) (for $k = n-1$) and inserting into it (14.21) we get:

$$ 3\,{}^3w_l^{n-1}(\tau^{n-1})^2 + 2\,{}^2w_l^{n-1}\tau^{n-1} + {}^1w_l^{n-1} + \frac{1}{2}\,{}^2w_l^n\tau^n = \frac{3\,{}_\Delta\mathscr{M}_l^n}{2\tau^n}. \tag{14.24} $$

Substituting (14.20) and (14.18) (for $k = n-1$) into (14.24) we obtain:

$$ {}^2w_l^n\left(\frac{2\tau^{n-1}}{3} + \frac{\tau^n}{2}\right) + {}^2w_l^{n-1}\frac{\tau^{n-1}}{3} = \frac{3\,{}_\Delta\mathscr{M}_l^n}{2\tau^n} - \frac{{}_\Delta\mathscr{M}_l^{n-1}}{\tau^{n-1}}. \tag{14.25} $$

The first formula of (14.16) and the result of (14.20) produce the following:

$$ \frac{2\tau^1}{3}\,{}^2w_l^1 + \frac{\tau^1}{3}\,{}^2w_l^2 = \frac{{}_\Delta\mathscr{M}_l^1}{\tau^1}. \tag{14.26} $$

Equation (14.26), equations (14.23), and equation (14.25) produce:

$$ \begin{cases} \frac{2\tau^1}{3}\,{}^2w_l^1 + \frac{\tau^1}{3}\,{}^2w_l^2 = \frac{{}_\Delta\mathscr{M}_l^1}{\tau^1} \\ \frac{\tau^{k+1}}{3}\,{}^2w_l^{i+2} + \frac{2}{3}\left(\tau^{k+1}+\tau^k\right){}^2w_l^{k+1} + \frac{\tau^k}{3}\,{}^2w_l^k = \frac{{}_\Delta\mathscr{M}_l^{k+1}}{\tau^{k+1}} - \frac{{}_\Delta\mathscr{M}_l^k}{\tau^k}, \quad k = 1,\ldots,n-2 \\ {}^2w_l^n\left(\frac{2\tau^{n-1}}{3} + \frac{\tau^n}{2}\right) + {}^2w_l^{n-1}\frac{\tau^{n-1}}{3} = \frac{3\,{}_\Delta\mathscr{M}_l^n}{2\tau^n} - \frac{{}_\Delta\mathscr{M}_l^{n-1}}{\tau^{n-1}} \end{cases} \tag{14.27} $$

The system of equations (14.27) can be expressed in matrix form $AW = M$, where

$$ W = \left[{}^2w_l^1, {}^2w_l^2, \ldots, {}^2w_l^k, \ldots, {}^2w_l^{n-1}, {}^2w_l^n\right]^T, \qquad M = $$

$$ \left[\frac{{}_\Delta\mathscr{M}_l^1}{\tau^1}, \frac{{}_\Delta\mathscr{M}_l^2}{\tau^2} - \frac{{}_\Delta\mathscr{M}_l^1}{\tau^1}, \ldots, \frac{{}_\Delta\mathscr{M}_l^{k+1}}{\tau^{k+1}} - \frac{{}_\Delta\mathscr{M}_l^k}{\tau^k}, \ldots, \frac{{}_\Delta\mathscr{M}_l^{n-1}}{\tau^{n-1}} - \frac{{}_\Delta\mathscr{M}_l^{n-2}}{\tau^{n-2}}, \frac{3\,{}_\Delta\mathscr{M}_l^n}{2\tau^n} - \frac{{}_\Delta\mathscr{M}_l^{n-1}}{\tau^{n-1}}\right]^T $$

and A assumes the form:

$$\begin{bmatrix}
\frac{2\tau^1}{3} & \frac{\tau^1}{3} & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\
\frac{\tau^1}{3} & \frac{2}{3}\left(\tau^2+\tau^1\right) & \frac{\tau^2}{3} & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
0 & \cdots & 0 & \frac{\tau^k}{3} & \frac{2}{3}\left(\tau^{k+1}+\tau^k\right) & \frac{\tau^{k+1}}{3} & 0 & \cdots & 0 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
0 & \cdots & \cdots & \cdots & \cdots & 0 & \frac{\tau^{n-2}}{3} & \frac{2}{3}\left(\tau^{n-1}+\tau^{n-2}\right) & \frac{\tau^{n-1}}{3} \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & \frac{\tau^{n-1}}{3} & \left(\frac{2\tau^{n-1}}{3}+\frac{\tau^n}{2}\right)
\end{bmatrix}.$$

Matrix $A = [a]$ need not be inverted to get W. We iteratively multiply row $k-1$ by $\frac{a_{k,k}}{a_{k-1,k}}$ and later subtract row k from the modified row $k-1$ (the result substitutes the previous equation $k-1$). In the last iteration a linear equation containing ${}^2w_l^1$ is obtained. Iterating from $k=2$ until $k=n$, through the equations, by inserting the values ${}^2w_l^{k-1}$ we compute the values of ${}^2w_l^k$, thus solving (14.27). The solution of (14.27) produces the values of $\left[{}^2w_l^1, {}^2w_l^2, \ldots, {}^2w_l^k, \ldots, {}^2w_l^{n-1}, {}^2w_l^n\right]^T$. Appropriate components of this vector have to be inserted into: formulas (14.21) and (14.22) to compute ${}^1w_l^n$ and ${}^3w_l^n$, respectively, the first formula of (14.16) and formula (14.18) (for $k=1$) to compute ${}^1w_l^1$ and ${}^3w_l^1$, respectively, and formulas (14.20) and (14.18) to compute ${}^1w_l^k$ and ${}^3w_l^k$ $(k=2,\ldots,n-1)$, respectively. Moreover, ${}^0w_l^n = {}_0\mathscr{M}_l^n$. Thus we have obtained a prescription for moving the motors. Formulas (14.6), (14.7), and (14.8) compute the position, velocity, and acceleration, respectively, of any motor at any instant of time, thus we can check whether the maximum velocity and acceleration limits are violated. Those computations are done by function ${}^{\Lambda}f'_{c_{c_{j_1}}}$ used in specification (14.31).

14.4 Embodied Agent a_{j_1}

Scarcity of space enables only partial description of one agent (a single transition function will be defined: Cartesian interpolation in operational space). Agent a_{j_1} has been chosen. Control of the PKM requires effector images. Agent a_{j_1} exchanges information with the coordinator, so transmission buffers $c_{T_{j_1 0}}$ have to exist. Through the input transmission buffer ${}_x c_{T_{j_1 0}}$ this agent receives from the coordinator the information about the current ${}^0_E\mathscr{T}$ and goal ${}^0_G\mathscr{T}$ end-effector pose as well as the total duration of motion τ and the number of interpolation nodes n. This agent does not need virtual sensor images as it does not use exteroceptors.

The input effector image ${}_x c_{e_{j_1}}$, among others, contains the following variables:

$${}_x c_{e_{j_1}} = \langle \ldots, {}_E\mathscr{M}, \beta, \varepsilon, \ldots \rangle . \tag{14.28}$$

This is the proprioceptive input from the effector. Variables ${}_{x}c_{e_{j_1}}[{}_{E}\mathscr{M}_l]$, $l=1,\ldots,6$, contain current motor positions, ${}_{x}c_{e_{j_1}}[\beta_l]$ show whether the motion has been executed, i.e., β_l contains *False* while the lth motor is in motion. Variables ${}_{x}c_{e_{j_1}}[\varepsilon_l]$ inform whether the EPOS2 buffer containing the commanded PVT data has empty space available: *False* – no space available, *True* – space is available.

The output effector image ${}_{y}c_{e_{j_1}}$, among others, contains the following variables:

$$ {}_{y}c_{e_{j_1}} = \langle \ldots, \mathscr{M}, \mathscr{V}, \Gamma, \mu, \ldots \rangle . \tag{14.29} $$

This image gets the results of trajectory generation. Variables ${}_{y}c_{e_{j_1}}[\mathscr{M}_l]$, ${}_{y}c_{e_{j_1}}[\mathscr{V}_l]$, and ${}_{y}c_{e_{j_1}}[\Gamma_l]$, contain the PVT data for the next segment of motion. Variables ${}_{y}c_{e_{j_1}}[\mu_l]$ decide whether CPPM (cubic position-profile motion), CAoBS (check availability of buffer space), or CMT (check motion termination) should be executed.

The internal data structures $c_{c_{j_1}}$, among others, contain:

$$ c_{c_{j_1}} = \langle \ldots, {}^{0}_{E}\mathscr{T}, {}^{0}_{G}\mathscr{T}, \tau, n, \lambda, \Lambda, \ldots \rangle . \tag{14.30} $$

$c_{c_{j_1}}[{}^{0}_{E}\mathscr{T}]$ and $c_{c_{j_1}}[{}^{0}_{G}\mathscr{T}]$ hold current and goal Cartesian end-effector pose, $c_{c_{j_1}}[\tau]$ and ${}_{c_{j_1}}[n]$ hold the duration of motion and the number of interpolation nodes, $c_{c_{j_1}}[\lambda_l]$ are the segment counters, and $c_{c_{j_1}}[\Lambda_{lk}]$ are the 6 lists of PVTs defining motion segments for each EPOS2. The coordinator provides the first 4 components of $c_{c_{j_1}}$ through ${}_{x}c_{T_{j_1 0}}$.

Now the transition functions and terminal conditions used by the pseudocode in Fig. 14.3 can be defined. We shall define the transition function ${}^{*}f'_{c_{j_1}}$ (superscript $*$ distinguishes this transition function from all others) in the decomposed form (14.1). Once ${}^{0}_{E}\mathscr{T}$, ${}^{0}_{G}\mathscr{T}$, τ, and n are obtained from the coordinator through ${}_{x}c_{T_{j_1 0}}$ the interpolation nodes and the segment execution times τ^k can be computed.

In the following the procedure outlined in Section 14.3 is represented by ${}^{*}f'_{c_{j_1}} \triangleq$

$$ \begin{cases} c^{i+1}_{c_{j_1}}[{}^{0}_{E}\mathscr{T}] = {}_{x}c^{i}_{T_{j_1 0}}[{}^{0}_{E}\mathscr{T}] & \text{for } i=i_0 \\ c^{i+1}_{c_{j_1}}[{}^{0}_{G}\mathscr{T}] = {}_{x}c^{i}_{T_{j_1 0}}[{}^{0}_{G}\mathscr{T}] & \text{for } i=i_0 \\ c^{i+1}_{c_{j_1}}[\tau] = {}_{x}c^{i}_{T_{j_1 0}}[\tau] & \text{for } i=i_0 \\ c^{i+1}_{c_{j_1}}[n] = {}_{x}c^{i}_{T_{j_1 0}}[n] & \text{for } i=i_0 \\ c^{i+1}_{c_{j_1}}[\lambda_l] = \begin{cases} 1 & \text{for } i=i_0 \\ c^{i}_{c_{j_1}}[\lambda_l]+1 & \text{for } i>i_0 \ \wedge \ {}_{x}c^{i}_{e_{j_1}}[\varepsilon_l]=\mathit{True} \end{cases} \\ c^{i+1}_{c_{j_1}}[\Lambda] = {}^{\Lambda}f'_{c_{j_1}}(c^{i}_{j_1}) & \text{for } i=i_0 \end{cases} \tag{14.31} $$

where $l=1,\ldots,6$ and i_0 is the instant of time in which the execution of this transition function within the currently executed behaviour is initiated. Transition function (14.31) produces 6 lists of PVT triplets, one for each of the microcontrollers: $\mathrm{P}^k_l = \mathscr{M}^i_l$, $\mathrm{V}^k_l = \mathscr{V}^k_l$, $\mathrm{T}^k_l = \tau^k$, where k is the running number of the interpolation

node, i.e., $k = 1,\ldots,n$, and $l = 1,\ldots,6$, are the numbers of the motors of the manipulator that take part in the motion. The results of those computations form the lists that are stored in the internal memory variable Λ. The transition function first forms those lists in memory and then subsequently uses them to load the PVT points into the EPOS2 motion microcontrollers, when space is available in their input data buffers. The above computational procedure is represented as follows.

In the very first step of the execution of the transition function (14.31) the values needed for interpolation computations are copied from the transmission buffer ${}_xc_{T_{j_1 0}}$ to the internal memory $c_{c_{j_1}}$. Simultaneously those values are also used for the purpose of computations (represented here by function ${}^{\Lambda}f'_{c_{c_{j_1}}}$) to produce the 6 lists of interpolation nodes, which are assigned to $c_{c_{j_1}}[\Lambda_l]$, $l = 1,\ldots,6$. Also all node counters are initiated to $c_{c_{j_1}}[\lambda_l] = 1$. Whenever any of the EPOS2 microcontrollers has available empty buffer space a PVT triplet is loaded into that buffer and so its respective counter needs to be incremented. When no space is available the counter retains its current value. The counters λ_l are incremented nearly synchronously, because all axes taking part in the motion are controlled by the same type of EPOS2 microcontrollers and each respective motion segment for all axes has the same duration. Nevertheless slight delays may be present, so this specification takes that into account. The specification (14.31), as well as the following specification (14.32), assume that if the value of a certain variable υ does not change in the period of time $i \to i+1$, the specification does not include the obvious statement $c^{i+1}_{c_{j_1}}[\upsilon] = c^{i}_{c_{j_1}}[\upsilon]$.

The definition of the transition function operating on the effector image: ${}^{*}f'_{c_{e_{j_1}}} \triangleq$

$$
\begin{cases}
{}_yc^{i+1}_{e_{j_1}}[\mathscr{M}_l] = c^{i}_{c_{j_1}}[\Lambda_{\lambda_l}[\mathscr{M}_l]] & \text{for } i > i_0 \ \wedge\ {}_xc^{i}_{e_{j_1}}[\varepsilon_l] = \mathit{True} \ \wedge\ c^{i}_{j_1}[\lambda_l] \leqslant c^{i}_{j_1}[n] \\
{}_yc^{i+1}_{e_{j_1}}[\mathscr{V}_l] = c^{i}_{c_{j_1}}[\Lambda_{\lambda_l}[\mathscr{V}_l]] & \text{for } i > i_0 \ \wedge\ {}_xc^{i}_{e_{j_1}}[\varepsilon_l] = \mathit{True} \ \wedge\ c^{i}_{j_1}[\lambda_l] \leqslant c^{i}_{j_1}[n] \\
{}_yc^{i+1}_{e_{j_1}}[\Gamma_l] = c^{i}_{c_{j_1}}[\Lambda_{\lambda_l}[\Gamma_l]] & \text{for } i > i_0 \ \wedge\ {}_xc^{i}_{e_{j_1}}[\varepsilon_l] = \mathit{True} \ \wedge\ c^{i}_{j_1}[\lambda_l] \leqslant c^{i}_{j_1}[n] \\
{}_yc^{i+1}_{e_{j_1}}[\mu_l] = \begin{cases}
\text{CPPM} & \text{for } i > i_0 \ \wedge\ {}_xc^{i}_{e_{j_1}}[\varepsilon_l] = \mathit{True} \ \wedge\ c^{i}_{j_1}[\lambda_l] \leqslant c^{i}_{j_1}[n] \\
\text{CAoBS} & \text{for } i > i_0 \ \wedge\ {}_xc^{i}_{e_{j_1}}[\varepsilon_l] = \mathit{False} \ \wedge\ c^{i}_{j_1}[\lambda_l] \leqslant c^{i}_{j_1}[n] \\
\text{CMT} & \text{for } i > i_0 \ \wedge\ c^{i}_{j_1}[\lambda_l] > c^{i}_{j_1}[n]
\end{cases}
\end{cases}
\tag{14.32}
$$

where CPPM stands for cubic position-profile motion, CAoBS – check availability of buffer space, CMT – check motion termination. If the EPOS2 has available buffer space and all interpolation nodes (PVT data) contained in Λ have not been transferred to it, the transfer of a single PVT data triplet (i.e., $\mathscr{M}_l$, $\mathscr{V}_l$ and Γ_l) takes place. If all of the nodes have not been transferred the microcontroller is being tested for available buffer space (by sending the CAoBS command). Once all nodes have been transferred the microcontroller is prompted for information regarding motion termination (by sending the CMT command). The terminal condition is defined as

$$
{}^{*}f'_{\tau_{j_1}} \triangleq {}_xc^{i}_{e_{j_1}}[\beta_l] \wedge \ldots \wedge {}_xc^{i}_{e_{j_1}}[\beta_l]\,. \tag{14.33}
$$

14.5 Conclusions

The specification forms the system implementation prescription. The full formal specification is too large to be presented here. However, this paper presents the most elaborate transition function in full. The definitions of other functions are much simpler. The proposed system definition method is very well suited to the task at hand. Many variants of the controller (different decompositions into agents and diverse attributions of resources and actions) can be considered at the specification stage. While working on the specification many problems that would be difficult to resolve at the implementation stage were spotted and dealt with without too much effort. Although the specification is not based on any particular programming language or operating system its statements can be easily converted into control programs. Thus we can conclude that the presented method of designing and describing multi-robot systems is very useful. Currently the most elaborate agent, the herein described agent a_{j_1}, has been implemented. Its real-time performance is satisfactory. The robot programming framework MRROC++ [5, 7] is used as the implementation tool.

Acknowledgements. The SwarmItFIX project is funded by the 7th Framework Programme (Collaborative Project 214678). We acknowledge the assistance of the European Commission and the partners of the project: DIMEC, University of Genova (Italy), Piaggio Aero Industries (Italy), Exechon (Sweden), ZTS VVÙ Košice a.s. (Slovakia), and Centro Ricerche FIAT (Italy).

References

1. Molfino, R., Zoppi, M., Zlatanov, D.: Reconfigurable swarm fixtures. In: ASME/IFToMM International Conference on Reconfigurable Mechanisms and Robots, pp. 730–735 (2009)
2. Neumann, K.: US patent number 4732525 (1988)
3. Slonneger, K., Kurtz, B.L.: Formal Syntax and Semantics of Programming Languages: A Laboratory Based Approach. Addison-Wesley Publishing Company, Reading (1995)
4. Szynkiewicz, W., Zielińska, T., Kasprzak, W.: Robotized machining of big work pieces: Localization of supporting heads. Frontiers of Mechanical Engineering in China 5(4), 357–369 (2010)
5. Zieliński, C.: The MRROC++ system. In: First Workshop on Robot Motion and Control (RoMoCo 1999), Kiekrz, Polska, pp. 147–152 (1999)
6. Zieliński, C.: Transition-function based approach to structuring robot control software. In: Kozowski, K. (ed.) Robot Motion and Control: Recent Developments. LNCIS, vol. 335, pp. 265–286. Springer, Heidelberg (2006)
7. Zieliński, C., Winiarski, T.: Motion Generation in the MRROC++ Robot Programming Framework. International Journal of Robotics Research 29(4), 386–413 (2010)
8. Zieliński, C., Winiarski, T., Trojanek, P., Kornuta, T., Szynkiewicz, W., Kasprzak, W., Zielińska, T.: Self reconfigurable intelligent swarm fixtures – Deliverable 4.2. FP7-214678. Warsaw University of Technology (2010)
9. Zoppi, M., Zlatanov, D., Molfino, R.: Kinematics analysis of the Exechon tripod. In: Proceedings of the ASME DETC, 34th Annual Mechanisms and Robotics Conference (MR), Montreal, Canada (2010)

Chapter 15
Indirect Linearization Concept through the Forward Model-Based Control System

Rafał Osypiuk

Abstract. The paper presents an effective method for indirect linearization based on a three-loop control system. This approach, due to robustness exhibited by the structure, enables one to determine a nonlinear component of the manipulated variable found by employing two forward plant models in pre-simulated loops. A constant-parameter character of the proposed approach and the control structure based on the classic PID loops ensures intuitive synthesizing and provides an interesting alternative to feedforward systems or adaptive control methods. Theoretical assumptions have been verified by simulation on a single-joint robot manipulator.

15.1 Introduction

The classic PID algorithm operated in a single-loop feedback system is undoubtedly the most common solution encountered to control industrial processes [19]. Its popularity is owed to favorable sensitivity [4] and robustness [5] to process plant perturbations, along with its simple synthesizing. However, if we have to do with significant process parameter variations brought about by strong static/dynamic nonlinear and/or time-variant properties exhibited by the process, the classic PID algorithm may be found to be ineffective to ensure a satisfying control performance. As a remedy, many more robust techniques have been developed [3]. However, frequently common to them are higher synthesis complexity [18] and not obvious stability conditions [8]. The exception is provided, among others, by systems of the Model-Based Control family, where the process is linearized directly through the use of the inverse process model. Here mention should be made of 1-DOF IMC (Model-Internal Control) [17], 2-DOF IMC [10], or feedforward systems [2]. Each

Rafał Osypiuk
Department of Control Engineering and Robotics, West Pomeranian University of Technology, ul. 26 Kwietnia 10, 71-126 Szczecin, Poland
e-mail: rafal.osypiuk@zut.edu.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 183–192.
springerlink.com

of them is based on the classic PID control, however with an additional compensator required.

There are not many processes mathematical identification of which leads to an inverse model. An exception is, for example, modeling of a kinematic chain, or generally, modeling of linear/rotational motion of a mass [16]. So, availability of inverse static/dynamic process models is not obvious. The total knowledge of the model requires using methods of global linearization [6], whereas adaptive methods of feedback linearization [14] are required to obtain a simplified form of the model.

To circumvent these difficulties an interesting modification of the 2-DOF IMC system, called Model-Following Control (MFC) [9] was developed in the late 1990s. It is a system composed of two PID loops where the process plant and its forward model are embraced. Although the MFC system synthesis has been demonstrated by an example of electric servo position control, the system may be used for the majority of processes, also if a significantly simplified model is employed [15]. The advantage of the MFC structure is its additional degree of freedom, which enables one, among others, to separate the process dynamics from the problem of suppression of disturbances [12]. However, the MFC system suffers from a substantial shortcoming, namely the plant forward model necessitates to be simplified in order for the classic PID control robustness limits not to be exceeded [11, 12].

To eliminate the above limitation the standard MFC system has been extended by an additional model loop. By this means the control structure is based on two forward models and enables one to determine the nonlinear manipulated variable in pre-simulated loops, where disturbances are nonexistent. Such a manipulation is possible due to high robustness offered by MFC. A closer look into the properties exhibited by MFC will be given later.

15.2 Problem Formulation

To examine limitations exhibited by the classic PID control, let us carry out its simple frequency-domain analysis under the effect of system disturbances z and output ones v. For the process plant P and the controller R the plant output/reference tracking is defined by $F_{PID}(j\omega) = \frac{R(j\omega)P(j\omega)}{1+R(j\omega)P(j\omega)}$. The reference r is tracked satisfactorily if $F(j\omega) \approx 1 \vee \omega \in \Omega_{r,z}$, which leads to the controller tuning condition $|R(j\omega)| \gg 1$. A similar requirement holds for suppression of system disturbances z governed by the transfer function $S_{PID}(j\omega) = \frac{1}{1+R(j\omega)P(j\omega)}$. The disturbances z are suppressed very well if $S(j\omega) \approx 0 \vee \omega \in \Omega_{r,z}$, which also leads to the condition $|R(j\omega)| \gg 1$. A different situation arises with output disturbances v governed by the transfer function $W_{PID}(j\omega) = \frac{R(j\omega)P(j\omega)}{1+R(j\omega)P(j\omega)}$, for which the condition $W(j\omega) \approx 0 \vee \omega \in \Omega_{r,z}$ leads to a contradictory requirement $|R(j\omega)| \ll 1$. Now, let us examine the sensitivity and the robustness offered by the single-loop structure. The sensitivity is defined by $Q(j\omega) = \frac{P(j\omega)}{F_{PID}(j\omega)} \frac{\delta F_{PID}(j\omega)}{\delta P(j\omega)}$. In view of $Q(j\omega) = S_{PID}(j\omega)$, for the sensitivity Q to be as small as possible, the following requirement is to be met: $|R(j\omega)| \gg 1$. However, the situation is different if robustness is concerned. Assuming the process

uncertainty is multiplicative $P(j\omega) = [1 + \Delta_p(j\omega)]\tilde{P}(j\omega)$, the condition for allowable perturbations the classic PID control may experience is [1]:

$$\left|\Delta_p(j\omega)\right|_{pid} < \left|\frac{1 + R(j\omega)\tilde{P}(j\omega)}{R(j\omega)\tilde{P}(j\omega)}\right|. \tag{15.1}$$

In order for the robustness Δ_p to variations in the process P structure and parameters to be as high as possible, the right-hand side of the inequality (15.1) should be as great as possible, which leads to the controller tuning condition $|R(j\omega)| \ll 1$. As seen from the foregoing, there are contradictory conditions the controller tuning rules are subject to. There is no other way to resolve the conflict between satisfying reference tracking, disturbance suppression, and robustness except by way of compromise.

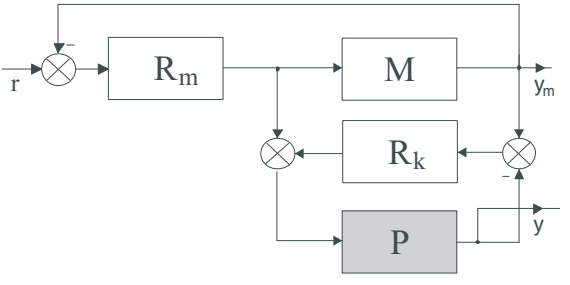


Fig. 15.1 Original form of the two-loop MFC system

Such a conflict does not exist in the case of the two-loop MFC system (Fig. 15.1) [12], which embraces a process model M together with its controller R_m and the process P together with a corrective controller R_k. However, the trouble with this system is that the process model is unduly complicated resulting in impaired control performance. This is directly apparent from the MFC transfer function [12]:

$$y_{mfc} = \frac{\frac{P}{M} + R_k P}{1 + R_k P} y_m, \qquad y_m = \frac{R_m M}{1 + R_m M} r. \tag{15.2}$$

As may be seen from (15.2), the plant output y_{mfc} tracks the output y_m of the model, which is embraced by the classic PID loop. Therefore, the control performance cannot be better than that obtained in the model loop. Since the robustness exhibited by the single-loop system (15.1) is limited, there is a need to simplify the model M, which results in impaired indirect process linearization [11]. To avoid the problem, the MFC system has been extended to the 3-loop case, which enable one to increase the global control system robustness with the use of two models of different complexity levels.

15.3 General Properties

In creating the 3-loop MFC structure advantage has been taken of the robustness offered by the classic PID feedback loop. Although the level of robustness exhibited by a single-loop is not impressive (15.1), yet repeated use of it in a parallel structure enables one to control more complex processes. Figure 15.2 depicts the proposed system, which is composed of the following blocks: R_1, R_2, and R_k are the classic PID controllers; $\widetilde{M_1}$ represents the linear process model; M_2 represents the nonlinear process model; P is the plant to be controlled.

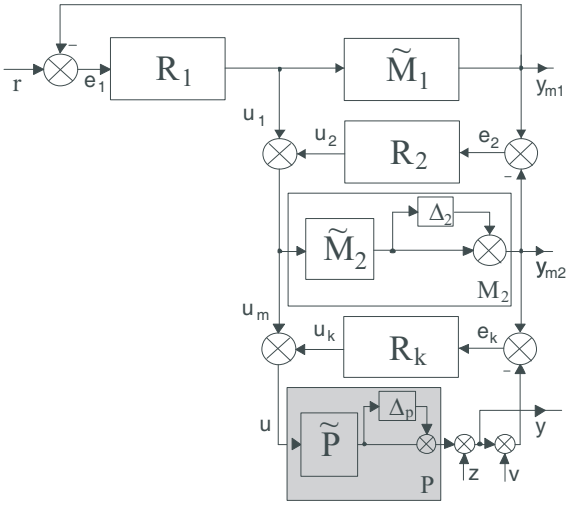


Fig. 15.2 General block diagram for the proposed 3-loop control structure

The general operation principle of the proposed system consists in finding component linear u_1 and nonlinear manipulated variable u_2; the components are then added up to give the resultant nonlinear manipulated variable u_m, which after having been corrected by u_k delivered by the R_k controller, is applied directly to the process. Hence, choosing appropriately the model complexity M_2 we can utilize the robustness exhibited by the pre-simulated 2-loops to control effectively strongly nonlinear processes.

The transfer function of the 3-loop MFC system has the form:

$$y = \underbrace{\frac{PR_1(1+R_2\widetilde{M_1})(1+R_kM_2)}{(1+R_1\widetilde{M_1})(1+R_2M_2)(1+R_kP)}}_{F} r + \underbrace{\frac{1}{(1+R_kP)}}_{S} z - \underbrace{\frac{R_kP}{(1+R_kP)}}_{W} v, \qquad (15.3)$$

where F describes tracking the reference r, S is responsible for suppression of system disturbances z, and W reflects suppression of output-related disturbances v.

Model $\widetilde{M_1}$ is embraced by a classic feedback loop. Since the output y_{m1} also represents the reference for the second loop, therefore the control performance obtained

in the first loop is decisive for the global control performance that may be observed at the plant output y. In view of the fact that the single-loop system (15.1) features limited robustness only, the $\widetilde{M_1}$ model should be sufficiently simplified. In regard to the second loop, as will be shown later, the model M_2 admits of its great complexity. This feature is employed to determine indirectly the nonlinear manipulated variable on the basis of the forward plant model.

Let us consider a dynamical model of the kinematic chain. Using the Newton-Euler or Lagrange identification method we get the motion equation, which after reversing assumes the form [16]: $\ddot{q} = M^{-1}(q)[\tau - C(q,\dot{q}) - B(q,\dot{q}) - G(q) - F(\dot{q})]$, where M is the manipulator inertia matrix, C is the matrix of centrifugal effects, B is the matrix of Coriolis effects, G is the vector of gravitational torques, F is the vector of friction forces, τ is the vector of joint torques, q, $\dot{q}$, $\ddot{q}$ are the vectors of the joint displacement, velocity, and acceleration terms, respectively. The above model presented in terms of gradual complexity might take on the following appearance:

$$\begin{cases} M_1 \rightarrow \ddot{q} = M^{-1}(q)[\tau - F(\dot{q})], \\ M_2 \rightarrow \ddot{q} = M^{-1}(q)[\tau - C(q,\dot{q}) - B(q,\dot{q}) - G(q) - F(\dot{q})]. \end{cases} \tag{15.4}$$

In such a case the structure of the 3-loop MFC can be employed for control. By this means the nonlinear manipulated variable u_m is created in the model loops as a sum of nonlinear components u_1 and u_2, which linearize indirectly the process to be controlled.

15.3.1 Quality of Control

Let us return again to the single-loop PID control and to its interesting feature called complementarity, manifesting itself in the equality $F_{pid}(j\omega) + S_{pid}(j\omega) = 1$, which is an obvious consequence of 1DOF control. This feature imposes some significant restrictions that are highlighted by the frequency-domain analysis. A desire to provide satisfactory tracking of the reference signal $F_{pid}(j\omega)$ for a given frequency $\omega_F \in (\omega_{1F}, \omega_{2F})$ determines automatically the range of frequencies $\omega_S \in (\omega_{1S}, \omega_{2S})$ for suppression of system disturbances $S_{pid}(j\omega)$, which may be incompatible with the process characteristics. In the proposed 3-loop MFC system the phenomenon of complementarity does not exist since $F(j\omega) + S(j\omega) = 1 - (W(j\omega) - F(j\omega))$ and there is the possibility to separate the task of tracking from that of disturbance suppression.

15.3.2 Robustness

To demonstrate the robustness exhibited by the proposed structure, the carried out analysis has been subdivided into two parts, i.e. that for the pre-simulated loops and that for the global structure. In reducing Fig. 15.2 to a single-loop system the block diagrams (Figs. 15.3, 15.4) shown below will be helpful in robustness analysis.

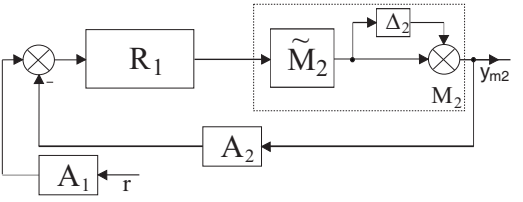


Fig. 15.3 General block diagram for the purpose of robustness analysis of two pre-simulated loops

To reduce the pre-simulated loops to a single-loop system (Fig. 15.3) two additional transfer functions A_1 and A_2 are to be introduced into the structure:

$$A_1 = \frac{1 + R_2\widetilde{M_1}}{1 + R_1\widetilde{M_1}}, \qquad A_2 = \frac{R_2}{R_1}. \tag{15.5}$$

In view of (15.1), the condition for allowable perturbations assumes the following form:

$$|\Delta_2(j\omega)| < \left| \frac{1 + R_2(j\omega)\widetilde{M_2}(j\omega)}{R_1(j\omega)\widetilde{M_2}(j\omega)} \right|. \tag{15.6}$$

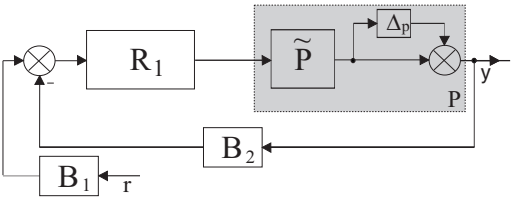


Fig. 15.4 General block diagram for the purpose of global structure robustness analysis

A comparison of stability conditions for the single-loop (15.1) and two-loop (15.6) systems shows a significant difference between them. While an increase in gain reduces the robustness of the classic system, the increased gain of the R_2 controller results in higher robustness of the proposed structure. This may be employed for indirect linearization of the nonlinear model M_2 in view of the fact that the pre-simulated loops are free from disturbances. In extending the analysis method to the whole structure, the latter may be reduced to an equivalent system by means of transfer functions B_1 and B_2 given as:

$$B_1 = \frac{(1 + R_2\widetilde{M_1})(1 + R_kM_2)}{(1 + R_1\widetilde{M_1})(1 + R_2M_2)}, \; B_2 = \frac{R_k}{R_1}. \tag{15.7}$$

The following condition

$$\left|\Delta_p(j\omega)\right| < \left|\frac{1+R_k(j\omega)\widetilde{P}(j\omega)}{R_1(j\omega)\widetilde{P}(j\omega)}\right| \tag{15.8}$$

defines permissible process perturbations Δ_p that may be increased by increasing the controller R_k gain, of course, accompanied by a limitation consisting in presence of system disturbances z or output-related ones v.

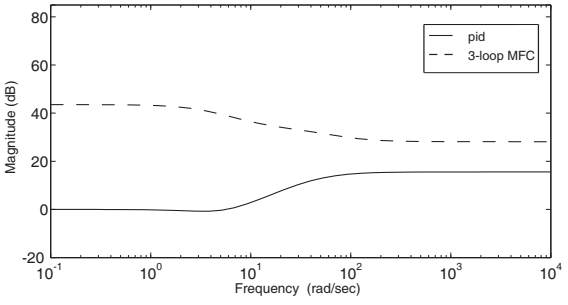


Fig. 15.5 Robustness to process parameters variations offered by the proposed 3-loop MFC system and the single-loop PID control

Figure 15.5 depicts the robustness to variations in process parameters and/or structure exhibited by the systems proposed. Employing the 3-loop structure makes it possible to improve many-fold the system robustness to plant perturbations, particularly over the range of low frequencies, i.e. the most interesting range of operation of a control system.

15.3.3 Stability

As may be easily noted in Fig. 15.2, the proposed control structure is composed of single-loop systems that are appropriately linked together. That is why intuition suggests that the global stability of a 3-loop MFC system will be secured if each of the model loops as well as the loop containing the process are stable. To convince oneself that this is true let us find the characteristic equation of the structure under study. Because of a less complicated mathematical description, we employ the corrective error value as that serving the purpose of stability analysis. The structure depicted in Fig. 15.2 is described by the following system of equations: $y = Pu$, $u = u_m + R_k e_k$, $e_k = y_{m2} - y$. Let the individual terms be given as rational functions: $P = \frac{n_p}{d_p}$, $R_k = \frac{n_{rk}}{d_{rk}}$, $y_{m2} = \frac{n_{y2}}{d_{y2}}$, and $u_m = \frac{n_{um}}{d_{um}}$. The corrective error takes the form:

$$e_k = \frac{d_{rk}(n_{y2}d_{rk}d_{um} - n_{rk}n_{um})}{d_{y2}d_{um}(d_p d_{rk} + n_p n_{rk})}. \tag{15.9}$$

By equating the denominator of (15.9) to zero we obtain the required characteristic equation of the 3-loop MFC system, which represents a sole source of

information about necessary conditions for system stability. As may be inferred from (15.9), the structure is stable if the polynomials d_{y2}, d_{um} and $d_p d_{rk} + n_p n_{rk}$ are Hurwitz ones, i.e. the real part of their every zero is negative. As one might expect, the stability of the second loop output y_{m2} and the stability of the manipulated variable u_m are required along with a stable corrective loop. Provision of stability for u_m is described as follows:

$$u_m = r\frac{R_1}{\prod_{i=1}^{2}(1+R_i M_i)} + \frac{R_2}{(1+R_2 M_2)}. \tag{15.10}$$

Substituting the equalities $M_i = \frac{n_{mi}}{d_{mi}}$, $R_i = \frac{n_{ri}}{d_{ri}}$ into (15.10) yields the characteristic equation for the pre-simulated loops of the following form:

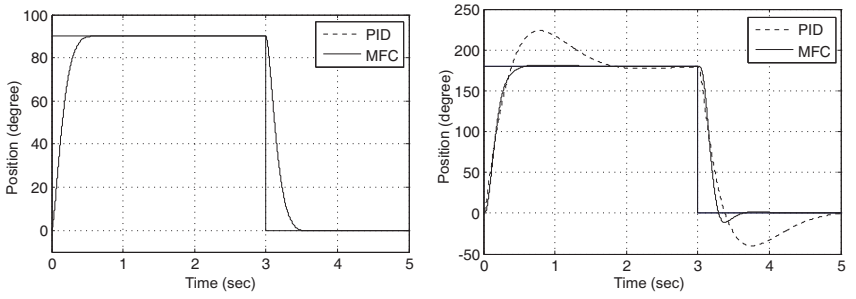
$$\prod_{i=1}^{2}(d_{mi}d_{ri} + n_{mi}n_{ri}) = 0. \tag{15.11}$$

According to (15.11), in order to provide a stable manipulated variable u_m, it is necessary that each simulated loop inclusive of the first reference loop be stable. Since on the one hand the stable output y_{m1} represents a necessary condition for stability of the second loop, and on the other hand stability of y_{m2} ensures global stability for the plant output y, therefore, in view of Eq. (15.10), stability is also required for models $\widetilde{M_1}$ and M_2.

15.4 Simulation

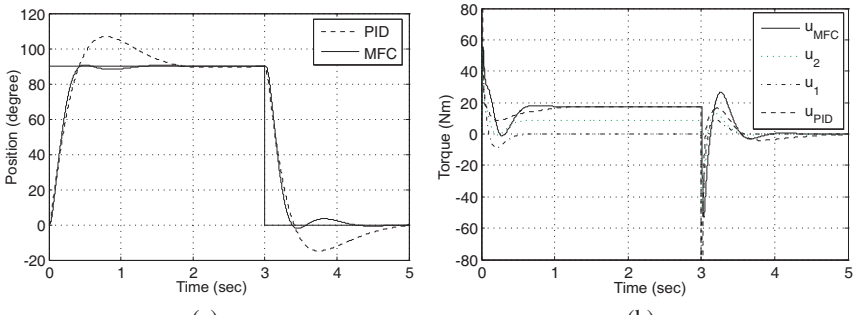
The aim of simulation tests described below is to provide an illustrative example of robustness of the proposed MFC system. Since the design of MFC systems is only slightly more complicated than that of the most frequently employed classic single-loop PID structure, therefore the control performance offered by MFC has been related to that offered by PID. *It should be stressed that the results presented here have been obtained for controller settings once found and remained unchanged for both structures, i.e. PID and MFC under torque constraint.* The tests have been carried out on a single-link robotic manipulator that is strongly nonlinear (statically, for the major part), and also may be time-variant. The control plant has been linearized around the operation point $q_0 = 90°$, and the controllers have been tuned up just for this case for all structures under study. The simulation result is displayed in Fig. 15.6(a). The control performance offered by PID and MFC is similar. However, the situation will be different if the operation point is changed to $q_0 = 180°$ (Fig. 15.6(b)).

Figure 15.7(a) illustrates the obtained control performance provided by the systems under comparison in the case when the link mass is increased 2-fold. Here the MFC systems also provide a significant improvement in reference tracking. In Fig. 15.7(b) the controller outputs are exemplified. It can be observed here how the form



(a) (b)

Fig. 15.6 (a) Controllers for each structure have been tuned up for the operation point $q_0 = 90^\circ$. (b) Control performance at the change in the operation point to $q_0 = 180^\circ$



(a) (b)

Fig. 15.7 (a) Robustness of the control systems at the 2-fold change in the link mass. (b) Manipulated variables in individual MFC system loops (u_1, u_2, u_{MFC}) and the PID loop

of the manipulated variable undergoes change, beginning from the simplest form u_1, through a more nonlinear one u_2, up to the general manipulated variable u_{MFC}.

15.5 Conclusion

In the paper an extension of the well-known two-loop MFC system to the three-loop case has been proposed. The additional degree of freedom makes it possible to calculate more effectively the nonlinear manipulated variable on the basis of the forward plant models. By this means one of the biggest limitations exhibited by MFC, namely the necessity of simplifying the plant model, is eliminated. The synthesis of the proposed structure is comparable to that of a single-loop PID system in simplicity, and does not require any special controller tuning rules. Additionally, intuitive stability conditions and ease of implementation make the system noteworthy, particularly when inverse properties of a strongly nonlinear plant are difficult to determine (if not impossible). In the second loop of the proposed structure that contains the M_2 model, the high R_2 controller gain enables tracking the linear model

$\widetilde{M_1}$ output for lack of disturbances, hence it linearizes indirectly the plant. A too high complexity of the model M_2 may however cause the manipulated variable u_2 to run in an impermissible way, when the R_2 gain is too high. If so, the M_2 can be written (if not impossible) in the form of several models of gradual complexity for a greater number of model loops to employ. In [11] an extension of the MFC system to an n-loop case under the above-mentioned assumption has been proposed.

References

1. Albertos, P.: Weighted model-based control. In: European Control Conference ECC 1993, Netherlands, pp. 1935–1939 (1993)
2. Coleman, B., Babu, J.: Techniques of model-based control. Prentice-Hall (2002)
3. Friedland, B.: Advanced control system design, 1st edn. Prentice-Hall (1996)
4. Garcia, D., Karimi, A., Longchamp, R., Dormido, S.: PID controller design with constraints on sensitivity functions using loop slope adjustment. In: IEEE American Control Conference, Minneapolis, Minnesota USA, June 14-16, pp. 268–273 (2006)
5. Ho, M.T., Lin, C.Y.: PID controller design for robust performance. IEEE Transaction on Automatic Control 48(8), 1404–1409 (2003)
6. Jordan, A.J., Nowacki, J.P.: Global linearization of non-linear state equations. International Journal of Applied Electromagnetics and Mechanics 19(1-4), 637–642 (2004)
7. Kirk, D.E.: Optimal control theory: An introduction. Dover Publications (2004)
8. Krstic, M., Kanellakopoulou, I., Kokotovic, P.: Nonlinear and adaptive control design. John Wiley and Sons (1995)
9. Li, G., Tsang, K.M., Ho, S.L.: A novel model following scheme with simple structure for electrical position servo systems. International Journal of Systems Science 29, 959–969 (1998)
10. Murad, G., Postlethwaite, I., Gu, D.W.: A discrete-time internal model-based H^∞ controller and its application to a binary distillation column. Journal of Process Control 7(6), 451–465 (1997)
11. Osypiuk, R., Finkemeyer, B., Wahl, F.M.: Multi-loop model following control for robot manipulators. In: Robotica, vol. 23(4), pp. 491–499. Cambridge University Press (2005)
12. Osypiuk, R., Finkemeyer, B., Skoczowski, S.: A simple two degree of freedom structures and their properties. In: Robotica, vol. 24(3), pp. 365–372. Cambridge University Press (2006)
13. Osypiuk, R., Kröger, T.: A three-loop model-based control structure: Theory and Implementation. International Journal of Control 83, 97–104 (2010)
14. Shin, H.-S., Choi, H.-L., Lim, J.-T.: Feedback linearisation of uncertain nonlinear systems with time-delay. IEEE Proc. Control Theory Appl. 153(6), 732–736 (2006)
15. Skoczowski, S.: Model following PID control with a fast model. In: Proc. of the 6th Portuguese Conference on Automatic Control, pp. 494–499 (2004)
16. Spong, M.W., Hutchinson, S., Vidyasagar, M.: Robot modeling and control. John Wiley & Sons (2005)
17. Tan, G.T., Chiu, M.S.: A multiple-model approach to decentralized internal model control design. Chemical Engineering Science 56(23), 6651–6660 (2001)
18. Tao, G.: Adaptive control design and analysis. Wiley-IEEE Press (2003)
19. Visioli, A.: Practical PID Control (Advances in industrial control). Springer, Heidelberg (2006)

Chapter 16
Research Oriented Motor Controllers for Robotic Applications

Michał Walęcki, Konrad Banachowicz, and Tomasz Winiarski

Abstract. Motor controllers are vital parts of robotic manipulators as well as their grippers. Typical, commercial motor controllers available on the market are developed to work with high level robot industrial controllers, hence their adaptation to work as a part of a scientific, experimental robotic system is problematic. The general concept of research oriented motor controllers for robotic systems is presented in this article as well as an exemplary gripper and manipulator application based on this concept.

16.1 Introduction

The majority of robot manipulators and mobile platforms are driven by electric motors, hence motor drivers are vital parts of industrial as well as scientific robots. There is a wide range of commercial motor controllers available on the market. Those controllers are often called ,,motion controllers" due to their capability of generating motion trajectories executed on a slave motor. The customer can easily select an appropriate driver for his needs in the case of typical, industrial systems. The selection will be guided by the type of the controlled motor, its power, the types of position sensors and their interfaces. The units are equipped with a number of safeguards in case of a hardware failure or human error. Manufacturers provide specific software that allows launching the drivers and performing calibration and automatic regulator tuning within a few minutes. The representative industrial motion controllers for robotic applications are briefly described in the following:

- Elmo SimplIQ product family [1]. Controllers of the SimplIQ series are dedicated to brush and brushless motors with maximum output power of 200W to 1800W. There are many options available for feedback (incremental encoder,

Michał Walęcki · Konrad Banachowicz · Tomasz Winiarski
Institute of Control and Computation Engineering, Warsaw University of Technology,
ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: mw@mwalecki.pl, tmwiniarski@gmail.com

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 193–203.
springerlink.com

Halls, resolver, interpolated analog encoder, tachometer and potentiometer, absolute encoder, Stegmann absolute encoder). The controller is equipped with current, velocity, and position regulators. RS-232, CANopen, DS 301, DS 402 digital interfaces are also available.
- Maxon EPOS2 product family [3]. The EPOS2 are easy to use compact motor drivers. They are directed to work with drives manufactured by Maxon, equipped with standarized feedback (Hall, encoder) interfaces. The most powerful device in the family, EPOS2 70/10, is capable of driving a motor with suply voltage 70VDC and continuous current 10A. It has current, speed, and position regulators implemented.
- National Instruments [4] offers a rich set of components to build motor control systems in specialized configurations.

Commercial motor controllers are selected to drive specific motors working in particular conditions. The expected parameters of the driver are well determined. Conditions of research oriented motion controllers are different. The most required features are versatility and open specification. Affordable prices are also important. Unlike industry, academic institutions can not spend a few thousands EUR on one motor driver.

A key feature of a research oriented controller is the ease of adapting it to an ongoing task. Therefore, the controller should meet the following requirements:

- The ability to control drives of a wide range of power, featuring a variety of sensors.
- An open specification that allows modifications of the control algorithms or introducing additional mechanisms.
- Ability to implement regulators in both the controller and the primary computer or to transfer some control tasks onto the computer (e.g. a cascade controller structure with internal, fast current control in the motor driver and external, slower position controller in the PC). Often the PC implementation of controllers is preferred due to ease of implementation, monitoring, and modification activities.
- High speed communication and short-term engagement with the computer is essential for running high-performance controllers with the feedback loop closed on a PC.
- Typical communication interface, allowing users to easily add additional drives to the system and to handle all the drives in a uniform way using software frameworks for robotic tasks.
- Access from a PC to parameters such as the position of the drive, the motor current and the state of limit switches.
- A design based on standard, ”off the shelf” components. The selection of electronic components significantly affects the cost of the solution and ease of maintenance or repair.

Commercially available industrial controllers do not meet those requirements, because they:

- Do not give access to the control algorithm structure.
- Use communication busses (CAN, MODBUS) that are expensive in the case of communication parallelization with high speed data transfers.
- Use a protocol which is not publicly available (CiA 402) or requires dedicated communication hardware (EtherCAT).
- Do not provide interfaces for additional sensors.
- Do not offer a possibility of custom control algorithm implementation (e.g. impedance control).

There are solutions for research purposes such as products of dSPACE company [2]. In the field of motor control dSPACE provides a set of tools for rapid prototyping of electronic control units. They developed a variety of modules containing half-bridge and full-bridge drivers for electric motors, real-time processor systems, and I/O extensions cards. A specialized real-time interface provides a link between dSPACE hardware and development software such as Matlab, Simulink, or Stateflow. Other research oriented motor controllers (e.g. DLR [8]) provide the necessary capabilities, but they are not available for purchase and the provided documentation is often insufficient for reimplementation.

Another way to develop an open system, adequate to current needs, is to use a programmable logic controller (PLC). PLCs may be equipped with various I/O interfaces and power stages. Their program implementation is easy using dedicated environments provided by manufacturers.

The main drawback of using one of the solutions mentioned above is their high cost that is not adequate to the efficiency of the final motion controller. A huge part of the resources of the main control unit are used to provide versatility, compatibility with many other units and ease of algorithm implementation, whereas in many cases the main goal is an economic, robust, efficient control system with a high speed PC link. Furthermore, the commercial electric drive control systems usually have a form of standarized box-shaped modules. In applications such as a gripper controller small dimensions, custom shape and proper cables placement are significant issues.

Hence, we decided to create a general solution (Section 16.2) suitable both for gripper controller application (Section 16.3) and manipulator control (Section 16.4) of scientific systems.

16.2 General Concept of the Controller

The single motor motion controller structure for research oriented applications is presented in Fig. 16.1. It consists of two main blocks, A and B. Block B consists of a power amplifier and an optional current controller. In general, it has to be implemented in a dedicated motor controller board, because current control needs a high frequency loop. The power amplifier supplies the motor and measures the current in the motor circuit. The measured current can act as a feedback for the current controller. Additionally the measured current value can be used e.g. to detect overcurrent in an external control loop in block A.

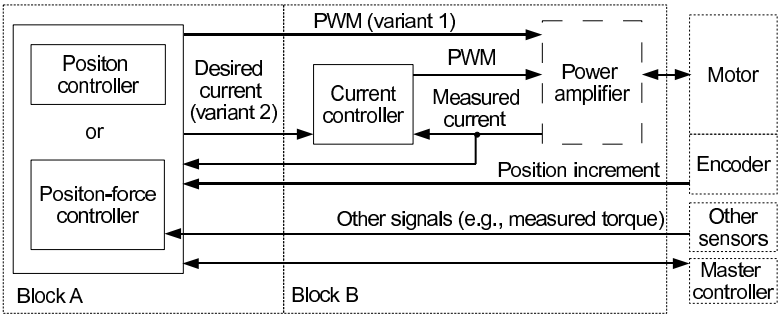


Fig. 16.1 General structure of single joint controller

In general block A consists of a position or position-force (position-torque) control loop. The term position-force control is a common name for the type of control where information on both the position and the force is utilized to produce the controller output. It should be noted that in the case of a motor with a rotating shaft the term position-torque is more precise because it corresponds to the torque produced on the motor shaft. The controller in block A operates under supervision of a master controller, e.g. Orocos [7] or MRROC++ [11]. In the position control variant the controller utilizes the information from the motor shaft encoders, i.e. the motor shaft position. The position-force control additionally needs the measured torque or force as an input, hence other sensors or measured current are used for this purpose. Both the position and position-force controllers can produce PWM to directly control the power amplifier (variant 1 in Fig. 16.1) or the desired current (variant 2 in Fig. 16.1) as the desired value for the internal current controller. Block A can be implemented in the motor controller board, e.g. when high control bandwidth is needed or in the PC, if the ease of controller implementation or high computation power is suitable.

To maintain the controller variant with block A implemented in the PC and block B implemented in the motor driver board, high frequency (in the meaning of master-slave rendezvous frequency) and high speed communication is needed between the PC and the controller board. It should be noted that the PC typically communicates with the number of boards equal to the number of manipulator (gripper) joints. It causes the ease of communication parallelization as the main assumption. Taking into account the above assumptions, the optimal communication interface was considered. The CAN communication period is 400us and a packet contains only 8 bytes of data, which makes it unsuitable to research applications. The parallelization is expensive because a typical PC CAN interface card is equipped with two CAN ports. EtherCAT is Fast Ethernet based protocol with the communication period of 100us with all devices on the bus, but the requirement of using a dedicated hardware interface makes the implementation of a custom slave device highly complicated. SERCOS is a very rare used optical bus. It provides very high performance at a very high cost. RS485 and RS422 can be implemented in almost any microcontroller and it can be simply parallelized if more bandwidth is required using an

inexpensive multi port PC interface card. Hence the RS485 and RS422 were chosen to communicate between the motor driver board and the PC computer.

16.2.1 General Structure of the Research Dedicated Motor Driver

The possibility of generating the controller structures in various variants is the main assumption for the hardware oriented general structure of the research oriented motor driver presented in Fig. 16.2. It consists of a microcontroller as the main part, its inputs (e.g. encoders to measure the motor shaft position), a power amplifier to supply the motor and a master controller (typically a PC). The structure takes into account all of the needed interfaces, i.e. an efficient communication interface via RS485 to the PC and an interface to the encoders. Additional custom sensors and output interfaces are provided, which significantly distinguishes the proposed solution from commercial products.

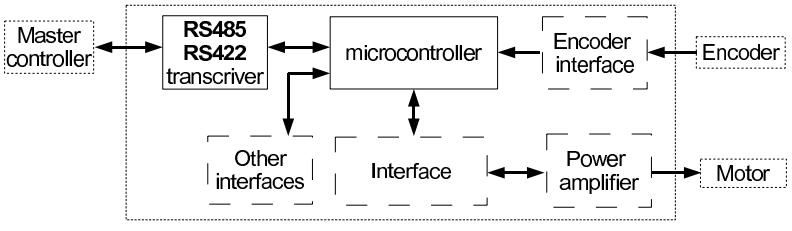


Fig. 16.2 General structure of the research dedicated motor driver

To fulfill the assumptions we chose the Microchip dsPIC Digital Signal Controller as the base of the driver. It provides integration of high diversity of peripherals with the computing power of a DSP. That gives us the possibility to implement sophisticated control algorithms [6]. It also provides peripherals dedicated for motion control (quadrature encoder interface, motor control PWM, fast ADC). In combination with different power stages and sensor interfaces it gives us the ability to control different actuators, ranging from sub-watt DC motors to kW BLDC motors.

16.2.2 Control Software

The flow of the servo controller's program is clocked with interrupts induced by a hardware timer/counter. The interrupt routines contain procedures for calculations of two PID regulators: position and motor current. These regulators can be coupled in a cascade or work independently, depending on the servo driver's operation mode.

The dsPIC microcontroller is equipped with a motor control PWM module. One of its features that we took advantage of is a special event comparator. It is used for scheduling the analog-digital converter's (ADC) trigger exactly in the middle of the power stage PWM signal's duty cycle. This ensures that at the moment of sampling the measured value of the motor's current there is no interference with switching the H-bridge in the power stage.

For acquiring the motor's position a specialized quadrature encoder interface is used. It provides functions such as generating interrupts on the encoder index signal, detecting encoder signal failures and a digital signal filter.

Communication with the PC is implemented using a UART interface. To gain high speed transmission, avoiding disruptions of other driver's functions, direct memory access (DMA) was utilized. It enables direct transfers of data between the UART and the RAM memory, not engaging the processor's core.

16.2.3 Communication between the Motor and the Master Controllers

The communication protocol is general and provides the ability to query device types and available control modes. Depending on those parameters, the controller accepts all or a subset of commands and there is a minimal subset of commands that have to be supported by every device:

- device info – reads information on the target device hw/sw version, device type, available control modes;
- device status – reads the state of the target device current control mode, motor position, measured current, user defined data.
- set mode – sets the control mode;

A device must support one or more predefined or user defined control modes. Each control mode defines a set of commands.

- PWM – direct control of the PWM generator, used for implementation of the PC based control loop (in Fig. 16.1 block A is implemented in the PC, without an internal current controller in block B (variant 1)). It is mainly used for rapid-prototyping of control algorithms.
- current – provides control of the motor's current (in Fig. 16.1 block A is implemented in the PC, with the internal current controller in block B (variant 2)). It is very similar to the PWM mode and is used for implementation of the external control loop.
- position – control of the motor's position. It is implemented on the motor driver, if the device supports it (in Fig. 16.1 block A and block B are implemented in the motor driver).

Each communication cycle is initiated by the master controller (the PC). The slaves (the motor drivers) send data only as an answer to a command, hence the master-slave rendezvous communication paradigm is used.

16.3 Compact Gripper Controller

The multi-finger gripper developed at the Institute of Control and Computation Engineering consists of three fingers and seven joints. Each joint is actuated by a small DC motor (Maxon RE-max 13: 12 V, 173 mA continuous current) and consists of an incremental encoder on the motor shaft, an absolute magnetic encoder in the joint and an absolute encoder measuring the displacement of the compliant clutch (Figs. 16.3 and 16.4). It uses control electronics integrated in the gripper similarly to other popular grippers [8, 5].

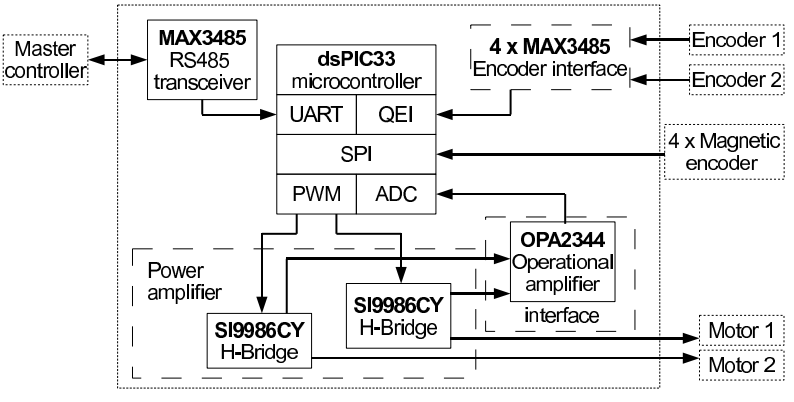


Fig. 16.3 Gripper controller building blocks, where here and in Fig. 16.5: UART - Universal Asynchronous Receiver Transmitter, QEI - Quadrature Encoder Interface, SPI - Serial Peripheral Interface, PWM - Pulse Width Modulator, ADC - Analog Digital Converter

A single microcontroller controls two joints and is interfaced to two incremental encoders by RS485 line receivers and four magnetic encoders on the SPI bus. It also controls two motors via two integrated H-bridges.

The gripper controller provides standard control modes (PWM, current, position), described in Section 16.2.3, and a dedicated joint impedance control mode. It also provides additional commands, only mentioned here: impedance move, reading from joint absolute encoder, time synchronization.

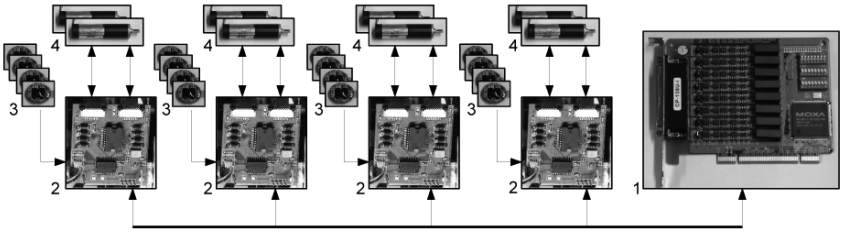


Fig. 16.4 Structure of the gripper control hardware: 1 – RS485 interface card, 2 – joint controller, 3 – absolute encoders, 4 – DC motor with an incremental encoder

16.4 Manipulator Controller

The robotics laboratory of the Institute of Control and Computation Engineering is equipped with two modified IRp6 manipulators. The manipulators' axes are driven by 35V, 28A, DC motors. For the purposes of position measurement, synchronization, and constraint detection each axis is equipped with an incremental optoelectronic encoder, a mechanical synchronization switch, and limit switches.

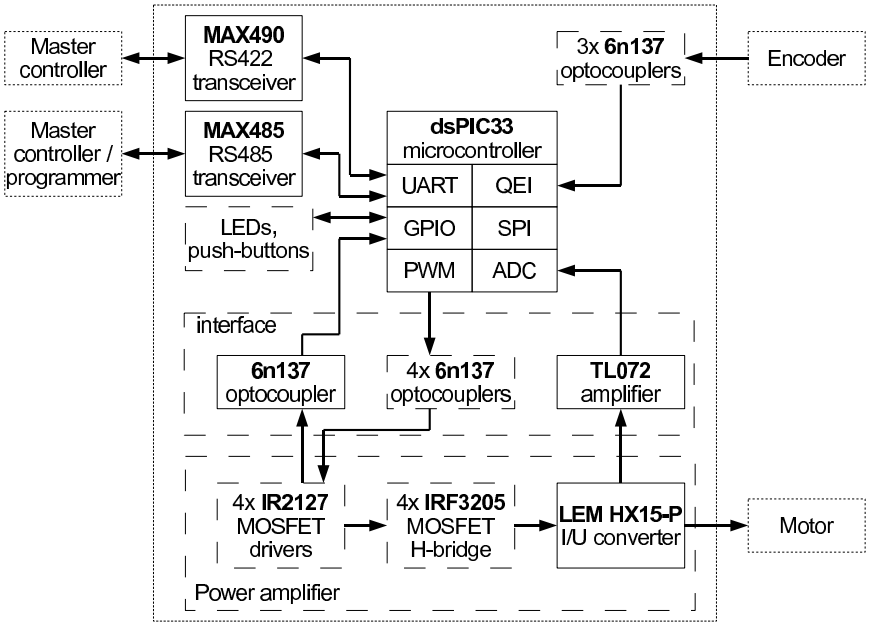


Fig. 16.5 Architecture of the manipulator axis driver, where: GPIO - General Purpose Input/Output

Figure 16.5 shows the manipulator controller's architecture. The dsPIC microcontroller is connected to external position sensors via optocouplers to minimize entering noise and to avoid the microcontroller's damage in case of a sensor failure. For the same reasons the power stage is galvanically isolated from the dsPIC, using optocouplers. The MOS field effect transistors of the H-bridge in the power stage are driven by specialized ICs – International Rectifier IR2127. These devices not only provide sufficient current for opening and closing MOSFETs, but also signal incorrect maintenance of the transistors to the microcontroller. The motor's current is measured using LEM HX series Hall effect sensor.

Two Maxim MAX490 and MAX485 converters provide communication with the computer through full duplex RS422 and half duplex RS485 interfaces. By default, RS422 is used for high speed communication with the application running on the PC, while RS485 is for programming the microcontroller and manual operation by typing commands on a text terminal. This allows linking a number of motor controllers with one bus and referring to each controller using its own address, set mechanically with a rotary switch.

The manipulator controller provides standard control modes (PWM, current, position), described in Section 16.2.3. Manual axis movement is also possible by using push-buttons placed on the controller's front panel. This is useful when there is a need to change the manipulator's position without running the PC application.

16.5 Conclusions and Experimental Results

The common approach to axis controllers' development of scientific robotic systems (Section 16.2) leads to successful applications. Both the gripper (Section 16.3) and the manipulator (Section 16.4) hardware controllers were attached as an effector part of `MRROC++` [10] based robot control system, that can be currently executed under supervision of both QNX and Linux operating systems, depending on the requirements. Finally the research of manipulator applications, as presented in [11, 9], or of the system containing both the manipulator and gripper was performed. Exemplary experimental results are presented in Fig. 16.6. The proposed motor drivers can be compared in a synthetic way to commercial solutions and the DLR gripper driver (Table 16.1).

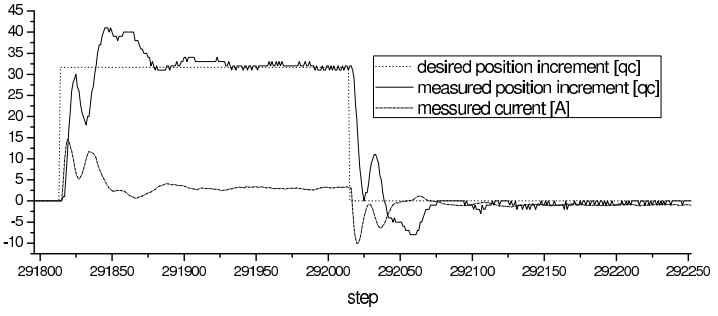


Fig. 16.6 Exemplary results of the motion of an IRp6 manipulator column under the supervision of the position controller. The desired position increment was initially instantly increased from 0 to 31.6 encoder pulses (qc) per controller step duration (2ms). Then it was decreased again to 0. Finally the controller slowly reduces the summarized position error under strong influence of static friction

Table 16.1 Synthetic comparison of the proposed motor drivers, commercial solutions and the DLR gripper driver

	Maxon Epos2 family	Elmo SimplIQ family	NI 734x	DLR LWR III	Proposed drivers
current controller sample rate	10kHz	max 16kHz	max 16kHz	40kHz to 80kHz	20kHz
position controller sample rate	1kHz	max 4kHz	max 16kHz	3kHz	1kHz
min. communication cycle period	(CANopen)	(CANopen, EtherCAT)	(PCI)	(SERCOS)	500 μs (RS-422)
open firmware	no	no	no	no	yes
modifiable firmware	no	no	yes, NI LabView	no	yes
integrated power amplifier	yes	yes	no	yes	yes
power amplifier max. continuous current, voltage	10A, 70V	180A, 390V	NA	no data	NA
custom sensors interface	no	no	yes - with dedicated cards	yes	yes

Acknowledgements. This work was founded in part by the Ministry of Science and Higher Education grant N514237137. The authors thank Karol Rejmanowski for language support.

References

1. Brochure - Servo Drives Catalog-Elmo SimplIQ (2009), http://www.elmomc.com
2. dSPACE website (2011), http://www.dspaceinc.com/
3. Maxon Product Range (2011), http://www.maxonmotor.com
4. National instruments on-line catalogue (2011), http://www.ni.com
5. Shunk on-line catalogue (2011), http://www.schunk.com
6. Bielewicz, Z., Debowski, L., Lowiec, E.: A DSP and FPGA based integrated controller development solutions for high performance electric drives. In: Proceedings of the IEEE International Symposium on Industrial Electronics, ISIE 1996, vol. 2, pp. 679–684. IEEE (1996)
7. Bruyninckx, H.: The real–time motion control core of the OROCOS project. In: Proceedings of the IEEE International Conference on Robotics and Automation, Taipei, Taiwan, pp. 2766–2771. IEEE (2003)
8. Liu, H., Wu, K., Meusel, P., Seitz, N., Hirzinger, G., Jin, M., Liu, Y., Fan, S., Lan, T., Chen, Z.: Multisensory five-finger dexterous hand: The DLR/HIT Hand II. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2008, pp. 3692–3697. IEEE (2008)
9. Wawrzyski, P., Winiarski, T.: Manipulator trajectory optimization based on learning techniques. In: XI Krajowa Konferencja Robotyki – Problemy Robotyki, Oficyna Wydawnicza Politechniki Warszawskiej, vol. 2, pp. 485–494 (2010) (in Polish), Optymalizacja trajektorii manipulatora w oparciu o metody uczenia się
10. Winiarski, T., Zieliński, C.: Specification of multi-robot controllers on an example of a haptic device. In: Kozłowski, K.R. (ed.) Robot Motion and Control 2009. LNCIS, vol. 396, pp. 227–242. Springer, Heidelberg (2009)
11. Zieliński, C., Winiarski, T.: Motion Generation in the MRROC++ Robot Programming Framework. International Journal of Robotics Research 29(4), 386–413 (2010)

Chapter 17
Efficient and Simple Noise Filtering for Stabilization Tuning of a Novel Version of a Model Reference Adaptive Controller

József K. Tar, Imre J. Rudas, Teréz A. Várkonyi, and Krzysztof R. Kozłowski

Abstract. For the adaptive control of imprecisely and partially modeled nonlinear systems a novel class of "Model Reference Adaptive Controllers" (MRAC) using "Robust Fixed Point Transformation" (RFPT) was recently proposed. It applies convergent iterative "learning sequence" of a local and bounded region of convergence. To make it competitive with the Lyapunov function based techniques, which normally guarantee global stability, this new approach was complemented with additional parameter tuning keeping the iterative sequence in the vicinity of the solution of the control task. For an "*n*-th order system" the novel controller also observes the *n*-th time-derivatives of the controlled coordinates that may make the adaptive tuning sensitive to measurement noises. To reduce this effect a simple noise filtering method is recommended. Extended simulation results are presented for the adaptive control of a pendulum of an indefinite mass center point determined by the presence of dynamic coupling with an internal degree of freedom that is only approximately modeled by the controller. It is neither observable nor directly controllable. It was found that the simple filtering recommended considerably reduced the sensitivity of the novel method proposed and substantiated the expectations that it could be applied in practice.

József K. Tar · Imre J. Rudas
Institute of Intelligent Engineering Systems, John von Neumann Faculty of Informatics, Óbuda University, Bécsi út 96/B, H-1034 Budapest, Hungary
e-mail: tar.jozsef@nik.uni-obuda.hu, rudas@uni-obuda.hu

Teréz A. Várkonyi
Applied Informatics Doctoral School, John von Neumann Faculty of Informatics, Óbuda University, Bécsi út 96/B, H-1034 Budapest, Hungary
e-mail: varkonyi.terez@phd.uni-obuda.hu

Krzysztof R. Kozłowski
Chair of Control and Systems Engineering, Poznań University of Technology, ul. Piotrowo 3a, 60-965 Poznań, Poland,
e-mail: krzysztof.kozlowski@put.poznan.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 205–214.
springerlink.com

17.1 Introduction

The *MRAC* technique has been a popular and efficient approach in the adaptive control of nonlinear systems for a long time. Countless papers can be found on the application of MRAC from the early nineties until present day, e.g. [1]– [4]. The mainstream of the adaptive control literature in the early nineties used parametric models and applied Lyapunov's "direct method" either for model parameter tuning (e.g. the "*Adaptive Inverse Dynamics Controller*" and the "*Slotine–Li Adaptive Controller*", cf. [5]) or, as mainly in the case of MRAC controllers, for tuning of certain control parameters without dealing too much with system models. The MRAC technique was successfully applied e.g. in motion control [2] of manipulator arms as well as for teleoperation purposes [3]. It was also combined with the use of "*Artificial Neural Networks*" [4].

The essence of the idea of the *MRAC* is the transformation of the actual system under control into a well behaving reference system (reference model) for which simple controllers can be designed. In practice the *reference model* used to be stable linear system of constant coefficients. To achieve this simple behavior normally special adaptive loops have to be developed. The scheme at the LHS of Fig. 17.1 describes a "traditional" realization in which the block "Parameter Tuning" is designed by the use of a Lyapunov function.

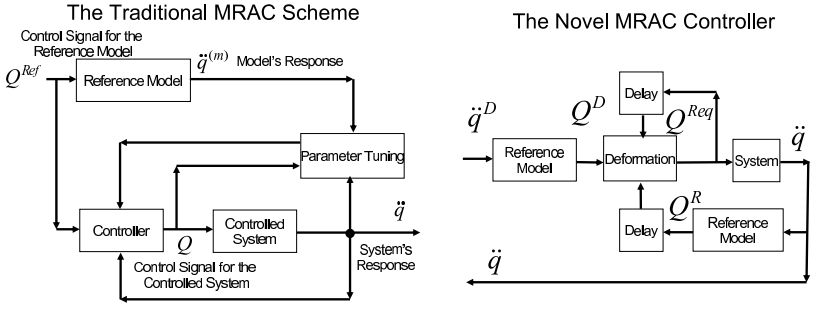


Fig. 17.1 The block scheme of a possible "traditional" (LHS) scheme with negative feedback in the lower branch, and the "novel" (RHS) MRAC controller constructed for Classical Mechanical systems, i.e. in the role of the "system response" the second time-derivatives of the "generalized coordinates" ($\ddot{q}$), and in the role of the "physical agent" that can be exerted on the controlled system to produce that appropriate response (the "generalized forces" Q)

Normally it is a difficult task to find an appropriate Lyapunov function and guarantee its non–increasing nature that leads to complicated estimations while deducing the appropriate tuning rule. To overcome the complexity of Lyapunov function based techniques alternative approaches are also present in the literature. The potential application of iterative fixed point transformations with local basin of attraction of convergence was studied for the adaptive control of "smooth" physical systems. The robustness of this approach was later improved in [6]. These controllers worked

on the basis of the concept of "*Complete Stability*" that is widely used in connection with the operation of Cellular Neural Networks as well. The method applied the concept of the "*expected – realized system response*" and was found to be appropriate to also developing MRAC technique for "*Single Input – Single Output (SISO)*" systems [7], and later was extended to "*Multiple Input – Multiple Output (MIMO)*" systems [8].

Assume that on purely kinematical basis we prescribe a trajectory tracking policy that needs a *desired joint coordinate acceleration* as $\ddot{q}^D$. From the behavior of the reference model for that acceleration we can calculate the physical agent that could result in the response Q^D for the reference model. The direct application of this Q^D to the actual system could result in a different response since its physical behavior differs from that of the reference model. Therefore it can be "deformed" into a "*required*" Q^{Req} value that can be directly applied to the actual system. Via substituting the realized response of the actual system $\ddot{q}$ into the reference model the "*realized or recalculated control action*" Q^R can be obtained instead of the "desired one" Q^D [RHS of Fig. 17.1]. Our aim is to find the proper deformation by the application of which Q^R well approaches Q^D, that is *at which the controlled system seems to behave as the reference system*. The proper deformation may be found by the application of an iteration as follows. Consider the iteration $Q^{Req}_{n+1} = G(Q^{Req}_n, Q^R_n, Q^D_{n+1})$ generated by some function in which n is the index of the control cycle. For a slowly varying scenario Q^D can be considered to be constant. In this case the iteration is reduced to $Q^{Req}_{n+1} = G(Q^{Req}_n, Q^R_n, Q^D)$ that has to converge to Q^{Req}_*. One possibility for guaranteeing that is the application of contractive maps in the arrays of real numbers that result in *Cauchy Sequences* that are convergent in complete linear normed spaces. By using the norm-inequality, for a convergent iterative sequence $x_n \to x_*$ it is obtained that

$$\begin{aligned} \|G(x_*, Q^D) - x_*\| &\le \|G(x_*, Q^D) - x_n\| + \|x_n - x_*\| = \\ &= \|G(x_*, Q^D) - G(x_{n-1}, Q^D)\| + \|x_n - x_*\|. \end{aligned} \tag{17.1}$$

It is evident from (17.1) that if G is continuous then the desired fixed point is found by this iteration because in the RHS of (17.1) both terms converge to 0 as $x_n \to x_*$. The next question is giving the necessary or at least a *satisfactory condition of this convergence*. It is also evident that for this purpose the contractivity of $G(\cdot)$, i.e. the property that $\|G(a) - G(b)\| \le K\|a - b\|$ with $0 \le K < 1$, is satisfactory since it leads to a *Cauchy Sequence* ($\|x_{n+L} - x_n\| \to 0 \; \forall L \in \mathbb{N}$):

$$\|x_{n+L} - x_n\| = \|G(x_{n+L-1}) - G(x_{n-1})\| \le \ldots \le K^n \|x_L - x_0\| \to 0 \quad \text{as} \quad n \to \infty. \tag{17.2}$$

For the role of function $G(x, Q^D)$ a novel fixed point transformation was introduced for *Single Input – Single Output (SISO) Systems* (e.g. in [7]) that is rather "robust" as far as the dependence of the resulting function on the behavior of $f(\cdot)$ is concerned (17.3). This robustness can approximately be investigated by the use of an affine approximation of $f(x)$ in the vicinity of x_* and it is the consequence of the strong nonlinear saturation of the sigmoid function $tanh(x)$:

$$G(x,Q^D) := (x+K)\times\left[1+Btanh(A[f(x)-Q^D])\right]-K,$$
$$\text{if}\quad f(x_\star)=x^d\quad\text{then}\quad G(x_*|Q^D)=x_*,\quad G(-K,Q^D)=-K, \tag{17.3}$$
$$G' := \partial G/\partial x,\quad G(x_*,Q^D)' = (x_*+K)ABf'(x_*)+1. \tag{17.4}$$

Eq. (17.3) evidently has a proper ($x_\star$) and a false ($-K$) fixed point, but by properly manipulating the control parameters A, B, and K the condition $|G'(x_\star,Q^D)|<1$ can be guaranteed and the good fixed point can be located within the basin of attraction of the procedure. This means that the iteration can have considerable speed of convergence even nearby $x_\star$, and the strongly saturated *tanh* function can make it robust in its vicinity, that is the properties of $f(x)$ have not too much influence on the behavior of G. (It can be noted that instead of the *tanh* function any sigmoid function with the property of $\sigma(0)=0$, e.g. $\sigma(x) := x/(1+|x|)$, can be similarly applied as well.)

The idea of keeping the iteration convergent by manipulating the width of the basin of attraction of the good fixed point can be developed in the following manner for SISO systems. On the basis of the available rough system model a simple PID controller can be simulated that reveals the order of magnitude of the occurring responses. Parameter K can be selected such that the $x+K$ values are considerable negative numbers. Depending on $sign(f')$ let $B=\pm1$ and let $A>0$ be a small number for which $|\partial G(x,Q^D)/\partial x|\approx 1-\varepsilon_{goal}$ for a small $\varepsilon_{goal}>0$. For Q^D varying in time the following estimation can be done in the vicinity of the fixed point when $|x_n-x_{n-1}|$ is small: $x_{n+1}-x_n = G(x_n,Q^D_n)-G(x_{n-1},Q^D_{n-1})\approx \frac{\partial G(x_{n-1},Q^D_{n-1})}{\partial x}(x_n-x_{n-1})+\frac{\partial G(x_{n-1},Q^D_{n-1})}{\partial Q^D}(Q^D_n-Q^D_{n-1})$. Since from the analytical form of $\sigma(x)$ the term $\frac{\partial G(x_{n-1},Q^D_{n-1})}{\partial Q^D}$ is known, and the past "desired" inputs as well as the arguments of function G are also known, this equation can be used for real-time estimation of $\frac{\partial G(x_{n-1},Q^D_{n-1})}{\partial x}$. The quantity ε_{goal} can be tried to be fixed around -0.5 by tuning parameter A for which various possibilities are available. For the $\sigma(x) := x/(1+|x|)$ choice a "moderate tuning strategy" can be chosen according to (17.5) as

$$d_n := x_n-x_{n-1},\quad d_{n-1} := x_{n-1}-x_{n-2},$$
$$h_n := BA\sigma'\left(A(f_{n-1}-Q^D_{n-1})\right),$$
$$d^d_n := Q^D_n-Q^D_{n-1},$$
$$\varepsilon := \left(d_n+(x_{n-1}+K)h_nd^d_n\right)/d_{n-1}-1, \tag{17.5}$$
$$\text{if}\quad \varepsilon-\varepsilon_{goal}<0\quad\text{then}\quad \dot{A}=\kappa_1\alpha_1\left(\sigma(\varepsilon-\varepsilon_{goal})+\kappa_2 sign(\varepsilon-\varepsilon_{goal})\right),$$
$$\text{if}\quad \varepsilon-\varepsilon_{goal}\geq 0\quad\text{then}\quad \dot{A}=\alpha_1\left(\sigma(\varepsilon-\varepsilon_{goal})+\kappa_2 sgn(\varepsilon-\varepsilon_{goal})\right)$$

with $\kappa_1>1$, $\kappa_2,\alpha_1>0$ parameters ($\sigma'(x)$ refers to the derivative of function $\sigma(x)$), and a more drastic "exponential tuning" according to (17.6)

$$\begin{aligned} &\text{if} \quad \varepsilon - \varepsilon_{goal} < 0 \quad \text{then} \quad \dot{A} = -\kappa_3 \alpha_2 A, \\ &\text{if} \quad \varepsilon - \varepsilon_{goal} \geq 0 \quad \text{then} \quad \dot{A} = \alpha_2 A \end{aligned} \tag{17.6}$$

with $\kappa_3 > 1$, and $\alpha_2 > 0$. In (17.5) κ_1 and in (17.6) κ_3 corresponds to faster decrease than increase in A since it was observed that decreasing A introduces little fluctuations in the consecutive x_n values in a discrete approach, therefore it is more expedient to quickly step over this fluctuating session by fast decrease in A.

It is evident that in the case of a second order physical system that can be controlled by directly setting the second time-derivatives of its generalized coordinate $\ddot{q}$ the measurement of the observed response may have considerable noise. While nearby the fixed point (17.3) is not very sensitive to this noise, the tuning rules (17.5) and (17.6) that contain different past values may evidently be noise-sensitive. In the case of a digital controller this noise can be modeled by adding random disturbance terms to the simulated/observed second time-derivatives. Any linear noise filter can be modeled by an integral or a sum in the discrete approximation as

$$\tilde{f}(t) := \int_0^\infty F(\tau) f(t-\tau) d\tau \quad \text{or} \quad \tilde{f}_k := \sum_{i=0}^{\infty} F_i f_{k-i} \tag{17.7}$$

with some monotone decreasing function $F(\tau)$ or discrete weights F_k that normally converge to zero as $\tau, k \to \infty$. Normally some weighted average can be calculated for a period corresponding to shorter or longer "memory". In the case of a discrete controller the smallest memory is needed for the very simple solution $F_k := \beta^k(1-\beta)$ with $0 < \beta < 1$ that can be calculated by a single buffer P according to the updating rule $P_{n+1} = \beta P_n + f_{k+1}$, $\tilde{f}_{k+1} = (1-\beta)P_{k+1}$. The actual value of β directly influences the "memory" of the system: a larger value corresponds to longer memory than a smaller one. In the sequel simulation results are given for the adaptive control of a pendulum of an indefinite mass center point, by the use of this control and noise filtering technique.

17.2 Simulation Results

The model of the pendulum with a coupled internal degree of freedom not precisely modeled by the controller is given in Fig. 17.2. Equations of motion can be derived from tle Lagrange-Euler equation:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = Q_i,$$

where $L = T - V$ is the Lagrange function and T, V are kinetic and potential energies of the system, respectively. They are given by the formulae:

$$T = \frac{1}{2}\Theta\dot{q}_i^2 + \frac{1}{2}m\left[(L_0+q_2)^2\dot{q}_1^2 + \dot{q}_2^2\right],$$
$$V = \left[C + mg(L_0+q_2)\right]\cos q_1 + \frac{1}{2}kq_2^2,$$

where Θ and C denote the constant rotational inertia and gravitational contribution of the casing of the hidden components, respectively; m denotes the inertia of the mass point that can move by prismatic coordinate q_2 around a fixed distance L_0 measured from the rotary axle with spring stiffness k and viscous friction coefficient μ. g denotes the gravitational acceleration, and q_1 denotes the angle of the rotary axle. The solution of the Lagrange-Euler equation for this system has the following form:

$$\begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} = \begin{bmatrix} \Theta + m(L_0+q_2)^2 & 0 \\ 0 & m \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} 2m(L_0+q_2)\dot{q}_1\dot{q}_2 - \left[C + mg(L_0+q_2)\right]\sin q_1 \\ -m(L_0+q_2)\dot{q}_1^2 + kq_2 + mg\cos q_1 + \mu\dot{q}_2 \end{bmatrix}.$$

The kinematically prescribed desired joint acceleration for trajectory tracking was determined by $\ddot{q}_1^D = \ddot{q}_1^{Nom} + 3\Lambda^2(q_1^{Nom} - q_1) + 3\Lambda(\dot{q}_1^{Nom} - \dot{q}_1) + \Lambda^3\int_0^t [q_1^{Nom}(\xi) - q_1(\xi)]d\xi$.

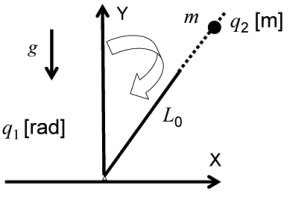


Fig. 17.2 The model of the pendulum; in the considered case $Q_2 = 0$ N, the controller can exert only torque Q_1

In the MRAC framework the "reference model" corresponded to a "nominal model" of parameters $\Theta_m = 50$ kg· m^2, $C_m = 70$ kg·(m/s)^2, $m_m = 20$ kg, $k_m = 100$ N/m, $g_m = 10$ m/s^2, $\mu_m = 0.01$ N·s/m, and $L_{0_m} = 2$ m. The actual parameters of the simulated system were as follows: $\Theta = 30$ kg·m^2, $C = 50$ kg·(m/s)^2, $m = 50$ kg, $k = 1000$ N/m, $g = 9.81$ m/s^2, $\mu = 0.1$ N·s/m, and $L_0 = L_{0_m}$. The noise in the measurement of $\ddot{q}_1$ was generated by even distribution within the interval $[-A_{noise}, A_{noise}]$ with $A_{noise} = 2$ rad/s^2. The nominal trajectory $q_1(t)$ was a third-order periodic spline function. For filtering $\beta = 0.9$ was chosen, and in the adaptive case the stabilization tuning happened according to (17.6) with $\kappa_3 = 2$, $\alpha_2 = 5/s$. The computations were made by the SCILAB v. 5.1.1 – SCICOS v. 4.2 system's ODE solver that automatically selected the integration method depending on the stiffness of the problem. Figure 17.3 well reveals the significance of the adaptive loop when no noise is present. In the "nonadaptive case" the controller cannot be hauled up for implementing the "MRAC" philosophy. However, the adaptive one very well realizes it according to Fig. 17.4.

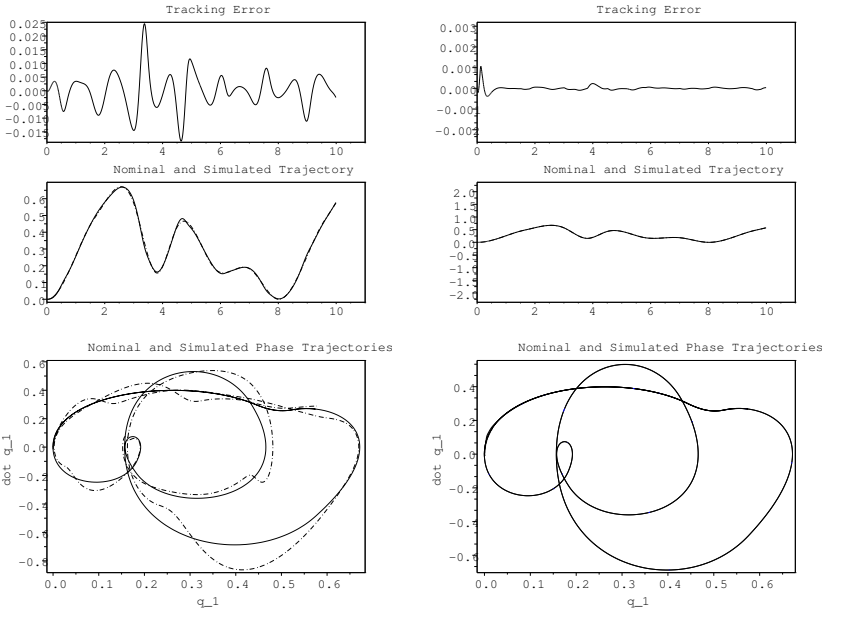


Fig. 17.3 The trajectory ([rad] vs [s]) (first row) and phase trajectory ([rad/s] vs [rad]) (second row) tracking of the nonadaptive (LHS) and adaptive controller with $\beta = 0.9$ (RHS) without noise in the observed $\ddot{q}_1$ signal {nominal: solid, simulated: longdash dot lines}

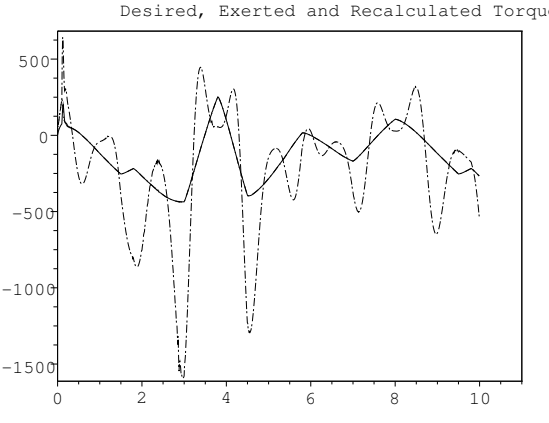


Fig. 17.4 The realization of the MRAC idea in the adaptive case without noise in the observed $\ddot{q}_1$ signal ([rad/s^2] vs time [s]): the desired torque calculated according to the reference model (solid line) and the recalculated torque from the observed response (bigdash longdash line) of the controlled system (almost indistinguishable lines in strict cover), and the torque exerted to the system according to the adaptive deformation (the well separated – longdash dot – line) with $\beta = 0.9$

The torque exerted on the actual system drastically differs from that calculated by the use of the reference model, and the torque recalculated from the reference model with the actual acceleration of the system is very close to that indicated by the reference model. That is the controller generates the illusion for the kinematically designed PID-type trajectory tracking loop that the actual system behaves like the reference model. The smoothness of the phase trajectory indicates that the nominal acceleration is also precisely tracked. To reveal the effect of noise filtering the results obtained for $\beta = 0$ (no filtering) and $\beta = 0.9$ (relatively long memory filter) are described in Fig. 17.5.

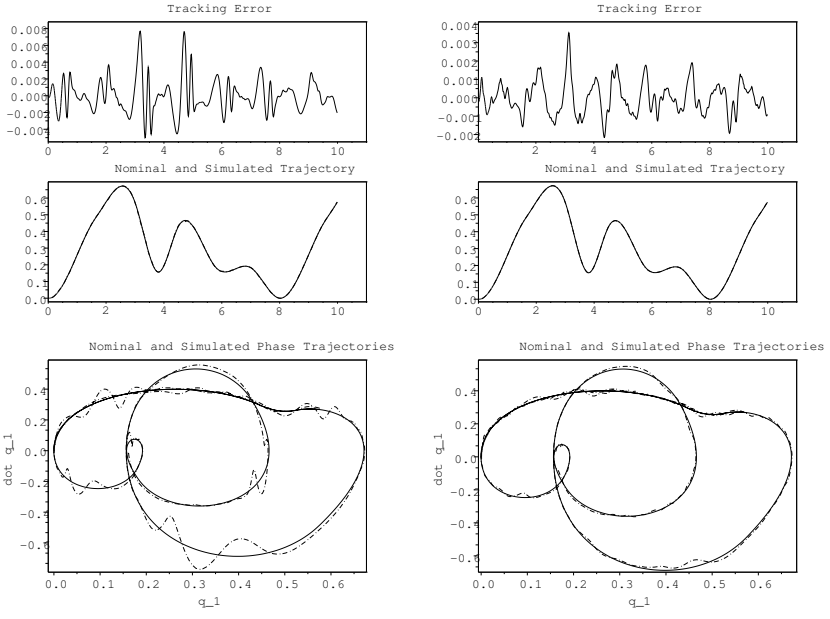


Fig. 17.5 The trajectory ([rad] vs [s]) (first line) and phase trajectory ([rad/s] vs [rad]) (second line) tracking of the adaptive controller for $\beta = 0$ (LHS) and $\beta = 0.9$ (RHS) with the same noise in the observed $\ddot{q}_1$ signal (nominal – solid, simulated – longdash dot lines)

The improvement obtained by the simple filter is evident. More significant improvement can be seen in the diagram of the appropriate torque components (Fig. 17.6). Figure 17.7 also reveals the significant improvement in acceleration tracking.

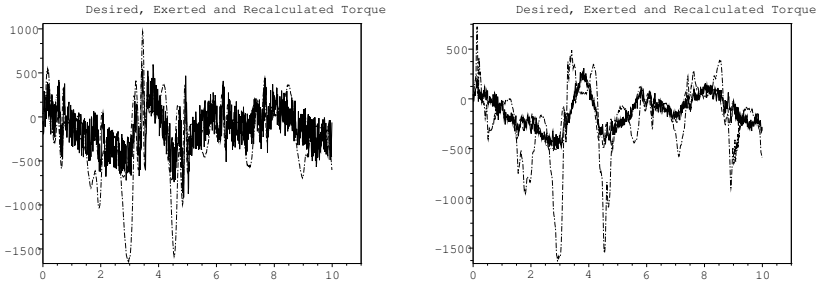


Fig. 17.6 The realization of the MRAC idea in the adaptive case without (LHS) and with noise filtering (RHS): the "desired" (solid line), "exerted" (longdash dot line) and "recalculated" (bigdash longdash line) torque vs time ([N·m] vs [s])

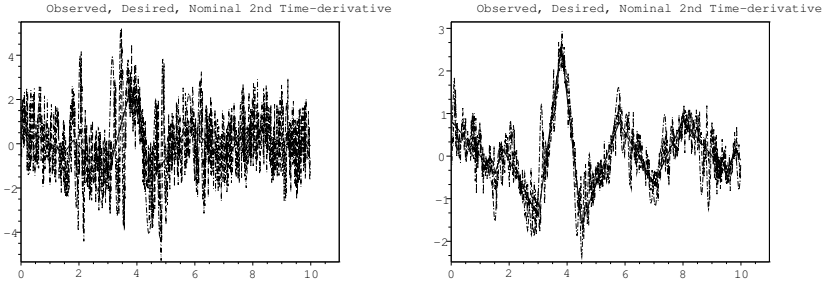


Fig. 17.7 The acceleration tracking in the adaptive case without (LHS) and with noise filtering (RHS) ([rad/s^2] vs [s]); nominal value (solid line), desired value i.e. the nominal one corrected by the PID-type trajectory tracking (longdash dot line), observed noisy quantity (bigdash longdash line)

17.3 Conclusions

In this paper the effects of a very simple noise filtering technique were investigated via simulation in a novel version of "Model Reference Adaptive Controllers", the design of which is based on the use of the simple "Robust Fixed Point Transformations" instead of Lyapunov's second or "direct" method. In comparison with the Lyapunov function based techniques that normally guarantee global stability at the cost of complicated considerations the novel one suffers from the deficiency that it can guarantee only a local, bounded region of convergence. This deficiency was relaxed by a suggested adaptive tuning that introduced some noise sensitivity into the controller. It was found that the simple filtering recommended considerably reduced the sensitivity of the novel method proposed and substantiated the expectations that it could be applied in practice.

Acknowledgements. The authors gratefully acknowledge the support by the *National Office for Research and Technology (NKTH)* using the resources of the *Research and Technology Innovation Fund* within the project *OTKA: No. CNK-78168*.

References

1. Nguyen, C.C., Antrazi, S.S., Zhou, Z.-L., Campbell Jr., C.E.: Adaptive control of a Stewart platform–based manipulator. Journal of Robotic Systems 10(5), 657–687 (1993)
2. Somló, J., Lantos, B., Cát, P.T.: Advanced robot control. Akadémiai Kiadó (2002)
3. Hosseini–Suny, K., Momeni, H., Janabi–Sharifi, F.: Model Reference Adaptive Control Design for a Teleoperation System with Output Prediction. J. Intell. Robot Syst., 1–21 (2010), doi:10.1007/s10846-010-9400-4
4. Khoh, C.J., Tan, K.K.: Adaptive robust control for servo manipulators. Neural Comput. & Applic. 12, 178–184 (2005)
5. Slotine, J.-J.E., Li, W.: Applied Nonlinear Control. Prentice Hall International, Inc., Englewood Cliffs (1991)
6. Tar, J.K., Bitó, J.F., Nádai, L., Tenreiro Machado, J.A.: Robust Fixed Point Transformations in Adaptive Control Using Local Basin of Attraction. Acta Polytechnica Hungarica 6(1), 21–37 (2009)
7. Tar, J.K., Bitó, J.F., Rudas, I.J.: Replacement of Lyapunov's Direct Method in Model Reference Adaptive Control with Robust Fixed Point Transformations. In: Proc. of the 14th IEEE International Conference on Intelligent Engineering Systems, Las Palmas of Gran Canaria, Spain, May 5-7, pp. 231–235 (2010)
8. Tar, J.K., Rudas, I.J., Bitó, J.F., Kozłowski, K.R.: A Novel Approach to the Model Reference Adaptive Control of MIMO Systems. In: Proc. of the 19th International Workshop on Robotics in Alpe–Adria–Danube Region (RAAD 2010), Budapest, Hungary, June 23-25, pp. 31–36 (2010) (CD issue, file: 4_raad2010.pdf)
9. Andoga, R., Főző, L., Madarász, L.: Digital Electronic Control of a Small Turbojet Engine MPM 20. Acta Polytechnica Hungarica 4(4), 83–95 (2007)

Chapter 18
Real-Time Estimation and Adaptive Control of Flexible Joint Space Manipulators

Steve Ulrich and Jurek Z. Sasiadek

Abstract. Most advanced trajectory tracking control laws for robot manipulators require a knowledge of all state variables. For lightweight flexible-joint space manipulators this objective is difficult to achieve since link positions are typically not measured. In this paper, an extended Kalman filter (EKF) observer to estimate all state variables of a manipulator system modeled with a nonlinear stiffness dynamics model is presented. In addition, it is shown that the observer can be modified in order to calibrate in real-time the sensor biases. The state variable estimates are coupled to a flexible joint adaptive controller to provide a complete closed-loop trajectory tracking solution. Simulation results show that the proposed solution provides satisfying tracking performance in a 12.6×12.6 m trajectory tracking scenario.

18.1 Introduction

The benefits of lightweight space robots are obtained at the price of higher elasticities in the joints leading to a more complex dynamic behaviour, which requires advanced control techniques in order to obtain accurate trajectory control. For such flexible manipulator systems, several controllers have been proposed in the literature. However, the vast majority of flexible joint control algorithms are classified as full-state feedback. Full state feedback control requires the knowledge of four state variables: link and motor angular positions, $\mathbf{q}$ and $\mathbf{q}_m$, and link and motor angular velocities, $\dot{\mathbf{q}}$ and $\dot{\mathbf{q}}_m$. Although some advanced space robot systems have access to measurements providing a knowledge of joint elasticity effects, typical robot manipulators are instrumented to measure only motor positions and velocities with an encoder and a tachometer on each motor axis of the manipulator.

Steve Ulrich · Jurek Z. Sasiadek
Department of Mechanical and Aerospace Engineering, Carleton University,
1125 Colonel By Drive, Ottawa, ON, K1S 5B6 Canada
e-mail: {sulrich,jsas}@connect.carleton.ca

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 215–224.
springerlink.com

Therefore, a class of solutions to estimate in real-time state variables not available through measurements consists in applying the Kalman filter theory. Timcenko and Kircanski developed a linear Kalman filter to estimate the control torque in a flexible joint robot and used it in a feedforward/feedback controller scheme [10]. In 2010, Lightcap and Banks presented an Extended Kalman Filter (EKF) to estimate link and motor positions/velocities based on motor measurements [6]. Although their approach does not use directly link position measurements, a real-time knowledge of link positions is provided by sets of retro-reflective markers positioned on the links, which represents an uncommon sensor for robotic manipulators, especially for those operating in space.

One major problem with these Kalman filter-based methodologies for flexible joint robots is that despite the fact that experimental studies have shown that flexible gears are much more complex than that of a linear spring, their design and simulation validation are based on the classic dynamic representation proposed by Spong [8] which models each joint as a linear torsional spring of constant stiffness. Another deficiency of the aforementioned approaches is that the measurements are assumed to be zero-mean Gaussian, whereas practically, the measurement accuracy obtained using encoders and tachometers is influenced by number of factors, including biases. In this paper, the nonlinear estimation and adaptive trajectory tracking control problem associated with flexible joint space manipulator equipped only with an encoder and a tachometer on each motor axis addressed. Unlike previous EKF-based approaches, a nonlinear joint stiffness model is considered and measurement biases are taken into account.

18.2 Flexible Joint Dynamics

Since the development of the linear stiffness model by Spong [8] several researchers have conducted experiments in order to derive a nonlinear, detailed dynamics model of flexible effects in the joints of robotic manipulators. The most relevant nonlinear effects are related to nonlinear stiffness. According to the literature, to replicate experimental joint stiffness curves, several studies recommend approximating the stiffness torque by a nonlinear cubic function. Another important characteristic related to nonlinear stiffness is the *soft-windup* effect which deforms the torque-torsion toward the torque axis in the region from 0 to 1 N·m [4]. Besides nonlinear joint stiffness effects, friction torques have an important impact on the behavior of robot manipulator systems. Therefore, neglecting gravitational forces for space operations and combining the cubic nonlinear stiffness torque term, soft-windup effect, and frictional torques, the following nonlinear joint dynamics representation was obtained by Ulrich and Sasiadek [13]

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{S}\ddot{\mathbf{q}}_m + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{f}(\dot{\mathbf{q}}) - \mathbf{k}(\mathbf{q},\mathbf{q}_m)(\mathbf{q}_m - \mathbf{q}) = \mathbf{0}, \tag{18.1}$$

$$\mathbf{S}^T\ddot{\mathbf{q}} + \mathbf{J}_m\ddot{\mathbf{q}}_m + \mathbf{k}(\mathbf{q},\mathbf{q}_m)(\mathbf{q}_m - \mathbf{q}) = \tau \tag{18.2}$$

where $\mathbf{q}$ denotes the link angle vector, $\mathbf{q}_m$ represents the motor angle vector, $\mathbf{M}(\mathbf{q})$ denotes the symmetric positive-definite link inertia matrix, $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$ represents the centrifugal-Coriolis matrix, $\mathbf{J}_m$ denotes the positive-definite motor inertia matrix, and τ represents the control torque vector. The inertial coupling matrix is given by

$$\mathbf{S} = \begin{bmatrix} 0 & J_{m2} \\ 0 & 0 \end{bmatrix}. \tag{18.3}$$

For the two-link manipulator shown in Fig. 18.1, the definition of $\mathbf{M}(\mathbf{q})$ and $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$ can be found in [9]. In Eqs. (18.1) and (18.2), the nonlinear stiffness torque term is given by [4]

$$\mathbf{k}(\mathbf{q},\mathbf{q}_m)(\mathbf{q}_m - \mathbf{q}) = \mathbf{a}_1 \begin{bmatrix} (q_{m1} - q_1)^3 \\ (q_{m2} - q_2)^3 \end{bmatrix} + \mathbf{a}_2(\mathbf{q}_m - \mathbf{q}) + \mathbf{K}_{sw}(\mathbf{q},\mathbf{q}_m)(\mathbf{q}_m - \mathbf{q}) \tag{18.4}$$

where $\mathbf{a}_1$ and $\mathbf{a}_2$ are positive definite diagonal matrices of stiffness coefficients and $\mathbf{K}_{sw}(\mathbf{q},\mathbf{q}_m)$ is the soft-windup correction factor that is modeled as a saddle-shaped function

$$\mathbf{K}_{sw}(\mathbf{q},\mathbf{q}_m) = -\mathbf{k}_{sw} \begin{bmatrix} e^{-a_{sw}(q_{m1}-q_1)^2} & 0 \\ 0 & e^{-a_{sw}(q_{m2}-q_2)^2} \end{bmatrix}. \tag{18.5}$$

In Eq. (18.1), $\mathbf{f}(\dot{\mathbf{q}})$ denotes the friction which is modeled as follows [7]

$$\mathbf{f}(\dot{\mathbf{q}}) = \gamma_1[\tanh(\gamma_2\dot{\mathbf{q}}) - \tanh(\gamma_3\dot{\mathbf{q}})] + \gamma_4\tanh(\gamma_5\dot{\mathbf{q}}) + \gamma_6\dot{\mathbf{q}} \tag{18.6}$$

where, for $i = 1,\ldots,6$, γ_i denotes positive parameters defining the different friction components.

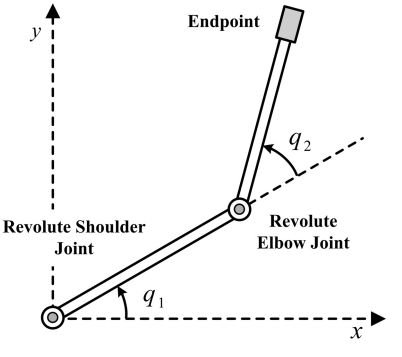


Fig. 18.1 Two-link flexible joint manipulator

18.3 Modified Simple Adaptive Control

In this section, the adaptive controller developed by Ulrich and Sasiadek for flexible joint manipulators [12] is presented. This adaptive control law uses the singular perturbation-based (SPB) methodology in which a fast control term that damps the elastic vibrations at the joints is added to a slow control term which controls the rigid dynamics. The fast control term is chosen as a linear correction of the form $\mathbf{K}_v(\dot{\mathbf{q}} - \dot{\mathbf{q}}_m)$. The rigid control term consists in an intuitive transpose Jacobian (TJ) Cartesian control law in which the control gains are adapted in real-time with the modified simple adaptive control (MSAC) methodology [11]. With this technique, a reference model serves as the basis to generate the commands for the unknown system to be controlled. The resulting adaptive composite controller is given by [12]

$$\tau = \mathbf{J}(\mathbf{q})^T \left[\mathbf{K}_p(t) \begin{pmatrix} e_x \\ e_y \end{pmatrix} + \mathbf{K}_d(t) \begin{pmatrix} \dot{e}_x \\ \dot{e}_y \end{pmatrix} \right] + \mathbf{K}_v(\hat{\dot{\mathbf{q}}} - \hat{\dot{\mathbf{q}}}_m) \tag{18.7}$$

where $\mathbf{J}(\mathbf{q})$ is the robot Jacobian matrix and $\mathbf{K}_p(t)$ and $\mathbf{K}_d(t)$ are the proportional and derivative adaptive control gain matrix, respectively. In Eq. (18.7), e_x, e_y, $\dot{e}_x$, and $\dot{e}_y$ are the errors between the reference model outputs and the actual Cartesian output estimates. For simplicity, the intuitive reference model is selected as a set of second order uncoupled linear equations.

The total control torque actuating each joint feeds into the inverse flexible joint dynamics equations resulting in a link and motor angular acceleration vectors, $\ddot{\mathbf{q}}$ and $\ddot{\mathbf{q}}_m$, which are double integrated to obtain link and motor rates, $\dot{\mathbf{q}}$ and $\dot{\mathbf{q}}_m$, and link and motor angular positions, $\mathbf{q}$ and $\mathbf{q}_m$. Then, the motor positions and velocities are measured by the sensors and used as inputs of the EKF in combination with the control torque in order to provide the best estimate of the state variables, $\hat{\mathbf{q}}$, $\hat{\dot{\mathbf{q}}}$, $\hat{\mathbf{q}}_m$, $\hat{\dot{\mathbf{q}}}_m$. The link position/velocity and motor velocity estimates are then used in the adaptive control law given by Eq. (18.7) and in the standard kinematics equations in order to calculate the end-effector position and velocity estimates, $\hat{x}$, $\hat{y}$, $\hat{\dot{x}}$, and $\hat{\dot{y}}$.

With the MSAC law, the adaptation mechanism of the controller gains is given as follows

$$\mathbf{K}_p(t) = \mathbf{K}_{pp}(t) + \int \dot{\mathbf{K}}_{pi}(t)dt, \tag{18.8}$$

$$\mathbf{K}_d(t) = \mathbf{K}_{dp}(t) + \int \dot{\mathbf{K}}_{di}(t)dt \tag{18.9}$$

where

$$\mathbf{K}_{pp}(t) = \begin{bmatrix} e_x^2\Gamma_{pp} & 0 \\ 0 & e_y^2\Gamma_{pp} \end{bmatrix}, \tag{18.10}$$

$$\dot{\mathbf{K}}_{pi}(t) = \begin{bmatrix} e_x^2\Gamma_{pi} - \sigma_p K_{pi11}(t) & 0 \\ 0 & e_y^2\Gamma_{pi} - \sigma_p K_{pi22}(t) \end{bmatrix}, \tag{18.11}$$

$$\mathbf{K}_{dp}(t) = \begin{bmatrix} \dot{e}_x^2\Gamma_{dp} & 0 \\ 0 & \dot{e}_y^2\Gamma_{dp} \end{bmatrix}, \tag{18.12}$$

$$\dot{\mathbf{K}}_{di}(t) = \begin{bmatrix} \dot{e}_x^2 \Gamma_{di} - \sigma_d K_{di11}(t) & 0 \\ 0 & \dot{e}_y^2 \Gamma_{di} - \sigma_d K_{di22}(t) \end{bmatrix}. \tag{18.13}$$

In Eqs. (18.10) to (18.13), Γ_{pp}, Γ_{pi}, Γ_{dp}, and Γ_{di} are control parameters to be adjusted and σ_p and σ_d are small positive control coefficients necessary to avoid divergence of the integral control gains. The derivation of the theoretical proof of stability of the modified simple adaptive control system presented in this section when applied to a flexible joint robot can be found in [13].

18.4 Nonlinear Estimation

In this section, the bias-free discrete-time EKF estimator developed in [14] for the nonlinear joint model are first presented. Second, the estimator is modified to estimate and compensate for the measurement biases in real-time.

Let the flexible joint robot system presented be described by the general nonlinear model

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, \mathbf{w}), \tag{18.14}$$

$$\mathbf{y} = h(\mathbf{x}, \mathbf{v}) \tag{18.15}$$

where $\mathbf{w}$ and $\mathbf{v}$ respectively denote the process and measurement noise, $\mathbf{u}$ represents the control input vector and $\mathbf{x}$ is the state vector given by

$$\mathbf{x} = \begin{bmatrix} \mathbf{q} & \dot{\mathbf{q}} & \mathbf{q}_m & \dot{\mathbf{q}}_m \end{bmatrix}^T. \tag{18.16}$$

The EKF can be thought as a predictor-corrector. In the prediction, or propagation, phase, $\hat{\mathbf{x}}_{k+1}^-$ is obtained by integrating the noise-free state equation $\dot{\hat{\mathbf{x}}}_{k+1} = f(\hat{\mathbf{x}}_k, \mathbf{u}_k)$ between t_k and t_{k+1}, using $\hat{\mathbf{x}}_0$ as an initial condition. Although the propagation of the state vector is performed with the nonlinear model of the dynamics, the propagation of state error covariance matrix $\mathbf{P}_k$ is done with the linearized versions of Eqs. (18.14) and (18.15), i.e. by taking the partial derivative of the robot dynamics with respect to the states, as follows

$$\partial f / \partial \mathbf{x} = \mathbf{F} = \begin{bmatrix} \mathbf{0} & \mathbf{I}_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{F}_{21} & \mathbf{F}_{22} & \mathbf{F}_{23} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_2 \\ \mathbf{F}_{41} & \mathbf{0} & \mathbf{F}_{43} & \mathbf{0} \end{bmatrix} \tag{18.17}$$

where

$$\mathbf{F}_{21} = -\mathbf{M}^{-1}(\mathbf{q}) \left[\frac{\partial \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}} \ddot{\mathbf{q}} + \frac{\partial \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \mathbf{q}} \dot{\mathbf{q}} - \frac{\partial \mathbf{k}(\mathbf{q}, \mathbf{q}_m)}{\partial \mathbf{q}} (\mathbf{q}_m - \mathbf{q}) + \mathbf{k}(\mathbf{q}, \mathbf{q}_m) \right], \tag{18.18}$$

$$\mathbf{F}_{22} = -\mathbf{M}^{-1}(\mathbf{q})\left[\frac{\partial \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}}\dot{\mathbf{q}} + \frac{\partial \mathbf{f}(\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\right], \tag{18.19}$$

$$\mathbf{F}_{23} = \mathbf{M}^{-1}(\mathbf{q})\left[\frac{\partial \mathbf{k}(\mathbf{q},\mathbf{q}_m)}{\partial \mathbf{q}_m}(\mathbf{q}_m - \mathbf{q}) + \mathbf{k}(\mathbf{q},\mathbf{q}_m)\right], \tag{18.20}$$

$$\mathbf{F}_{41} = -\mathbf{J}_m^{-1}\left[\frac{\partial \mathbf{k}(\mathbf{q},\mathbf{q}_m)}{\partial \mathbf{q}}(\mathbf{q}_m - \mathbf{q}) - \mathbf{k}(\mathbf{q},\mathbf{q}_m)\right], \tag{18.21}$$

$$\mathbf{F}_{43} = -\mathbf{J}_m^{-1}\left[\frac{\partial \mathbf{k}(\mathbf{q},\mathbf{q}_m)}{\partial \mathbf{q}_m}(\mathbf{q}_m - \mathbf{q}) + \mathbf{k}(\mathbf{q},\mathbf{q}_m)\right]. \tag{18.22}$$

The partial derivatives are given by

$$\frac{\partial \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}}\ddot{\mathbf{q}} = -m_2 l_1 l_{c2} \sin q_2 \begin{bmatrix} 0 & 2\ddot{q}_1 + \ddot{q}_2 \\ 0 & \ddot{q}_1 \end{bmatrix}, \tag{18.23}$$

$$\frac{\partial \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}}\dot{\mathbf{q}} = -m_2 l_1 l_{c2} \cos q_2 \begin{bmatrix} 0 & \dot{q}_1\dot{q}_2 + \dot{q}_2(\dot{q}_1 + \dot{q}_2) \\ 0 & -\dot{q}_1^2 \end{bmatrix}, \tag{18.24}$$

$$\frac{\partial \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}}\dot{\mathbf{q}} = -m_2 l_1 l_{c2} \sin q_2 \begin{bmatrix} \dot{q}_2 & (\dot{q}_1 + \dot{q}_2) \\ -\dot{q}_1 & 0 \end{bmatrix}, \tag{18.25}$$

$$\frac{\partial \mathbf{k}(\mathbf{q},\mathbf{q}_m)}{\partial \mathbf{q}}(\mathbf{q}_m - \mathbf{q}) = -\frac{\partial \mathbf{k}(\mathbf{q},\mathbf{q}_m)}{\partial \mathbf{q}_m}(\mathbf{q}_m - \mathbf{q}) = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}, \tag{18.26}$$

$$\frac{\partial \mathbf{f}(\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} = \begin{bmatrix} c & 0 \\ 0 & d \end{bmatrix} \tag{18.27}$$

with

$$a = -2(q_{m1} - q_1)^2\left[a1_{11} + k_{sw11}a_{sw}e^{-a_{sw}(q_{m1}-q_1)^2}\right], \tag{18.28}$$

$$b = -2(q_{m2} - q_2)^2\left[a1_{22} + k_{sw22}a_{sw}e^{-a_{sw}(q_{m2}-q_2)^2}\right], \tag{18.29}$$

$$c = \gamma_1\left[\gamma_2 \operatorname{sech}^2(\gamma_2\dot{q}_1) - \gamma_3 \operatorname{sech}^2(\gamma_3\dot{q}_1)\right] + \gamma_4\gamma_5 \operatorname{sech}^2(\gamma_5\dot{q}_1) + \gamma_6, \tag{18.30}$$

$$d = \gamma_1\left[\gamma_2 \operatorname{sech}^2(\gamma_2\dot{q}_2) - \gamma_3 \operatorname{sech}^2(\gamma_3\dot{q}_2)\right] + \gamma_4\gamma_5 \operatorname{sech}^2(\gamma_5\dot{q}_2) + \gamma_6 \tag{18.31}$$

where expressions for link accelerations in Eqs. (18.23) to (18.25) can be obtained by inverting the link dynamics equation (18.1).

Once the covariance matrix has been propagated using the linearized model $\mathbf{F}$, the corrector obtains $\hat{\mathbf{x}}_{k+1}^+$ by using the usual EKF correction equations given by

$$\hat{\mathbf{x}}_{k+1}^+ = \hat{\mathbf{x}}_{k+1}^- + \mathbf{K}_{k+1}[\mathbf{z}_k - h(\hat{\mathbf{x}}_{k+1}^-)] \tag{18.32}$$

where $\mathbf{K}_{k+1}$ is the Kalman gain computed using the propagated covariance matrix $\mathbf{P}_{k+1}^-$ and $\mathbf{H}$, the linearized measurement model. Considering that the only measurements are provided by an encoder and tachometer on each motor axis, let define the measurement model as

$$h(\mathbf{x}) = \begin{bmatrix}\mathbf{q}_m & \dot{\mathbf{q}}_m\end{bmatrix}^T. \tag{18.33}$$

Thus, the linearization of this measurement model for the robot dynamics is as follows

$$\frac{\partial h(\mathbf{x})}{\partial \mathbf{x}} = \mathbf{H} = \begin{bmatrix} \mathbf{0}\ \mathbf{0}\ \mathbf{I}_2\ \mathbf{0} \\ \mathbf{0}\ \mathbf{0}\ \mathbf{0}\ \mathbf{I}_2 \end{bmatrix}^T. \tag{18.34}$$

One of the underlying assumptions of the EKF formulation is that the measurement noise is zero-mean Gaussian with a standard deviation σ. However, measurements are generally corrupted by biases, hence the need for on-board calibration. In this subsection, the bias-free extended Kalman filter (EKF) derivation is modified to determine and compensate for the biases in real-time.

Let the augmented state vector be given by

$$\mathbf{x}_{aug} = \begin{bmatrix} \mathbf{x}\ \mathbf{b} \end{bmatrix}^T \tag{18.35}$$

where $\mathbf{b}$ is a 4×1 vector that includes the measurement bias for $\mathbf{q}_m$ and $\dot{\mathbf{q}}_m$. Because there is no a priori information about $\mathbf{b}$, it is assumed that $\dot{\mathbf{b}} = \mathbf{0}$ which implies that the biases vary slowly relative to the system dynamics. Because of this model, the propagation of $\mathbf{b}$ is

$$\hat{\mathbf{b}}_{k+1}^{-} = \hat{\mathbf{b}}_k. \tag{18.36}$$

Because the additional state $\mathbf{b}$ only affects the measurements without impacting directly the original state vector $\mathbf{x}$, the dimensions of the $\mathbf{F}$ matrix are simply adjusted with additional zeros in order to accomodate the dimensions of the augmented state vector $\mathbf{x}_{aug}$ and the linear measurement model becomes

$$\frac{\partial h(\mathbf{x}_{aug})}{\partial \mathbf{x}_{aug}} = \mathbf{H}_{aug} = \begin{bmatrix} \mathbf{0}\ \mathbf{0}\ \mathbf{I}_2\ \mathbf{0}\ \mathbf{I}_2\ \mathbf{0} \\ \mathbf{0}\ \mathbf{0}\ \mathbf{0}\ \mathbf{I}_2\ \mathbf{0}\ \mathbf{I}_2 \end{bmatrix}^T. \tag{18.37}$$

18.5 Simulation Results

To assess the performance of the EKF estimator-adaptive controller, a 12.6×12.6 m square trajectory was required to be tracked in 60 sec. with constant velocity by the end-effector of a two-link flexible joint space robot for which the shoulder joint coincides with the fixed base of the robot located at the center of the square trajectory.

The robot parameters are given by $l_1 = l_2 = 4.5$ m, $m_1 = m_2 = 1.5075$ kg, $\mathbf{J}_m = \text{diag}[1]$ kg·m^2, $\mathbf{a}_1 = \mathbf{a}_2 = \text{diag}[500]$ N·m/rad, $\mathbf{k}_{sw} = \text{diag}[10]$, $a_{sw} = 3000$, $\gamma_1 = 0.5$, $\gamma_2 = 150$, $\gamma_3 = 50$, $\gamma_4 = 2$, $\gamma_5 = 100$, and $\gamma_6 = 0.5$. The adaptive controller gains and parameters are defined in Ulrich and Sasiadek [12]. Random zero-mean Gaussian noise with standard deviation of 1 deg and 1 deg/s were added to the measurement of each motor angular position and velocity, respectively. Since it is assumed that no apriori information is available on the measurement biases, the bias estimator has been initialized with $\hat{\mathbf{b}}_0 = \mathbf{0}$.

Figures 18.2 to 18.4 show the results of tracking end-effector trajectories with the EKF-adaptive controller combination in a counter-clockwise direction starting at the lower-right-hand corner. As shown in Fig. 18.2, when the sensors are assumed to provide zero-mean measurements, the closed-loop bias-free estimator-adaptive

control strategy provides rapid settling to a steady-state, such that tracking is close to a straight line along each side of the square trajectory. However, when biases of 2 deg and 2 deg/s are respectively added to the motor encoder and tachometer measurements, the tracking results obtained with the same bias-free estimator and adaptive control scheme are highly aggravated. This is shown in Fig. 18.3 where the robot end-point trajectory fails to follow adequately the commanded square trajectory. Finally, Fig. 18.4 shows that by calibrating the sensors in real-time, i.e. by

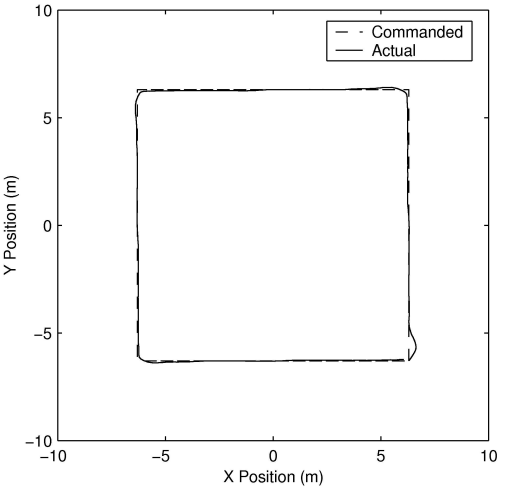


Fig. 18.2 Adaptive trajectory tracking with no measurement biases

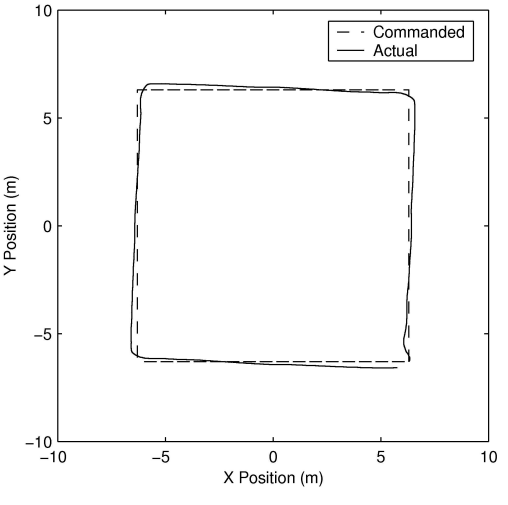


Fig. 18.3 Adaptive trajectory tracking with measurement biases but without real-time calibration

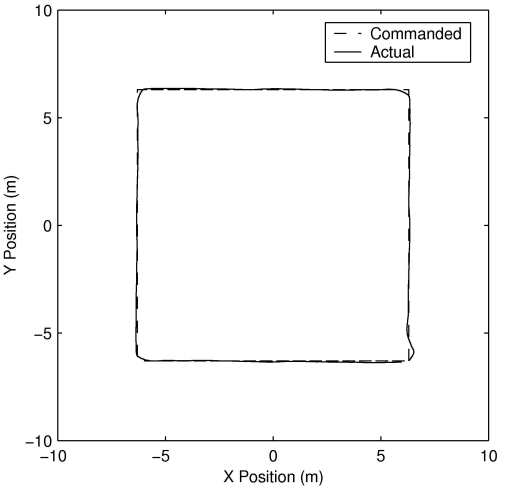


Fig. 18.4 Adaptive trajectory tracking with measurement biases and real-time calibration

using the bias estimator, the obtained results are similar to those obtained with bias-free measurements. Hence, this result gives an indication of the increased robustness provided by the bias estimator-adaptive control strategy to measurement biases.

18.6 Conclusion

In this paper, the problem associated with the estimation and control a flexible joint space manipulator has been addressed. The original contribution of this paper relies in the design of an EKF estimator that is applicable to space robot manipulators equipped with only motor encoders and tachometers that are providing noisy and biased measurements. Also, the proposed nonlinear estimator takes into account practical considerations such as nonlinear joint stiffness, soft-windup and frictional effects. This novel EKF estimator was combined to a composite adaptive controller for which the controller gains are adapted in real-time using the modified simple adaptive control law. The resulting estimation and control system was evaluated in closed-loop numerical simulations which demonstrated that satisfying tracking performance can be achieved dispite nonlinear effects included in the dynamics formulation and measurement biases.

References

1. Green, A., Sasiadek, J.Z.: Robot manipulator control for rigid and assumed mode flexible dynamics models. In: Proc. AIAA Guidance, Navigation and Control Conf. AIAA, Reston (2003)

2. Green, A., Sasiadek, J.Z.: Flexible robot trajectory tracking control. In: Kozlowski, K. (ed.) Robot Motion And Control. Springer, London (2006)
3. Hollars, M.G., Cannon Jr., R.H.: Experimental implementation of a nonlinear estimator in the control of flexible joint manipulators. Automatic Control in Aerospace, 133–140 (1990)
4. Kircanski, N.M., Goldenberg, A.A.: An experimental study of nonlinear stiffness, hysteresis, and friction effects in robot joints with harmonic drives and torque sensors. The Int. J. of Robot. Res. 16, 214–239 (1997)
5. Lewis, F.L.: Optimal estimation with an introduction to stochastic control theory. Wiley, New York (1986)
6. Lightcap, C.A., Banks, S.A.: An extended Kalman filter for real-time estimation and control of a rigid-link flexible-joint manipulator. IEEE Trans. Control Syst. Technol. 18, 91–103 (2010)
7. Makkar, C., Dixon, W.E., Sawyer, W.G., et al.: A new continuously differentiable friction model for control systems design. In: IEEE/ASME Int. Conf. on Adv. Intel. Mecha., pp. 600–605. IEEE Press, Piscataway (2005)
8. Spong, M.W.: Modeling and Control of Elastic Joint Robots. J. Dyn. Syst., Meas., Contr. 109, 310–319 (1987)
9. Spong, M.W., Hutchinson, S., Vidyasagar, M.: Robot modeling and control. Wiley, New York (2006)
10. Timcenko, A., Kircanski, N.: Control of robots with elastic joints: deterministic observer and Kalman filter approach. In: Proc. IEEE Conf. Robot. Autom., pp. 722–727. IEEE Press, Piscataway (1992)
11. Ulrich, S., de Lafontaine, J.: Autonomous atmospheric entry on Mars: performance improvement using a novel adaptive control algorithm. J. Astronaut. Sci. 55, 431–449 (2007)
12. Ulrich, S., Sasiadek, J.Z.: Direct model reference adaptive control of a flexible joint robot. In: Proc. AIAA Guidance, Navigation and Control Conf. AIAA, Reston (2010)
13. Ulrich, S., Sasiadek, J.Z.: Modeling and direct adaptive control of a flexible joint space manipulator. J. Guid., Contr., Dynam. (2011) (accepted for publication)
14. Ulrich, S., Sasiadek, J.Z.: Extended kalman filtering for flexible joint space robot control. In: Proc. American Control Conf. IEEE Press, Piscataway (2011)

Chapter 19
Visual Servo Control Admitting Joint Range of Motion Maximally

Masahide Ito and Masaaki Shibata

19.1 Introduction

For robotic systems, a camera is a very useful sensory device to understand their workspace without contact. Introducing visual information extracted from the image into a control loop has potential to increase the flexibility and accuracy of a given task. In particular, feedback control with visual information, so-called *visual servoing* or *visual feedback control*, is an important technique for robotic systems [1].

Visual servoing is classified roughly into position-based visual servoing (PBVS) and image-based visual servoing (IBVS) approaches. Their main difference depends on how to use the visual features of the target object which are extracted from the image. The PBVS controller is designed using the relative three-dimensional (3D) pose between the camera and the target object, which is estimated from the visual features. On the other hand, the IBVS controller is designed directly using the visual features. One advantage of the IBVS approach over the PBVS approach is robustness against calibration errors. Recently, hybrid visual servoing that incorporates the advantage of both PBVS and IBVS approaches is proposed, e.g., 2-1/2-D visual servoing [10], partitioned visual servoing [4], and Deguchi's method [5].

Meanwhile, robotic systems are practically subject to some physical constraints such as actuator saturation, joint limits, and task feasible region. If such constraints are violated in the robot's run, this can deteriorate control performance drastically and, at worst, lead to instability and a breakdown of the system. Accordingly, it is inherently important to consider these constraints of practical robotic systems.

A controlled object for which it is relatively easy to consider the constraints in its control is a kinematically redundant manipulator. The kinematically redundant manipulator has more degrees-of-freedom (dof) than are required to perform a given

Masahide Ito · Masaaki Shibata
Seikei University, 3-3-1 Kichijoji-kitamachi, Musashino-shi, Tokyo 180-8633, Japan
e-mail: {masahide_i,shibam}@st.seikei.ac.jp

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 225–235.
springerlink.com

task (e.g., at the end-effector); i.e., there exists redundancy in the solution of the inverse differential kinematics problem [3] that obtains a joint motion from a task motion. We usually adopt the *Gradient Projection Method (GPM)* [9] and introduce a performance criterion function (PCF) to resolve the redundancy. Consequently, we obtain the solution so as to maximize the PCF. The secondary task executed according to the PCF has no effect on the primary task, because the secondary task is executed in the null space of the primary task. We can build a desired secondary task which satisfies a constraint condition into the PCF.

Similarly to kinematically redundant manipulators, redundancy appears in IBVS problems. Avoiding joint limits in the secondary task of IBVS is discussed in some literature [14, 12, 2, 11, 13]. The authors have also proposed an original PCF for avoiding joint limits of kinematically redundant manipulators [6]. This function is based on admitting joint range of motion maximally; that concept is different from any conventional functions [9, 16, 18, 12].

In this paper, we apply the maximal admission method of joint range of motion using redundancy to IBVS in an eye-in-hand system. The controlled object is a hand-eye robot as depicted in 19.1. The performance of the application is evaluated experimentally in comparison with the conventional ones.

19.2 Image-Based Visual Servoing

In this section, we recall generic IBVS in eye-in-hand. Note that we assume the target object to be static in visual servoing.

Suppose that a target object is equipped with k markers. We refer to the markers as feature points on the image plane. The geometric relation between the camera and the i-th marker is depicted in 19.2, where the coordinate frames Σ_w, Σ_c, Σ_s, and Σ_f represent the world frame, the camera frame, the standard camera frame, and the image plane frame, respectively. The frame Σ_w is located at the base of the robot. The frame Σ_s is static on Σ_w at a given time. Let ${}^s\mathbf{p}_{o_i} := [{}^sx_{o_i}, {}^sy_{o_i}, {}^sz_{o_i}]^\top$, ${}^s\mathbf{p}_c := [{}^sx_c, {}^sy_c, {}^sz_c]^\top$, and ${}^c\mathbf{p}_{o_i} := [{}^cx_{o_i}, {}^cy_{o_i}, {}^cz_{o_i}]^\top$ be the position vectors of the i-th marker on Σ_s, the camera on Σ_s, and the i-th marker on Σ_c, respectively. The coordinates of the i-th feature point on the image plane are denoted as $\mathbf{f}_i = [u_i, v_i]^\top$.

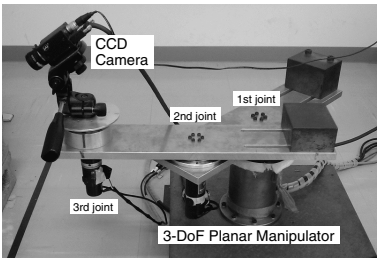


Fig. 19.1 Hand-eye robot

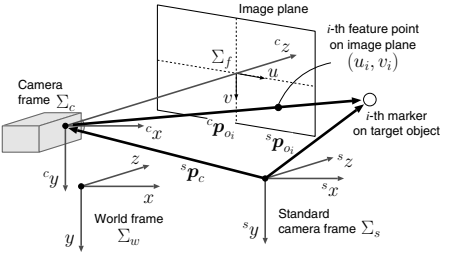


Fig. 19.2 Vision system model

Assume an ideal pinhole camera as an imaging model. Then, from the perspective projection and the 3D geometric relation between the camera and the i-th marker, we obtain

$$\dot{\mathbf{f}}_i = \mathbf{J}_{\rm img}(\mathbf{f}_i, {}^c z_{oi})\, {}^s\mathbf{w}_c - \mathbf{J}^{(1,1)}_{\rm img}(\mathbf{f}_i, {}^c z_{oi})\, {}^s\dot{\mathbf{p}}_{oi}\,, \tag{19.1}$$

where ${}^s\mathbf{w}_c := [\,{}^s\dot{\mathbf{p}}_c^\top, {}^s\boldsymbol{\omega}_c^\top\,]^\top$, $\mathbf{J}_{\rm img}(\mathbf{f}_i, {}^c z_{oi}) = \left[\,\mathbf{J}^{(1,1)}_{\rm img}(\mathbf{f}_i, {}^c z_{oi}), \mathbf{J}^{(1,2)}_{\rm img}(\mathbf{f}_i)\,\right] \in \mathbb{R}^{2\times 6}$,

$$\mathbf{J}^{(1,1)}_{\rm img}(\mathbf{f}_i, {}^c z_{oi}) := \begin{bmatrix} -\frac{\lambda_x}{{}^c z_{oi}} & 0 & \frac{u_i}{{}^c z_{oi}} \\ 0 & -\frac{\lambda_y}{{}^c z_{oi}} & \frac{v_i}{{}^c z_{oi}} \end{bmatrix},\quad \mathbf{J}^{(1,2)}_{\rm img}(\mathbf{f}_i) := \begin{bmatrix} \frac{1}{\lambda_y} u_i v_i & -\left(\lambda_x + \frac{u_i^2}{\lambda_x}\right) & \frac{\lambda_x}{\lambda_y} v_i \\ \lambda_y + \frac{v_i^2}{\lambda_y} & -\frac{1}{\lambda_x} u_i v_i & -\frac{\lambda_y}{\lambda_x} u_i \end{bmatrix},$$

and $\lambda_x,\ \lambda_y$ are the horizontal and vertical focal lengths, respectively. Furthermore, summarizing (19.1) in terms of k feature points, we obtain

$$\dot{\mathbf{f}} = \bar{\mathbf{J}}_{\rm img}(\mathbf{f}, {}^c\mathbf{z}_o)\, {}^s\mathbf{w}_c - \bar{\mathbf{J}}^{(1,1)}_{\rm img}(\mathbf{f}, {}^c\mathbf{z}_o)\, {}^s\dot{\mathbf{p}}_o\,, \tag{19.2}$$

where $\mathbf{f} := [\mathbf{f}_1^\top, \cdots, \mathbf{f}_k^\top]^\top \in \mathbb{R}^{2k}$, ${}^c\mathbf{z}_o := [{}^c z_{o1}, \cdots, {}^c z_{ok}]^\top \in \mathbb{R}^k$, ${}^s\dot{\mathbf{p}}_o := [{}^s\dot{\mathbf{p}}_{o1}^\top, \cdots, {}^s\dot{\mathbf{p}}_{ok}^\top]^\top \in \mathbb{R}^{3k}$, $\bar{\mathbf{J}}_{\rm img}(\mathbf{f}, {}^c\mathbf{z}_o) := [\mathbf{J}^\top_{\rm img}(\mathbf{f}_1, {}^c z_{o1}), \cdots, \mathbf{J}^\top_{\rm img}(\mathbf{f}_k, {}^c z_{ok})]^\top \in \mathbb{R}^{2k\times 6}$, and $\bar{\mathbf{J}}^{(1,1)}_{\rm img}(\mathbf{f}, {}^c\mathbf{z}_o) := {\rm diag}\{\mathbf{J}^{(1,1)}_{\rm img}(\mathbf{f}_1, {}^c z_{o1}), \cdots, \mathbf{J}^{(1,1)}_{\rm img}(\mathbf{f}_k, {}^c z_{ok})\} \in \mathbb{R}^{2k\times 3k}$, respectively. The matrix $\bar{\mathbf{J}}_{\rm img}$ is the so-called *image Jacobian matrix* or *interaction matrix*.

Consider an n-joint robot. Then, ${}^s\mathbf{w}_c$ and $\dot{\mathbf{q}}$ are associated by ${}^s\mathbf{w}_c = {}^c\mathbf{J}(\mathbf{q})\dot{\mathbf{q}}$, where $\mathbf{q} \in \mathbb{R}^n$ is the joint vector. Accordingly, all general solutions of (19.2) are given as

$$\dot{\mathbf{q}} = \mathbf{J}^+_{\rm vis}(\mathbf{f}, {}^c\mathbf{z}_o, \mathbf{q})\left\{\dot{\mathbf{f}} + \bar{\mathbf{J}}^{(1,1)}_{\rm img}(\mathbf{f}, {}^c\mathbf{z}_o)\, {}^s\dot{\mathbf{p}}_o\right\} + \mathbf{J}^\perp_{\rm vis}(\mathbf{f}, {}^c\mathbf{z}_o, \mathbf{q})\,\boldsymbol{\varphi}\,, \tag{19.3}$$

where $\mathbf{J}_{\rm vis} := \bar{\mathbf{J}}_{\rm img}\, {}^c\mathbf{J} \in \mathbb{R}^{2k\times n}$, $\mathbf{J}^+_{\rm vis}$ denotes the pseudo-inverse of $\mathbf{J}_{\rm vis}$, $\mathbf{J}^\perp_{\rm vis} = \mathbf{I}_n - \mathbf{J}^+_{\rm vis}\mathbf{J}_{\rm vis} \in \mathbb{R}^{n\times n}$ is the orthogonal projection operator into the null-space of $\mathbf{J}_{\rm vis}$, $\ker\mathbf{J}_{\rm vis}$, and $\boldsymbol{\varphi} \in \mathbb{R}^n$ is an arbitrary vector, respectively.

Now, the control objective is to keep all feature points around their desired positions on the image plane. We here suppose the target object to be static on Σ_w in visual servoing, i.e. ${}^s\dot{\mathbf{p}}_o \equiv \mathbf{0}_{3k}$. Then, on the basis of (19.3), we design the following desired angular velocity $\dot{\mathbf{q}}^d$ so as to drive $\mathbf{f}$ to $\mathbf{f}^d$:

$$\dot{\mathbf{q}}^d = \mathbf{J}^+_{\rm vis}(\mathbf{f}, {}^c\mathbf{z}_o, \mathbf{q})\left\{-\mathbf{K}_{\rm img}(\mathbf{f} - \mathbf{f}^d)\right\} + \mathbf{J}^\perp_{\rm vis}(\mathbf{f}, {}^c\mathbf{z}_o, \mathbf{q})\,\boldsymbol{\varphi}\,, \tag{19.4}$$

where $\mathbf{K}_{\rm img} = {\rm diag}\{\mathbf{K}_{{\rm img}\,1}, \cdots, \mathbf{K}_{{\rm img}\,k}\}$, $\mathbf{K}_{{\rm img}\,i} := {\rm diag}\{K^u_{{\rm img}\,i}, K^v_{{\rm img}\,i}\} \succ 0$, denotes the positive gain matrix and $\mathbf{f}^d = [\,(\mathbf{f}_1^d)^\top,\ \cdots,\ (\mathbf{f}_k^d)^\top\,]^\top$, $\mathbf{f}_i^d := [\,u_i^d,\ v_i^d\,]^\top$, denotes the desired coordinates of all feature points on the image plane ($i = 1, \ldots, k$). The desired joint angle θ^d can be calculated by a step-by-step integration of $\dot{\mathbf{q}}^d$, such as the Euler method and the Runge-Kutta method. When we implement IBVS to the robot, we adopt a feedback controller so that the robot behaves according to the desired trajectories $(\mathbf{q}^d, \dot{\mathbf{q}}^d)$.

In this paper, we consider the case when $2k < n$ and ${\rm rank}\mathbf{J}_{\rm vis} = 2k$ in (19.4). This case causes $\dot{\mathbf{q}}^d$ to have redundancy for $\boldsymbol{\varphi}$. We therefore need to resolve this

redundancy. In the following section, on the basis of GPM [9], we introduce a PCF for admitting joint range of motion maximally [6] into $\boldsymbol{\varphi}$ in (19.4).

19.3 Maximal Admission of Joint Range of Motion via Redundancy

This section recalls the maximal admission method of joint range of motion [6].

In (19.4), $\dot{\mathbf{q}}^d$ has redundancy for $\boldsymbol{\varphi}$ when $2k < n$ and $\mathrm{rank}\mathbf{J}_{\mathrm{vis}} = 2k$. One of resolution methods for the redundancy is GPM [9]. Setting $\boldsymbol{\varphi} = (\partial V/\partial \mathbf{q})^\top$ in (19.4) according to GPM, we obtain the solution

$$\dot{\mathbf{q}}^d = \mathbf{J}^+_{\mathrm{vis}}(\mathbf{f}, {}^c\mathbf{z}_o, \mathbf{q})\left\{-\mathbf{K}_{\mathrm{img}}(\mathbf{f}-\mathbf{f}^d)\right\} + \mathbf{J}^\perp_{\mathrm{vis}}(\mathbf{f}, {}^c\mathbf{z}_o, \mathbf{q})\left(\frac{\partial V}{\partial \mathbf{q}}\right)^\top \tag{19.5}$$

so as to maximize a given PCF $V(\mathbf{q})$.

For the joint limit avoidance problem, some PCFs have been proposed so far [9, 16, 18, 12]. However, with the conventional functions it is difficult to achieve joint limit avoidance, for example, in the case when a joint variable behaves at the close vicinity of the limit. The authors have proposed a new concept so as to avoid joint limits strictly and to exploit the joint range of motion maximally [6]. Let us call this concept the *maximal admission of joint range of motion*.

Let $V(\mathbf{q}) = k_r \sum_{i=1}^n V_i(q_i)$. Then, conditions that a PCF for admitting joint range of motion maximally should satisfy are defined as follows:

i) Let $\mathscr{N}_i^{\max} := \{\, q_i \mid \bar{q}_i^{\max} \le q_i < q_i^{\max} \,\}$ and $\mathscr{N}_i^{\min} := \{\, q_i \mid q_i^{\min} < q_i \le \bar{q}_i^{\min} \,\}$, where $q_i^{\max}$ and $q_i^{\min}$ are the upper and lower limits on the i-th joint, $\bar{q}_i^{\max} := q_i^{\max} - \rho\Delta q_i$, $\bar{q}_i^{\min} := q_i^{\min} + \rho\Delta q_i$, $\rho \in (0, 1/2)$, and $\Delta q_i := q_i^{\max} - q_i^{\min}$, respectively. Function $V_i(q_i)$ satisfies

$$\frac{\mathrm{d}V_i}{\mathrm{d}q_i} \to \begin{cases} -\infty & \text{as } \ q_i \to q_i^{\max} - 0 \\ 0 & \text{as } \ q_i \to \bar{q}_i^{\max} + 0 \end{cases}, \ \text{for } \ q_i \in \mathscr{N}_i^{\max}$$

$$\text{and} \qquad \frac{\mathrm{d}V_i}{\mathrm{d}q_i} \to \begin{cases} 0 & \text{as } \ q_i \to \bar{q}_i^{\min} - 0 \\ +\infty & \text{as } \ q_i \to q_i^{\min} + 0 \end{cases}, \ \text{for } \ q_i \in \mathscr{N}_i^{\min}.$$

ii) Let $\mathscr{M}_i := \{\, q_i \,|\, q_i^{\min} < q_i < q_i^{\max} \,\}$. For $q_i \in \mathscr{M}_i - (\mathscr{N}_i^{\max} + \mathscr{N}_i^{\min})$, function $V_i(q_i)$ satisfies $\partial V_i/\partial q_i = 0$.

We introduce the region for avoiding joint limits at the neighborhood of the limits in the same way as in [12]. The condition i) means to avoid each joint limit only in the region. On the other hand, the condition ii) means to perform only the primary task (i.e., to do nothing as the secondary task) everywhere except in the region. Every conventional PCF [9, 16, 18, 12] does not satisfy both conditions i) and ii).

Condition i) requires an upward-convex function bounded with respect to q_i. There exist various such functions. In [6], focusing on the boundedness of the tangent function with respect to the domain of definition, we have proposed the

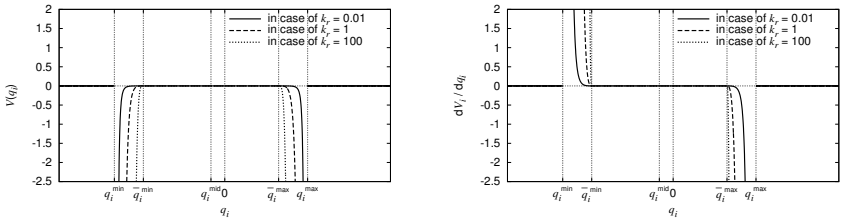


Fig. 19.3 Profiles of proposed tangential function

following tangential function as one candidate of performance criteria which satisfies the above conditions i) and ii):

$$V(\mathbf{q}) = k_r \sum_{i=1}^{n} V_i(q_i), \quad V_i(q_i) = \begin{cases} -\tan^4(\alpha_i(q_i - \bar{q}_i^{\max})), & \text{if } q_i \in \mathscr{N}_i^{\max} \\ -\tan^4(\alpha_i(q_i - \bar{q}_i^{\min})), & \text{if } q_i \in \mathscr{N}_i^{\min} \\ 0, & \text{otherwise} \end{cases}, \tag{19.6}$$

where $\alpha_i := \pi/(2\rho\Delta q_i)$, k_r and ρ are two design parameters. This function is obtained when a tangent function raised to the fourth power is split and patched so as to satisfy condition ii). Differentiating (19.6) with respect to q_i, the gradient vector $\partial V/\partial \mathbf{q} = [\partial V/\partial q_1, \cdots, \partial V/\partial q_n]$ is derived as follows:

$$\frac{\partial V}{\partial q_i} = \frac{\mathrm{d}V_i}{\mathrm{d}q_i} = \begin{cases} -\frac{4k_r\alpha_i \tan^3(\alpha_i(q_i - \bar{q}_i^{\max}))}{\cos^2(\alpha_i(q_i - \bar{q}_i^{\max}))}, & \text{if } q_i \in \mathscr{N}_i^{\max} \\ -\frac{4k_r\alpha_i \tan^3(\alpha_i(q_i - \bar{q}_i^{\min}))}{\cos^2(\alpha_i(q_i - \bar{q}_i^{\min}))}, & \text{if } q_i \in \mathscr{N}_i^{\min} \\ 0, & \text{otherwise} \end{cases}. \tag{19.7}$$

The profiles of the proposed function and its gradient are depicted in 19.3. Using $\alpha(q_i - \bar{q}_i^{\max}) = \pi(q_i - \bar{q}_i^{\max})/(2\rho\Delta q_i) \to \frac{\pi}{2}$ as $q_i \to q_i^{\max} - 0$ and $\alpha(q_i - \bar{q}_i^{\min}) = \pi(q_i - \bar{q}_i^{\min})/(2\rho\Delta q_i) \to -\frac{\pi}{2}$ as $q_i \to q_i^{\min} + 0$, it follows from (19.7) that

$$\lim_{q_i \to q_i^{\max} - 0} \frac{\mathrm{d}V_i}{\mathrm{d}q_i} = -\infty, \quad \lim_{q_i \to \bar{q}_i^{\max} + 0} \frac{\mathrm{d}V_i}{\mathrm{d}q_i} = 0, \quad \lim_{q_i \to \bar{q}_i^{\min} - 0} \frac{\mathrm{d}V_i}{\mathrm{d}q_i} = 0, \quad \lim_{q_i \to q_i^{\min} + 0} \frac{\mathrm{d}V_i}{\mathrm{d}q_i} = \infty.$$

So, it is certain that the gradient (19.7) satisfies both conditions i) and ii). Furthermore, differentiating each element of the gradient vector (19.7) with respect to q_i leads to $\lim_{q_i \to \bar{q}_i^{\max} + 0} \mathrm{d}^2V_i/\mathrm{d}q_i^2 = 0$ and $\lim_{q_i \to \bar{q}_i^{\min} - 0} \mathrm{d}^2V_i/\mathrm{d}q_i^2 = 0$. This means that $\mathrm{d}^2V_i/\mathrm{d}q_i^2$ is continuous in $\mathscr{M}_i$, i.e. $\mathrm{d}V_i/\mathrm{d}q_i$ is a C^1-function in $\mathscr{M}_i$.

In this paper we adopt the proposed tangential function (19.6) (or (19.7)) in (19.5). Consequently, we obtain a desired joint velocity trajectory $\dot{\mathbf{q}}^d$ so as to achieve simultaneously IBVS as the primary task and admitting joint range of motion maximally as the secondary task. Note that the proposed tangent-based function can also adopt another solution, e.g., the Weighted Least-Norm solution [17], the Chaumette *et al.*'s iterative solution [2], the Mansard *et al.*'s solution [11], etc.

19.4 Experiment

This section presents experimental results to evaluate the application of the maximal admission method of joint range of motion to IBVS. The controlled object is a hand-eye robot, which is composed of a 3-dof planar manipulator and a single CCD camera that is mounted on the manipulator's end-effector as depicted in 19.1. For simplicity, we consider a case when the number of feature points is one, i.e., $k = 1$.

19.4.1 Dynamics Linearization Using Disturbance Observer, and PD Feedback Control

We adopt a proportional-differential (PD) feedback control with a disturbance observer [15] to track $(\mathbf{q}^d, \dot{\mathbf{q}}^d)$, similarly as in [7, 8]. It is well known that a feedback control system using the disturbance observer does not need exact parameter identification but is robust against parameter variations and external disturbances.

The dynamics of the hand-eye robot is described as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau}_a - \boldsymbol{\tau}_f + \boldsymbol{\tau}_{\text{ext}} = \mathbf{K}_t \mathbf{i}_a - \boldsymbol{\tau}_f + \boldsymbol{\tau}_{\text{ext}}, \tag{19.8}$$

where $\mathbf{q} \in \mathbb{R}^3$ is the joint vector, $\boldsymbol{\tau}_a \in \mathbb{R}^3$ is the actuator torque vector, $\boldsymbol{\tau}_f$ is the friction torque vector, $\boldsymbol{\tau}_{\text{ext}}$ is the external torque vector, $\mathbf{M} \in \mathbb{R}^{3\times 3}$ is the inertia matrix, $\mathbf{c} \in \mathbb{R}^3$ is the Coriolis and centrifugal force vector, $\mathbf{K}_t = \text{diag}\{K_{t1}, K_{t2}, K_{t3}\}$ is the torque constant matrix, and $\mathbf{i}_a \in \mathbb{R}^3$ is the armature current vector, respectively. In practice, $\mathbf{K}_t$ is not constant but fluctuant. Let $\mathbf{K}_t = \bar{\mathbf{K}}_t + \Delta\mathbf{K}_t$, where $\bar{\mathbf{K}}_t$ and $\Delta\mathbf{K}_t$ are the nominal and fluctuant parts of $\mathbf{K}_t$. Also, we regard $\mathbf{M}$ as $\mathbf{M}(\mathbf{q}) = \mathbf{M}_d(\mathbf{q}) + \mathbf{M}_c(\mathbf{q})$, where $\mathbf{M}_d$ and $\mathbf{M}_c$ are the diagonal part and the remainder of $\mathbf{M}$. Furthermore, we suppose that $\mathbf{M}_d$ is divided into the nominal part $\bar{\mathbf{M}}_d$ and the remainder $\Delta\mathbf{M}_d$ as $\mathbf{M}_d(\mathbf{q}) = \bar{\mathbf{M}}_d + \Delta\mathbf{M}_d(\mathbf{q})$. Then, (19.8) can be rewritten as

$$\bar{\mathbf{M}}_d\ddot{\mathbf{q}} = \bar{\mathbf{K}}_t\mathbf{i}_a - \boldsymbol{\tau}_{\text{dis}} \;\Leftrightarrow\; \ddot{\mathbf{q}} = \bar{\mathbf{M}}_d^{-1}(\bar{\mathbf{K}}_t\mathbf{i}_a - \boldsymbol{\tau}_{\text{dis}}), \tag{19.9}$$

where $\boldsymbol{\tau}_{\text{dis}} := \{\Delta\mathbf{M}_d + \mathbf{M}_c\}\ddot{\mathbf{q}} + \mathbf{c} - \Delta\mathbf{K}_t\mathbf{i}_a + \boldsymbol{\tau}_f - \boldsymbol{\tau}_{\text{ext}}$ is the disturbance torque.

To estimate $\boldsymbol{\tau}_{\text{dis}}$, we adopt the following disturbance observer [15]:

$$\hat{\mathbf{T}}_{\text{dis}}(s) = \mathbf{G}_{\text{LP}}(s)\{\bar{\mathbf{K}}_t\mathbf{I}_a(s) - \bar{\mathbf{M}}_d s \cdot s\mathbf{Q}(s)\}, \tag{19.10}$$

where $\hat{\mathbf{T}}_{\text{dis}}(s) := \mathscr{L}[\hat{\boldsymbol{\tau}}_{\text{dis}}(t)]$, $\mathbf{I}_a(s) := \mathscr{L}[\mathbf{i}_a(t)]$, $\mathbf{Q}(s) := \mathscr{L}[\mathbf{q}(t)]$, $\mathbf{G}_{\text{LP}}(s) = \{\omega/(s+\omega)\}\mathbf{I}_3$ is the transfer function matrix of the first-order low-pass filter, ω is the cutoff frequency and $\mathbf{I}_3$ is the third order identity matrix. The block diagram of the system (19.9) and the disturbance observer (19.10) are shown in 19.4. A disturbance torque compensation $\mathbf{i}_a = \bar{\mathbf{K}}_t^{-1}\bar{\mathbf{M}}_d\mathbf{u} + \hat{\boldsymbol{\tau}}_{\text{dis}}$ with a new input $\mathbf{u} \in \mathbb{R}^3$ yields the linearized and decoupled system $\ddot{\mathbf{q}} = \mathbf{u}$, if $\hat{\boldsymbol{\tau}}_{\text{dis}}$ is close enough to $\boldsymbol{\tau}_{\text{dis}}$.

Finally, adopting a PD feedback control law, we can obtain a closed-loop system

$$\ddot{\mathbf{q}}(t) = -\mathbf{K}_p(\mathbf{q}(t) - \mathbf{q}^d) - \mathbf{K}_v(\dot{\mathbf{q}}(t) - \dot{\mathbf{q}}^d), \tag{19.11}$$

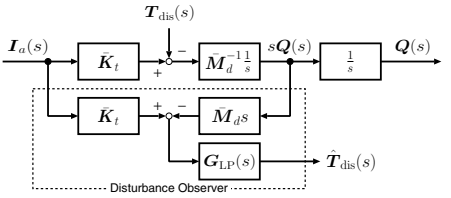


Fig. 19.4 Block diagram of the system and the disturbance observer

where the superscript 'd' refers to the desired value. Gain matrices $\mathbf{K}_p$ and $\mathbf{K}_v$ are positive definite diagonal matrices. Using the PD control law with well-tuned gain matrices, the manipulator dynamics behaves according to the desired trajectories.

19.4.2 Experimental Setup and Results

The 3-dof planar manipulator is controlled by a PC running a real-time Linux OS. The Linux kernel is patched with Xenomai (http://www.xenomai.org/). The control period is 1 ms. The rotating angle of each joint is obtained from an encoder attached to each DC servo motor via a counter board. Armature currents based on (19.11), (19.4) and (19.7) are interpolated to each DC servo motor by a D/A board through each DC servo driver.

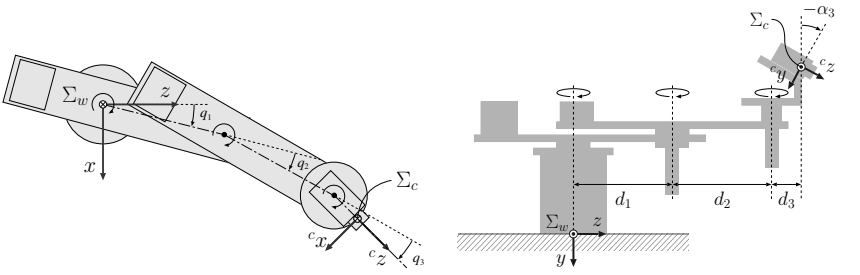
Here we consider the manipulator model as depicted in 19.5, where q_i, α_i, and d_i denote the i-th joint angle, the tilt angle, and the distance from the $(i-1)$-th joint to the i-th joint, respectively. Note that ${}^s\mathbf{w}_c$ and $\dot{\mathbf{q}}$ are associated by the relation

$$
{}^s\mathbf{w}_c = {}^c\mathbf{J}(\mathbf{q}_{23})\dot{\mathbf{q}} = \begin{bmatrix} d_1\mathrm{C}_{23} + d_2\mathrm{C}_3 + d_3 & d_2\mathrm{C}_3 + d_3 & d_3 \\ -d_1\mathrm{S}_{\alpha_3}\mathrm{S}_{23} - d_2\mathrm{S}_{\alpha_3}\mathrm{S}_3 & -d_2\mathrm{S}_{\alpha_3}\mathrm{S}_3 & 0 \\ d_1\mathrm{C}_{\alpha_3}\mathrm{S}_{23} + d_2\mathrm{C}_{\alpha_3}\mathrm{S}_3 & d_2\mathrm{C}_{\alpha_3}\mathrm{S}_3 & 0 \\ 0 & 0 & 0 \\ \mathrm{C}_{\alpha_3} & \mathrm{C}_{\alpha_3} & \mathrm{C}_{\alpha_3} \\ \mathrm{S}_{\alpha_3} & \mathrm{S}_{\alpha_3} & \mathrm{S}_{\alpha_3} \end{bmatrix} \dot{\mathbf{q}}, \tag{19.12}
$$

where $\mathbf{q}_{23} := [q_2,\ q_3]^\top$, $\mathrm{S}_{ijk} := \sin(q_i + q_j + q_k)$, $\mathrm{C}_{ijk} := \cos(q_i + q_j + q_k)$, $\mathrm{S}_{\alpha_3} := \sin\alpha_3$, and $\mathrm{C}_{\alpha_3} := \cos\alpha_3$, respectively. The physical and nominal parameters of the manipulator are as follows: $(q_1^{\max}, q_2^{\max}, q_3^{\max}) = (-q_1^{\min}, -q_2^{\min}, -q_3^{\min}) = (\pi/2\,\mathrm{rad}, 5\pi/18\,\mathrm{rad}, \pi/2\,\mathrm{rad})\,(=(90^\circ, 50^\circ, 90^\circ))$, $d_1 = d_2 = 0.2\,\mathrm{m}$, $d_3 = 0.047\,\mathrm{m}$, $\alpha_3 = \pi/6\ (=30^\circ)$, $\bar{\mathbf{M}}_d = \mathrm{diag}\{0.65\,\mathrm{kg{\cdot}m^2}, 0.25\,\mathrm{kg{\cdot}m^2}, 0.06\,\mathrm{kg{\cdot}m^2}\}$, $\bar{\mathbf{K}}_t = \mathrm{diag}\{21\,\mathrm{N{\cdot}m/A}, 17.64\,\mathrm{N{\cdot}m/A}, 4.96\,\mathrm{N{\cdot}m/A}\}$.

The image resolution and the focal lengths of the CCD camera are 640×480 pixels, $\lambda_x = 800.0$ pixels, and $\lambda_y = 796.4$ pixels, respectively. The frame rate of the camera is 120 fps. Hence, the image data is updated every 8.3 ms. The coordinate of the feature point is calculated as the barycentric coordinate of the marker area on the binarized image.

The single camera system can not measure directly the depth on Σ_c, i.e. ${}^cz_{o_1}$. The matrix $\mathbf{J}_{\rm img}$ which consists $\mathbf{J}_{\rm vis}$ depends on the depth ${}^cz_{o_1}$, and therefore we need



(a) (b)

Fig. 19.5 Manipulator model: (a) top view, (b) side view in the case of $\mathbf{q} = \mathbf{0}_3$

to estimate it. Similarly to our previous work in [7, 8], we collected some measured data of set $({}^c z_{o_1}, v_1)$ in advance and fitted them to a second-order polynomial function with respect to v_1. The depth ${}^c z_{o_1}$ is estimated by ${}^c \hat{z}_{o_1}(v_1) = a v_1^2 + b v_1 + c$, where $a = 1.07348 \times 10^{-6}$, $b = -6.24461 \times 10^{-4}$, $c = 3.10071 \times 10^{-1}$.

The procedure of the experiment is as follows:

Step 1: We set $\mathbf{q}^d = (-\pi/4\,\text{rad}, 2\pi/9\,\text{rad}, \pi/36\,\text{rad})$ $(= (-45^\circ, 40^\circ, 5^\circ))$ and $\dot{\mathbf{q}}^d = \mathbf{0}_3$ in (19.11) to drive the hand-eye robot to the initial configuration, so that the target object is inside the boundaries of the image plane and the 3-dof planar manipulator does not configure a kinematic singularity[1]. In consequence, the feature point is located around $\mathbf{f} = (140\,\text{pixels}, 60\,\text{pixels})$.

Step 2: We start visual servoing based on (19.11), (19.4), and (19.6) with $\mathbf{f}^d = (0, 0)$.

Step 3: When two seconds have passed, visual servoing is finished.

The experiments were carried out with design parameters $\omega = 150\,\text{rad/s}$, $\mathbf{K}_p = \text{diag}\{144, 144, 144\}$, $\mathbf{K}_v = \text{diag}\{48, 48, 48\}$, $\mathbf{K}_{\text{img}} = \text{diag}\{4, 4\}$, $\rho = 0.01$. The experimental results are shown in Figs. 19.6–19.8. For comparison, Figs. 19.7 and 19.8 show the results in the cases of the Zghal-type [18] and the Marchand-type [12] functions. These are typical performance functions for avoiding joint limits. Each graph includes the case of three kinds of k_r and the case of no redundancy ($V = 0$).

The two graphs in the second row of 19.6 show that the feature point $\mathbf{f}$ converged to $\mathbf{f}^d$, i.e. visual servoing as the primary task was achieved, at about 1 s in every case. We here omit time responses of $\mathbf{f}$ in the case of the Zghal-type and Marchand-type functions, because these are similar to the case of the proposed tangential function. On the other hand, the results for joint limit avoidance are as follows: in the cases of the proposed and the Zghal-type functions, the joint limit avoidance succeeded without depending on the value of k_r; in the case of the Marchand-type function, the second joint angle exceeded its limit depending on the value of k_r.

Focusing on the time response of q_2 after $\mathbf{f}$ converged to $\mathbf{f}^d$, we can find a difference between the cases. In the cases of the proposed tangential function and the

[1] Note that when $(q_2, q_3) = (\pm k_2 \pi\,\text{rad}, \pm k_3 \pi\,\text{rad})$, $\forall k_2, k_3 \in \mathbb{Z}$, the manipulator configures kinematic singularities.

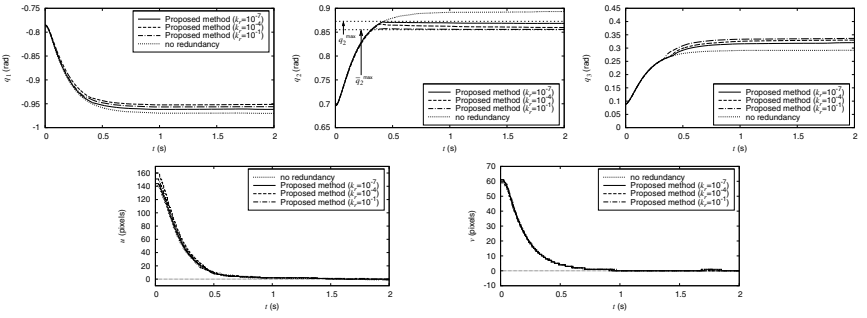


Fig. 19.6 Experimental results in the case of the proposed tangential function

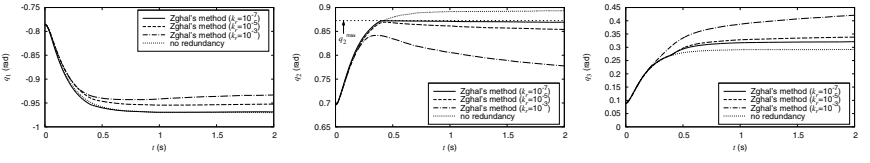


Fig. 19.7 Experimental results in the case of the Zghal-type function

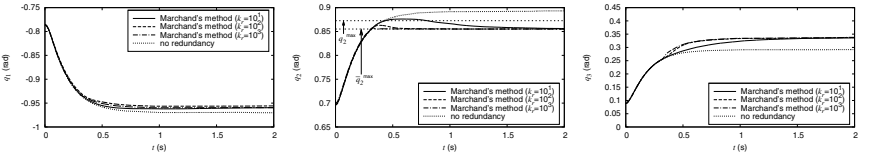


Fig. 19.8 Experimental results in the case of the Marchand-type function

Marchand-type function, q_2 converges to the neighborhood of the braking angle and $\bar{q}_2^{\max}$, respectively. This means that q_2 stays in $\mathcal{N}_2^{\max}$ in both cases. Meanwhile, in the case of the Zghal-type function, q_2 goes to zero, i.e. a kinematic singularity. This case needs avoidance of not only joint limits but also kinematic singularities.

As described above, we confirmed that in IBVS the proposed tangential function can achieve both avoiding joint limits and exploiting joint range of motion effectively, i.e. admitting joint range of motion maximally. Therefore the effectiveness of the application was evaluated experimentally.

19.5 Conclusions

In this paper, we applied the maximal admission method of joint range of motion using redundancy to IBVS in an eye-in-hand system. The effectiveness was demonstrated in an experiment. The difference between the proposed and conventional methods was also discussed in the experimental results. Considering not only

physical limits but also effective use of limited resources, such as the proposed method, is a necessity for achieving more dynamic and complex robot motion.

Acknowledgements. This research was partially supported by a research grant from The Murata Science Foundation.

References

1. Chaumette, F., Hutchinson, S.: Visual servoing and visual tracking. In: Siciliano, B., Khatib, O. (eds.) Springer Handbook of Robotics, ch. 24, pp. 563–583. Springer, Heidelberg (2008)
2. Chaumette, F., Marchand, E.: A redundancy-based iterative approach for avoiding joint limits: application to visual servoing. IEEE Transactions on Robotics and Automation 17(5), 719–730 (2001)
3. Chiaverini, S., Oriolo, G., Walker, I.D.: Kinematically redundant manipulators. In: Siciliano, B., Khatib, O. (eds.) Springer Handbook of Robotics, ch.11, pp. 245–268. Springer, Heidelberg (2008)
4. Corke, P., Hutchinson, S.: A new partitioned approach to image-based visual servo control. IEEE Transactions on Robotics and Automation 17(4), 507–515 (2001)
5. Deguchi, K.: Optimal motion control for image-based visual servoing by decoupling translation and rotation. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 1998), Victoria, Canada, pp. 705–711 (1998)
6. Ito, M., Kawatsu, K., Shibata, M.: Maximal admission of joint range of motion for kinematically redundant manipulators with joint limits. In: Proceedings of The UKACC International Conference on Control (CONTROL 2010), Coventry, UK, pp. 489–494 (2010)
7. Ito, M., Shibata, M.: Non-delayed visual tracking of hand-eye robot for a moving target object. In: Proceedings of ICROS-SICE International Joint Conference 2009 (ICCAS-SICE 2009), Fukuoka, Japan, pp. 4035–4040 (2009)
8. Ito, M., Shibata, M.: Visual tracking of a hand-eye robot for a moving target object with multiple feature points: Translational motion compensation approach. Advanced Robotics 25(3), 355–369 (2011)
9. Liégeois, A.: Automatic supervisory control of the configuration and behavior of multibody mechanisms. IEEE Transactions on Systems, Man, and Cybernetics SMC-7(12), 868–871 (1977)
10. Malis, E., Chaumette, F., Boudet, S.: 2-1/2-D visual servoing. IEEE Transactions on Robotics and Automation 15(2), 238–250 (1999)
11. Mansard, N., Chaumette, F.: Directional redundancy for robot control. IEEE Transactions on Automatic Control 54(6), 1179–1192 (2009)
12. Marchand, E., Chaumette, F., Rizzo, A.: Using the task function approach to avoid robot joint limits and kinematic singularities in visual servoing. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 1996), Osaka, Japan, vol. 3, pp. 1083–1090 (1996)
13. Marey, M., Chaumette, F.: New strategies for avoiding robot joint limits: Application to visual servoing using a large projection operator. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2010), Taipei, Taiwan, pp. 6222–6227 (2010)

14. Nelson, B.J., Khosla, P.K.: Strategies for increasing the tracking region of an eye-in-hand system by singularity and joint limit avoidance. International Journal of Robotics Research 14(3), 255–269 (1995)
15. Ohnishi, K., Shibata, M., Murakami, T.: Motion control for advanced mechatronics. IEEE/ASME Transactions on Mechatronics 1(1), 56–67 (1996)
16. Tsai, M.J.: Workspace geometric characterization and manipulability of industrial robots. Ph.D. thesis, Department of Mechanical Engineering, Ohio State University (1986)
17. Whitney, D.E.: The mathematics of coordinated control of prosthetic arms and manipulators. ASME Journal of Dynamic Systems, Measurement, and Control 94(4), 303–309 (1972)
18. Zghal, H., Dubey, R.V., Euler, J.A.: Efficient gradient projection optimization for manipulators with multiple degrees of redundancy. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA 1990), Cincinnati, OH, USA, vol. 2, pp. 1006–1011 (1990)

Chapter 20
Optimal Extended Jacobian Inverse Kinematics Algorithm with Application to Attitude Control of Robotic Manipulators

Joanna Karpińska and Krzysztof Tchoń

Abstract. We study the approximation problem of Jacobian inverse kinematics algorithms for robotic manipulators. A novel variational formulation of the problem is explored in the context of the optimal approximation of the Jacobian pseudo inverse algorithm by the extended Jacobian algorithm for the coordinate-free definition of the manipulator's kinematics. The attitude control problem of a robotic manipulator is solved as an illustration of the approach.

20.1 Introduction

The kinematics of a robotic manipulator defines the position and the orientation of the manipulator's end effector with respect to an inertial coordinate frame as a function of the manipulator's joint positions,

$$\mathscr{K} : \mathbf{R}^n \longrightarrow SE(3), \qquad Y = \mathscr{K}(q) = \begin{bmatrix} R(q) & T(q) \\ 0 & 1 \end{bmatrix}, \tag{20.1}$$

where $SE(3) \cong SO(3) \times \mathbf{R}^3$ denotes the special Euclidean group of the rigid body motions. Usually, the kinematics (20.1) comes from the Denavit-Hartenberg algorithm. Given the kinematics (20.1), the standard inverse kinematic problem amounts to determining a joint position such that the end effector reaches a desirable $Y_d \in SE(3)$. In this work we shall address the inverse kinematic problem restricted solely to the rotational motion, consisting in defining a joint position such that the manipulator's end effector takes a desirable orientation $R_d \in SO(3)$. Further on, this problem will be referred to as the manipulator attitude control problem. Observe

Joanna Karpińska · Krzysztof Tchoń
Institute of Computer Engineering, Control and Robotics,
Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: {joanna.karpinska,krzysztof.tchon}@pwr.wroc.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 237–246.
springerlink.com

that a solution to the attitude control problem is a prerequisite for tracking the manipulator's attitude at the dynamic level.

The attitude control problem originates from the analysis of the rigid body dynamics [1]. The existing literature provides numerous feedback control algorithms of the attitude of rigid body [2, 3, 4], at the kinematic as well as at the dynamic level. Similarly to [5], here we shall exploit the attitude control problem as a demanding framework for the development of inverse kinematics/motion planning algorithms for robotic manipulators. Being a specific inverse kinematic problem, the manipulator attitude control problem can be solved using a Jacobian inverse kinematics algorithm based on a right inverse of the Jacobian of the manipulator. In this work we shall concentrate on two Jacobian algorithms: the Jacobian pseudo inverse and the extended Jacobian algorithm [6, 7]. The former is distinguished by its efficient convergence, the latter has the property of repeatability [8, 9]. We remind that repeatability means that the inverse kinematics algorithm converts closed paths in the task space into closed paths in the joint space. An attempt at providing an inverse kinematics algorithm being both efficiently convergent and repeatable was originally undertaken in [10, 11], where the problem of optimal approximation was addressed of the Jacobian pseudo inverse by an extended Jacobian inverse kinematics algorithm. A complementary approximation error functional, leading to more tractable optimality conditions has recently been introduced in [12]. Both these conditions come from the calculus of variations, they require that the kinematics of the manipulator is expressed in coordinates, and result in a collection of second-order partial differential equations for the augmenting kinematic map. The partial differential equations derived in [10, 11] are nonlinear, whereas ours are linear elliptic.

The main contribution of this work consists in showing that the approach developed in [12] applies to the manipulator attitude control problem, formulated in a coordinate-free setting. To cope with this problem we shall introduce a logarithmic measure of the attitude error, define the extended error Jacobian, and finally obtain the optimal extended Jacobian inverse kinematics algorithm. To illustrate the applicability of our approach, the attitude control problem will be addressed of a redundant wrist being a 4 d.o.f. derivative of the Stanford manipulator. The extended Jacobian algorithm is used, obtained by solving the approximation problem by means of the Ritz method. Performance and convergence of the optimal algorithm have been compared with those provided by the Jacobian pseudo inverse. It is worth mentioning that an alternative to the variational calculus, differential geometric solution of the approximation problem, based on the approximation of a non-integrable co-distribution by an integrable one, has been presented in [13]. A case study of variational and differential geometric approaches is dealt with in [14].

The composition of the remaining part of this work is the following. Section 20.2 introduces basic concepts and brings a statement of the approximation problem in the coordinate setting. The manipulator attitude control problem is dealt with in Section 20.3. Section 20.4 presents a solution to the attitude control problem for the redundant wrist. Section 20.5 provides conclusions.

20.2 Basic Concepts

For the reader's convenience we shall explain the basic concepts using a coordinate representation of the manipulator's kinematics. In Section 20.3 our conclusions will be adopted to the coordinate-free setting. Let this coordinate representation take the form of a map

$$k : \mathbf{R}^n \longrightarrow \mathbf{R}^m, \quad y = k(q), \tag{20.2}$$

transforming joint space coordinates into task space coordinates. We assume that the kinematics is redundant, i.e. $n > m$. The inverse kinematics problem for the kinematics (20.2) means computing a joint position $q_d \in \mathbf{R}^n$ such that the manipulator's end effector reaches a desirable point $y_d \in \mathbf{R}^m$ in the task space. Usually, the inverse kinematics problem is solved by a Jacobian inverse kinematics algorithm obtained in the following way. We choose a smooth joint trajectory $q(t)$, compute the error $e(t) = k(q(t)) - y_d$, and request that the error decays exponentially with a rate $\gamma > 0$. This request leads to the error differential equation

$$\dot{e} = J(q)\dot{q} = -\gamma e(t), \tag{20.3}$$

where $J(q) = \frac{\partial k}{\partial q}(q)$ denotes the analytic Jacobian. Suppose that $J^{\#}(q)$ is any right inverse of the Jacobian, so that $J(q)J^{\#}(q) = I_m$. Then, the Jacobian inverse kinematics algorithm can be defined by the dynamic system

$$\dot{q} = -\gamma J^{\#}(q)(k(q) - y_d). \tag{20.4}$$

It is easily checked that the trajectory of (20.4) exponentially converges to a solution of the inverse kinematic problem, $\lim_{t\to+\infty} q(t) = q_d$. As a rule, the existence of the Jacobian right inverse requires that the manipulator's joints stay away of kinematic singularities. A well-known example of the Jacobian right inverse is the Jacobian pseudo inverse defined as $J^{P\#}(q) = J^T(q)M^{-1}(q)$, where $M(q) = J(q)J^T(q)$ denotes the manipulability matrix. Alternatively, the right inverse can be obtained by the following Jacobian extension procedure. Setting $s = n - m$, we introduce an augmenting kinematics map

$$h : \mathbf{R}^n \longrightarrow \mathbf{R}^s, \quad \tilde{y} = h(q) = (h_1(q), \ldots, h_s(q)), \tag{20.5}$$

such that the extended kinematics

$$l = (k,h) : \mathbf{R}^n \longrightarrow \mathbf{R}^n, \quad \bar{y} = (k(q), h(q)) \tag{20.6}$$

becomes a local diffeomorphism of $\mathbf{R}^n$, i.e. the extended Jacobian $\bar{J}(q) = \begin{bmatrix} \frac{\partial k(q)}{\partial q} \\ \frac{\partial h(q)}{\partial q} \end{bmatrix}$ is an $n \times n$ invertible matrix. This being so, we define the extended Jacobian inverse as

$$J^{E\#}(q) = \bar{J}^{-1}(q)|_{m\ first\ columns}, \tag{20.7}$$

i.e. a matrix comprising the columns no. $1\ldots,m$ of the inverse extended Jacobian. By definition, $J^{E\#}(q)$ is a right inverse of the Jacobian, furthermore it is annihilated by the differential of the augmenting map, i.e. $\frac{\partial h(q)}{\partial q}J^{E\#}(q)=0$.

We recall that, for given inverses $J^{P\#}(q)$ and $J^{E\#}(q)$, the approximation problem studied in [10, 11] amounts to determining the augmenting map (20.5) that minimizes the error functional

$$\mathscr{E}_1(h)=\int\limits_{\mathscr{Q}}||J^{P\#}(q)-J^{E\#}(q)||_F^2dq, \tag{20.8}$$

where $||M||_F=\sqrt{tr(MM^T)}$ denotes the Frobenius norm of matrix M, and the integration is accomplished over a singularity-free subset $\mathscr{Q}$ of the joint space. A complementary approach set forth in [12] relies on two appealing embeddings of the extended Jacobian inverse

$$A(q)=\begin{bmatrix}J(q)\\ \frac{\partial h(q)}{\partial q}\end{bmatrix}^{-1}=\left[J^{E\#}(q)\ Q(q)\right],$$

$Q(q)$ being a certain matrix, and

$$B(q)=\begin{bmatrix}J(q)\\ K^T(q)\end{bmatrix}^{-1}=\left[J^{P\#}(q)\ K(q)\right]$$

of the Jacobian pseudo inverse. The columns of the matrix $K(q)$ span the Jacobian null space, $J(q)K(q)=0$, and are assumed mutually orthogonal $K^T(q)K(q)=I_s$. Consequently, in a region $\mathscr{Q}\subset\mathbf{R}^n$ of regular configurations the distance between $J^{P\#}(q)$ and $J^{E\#}(q)$ can be computed in the following way

$$\mathscr{E}_2(h)=\int\limits_{\mathscr{Q}}||A^{-1}(q)B(q)-I_n||_F^2m(q)dq=$$
$$\int\limits_{\mathscr{Q}}tr\left(\frac{\partial h(q)}{\partial q}P(q)\left(\frac{\partial h(q)}{\partial q}\right)^T-2\frac{\partial h(q)}{\partial q}K(q)+I_s\right)m(q)dq, \tag{20.9}$$

where

$$P(q)=J^{P\#}(q)J^{P\#T}(q)+K(q)K^T(q), \tag{20.10}$$

and $m(q)=\sqrt{\det M(q)}$ denotes the manipulability of the configuration q. A formal derivation and a justification of the error functional formula (20.9) can be found in [12]. It follows that the Euler-Lagrange equations for the error functional $\mathscr{E}_2(h)$ assume the form of a linear elliptic partial differential equation for every component $h_j(q)$ of the augmenting kinematics map (20.5), $j=1,\ldots,s$. For very basic robot kinematics these equations can be solved either analytically or numerically using available general purpose software packages [12, 14]. However, in more realistic

situations we need to resort to the direct methods of minimizing the error functional, due to Ritz or Galerkin [15].

In order to reveal advantages of the functional (20.9) relevant to the Ritz method, suppose that $s = 1$, and we are seeking for the augmenting function $h(q) = c^T \Phi(q)$, where $c \in R^p$ is a vector of parameters, and $\Phi(q) = (\varphi_1(q), \ldots, \varphi_p(q))$ denotes a vector of basic functions. We compute $dh(q) = c^T \frac{\partial \Phi(q)}{\partial q}$, and conclude that the approximation error becomes a quadratic form

$$\mathscr{E}_2(c) = c^T Dc - 2c^T E, \tag{20.11}$$

where

$$D = \int_{\mathscr{Q}} \frac{\partial \Phi(q)}{\partial q} P(q) \left(\frac{\partial \Phi(q)}{\partial q} \right)^T m(q) dq \quad \text{and} \quad E = \int_{\mathscr{Q}} \frac{\partial \Phi(q)}{\partial q} K(q) m(q) dq.$$

It is easily checked that the minimum of (20.11) is achieved for $c^* = D^{-1}E$.

20.3 Manipulator Attitude Control Problem

Given the rotational part of the kinematics (20.1), $Y = \mathscr{K}(q) = R(q) \in SO(3)$, we shall address the following manipulator attitude control problem: find the vector of joint positions $q_d \in \mathbf{R}^n$ such that the manipulator attains the prescribed, desirable attitude $R_d \in SO(3)$, so that $\mathscr{K}(q_d) = R_d$. It will be assumed that the manipulator is attitude redundant, which means $n \geq 3$. To proceed further, we need to introduce the attitude error. By analogy to [5], this error

$$E = \log\left(R(q) R_d^T\right) \tag{20.12}$$

is defined by means of the matrix logarithm $\log R = \frac{\varphi}{2\sin\varphi}\left(R - R^T\right)$, where $0 \leq \varphi < \pi$ denotes the rotation angle satisfying the identity $1 + 2\cos\varphi = trR$. Using the standard isomorphism $[(v_1, v_2, v_3)] = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}$ of $\mathbf{R}^3$ and the Lie algebra $so(3)$, the attitude error can be conveniently represented by a vector $e \in \mathbf{R}^3$, so that $E = [e]$. Now, let us choose a joint trajectory $q(t)$, and set $R(q(t)) = R(t)$. The corresponding attitude error

$$E(t) = [e(t)] = \log\left(R(t) R_d^T\right).$$

By definition, we have $R(t) R_d^T = \exp(E(t))$. The time derivative

$$\frac{d}{dt} \exp(E(t)) \exp(-E(t)) = \dot{R}(t) R_d^T \left(R(t) R_d^T\right)^T = \dot{R}(t) R^T(t)$$

can be computed using the Hausdorff formula, that yields [16]

$$\frac{d}{dt}\exp(E(t))\exp(-E(t)) = [V(t)\dot{e}],$$

for $V(t) = I_3 + \frac{1-\cos\varphi(t)}{\varphi^2(t)}[e(t)] + \frac{\varphi(t)-\sin\varphi(t)}{\varphi^3(t)}[e(t)]^2$, and $1+2\cos\varphi(t) = tr\left(R(t)R_d^T\right)$. On the other hand,

$$\dot{R}(t)R^T(t) = \sum_{i=1}^{3}\frac{\partial R(t)}{\partial q_i}R^T(t)\dot{q}_i = \sum_{i=1}^{3}[s_i(t)]\dot{q}_i = [S(t)\dot{q}],$$

where $S(t) = \left[s_1(t),\ s_2(t),\ s_3(t)\right]$ is the manipulator Jacobian. Actually, the matrices W and S depend on t through the joint position q, therefore $W(t) = W(q(t))$ and $S(t) = S(q(t))$. Summarizing these developments, we conclude that

$$\dot{e} = W(t)S(t)\dot{q} = J_d(q(t))\dot{q}, \tag{20.13}$$

where

$$W(t) = V^{-1}(t) = I_3 - \frac{1}{2}[e(t)] + \left(\frac{1}{\varphi^2(t)} + \frac{\sin\varphi(t)}{2\varphi(t)(\cos\varphi(t)-1)}\right)[e(t)]^2.$$

The matrix $J_d(q) = W(q)S(q)$ appearing in (20.13) will be referred to as the manipulator's attitude Jacobian. The subscript "d" points out that this Jacobian depends on the desirable attitude R_d. The attitude control problem will be solved, if the error (20.13) satisfies the equation (20.3). After applying the reasoning presented in Section 20.2 to the Jacobian $J_d(q)$, we shall obtain the Jacobian pseudo inverse, and define the extended Jacobian algorithm that approximates the former in the sense of minimizing the functional (20.9). It can be shown that although the manipulator attitude Jacobian depends on R_d, the corresponding Jacobian pseudo inverse and the extended Jacobian inverse kinematics algorithms (20.4) can be made independent of the desirable attitude.

20.4 Case Study

As an illustration of our design of the extended Jacobian algorithm, we shall solve the attitude control problem for a redundant wrist shown in Fig. 20.1, that can be regarded as a certain 4 d.o.f. sub-manipulator of the Stanford manipulator. The attitude of the wrist is defined by the rotation matrix

$$R(q) = \begin{bmatrix} -c_4s_1s_3 + c_1\left(c_2c_3c_4 - s_2s_4\right) & s_1s_3s_4 - c_1\left(c_4s_2 + c_2c_3s_4\right) & c_3s_1 + c_1c_2s_3 \\ c_3c_4s_2 + c_2s_4 & c_2c_4 - c_3s_2s_4 & s_2s_3 \\ s_1\left(s_2s_4 - c_2c_3c_4\right) - c_1c_4s_3 & c_4s_1s_2 + \left(c_2c_3s_1 + c_1s_3\right)s_4 & c_1c_3 - c_2s_1s_3 \end{bmatrix},$$

where $q = (q_1, \ldots, q_4)$, while s_i and c_i denote, respectively, $\sin q_i$ and $\cos q_i$.

Our objective consists in determining an augmenting kinematics function $h(q)$ that minimizes the error functional $\mathscr{E}_2(h)$ defined by (20.9). To this aim, we shall use the Ritz method along the lines sketched at the end of Section 20.2, so we set

$$h(q) = \sum_{i=1}^{p} c_i \varphi_i(q) = c^T \Phi(q),$$

compute the data

$$D = \int_{\mathscr{Q}} \frac{\partial \Phi(q)}{\partial q} P(q) \left(\frac{\partial \Phi(q)}{\partial q} \right)^T m(q) dq \quad \text{and} \quad E = \int_{\mathscr{Q}} \frac{\partial \Phi(q)}{\partial q} K(q) m(q) dq$$

that determine the quadratic form (20.11), and finally, find the minimum of this form at the point $c^* = D^{-1}E$. The matrices $P(q)$ and $K(q)$ are to be computed for the manipulator Jacobian $S(q)$, similarly the volume form $m(q)$.

The computations have been performed for linear and quadratic polynomial basic functions $\varphi_i(q)$ without the constant term, so we have either

$$h_1(q) = c_1 q_1 + c_2 q_2 + c_3 q_3 + c_4 q_4$$

or

$$h_2(q) = c_1 q_1 + c_2 q_2 + c_3 q_3 + c_4 q_4 + c_5 q_1 q_2 + c_6 q_1 q_3 + c_7 q_1 q_4 + c_8 q_2 q_3 + c_9 q_2 q_4 + c_{10} q_3 q_4 + c_{11} q_1^2 + c_{12} q_2^2 + c_{13} q_3^2 + c_{14} q_4^2.$$

The singularity-free region of the joint space is chosen as

$$\mathscr{Q} = \left\{ q \in R^4 \,|\, -\frac{\pi}{2} \le q_1 \le \frac{\pi}{2},\ 0.01 \le q_2 \le \frac{\pi}{2},\ 0 \le q_3 \le \frac{\pi}{2},\ -\frac{\pi}{2} \le q_4 \le \frac{\pi}{2} \right\}.$$

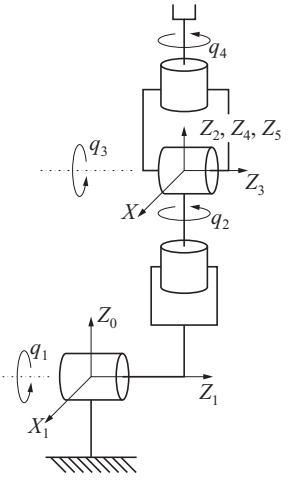


Fig. 20.1 Redundant wrist

The following results have been obtained: the linear optimal augmenting kinematics function is equal to

$$h_1(x) = -0.454353q_1 - 0.237304q_2 + 0.231516q_3 + 0.464759q_4,$$

while the quadratic optimal function is

$$\begin{aligned} h_2(q) = &-0.461795q_1 + 0.136588q_2 - 0.150693q_3 + 0.46637q_4 + \\ &0.0360187q_1q_2 - 0.0992883q_1q_3 - 0.000570661q_1q_4 + \\ &0.00524534q_2q_3 + 0.097564q_2q_4 - 0.0365793q_3q_4 - \\ &0.0189813q_1^2 - 0.247838q_2^2 + 0.250601q_3^2 + 0.0180467q_4^2. \end{aligned}$$

The error functional $\mathscr{E}_2(h_1) = 10.0521$ for the linear case, and $\mathscr{E}_2(h_2) = 5.38958$ for the quadratic augmenting kinematics function. Using these functions, we have found a pair of extended Jacobian algorithms denoted, respectively, $J_1^{E\#}(q)$ and $J_2^{E\#}(q)$.

Performance and convergence of the extended Jacobian and the Jacobian pseudo inverse algorithms are compared in Fig. 20.2 below. In the computations, we set the initial state $q(0) = (0, \frac{\pi}{2}, \frac{\pi}{3}, 0)$ and chose the desirable attitude

$$R_d = \begin{bmatrix} \frac{1}{2} & -\frac{\sqrt{2}}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{2}}{2} & \frac{1}{2} \\ -\frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} \end{bmatrix},$$

which is equivalent to $RPY(\frac{\pi}{4}, \frac{\pi}{4}, 0)$ in the Roll-Pitch-Yaw representation. The algorithm convergence rate $\gamma = 2$. Intuitively, it might be expected that the closeness of the algorithms results in the closeness of their trajectories. Indeed, this is discovered in Fig. 20.2, where we can see that the trajectory $q_{e2}(t)$ of the extended Jacobian algorithm $J_2^{E\#}(q)$ stays closer to the trajectory $q_p(t)$ of the Jacobian pseudo inverse algorithm $J^{P\#}(q)$ than the trajectory $q_{e1}(t)$ of the extended Jacobian algorithm $J_1^{E\#}(q)$. The error values and the joint positions obtained within the simulation time $T = 12$ s are collected in Table 20.1.

Table 20.1 The error value and joint positions

Algorithm	Jacobian pseudo inverse	Extended Jacobian 1	Extended Jacobian 2
$e(T)$	$\begin{bmatrix} -1.1488\,10^{-11} \\ 4.7585\,10^{-12} \\ 3.6144\,10^{-11} \end{bmatrix}$	$\begin{bmatrix} -1.2419\,10^{-11} \\ 5.1445\,10^{-12} \\ 3.9076\,10^{-11} \end{bmatrix}$	$\begin{bmatrix} -1.1632\,10^{-11} \\ 4.8182\,10^{-12} \\ 3.6598\,10^{-11} \end{bmatrix}$
$q(T)$	$\begin{bmatrix} 0.1580 \\ 0.9177 \\ 0.6809 \\ -0.1776 \end{bmatrix}$	$\begin{bmatrix} 0.0969 \\ 0.8614 \\ 0.7194 \\ -0.1041 \end{bmatrix}$	$\begin{bmatrix} 0.1779 \\ 0.9376 \\ 0.6690 \\ -0.2031 \end{bmatrix}$

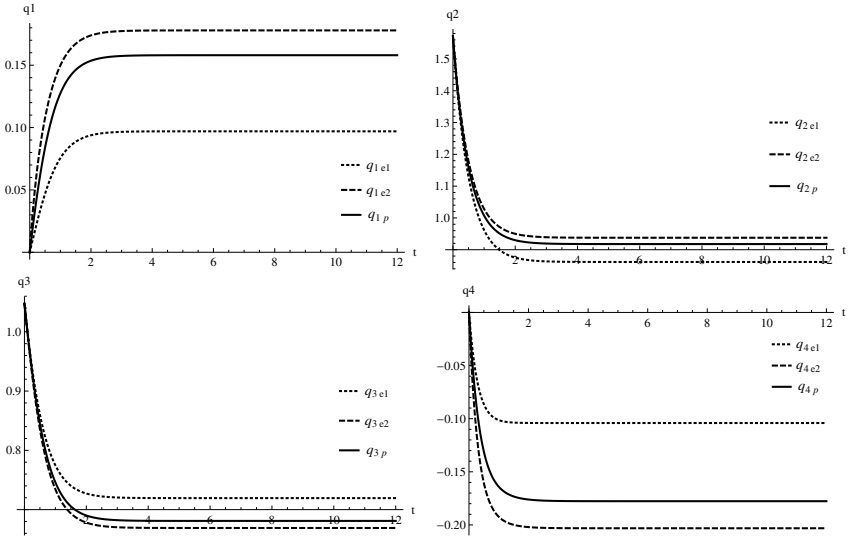


Fig. 20.2 Extended Jacobian vs Jacobian pseudo inverse algorithm: joint space trajectories

In order to check repeatability of the obtained extended Jacobian algorithms, we have commanded the end effector to move between three vertexes $A = RPY(\frac{\pi}{2}, \frac{\pi}{3}, 0)$, $B = RPY(\frac{\pi}{4}, \frac{\pi}{4}, 0)$, $C = RPY(\frac{\pi}{3}, \frac{\pi}{2}, \frac{\pi}{4})$ of a triangle of attitudes. The joint positions produced by the algorithms are collected in Table 20.2.

Table 20.2 Checking repeatability

Algorithm	Jacobian pseudo inverse	Extended Jacobian 1	Extended Jacobian 2
A	$\begin{bmatrix} 0 \\ 1.5708 \\ 1.0472 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1.5708 \\ 1.0472 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1.5708 \\ 1.0472 \\ 0 \end{bmatrix}$
B	$\begin{bmatrix} 0.1580 \\ 0.9177 \\ 0.6809 \\ -0.1776 \end{bmatrix}$	$\begin{bmatrix} 0.0969 \\ 0.8614 \\ 0.7194 \\ -0.1041 \end{bmatrix}$	$\begin{bmatrix} 0.1779 \\ 0.9376 \\ 0.6690 \\ -0.2031 \end{bmatrix}$
C	$\begin{bmatrix} 0.6558 \\ 0.3260 \\ 0.9409 \\ -0.1965 \end{bmatrix}$	$\begin{bmatrix} 0.5227 \\ 0.2999 \\ 1.0676 \\ -0.1480 \end{bmatrix}$	$\begin{bmatrix} 0.5734 \\ 0.3087 \\ 1.0191 \\ -0.1656 \end{bmatrix}$
A	$\begin{bmatrix} 0.1833 \\ 1.6756 \\ 1.0568 \\ -0.2109 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1.5708 \\ 1.0472 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1.5708 \\ 1.0472 \\ 0 \end{bmatrix}$

20.5 Conclusion

Assuming the coordinate-free setting, we have designed an extended Jacobian inverse kinematic algorithm that optimally approximates the Jacobian pseudo inverse algorithm, and applied this algorithm to the manipulator attitude control problem. Performance and convergence of the extended Jacobian algorithm have been assessed by solving the attitude control problem for a redundant robotic wrist. Future research will include an extension of the presented approach to the full manipulator's kinematics defined in $SE(3)$.

Acknowledgements. The work of the first author has been supported by the funds for Polish science in 2010–2012 as a research project. The second author has benefited from a statutory grant provided by the Wrocław University of Technology.

References

1. Arnold, V.I.: Mathematical Methods of Classical Mechanics. Springer, New York (1978)
2. Crouch, P.E.: Spacecraft attitude control and stabilization. IEEE Trans. Autom. Control 29, 321–333 (1984)
3. Wen, J.T., Kreutz-Delgado, K.: The attitude control problem. IEEE Trans. Autom. Control 36, 1148–1161 (1991)
4. Cunha, R., Silvestre, C., Hespanha, J.: Output-feedback control for stabilization on $SE(3)$. Systems & Control Letters 57, 1013–1022 (2008)
5. Jakubiak, J., Tchoń, K., Magiera, W.: Motion planning in velocity affine mechanical systems. International Journal of Control 83, 1965–1974 (2010)
6. Klein, C., Huang, C.: Review of pseudoinverse control for use with kinematically redundant manipulators. IEEE Trans. Syst., Man, Cybernetics 13, 245–250 (1983)
7. Baillieul, J.: Kinematic programming alternatives for redundant manipulators. In: Proc. IEEE Int. Conf. Robot. Automat., St. Louis, vol. 1, pp. 818–823 (1985)
8. Brockett, R.W.: Robotic manipulators and the product of exponentials formula. In: Mathematical Theory of Networks and Systems, pp. 120–129. Springer, Berlin (1984)
9. Shamir, T., Yomdin, Y.: Repeatability of redundant manipulators: mathematical solution of the problem. IEEE Trans. Autom. Contr. 33, 1004–1009 (1988)
10. Roberts, R.G., Maciejewski, A.A.: Nearest optimal repeatable control strategies for kinematically redundant manipulators. IEEE Trans. Robot. Automat. 8, 327–337 (1992)
11. Roberts, R.G., Maciejewski, A.A.: Repetable generalized inverse control strategies for kinematically redundant manipulators. IEEE Trans. Autom. Control 38, 689–699 (1993)
12. Tchoń, K.: Optimal extended Jacobian inverse kinematics algorithms for robotic manipulators. IEEE Trans. on Robotics 28, 1440–1445 (2008)
13. Tchoń, K., Janiak, M.: Repeatable approximation of the Jacobian pseudoinverse. Systems & Control Letters 58, 849–856 (2009)
14. Tchoń, K., Karpińska, J., Janiak, M.: Approximation of Jacobian inverse kinematics algorithms. Int. J. Math., Comput. Sci. 19, 519–531 (2009)
15. Rektorys, K.: Variational Methods in Mathematics, Science and Engineering. Springer, Berlin (1980)
16. Selig, J.M.: Geometric Fundamentals of Robotics. Springer, Berlin (2005)

Chapter 21
Active Disturbance Rejection Control for a Flexible-Joint Manipulator

Marta Kordasz, Rafał Madoński, Mateusz Przybyła, and Piotr Sauer

Abstract. This paper focuses on experimental verification of Active Disturbance Rejection Control (ADRC) on a one-link flexible-joint manipulator. The complexity of the considered system limits the ease of modeling and thus the performance of model-based control methods. ADRC is proposed in this research as an alternative to these techniques. It uses a disturbance observer to actively compensate the effects of perturbations acting on the system. A set of experiments was conducted in order to examine the robustness of the proposed control framework. The results obtained show that the ADRC method managed to stay robust against nonlinear behavior of the flexible-joint system as well as its dynamics parameters variations.

21.1 Introduction

The elasticity in manipulator's joint is usually introduced to ensure a higher safety level for a human operator as well as the machine itself. Due to the flexible properties of the joint it is difficult to achieve satisfying control performance since many of currently used control techniques need a mathematical model of the system [6]. Furthermore, the accuracy of the analytical description of the plant has direct influence on the control quality.

A model of a typical flexible-joint system, however, is difficult to obtain. Nonlinear elements of dynamics, system's time-variance, and the occurrence of vibration modes effectively complicate the capture of its reliable mathematical description.

Development of model-free techniques emerged from the need to control flexible systems without precise knowledge of the process. Control frameworks based on

Marta Kordasz · Rafał Madoński · Mateusz Przybyła · Piotr Sauer
Chair of Control and Systems Engineering, Poznań University of Technology,
ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: {marta.kordasz,rafal.madonski,
mateusz.przybyla}@doctorate.put.poznan.pl,
piotr.sauer@put.poznan.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 247–256.
springerlink.com

genetic algorithms, neural networks, and fuzzy logic are used as solutions to limitations of conventional model-based controllers (examples can be found in [1, 5]). However, the above techniques can be difficult to design (e.g. finding proper membership functions for fuzzy controllers) and are usually nondeterministic.

Active Disturbance Rejection Control (ADRC), originally proposed in [2], can provide significant contribution to the above discussion. It combines the advantages of both model-based and model-free methods. The main idea of the ADRC approach is to consider both internal (e.g. model uncertainties) and external disturbances as an additional state variable of the system. This state variable is estimated by a state observer and canceled out in the control signal. ADRC allows the user to represent a complex plant with a linear and time-invariant model.

This paper focuses on experimental verification of the robustness of ADRC on a manipulator with an elastic joint. The aim of the work is to investigate whether the proposed method is appropriate for controlling flexible-joint manipulators. A Changeable Stiffness Manipulator (CSM) is used in this research as an experimental laboratory testbed. A case study is prepared to analyze the performance of ADRC with different dynamics parameters of the system.

The paper is structured as follows. Section 21.2 provides some key information on the design as well as the mathematical model of the CSM system. Section 21.3 describes the implementation of the control algorithm applied. Study preparation and experimental results are presented in Section 21.4. Future work and some concluding remarks are in Section 21.5.

21.2 System Description

The CSM is a one-degree-of-freedom flexible-joint manipulator with a rigid link. It is an experimental setup built by a research group affiliated with the Chair of Control and System Engineering at the Poznan University of Technology. The construction is based on the system shown in [8]. The CSM can be seen on Fig. 21.1, where q_m represents the link angular position, q_1, q_2 denote the angular positions of the first and second drives, respectively[1], m is the length of the link, and g is the gravitational acceleration.

The transmission belt in the CSM links three pulleys: two of them are connected to DC motors with reducing gears (ratio 1:6.3) and one is connected to the arm rigidly. The elasticity of the actuator is obtained through three tensing structures, consisting of clamps combined with springs, each mounted between two different pulleys. The construction allows the link to rotate in the horizontal plane. A simplified connection diagram of the CSM is presented in Fig. 21.2. Details on the principles of operation can be found in [8]. The CSM can perform the following tasks:

1. changing the link angular position, obtained by simultaneous movement of both motors ($\Delta q_1 = \Delta q_2$);

[1] Signals q_1, q_2, and q_m are available by the use of rotary encoders.

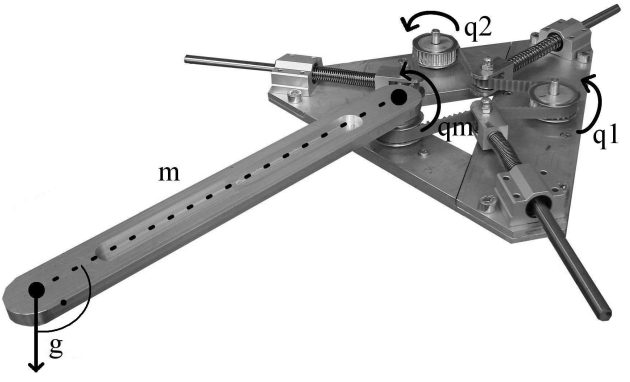


Fig. 21.1 The Changeable Stiffness Manipulator (CSM), an isometric view

2. altering the system stiffness, achieved by changing the relative angular position between the motors ($q_d = \frac{q_1 - q_2}{2}$). Enlargement of q_d results in increasing the system flexibility.

Based on [8], the dynamics of the considered system can be described by the following equations:

$$\begin{cases} I_R \ddot{q}_1 + \beta \dot{q}_1 = \varphi_{1,2} - \varphi_{m,1} + \tau_1, \\ I_R \ddot{q}_2 + \beta \dot{q}_2 = \varphi_{2,m} - \varphi_{1,2} + \tau_2, \\ I_L \ddot{q}_m + \beta \dot{q}_m = \varphi_{m,1} - \varphi_{2,m} - \tau_{ext}, \end{cases} \tag{21.1}$$

where I_R, I_L represent the moments of inertia of the rotor and link, respectively[2], β denotes the friction coefficient, τ_{ext} denotes the external disturbance acting on the link, τ_1 and τ_2 stand for the torques of the rotors, and $\varphi_{m,1}$ represents the torque acting on the pulley attached to the link (q_m) caused by the spring of stiffness coefficient $K_{m,1}$[3]. By adding the first and second parts of the equation given in (21.1), the following form of the dynamics equations is obtained:

$$\begin{cases} I_R \ddot{q}_s + \beta \dot{q}_s + \frac{1}{2}\tau = \tau_s, \\ I_L \ddot{q}_m + \beta \dot{q}_m = \tau - \tau_{ext}, \end{cases} \tag{21.2}$$

where $\tau_s = \frac{\tau_1 + \tau_2}{2}$ is an auxiliary torque of the CSM, $q_s = \frac{q_1 + q_2}{2}$ represents an auxiliary position of rotor, and $\tau = \varphi_{m,1} - \varphi_{2,m}$ is the overall torque acting on the link. Assuming that the stiffness of the system does not change during operation then the following relations are true: $\tau_s = \tau_1 = \tau_2$ and $q_s = q_1 = q_2$. Thus, the system is simplified to a one-rotor-one-link case.

Nevertheless, the nonlinearity of the model is significantly difficult to be measured or captured with a mathematical equation. This characteristic narrows the

[2] The moments of inertia of both rotors are assumed to be equal.

[3] Torques $\varphi_{1,2}$ and $\varphi_{2,m}$ are denoted analogically.

usability of model-based control strategies, since the control law can be overly dependent on the precise analytical representation of the system. That can lead to robustness problems due to possible unpredictable and surprisingly often effects of the real environment phenomena.

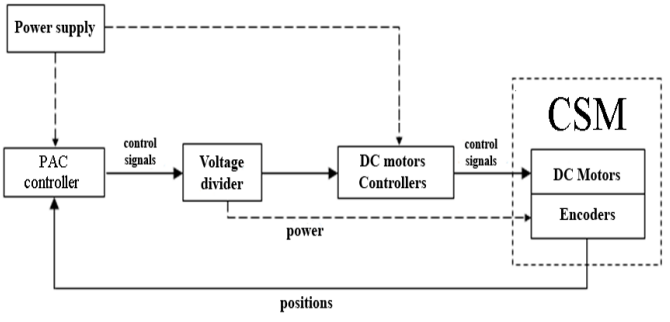


Fig. 21.2 A block diagram of the electrical parts of the CSM system

21.3 Active Disturbance Rejection Control

The ADRC framework allows the user to treat all of the system's uncertainties (both internal and external, referred from now on as *total disturbance*) as an additional state variable. The augmented state is estimated using an Extended State Observer (ESO) and the *total disturbance* estimate is further compensated in real time making the controller inherently robust. In order to introduce the ADRC concept, the CSM system from (2) is rewritten in a following form:

$$\ddot{q}_m = -\frac{2}{I_L}\left(I_R\ddot{q}_s + \beta\dot{q}_s + \frac{1}{2}\beta\dot{q}_m + \frac{1}{2}\tau_{ext}\right) + \frac{2}{I_L}\tau_s, \tag{21.3}$$

which can be further presented as:

$$\ddot{q}_m = f(\ddot{q}_s, \dot{q}_s, \dot{q}_m, \tau_{ext}) + b\tau_s, \tag{21.4}$$

where $f(\cdot)$ describes the *total disturbance* and b is a system parameter, which is generally unknown. The above system can be described using the following state space model:

$$\begin{cases} \dot{x}_1 = x_2, \\ \dot{x}_2 = f(\ddot{q}_s\dot{q}_s, x_1, \tau_{ext}) + b\tau_s, \\ q_m = x_1. \end{cases} \tag{21.5}$$

An augmented state (x_3) is introduced and the plant from (5) is rewritten in the following state space form [2, 4]:

$$\begin{cases} \dot{x}_1 = x_2, \\ \dot{x}_2 = x_3 + b\tau_s, \text{ for } x_3 = f(\cdot), \\ \dot{x}_3 = \dot{f}(\cdot), \\ q_m = x_1. \end{cases} \tag{21.6}$$

A third order ESO (with an assumption made that $\dot{f}(\cdot) = 0$) for the above system is defined as:

$$\begin{cases} \dot{\hat{x}}_1 = \hat{x}_2 - \beta_1 \hat{e}, \\ \dot{\hat{x}}_2 = \hat{x}_3 - \beta_2 \hat{e} + b_0 \tau_s, \\ \dot{\hat{x}}_3 = -\beta_3 \hat{e}, \end{cases} \tag{21.7}$$

where $\hat{x}_1$, $\hat{x}_2$, $\hat{x}_3$ are estimates of the link's angular position, the link's velocity, and the *total disturbance* respectively, β_1, β_2, β_3 are the observer gains, $\hat{e} = q_m - \hat{x}_1$ is the estimation error of state variable x_1, and b_0 describes a constant value, an approximation of b from Equation (4). The control signal in ADRC is defined as:

$$\tau_s = \frac{u_0 - \hat{x}_3}{b_0}, \tag{21.8}$$

where u_0 stands for the output signal from a controller[4]. Assuming that $\hat{x}_3$ estimates $f(\cdot)$ closely (i.e. $\hat{x}_3 \approx f(\cdot)$) and b_0 is chosen as an accurate approximation of b (i.e. $b_0 \approx b$), Equation (4) can be rewritten with the new control signal from (8):

$$\ddot{q}_m = f(\cdot) - b\left(\frac{u_0 - \hat{x}_3}{b_0}\right) \approx u_0. \tag{21.9}$$

The CSM can be now considered as a double integrator system and for this particular plant a simple linear state feedback controller can be chosen:

$$\begin{cases} \dot{x}_1 = x_2, \\ \dot{x}_2 = k_p \hat{e} - k_d \hat{x}_2, \\ q_m = x_1, \end{cases} \tag{21.10}$$

where: $k_p > 0$ – proportional gain, $k_d > 0$ – derivative gain. The ADRC block diagram is presented in Fig. 21.3.

[4] The type of controller in ADRC is optional but should be related to a given control task and meet assumptions of the technological process.

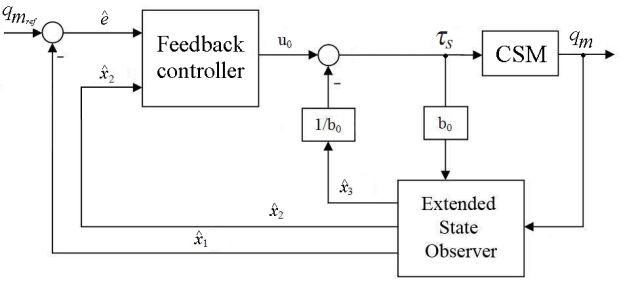


Fig. 21.3 The ADRC design for the CSM system

21.4 Experiments

21.4.1 Study Preparation

Through introducing changes in the dynamics parameters of the system (i.e. by attaching an additional mass $m_{add} = 0.3$ kg and/or by changing the tension of the transmission belt) the robustness of the control system is examined. The proposed case study is divided into three following experiments:

- E1 – parameter tuning and control verification on the pure system ($q_d = 0$ rad, $m_t = m$),
- E2 – verification with additional mass mounted on the link ($q_d = 0$ rad, $m_t = m + m_{add}$),
- E3 – verification with additional mass and enlarged flexibility ($q_d = 0.2$ rad, $m_t = m + m_{add}$),

where m_t is the total mass of the link. Remark: no extra tuning is conducted after experiment E1.

To evaluate the system behavior, a square reference signal of amplitude $A = \{-\frac{\pi}{6}; \frac{\pi}{6}\}$ rad is proposed for all three experiments (E1-E3). The system sampling time is set to $T_s = 0.01$ s.

21.4.2 Tuning Guidelines

Since the angular position as well as the angular velocity of the link are estimated by the ESO, a simple linear PD controller is incorporated in the final implementation. To simplify the tuning process an analytical procedure is used (see [3]) to make $\beta_{1,2,3}$ functions of just one design parameter – the observer's bandwidth ($\omega_0 > 0\,\frac{\text{rad}}{\text{s}}$). In order to tune the ADRC, the following algorithm is proposed[5]:

[5] A similar tuning action was successfully used for different systems, see e.g. [7].

1. Parameters k_p, k_d, and ω_0 are positive and set close to zero. Parameter b_0 is set to one.
2. Parameter ω_0 in the ESO is gradually increased until state variable $\hat{x}_1$ estimates system output q_m closely and state variable $\hat{x}_2$ is nonoscillatory. Remark: the position of the link is changed manually to verify the convergence speed of the estimates.
3. Proportional gain k_p is gradually increased until the output signal reaches the desired behavior.
4. Derivative gain k_d is added if an unacceptable overshoot after step 3 appears and is gradually increased until the output signal reaches the desired behavior. Remark: in many cases, thanks to the particular features of the ADRC concept, no integrating action is needed [4].
5. Parameter b_0 scales the control signal (see Fig. 21.3) so it effects the tracking error ($\hat{e}$) convergence rapidity. Remark: choosing b_0 should be related to the given control task, defined by the system operator.

The most important part of ADRC tuning is obtaining a reliable (i.e. fast-converging) estimation error, thus the ESO parameters are tuned first. Secondly, the PD regulator is tuned to achieve minimum settling time with no signal overshoot. These parameters are chosen empirically and left constant for experiments E1-E3. The obtained parameters are chosen as:

$$k_p = 13,\ k_d = 15,\ b_0 = 3.5,\ \omega_0 = 12,\ \beta_1 = 3\omega_0,\ \beta_2 = 3\omega_0^2,\ \beta_3 = \omega_0^3.$$

21.4.3 Experimental Results

The outcomes of the conducted experiment are plotted in Figs. 21.4 to 21.9. A comparison between experiments E1 and E2 is shown in Figs. 21.4 to 21.6. The greater mass of the system and no retuning procedure after E1 results in more oscillary output signal profile in the transient state. The tracking error in both cases, however, converges with similar settling times and is close to zero.

In Figs. 21.7 to 21.9 a comparison between experiment E1 and E3 is shown. As the mass augmentation did not influence the system noticeably, changes in the transmission belt tension and no retuning process after experiment E1 have important impact on the system behavior. The output signal encounters a significant overshoot but thanks to the controller's disturbance rejection feature, the tracking error converges to a close neighborhood of zero eventually.

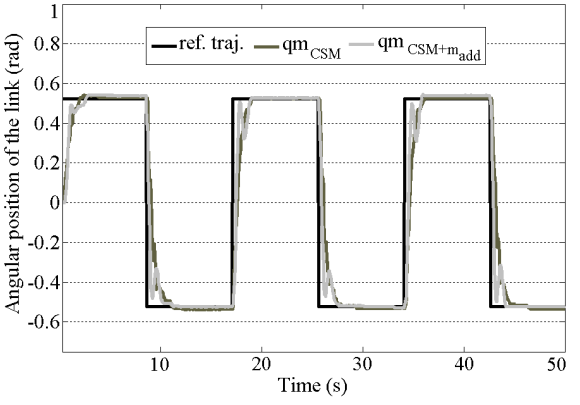


Fig. 21.4 Experiments E1 and E2: angular position of the system with base (subscript CSM) and additional mass (subscript CSM+m_{add})

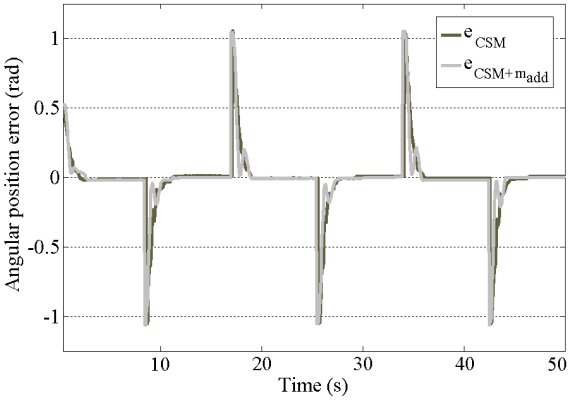


Fig. 21.5 Experiments E1 and E2: position tracking error of the system with base (subscript CSM) and additional mass (subscript CSM+m_{add})

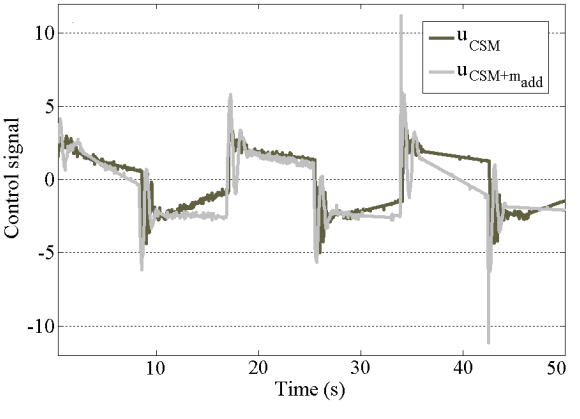


Fig. 21.6 Experiments E1 and E2: control signal of the system with base (subscript CSM) and additional mass (subscript CSM+m_{add})

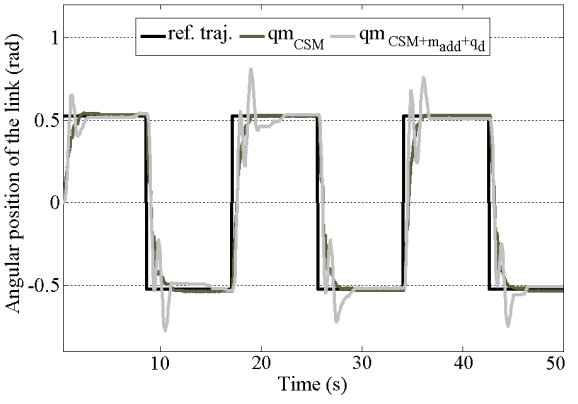


Fig. 21.7 Experiments E1 and E3: angular position of the system with base (subscript CSM) and additional mass as well as with enlarged flexibility (subscript CSM+$m_{add} + q_d$)

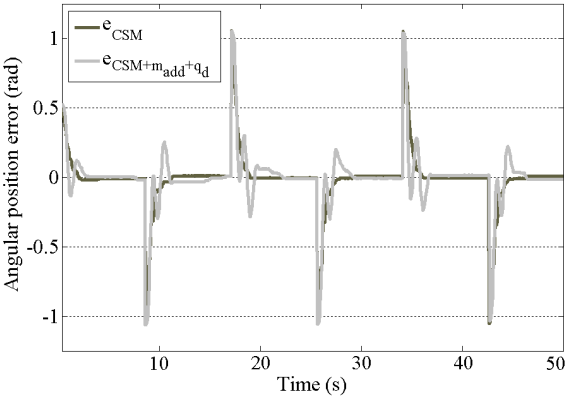


Fig. 21.8 Experiments E1 and E3: position tracking error of the system with base (subscript CSM) and additional mass as well as with enlarged flexibility (subscript CSM+$m_{add} + q_d$)

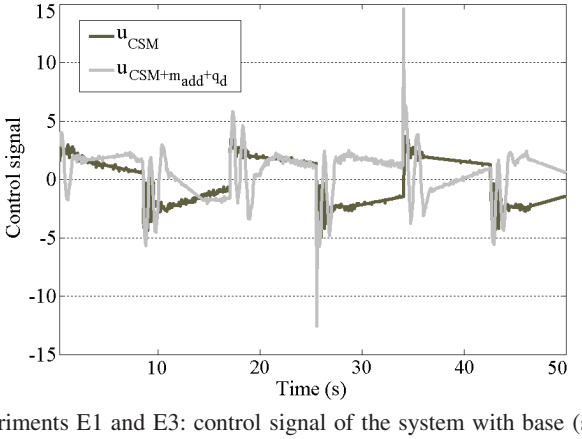


Fig. 21.9 Experiments E1 and E3: control signal of the system with base (subscript CSM) and additional mass as well as with enlarged flexibility (subscript CSM+$m_{add} + q_d$)

21.5 Conclusions and Future Work

This paper presented an Active Disturbance Rejection Control framework implemented on a flexible-joint manipulator. The aim of the work was to verify the robustness of the proposed control scheme against both nonlinear behavior of the plant and changeable dynamics parameters. The ADRC strategy estimated and limited the effects of disturbances acting on the plant. Hence, the CSM could be considered as a simple double integrator system and controlled with a linear PD controller.

For this particular research no information about the dynamics model of the flexible-joint manipulator was used in designing the control law (except the plant's order). The experimental results showed that the ADRC method achieved satisfying robustness and tracking performance (with only one set of tuning parameters for all of the conducted experiments), even with the presence of real environment phenomena and working condition variations (E1-E3).

Future work will be concentrated on verifying ADRC's robustness with ESO estimating higher order disturbances and on comparing ADRC with other model-free control schemes.

Acknowledgements. This work was supported by grant NR13-0028/2011, which was funded by the Polish Ministry of Science and Higher Eduction.

References

1. Siddique, M.N.H., Tokhi, M.O.: GA-based neuro-fuzzy controller for flexible-link manipulator. In: International Conference on Control Applications, pp. 471–476 (2002)
2. Gao, Z., Huang, Y., Han, J.: An alternative paradigm for control system design. In: IEEE Conference on Decision and Control, pp. 4578–4585 (2001)
3. Gao, Z.: Scaling and Bandwidth-Parameterization Based Controller Tuning. In: IEEE American Control Conference, pp. 4989–4995 (2003)
4. Han, J.: From PID to active disturbance rejection control. IEEE Transactions on Industrial Electronics 56(3), 900–906 (2009)
5. Kim, H., Parker, J.K.: Artificial neural network for identification and tracking control of a flexible joint single-link robot. In: SSST Twenty-Fifth Southeastern Symposium on System Theory, pp. 233–237 (1993)
6. Loria, A., Ortega, R.: On tracking control of rigid and flexible joint robots. Applied Mathematics and Computer Science 6(2), 329–341 (1995)
7. Madoński, R., Przybyła, M., Kordasz, M., Herman, P.: Application of Active Disturbance Rejection Control to a reel-to-reel system seen in tire industry. To be Published in Proc. of the IEEE Conference on Automation Science and Engineering (2011)
8. Tonietti, G., Schiavi, R., Bicchi, A.: Design and control of a variable stiffness actuator for safe and fast physical human/robot Interaction. In: IEEE International Conference on Robotics and Automation, pp. 526–531 (2005)

Part IV
Motion Planning and Control of Nonholonomic Systems

Chapter 22
Collision Free Coverage Control with Multiple Agents

Dušan M. Stipanović, Claire J. Tomlin, and Christopher Valicka

Abstract. In this paper we consider a problem of sensing a given area with a group of mobile agents equipped with appropriate sensors. The goal of the agents is to satisfy multiple objectives where one of the objectives is to sense or cover the area, the other one is to avoid collisions with the other vehicles as well as with the static obstacles, and finally to stay close enough during the operation time so that the wireless communication between the agents would be kept reliable. The mathematical formulation of the problem is done by first designing nonnegative functions or functionals which correspond to the objectives and then by constructing control laws based on the objective functions/functionals. To illustrate the procedure we provide an example with three agents with nonhomogeneous sensing capabilities and two obstacles.

22.1 Introduction

Problems related to an accomplishment of multiple objectives by multiple agents are known to be extremely difficult. Very few of these problems have been solved since the problems include scenarios that fit forms of multi-player differential games [28] and multiobjective optimization [18] problems which are still known as open problems in general terms. Furthermore the problem of formulating objectives in mathematical terms adds another level of complexity. In this paper we consider

Dušan M. Stipanović · Christopher Valicka
Department of Industrial and Enterprise Systems Engineering and
Coordinated Science Laboratory, University of Illinois at Urbana-Champaign,
Urbana, IL 61801, USA
e-mail: {dusan,valicka}@illinois.edu

Claire J. Tomlin
Department of Electrical Engineering and Computer Science,
University of California at Berkeley, Berkeley, USA
e-mail: tomlin@eecs.berkeley.edu

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 259–272.
springerlink.com

a particular multi-objective problem of coverage control with guaranteed collision avoidance between the agents as well as with the obstacles. The problem of coverage control can be traced back to the search of an unknown (mobile or static) object which was considered in various yet particular forms in [2, 17, 20] and references reported therein. On the other hand the problem of collision avoidance has been treated in the context of differential games and Hamilton-Jacobi-Bellman-Isaacs partial differential equations (see, for example, [11, 1, 19] and references reported therein).

In this study we follow a coverage control design based on an approach introduced and developed in [9, 10] for agents modeled as simple integrators and where no obstacles were considered. We develop control strategies for agents' dynamics that are nonlinear and affine in control and we also discuss some important issues related to the differentiation under the double integral sign which were not addressed earlier. Our avoidance control approach is based on the concept of avoidance control [14, 13, 15], initially formulated for a noncooperative scenario with two agents, and modified avoidance functions [26] used for avoidance strategies in the case of multiple agents. Furthermore we formulate control laws for avoiding collisions with other agents as well as obstacles. Finally we also impose proximity constraints in order to establish more reliable communication links between the agents yet the proximity objective functions we use are simpler than the ones considered in [30]. This is because in our scenario proximity conditions are considered more as "soft" constraints.

22.2 Avoidance Control for Multiple Agents

To streamline the presentation, we assume that all agents have multiple objectives which are separated into two groups. One group consists of avoidance objectives and the other group of non-avoidance objectives. In this section we will concentrate on the collision avoidance objectives. Furthermore let us assume a group of N agents operating in an environment with N_o obstacles. We also assume that the agents' dynamics are affine in control so that

$$\dot{x}_i = f_i(x_i,u_i) = g_i(x_i)u_i + h_i(x_i),\, x_i(0) = x_{io},\, \forall t \in [0,+\infty),\, \forall i \in \mathbf{N} \tag{22.1}$$

where $\mathbf{N} = \{1,\ldots,N\}$, $x_i \in \mathbb{R}^{n_i}$ is the state, $u_i \in \mathbb{R}^{m_i}$ is the control input/law, and x_{io} is a given initial condition of the i-th agent. The n_i-dimensional vector functions $f_i(\cdot,\cdot)$, $i \in \mathbf{N}$, are assumed to be continuously differentiable with respect to both arguments. The agents' control inputs/laws are assumed to belong to the sets of admissible feedback strategies with respect to the overall state $x = [x_1^T,\ldots,x_N^T]^T \in \mathbb{R}^n$, that is, $u_i \in \mathscr{U}_i = \{\varphi_i(\cdot) : \mathbb{R}^n \to \mathbb{R}^{m_i}\}$ for all $i \in \mathbf{N}$. By admissible set of feedback strategies we assume a set of those strategies which would guarantee unique closed-loop state trajectories for any initial conditions for all agents (for more details we refer to, for example, [4, 5]).

The avoidance objectives include avoiding obstacles as well as collisions between the vehicles. To describe an objective of an agent i to avoid collisions with an agent j we recall the following modified avoidance functions [26]:

$$v_{ij}^a(x) = \left(\min \left\{ 0, \frac{(x_i - x_j)^T P_{ij} (x_i - x_j) - R_{ij}^2}{(x_i - x_j)^T P_{ij} (x_i - x_j) - r_{ij}^2} \right\} \right)^2, \ i, j \in \mathbf{N}, i \neq j \tag{22.2}$$

where $R_{ij} > r_{ij} > 0$ and P_{ij} are positive definite matrices for all $i, j \in \mathbf{N}$. Values of these parameters are chosen according to the appropriate safety regulations and the sensing capabilities of agents. Construction of the modified avoidance functions was motivated by the concept of avoidance control [14]. In order to quantify an objective of an agent $i \in \mathbf{N}$ to avoid static obstacles we propose the following avoidance functions:

$$v_{ij}^a(x) = \left(\min \left\{ 0, \frac{(x_i - x_j^o)^T P_{ij} (x_i - x_j^o) - R_{ij}^2}{(x_i - x_j^o)^T P_{ij} (x_i - x_j^o) - r_{ij}^2} \right\} \right)^2, \ j \in \{N+1, \ldots, N+N_o\} \tag{22.3}$$

where $R_{ij} > r_{ij} > 0$ and P_{ij} are positive definite matrices for all $i \in \mathbf{N}$ and $j \in \{N+1, \ldots, N+N_o\}$. Values are chosen so that the set $\{z : (z - x_j^o)^T P_{ij} (z - x_j^o) \leq r_{ij}^2\}$ *over-bounds* an object j and capital R_{ij} depends on sensing capabilities of an agent i. Thus, one of the design requirements would be to impose that the set $\{z : (z - x_j^o)^T P_{ij} (z - x_j^o) \leq R_{ij}^2\}$ is a subset of the set of states that can be covered or sensed by the i-th agent.

To simplify the presentation we assume that the state vectors x_i, $i \in \mathbf{N}$, represent agents' positions while, in general, only subsets of the state variables represent the positions of the agents. This assumption leads to no loss of generality since the analysis remains the same as pointed out in [27, 26]. The overall avoidance function related to an agent i is given as:

$$v_i^a(x) = \sum_{j \in \mathbf{N}_i} v_{ij}^a(x), \ \forall i \in \mathbf{N} \tag{22.4}$$

where $\mathbf{N}_i \subseteq \{1, \ldots, N+N_o\} \setminus \{i\}$ is a set of indices corresponding to avoidance functions related to the i-th agent. Then, the agents' avoidance control laws may be constructed using gradients of the avoidance functions in (22.4) as [26],

$$u_i^a(x) = -k_i^a g(x_i)^T \left(\frac{\partial v_i^a}{\partial x_i} \right)^T = -k_i^a g(x_i)^T \sum_{j \in \mathbf{N}_i} \left(\frac{\partial v_{ij}^a}{\partial x_i} \right)^T, \ \forall i \in \mathbf{N} \tag{22.5}$$

where k_i^a is a positive avoidance gain for an agent i. The gradients of the avoidance functions v_{ij}^a are given by

$$\frac{\partial v_{ij}^a}{\partial x_i} = \begin{cases} 0, & \text{if } \left\| x_i - \hat{x}_j \right\|_{P_{ij}} \geq R_{ij} \\ 4 \frac{(R_{ij}^2 - r_{ij}^2)((x_i - \hat{x}_j)^T P_{ij} (x_i - \hat{x}_j) - R_{ij}^2)}{((x_i - \hat{x}_j)^T P_{ij} (x_i - \hat{x}_j) - r_{ij}^2)^3} (x_i - \hat{x}_j)^T P_{ij}, & \text{if } R_{ij} > \left\| x_i - \hat{x}_j \right\|_{P_{ij}} > r_{ij} \\ \text{not defined}, & \text{if } \left\| x_i - \hat{x}_j \right\|_{P_{ij}} = r_{ij} \\ 0, & \text{if } \left\| x_i - \hat{x}_j \right\|_{P_{ij}} < r_{ij} \end{cases} \tag{22.6}$$

where $\hat{x}_j$ denotes x_j if j corresponds to an agent or x_j^o if j corresponds to an obstacle. $\|y\|_{P_{ij}} = \sqrt{y^T P_{ij} y}$ where y is a vector and P_{ij} is a positive definite matrix of appropriate dimension. If the gradients of avoidance functions are finite (that is, as long as there are no penetrations of avoidance regions which we consider as collisions) then the avoidance control laws will, most expectedly, be admissible (depending on agents' dynamics one would still have to check this in accordance to some well known conditions as one may find in [4]). The motivation for choosing these gradient control laws comes from the expression of the time derivative of the avoidance functions

$$\frac{dv_i^a}{dt} = \frac{\partial v_i^a}{\partial x_i} g(x_i) u_i + W(x, \bar{u}^i) = -k_i^a \left\| \frac{\partial v_i^a}{\partial x_i} g(x_i) \right\|^2 + W(x, \bar{u}^i) \tag{22.7}$$

where $W(\cdot)$ is a function of the state vector x and a concatenation of all control inputs that appear in dv_i^a/dt except for u_i, which is denoted as $\bar{u}^i$.

The first Lyapunov based method for designing control laws to avoid collisions was named *avoidance control* and reported in [14, 13, 15]. A recent survey on avoidance control results is provided in [25]. In addition, control laws as defined in (22.5) may be arbitrarily large which was a motivation to modify them as proposed in [21] where the avoidance functions have bounded gradients which implies bounded control laws. The efficiency of the avoidance control was already proved by its implementation on testbeds with ground [16] and aerial vehicles [29, 22]. These experiments are very encouraging and show great potential of the avoidance control approach in safe autonomous and semi-autonomous control and coordination of multiple vehicles. In addition, Lyapunov-like analysis was used to append collision avoidance to a number of different objectives. Specific formulations, illustrated with numerous simulations, were provided for coverage control in [9, 10] (without control laws to avoid obstacles and where the dynamics of the agents were simple integrators), formations of nonholonomic vehicles such as unicycles and car-like vehicles in [16, 23], and control of multiple Lagrangian systems [3, 8].

22.3 Coverage Control

In this section we show how to design coverage control laws. Basically the problem is to satisfactorily sense a given domain with a group of agents equipped with appropriate sensors. What we consider as satisfactory will be precisely defined later and is based on the approach proposed and introduced in [9]. One of the issues when using a Lyapunov approach as in [9, 10] is the differentiation under the integral sign (for double integrals) which is also known as Leibniz's integral rule. Since we feel that this issue has not been satisfactorily addressed we plan to provide an overview with some explanations of the process of differentiation of double integrals (subject to a set of differential equations) in the next subsection and then proceed to show how to design coverage control laws for nonlinear systems affine in control.

22.3.1 Differentiation of Double Integrals Depending on a Parameter

In this section we provide some details on differentiation of double integrals that depend on a parameter (which is time in our case) subject to particular constraints which are given as ordinary differential equations. Let us assume that $f(t,\tilde{z}_1,\tilde{z}_2)$ and $\frac{\partial f(t,\tilde{z}_1,\tilde{z}_2)}{\partial t}$ are continuous functions in the domain $\{(t,\tilde{z}_1,\tilde{z}_2) : (t,\tilde{z}_1,\tilde{z}_2) \in [t_1,t_2] \times D(t)\}$. t_1 and t_2 are given constants and $D(t)$ is a compact region in $\mathbb{R}^2$ bounded by $C(t)$ (often denoted as $\partial D(t)$) which is assumed to be a smooth Jordan curve for each t. If these assumptions are satisfied then the time derivative of the following integral:

$$J(t) = \iint_{D(t)} f(t,\tilde{z}_1,\tilde{z}_2)d\tilde{z}_1 d\tilde{z}_2 \tag{22.8}$$

can be expressed as [6, 12]

$$\frac{dJ(t)}{dt} = \frac{d}{dt}\left(\iint_{D(t)} f(t,\tilde{z}_1,\tilde{z}_2)d\tilde{z}_1 d\tilde{z}_2\right) = \iint_{D(t)} \frac{\partial f(t,\tilde{z}_1,\tilde{z}_2)}{\partial t} d\tilde{z}_1 d\tilde{z}_2 + I(t) \tag{22.9}$$

where

$$I(t) = \oint_{C(t)} f(t,\tilde{z}_1,\tilde{z}_2)\left(\frac{\partial \tilde{z}_1}{\partial t}d\tilde{z}_2 - \frac{\partial \tilde{z}_2}{\partial t}d\tilde{z}_1\right) \tag{22.10}$$

and the integration is done in the counterclockwise (that is, positive) direction. Now let us assume that for each t the region $D(t)$ is a circle, that is,

$$\begin{aligned} D(t) &= \{(\tilde{z}_1,\tilde{z}_2) : (\tilde{z}_1 - z_1(t))^2 + (\tilde{z}_2 - z_2(t))^2 \le R^2(t)\}, \\ C(t) &= \{(\tilde{z}_1,\tilde{z}_2) : (\tilde{z}_1 - z_1(t))^2 + (\tilde{z}_2 - z_2(t))^2 = R^2(t)\} \end{aligned} \tag{22.11}$$

where the radius $R(t)$ may be time dependent. Let us also assume that the function $f(\cdot,\cdot,\cdot) : \mathbb{R}^3 \to \mathbb{R}$ has the following special form:

$$f(t,\tilde{z}_1,\tilde{z}_2) = g(t,(\tilde{z}_1 - z_1(t))^2 + (\tilde{z}_2 - z_2(t))^2) \tag{22.12}$$

for some nonnegative and continuously differentiable function $g(\cdot,\cdot) : \mathbb{R} \times \mathbb{R}_+ \to \mathbb{R}_+$, $\mathbb{R}_+ = [0,+\infty)$. These assumptions would correspond to the case of an agent having a circular sensing region where the quality of sensing depends on how far the point $(\tilde{z}_1,\tilde{z}_2) \in \mathbb{R}^2$ is away from $(z_1(t),z_2(t))$ which is the position of the agent at time t. Then equation (22.10) becomes

$$I(t) = \oint_{C(t)} g(t,(\tilde{z}_1 - z_1(t))^2 + (\tilde{z}_2 - z_2(t))^2)\left(\frac{\partial \tilde{z}_1}{\partial t}d\tilde{z}_2 - \frac{\partial \tilde{z}_2}{\partial t}d\tilde{z}_1\right). \tag{22.13}$$

To simplify the expression in equation (22.13) let us introduce a change of variables as

$$\begin{aligned} \tilde{z}_1 &= z_1(t) + R(t)\cos\varphi, \\ \tilde{z}_2 &= z_2(t) + R(t)\sin\varphi \end{aligned} \tag{22.14}$$

which implies

$$g(t,(\tilde{z}_1 - z_1(t))^2 + (\tilde{z}_2 - z_2(t))^2) = g(t,R^2(t)) \text{ on } C(t) \tag{22.15}$$

and [12]

$$\begin{aligned} \frac{\partial \tilde{z}_1}{\partial t} &= \dot{z}_1(t) + \dot{R}(t)\cos\varphi, \\ d\tilde{z}_1 &= -R(t)\sin\varphi d\varphi, \\ \frac{\partial \tilde{z}_2}{\partial t} &= \dot{z}_2(t) + \dot{R}(t)\sin\varphi, \\ d\tilde{z}_2 &= R(t)\cos\varphi d\varphi. \end{aligned} \tag{22.16}$$

Thus,

$$\begin{aligned} I(t) &= \int_0^{2\pi} g(t,R^2(t))\{(\dot{z}_1(t) + \dot{R}(t)\cos\varphi)R(t)\cos\varphi \\ &\quad -(\dot{z}_2(t) + \dot{R}(t)\sin\varphi)(-R(t)\sin\varphi)\}d\varphi \\ &= \dot{R}(t)R(t)g(t,R^2(t))\int_0^{2\pi}(\cos^2\varphi + \sin^2\varphi)d\varphi \\ &\quad +\dot{z}_1(t)R(t)g(t,R^2(t))\int_0^{2\pi}\cos\varphi d\varphi - \dot{z}_2(t)R(t)g(R^2(t))\int_0^{2\pi}\sin\varphi d\varphi \\ &= 2\pi\dot{R}(t)R(t)g(t,R^2(t)) \end{aligned} \tag{22.17}$$

and finally

$$\frac{dJ(t)}{dt} = \iint_{D(t)} \frac{\partial f(t,\tilde{z}_1,\tilde{z}_2)}{\partial t} d\tilde{z}_1 d\tilde{z}_2 + 2\pi\dot{R}(t)R(t)g(t,R^2(t)). \tag{22.18}$$

The expressions in equations (22.10), (22.17) and (22.18) provide few interesting features that should be pointed out. First if the function $f(t,\cdot)$ is equal to zero on $C(t)$ then $I(t) = 0$. If the radius $R(t)$ does not depend on time then we get even stronger result, that is, $I(t) \equiv 0$. Finally if the radius $R(t)$ is decreasing with time then $I(t)$ is negative which is desirable if a designer uses a Lyapunov approach in coverage control where the goal would be to guarantee that dJ/dt is negative if the full coverage is not achieved (like in [9, 10]).

22.3.2 Coverage Error Functional and Control Laws

In this subsection we show how to design coverage control laws based on a coverage error used in [9, 10]. Sensing capabilities of an agent i are assumed to be captured by the following function:

$$S_i(p) = M_i \max\{0, R_i^2 - p\}^2 \Rightarrow S_i'(p) = -2M_i \max\{0, R_i^2 - p\} \tag{22.19}$$

and assuming again that we have N agents, a cumulative sensing functional is given by

$$Q(t,\tilde{x}) = \int_0^t \left(\sum_{i=1}^{N} S_i(\|x_i(\tau) - \tilde{x}\|^2) \right) d\tau \tag{22.20}$$

where $\tilde{x} = [\tilde{x}_1, \tilde{x}_2]^T \in \mathbb{R}^2$.

Let us define $h(w) = (\max\{0,w\})^3$, so that $h'(w) = 3(\max\{0,w\})^2$ and $h''(w) = 6\max\{0,w\}$. The coverage error functional can now be constructed as [9]

$$e(t) = \iint_{\mathscr{D}} h(C^* - Q(t,\tilde{x}))\varphi(\tilde{x}) d\tilde{x}_1 d\tilde{x}_2 \tag{22.21}$$

where $\mathscr{D}$ is a given domain to be covered, C^* is a given positive constant and $\varphi(\tilde{x})$ is a nonnegative scalar function which may be used to encode some prior information about the domain to be covered. Our quantification of the *satisfactory coverage* is expressed by the value of the parameter C^* [9, 10]. Also notice that we refer to $Q(\cdot)$ and $e(\cdot)$ as functionals since they depend on $x_i(\tau)$, $\tau \in [0,t]$, $i \in \mathbf{N}$, yet we denote them as functions of time. We choose this particular abuse of the standard notation since it does not influence mathematical derivations of results yet it significantly simplifies the notation. By abusing the notation in the same way let us define $\hat{e}_t(t)$ as

$$\begin{aligned} \hat{e}_t(t) &= \iint_{\mathscr{D}} \frac{d}{dt} \left(h(C^* - Q(t,\tilde{x}))\varphi(\tilde{x}) \right) d\tilde{x}_1 d\tilde{x}_2 \\ &= -\iint_{\mathscr{D}} h'(C^* - Q(t,\tilde{x})) \left(\sum_{i=1}^{N} S_i(\|x_i(t) - \tilde{x}\|^2) \right) \varphi(\tilde{x}) d\tilde{x}_1 d\tilde{x}_2 \end{aligned} \tag{22.22}$$

and similarly we define $\hat{e}_{tt}(t)$ as

$$\begin{aligned} \hat{e}_{tt}(t) &= -\iint_{\mathscr{D}} \frac{d}{dt} \left(h'(C^* - Q(t,\tilde{x})) \left(\sum_{i=1}^{N} S_i(\|x_i(t) - \tilde{x}\|^2) \right) \varphi(\tilde{x}) \right) d\tilde{x}_1 d\tilde{x}_2 \\ &= \iint_{\mathscr{D}} h''(C^* - Q(t,\tilde{x})) \left(\sum_{i=1}^{N} S_i(\|x_i(t) - \tilde{x}\|^2) \right)^2 \varphi(\tilde{x}) d\tilde{x}_1 d\tilde{x}_2 \\ &\quad -2\sum_{i=1}^{N} \iint_{\mathscr{D}} h'(C^* - Q(t,\tilde{x})) S_i'(\|x_i(t) - \tilde{x}\|^2)(x_i(t) - \tilde{x})^T \dot{x}_i(t) d\tilde{x}_1 d\tilde{x}_2. \end{aligned} \tag{22.23}$$

Equation (22.23) can be rewritten as

$$\hat{e}^i_{tt}(t) = a_0(t) - \sum_{i=1}^{N}(a_{i1}(t)\dot{x}_{i1}(t) + a_{i2}(t)\dot{x}_{i2}(t)) = a_0(t) - \sum_{i=1}^{N} a_i^T(t)\dot{x}_i(t) \tag{22.24}$$

where $\dot{x}_i(t) = [\dot{x}_{i1}(t), \dot{x}_{i2}(t)]^T$, $a_i(t) = [a_{i1}(t), a_{i2}(t)]^T$, and

$$\begin{aligned}
a_0(t) &= \iint_{\mathscr{D}} h''(C^* - Q(t,\tilde{x}))\left(\sum_{i=1}^{N} S_i(\|x_i(t) - \tilde{x}\|^2)\right)^2 \varphi(\tilde{x})d\tilde{x}_1 d\tilde{x}_2, \\
a_{i1}(t) &= 2\iint_{\mathscr{D}} h'(C^* - Q(t,\tilde{x}))S_i'(\|x_i(t) - \tilde{x}\|^2)(x_{i1}(t) - \tilde{x}_1)d\tilde{x}_1 d\tilde{x}_2, \\
a_{i2}(t) &= 2\iint_{\mathscr{D}} h'(C^* - Q(t,\tilde{x}))S_i'(\|x_i(t) - \tilde{x}\|^2)(x_{i2}(t) - \tilde{x}_2)d\tilde{x}_1 d\tilde{x}_2.
\end{aligned} \tag{22.25}$$

Now, if we assume that the agents' dynamics are given by $\dot{x}_i = g_i(x_i)u_i$ we propose to design control laws as

$$u_i(t) = -k_i^c g_i(x_i)^T a_i(t), i \in \mathbf{N} \tag{22.26}$$

where k_i^c is a positive coverage gain for an agent i. These control laws result in

$$\hat{e}_{tt}(t) = a_0(t) + \sum_{i=1}^{N} k_i^c \left\| g_i(x_i)^T a_i(t) \right\|^2. \tag{22.27}$$

Even if the agents' dynamics were $\dot{x}_i = g_i(x_i)u_i + h_i(x_i)$ the control laws would stay the same yet there would be an additional term in equation (22.27). Furthermore notice that the coverage control laws result in closed-loop dynamics being described by functional differential equations [7] and thus if one wants to use Lyapunov stability arguments they have to be formulated accordingly. Another issue is that if one chooses $-\hat{e}_t(t)$ as a Lyapunov candidate (as in [9, 10]) then from the derivations provided in Subsection 22.3.1 we know that its time derivative is not always equal to $-\hat{e}_{tt}(t)$. In some special cases when the line integral denoted as $I(t)$ in Subsection 22.3.1 is either zero or of appropriate sign (as pointed out at the end of the subsection) the meaningful connection may be established.

To make our presentation more concise we assumed that each agent is basing its coverage control law on the full coverage information which is captured in the cumulative coverage sensing functional given in (22.20). In reality this is not always true and in the next section we will introduce a proximity objective in order to impose that the agents try to stay close enough during the operation time so they can exchange the information.

22.4 Proximity Control

Since the agents are mobile, the communication between the agents is done via wireless links. However it is known that the reliability of wireless communication links depends on distances between the agents and thus we impose that the agents stay sufficiently close. This means that the agents move while trying to keep mutual distances to be less than a prescribed value which guarantees reliable links. To quantify a proximity objective for agents i and j to be distance wise closer than $\hat{R}_i$, we use the following (relatively simple) function (similar to the ones used in [10]):

$$v_{ij}^p(x_i,x_j) = \max\{0, \|x_i - x_j\|^2 - \hat{R}_i^2\}^2. \tag{22.28}$$

Now let as assume that the agent i wants to communicate and thus be close to all agents j, $j \in \mathbf{N}_i \subseteq \mathbf{N} \setminus \{i\}$. Then, its overall proximity function may be designed as

$$v_i^p(x) = \sum_{j\in\mathbf{N}_i} v_{ij}^p(x_i,x_j) = \sum_{j\in\mathbf{N}_i} \max\{0, \|x_i - x_j\|^2 - \hat{R}_i^2\}^2. \tag{22.29}$$

Finally we propose the proximity control for the i-th agent as follows:

$$\begin{aligned} u_i^p &= -k_i^p g_i(x_i)^T \left(\frac{\partial v_i^p}{\partial x_i}\right)^T \\ &= -4k_i^p g_i(x_i)^T \sum_{j\in\mathbf{N}_i} \max\{0, \|x_i - x_j\|^2 - \hat{R}_i^2\}(x_i - x_j) \end{aligned} \tag{22.30}$$

where k_i^p is a positive proximity gain.

22.5 Simulation Results

In order to illustrate how the objective control laws given in (22.5), (22.26) and (22.30) perform when applied at the same time we provide an illustrative example in this section. We assume that there are three nonhomogeneous agents and two obstacles. Since the agents' dynamics are affine in control, in the simulations for each agent its overall control law is obtained as the sum of the corresponding objective control laws. Just for the simulation purposes the agents' dynamics are assumed to be simple integrators. Agents are denoted with indices $\{1,2,3\}$ and obstacles with indices $\{4,5\}$. The domain to be covered is a 32×32 square area placed in the positive quadrant with its lower left vertex positioned at the origin. The initial positions for the agents 1, 2, and 3 are at $(12,7)$, $(14,15)$ and $(24,7)$, respectively. All the units are normalized. Obstacles denoted by indices 4 and 5 are of ellipsoidal shapes centered at $(7,12)$ and $(16,25)$, respectively. Positive definite matrices for avoidance functions (given in equations (22.2) and (22.3)) which are not 2×2 identity matrices are given by P_{14}=P_{15}=P_{24}=P_{25}=P_{34}=P_{35}=diag$\{0.5,1\}$. Avoidance and detection regions are spherical regions specified by $r_{1j} = 1$, $R_{1j} = 2$, $r_{2j} = 1$, $R_{2j} = 2$, $r_{3j} = 1$, $R_{3j} = 2$, and the agents' collision avoidance gains are $k_1^a = .015$, $k_2^a = .0015$,

and $k_3^a = .0015$. The two obstacles are defined by their centers and by the shapes of avoidance regions of the agents corresponding to those obstacles. As mentioned in Section 22.2, these avoidance regions either coincide with obstacles' footprints or over-bound them. It is important to note that we consider a scenario where only agents 2 and 3 are in charge of avoiding collisions between the agents. So the first agent only applies avoidance control with respect to the obstacles. The same scenario applies to the implementation of proximity conditions which are to be handled only by agents 2 and 3. The proximity region is defined by $\hat{R}_i = 8$ and the proximity gains are given by $k_2^p = .0001$ and $k_3^p = .0004$.

Coverage sensing capacities are also assumed to be nonhomogeneous and are defined by the agents coverage regions' radii $R_1 = 32\sqrt{2}$, $R_2 = 2$, $R_3 = 2$, and their maximal peak sensing capacities $M_1 = 1/75$, $M_2 = 4$, and $M_3 = 4$. Here we point out that the coverage/sensing region for agent 1 is designed so that the agent can sense the whole domain from any position in the domain (with the smallest peak sensing capability) thus avoiding a need for the global coverage control as proposed and used in [9, 10]. Agents' coverage control gains are $k_1^c = .09$, $k_2^c = .0032$, $k_3^c = .002$, and the coverage constant is $C^* = 40$ while the prior knowledge function $\varphi(\cdot)$ is set to be identically equal to one. It is assumed that a reliable communication link between any two agents exists if they are closer than the proximity distance $\hat{R}_i = 8$. Any pair of agents exchange the coverage information, incorporated in the computation of their coverage control laws via the sensing functional (22.20), if there is a communication path that connects them. By a communication path we mean a series of communication links that connect two agents where the agents are treated as vertices and links are considered as edges of the associated graph [24]. This is done by modifying the coverage sensing functional (22.20) for each agent by including only coverage information from the other agents if there is a communication path between them. Technically this is done by replacing the summation over all agents with a summation over a set of agents that can exchange the information in equations (22.20)-(22.26).

Top view of agents' trajectories as well as positions and shapes of two obstacles are provided in Figure 22.1. Agents' initial positions are depicted as green dots and agents' final positions as red dots. Trajectories for agents 1, 2 and 3 are depicted in magenta, black and blue, respectively. The coverage error functional (22.21) versus time is given in Figure 22.2. A color coded depiction of the quality of coverage with agents' final positions is provided in Figure 22.3 where the satisfactory coverage is quantified by $C^* = 40$ and color coded as light green. Finally pairwise distances between the agents themselves as well as agents and obstacles together with avoidance, detection and proximity levels are given in Figure 22.4. The coverage error functional converged to zero after $T = 5348$[s].

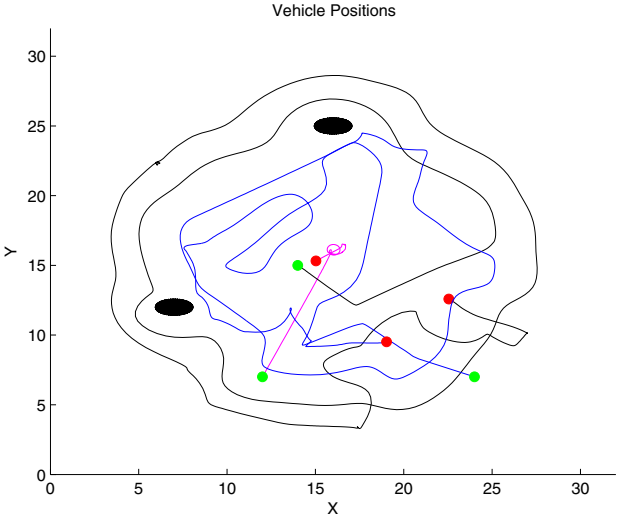


Fig. 22.1 Agents' trajectories

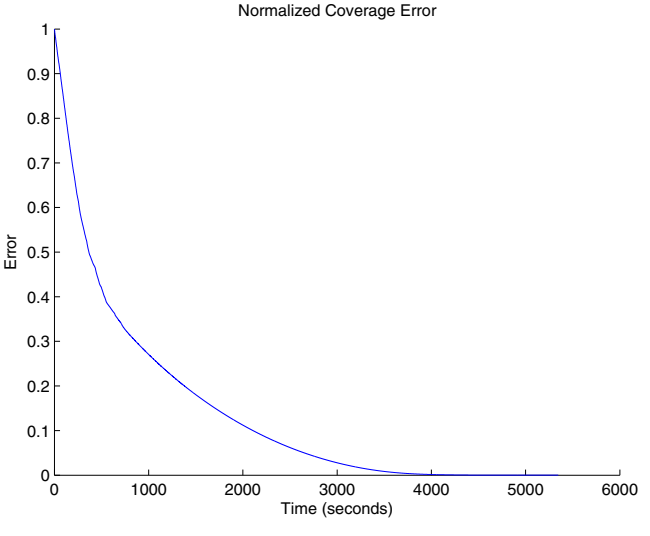


Fig. 22.2 The coverage error functional versus time

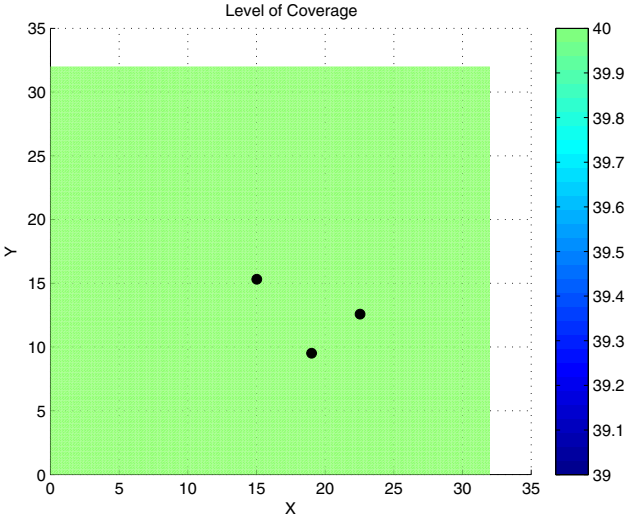


Fig. 22.3 Color coded quality of coverage with agents' final positions

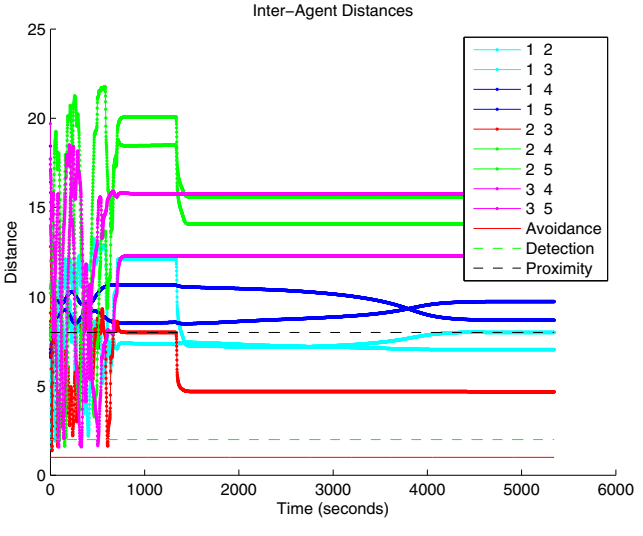


Fig. 22.4 Distances between agents and obstacles

22.6 Conclusions

Particular formulations to design avoidance, coverage and proximity control laws for agents with dynamics which are nonlinear yet affine in control were presented. Some particular issues and challenges in accomplishing these multiple objectives were discussed and pointed out. A numerical example with three nonhomogeneous agents and two static obstacles was presented to illustrate the effectiveness of the proposed control laws.

Acknowledgements. This work was supported in part by the National Science Foundation under grant CMMI 08-25677 and in part by the ONR (HUNT) grant N00014-08-1-0696 and the AFOSR (MURI) grant FA9550-06-1-0312.

References

1. Breakwell, J.V., Hagedorn, P.: Point capture of two evaders in succession. Journal of Optimization Theory and Applications 27, 89–97 (1979)
2. Chernousko, F.L.: Controlled search of movable object. Prikladnia Matematika i Mekhanika 44(1), 3–12 (1980) (in Russian)
3. Chopra, N., Stipanović, D.M., Spong, M.W.: On synchronization and collision avoidance for mechanical systems. In: Proceedings of the 2008 American Control Conference, pp. 3713–3718 (2008)
4. Coddington, E.A., Levinson, N.: Theory of Ordinary Differential Equations. Mc-Graw Hill, New York (1955)
5. Filippov, A.F.: Differential Equations with Discontinuous Righthand Sides. Kluwer Academic Publishers, Dordrecht (1988)
6. Flanders, H.: Differentiation under the integral sign. The American Mathematical Monthly 80(6), 615–627 (1973)
7. Hale, J.K., Lunel, S.M.V.: Introduction to Functional Differential Equations. Springer, New York (1993)
8. Hokayem, P.F., Stipanović, D.M., Spong, M.W.: Semiautonomous control of multiple networked Langrangian systems. International Journal of Robust and Nonlinear Control 19, 2040–2055 (2009)
9. Hussein, I.I., Stipanović, D.M.: Effective coverage control for mobile sensor networks with guaranteed collision avoidance. IEEE Transactions on Control Systems Technology 15(4), 642–657 (2007)
10. Hussein, I.I.: Stipanović, D.M.: Effective coverage control using dynamic sensor networks with flocking and guaranteed collision avoidance. In: Proceedings of the 2007 American Control Conference, pp. 3420–3425 (2007)
11. Isaacs, R.: Differential Games: A Mathematical Theorey with Applications to Warfare and Pursuit, Control and Optimization. John Wiley and Sons, Inc., New York (1965)
12. Lazarević, I.B.: Multivariable Mathematical Analysis, vol. 3. Orion-Art, Belgrade (2005)
13. Leitmann, G.: Guaranteed avoidance strategies. Journal of Optimization Theory and Applications 32, 569–576 (1980)
14. Leitmann, G., Skowronski, J.: Avoidance control. Journal of Optimization Theory and Applications 23, 581–591 (1977)
15. Leitmann, G., Skowronski, J.: A note on avoidance control. Optimal Control Applications & Methods 4, 335–342 (1983)

16. Mastellone, S., Stipanović, D.M., Graunke, C.R., Intlekofer, K.A., Spong, M.W.: Formation control and collision avoidance for multi-agent nonholonomic systems: theory and experiments. International Journal of Robotics Research 13, 107–126 (2008)
17. Melikyan, A.A.: The problem of time-optimal control with the search for a target point. Prikladnia Matematika i Mekhanika 54(1), 1–7 (1990) (in Russisn)
18. Miettinen, K.M.: Nonlinear Multiobjective Optimization. Kluwer Academic Publishers, Norwell (1998)
19. Mitchell, I., Bayen, A.M., Tomlin, C.J.: A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games. IEEE Transactions on Automatic Control 50, 947–957 (2005)
20. Petrosjan, L.A.: Differential Games of Pursuit. Series on Optimization, vol. 2. World Scientific, Singapore (1993)
21. Rodríguez-Seda, E.J., Stipanović, D.M., Spong, M.W.: Collision avoidance control with sensing uncertainties. In: Proceedings of the 2011 American Control Conference (to appear, 2011)
22. Rodríguez-Seda, E.J., Troy, J.J., Erignac, C.A., Murray, P., Stipanović, D.M., Spong, M.W.: Bilateral teleoperation of multiple mobile agents: Formation control and collision avoidance. IEEE Transactions on Control Systems Technology 18, 984–992 (2010)
23. Saska, M., Mejía, J.S., Stipanović, D.M., Schilling, K.: Control and navigation of formations of car-like robots on a receding horizon. In: Proceedings of the 2009 IEEE Multi-Conference on Systems and Control, St Petersburg, Russia (2009)
24. Šiljak, D.D.: Dynamic graphs. Nonlinear Analysis: Hybrid Systems 2, 544–567 (2008)
25. Stipanović, D.M.: A survey and some new results in avoidance control. In: Proceedings of the 15th International Workshop on Dynamics and Control, Tossa de Mar, Spain, pp. 166–173 (2009)
26. Stipanović, D.M., Hokayem, P.F., Spong, M.W., Šiljak, D.D.: Cooperative avoidance control for multi-agent systems. Journal of Dynamic Systems, Measurement, and Control 129, 699–707 (2007)
27. Stipanović, D.M., Sriram, Tomlin, C.J.: Multi-agent avoidance control using an M-matrix property. Electronic Journal of Linear Algebra 12, 64–72 (2005)
28. Vaisbord, E.M., Zhukovskiy, V.I.: Introduction to Multi-Player Differential Games and Their Applications. Gordon and Breach, New York (1988)
29. Valicka, C.G., Bieniawski, S.R., Vian, J.: Stipanović, D.M.: Cooperative avoidance control for UAVs. In: Proceedings of the Tenth International Conference on Control, Automation, Robotics and Vision (ICARCV 2008), Hanoi, Vietnam, pp. 1462–1468 (2008)
30. Zavlanos, M.M., Pappas, G.J.: Potential fields for maintaining connectivity of mobile networks. IEEE Transactions on Robotics 23, 812–816 (2007)

Chapter 23
Manipulating Ergodic Bodies through Gentle Guidance

Leonardo Bobadilla, Katrina Gossman, and Steven M. LaValle

Abstract. This paper proposes methods for achieving basic tasks such as navigation, patrolling, herding, and coverage by exploiting the wild motions of very simple bodies in the environment. Bodies move within regions that are connected by gates that enforce specific rules of passage. Common issues such as dynamical system modeling, precise state estimation, and state feedback are avoided. The method is demonstrated in a series of experiments that manipulate the flow of weasel balls (without the weasels) and Hexbug Nano vibrating bugs.

23.1 Introduction

In everyday life we see many examples of independently moving bodies that are gracefully corralled into behaving in a prescribed way. For example, when the free breakfast area closes in a hotel, the patron usually locks the door from the outside so that no one else can enter, but people eating are able to finish their meals and leave. This has the effect of clearing everyone from the room without people feeling that they have been tightly controlled or coerced. People install a "doggie door" on their house door to enable pets to move in either one direction or both. In a subway system, turnstiles cause people to flow in prescribed directions to ensure that proper fares are paid. The popular Hexbug Nano toy provides a habitat for simple vibrating bugs that can be channeled through rooms and corridors by reconfiguring static gates. All of these scenarios, and many others, involve numerous bodies moving together in one environment with two important principles:

1. Each body moves independently, possibly with a "mind of its own", in a way that seems to exhaustively explore its environment.

Leonardo Bobadilla · Katrina Gossman · Steven M. LaValle
Department of Computer Science, University of Illinois at Urbana-Champaign,
Urbana, IL 61801, USA
e-mail: {bobadil1,kgossma2,lavalle}@uiuc.edu

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 273–282.
springerlink.com

2. The bodies are effectively controlled without precisely measuring their state and without directly actuating them.

We propose a paradigm for developing robotic systems that is inspired by these examples and two common principles. Each body may be a robot, human, animal, or even some simple automaton in the mechanical sense. We want its behavior, however, to seem sufficiently wild, erratic, random, or systematic so that it appears to try every kind of motion. Ideally, we would like the body to move on a trajectory that is *dense* in the boundary of any bounded region in which it is trapped. This means that it will strike every part of the boundary until eventually, there does not exist any open interval that has not contacted. One way to achieve this is through *ergodic* motion[1] [23]. The sufficiently wild trajectory of each body is then exploited to yield effective control strategies that gently guide bodies into desired states, rather than trying to tightly manipulate them. This gives the feeling of herding, corralling, or even encouraging them to behave.

We want to solve tasks such as navigation, patrolling, coverage, herding, and separating into groups. Our approach is to divide the workspace of the bodies into a finite set of *regions*. Bodies are constrained to move within a region, but are able to transition to another region by the use of a *gate*. Sections 23.2, 23.3, and 23.4 introduce, respectively, systems with increasingly sophisticated gates:

- **Static gates:** The gates are fixed in advance and allow one-way motions from region to region.
- **Pliant gates:** The gates have internal modes that affect how bodies are permitted to transition between regions and the modes may passively change via contact with bodies.
- **Controllable gates:** Based on sensor feedback, the gate modes are externally controlled during execution.

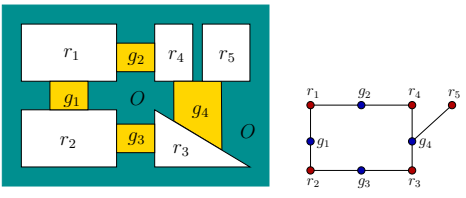
The ideas relate closely to many existing approaches to manipulation. Our work generally falls under the category of *nonprehensile manipulation* [11, 20], which includes tilting a tray to align parts [12], squeezing with a parallel-jaw gripper [15], batting at objects [1], pushing boxes [19], manipulation using passive fences [4, 26], self assembly [16, 21, 25], and inducing a knot in a thread by forcing it through a carefully designed block [3]. Field-based approaches to manipulation also closely fit, including MEMS [6] and vibrating plates [5, 22]. In addition, randomization was shown to help for manipulation in [10], which is similar to the wild motions exploited in our approach.

In addition to manipulation research, our approach has some similarity to *behavior-based robotics* in which teams of agents can achieve complex tasks by composing several simple behaviors [2]; however, in our case we exploit one constant "behavior": The wild, systematic motions of a body. Even more closely related are designing *virtual fences* to control herds of cows [8] and designing fire evacuation strategies to safely "herd" humans out of a burning building [9].

[1] Our notion of ergodicity is derived from dynamical systems analysis, having more to do with dense and uniform coverage of trajectories, rather than its more popular use of probabilistic analysis of Markov chains.

Since our gates have discrete modes and the bodies transition between continuous regions, there are natural connections to hybrid systems and their use in the design and control of multi-robot systems (e.g., [13, 14]).

In math and physics, there is extensive literature on ergodic systems. One of the most relevant branches here is *dynamical billiards* [23], in which conditions are determined under which a ball that "rolls forever" will fill the entire space and achieve other uniformity properties (a famous example is the Bunimovich stadium, which looks like a hockey rink in which the puck ricochets and travels forever).



(a) (b)

Fig. 23.1 a) A simple arrangement of five regions and four gates; b) the corresponding bipartite graph

23.2 Static Gates

Consider a planar workspace $\mathbb{R}^2$ that is partitioned into an obstacle set O and a finite set of bounded cells with connected open interior, each of which is either a *region* or a *gate*; Figure 23.1 shows a simple example. The following conditions are imposed: 1) No region shares a boundary with any other region, 2) No gate shares a boundary with any other gate; 3) Every region shares a boundary with at least one gate; 4) If a gate and a region share a boundary, then the boundary is a connected interval (rather than being a point or being disconnected). Let R denote the set of all regions and G denote the set of all gates. The union of all $r \in R$, all $g \in G$, and O yields $\mathbb{R}^2$.

Now place a *body* b into the workspace. The body is assumed to be "small" with respect to the sizes of regions, gates, and their shared boundaries. It is therefore modeled geometrically as a point even though it may have complicated kinematics and dynamics. For any region $r \in R$, it is assumed that b moves on a trajectory that is dense on the boundary of r. One sufficient condition to achieve this is *ergodicity* (see the appendix for a formal definition and example), which implies that from any initial state, no open subset of its state space, confined to the region r, will be unvisited. We will exploit this property to ensure that b repeatedly strikes every open interval in ∂r (the boundary of r), from various directions.

The precise locations or configurations of b do not need to be measured in our work. The only information that we will use for design and analysis is which region

(a) (b) (c) (d)

Fig. 23.2 a) A weasel ball serves as a wildly behaving body; b) it consists entirely of a battery and slowly oscillating motor mounted to a plastic shell. c) The vibrating Hexbug Nano toy also has wild motion properties and is useful for our experiments; d) it in fact comes equipped with a "habitat" that it nicely explores

or gate contains it. Therefore, we can immediately obtain a combinatorial description of the states and state transitions.

We define a bipartite graph $\mathscr{G}$ with set of vertices $V = R \cup G$. The edge set E corresponds to every region-gate pair that shares a boundary. Due to the constraints on regions and gates, note that $\mathscr{G}$ is bipartite: There are no edges between regions and none between gates.

Each gate has the ability to manipulate the body b, sending it from one of the gate's adjacent regions to another. The simplest gate behavior is assumed here; more sophisticated gates are introduced in later sections. Let each pair (r,g) of adjacent regions and gates be called an *interface*. For each interface, a *direction* is associated: 1) *incoming*, which means that a body approaching the common boundary from r is sent into g; however, it cannot enter r from g. 2) *outgoing*, which means that a body approaching the common boundary from g is sent into r. The gates are considered *static*, meaning that once their directions are set they cannot be changed during execution.

Now we want to use the setup to solve a task. Suppose that an environment is given with regions, gates, and a body. We have the ability to set the direction of every interface to force a desirable behavior. What can be accomplished? In this section, we demonstrate two tasks: 1) Navigation to a specified region and 2) traveling along a cyclic patrolling route without termination.

To develop an implementation, it would be ideal to design bodies that have ergodic behavior. This is an interesting direction for future research; here, we instead borrowed some existing low-cost mechanical systems that have very similar properties. For most experiments, we used a *weasel ball*, pictured in Figure 23.2 (a) and (b). This inexpensive (around $5 US) toy consists of a plastic ball of radius 8.5cm that has only a single offset motor inside that oscillates at about 2Hz. There is no other circuitry: Only a battery and motor. It was designed to roll wildly while appearing to chase a small stuffed weasel toy. After removing the weasel, we discovered in experiments that its exploration properties are remarkably systematic. For other experiments, we used the *Hexbug Nano* (Figure 23.2 (c) and (d)), which is a cheap (around $10 US) vibrating toy that looks like the end of a toothbrush with rubberized bristles with a vibrating motor mounted on top (an earlier YouTube

video demonstrated precisely this). This highly popular toy has been demonstrated to explore complex habitats with regions and gates, which can be purchased. In this case, the gates are static and can be made either *closed* or *bidirectional*.

We place these bodies into planar environments for which obstacles are formed using bricks and cinder blocks. The only part remaining is to design directional gates. A simple way to achieve this is illustrated in Figure 23.3(a). A body moving from the bottom region to the top region can pass through the right side by bending the paper; a body in the other direction is blocked. This simple setup proved to reliably implement the directional gate in numerous experiments. Using these experimental components, we solve the tasks below.

(a) (b)

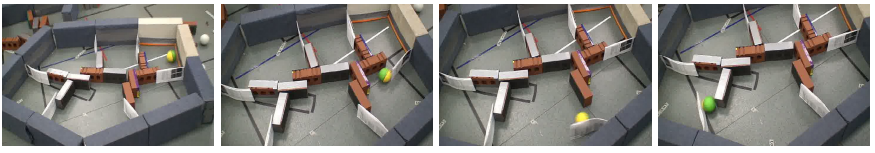
Fig. 23.3 a) A static, directional gate can be implemented making a flexible "door" from a stack of paper; in this case, the body can transition only from the bottom region to the top; b) this works much like a "doggie door"

TASK 1: Navigation to a Specified Region

The task here is to force the body to a particular region, r_{goal}, regardless of where it starts. This can be accomplished by following the classical "funneling" approach whereby in each vertex (region or gate) the body is forced into another vertex that brings it closer to the goal [7, 18, 20]. We simply set all of the interfaces so that all discrete flows in $\mathscr{G}$ lead into r_{goal}. This can be computed in time linear in the size of $\mathscr{G}$ by a simple wavefront propagation algorithm (see Chapter 8 of [17]). Alternatively, Dijkstra's algorithm can be used. These result in a discrete navigation function in which the cost-to-go decreases from vertex to vertex until the goal is reached. The interface edge directions are then set to "point" from the higher-cost vertex to the lower-cost vertex.

We implemented the navigation approach for a weasel ball in an environment of approximately 2 by 3 meters and six gates; see Figure 23.4. The gate directions were set so that the body is led from the region in the upper right to the lower left. It was allowed to achieve this by traveling along the top chain of regions or along the bottom; in this particular run it chose the bottom. The videos for this execution and all others in this paper can be found at:

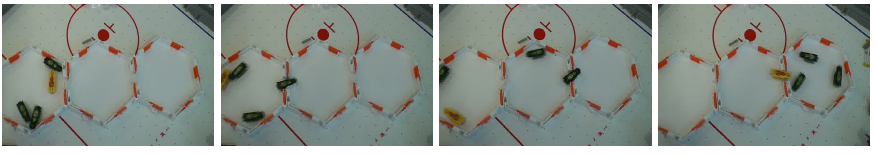
`http://msl.cs.uiuc.edu/ergodic/2010/`



(a) (b) (c) (d)

Fig. 23.4 A basic navigation experiment: a) the weasel ball is placed initially in the upper right corner; b) after 25 seconds it changes regions; c) after 32 it changes again; d) after 45 seconds it reaches its destination

We then tried the method for multiple bodies moving together. This should work provided that the bodies do not interfere with each others' systematic motions and they do not interfere with gate operation. To illustrate the navigation task on another platform, four Hexbug Nanos were placed in an environment with three regions; see Figure 23.5.



(a) (b) (c) (d)

Fig. 23.5 Navigation of Nanos: a) initially, all the four Nanos are together in the left region; b) after 10 seconds one Nano changes regions; c) after 17 seconds one Nano crosses from the second region to the third; d) after 70 seconds all Nanos are in the third region

23.3 Pliant Gates

In Section 23.2, every gate was configured at the outset and remained fixed during execution. In this section, we allow *pliant* gates, which change their behavior while interacting with bodies, thereby having their own internal state. These changes are caused entirely by the motions of bodies, rather then being induced from some external forces. The gates passively configure themselves based on interactions with bodies.

A wide variety of mechanisms can achieve this, leaving many interesting, open design issues. Each gate g has an associated discrete mode space $M(g)$. Based on its mode $m \in M(g)$, transitions between its neighboring regions are allowed or prohibited. In terms of $\mathscr{G}$ from Section 23.2, imagine that the edge directions for all neighboring edges of g are determined as a function of m. Furthermore, a *mode transition equation* is made for each gate: $m' = f(m,r)$, which indicates that the gate enters mode m', when it starts in mode m and a body enters from region r.

TASK 2: Maintaining Fixed Coverage

To illustrate this principle, we designed the gate shown in Figure 23.6. An "L" shaped door is attached to hinges on a vertical post. It can rotate 90 degrees back and forth, depending on forces from the body. The gate has two modes. In one mode, the body is allowed to pass from left to right, but is blocked in the other direction. In the other other mode, it instead permitted from right to left, but blocked from left to right. Furthermore, when a body passes through the gate in either direction, its mode is forced to change.

(a) (b) (c) (d)

Fig. 23.6 a) Initially, two balls are in the right region and three are in the left; the gate mode allows a right to left transition; b) after 18 seconds a body crosses right to left, changing the gate mode; c) after 40 seconds a body moves left to right, changing the gate mode again; d) number of bodies in each region alternates between two and three for the rest of the experiment

For a single body, this has the effect of allowing it to double back on its course. If there are multiple bodies, then the gate keeps the number of bodies per adjacent room roughly constant. Two bodies are not allowed to pass sequentially through the gate; the second one must wait for a body to pass in the other direction. Figure 23.6 shows an experiment that illustrates this for five weasel balls.

23.4 Controllable Gates

In this section, we propose actuating the gates as opposed to passively allowing gate modes to change as in Section 23.3. We now have the ability to set the mode at will during execution; however, a crucial issue arises:

What information will be used during execution to change gate modes?

Let M be the composite mode space, obtained as the $|G|$-fold Cartesian product of $M(g)$ for every $g \in G$. A plan or control law can generally be expressed as a mapping $\pi : \mathcal{I} \rightarrow M$, in which $\mathcal{I}$ is an *information space* that takes into account actuation histories and sensor observation histories (see Chapter 11 of [17]).

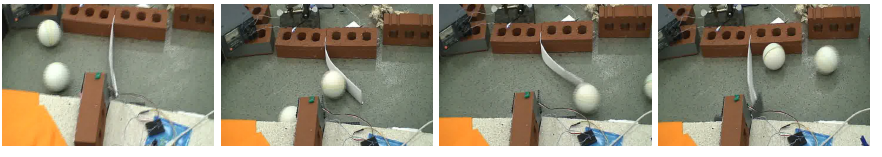
A common approach is to let $\mathcal{I}$ represent the full state space and design powerful sensors and filters to estimate the state (all gate and body configurations) at all times. We instead take a minimalist approach and control the gates using as little sensing information as possible.

One approach is to simply time the switching of the gates. Let $T = [0,t]$ be the execution time interval. We let $\mathscr{I} = T$ and a plan is represented as $\pi : T \rightarrow M$. In this case, the expected behavior of the body or bodies needs to be carefully measured so that the gates are switched at the best times. Without no other feedback, however, this approach is limited.

The shortcomings of time feedback motivate the introduction of simple sensors that can detect whether one or more bodies has crossed part of a region or has traveled through a gate. This allows strategies based on *sensor feedback*. In this case, let Y denote the set of all sensor outputs. A plan is expressed as $\pi : Y \rightarrow M$. More complex plans are possible by reasoning about sensing and actuation *histories*; however, this is left for future work.

We now present a simple experiment with controllable gates; much more is left to do in this direction. Our controllable gate is made from a stack of paper, as the static gates in Section 23.2; however, we additionally have a controllable, "L" shaped part. By rotating the part 90 degrees, the direction of the gate is altered. The *L*-shaped rotating element is made out of Lego pieces attached to a servo motor that is commanded by the an Arduino 8-bit Atmel microcontroller, costing about $30 US; see Figure 23.7.

We developed some simple sensor beams to provide feedback. An emitter-detector pair was placed on each side of the gate, indicating whether a body passes. The detector is a photodiode (about $2 US) and the emitter is a cheap laser pointer (about $3 US).



(a) (b) (c) (d)

Fig. 23.7 Using sensor beam feedback for two bodies: a) Initially, two weasel balls are in left region; b) after 15 seconds one ball transitions; c) after 30 seconds the other body follows; d) the sensor detects both crossings and changes the gate mode

We show an experiment to illustrate sensor feedback with sensor beams. In Figure 23.7, the system was programmed to react to the crossings of two balls: As soon as two balls have crossed, the gate responds by changing directions.

23.5 Conclusion

We developed a manipulation paradigm based on placing wildly moving bodies into a complicated environment and then gently guiding them through gates that can be reconfigured. Several encouraging experiments were shown for weasel balls and Hexbug Nanos performing tasks such as navigation, patrolling, and coverage.

Various types of gates were designed, including static, pliant, and controllable with sensor feedback. With the successes so far, it seems we have barely scratched the surface on the set of possible systems that can be developed in this way to solve interesting tasks.

Other possible experimental platforms come to mind. What other simple mechanisms can be designed to yield effective behavior? Which designs lead to more effective overall systems with regions and gates? Can "virtual" gates be designed on simple robots using sensor feedback, rather than relying on bouncing from walls. What other media are possible? Instead of a planar surface, we can imagine manipulating ergodic boats, underwater vessels, and helicopters. Perhaps even insects can be effectively manipulated. It is also interesting to study asymptotic properties of these systems in various settings to understand the limiting distribution of bodies across the rooms. What limiting distributions can be achieved by simple gate designs? Can the expected running time be calculated for certain combinations of rooms, gates, and body models?

Finally, the experiments appear like a macro-scale version of Maxwell's demon, which was a thought experiment that violates the Second Law of Thermodynamics. Our approach uses controlled gates to interfere with a natural equilibrium that should otherwise result from ergodic particles [24].

Acknowledgements. This work was supported in part by NSF grants 0904501 (IIS Robotics) and 1035345 (Cyberphysical Systems), DARPA SToMP grant HR0011-05-1-0008, and MURI/ONR grant N00014-09-1-1052. The authors thank Noah Cowan for helpful comments.

References

1. Akella, S., Huang, W.H., Lynch, K.M., Mason, M.T.: Sensorless parts feeding with a one joint robot. In: Laumond, J.-P., Overmars, M. (eds.) Algorithms for Robotic Motion and Manipulation, pp. 229–237. A.K. Peters, Wellesley (1997)
2. Balch, T., Arkin, R.C.: Behavior-based formation control for multirobot teams. IEEE Transactions on Robotics and Automation 14(6), 926–939 (1998)
3. Bell, M., Balkcom, D.: Knot tying with single piece fixtures. In: IEEE International Conference on Robotics and Automation, pp. 4429–4436 (2008)
4. Berretty, R.-P., Goldberg, K., Overmars, M.H., van der Stappen, A.F.: On fence design and the complexity of push plans for orienting parts. In: ACM Symposium on Computational Geometry, pp. 21–29 (1997)
5. Böhringer, K.-F., Bhatt, V., Donald, B.R., Goldberg, K.: Algorithms for sensorless manipulation using a vibrating surface. Algorithmica 26, 389–429 (2000)
6. Bohringer, K.F., Donald, B.R., Mihailovich, R., MacDonald, N.C.: Sensorless manipulation using massively parallel microfabricated actuator arrays. In: IEEE International Conference on Robotics and Automation, vol. 1, pp. 826–833 (1994)
7. Burridge, R.R., Rizzi, A.A., Koditschek, D.E.: Sequential composition of dynamically dexterous robot behaviors. International Journal of Robotics Research 18(6), 534–555 (1999)

8. Butler, Z., Corke, P., Peterson, R., Rus, D.: Virtual fences for controlling cows. In: IEEE International Conference on Robotics and Automation, pp. 4429–4436 (2004)
9. Chalmet, L.G., Francis, R.L., Saunders, P.B.: Network models for building evacuation. Fire Technology 18(1), 90–113 (1982)
10. Erdmann, M.A.: Randomization in robot tasks. International Journal of Robotics Research 11(5), 399–436 (1992)
11. Erdmann, M.A.: An exploration of nonprehensile two-palm manipulation using two zebra robots. In: Laumond, J.-P., Overmars, M. (eds.) Algorithms for Robotic Motion and Manipulation, pp. 239–254. A.K. Peters, Wellesley (1997)
12. Erdmann, M.A., Mason, M.T.: An exploration of sensorless manipulation. IEEE Transactions on Robotics & Automation 4(4), 369–379 (1988)
13. Fierro, R., Das, A., Kumar, V., Ostrowski, J.P.: Hybrid control of formations of robots. In: Proceedings IEEE International Conference on Robotics & Automation, pp. 157–162 (2001)
14. Frazzoli, E., Dahleh, M.A., Feron, E.: Robust hybrid control for autonomous vehicles motion planning. Technical Report LIDS-P-2468, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology (1999)
15. Goldberg, K.Y.: Orienting polygonal parts without sensors. Algorithmica 10, 201–225 (1993)
16. Klavins, E.: Toward the control of self-assembling systems. In: Bicchi, A., Christensen, H.I., Prattichizzo, D. (eds.) Control Problems in Robotics, pp. 153–168. Springer, Berlin (2002)
17. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006), http://planning.cs.uiuc.edu/
18. Lozano-Pérez, T., Mason, M.T., Taylor, R.H.: Automatic synthesis of fine-motion strategies for robots. International Journal of Robotics Research 3(1), 3–24 (1984)
19. Lynch, K.M., Mason, M.T.: Stable pushing: Mechanics, controllability, and planning. International Journal of Robotics Research 15(6), 533–556 (1996)
20. Mason, M.T.: Mechanics of Robotic Manipulation. MIT Press, Cambridge (2001)
21. Napp, N., Burden, S., Klavins, E.: The statistical dynamics of programmed self-assembly. In: IEEE International Conference on Robotics and Automation, pp. 1469–1476 (2006)
22. Reznik, D., Moshkoich, E., Canny, J.: Building a universal planar manipulator. In: Bohringer, K.F., Choset, H. (eds.) Distributed Manipulation, pp. 147–171. Kluwer, Norwell (2000)
23. Tabachnikov, S.: Geometry and Billiards. American Mathematical Society, Providence, Rhode Island (2005)
24. Toyabe, S., Sagawa, T., Ueda, M., Muneyuki, E., Sano, M.: Experimental demonstration of information-to-energy conversion and validation of the generalized jarzynski equality. Nat. Phys. 6(12), 988–992 (2010)
25. Whitesides, G.M., Grzybowski, B.: Self-assembly at all scales. Science 295, 2418–2421 (2002)
26. Wiegley, J., Goldberg, K., Peshkin, M., Brokowski, M.: A complete algorithm for designing passive fences to orient parts. In: Proceedings IEEE International Conference on Robotics & Automation, pp. 1133–1139 (1996)

Chapter 24
Practical Efficiency Evaluation of a Nilpotent Approximation for Driftless Nonholonomic Systems

Ignacy Dulęba and Jacek Jagodziński

24.1 Introduction

While planning motion of any system, it is desirable to have a reliable and possibly analytic method to perform the task. Usually the analytic (or almost analytic) methods are not offered for general systems but are available for their special subclasses (flat, nilpotent, in a chain form). Therefore a quite impressive amount of work has been done towards transforming a given system into its easy-to-control equivalent. The transformations are usually local, i.e. valid in an open neighborhood of a given configuration. From a practical point of view, it is important not only to know whether such a local transformation exists but also how large the neighborhood is and what kind of equivalence between the original and the transformed systems is obtained.

In this paper a nilpotent approximation of a driftless nonholonomic system coupled with Lafferriere and Sussmann's (LS) method of motion planning is examined. Nilpotent approximation procedures [1, 8] locally, around a given point in the configuration space, approximate any controllable driftless nonholonomic system with its nilpotent equivalent. Despite being quite complex and computationally involved, the approximation can be pre-computed before solving a motion task. The approximation is a fully deterministic procedure in the sense that no parameters should be assumed or fixed. Lafferriere and Sussmann [4, 5] designed a motion planning method to control nilpotent nonholonomic systems. The algorithm based on the LS method allows a user to implement his own methods to solve some sub-tasks.

In this paper some general measures of accuracy of the nilpotent approximation algorithm are proposed and a practical measure based on a motion in a particular direction in the state space is introduced. The measures are verified for a unicycle robot around selected configurations $\boldsymbol{p}$ in its configuration spaces, where a family

Ignacy Dulęba · Jacek Jagodziński
Institute of Computer Engineering, Control, and Robotics,
Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: {ignacy.duleba,jacek.jagodzinski}@pwr.wroc.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 283–292.
springerlink.com

of motion planning tasks characterized by selecting goals $\boldsymbol{q}_f$ will be solved in a five stage procedure: 1) find a nilpotent approximation of a given system at $\boldsymbol{p}$, 2) move boundary points $\boldsymbol{p},\boldsymbol{q}_f \in \mathbb{Q}$ of each planning task into the space $\mathbb{Z}$ where the transformed original system becomes nilpotent, 3) find controls that solve the motion space in the transformed space, 4) apply those controls to the initial system starting at $\boldsymbol{p}$, 5) check the efficiency of the approximation verifying the distance between the obtained state and $\boldsymbol{q}_f$.

The paper is organized as follows. In Section 24.2 some preliminary Lie algebraic facts are recalled and a sketch of the LS algorithm is provided. In Section 24.3 some measures of efficiency of a nilpotent approximation algorithm are introduced. In Section 24.4 simulation results are presented to verify the measures in practice. Section 24.5 concludes the paper.

24.2 Preliminaries and the LS Algorithm

A driftless nonholonomic system (mobile platforms and robots, free-floating robots, manipulators with nonholonomic ball-gears) is described by the equation

$$\dot{\boldsymbol{q}} = \sum_{i=1}^{m} \boldsymbol{g}_i(\boldsymbol{q})u_i, \quad \dim \boldsymbol{u} = m < n = \dim \boldsymbol{q}, \tag{24.1}$$

where $\boldsymbol{q} = (q_1,\ldots,q_n)^T$ is a configuration, u_i, $i = 1,2,\ldots,m$ are controls, and $\boldsymbol{g}_1,\boldsymbol{g}_2,\ldots,\boldsymbol{g}_m$ are smooth vector fields – generators of system (24.1).

With each generator its formal counterpart (Lie monomial) is associated. From any two Lie monomials $\boldsymbol{X},\boldsymbol{Y}$ a (formal) Lie bracket operation generates another Lie monomial $[\boldsymbol{X},\boldsymbol{Y}]$. Lie monomials satisfy the Jacobi identity and the antisymmetry property [9]. A minimal set of Lie monomials such that any Lie polynomial can be expressed as a linear combination of elements from the set is called a basis. Later on the Ph. Hall basis will be used. To any Lie monomial its degree is assigned: the generators have degree one, while the degree of a compound Lie monomial is the sum of the degrees of its arguments. All Lie monomials sharing the same degree form a layer. Lie monomials when projected into the space of vector fields define a controllability Lie algebra of vector fields. In this space the Lie bracket assigns to any pair of vector fields $\boldsymbol{a},\boldsymbol{b}$ another vector field expressed in coordinates as $[\boldsymbol{a},\boldsymbol{b}] = \frac{\partial \boldsymbol{b}}{\partial \boldsymbol{q}}\boldsymbol{a} - \frac{\partial \boldsymbol{a}}{\partial \boldsymbol{q}}\boldsymbol{b}$. The Lie algebra of vector fields is nilpotent (with order of nilpotency equal to k) if vector fields with degrees higher than k vanish. It is assumed that system (24.1) is small time locally controllable, i.e. the rank of its controllability Lie algebra equals to n at each configuration.

Having introduced the indispensable terminology, we are in position to sketch the algorithms used in this paper. Rather than giving their exhaustive description we concentrate on their ideas leaving out the details.

The first stage of the procedure discussed in Section 24.1 calls for a nilpotent approximation algorithm of system (24.1). The algorithm (taken from [1]) determines a local (around a given configuration $\boldsymbol{p}$) diffeomorphism $\boldsymbol{z} = \varphi_{\boldsymbol{p}}(\boldsymbol{q})$, where $\boldsymbol{z}$ are

privileged coordinates and $\varphi_{\boldsymbol{p}}(\boldsymbol{p}) = \mathbf{0}$. The inverse $\boldsymbol{q} = \varphi_{\boldsymbol{p}}^{-1}(\boldsymbol{z})$ applied to generators $\boldsymbol{g}_i(\boldsymbol{q})$ of system (24.1) produces new generators $\boldsymbol{g}_i^z(\boldsymbol{z})$. Each component of $\boldsymbol{g}_i^z(\boldsymbol{z})$ is expanded into a Taylor series and only terms with appropriate weighted degrees are included into the resulting generators $\boldsymbol{g}_i^{az}(\boldsymbol{z})$. (To point out the difference between a standard and a weighted expansion, let us expand only one exemplary function $\cos(z_1) + \cos(z_3)$ around $\mathbf{0}$ up to terms with degrees 2. When the degrees of coordinates $deg(z_1) = deg(z_3) = 1$ (standard expansion), the result is $2 - (z_1^2 + z_3^2)/2$, but for $deg(z_1) = 1, deg(z_3) = 2$ (weighted expansion), it is equal to $2 - z_1^2/2$. In this way the nilpotent system is obtained

$$\dot{\boldsymbol{z}} = \sum_{i=1}^{m} \boldsymbol{g}_i^{az}(\boldsymbol{z}) u_i = \boldsymbol{G}^{az}(\boldsymbol{z})\boldsymbol{u}, \tag{24.2}$$

where the superscript denotes approximation, expressed in z coordinates. Note that taking a standard Taylor expansion with terms with fixed degrees left, no nilpotent system can be obtained. Using $\boldsymbol{z} = \varphi_{\boldsymbol{p}}(\boldsymbol{q})$, Eq. (24.2) can be transformed into the Q space:

$$\dot{\boldsymbol{q}} = \sum_{i=1}^{m} \boldsymbol{g}_i^{aq}(\boldsymbol{q}) u_i = \boldsymbol{G}^{aq}(\boldsymbol{q})\boldsymbol{u}\,. \tag{24.3}$$

In the second stage, the initial ($\boldsymbol{q} = \boldsymbol{p}$) and final ($\boldsymbol{q} = \boldsymbol{q}_f$) points are transformed via the diffeomorphism $\boldsymbol{z} = \varphi_{\boldsymbol{p}}(\boldsymbol{q})$ into the points $\boldsymbol{z}_0 = \mathbf{0}$ and $\boldsymbol{z}_f$, respectively. In the next stage, controls $\boldsymbol{u}(t) = (u_1(t), \ldots, u_m(t))^T$ that steer system (24.2) from $\boldsymbol{z}_0$ to $\boldsymbol{z}_f$ are searched for. The steps of this motion planning algorithm are presented with some details as they impact the data used in the simulation section. The algorithm is composed of five major steps.

Step 1: A smooth reference trajectory (a straight line is a good choice) $\boldsymbol{\lambda}(\cdot)$ is selected joining $\boldsymbol{z}_0 = \boldsymbol{\lambda}(0)$ with $\boldsymbol{z}_f = \boldsymbol{\lambda}(T)$, where T denotes a prescribed time horizon.

Step 2: Based on the generators of system (24.2), formulate an extended system

$$\dot{\boldsymbol{z}} = \sum_{i=1}^{r} \boldsymbol{g}_i^{az}(\boldsymbol{z}) v_i = \boldsymbol{G}_{ext}^{az}(\boldsymbol{z})\boldsymbol{v}, \tag{24.4}$$

where v_i, $i = 1, 2, \ldots, r$, are extended controls, $\boldsymbol{G}_{ext}^{az} = (\boldsymbol{g}_{gen}^{az}, \boldsymbol{g}_{add}^{az})$ where $\boldsymbol{g}_{gen}^{az} = (\boldsymbol{g}_1^{az}, \ldots, \boldsymbol{g}_m^{az})$ are generators of (24.2), and $\boldsymbol{g}_{add}^{az} = \{\boldsymbol{g}_{m+1}^{az}, \ldots, \boldsymbol{g}_r^{az}\}$ are all non-zero vector fields with degrees higher than one. Because it was assumed that the system is controllable (it satisfies the Lie Algebra Rank Condition), thus $\forall\, \boldsymbol{z}\ \mathrm{rank}(\boldsymbol{G}_{ext}^{az})(\boldsymbol{z}) = n$.

Step 3: Compute extended controls

$$\forall t \in [0, T] \quad \boldsymbol{v}(t) = (\boldsymbol{G}_{ext}^{az})^{\#}(\boldsymbol{\lambda}(t))\frac{d\boldsymbol{\lambda}}{dt}(t), \tag{24.5}$$

where $(\boldsymbol{G}_{ext}^{az})^{\#}$ is the Moore-Penrose matrix inversion.

Step 4: Formulate and solve the Chen-Fliess-Sussmann equation as follows:
a) for system (24.4) define its formal counterpart

$$\dot{\boldsymbol{S}}(t) = \boldsymbol{S}(t)(v_1\boldsymbol{B}_1 + v_2\boldsymbol{B}_2 + \ldots + v_{r-1}\boldsymbol{B}_{r-1} + v_r\boldsymbol{B}_r), \tag{24.6}$$

where $\boldsymbol{B}_i$ are Ph. Hall basis elements (B_i corresponds to g_i^{az}) and $\boldsymbol{S}(t)$ is a designed motion operator initialized with the identity $\boldsymbol{S}(0) = \boldsymbol{I}$,
b) select a motion representation $\boldsymbol{S}(t)$, for example backward

$$\boldsymbol{S}(t) = e^{h_r(t)\boldsymbol{B}_r} e^{h_{r-1}(t)\boldsymbol{B}_{r-1}} \ldots e^{h_2(t)\boldsymbol{B}_2} e^{h_1(t)\boldsymbol{B}_1}, \tag{24.7}$$

or forward one

$$\boldsymbol{S}(t) = e^{h_1(t)\boldsymbol{B}_1} e^{h_2(t)\boldsymbol{B}_2} \ldots e^{h_{r-1}(t)\boldsymbol{B}_{r-1}} e^{h_r(t)\boldsymbol{B}_r}, \tag{24.8}$$

where $h_i(t)$, $i = 1, \ldots, r$, are the functions to be determined and the concatenation of exponents should be read from left to right. The exponential mapping $e : \boldsymbol{B} \to e^{\boldsymbol{B}}$ used in Eqns. (24.7), (24.8) for a given velocity $\boldsymbol{B}$ and a given state $\boldsymbol{q}$, determines the short-term future state $(\boldsymbol{q})e^{\boldsymbol{B}}$. $e^{a(t)\boldsymbol{B}}$ means that $\boldsymbol{B}$ is scaled with the function $a(t)$, depending on controls.
c) for a selected motion representation (24.7) or (24.8), Eq. (24.6) has to be solved (usually expressed as $\boldsymbol{S}^{-1}\dot{\boldsymbol{S}} = \sum_{i=1}^{r} \boldsymbol{B}_i v_i$). Each term $e^{(\cdot)}$ is expanded, using the Taylor formula $e^{\pm h_k \boldsymbol{B}_k} = \sum_{i=0} (\pm 1)^i (h_k^i \boldsymbol{B}_k^i)/i!$, into a finite series, due to nilpotency of the system. $\boldsymbol{S}^{-1}$ inverts consecutive exponents in a given representation with inverted signs, for example $\boldsymbol{S}^{-1} = e^{-h_r(t)\boldsymbol{B}_r} \ldots e^{-h_1(t)\boldsymbol{B}_1}$ for the forward representation. Using the definition of the time derivative $\frac{d}{dt}(e^{\pm h_k \boldsymbol{B}_k}) = (\pm \dot{h}_k \boldsymbol{B}_k) e^{\pm h_k \boldsymbol{B}_k}$ and non-commutative multiplications the equation $\sum_i \alpha_i(\boldsymbol{h}(t), \dot{\boldsymbol{h}}(t))\boldsymbol{B}_i = \boldsymbol{0} = \sum_i (0 \cdot \boldsymbol{B}_i)$ is obtained. To be satisfied for all $\boldsymbol{B}_i$, it has to obey $\alpha_i = 0$ and from this requirement the Chen-Fliess-Sussmann (CFS) equation is obtained in the form

$$\dot{\boldsymbol{h}}(t) = \boldsymbol{M}(\boldsymbol{h}(t))\boldsymbol{v}(t), \tag{24.9}$$

where $\boldsymbol{M}$ is an $(r \times r)$ matrix with polynomial elements. Equation (24.9) is initialized with $\boldsymbol{h}(0) = \boldsymbol{0}$ resulting from $\boldsymbol{S}(0) = \boldsymbol{I}$.
d) for $\boldsymbol{v}(t)$ given in Eq. (24.5), integrate numerically Eq. (24.9) to get $\boldsymbol{h}(T)$ and consequently $\boldsymbol{S}(T)$. For a given representation the process of deriving the CFS equation can be algorithmized [2] and the calculations are to be performed in off-line mode.

Step 5: Generate $\boldsymbol{S}(T)$ with real $(\boldsymbol{u})$ controls, i.e. generate each segment $e^{h_i(T)\boldsymbol{B}_i}$ of the selected motion representation with controls and then concatenate the obtained control pieces. The user may select a method to generate $e^{h_i(T)\boldsymbol{B}_i}$, using continuous or piecewise-constant controls (examples of the controls are given later on).

The controls generated in Step 5, when applied to system (24.3), move it from $\boldsymbol{z}_0$ to $\boldsymbol{z}_f$.

24.3 Measures

The best and mathematically correct method to compare the original system (24.1) with the approximated one (24.2) should be based on a pseudo-norm defined in the sub-Riemannian geometry [6]. For a single system, locally, around a point $\boldsymbol{p}$ the distance to a point $\boldsymbol{q}$ placed in its neighborhood is defined as a minimal energy among those trajectories joining the two points. According to the Ball-Box theorem [6] the distance can be estimated (bounded) by a function of privileged coordinates (multiplied by some $\boldsymbol{p}$-dependent constants) used to approximate the original system with its nilpotent version (the type of the function is $\sum_{i=1}^{n}|z_i|^{1/w_i}$ with appropriately determined non-decreasing sequence of integers w_i). The validity of the estimation is local and restricted to a nonholonomic ball with the radius varying from one point in the configuration space to another. Thus, to check the distance between the two systems it is enough to take as the generators of the system being checked the differences between the generators of system (24.1) and their nilpotent version (24.2) moved back to $\boldsymbol{q}$ coordinates (24.3).

Below two other (simpler) measures to evaluate a "distance" between the original system and its nilpotent approximation will be presented. Both systems should be defined in the same frame as the first one is defined in $\boldsymbol{q}$ coordinates while the second in the privileged coordinates $\boldsymbol{z}$. Consequently, either the vector fields of the original system (24.1) should be moved via the diffeomorphism $\boldsymbol{q}=\varphi_{\boldsymbol{p}}^{-1}(\boldsymbol{z})$ into the $\mathbb{Z}$ space and then compared to the vector fields defined in Eq. (24.2), or those vector fields in (24.2) transformed into $\mathbb{Q}$ via $\boldsymbol{z}=\varphi_{\boldsymbol{p}}(\boldsymbol{q})$. In this way two local measures

$$m_{\mathbb{Q}}(\boldsymbol{p},\boldsymbol{q})=\sum_{i=1}^{m}\|\boldsymbol{g}_i(\boldsymbol{q})-\boldsymbol{g}_i^{aq}(\boldsymbol{q})\|,\quad \boldsymbol{g}_i^{aq}(\boldsymbol{q})=\left(\frac{\partial\varphi_{\boldsymbol{p}}(\boldsymbol{q})}{\partial\boldsymbol{q}}\right)^{-1}\boldsymbol{g}_i^{az}(\varphi_{\boldsymbol{p}}(\boldsymbol{q})),\quad (24.10)$$

valid around $\boldsymbol{p}\in\mathbb{Q}$, and

$$m_{\mathbb{Z}}(\boldsymbol{p},\boldsymbol{z})=\sum_{i=1}^{m}\|\boldsymbol{g}_i^{z}(\boldsymbol{z})-\boldsymbol{g}_i^{az}(\boldsymbol{z})\|,\quad g_i^{z}(\boldsymbol{z})=\left(\frac{\partial\varphi_{\boldsymbol{p}}^{-1}(\boldsymbol{z})}{\partial\boldsymbol{z}}\right)^{-1}\boldsymbol{g}_i(\varphi_{\boldsymbol{p}}^{-1}(\boldsymbol{z})),\quad (24.11)$$

valid around $\varphi_{\boldsymbol{p}}(\boldsymbol{p})=\boldsymbol{0}\in\mathbb{Z}$, can be defined. Any norm can be used in (24.10), (24.11), but weighted (Euclidean) norms should be preferred due to a nice interpretation. The Euclidean weighted norm with a vector of positive weights $\boldsymbol{w}=(w_1,\ldots,w_m)^T$ is defined as follows: $\|a=(a_1,\ldots,a_m)^T\|=\sqrt{\sum_{i=1}^{m}w_ia_i^2}$. For measure (24.10) weights can soften the differences in ranges and units of coordinates $\boldsymbol{q}$. The weighted norm in (24.11) is preferred for quite a different reason. The coordinates of $\boldsymbol{z}$ correspond to motions in consecutive layers of vector fields. The first m coordinates correspond to cheap motions within the first layer. Motions in the first layer are generated by switching on one control u_i and switching off the others. For example, generating a motion $T\boldsymbol{g}_1^{az}$ on a time horizon T requires the amount of energy equal to $\int_0^T 1dt=T$ $(u_1=1)$. For the next $m(m-1)/2$ coordinates (the second layer) the scenario of controls is much more complex. To generate a motion

$T[\boldsymbol{g}_1^{az}, \boldsymbol{g}_2^{az}]$ the Campbell-Baker-Hausdorff-Dynkin fomula can be used [9] and the exemplary scenario of controls is composed of four $\sqrt{T}$-length intervals with controls $u_1 = (+1, 0, -1, 0)^T$, $u_2 = (0, +1, 0, -1)^T$ and the total energy equal to $4\sqrt{T}$. Because the formula is valid locally, T has to be small and $T \ll 4\sqrt{T}$.

Measures (24.10) and (24.11) satisfy the minimal requirement for any well-defined measure as they attain the zero value for originally nilpotent systems and positive for non-nilpotent. In the definition of the measures only generators $\boldsymbol{g}_i$, $i = 1, \ldots, m$ (or their nilpotent equivalents) were involved. The definition can be extended to cover also vector fields from higher layers $\boldsymbol{g}_i$, $i = m+1, \ldots, r$, as they also contribute to the approximation procedure, cf. Eq. (24.4). In this case the upper limit in the sums in (24.10) and (24.11) should be set to r. There is no clear answer which of the measures is better as both have their own advantages and disadvantages. Measure (24.10) evaluates the approximation quality in a natural configuration space with a clear physical interpretation of coordinates while coordinates in $\mathbb{Z}$ used in measure (24.11) better reflect difficulty of motions.

Both measures share one drawback as they do not evaluate a particular direction of motion around a point $\boldsymbol{p} \in \mathbb{Q}$. This feature is especially important in motion planning tasks where a motion in a particular direction is planned. Therefore another measure should be proposed to avoid the drawback. Let a goal point $\boldsymbol{q}_f$ be given and the time of motion is equal to T. Applying the nilpotent approximation procedure at the point $\boldsymbol{p}$, system (24.1) is transformed into the nilpotent one given by Eq. (24.2). Then, with the use of LS algorithm, a motion planning is performed between the boundary points $\boldsymbol{z}_0 = \varphi_{\boldsymbol{p}}(\boldsymbol{p}) = \boldsymbol{0}$ and $\boldsymbol{z}_f = \varphi_{\boldsymbol{p}}(\boldsymbol{q}_f)$ and the resulting controls $\boldsymbol{u}(\cdot)$ are applied to system (24.1) initialized at $\boldsymbol{p}$ to generate configuration $\boldsymbol{q}(T)$. The measure

$$M(\boldsymbol{p}, \boldsymbol{q}_f) = \frac{\|\boldsymbol{q}(T) - \boldsymbol{p}\|}{\|\boldsymbol{q}_f - \boldsymbol{p}\|} \cdot 100\% \tag{24.12}$$

defines how efficient the approximation is in planning the motion towards $\boldsymbol{q}_f$. This measure clearly depends on the selection of representation $\boldsymbol{S}(t)$ in (24.6) and on a method to generate the controls (Step 5). When the value of measure (24.12) is unacceptably high, the goal point $\boldsymbol{q}_f$ should be moved towards $\boldsymbol{p}$. An advantage of the measure is that it evaluates real motions and also tends to zero as $\boldsymbol{q}_f \to \boldsymbol{p}$ as the accuracy of approximation is better and better.

24.4 Simulations

Evaluation of the nilpotent approximation procedure was performed on a model of the unicycle. The robot is described by equations

$$\dot{\boldsymbol{q}} = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \begin{bmatrix} \cos(q_3) \\ \sin(q_3) \\ 0 \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u_2 = \boldsymbol{g}_1(\boldsymbol{q}) u_1 + \boldsymbol{g}_2(\boldsymbol{q}) u_2 \,, \tag{24.13}$$

where (q_1, q_2) denotes its Cartesian position on the XY plane while q_3 its orientation wrt. the OX axis. System (24.13) is controllable as vector fields $\boldsymbol{g}_1, \boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2]$ span the state space everywhere, so $r = 3$. Let $\boldsymbol{p} = (p_1, p_2, p_3)^T \in \mathbb{Q}$ denote the point where the approximation is performed. Using the abbreviation $\boldsymbol{q} - \boldsymbol{p} = \boldsymbol{\Delta} = (\Delta_1, \Delta_2, \Delta_3)^T$ and applying the nilpotent approximation algorithm [1] the forward $\boldsymbol{z} = \varphi_{\boldsymbol{p}}(\boldsymbol{q})$ and inverse $\boldsymbol{q} = \varphi_{\boldsymbol{p}}^{-1}(\boldsymbol{z})$ mapping were obtained:

$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} \Delta_1 \cos(p_3) + \Delta_2 \sin(p_3) \\ \Delta_3 \\ -\Delta_2 \cos(p_3) + \Delta_1 \sin(p_3) \end{bmatrix}, \quad \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} p_1 + z_1 \cos(p_3) + z_3 \sin(p_3) \\ p_2 - z_3 \cos(p_3) + z_1 \sin(p_3) \\ p_3 + z_2 \end{bmatrix}. \tag{24.14}$$

The original system (24.13) moved into the $\mathbb{Z}$ space is described by equations

$$\dot{\boldsymbol{z}} = \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \dot{z}_3 \end{bmatrix} = \begin{bmatrix} \cos(z_2) \\ 0 \\ -sin(z_2) \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} u_2 = \boldsymbol{g}_1^z(\boldsymbol{z}) u_1 + \boldsymbol{g}_2^z(\boldsymbol{z}) u_2. \tag{24.15}$$

The privileged coordinates have degrees $deg(z_1) = deg(z_2) = 1$ and $deg(z_3) = 2$, and the first two coordinates of $\boldsymbol{g}_i^{az}(\boldsymbol{z})$ should have degrees 0 while the third – degree of 1. Thus after expanding $\cos(z_2) = 1 - z_2^2/2 + \ldots$, $-\sin(z_2) = -z_2 + z_2^3/6 \ldots$ only the first terms should be taken into the resulting vector fields. Consequently, the nilpotent approximation of (24.13) is given by the equation

$$\dot{\boldsymbol{z}} = \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \dot{z}_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -z_2 \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} u_2 = \boldsymbol{g}_1^{az}(\boldsymbol{z}) u_1 + \boldsymbol{g}_2^{az}(\boldsymbol{z}) u_2. \tag{24.16}$$

Using the inverse mapping (24.14), the nilpotent approximation system (24.16) was moved back to the $\mathbb{Q}$ space:

$$\dot{\boldsymbol{q}} = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \begin{bmatrix} \cos(p_3) - (q_3 - p_3)\sin(p_3) \\ (q_3 - p_3)\cos(p_3) + \sin(p_3) \\ 0 \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u_2 = \boldsymbol{g}_1^{aq}(\boldsymbol{q}) u_1 + \boldsymbol{g}_2^{aq}(\boldsymbol{q}) u_2. \tag{24.17}$$

Measure (24.10) based on (24.13), (24.17) for the weighted vector $\boldsymbol{w} = (1,1,1)^T$ attained the value of

$$m_{\mathbb{Q}}(\boldsymbol{p}, \boldsymbol{q}) = \sqrt{2 + \Delta_3^2 - 2\cos(\Delta_3) - 2\Delta_3 \sin(\Delta_3)}. \tag{24.18}$$

When trigonometric functions in Eq. (24.18) were expanded into a Taylor series around $\boldsymbol{p}$ (polynomials up to the fourth degree), the measure was equal to

$$m_{\mathbb{Q}} \simeq \Delta_3^2/2. \tag{24.19}$$

Equations (24.18) and (24.19) do not depend on $\boldsymbol{p}$ but on $\boldsymbol{\Delta}$. Similarly, measure (24.11) based on (24.16), (24.15) is equal to $m_{\mathbb{Z}}(\boldsymbol{p}, \boldsymbol{z}) = \sqrt{2 + z_2^2 - 2\cos(z_2) - 2z_2 \sin(z_2)} \simeq \frac{1}{2} z_2^2$ for $\boldsymbol{w} = (1,1,1)^T$. To test measure (24.12)

two motion representations (24.7) and (24.8) were selected. For each representation, the corresponding Chen-Fliess-Sussmann equation described by Eq. (24.9) was calculated [2]:

$$\begin{aligned} e^{h_1\boldsymbol{B}_1}e^{h_2\boldsymbol{B}_2}e^{h_3[\boldsymbol{B}_1,\boldsymbol{B}_2]} &\Rightarrow (\dot{h}_1,\dot{h}_2,\dot{h}_3)=(v_1,v_2,-h_2v_1+v_3) \\ e^{h_3[\boldsymbol{B}_1,\boldsymbol{B}_2]}e^{h_2\boldsymbol{B}_2}e^{h_1\boldsymbol{B}_1} &\Rightarrow (\dot{h}_1,\dot{h}_2,\dot{h}_3)=(v_1,v_2,h_1v_2+v_3) \end{aligned} \tag{24.20}$$

both initialized with $\boldsymbol{h}(0)=\boldsymbol{0}$. Three tasks were considered for initial points varied: Task 1: $\boldsymbol{p}=(5,2,0^\circ)^T$, Task 2: $\boldsymbol{p}=(5,2,45^\circ)^T$, Task 3: $\boldsymbol{p}=(5,2,90^\circ)^T$ (the first two coordinates were fixed as the right-hand side of Eq. (24.13) does not depend on q_1,q_2). Using spherical coordinates a set of goal points $\boldsymbol{q}_f=\boldsymbol{p}+\Delta\boldsymbol{q}$ was generated,

$$\Delta\boldsymbol{q}=(R\cos(\varphi)\cos(\psi),R\cos(\varphi)\sin(\psi),R\sin(\varphi))^T \tag{24.21}$$

with the range of motion $R=\|\Delta\boldsymbol{q}\|$ and spherical angles φ,ψ varied. The time of motion was fixed to $T=1$ and the reference trajectory, used in Step 3, was a straight line segment $\lambda(t)=(t/T)\cdot\boldsymbol{z}_f=(t/T)\cdot\varphi_{\boldsymbol{p}}(\boldsymbol{q}_f)$. To generate the controls the generalized formula (gCBHD) was used [7] (a detailed presentation of the gCBHD formula and a discussion of its impact on motion planning of nonholonomic systems can be found in [3]). After solving Eq. (24.20) with known $\boldsymbol{v}(\cdot)$ (Step 3), the controls to generate motions along $\boldsymbol{B}_1\rightarrow\boldsymbol{g}_1^{az}(\boldsymbol{z}),\boldsymbol{B}_2\rightarrow\boldsymbol{g}_2^{az}(\boldsymbol{z}),\boldsymbol{B}_3\rightarrow[\boldsymbol{g}_1^{az},\boldsymbol{g}_2^{az}](\boldsymbol{z})$ were the following

$$\begin{aligned} e^{h_1(T)\boldsymbol{B}_1} &\Rightarrow u_1(t)=h_1(T)/T',\quad u_2(t)=0, \\ e^{h_2(T)\boldsymbol{B}_2} &\Rightarrow u_1(t)=0,\quad u_2(t)=h_2(T)/T', \\ e^{h_3(T)\boldsymbol{B}_3} &\Rightarrow u_1(t)=\mathrm{sgn}(h_3(T))\xi\cos(2\pi t/T'),\ u_2(t)=\xi\sin(2\pi t/T'), \end{aligned} \tag{24.22}$$

where $\xi=(2/T')\sqrt{\pi|h_3(T)|}$. To preserve the time of completing the motion equal to T, each segment $e^{h_1(T)\boldsymbol{B}_1}$, $e^{h_2(T)\boldsymbol{B}_2}$, $e^{h_3(T)\boldsymbol{B}_3}$ lasts $T'=T/3$.

Figure 24.1 presents the simulation results (the layers – the solid and dashed lines – correspond to a constant value of $M(\boldsymbol{p},\boldsymbol{q}_f)$ marked with the value and expressed in %). It appears that the representation only slightly impacts the characteristics while the goal of motion varies $M(\boldsymbol{p},\boldsymbol{q}_f)$ significantly. The direction of the smallest value of $M(\boldsymbol{p},\boldsymbol{q}_f)$ was attained for $\varphi=0^\circ$, which coincides with the orientation p_3 of the initial point. When φ was increased, $M(\boldsymbol{p},\boldsymbol{q}_f)$ also increased. For a small range motion, $R=0.01$, the accuracy of motion based on the nilpotent approximation procedure was satisfactory. For larger values of R, only motion in selected directions was realized with relatively small errors. For the other two tasks only the forward representation was tested (qualitatively results for the backward representation were similar) and is visualized in Fig. 24.2. All observations made for Task 1 remain valid for the two tasks. In this case only good quality motions are different (but also coincide with p_3 of the initial configurations). In all simulations (cf. Figs. 24.1 and 24.2) it can be noticed that for $\psi\simeq\pm90^\circ$ the value of $M(\boldsymbol{p},\boldsymbol{q}_f)$ is relatively small, because almost pure motion in $[g_1(\boldsymbol{p}),g_2(\boldsymbol{p})]$ was performed. Consequently,

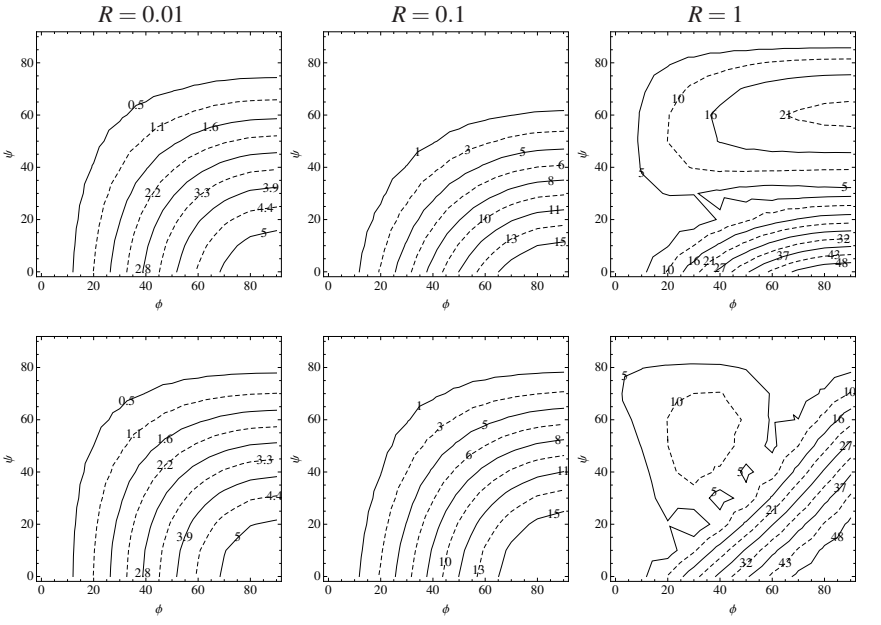


Fig. 24.1 Task 1, measure $M(\boldsymbol{p},\boldsymbol{q}_f)$ for the forward representation (first row), and the backward representation (second row) and a family of goal points described by $\varphi,\psi \in (0,90^\circ)$, changed with a step of 10°

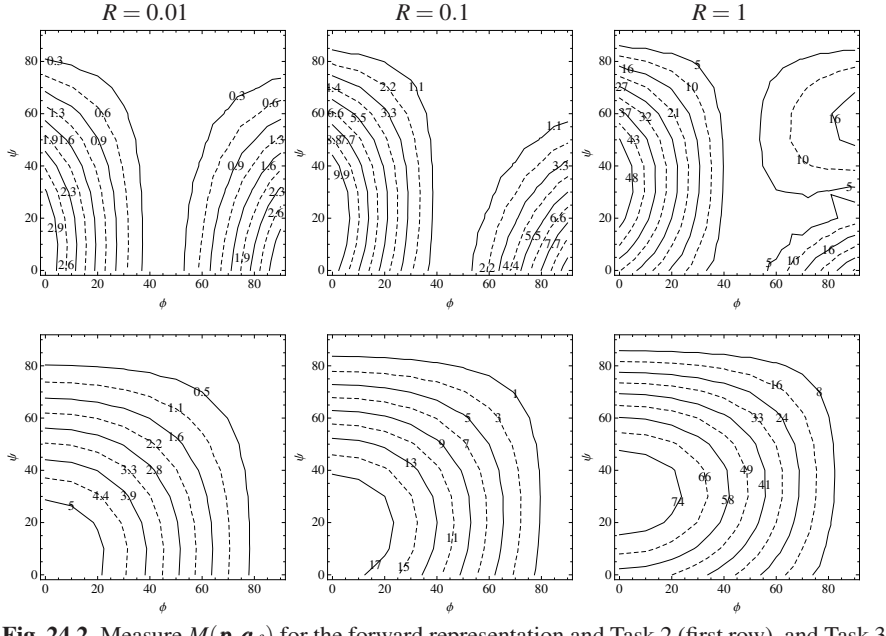


Fig. 24.2 Measure $M(\boldsymbol{p},\boldsymbol{q}_f)$ for the forward representation and Task 2 (first row), and Task 3 (second row) and a family of goal points described by $\varphi,\psi \in (0,90^\circ)$, changed with a step of 10°

the values of $h_1(T)$ and $h_2(T)$ were close to zero and large scale motions along $\boldsymbol{g}_1(\boldsymbol{p}), \boldsymbol{g}_2(\boldsymbol{p})$ did not interrupt the motion in the $[\boldsymbol{g}_1(\boldsymbol{p}), \boldsymbol{g}_2(\boldsymbol{p})]$ direction.

24.5 Conclusions

In this paper a nilpotent approximation algorithm has been evaluated on a simple unicycle robot. Measures of efficiency of the approximation were proposed. Two of them compare the vector fields of the original system and its approximated version. Versions of the measures to cover higher degree vector fields were also mentioned. Although theoretically sound, the measures do not give a hint on real usefulness of the approximation when solving motion planning tasks. To overcome this drawback a third measure has been proposed. The measure evaluates the efficiency of the approximation in a given direction of motion. Using this measure, it has been established that the efficiency of the approximation varies significantly with the designed direction of motion. This measure seems to be better suited to support motion planning algorithms than the two others. Although it depends on some data to be fixed (a representation of motion and a method to generate controls) and some computations need to be performed to calculate the measure, it provides a real evaluation of the approximation algorithm.

References

1. Bellaiche, A., Laumond, J.-P., Chyba, M.: Canonical nilpotent approximation of control systems: application to nonholonomic motion planning. In: IEEE CDC, pp. 2694–2699 (1993)
2. Dulęba, I., Jagodziński, J.: Computational algebra support for the chen-fliess-sussmann differential equation. In: Kozłowski, K.R. (ed.) Robot Motion and Control 2009. LNCIS, vol. 396, pp. 133–142. Springer, Heidelberg (2009)
3. Dulęba, I., Khefifi, W.: Pre-control form of the generalized Campbell-Baker-Hausdorff-Dynkin formula for affine nonholonomic systems. Systems and Control Letters 55(2), 146–157 (2006)
4. Lafferriere, G.: A general strategy for computing steering controls of systems without drift. In: IEEE CDC, pp. 1115–1120 (1991)
5. Lafferriere, G., Sussmann, H.: Motion planning for controllable systems without drift: a preliminary report. Rutgers Center for System and Control. Technical report (1990)
6. Jean, F., Oriolo, G., Vendittelli, M.: A global convergent steering algorithm for regular nonholonomic systems. In: IEEE CDC-ECC, Seville, pp. 7514–7519 (2005)
7. Strichartz, R.S.: The Campbell-Baker-Hausdorff-Dynkin formula and solutions of differential equations. Journal of Functional Analysis 72(2), 320–345 (1987)
8. Vendittelli, M., Oriolo, G., Jean, F., Laumond, J.-P.: Nonhomogeneous nilpotent approximations for nonholonomic system with singularities. IEEE Transactions on Automatic Control 49(2), 261–266 (2004)
9. Wojtyński, W.: Lie groups and algebras. PWN Publisher, Warsaw (1986) (in Polish)

Chapter 25
Obstacle Avoidance and Trajectory Tracking Using Fluid-Based Approach in 2D Space

Paweł Szulczyński, Dariusz Pazderski, and Krzysztof R. Kozłowski

Abstract. In this paper we present an on-line algorithm of motion planning in two-dimensional environment with a moving circular obstacle based on hydrodynamics description. The theoretical background refers to a solution of the Laplace equation using complex algebra. A method of a complex potential design with respect to a dynamic obstacle and a dynamic goal is formally shown. Next, the planning motion problem is extended assuming that the reference trajectory may go through the obstacle. Theoretical considerations are supported by numerical simulations.

25.1 Introduction

The issue of motion planning and control in a constrained space can be regarded one of the fundamental theoretical and practical problems in robotics [5, 10]. Considering existing solutions to motion planning one can first refer to the well-known potential function approach introduced in robotics by Khatib [7]. A formal analysis of this method was carried out by Rimon and Koditschek, who introduced the concept of so-called *navigation function* ensuring one global minimum for star obstacles [14]. The potential function paradigm has proved to be an effective tool for control design also for phase-constrained systems [15, 9].

A complementary method of motion planning based on potential functions may take advantage of *harmonic functions* which solve the Laplace equation. The fundamental advantage of these functions is lack of local minima. The majority of works devoted to this method in robotics refer to a discrete solution of the Laplace equation [3, 1]. Such an approach gives a possibility to describe even complicated environments and to optimize paths with respect to some particular quality indexes

Paweł Szulczyński · Dariusz Pazderski · Krzysztof Kozłowski
Chair of Control and Systems Engineering, Poznań University of Technology,
ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: {pawel.szulczynski,dariusz.pazderski,
krzysztof.kozlowski}@put.poznan.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 293–304.
springerlink.com

and constraints (for example, phase constraints) [11]. The main disadvantage of the discrete methods comes from their high computational complexity demand. As a result, they are usually limited to off-line planning assuming prior knowledge of the static environment structure.

To overcome this limitation, a so-called panel method was proposed in [8]. It can be seen as a modification of the discrete method assuming representation of the environment as a set of primitive segments which are described locally by analytic means.

Another approach is related to motion planning and control defined in pure continuous domain. In [6] Feder and Slotine outlined some possible solutions of a formal description of two-dimensional environments with stationary and non-stationary planar obstacles. This problem was next investigated by Waydo and Murray in [17]. Moreover, techniques based on the Laplace equation can be used for hybrid planners, where classical potential functions and harmonic functions are locally combined [4].

This paper is mainly inspired by the idea presented in [17]. It extends the results shown in [16] and considers tracking the reference trajectory which may go through a non-stationary circular obstacle. The design procedure of static and dynamic potential associated with the goal and the obstacle are given in detail. In order to overcome harmonic potential singularities and to ensure asymptotic convergence of the tracking error in the collision-free area local interaction between the robot and the obstacle is assumed. The performance of the proposed motion-planning algorithm is illustrated by numerical simulation results.

25.2 Overview of Basic Tools for Motion Planning Using Harmonic Functions

The foundations of methods considered in this paper are strictly related to the harmonic function theory. Recalling a formal definition [2], a function $\boldsymbol{\xi} \in C^2(\Omega)$, where Ω is an open subset of $\mathbb{R}^n$, is called a harmonic function on Ω (and this is written as $\boldsymbol{\xi} \in \mathscr{H}(\Omega)$) if it solves the Laplace equation: $\Delta\boldsymbol{\xi} = \mathbf{0}$, where Δ is the Laplace operator.

In motion planning the following properties of harmonic functions can be pointed out as the most important:

- The linearity of the Laplace equation implies that harmonic functions satisfy the principle of superposition. That is, if $\mathscr{H}_1$ and $\mathscr{H}_2$ are harmonic, then any linear combination of them is also harmonic.
- A harmonic function has extremes *only* on the boundary of Ω, so it does not have local maxima (minima) inside Ω. This property can be derived from the maxima (minima) principle [2].

Now we refer to the description of the harmonic functions in two-dimensional space, assuming that $\boldsymbol{p} = [x\, y]^T \in \Omega \supset \mathbb{R}^2$ denotes the Euclidean coordinates of

some point P. It is assumed that $\psi, \varphi : \mathbb{R}^2 \to \mathbb{R}$ describe the *potential functions* satisfying the Cauchy-Riemann equation given by:

$$\frac{\partial \varphi}{\partial x} = \frac{\partial \psi}{\partial y}, \qquad \frac{\partial \varphi}{\partial y} = -\frac{\partial \psi}{\partial x}. \tag{25.1}$$

Eq. (25.1) can be written in the more compact form as follows: $(\nabla \varphi)^T = -\boldsymbol{J}(\nabla \psi)^T$, where ∇ denotes the gradient operator and $\boldsymbol{J} \triangleq \left[\begin{smallmatrix} 0 & -1 \\ 1 & 0 \end{smallmatrix}\right]$ is a skew-symmetric matrix which acts as the rotation operator on the plane along the axis perpendicular to the plane about $\pi/2$ rad. It can be concluded from (25.1) that gradients of these functions describe vector fields in $\mathbb{R}^2$, such that the tangent vectors at some point P, denoted by $(\nabla \varphi)^T|_P$ and $(\nabla \psi)^T|_P$, are orthogonal. Thus, both vector fields are coupled and it is sufficient to consider only one of them. For example, one can assume that vector field $(\nabla \varphi)^T$ describes the velocity of point P, namely

$$\boldsymbol{v} = [v_x \; v_y]^T \triangleq (\nabla \varphi)^T. \tag{25.2}$$

Then, using terminology taken from *hydrodynamics* one can say that ψ is a *streamline* function, while φ is just the *potential* function from which the considered vector field is calculated.

In order to define the potential harmonic function one can conveniently refer to complex analysis. Taking into account the isomorphism between $\mathbb{R}^2$ and $\mathbb{C}$ it is assumed that $\boldsymbol{p} \in \mathbb{R}^2$ can be represented by $\bar{z} \triangleq x + iy \in \mathbb{C}$, where $x = \mathbb{R}\{\bar{z}\}, y = \Im\{\bar{z}\} \in \mathbb{R}$, and $i^2 \triangleq -1$. Consequently, one can define the complex potential by $\bar{w}(\bar{z}) \triangleq \varphi(x,y) + i\psi(x,y) \in \mathbb{C}$ [17]. Taking into account the given potential, the following *complex velocity* associated with it can be considered:

$$\bar{v} = \frac{d\bar{w}}{d\bar{z}} = v_x - iv_y, \tag{25.3}$$

where v_x and $v_y \in \mathbb{R}$ are the components of the vector $\boldsymbol{v}$ defined by Eq. (25.2). Alternatively, the following relationship can be defined:

$$\boldsymbol{v} = \left[\mathbb{R}\left\{\frac{d\bar{w}}{d\bar{z}}\right\} - \Im\left\{\frac{d\bar{w}}{d\bar{z}}\right\}\right]^T. \tag{25.4}$$

Taking into account the design of the potential harmonic function in a 2D environment we refer to hydrodynamics. In this framework we take into account two sources and a circular obstacle. These elements can be regarded the basic tools which give a possibility to define more complicated environments.

The sources $\mathscr{S}$ considered here are modeled by a *uniform flow* or *point sink* with complex potentials $\bar{f}$. The complex potential of a uniform (homogeneous) flow corresponds to the goal placed at infinity and is defined by $\bar{f}(\bar{z}) = u\bar{z}\exp(-i\alpha)$, with $u > 0$ denoting the magnitude of the stream velocity and $\alpha \in \mathbb{S}^1$ being the angle of the stream lines determined with respect to the inertial frame. Assuming

that the velocity of the flow is represented by the vector $\boldsymbol{v}_r = [v_{rx}\ v_{ry}]^T \in \mathbb{R}^2$, one can rewrite the given potential in the following form:

$$\bar{f}(\bar{z}) = (v_{rx} - iv_{ry})\,\bar{z}. \tag{25.5}$$

Next, the potential of an elementary sink (a goal in our case) placed at a point represented by $\bar{z}_r$ is described by

$$\bar{f}(\bar{z}) = -u \log(\bar{z} - \bar{z}_r), \tag{25.6}$$

with u denoting the intensity of the source.

We assume that in the environment a circular obstacle $\mathscr{O}$ with radius r and the center point placed at the origin is present. According to [12], its complex potential can be defined by the potential of a *dipol* as follows:

$$\bar{w}_o = \frac{r^2}{\bar{z}}. \tag{25.7}$$

The resultant complex potential of the circular obstacle - source can be designed based on the following theorem.

Theorem 25.1 (Circle theorem [12]). *Assuming that $\bar{f}(\bar{z})$ denotes the potential of the source $\mathscr{S}$ and $\bar{w}_o$ is the basic potential of the circular obstacle $\mathscr{O}$ then:*

$$\bar{w}(\bar{z}) \triangleq \bar{f}(\bar{z}) + \bar{f}^*(\bar{w}_o(\bar{z})), \tag{25.8}$$

where $(\cdot)^*$ is the operator of complex conjugate, denotes the resultant complex potential such that

$$\forall \boldsymbol{p} \in \partial\mathscr{O} \quad \Im\{\bar{w}(\bar{z})\} = 0. \tag{25.9}$$

From (25.9) it follows that the boundary of the obstacle must contain so-called *zero flow*.

25.3 On-Line Motion Planning Algorithm

We consider an on-line planning (control) task with respect to a *point robot* (i.e. with zero area) moving in the two-dimensional Cartesian workspace $\mathscr{Q} \in \mathbb{R}^2$, where the circular obstacle $\mathscr{O} \in \mathscr{Q}$ with radius $r > 0$ is present. The edge of the obstacle is denoted by $\partial\mathscr{O}$, while the non-colliding (free) space is defined as $\mathscr{Q}_{free} \triangleq \mathscr{Q} \setminus \mathscr{O}$. It is assumed that the configuration of the point robot is described by $\boldsymbol{p} \triangleq [x\ y]^T \in \mathscr{Q}$ and its unconstrained kinematics is given by $\dot{\boldsymbol{p}} = \boldsymbol{v}$, with $\boldsymbol{v}$ denoting the velocity input.

We assume that the obstacle may change its position. Then the coordinates of its center are described by a time-varying function $\boldsymbol{o}(t) \triangleq [o_x(t)\ \ o_y(t)]^T \in \mathbb{R}^2$ such that $\|\dot{\boldsymbol{o}}\| \in \mathscr{L}_\infty$. Unlike in [16], we suppose that the obstacle may change the robot's motion only locally. Accordingly, we introduce the *influence space*, around the obstacle defined as follows:

$$\tilde{\mathscr{O}} \triangleq \left\{ \boldsymbol{p} \in \mathscr{Q}_{free} : \rho\left(\boldsymbol{p}, \partial\mathscr{O}\right) < R \right\}, \tag{25.10}$$

with $\rho\left(\boldsymbol{p}, \partial\mathscr{O}\right) = \inf\left(\|\boldsymbol{p} - \boldsymbol{p}'\|, \boldsymbol{p}' \in \partial\mathscr{O}\right)$ determining the distance to the obstacle boundary with respect to the Euclidean metrics and $R > r$ being the assumed maximum distance of interaction (cf. Fig. 25.1).

It is supposed that the goal position is governed by some reference trajectory $\boldsymbol{p}_r(t) = [x_r(t)\ y_r(t)]^T$ which satisfies the following conditions:

A1: $\boldsymbol{p}_r(t)$ is an at least once differentiable function with respect to time ($\boldsymbol{p}_r(t) \in C^l$, where $l = 1, 2, \ldots$),

A2: $\boldsymbol{p}_r(t)$ will never get stuck at $\tilde{\mathscr{O}} \cup \mathscr{O}$, namely $\lim_{t\to\infty} \boldsymbol{p}_r(t) \notin \tilde{\mathscr{O}} \cup \mathscr{O}$.

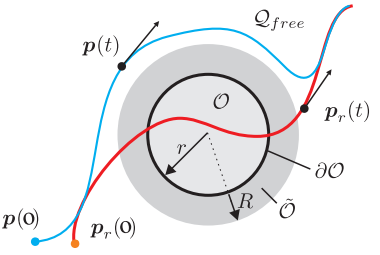


Fig. 25.1 Illustration of motion planning problem

To facilitate the algorithm design let us introduce the tracking error $\boldsymbol{e} \triangleq \boldsymbol{p} - \boldsymbol{p}_r$ and a positive definite function $V \triangleq \frac{1}{2}\boldsymbol{e}^T\boldsymbol{e}$. Then the formal definition of the planning problem can be stated as follows.

Problem 25.1 (On-line path planning). For any reference trajectory $\boldsymbol{p}_r$ satisfying assumptions A1 and A2 find a bounded control input $\boldsymbol{v}$ such that $\forall \boldsymbol{p}(0) \in \tilde{\mathscr{O}}$ the following requirements are satisfied:

- trajectory $\boldsymbol{p}$ does not intersect the obstacle (i.e. the obstacle is avoided): $\forall\ t \geq 0\ \boldsymbol{p}(t) \in \mathscr{Q}_{free}$,
- trajectory $\boldsymbol{p}$ converges to trajectory $\boldsymbol{p}_r$ in collision-free space such that $\forall \boldsymbol{p}, \boldsymbol{p}_r \in \mathscr{Q}_{free} \setminus \tilde{\mathscr{O}}\ \dot{V} \leq -kV$, with $k > 0$, namely the trajectory tracking error in non-colliding space is decreasing,
- set $\tilde{\mathscr{O}}$ is unstable so that $\forall \boldsymbol{p}(t_1) \in \tilde{\mathscr{O}}\ \exists t_2 \in (t_1, \infty)\ \exists T \in (t_1, t_2)\ \forall t \in (t_1, t_2)\ \boldsymbol{p}_r(t) \in \mathscr{Q}_{free} \setminus \tilde{\mathscr{O}}\ \forall t > T\ \boldsymbol{p}(t) \notin \tilde{\mathscr{O}}$, namely trajectories $\boldsymbol{p}$ leave set $\tilde{\mathscr{O}}$ in finite time if $\boldsymbol{p}_r$ remains in $\mathscr{Q}_{free} \setminus \tilde{\mathscr{O}}$.

25.3.1 *Design of Goal and Obstacle Potential in the Case of Non-colliding Trajectory*

Now we recall the result previously investigated in [16], assuming that $\forall t \geq 0\ \boldsymbol{p}_r(t) \in \mathscr{Q}_{free}$. In the static case ($\boldsymbol{p}_r = \boldsymbol{const} \in \mathscr{Q}_{free}$) it is required that $\bar{z}_r$ is

a unique attractor for all trajectories $\bar{z}(t)$ with the initial condition included in the collision-free space, namely $\boldsymbol{p}(0) \in \mathscr{D}_{free}$.

Using (25.6) and (25.8) the following complex potential for the goal and the circular obstacle $\mathscr{O}$ can be designed

$$\bar{w}^{s}(\bar{z}) = \bar{w}^{As}(\bar{z}) + \bar{w}^{Os}(\bar{z}), \tag{25.11}$$

where $\bar{w}^{As}(\bar{z})$ and $\bar{w}^{Os}(\bar{z})$ denote the potentials of the *static attractor* and the *static obstacle*, respectively, defined by

$$\bar{w}^{As} \triangleq -u \log(\bar{z} - \bar{z}_r), \tag{25.12}$$

$$\bar{w}^{Os} \triangleq -u \log\left(\frac{r^2}{\bar{z} - \bar{o}} - \bar{z}_r^* + \bar{o}^*\right). \tag{25.13}$$

Taking into account definition (25.3) from (25.12) and (25.13) one can derive the following complex velocities: $\bar{v}^{s} = \bar{v}^{As} + \bar{v}^{Os}$, with

$$\bar{v}^{As} = \frac{d\bar{w}^{As}(\bar{z})}{d\bar{z}} = -u\frac{\bar{z}^* - \bar{z}_r^*}{|\bar{z} - \bar{z}_r|^2}, \tag{25.14}$$

$$\bar{v}^{Os} = \frac{d\bar{w}^{Os}(\bar{z})}{d\bar{z}} = u\frac{r^2}{(r^2 - (\bar{z}_r^* - \bar{o}^*)(\bar{z} - \bar{o}))(\bar{z} - \bar{o})}. \tag{25.15}$$

Remark 25.1. Considering terms described by (25.14) and (25.15) one should be aware of the following singular points:

- The goal singularity is as follows: $\bar{z} - \bar{z}_r = 0$. This point is the attractor and any trajectory with the initial condition $\boldsymbol{p}(0) \in \mathscr{O}_{free}$ (assuming that no isolated saddle points is reached) converges to it in finite time.
- The obstacle center singularity is as follows: $\bar{z} - \bar{o} = 0$. This point is never reached by the robot by assumption.
- The obstacle internal singularities defined as $r^2 - (\bar{z}_r^* - \bar{o}^*)(\bar{z} - \bar{o}) = 0$. One can prove that this kind of singularity may appear if $|\bar{z} - \bar{o}| \leq r$ ($\boldsymbol{p} \in \mathscr{O}$) or $|\bar{z}_r - \bar{o}_r| \leq r$ ($\boldsymbol{p}_r \in \mathscr{O}$). This means that if the reference point is in the obstacle one can no longer guarantee a unique minimum at $\boldsymbol{p}_r$ (25.12)-(25.13)[1].

Let us assume that the goal is moving so that $\forall t \geq 0\ \boldsymbol{p}_r(t) \in \mathscr{D}_{free}$. This motion can be associated to the homogeneous flow. Taking into account the resultant complex potential in view of (25.5) one has:

$$\bar{w}(\bar{z}) = \bar{v}_r(\bar{z} - \bar{o}) + \bar{v}_r^*\left(\frac{r^2}{\bar{z} - \bar{o}}\right), \tag{25.16}$$

with $\bar{v}_r$ determining the complex velocity of the goal.

[1] The interior of the obstacle can be interpreted as the forbidden set in which the harmonic functions do not satisfy the desired properties for motion planning.

Next, we extend our consideration to the case of a nonstationary (dynamic) obstacle, assuming that $\bar{v}_o$ is the complex velocity of its origin (cf. [17]). Taking into account that in the local frame fixed to the obstacle its boundary is invariant with respect to obstacle motion one can modify potential (25.16) as follows:

$$\bar{w}'(\bar{z}) = \bar{w}(\bar{z}) - \bar{v}_o(\bar{z}-\bar{o}) - \bar{v}_o^*\left(\frac{r^2}{\bar{z}-\bar{o}}\right), \tag{25.17}$$

where $\bar{w}'(\bar{z})$ determines the dynamic complex potential in the local frame. Accordingly, in the inertial frame the potential becomes:

$$\bar{w}^d(\bar{z}) = \bar{w}^{Ad}(\bar{z}) + \bar{w}^{Od}(\bar{z}), \tag{25.18}$$

where $\bar{w}^{Ad}(\bar{z})$ and $\bar{w}^{Os}(\bar{z})$ denote the potentials of the *dynamic attractor* and the *dynamic obstacle*, respectively, defined by

$$\bar{w}^{Ad} \triangleq \bar{v}_r(\bar{z}-\bar{o}) \tag{25.19}$$

and

$$\bar{w}^{Od} \triangleq (\bar{v}_r^* - \bar{v}_o^*)\frac{r^2}{\bar{z}-\bar{o}}. \tag{25.20}$$

From (25.19) and (25.20) one can derive the following complex velocities: $\bar{v}^d = \bar{v}^{Ad} + \bar{v}^{Od}$, where

$$\bar{v}^{Ad} = \frac{d\bar{w}^{Ad}(\bar{z})}{d\bar{z}} = \bar{v}_r, \tag{25.21}$$

$$\bar{v}^{Od} = \frac{d\bar{w}^{Od}(\bar{z})}{d\bar{z}} = -\frac{r^2(\bar{v}_r^* - \bar{v}_o^*)}{(\bar{z}-\bar{o})^2}. \tag{25.22}$$

Referring to (25.21) and (25.22) it is clear that the only singular point is present at $\bar{z} = \bar{o}$, namely at the center of the obstacle. However, since $\boldsymbol{p} \notin \mathscr{O}$, this obstruction is not relevant.

The static and dynamic cases can be considered simultaneously by using a superposition of potentials $\bar{w}^s$ and $\bar{w}^d$. Taking into account that the potentials are harmonic functions, their superposition still guarantees that there is a unique attractor $\bar{z} = \bar{z}_r$ (this is one of the reasons why the reference trajectory should satisfy the non-colliding condition). The vector field associated to the considered potential is given by

$$\bar{v} = \bar{v}^{As} + \bar{v}^{Ad} + \bar{v}^{Os} + \bar{v}^{Od}. \tag{25.23}$$

In order to overcome the singularity at $\bar{z}_r$ we introduce a scaling function defined by

$$\mu(\bar{z}-\bar{z}_r) \triangleq \frac{\|\bar{z}-\bar{z}_r\|^2}{\|\bar{z}-\bar{z}_r\|^2 + \varepsilon^2}, \tag{25.24}$$

where $\varepsilon > 0$ is a design coefficient (cf. also [16]). Next, function μ is used for modification of $\bar{v}$ as follows:

$$\bar{v} = \mu\left(\bar{z} - \bar{z}_r\right)\left(\bar{v}^{As} + \bar{v}^{Os}\right) + \bar{v}^{Ad} + \bar{v}^{Od}. \tag{25.25}$$

From the following relationship: $\lim_{\bar{z}\to\bar{z}_r} \mu\left(\bar{z} - \bar{z}_r\right)\left(\bar{v}^{As} + \bar{v}^{Os}\right) = 0$ it follows that the singularity at $\bar{z} = \bar{z}_r$ is removed. However, in general for $\bar{z} = \bar{z}_r$ term $\bar{v}^{Od}$ is different from zero. As a result, trajectories $\bar{z}(t)$ converge to some neighborhood of $\bar{z}_r$ with the radius dependent on parameter ε, the magnitude of reference velocities $\bar{v}_r$ and $\bar{v}_o$, as well as the distance from the obstacle boundary.

In order to restore the asymptotic convergence to the reference trajectory we require that dynamic component $\bar{v}^{Od}$ should vanish when the obstacle is far away from the robot, namely $\forall \boldsymbol{p} \in \left(\mathscr{Q}_{free} \setminus \tilde{\mathscr{O}}\right)\ \bar{v}^{Ad} = 0$. For that purpose we introduce a continuous switching function defined by $\sigma_{\delta,R} : \mathbb{R} \to [0,\ 1]$ which satisfies the following conditions:

$$\forall \rho < \delta,\ \sigma_{\delta,R} = 0 \text{ and } \forall \rho > R,\ \sigma_{\delta,R} = 1, \tag{25.26}$$

$$\left.\frac{d\sigma_{\delta,R}}{d\rho}\right|_{\rho=\delta} = \left.\frac{d\sigma_{\delta,R}}{d\rho}\right|_{\rho=R} = 0, \tag{25.27}$$

$$\forall \rho \in (\delta, R),\ \frac{d\sigma_{\delta,R}}{d\rho} < 0. \tag{25.28}$$

Assuming that $\rho\left(\bar{z}\right)$ is the Euclidean distance of the point-robot from the obstacle boundary (additionally we require that $\forall \boldsymbol{p} \in \mathscr{O}\ \rho\left(\bar{z}\right) = 0$), we rewrite (25.25) as follows:

$$\bar{v} = \mu\left(\bar{z} - \bar{z}_r\right)\left(\bar{v}^{As} + \bar{v}^{Os}\right) + \bar{v}^{Ad} + \left(1 - \sigma_{\delta,R}\left(\rho\left(\bar{z}\right)\right)\right)\bar{v}^{Od}. \tag{25.29}$$

Taking into account the properties of σ one can show that if the distance between the robot and the obstacle exceeds the assumed value R then the dynamic obstacle potential does not influence the robot motion. This means that the obstacle interacts with the robot only locally.

25.3.2 Motion Planning for Colliding Reference Trajectory

Now we take into account a more general case assuming that the reference trajectory may go through the obstacle, namely $\exists t_2 > t_1 > 0,\ \forall t \in (t_1, t_2),\ \boldsymbol{p}_r(t) \in \mathscr{O}$. Recalling the analysis of singularities for $\boldsymbol{p}_r \in \mathscr{O}$, potential $\bar{w}^s$ is not well defined. Instead of modifying the potential we directly operate on the vector field defined by (25.29) and assume that

$$\bar{v} = \sigma_{\delta,R}\left(\rho\left(\bar{z}_r\right)\right)\mu\left(\bar{z} - \bar{z}_r\right)\left(\bar{v}^{As} + \bar{v}^{Os}\right) + \bar{v}^{Ad} + \left(1 - \sigma_{\delta,R}\left(\rho\left(\bar{z}\right)\right)\right)\bar{v}^{Od}. \tag{25.30}$$

Now it is straightforward to show that $\forall \boldsymbol{p}_r(t) \in \mathscr{O}\ \bar{v} = \bar{v}^{Ad} + \left(1 - \sigma_{\delta,R}\left(\rho\left(\bar{z}\right)\right)\right)\bar{v}^{Od}$, which indicates that the robot motion is governed only by the dynamic part of the

potential. Hence, assuming that the reference trajectory stops at $\mathscr{O}$, the robot does not move. However this case is excluded by assumption A2.

Next, considering the case for which $\sigma_{\delta,h}(\rho(\bar{z})) = 0$ and $\boldsymbol{p} \in \tilde{\mathscr{O}}$ one can show that $\boldsymbol{p}$ cannot stay at $\tilde{\mathscr{O}}$ as a result of a uniform flow (it pushes the robot into the external area of the obstacle) and attractivity of the reference trajectory.

25.4 Simulation Results

To illustrate the theoretical considerations numerical simulations were conducted. The parameters of the algorithm were chosen as $\varepsilon = 1$, $u = 1$, $\delta = 0$, and $R = 0.4$, with the initial position of the robot given by $\boldsymbol{p}(0) = [0\ 0.3]^T$. The trajectories of the goal and the circular obstacle with radius $r = 0.3$ were defined by:

$$\boldsymbol{p}_r(t) = \begin{bmatrix} 0.8\sin(0.1t) \\ 0.8\sin(0.2t) \end{bmatrix} \quad \text{and} \quad \boldsymbol{o}(t) = \begin{bmatrix} 0.6\sin(0.3t) + 0.45 \\ 0.6\sin(0.6t) + 0.25 \end{bmatrix}. \tag{25.31}$$

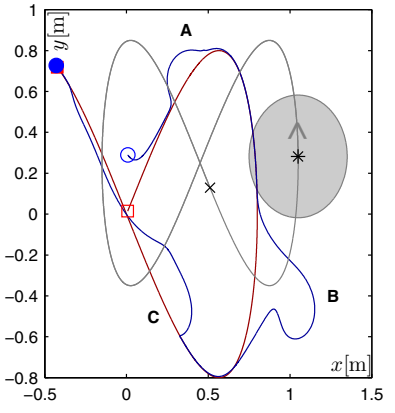


Fig. 25.2 Position trajectories of reference point ($\boldsymbol{p}_r(t)$ – red), center point of the obstacle ($\boldsymbol{o}(t)$ – gray), and point-robot ($\boldsymbol{p}(t)$ – blue)

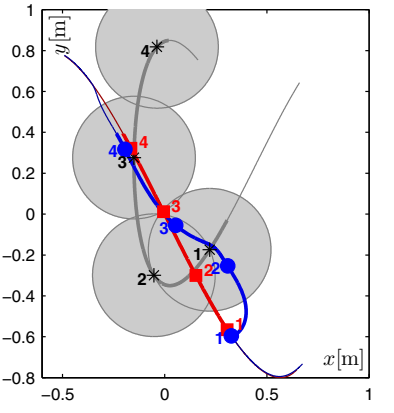


Fig. 25.3 Stroboscopic view illustrating collision avoidance: ($\boldsymbol{p}_r(t)$ – red), ($\boldsymbol{p}(t)$ – blue), ($\boldsymbol{o}(t)$ – gray)

The results of the simulation are given in Figs. 25.2–25.7. Figure 25.2 illustrates the robot, reference, and obstacle paths. It can be seen that trajectory $\boldsymbol{p}(t)$ converges to $\boldsymbol{p}_r(t)$ if $\boldsymbol{p}_r(t)$ is in the collision-free area. From Fig. 25.2 one can notice three collision areas denoted by A, B, and C. In those areas the robot has to avoid the obstacle by increasing the tracking error – this can also be concluded from Fig. 25.6. Moreover, it should be noted that in some more demanding cases the curvature of the robot path may increase rapidly.

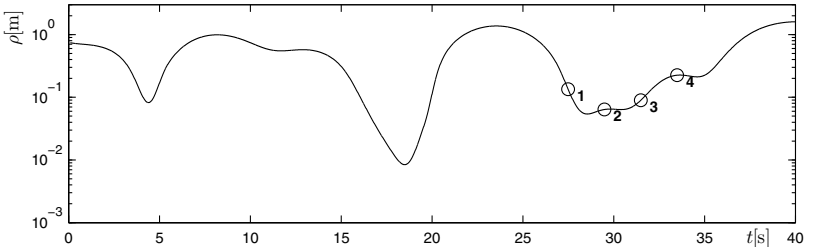


Fig. 25.4 Time plot of the distance between the robot and the the obstacle

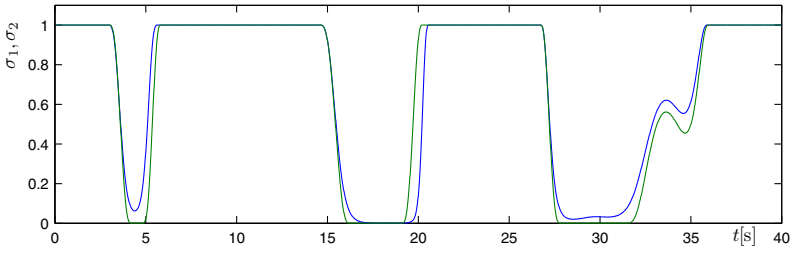


Fig. 25.5 Time plot of switching functions: σ_1 (green), σ_2 (blue)

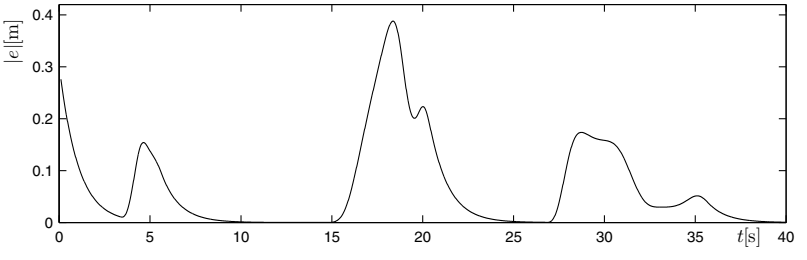


Fig. 25.6 Time plot of euclidean norm of tracking error

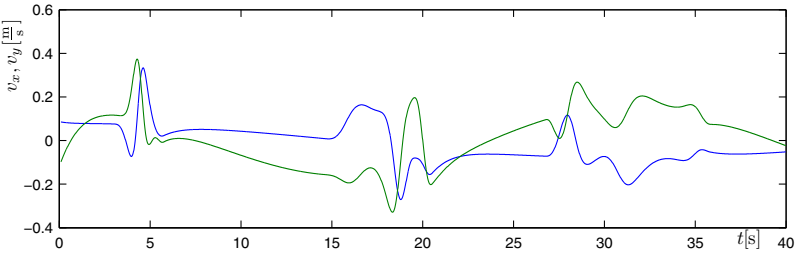


Fig. 25.7 Time plot of velocities v_x (green) and v_y (blue)

One of the collision avoidance scenarios that appeared in area C is illustrated more precisely in Fig. 25.3 using a stroboscopic view. Numbers 1 to 4 are used to point out selected time instants of motion. It can be observed that the robot avoids collision, however it moves relatively close with respect to the obstacle boundary. The collision avoidance can be confirmed from Fig. 25.4, which illustrates the distance between the robot and the obstacle. The time plot of the switching functions presented in Fig. 25.5 (the following notation is used: $\sigma_1 = \sigma_{\delta,R}(\bar{z})$ and $\sigma_2 = \sigma_{\delta,R}(\bar{z}_r)$) indicates that in some time interval the reference trajectory intersects the obstacle and both functions change in quite a smooth way. In spite of the obstacle motion with relatively high velocity, the magnitudes of the input signals v_x and v_y shown in Fig. 25.7 do not exceed the reference values considerably and no oscillatory behavior is found.

25.5 Summary

In this paper a proposition of on-line motion planning for a point-robot in a 2D environment with a moving circular obstacle is formulated. The given algorithm considers the trajectory tracking problem based on the tools taking advantage of the harmonic functions. The discussed method ensures collision avoidance and convergence to the moving goal which is allowed to go through the obstacle. In order to overcome harmonic potential singularities appropriate velocity scaling is proposed. The asymptotic convergence to the goal in collision-free area is ensured assuming local interaction between the robot and the obstacle.

The simulation results confirm the theoretical expectations with respect to the unconstrained point-robot. A possible extension for unicycle-like vehicles can be proposed using, for example, the decoupling techniques presented in [16] or the nonlinear method considered in [13] taking advantage of differentiability of the paths produced by the fluid-based approach.

Considering the possibility of application of the algorithm one can use it for defining the behavioral layer in the overall control structure. The assumed local interaction between the robot and the obstacle seems to corresponds well to the limited measurement range of a typical sensor system. Moreover, it gives a possibility to extend the algorithm for multiple obstacles (assuming that they do not collide with one another) relatively easily.

Then in order to improve the practical properties of the given method one should take into account robot dynamics and imprecise velocity tracking. In this case an additional safety area of a repulsive potential can be taken into account. This problem will be investigated in future work.

Acknowledgements. This work was supported under university grant No. 93/193/11 DS-MK.

References

1. Akishita, S., Kawamura, S., Hayashi, K.: Laplace potential for moving obstacle avoidance and approach of a mobile robot. In: Japan-USA Symposium on Flexible Automation, A Pacific Rim Conference, pp. 139–142 (1990)
2. Axler, S., Bourdon, P., Ramey, W.: Harmonic Function Theory. In: Graduate Texts in Mathematics. Springer, New York (2001)
3. Connolly, C.I., Burns, J.B., Weiss, R.: Path planning using Laplace's equation. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2102–2106 (1990)
4. De Luca, A., Oriolo, G.: Local incremental planning for nonholonomic mobile robots. In: Proceedings of the IEEE International Conference on Robotics and Automation, San Diego, CA, USA, pp. 104–110 (1994)
5. Dulęba, I.: Algorithms of motion planning for nonholonomic robots. Publishing House of Wrocław University of Technology (1998)
6. Feder, H.J.S., Slotine, J.J.E.: Real-time path planning using harmonic potentials in dynamic environments. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 874–881 (1997)
7. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. International Journal of Robotics Research 5, 90–98 (1986)
8. Kim, J., Khosla, P.: Real-time obstacle avoidance using harmonic potential functions. IEEE Transactions on Robotics and Automation 8(3), 338–349 (1992)
9. Kowalczyk, W., Kozłowski, K., Tar, J.: Trajectory tracking for formation of mobile robots. LNCIS, pp. 57–66. Springer, Heidelberg (2009)
10. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
11. Louste, C., Liégeois, A.: Path planning for non-holonomic vehicles: a potential viscous fluid field method. Robotica 20, 291–298 (2002)
12. Milne-Thomson, L.: Theoretical hydrodynamics. Dover Publications (1996)
13. Pazderski, D., Szulczyński, P., Kozłowski, K.: Kinematic tracking controller for unicycle mobile robbot based on polar-like representation and lyapunov analysis. LNCIS, pp. 45–56. Springer, Heidenberg (2009)
14. Rimon, E., Koditschek, D.E.: The construction of analytic diffeomorphisms for exact robot navigation on star worlds. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 21–26 (1989)
15. Roussos, G.P., Dimarogonas, D.V., Kyriakopoulos, K.J.: 3d navigation and collision avoidance for a non-holonomic vehicle. In: Proceedings of American Control Conference, Seattle, WA, USA, pp. 3512–3517 (2008)
16. Szulczyński, P., Pazderski, D., Kozłowski, K.: Real-time obstacle avoidance using harmonic potential functions. Jamris (in Print, 2011)
17. Waydo, S., Murray, R.M.: Vehicle motion planning using stream functions. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2484–2491 (2003)

Chapter 26
Motion Planning of Nonholonomic Systems – Nondeterministic Endogenous Configuration Space Approach

Mariusz Janiak

Abstract. This paper presents a new nondeterministic motion planning algorithm for nonholonomic systems. Such systems are represented by a driftless control system with outputs. The presented approach combines two different methods: the endogenous configuration space approach and the Particle Filters. The former, fully deterministic, was originally dedicated to the motion planning problem for mobile manipulators. The latter, of stochastic approach, was designed for solving optimal estimation problems in non-linear non-Gaussian systems. A mixture of these methods results in a nondeterministic endogenous configuration space approach that outperforms the traditional one in regions where the classical inverse Jacobian algorithm loses convergence. In accordance with the Particle Filters approach the new algorithm consists of three major steps: prediction, update, and resampling. In contrast to its original version, the presented algorithm contains an additional step of dividing the particles into different subsets. Each subset is processed in a different way during the prediction phase. The performance of the new algorithm is illustrated by solving the motion planning problem for a rolling ball.

26.1 Introduction

The presented nondeterministic approach is closely related to the Particle Filter method [1, 4, 5], originally dedicated to finding a solution of optimal estimation problems in non-linear non-Gaussian systems. It is a technique for implementing a recursive Bayesian filter by a Monte Carlo simulation. Particle Filters belong to the large family of Sequential Monte Carlo methods. The unknown posterior density function is estimated by a set of random particles, representing the system state, and weights associated with these particles, that define the quality of individual

Mariusz Janiak
Institute of Computer Engineering, Control and Robotics,
Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: mariusz.janiak@pwr.wroc.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 305–315.
springerlink.com

particles. An estimate of the variable of interest can be obtained, for example, by the weighted sum of all the particles. The resulting estimate approaches the optimal Bayesian estimate when the number of particles increases to infinity. The Particle Filter algorithm is recursive by design, and consists of three major steps: prediction, update, and resampling. During the prediction step particles are generated in accordance with the system model containing noise. Then for each particle its weight is computed, based on the latest measurement and the measurement model – this is the update step. After that, the resampling procedure is executed, which consists in choosing the set of the best particles to form a new population of particles for the next algorithm step. Resampling is a method that prevents particle weight degeneration. For more detailed description of the Particle Filter method the reader is referred to [4]. A non standard application of Particle Filters has been presented in [8] and [13], where the method has been successfully adopted to optimization problems.

The endogenous configuration space approach has been thoroughly presented in many recent publications [6, 11, 12]. With this approach, it is possible to generalize Jacobian inverse kinematics algorithms known for classic stationary manipulators to other robotics systems. The fundamental concept of the endogenous configuration includes all admissible controls of the platform and joint positions of the on-board manipulator. This method has been widely applied to mobile platforms [6, 12], mobile manipulators [7, 11], and underactuated systems with dynamics [9, 10]. The standard inverse Jacobian algorithm derived from the endogenous configuration space approach has several drawbacks. It is a kind of Newton algorithm defined in a Hilbert space, therefore it is local and safer from local minima. Moreover, a starting point selection method which guarantees global convergence is not defined, so if we choose an inappropriate starting point, the algorithm will not find a solution. Finally, the Jacobian algorithm is an iterative method, difficult to parallelize.

In the presented nondeterministic endogenous configuration space approach the Particle Filter framework is used in a slightly different fashion, namely to randomly search the endogenous configuration space of the nonholonomic system in order to find the solution of the motion planning problem. In accordance to the Particle Filter approach, the presented method also consists of three major steps: prediction, update, and resampling. There is an additional last step, when particles are sorted according to the value of their weights, and divided into three different particle subsets. During the prediction phase each subset is processed by a specific algorithm. The best particle is processed by the deterministic inverse Jacobian algorithm. Particles of the next subset are evaluated by means of a nondeterministic inverse Jacobian algorithm. The last subset of particles, with the lowest weight, acts as a perturbed copy of the best particle. The nondeterministic inverse Jacobian algorithm compared to the deterministic one, has additional noise which perturbs resulting control parameters. All disturbances occurring in the system vanish when the algorithm approaches the solution. This should prevent the algorithm from losing convergence. The presented nondeterministic endogenous configuration space approach removes the mentioned weakness of the original method. The resulting algorithm is still local but it allows for exploring many different solutions simultaneously. The starting

points are selected randomly, so if one or more points are inappropriate, the algorithm is still able to converge. Finally, the algorithm can be easily parallelized, like a Particle Filter method. The performance of the presented nondeterministic approach has been tested with computer simulations of a motion planning problem for a rolling ball.

The paper is organized as follows. Section 26.2 contains a description of the classical deterministic endogenous configuration space approach, including the inverse Jacobian algorithm and implementation issues. The nondeterministic approach is discussed in detail in Section 26.3. In Section 26.4 we present the results of computer simulations. The paper is concluded with Section 26.5.

26.2 Classical Deterministic Approach

We shall consider a nonholonomic system characterized by generalized coordinates $q \in \mathbb{R}^n$ and velocities $\dot{q} \in \mathbb{R}^n$, subject to $l < n$ non-integrable Pfaffian constraints $A(q)\dot{q} = 0$. Assuming that the distribution $q \mapsto \mathrm{Ker}A(q)$ is spanned by vector fields $g_1(q), \ldots, g_m(q)$, $m = n - l$, and that $y = (y_1, \ldots, y_r) \in \mathbb{R}^r$ denotes a vector of task space coordinates, we obtain a representation of the nonholonomic system in the form of a driftless control system with output

$$\begin{cases} \dot{q} = G(q)u = \sum_{i=1}^{m} g_i(q)u_i, \\ y = k(q) = \big(k_1(q), \ldots, k_r(q)\big), \end{cases} \tag{26.1}$$

where $u = (u_1, \ldots, u_m) \in \mathbb{R}^m$ denotes a control vector. Each control function $u(\cdot) \in \mathscr{U} = \mathbb{L}^2_m[0,T]$ is assumed Lebesgue square integrable over a time interval $[0,T]$. The control space $\mathscr{U}$ will be called endogenous configuration space of the nonholonomic system. The endogenous configuration space $\mathscr{U}$ is an infinite-dimensional Hilbert space with inner product $\langle (u_1(\cdot), u_2(\cdot)\rangle = \int_0^T u_1^T(t)u_2(t)dt$.

A control $u(\cdot)$ applied to 26.1 produces a state trajectory $q(t) = \varphi_{q_0,t}\big(u(\cdot)\big)$, initialized from $q_0 = q(0)$. We shall assume that this trajectory is well defined for every $t \in [0,T]$; then the end-point map $K_{q_0,T} : \mathscr{U} \to \mathbb{R}^r$ of this system will be defined as

$$K_{q_0,T}\big(u(\cdot)\big) = k\big(q(T)\big) = k\Big(\varphi_{q_0,T}\big(u(\cdot)\big)\Big). \tag{26.2}$$

Given the control system representation 26.1, the motion planning problem of the nonholonomic system consists in defining a control function $u(\cdot)$ such that the system output reaches the desirable point $y_d \in \mathbb{R}^r$ in the task space at a prescribed time instance T, thus $K_{q_0,T}\big(u(\cdot)\big) = y_d$.

In accordance to the endogenous configuration space approach, the motion planning problem is solved numerically by means of the inverse Jacobian algorithm. The derivation of the algorithm is the following. First, we choose a curve $u_\theta(\cdot) \in \mathscr{U}$, parameterized by $\theta \in \mathbb{R}$, and starting from an arbitrarily chosen point $u_0(\cdot)$ in the endogenous configuration space. Along this curve, we define the task space error

$$e(\theta) = K_{q_0,T}\big(u_\theta(\cdot)\big) - y_d. \tag{26.3}$$

Differentiation of error 26.3 with respect to θ yields

$$\frac{de(\theta)}{d\theta} = J_{q_0,T}\big(u_\theta(\cdot)\big)\frac{du_\theta(\cdot)}{d\theta}, \tag{26.4}$$

where

$$J_{q_0,T}\big(u(\cdot)\big)v(\cdot) = \mathrm{D}K_{q_0,T}\big(u(\cdot)\big)v(\cdot) = \frac{d}{d\alpha}\bigg|_{\alpha=0} K_{q_0,T}\big(u(\cdot)+\alpha v(\cdot)\big) = C(T)\int_0^T \Phi(T,t)B(t)v(t)dt \tag{26.5}$$

denotes the Jacobian of the nonholonomic system. The matrices appearing on the right-hand side of 26.5 come from the linear approximation of the system 26.1 along $\big(u(t),q(t)\big)$, therefore $A(t) = \frac{\partial}{\partial q}G\big(q(t)\big)u(t)$, $B(t) = G\big(q(t)\big)$, $C(t) = \frac{\partial}{\partial q}k\big(q(t)\big)$ and the fundamental matrix $\Phi(t,s)$ fulfils the evolution equation $\frac{\partial}{\partial t}\Phi(t,s) = A(t)\Phi(t,s)$, $\Phi(s,s) = I_n$. For a fixed $u(\cdot)$, the Jacobian transforms the endogenous configuration space into the task space $J_{q_0,T}\big(u(\cdot)\big) : \mathscr{U} \to \mathbb{R}^r$. Configurations at which this map is surjective are named regular, otherwise they are singular. Equivalently, $u(\cdot)$ is regular if and only if the Gram matrix

$$\mathscr{G}_{q_0,T}\big(u(\cdot)\big) = C(T)\int_0^T \Phi(T,t)B(t)B^T(t)\Phi^T(T,t)dtC^T(T) \tag{26.6}$$

has full rank r.

The curve $u_\theta(\cdot)$ is requested to decrease the error 26.3 exponentially with a rate $\gamma > 0$, so

$$\frac{de(\theta)}{d\theta} = -\gamma e(\theta). \tag{26.7}$$

A comparison of 26.4 and 26.7 results in the following implicit differential equation

$$J_{q_0,T}\big(u_\theta(\cdot)\big)\frac{du_\theta(\cdot)}{d\theta} = -\gamma e(\theta). \tag{26.8}$$

Using a right inverse Jacobian operator $J^{\#}_{q_0,T}\big(u(\cdot)\big) : \mathbb{R}^r \to \mathscr{U}$, the equation 26.8 is converted into a dynamic system

$$\frac{du_\theta(\cdot)}{d\theta} = -\gamma J^{\#}_{q_0,T}\big(u_\theta(\cdot)\big)e(\theta), \tag{26.9}$$

whose limit trajectory $u_d(t) = \lim_{\theta\to+\infty} u_\theta(t)$ provides a solution to the motion planning problem. Specifically, in this paper we shall use the Jacobian pseudo inverse

$\left(J^{\#P}_{q_0,T}(u(\cdot))\eta\right)(t) = B^T(t)\Phi^T(T,t)\mathscr{G}^{-1}_{q_0,T}(u(\cdot))\eta$, obtained as the least squares solution of the Jacobian equation [11].

For practical and computational reasons, it is useful to employ a finite-dimensional representation of the endogenous configuration as well as a discrete version of the inverse Jacobian algorithm. For this purpose, we assume that the control functions in 26.1 are represented by truncated orthogonal series, i.e. $u(t) = P(t)\lambda$, where $P(t)$ is a block matrix comprising basic orthogonal functions in the Hilbert space $\mathbb{L}^2_m[0,T]$, and $\lambda \in \mathbb{R}^s$ denotes control parameters. It follows that the endogenous configuration $u(\cdot)$ gets represented by a point $\lambda \in \mathbb{R}^s$, while the system Jacobian is defined by a matrix

$$\bar{J}_{q_0,T}(\lambda) = C_\lambda(T)\int_0^T \Phi_\lambda(T,t)B_\lambda(t)P(t)dt\,, \tag{26.10}$$

where matrices $\Phi_\lambda(T,t)$, $B_\lambda(t)$, and $C_\lambda(T)$ are finite-dimensional equivalents of $\Phi(T,t)$, $B(t)$ and $C(T)$. In the finite-dimensional case, the Jacobian pseudo inverse of 26.10 takes a standard form $\bar{J}^{\#P}_{q_0,T}(\lambda) = \bar{J}^T_{q_0,T}(\lambda)\left(\bar{J}_{q_0,T}(\lambda)\bar{J}^T_{q_0,T}(\lambda)\right)^{-1}$. Consequently, in agreement with 26.9, the discrete version of the inverse Jacobian algorithm is defined by the following dynamic system

$$\lambda(\theta+1) = \lambda(\theta) - \gamma\bar{J}^{\#P}_{q_0,T}\left(\lambda(\theta)\right)e(\theta). \tag{26.11}$$

Within the endogenous configuration space approach, the algorithm 26.11 is most commonly used to solve the motion planning problem for nonholonomic systems.

26.3 Nondeterministic Approach

Let $\Lambda = (\lambda, w)$ denote a particle composed of a control parameter $\lambda \in \mathbb{R}^s$, representing a point in the parameterized endogenous configuration space, and an associated weight $w \in [0,1]$. We define three sets of particles $\mathscr{A} = \{\Lambda^i : i = 1,\ldots,N_a\}$, $\mathscr{B} = \{\Lambda^i : i = 1,\ldots,N_b\}$, and a set containing only one element, the best particle Λ^*. Each set of particles will be processed in a different way during the prediction phase. For this reason, there are three different strategies of evaluating the system state during this phase.

The best particle $\Lambda^* = (\lambda^*, w^*)$ is the particle with the highest weight. The system state of this particle is processed in the deterministic way, in accordance with equation 26.11. This should improve convergence of the algorithm in close proximity of the solution.

The system state of the particles $\Lambda^i = (\lambda^i, w^i) \in \mathscr{A}$ is processed using the rule

$$\lambda^i(\theta+1) = \lambda^i(\theta) - \gamma\bar{J}^{\#P}_{q_0,T}\left(\lambda^i(\theta)\right)e^i(\theta) - \omega^i(\theta)\,, \tag{26.12}$$

where $i = 1,\ldots,N_a$ and $\omega^i \in \mathbb{R}^s$ denotes system noise. This noise is computed using the following procedure. First, the error noise vector defined as $e^i_\omega(\theta) = \big(e^i_{\omega,1}(\theta),\ldots,e^i_{\omega,r}(\theta)\big)$ is generated, whose elements are selected from the normal distribution

$$e^i_{\omega,j}(\theta) = N\big(0,|e^i_j(\theta)|\sigma^e_j\big), \quad j = 1,\ldots,r, \tag{26.13}$$

where σ^e_j denotes the error standard deviation scaling factor. Then this error is propagated through the inverse Jacobian according to $\omega^i_e(\theta) = \gamma \bar{J}^{\#P}_{q_0,T}\big(\lambda^i(\theta)\big)e^i_\omega(\theta)$. The vector $\omega^i_e(\theta)$ is a candidate for the system noise vector. The elements of $\omega^i_e(\theta) \in \mathbb{R}^s$ have been computed using only r-dimensional random variable $e^i_\omega(\theta)$ and $r < s$. For this reason, in order to compute the system noise, each element of $\omega^i_e(\theta)$ is perturbed by additional small random noise, so finally $\omega^i_j(\theta) = \omega^i_{e,j}(\theta) + N\big(0,|\omega^i_{e,j}(\theta)|\sigma^\lambda\big)$, where $j = 1,\ldots,s$ and σ^λ denotes standard deviation scaling factor of the control parameters. According to 26.13, when $\|e^i(\theta)\| \to 0$ then $\|e^i_\omega(\theta)\| \to 0$. This means that the system noise vanishes when the particle approaches the solution of the motion planning problem.

The modification of the system state of particles $\Lambda^i \in \mathscr{B}$ consists in perturbing the best particle $\Lambda^* = (\lambda^*, w^*)$ by the randomly generated noise

$$\lambda^i_j(\theta+1) = \lambda^*_j(\theta) + N(0,|\lambda^*_j(\theta)|\mu(\theta)\sigma^*)),$$

where $i = 1,\ldots,N_b$, $j = 1,\ldots,s$, σ^* denotes the standard deviation scaling factor of the best particle control parameters, and decay factor $\mu(\theta) = \frac{\|e^*(\theta)\|}{\|e(0)\|}$, $e(0) = k(q_0) - y_d$, $\mu(\theta) \to 0$ when the best particle approaches the solution. It is possible to define a different modification strategy, for set $\mathscr{B}$, or define a new set of particles with a new prediction algorithm. Such an algorithm should increase the resultant convergence in regions where the standard inverse Jacobian algorithm suffers from a lack of convergence.

During the update phase, particle weights are generated. This procedure is similar for all particle sets. The measurement model is defined by equation 26.2. For each particle, using new state $\lambda(\theta+1)$, the task space error is computed from equation 26.3. Then, the candidate for the best particle with the lowest norm of task space error $e^*(\theta+1)$ is selected. After that, the particle weights are computed using the multivariate normal distribution

$$w^i(\theta+1) = \frac{1}{(2\pi)^{\frac{r}{2}}\sqrt{\det\Sigma(\theta+1)}}\exp\Big(-0.5\big(e^i(\theta+1)\big)^T\Sigma^{-1}(\theta+1)e^i(\theta+1)\Big),$$

where covariance matrix $\Sigma(\theta+1) = diag\big\{\big(e^*_1(\theta+1)\sigma^y_1\big)^2,\ldots,\big(e^*_r(\theta+1)\sigma^y_r\big)^2\big\}$, and $\sigma^y \in \mathbb{R}^r$ is the standard deviation scaling factor of the measurements. When $\|e^*\| \to 0$, then $\det\Sigma \to 0$, so the particles get lower weights when the best particle approaches the solution. Finally, the weights of all particles in the population are normalized: $w^i(\theta+1) = \frac{w^i(\theta+1)}{\sum_{i=1}^N w^i(\theta+1)}$, $N = 1 + N_a + N_b$.

One of the well-known problems associated with particle filters is the particle weights degeneration. It turns out that after a certain number of recursive steps, most

of the particles will have negligible normalized weights. This implies that a large portion of the computation is devoted to updating particles whose contribution to solution finding is almost zero. The effective sample size N_{eff} is a suitable measure of the algorithm degeneracy, and can be estimated by

$$\hat{N}_{eff}(\theta+1) = \frac{1}{\sum_{i=1}^{N}(w^i(\theta+1))^2}, \quad N = 1 + N_a + N_b.$$

When $\hat{N}_{eff}(\theta+1)$ drops below a certain threshold, a resampling procedure is executed. Many different resampling algorithms have been proposed [2, 3]; most commonly used are: residual, systematic, and multinomial. According to the recommendation presented in [4], we have chosen the systematic approach; for more information the reader is directed to [3]. Although the resampling step reduces the weight degeneracy effect, it introduces other problems. For example, particles with high weights are selected many times, which leads to the loss of diversity among the particles. In the case of small process noise, all particles may even collapse to a single point within a few iterations. The selection of the resampling algorithm suitable for nondeterministic endogenous configuration space approach will be the subject of the further investigation.

The last phase of the presented nondeterministic approach consists in dividing the particles into three different subsets Λ^*, $\mathscr{A}$, and $\mathscr{B}$. In its course, all particles are placed together and sorted by decreasing values of their weights. The first particle is the best particle Λ^*, the next N_a particles are placed in subset $\mathscr{A}$, the rest of them is placed in subset $\mathscr{B}$.

For completeness of this section the algorithm design parameters will be discussed. The classical deterministic inverse Jacobian algorithm has four of them: γ, s and λ_0; for more information the reader is directed to [11]. In addition to these parameters, the new approach introduces additional ones: σ^e, σ^λ, σ^*, σ^y, N, N_a, and N_b. The parameters σ^e, σ^λ, σ^*, and σ^y denote the standard deviation scaling factors of: the error, the control parameters, the best particle control parameters, and the measurements, respectively. Generally speaking, the parameters σ^e, σ^λ, σ^* affect the particle scattering during the prediction phase. Greater values of these parameters increase particle spreading, but also reduce the algorithm's convergence abilities. The parameter σ^y influences the assessment of particles during the update phase. Increasing the value of this parameter promotes particles with greater task space error, which increases the probability of these particles becoming part of a new particle population after the resampling phase. The parameter N defines the number of the particle population. Increasing the population size improves the algorithm's search abilities but also increases its computational complexity. The parameter N_a defines the number of the particles processed with the nondeterministic inverse Jacobian algorithm, while N_b is the number of particles that will become perturbed copies of the best particle. The ratio between N_a and N_b affects the promotion of the model based search algorithm ($N_a > N_b$) or the random search algorithm ($N_a < N_b$).

26.4 Computer Simulations

The performance of the presented nondeterministic endogenous configuration space approach will be illustrated by computer simulation of a motion planning problem for a rolling ball. The rolling ball is shown in Fig. 26.1. The ball is not permitted to slip along meridian or parallel, and to veer at the point of contact with the ground. Respecting the rolling conditions, the motion of the ball can be described in coordinates by the following control system

$$\begin{cases} \dot{q}_1 = u_1 \sin q_4 \sin q_5 + u_2 \cos q_5, \quad \dot{q}_2 = -u_1 \sin q_4 \cos q_5 + u_2 \sin q_5, \\ \dot{q}_3 = u_1, \quad \dot{q}_4 = u_2, \quad \dot{q}_5 = -u_1 \cos q_4, \\ y = k(q) = (q_1, q_2, q_5), \end{cases} \tag{26.14}$$

where $(q_1, q_2, q_3, q_4, q_5) = (x, y, \varphi, \theta, \psi)$, see the figure. The chosen output function allows for tracking the ball position and orientation on a plane, therefore the exemplary motion planning problem will be defined only with respect to these platform coordinates. By choosing a different output function it is possible to define the planning task with respect to all platform coordinates. Given time $T = 1$, the platform controls entering the system 26.14 will be chosen in the form of a finite piecewise constant series $u_i(t) = \sum_{j=1}^{s_i} \lambda_{ij} \zeta_{t_{j-1}}^{t_j}(x)$, $\zeta_{t_b}^{t_e}(x) \begin{cases} 1, & t_b \le x < t_e, \\ 0, & \text{otherwise}, \end{cases}$ where $i = 1, 2$, $t_0 < t_1 < \ldots < t_{s_i}$, $t_{i+1} - t_i = \frac{T}{si}$, and s_i is length of the ith control series, $s_1 = s_2 = 5$, so $\lambda \in \mathbb{R}^{10}$. The initial state of the ball is $q_0 = (1000, 0, 0, \frac{\pi}{2}, 0)$, and the desirable task space point $y_d = (0, 0, 0)$. The parameters of the algorithm have been set to $\gamma = 0.2$, $\sigma^e = (0.4, 0.4, 0.2)$, $\sigma^\lambda = 0.5$, $\sigma^* = 0.2$, $\sigma^y = (10, 10, 10)$. The number of the particle population has been chosen experimentally as $N = 30$, as well as the sizes of sets $N_a = 14$, $N_b = 15$. This was a trade off between the algorithm's search abilities and its computational complexity. The initial state of particles has been randomly generated. The algorithm stops when the final error drops below 10^{-1}. The maximum number of iterations is 100. In order to solve the motion planning problem, the algorithm has performed 55 iterations. The results of the computations are displayed in Fig. 26.2. On the left-hand side there are plots of particle positions

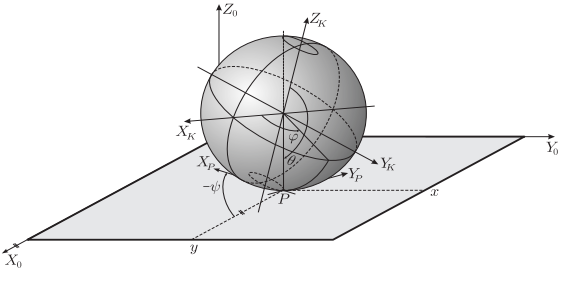


Fig. 26.1 The rolling ball

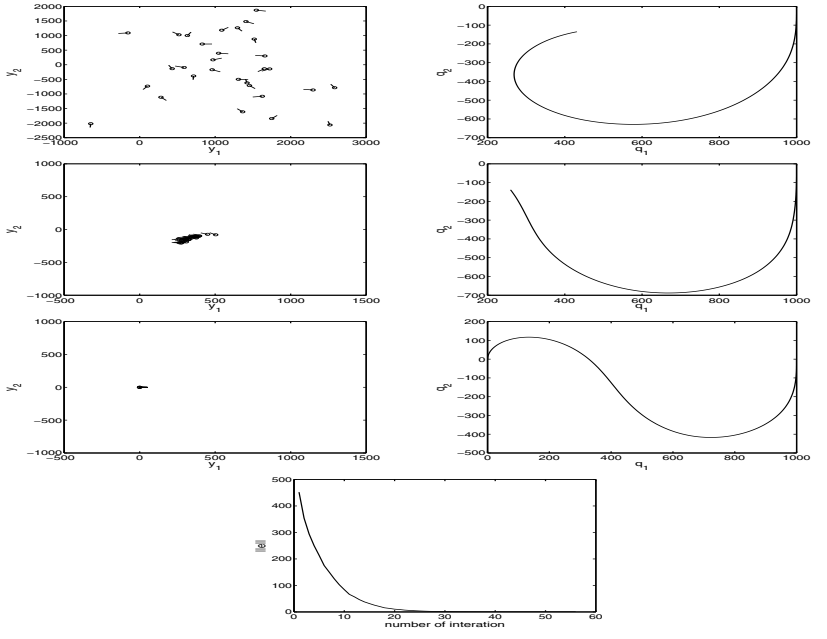


Fig. 26.2 Simulation results

mapped to the task space (a circle marks the ball position, a line segment marks the ball orientation). The ball paths corresponding to the best particle are shown on the right-hand side. The plots in the first row illustrate the initial algorithm state; the next row contains the state after the second algorithm iteration. The third row contains the solution. The last row illustrates the convergence of the algorithm.

26.5 Conclusions

The paper has introduced a new nondeterministic approach designed for a motion planning problem of nonholonomic systems. The presented approach combines two different methods, the deterministic endogenous configuration space approach, originally dedicated to the motion planning problem of mobile manipulators, and the stochastic method of Particle Filters, created for solving optimal estimation problems in non-linear non-Gaussian systems. The presented algorithm removes the main drawbacks of the deterministic inverse Jacobian algorithm derived from the endogenous configuration space approach.

Within the presented approach the Particle Filter framework has been used for randomly searching the endogenous configuration space of the nonholonomic system in order to find the solution of the motion planning problem. The particles are composed of the control parameter, representing a point in the parameterized endogenous configuration space, and the associated weight. The particles are divided

into three different sets: a one-element set containing the best particle and two other sets. The state evolution of the particles belonging to each set is defined by three different algorithms, during the prediction phase. The best particle is evaluated by the standard deterministic inverse Jacobian algorithm. The second set is processed using a nondeterministic version of the inverse Jacobian algorithm endowed with random noise. The third set of particles contains randomly perturbed copies of the best particle. During the updating phase, based on the measurement model and the appropriate multivariate normal distribution, the weights of all particles are generated. Then, if the effective sample size factor drops below a certain threshold, the systematic resampling procedure is executed. Unlike in the Particle Filter method, there is an additional step which consists in dividing particles into different particle subsets according to their weights. Simulations have confirmed the expected algorithm performance, and simultaneously demonstrated its fast convergence and excellent accuracy for very large distances of the mobile robot from its destination. The original deterministic algorithm could not find a solution of this problem.

Acknowledgements. This research was supported by a statutory grant from the Wrocław University of Technology. I would like to thank Professor Krzysztof Tchoń, who helped me improve this document.

References

1. Arulampalam, M.S., Maskell, S., Gordon, N.: A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. IEEE Trans. Sig. Process. 50, 174–188 (2002)
2. Bolic, M., Djuric, P.M., Hong, S.: Resampling algorithms and architectures for distributed particle filters. IEEE Trans. Sig. Process. 53, 2442–2450 (2004)
3. Douc, R.: Comparison of resampling schemes for particle filtering. In: 4th International Symposium on Image and Signal Processing and Analysis (ISPA), pp. 64–69 (2005)
4. Doucet, A., Johansen, A.M.: A Tutorial on Particle Filtering and Smoothing: Fifteen years Later. In: Crisan, D., Rozovsky, B. (eds.) Handbook of Nonlinear Filtering. Oxford University Press (2009)
5. Gordon, N., Salmond, D., Smith, A.: Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In: IEE-Proceedings-F, vol. 140, pp. 107–113 (1993)
6. Janiak, M., Tchoń, K.: Constrained robot motion panning: Imbalanced jacobian algorithm vs. optimal control approach. In: Proc. 15th Int. MMAR Conf., Miedzyzdroje, pp. 25–30 (2010)
7. Janiak, M., Tchoń, K.: Towards constrained motion planning of mobile manipulators. In: IEEE Int. Conf. Robot. Automat., Anchorage, Alaska, pp. 4990–4995 (2010)
8. Pantrigo, J.J., Sanchez, A.: Hybridizing particle filters and population-based metaheuristics for dynamic optimization problems. In: Proc. of the V International Conf. on Hybrid Intelligent Systems, pp. 41–48. IEEE Computer Society Press, Washington, DC, USA (2005)
9. Ratajczak, A., Janiak, M.: Motion planning of an underactuated manipulators with state space constraints. Scientific Papers of Warsaw University of Technology 175(2), 495–504 (2010) (in Polish)

10. Ratajczak, A., Karpiska, J., Tchoń, K.: Task-priority motion planning of underactuated systems: an endogenous configuration space approach. Robotica 28, 885–892 (2009)
11. Tchoń, K., Jakubiak, J.: Endogenous configuration space approach to mobile manipulators: a derivation and performance assessment of Jacobian inverse kinematics algorithms. Int. J. Contr. 76(14), 1387–1419 (2003)
12. Tchoń, K., Jakubiak, J.: Extended Jacobian inverse kinematics algorithm for nonholonomic mobile robots. Int. J. Contr. 79, 895–909 (2006)
13. Zhou, E., Fu, M.C., Marcus, S.I.: A particle filtering framework for randomized optimization algorithms. In: Proc. of the 40th Conf. on Winter Simulation, WSC 2008, pp. 647–654 (2008)

Part V
Control of Flying Robots

Chapter 27
Generation of Time Optimal Trajectories of an Autonomous Airship

Yasmina Bestaoui and Elie Kahale

27.1 Introduction

The natural wind proved itself to be a major parameter to successful flights of airships. It mostly affects a trajectory through its speed. In general, the wind speed can be modeled as a sum of two components: a nominal deterministic component (available through meteorological forecasts or measured with a Doppler radar) and a stochastic component, representing deviations from the nominal one [1, 2]. The closed loop controller takes care of the stochastic part considered as perturbations, while the deterministic component is introduced into the motion planner. In general, the optimality of a trajectory can be defined according to several objectives, like minimizing the transfer time or the energy. Traditionally, trajectories are optimized by the application of numerical optimal control methods that are based on the calculus of variations. Dubins [3] considered a particle moving at a constant velocity in the plane with a constraint of trajectory curvature which is equivalent to the minimum time optimal trajectory under limitations on control and without wind. He proved the existence of the shortest paths for his problem and showed that the optimal trajectory is one of the following six solutions: $\{RSL, RSR, LSR, LSL, RLR, LRL\}$, where R stands for Right turn, S for straight line and L for Left turn. Knowing that all subpaths are allowed to have zero length. In other words the optimal trajectories are a combination of arcs of circles and segments of lines. Boukraa et al [4] presented a 3D trim trajectory planner algorithm for an autonomous plane. The proposed algorithm used a sequence of five elementary trim trajectories to generate a 3D global trajectory in space. A family of trim trajectories in level flight is used in all these references to construct paths. In the papers cited above, the atmosphere was considered to be an isotropic and homogeneous medium, i.e. with no wind and the air density constant with altitude. However, wind cannot be ignored. McGee et al [5] describe a method for finding the minimum time

Yasmina Bestaoui · Elie Kahale
Laboratoire IBISC, Université d'Evry Val d'Essonne, 91025 Evry Cedex, France
e-mail: {bestaoui,kahale}@iup.univ-evry.fr

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 319–329.
springerlink.com

path from an initial position and orientation to a final position and orientation in the 2D plane for an airplane with a bounded turning rate in the presence of a known constant wind with a magnitude less than the airplane velocity. The problem statement is equivalent to finding the minimum time path from an initial configuration to a final one, over a moving virtual target, where the velocity of the virtual target is equal and opposite to the velocity of the wind. Nelson et al [6] have introduced a method for mini aerial vehicle path following based on the concept of vector field in the presence of constant wind disturbances. Rysdyk [7] presents a path formulation for maneuvering a fixed wing aircraft in wind. The inertial path of a fixed wing aircraft circling in wind can be formulated as a trochoid curve. In these papers, only 2D horizontal motion was considered. The originality of the work is twofold: firstly, planning in 3D with varying velocity, heading angle, and path angle, secondly, taking into account the wind effect.

This paper consists of five sections. Section 27.2 presents an analysis of accessibility and controllability while Section 27.3 formulates the time optimal problem with an analysis of the Lagrange multipliers and of one set of solutions. Section 27.4 presents some numerical results. Finally, some conclusions and perspectives are the subject of Section 27.5.

27.2 Accessibility and Controllability

Driftless nonholonomic control systems have been extensively studied in recent years. Chow's theorem [8] leads to the characterization of controllability for systems without drift. It provides a Lie algebra rank test, for controllability of nonlinear systems without drift, similar in spirit to that of Kalman's rank condition for linear systems. For controlled mechanical systems, the Lagrangian dynamics, being second order, necessarily includes drift. In this setting, Chow's theorem cannot be used to conclude controllability. Discussion of nonholonomic systems with drift in the literature has been concentrated on the so-called dynamic extension of drift-free systems with the addition of integrators. Sufficient conditions for the controllability of a conservative dynamical nonlinear affine control system on a compact Riemannian manifold are presented, if the drift vector field is assumed to be weakly positively Poisson stable.

Let us begin with a brief review of some concepts in controllability of nonlinear systems applied to our system. We can write the translational kinematics of an airship as the following affine nonlinear control system with drift:

$$\dot{X} = f(X) + g_1 u_1 + g_2 u_2 + g_3 u_3, \tag{27.1}$$

where the state variable is defined as $X = (x, y, z, \gamma, \chi, V)^T$, the control variable by $U = \left(\dot{\gamma}, \dot{\chi}, \dot{V}\right)^T$, the drift function $f(x)$ and the input function $g(x)$ by:

$$f(X)=\begin{pmatrix} V\cos\chi\cos\gamma+W_x \\ V\sin\chi\cos\gamma+W_y \\ V\sin\gamma+W_z \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \text{and} \quad \begin{aligned} g_1&=(0,0,0,1,0,0)^T, \\ g_2&=(0,0,0,0,1,0)^T, \\ g_3&=(0,0,0,0,0,1)^T. \end{aligned} \tag{27.2}$$

Several important results have been derived based on the structure of the Lie algebra generated by the control vector fields. Assume $X \in M \subset \mathbb{R}^6$, where M is a smooth manifold. Let $X(t)$ denote the solution of (27.1) for $t \geq 0$, particular input function u and initial condition $X(0)=X_0$. The nonlinear system (27.1) is controllable if for two points x_1 and x_2 in M there exists a finite time T and an admissible control function $u:[0,T]\rightarrow U$ such that $x(T)=x_2$. Let ψ be a neighborhood of the point $X \in M$ and $R^{\psi}(x_0,t)$ indicate the set of reachable states from x_0 at time T as:

$$R^M(x_0,T)=\bigcup_{0\leq t\leq T} R^M(x_0,t).$$

The ***accessibility algebra*** A of the system (27.1) is the smallest Lie algebra of vector fields on M that contains the vector fields f and g_1,g_2,g_3. The ***accessibility distribution*** $\triangle_A$ of (27.1) is the distribution generated by the vector fields in A; i.e. $A(x)$ is the span of vector fields v in A at x. So, we can determine $\triangle_A$ as $\triangle_A=\text{span}\{v \mid v\in A\}$. The computation of $\triangle_A$ may be organized as an iterative procedure: $\triangle_A=\text{span}\{v \mid v\in A_i,\forall i\geq 1\}$, with:

$$\begin{aligned} &\triangle_1=\triangle=\text{span}\{f,g_1,g_2,g_3\}, \\ &\triangle_i=\triangle_{i-1}+\text{span}\{[g,v] \mid g\in\triangle_1, v\in\triangle_{i-1}\}; i\geq 2. \end{aligned} \tag{27.3}$$

This procedure stops after K steps, where K is the smallest integer such that $\triangle_{K+1}=\triangle_K>\triangle_A$. This number is the non-holonomy degree of the system and is related to the level of Lie brackets that need to be included in $\triangle_A$. The system (27.1) is ***accessible*** from $x_0\in M$ if for every $T>0$, $R^M(x_0,T)$ contains a nonempty open set. Moreover, the system (27.1) is ***locally accessible*** from $x\in M$ if for every $T>0$, $R^{\Psi}(x_0,T)$ contains a nonempty open set. If the vector fields are C^{∞} in (27.1) and if $\dim\triangle_A(x_0)=n$ (i.e. the accessibility algebra spans the tangent space to M at x_0), then for any $T>0$, the set $R^{\Psi}(x_0,T)$ has a nonempty interior. The previous condition is called the ***Lie Algebra Rank Condition*** (***LARC***). Using (27.3) we can find $\triangle_A$ as follows:

$$\triangle_A=\triangle_3=\text{span}\left\{\begin{matrix} f,g_1,g_2,g_3 \\ g_4=[f,g_1],g_5=[f,g_2],g_6=[f,g_3] \\ g_7=[g_2,[f,g_2]] \end{matrix}\right\}. \tag{27.4}$$

The LARC condition leads to the following condition:

$$-V^2\cos\gamma(V+W_z\sin\gamma+W_y\sin\chi\cos\gamma+W_x\cos\chi\cos\gamma)\neq 0. \tag{27.5}$$

So, either: $V \neq 0$, or: $\gamma \neq \frac{\pi}{2}$, or: $(V + W_z \sin\gamma + W_y \sin\chi \cos\gamma + W_x \cos\chi \cos\gamma) \neq 0$. Thus with the previous condition, the system (27.1) verifies the Lie Algebra rank condition and is locally accessible. Therefore the non-holonomy degree of the system is $K = 3$.

27.3 Time Optimal Control

The subject of this section is formulating the trajectory generation problem in minimum time. Time optimal trajectory generation can be formulated as follows:

$$\min \int_0^T dt \tag{27.6}$$

subject to

$$\begin{aligned} \dot{x} &= V\cos\chi\cos\gamma + W_x, & \dot{\gamma} &= u_1, \\ \dot{y} &= V\sin\chi\cos\gamma + W_y, & \dot{\chi} &= u_2, \\ \dot{z} &= V\sin\gamma + W_z, & \dot{V} &= u_3 \end{aligned} \tag{27.7}$$

with initial and final conditions

$$\begin{aligned} &x(0) = x_0, \quad y(0) = y_0, \quad z(0) = z_0, \quad \chi(0) = \chi_0, \quad \gamma(0) = \gamma_0, \quad V(0) = V_0, \\ &x(T) = x_f, \quad y(T) = y_f, \quad z(T) = z_f, \quad \chi(T) = \chi_f, \quad \gamma(T) = \gamma_f, \quad V(T) = V_f \end{aligned} \tag{27.8}$$

and limitations on the control inputs and states

$$|u_1| \leq u_{1max}, \qquad |u_2| \leq u_{2max}, \qquad |u_3| \leq u_{3max}, \quad |V| \leq V_{max}. \tag{27.9}$$

This formulation is a generalization of Zermelo's navigation problem, where the problem consists of finding the quickest nautical path for a ship at sea in the presence of currents [8]. As this system has bounds on the magnitudes of the inputs and as the set of the allowable inputs is convex, the time optimal paths result from saturating the inputs at all times (or zero for singular control). For points that are reachable, the resolution is based on the Pontryagin Minimum Principle, which constitutes a generalization of the Lagrange problem of the calculus of variations. It is a local reasoning based on the comparison of trajectories corresponding to infinitesimally close control laws. It provides necessary conditions for paths to be optimal. Of course, the kinematic model used below implies a perfect response to the turn commands. A major reason for using the kinematic model is the fact that only necessary conditions for optimality exist for the second order model (given by the Pontryagin minimum principle). The Hamiltonian, formed by adjoining the state equation with the appropriate adjoint variable $\lambda_1, ..., \lambda_6$, is defined as:

$$H = 1+\lambda_1(V\cos\chi\cos\gamma+W_x)+\lambda_2(V\sin\chi\cos\gamma+W_y)+ \\ \lambda_3(V\sin\gamma+W_z)+\lambda_4 u_1+\lambda_5 u_2+\lambda_6 u_3, \tag{27.10}$$

where λ represents the Lagrange multiplier. The optimal control input has to satisfy the following set of necessary conditions:

$$\dot{X} = \frac{\partial H}{\partial \lambda}, \qquad \dot{\lambda} = -\frac{\partial H}{\partial X}. \tag{27.11}$$

The co-state variables are free, i.e. unspecified, at both the initial and final times because the corresponding state variables of the system are specified. The first interesting result is the determination of a sufficient family of trajectories, i.e. a family of trajectories containing the optimal solution for linking any two configurations.

As the drift is not an explicit function of time t, the first integral of the two point boundary value problem exists and thus the hamiltonian H is constant on the optimal trajectory. Because $H(T) = 0$ from the transversality condition, $H(t) = 0, \forall t \in [0,T]$. The co-state equations are then obtained in the standard fashion by differentiating the negative of the Hamiltonian with respect to the states where:

$$\begin{aligned}
&\dot{\lambda_1} = 0, \qquad \dot{\lambda_2} = 0, \qquad \dot{\lambda_3} = 0, \\
&\dot{\lambda_4} = \lambda_1 V\cos\chi\sin\gamma+\lambda_2 V\sin\chi\sin\gamma-\lambda_3 V\cos\gamma \\
&\dot{\lambda_5} = \lambda_1 V\sin\chi\cos\gamma-\lambda_2 V\cos\chi\cos\gamma = \lambda_1(\dot{y}-W_y)-\lambda_2(\dot{x}-W_x) \\
&\dot{\lambda_6} = -\lambda_1\cos\chi\cos\gamma-\lambda_2\sin\chi\cos\gamma-\lambda_3\sin\gamma = -\lambda_1\frac{\dot{x}-w_x}{v}-\lambda_2\frac{\dot{y}-w_y}{v}-\lambda_3\frac{\dot{z}-w_z}{v}.
\end{aligned} \tag{27.12}$$

Defining the Hamiltonian and multiplier dynamics in this way, the minimum principle of Pontryagin states that the control variable has to be chosen to minimize the Hamiltonian at every instant:

$$H(X^*,\lambda,u^*) \le H(X^*,\lambda,u), \tag{27.13}$$

leading to the following solution:

$$u_1^* = \delta_1 u_{1max}, \quad u_2^* = \delta_2 u_{2max} \quad \text{and} \quad u_3^* = \delta_3 u_{3max}, \tag{27.14}$$

with $\delta_i = -\text{sign}(\lambda_{i+3}) \in \{+1,-1\}$ for the regular case and $\delta_i = 0$ for the singular case, where $i = 1,2,3$.

We call this type of control "bang-bang", and we notice that it depends on Lagrange multipliers, so in the following section we are going to analyse these multipliers to show their effect on controls.

27.3.1 Lagrange Multiplier Analysis

We call the multipliers $\lambda_{i+3}(t), i = 1,2,3$, the *switching functions*. Figure 27.1 shows the relation between switching functions and controls. Actually, when $\lambda_{i+3}(t)$ passes

through zero, a switching time of the control $u_i^*(t)$ is indicated. If $\lambda_{i+3}(t)$ is zero for some finite time interval, then the minimal condition provides no information about how to select $u_i^*(t)$, and the control is *singular control*.

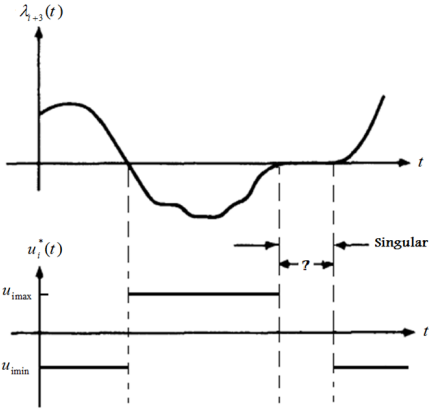


Fig. 27.1 The relationship between a time-optimal control and its corresponding Lagrange multiplier

From the co-state equations (27.12) we notice that λ_1, λ_2, and λ_3 are time-invariant variables. In the other side λ_4, λ_5, and λ_6 are time-variant variables. So, let us calculate the time profiles of these multipliers in the time interval $t \in [t_k, t_{k+1}]$ for regular control input case. By integration of the Eqs. (27.12) we obtain:

$$\begin{aligned}
\lambda_4 &= \left(\frac{\lambda_1 v}{2(\dot{\chi}+\dot{\gamma})} - \frac{\lambda_2 \dot{v}}{2(\dot{\chi}+\dot{\gamma})^2}\right)\sin(\chi+\gamma) + \left(\frac{\lambda_1 \dot{v}}{2(\dot{\chi}+\dot{\gamma})^2} + \frac{\lambda_2 v}{2(\dot{\chi}+\dot{\gamma})}\right)\cos(\chi+\gamma) + \\
&+ \left(\frac{\lambda_1 v}{2(\dot{\chi}-\dot{\gamma})} - \frac{\lambda_2 \dot{v}}{2(\dot{\chi}-\dot{\gamma})^2}\right)\sin(\chi-\gamma) + \left(\frac{\lambda_1 \dot{v}}{2(\dot{\chi}-\dot{\gamma})^2} + \frac{\lambda_2 v}{2(\dot{\chi}-\dot{\gamma})}\right)\cos(\chi-\gamma) - \\
&- \lambda_3\left(\frac{v\sin\chi}{\dot{\chi}} + \frac{\dot{v}\cos\chi}{\dot{\chi}^2}\right) + \lambda_4', \\
\lambda_5 &= \lambda_1(y-y_k) - \lambda_2(x-x_k) + (\lambda_2 w_x - \lambda_1 w_y)(t-t_k) + \lambda_{5k} \qquad (27.15) \\
\lambda_6 &= \lambda_{6k} + \frac{-\lambda_1(\sin(\chi-\gamma) - \sin(\chi_k-\gamma_k)) + \lambda_2(\cos(\chi-\gamma) - \cos(\chi_k-\gamma_k))}{2(\dot{\chi}-\dot{\gamma})} + \\
&+ \frac{-\lambda_1(\sin(\chi+\gamma) - \sin(\chi_k+\gamma_k)) + \lambda_2(\cos(\chi+\gamma) - \cos(\chi_k+\gamma_k))}{2(\dot{\chi}+\dot{\gamma})} + \frac{\lambda_3(\cos\gamma - \cos\gamma_k)}{\dot{\gamma}},
\end{aligned}$$

where:

$$\lambda_4' = \lambda_{4k} - \left(\frac{\lambda_1 v_k}{2(\dot{\chi}+\dot{\gamma})} - \frac{\lambda_2 \dot{v}}{2(\dot{\chi}+\dot{\gamma})^2}\right)\sin(\chi_k+\gamma_k) - \left(\frac{\lambda_1 \dot{v}}{2(\dot{\chi}+\dot{\gamma})^2} + \frac{\lambda_2 v_k}{2(\dot{\chi}+\dot{\gamma})}\right)\cos(\chi_k+\gamma_k) -$$
$$- \left(\frac{\lambda_1 v_k}{2(\dot{\chi}-\dot{\gamma})} - \frac{\lambda_2 \dot{v}}{2(\dot{\chi}-\dot{\gamma})^2}\right)\sin(\chi_k-\gamma_k) - \left(\frac{\lambda_1 \dot{v}}{2(\dot{\chi}-\dot{\gamma})^2} + \frac{\lambda_2 v_k}{2(\dot{\chi}-\dot{\gamma})}\right)\cos(\chi_k-\gamma_k) +$$
$$+ \lambda_3 \left(\frac{v_k \sin\chi_k}{\dot{\chi}} + \frac{\dot{v}\cos\chi_k}{\dot{\chi}^2}\right),$$
$$\dot{\gamma} = \delta_1 U_{1max}, \qquad \dot{\chi} = \delta_2 U_{2max}, \qquad \dot{v} = \delta_3 U_{3max}$$

and γ_k, χ_k, v_k are the initial values at $t = t_k$.

Let us now define the cases when singularity on controls occurs.

Singularity on u_1: In this case we have $\lambda_4 = \dot{\lambda}_4 = 0$ and $\gamma = \gamma_k$. So, by substituting the value of γ in the equation $\dot{\lambda}_4 = 0$ we find that the singularity occurs when: $v = 0$ or $\tan\gamma_k = \frac{\lambda_3}{\lambda_1\cos\chi + \lambda_2\sin\chi}$.

Singularity on u_2: In this case we have $\lambda_5 = \dot{\lambda}_5 = 0$ and $\chi = \chi_k$. So, from the equation $\dot{\lambda}_5 = 0$ we find that the singularity occurs when: $y = \frac{\lambda_2}{\lambda_1}x + (\lambda_1 w_y - \lambda_2 w_x)t + \lambda_{5c}$, where λ_{5c} is the integral constant. The previous equation presents a straight line in the plane if there is no wind.

Singularity on u_3: In this case we have $\lambda_6 = \dot{\lambda}_6 = 0$ and $v = v_k$. So, form the equation $\dot{\lambda}_6 = 0$ we find that the singularity occurs when: $z = -\frac{\lambda_1}{\lambda_3}x - \frac{\lambda_2}{\lambda_3}y + (\lambda_1 w_x + \lambda_2 w_y + \lambda_3 w_z)t + \lambda_{6c}$, where λ_{6c} is the integral constant. The previous equation presents a straight line in the space if there is no wind. In addition, as we have a limitation on velocity, we add the condition $|v| = v_{max}$ as an additional case when a singularity on u_3 must occur.

The remaining problem is to find the initial values of the Lagrange multipliers such that the two-point boundary value problem formulated by the Pontryagin minimum principle is solved, i.e. finding the initial values of the Lagrange multipliers that ensure that the correspond switching times allow steering the system (i.e. Eqs. [27.7] and [27.12]) from the given initial point to the desired final point.

27.3.2 Optimal Path Analysis

In this paragraph we are going to calculate the time profiles of x, y, z, γ, χ, and v for a time interval $t \in [t_k \quad t_{k+1}]$ under two types of control inputs, i.e. regular ($\delta_i U_{imax} \in \{+1, -1\}$) and singular ($\delta_i U_{imax} = 0$) control. We give only the final result of the analytical integrations:

- ***Regular arcs:***
  For regular control inputs, we have 8 different arcs shown in Table 27.1. So, the general arc of regular control is given as follows:

$$x = x_k + \frac{v\sin(\chi+\gamma) - v_k\sin(\chi_k+\gamma_k)}{2(\delta_2 U_{2max} + \delta_1 U_{1max})} + \frac{\delta_3 U_{3max}(\cos(\chi+\gamma) - \cos(\chi_k+\gamma_k))}{2(\delta_2 U_{2max} + \delta_1 U_{1max})^2} + \tag{27.16}$$

$$+ \frac{v\sin(\chi-\gamma) - v_k\sin(\chi_k-\gamma_k)}{2(\delta_2 U_{2max} - \delta_1 U_{1max})} + \frac{\delta_3 U_{3max}(\cos(\chi-\gamma) - \cos(\chi_k-\gamma_k))}{2(\delta_2 U_{2max} - \delta_1 U_{1max})^2} + w_x(t-t_k),$$

$$y = y_k - \frac{v\cos(\chi+\gamma) - v_k\cos(\chi_k+\gamma_k)}{2(\delta_2 U_{2max} + \delta_1 U_{1max})} + \frac{\delta_3 U_{3max}(\sin(\chi+\gamma) - \sin(\chi_k+\gamma_k))}{2(\delta_2 U_{2max} + \delta_1 U_{1max})^2} - \tag{27.17}$$

$$- \frac{v\cos(\chi-\gamma) - v_k\cos(\chi_k-\gamma_k)}{2(\delta_2 U_{2max} - \delta_1 U_{1max})} + \frac{\delta_3 U_{3max}(\sin(\chi-\gamma) - \sin(\chi_k-\gamma_k))}{2(\delta_2 U_{2max} - \delta_1 U_{1max})^2} + w_y(t-t_k),$$

$$z = z_k - \frac{v\cos\gamma - v_k\cos\gamma_k}{\delta_1 U_{1max}} + \frac{\dot{v}(\sin\gamma - \sin\gamma_k)}{U_{1max}^2} + w_z(t-t_k), \tag{27.18}$$

$$\gamma = \delta_1 U_{1max}(t-t_k) + \gamma_k, \tag{27.19}$$

$$\chi = \delta_2 U_{2max}(t-t_k) + \chi_k, \tag{27.20}$$

$$v = \delta_3 U_{3max}(t-t_k) + v_k. \tag{27.21}$$

- ***Singular arcs:***
  In this case we have 7 different arcs shown in Table 27.2. The following equations express the first type of singular arcs, i.e. $u_1 = u_2 = u_3 = 0$:

$$x = (v_k\cos\chi_k\cos\gamma_k + w_x)(t-t_k) + x_k,$$
$$y = (v_k\sin\chi_k\cos\gamma_k + w_y)(t-t_k) + y_k,$$
$$z = (v_k\sin\gamma_k + w_z)(t-t_k) + z_k, \tag{27.22}$$
$$\gamma = \gamma_k, \qquad \chi = \chi_k, \qquad v = v_k. \tag{27.23}$$

  The other types of singular arcs can be obtained by integrating the Eqs. (27.7) after replacing the controls by their values.

In the next section we are going to show some numerical results.

Table 27.1 The different arc types of regular control

	u_1	u_2	u_3
Reg. type 1	$+U_{1max}$	$+U_{2max}$	$+U_{3max}$
Reg. type 2	$+U_{1max}$	$+U_{2max}$	$-U_{3max}$
Reg. type 3	$+U_{1max}$	$-U_{2max}$	$+U_{3max}$
Reg. type 4	$+U_{1max}$	$-U_{2max}$	$-U_{3max}$
Reg. type 5	$-U_{1max}$	$+U_{2max}$	$+U_{3max}$
Reg. type 6	$-U_{1max}$	$+U_{2max}$	$-U_{3max}$
Reg. type 7	$-U_{1max}$	$-U_{2max}$	$+U_{3max}$
Reg. type 8	$-U_{1max}$	$-U_{2max}$	$-U_{3max}$

Table 27.2 The different arc types of singular control

	u_1	u_2	u_3
Sing. type 1	0	0	0
Sing. type 2	0	0	$\pm U_{3max}$
Sing. type 3	0	$\pm U_{2max}$	0
Sing. type 4	0	$\pm U_{2max}$	$\pm U_{3max}$
Sing. type 5	$\pm U_{1max}$	0	0
Sing. type 6	$\pm U_{1max}$	0	$\pm U_{3max}$
Sing. type 7	$\pm U_{1max}$	$\pm U_{2max}$	0

27.4 Numerical Solution

The numerical solution of the optimal control problem can be categorized into two main approaches. The first approach corresponds to direct methods, which are based on discretization of state and control variables over time, so that we transform the problem to a nonlinear optimization problem (called also NonLinear Programming, NLP) which can be solved by an NLP solver as the fmincon solver in Matlab®. The second approach corresponds to indirect methods, where a Two-Point Boundary Value Problem (TPBVP) is solved numerically. So, the first step of this method is to formulate an appropriate TPBVP using the Pontryagin Minimum Principle and the second step is to solve it numerically with a shooting method for example.

In this paper, we are going to use a direct method notably the fmincon solver in Matlab®. We choose the values $u_1 = 0.26$ [rad/sec] $\simeq 15$ [deg/sec], $u_2 = 0.52$ [rad/sec] $\simeq 30$ [deg/sec], $u_3 = 1.25$ [m/sec^2] $= 0.125g$ and $v_{max} = 4.286$ [m/sec] $\simeq 15$[km/h] and the initial point $x_0 = y_0 = z_0 = 0$, $\gamma_0 = 0.25$ [rad/sec], $\chi_0 = 0.5$ [rad/sec], and $v_0 = 2$ [m/sec] $\simeq 7$ [km/h].

- ***Case 1:*** 3D trajectory with only regular arcs and without wind:
  Using a final point sufficiently close to the initial point we obtain the results shown in Figs. 27.2 and 27.3. We choose $x_f = 8$ [m], $y_f = 7$ [m], $z_f = 10$ [m], $\gamma_f = \chi_f = 0$, and $v_f = 0.5$ [m/sec].

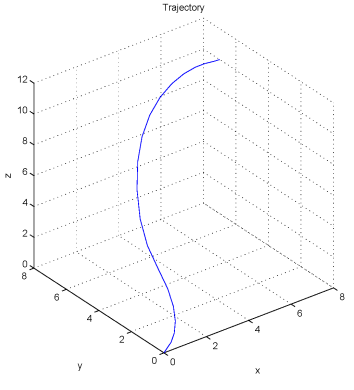


Fig. 27.2 3D trajectory between two configurations in the space sufficiently close

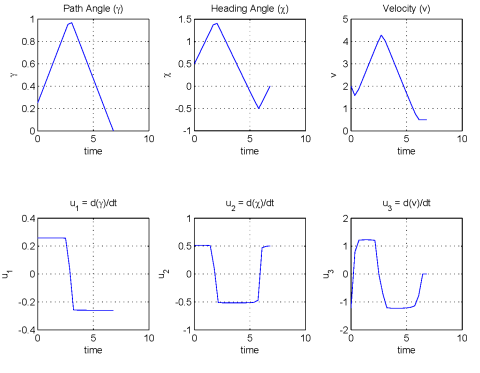


Fig. 27.3 Optimal controls and time profiles for γ, χ, and v with only regular arcs

- ***Case 2:*** 3D trajectory with regular and singular arcs and without wind:
  Using a final point sufficiently far to the initial point we obtain the results shown in Figs. 27.4 and 27.5. We choose $x_f = 80$ [m], $y_f = 70$ [m], $z_f = 100$ [m], $\gamma_f = \chi_f = 0$ and $v_f = 0.5$ [m/sec].

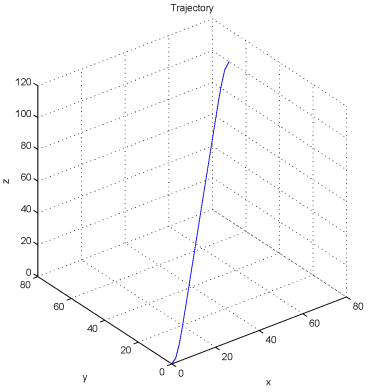


Fig. 27.4 3D trajectory between two configurations in the space sufficiently far

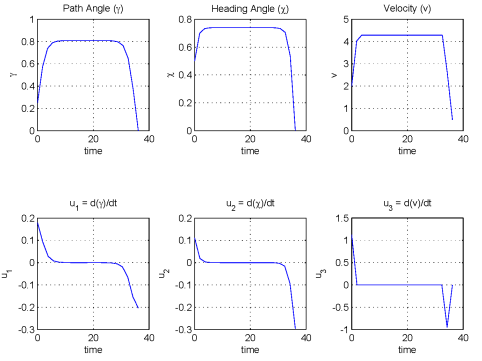


Fig. 27.5 Optimal controls and time profiles for γ, χ, and v with singular and regular arcs

- ***Case 3:*** 3D trajectory taking into account the wind speed:
  Using the same final point as the first case and supposing that we have a wind component according to x axis i.e. $w_x = 0.1$ [m/sec] we obtain the results shown in Figs. 27.6 and 27.7, where the solid lines denote the case without wind and the triangle lines denote the case with wind.

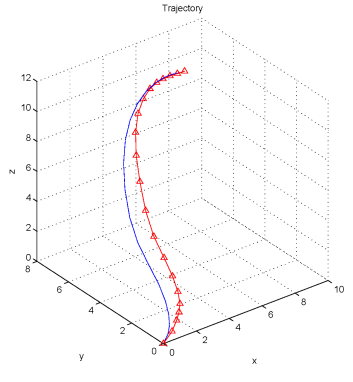


Fig. 27.6 3D trajectory taking into account the wind speed according to x axis

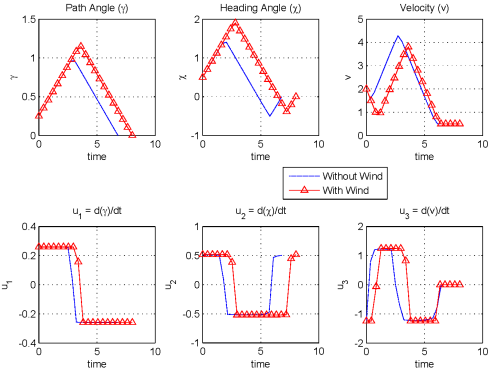


Fig. 27.7 Optimal controls and time profiles for γ, χ, and v taking into account the wind speed according to x axis

We can link two configurations in the space sufficiently close by applying a bang-bang control without singularity. In the other side, if both configurations are sufficiently far then a singularity on control occurs. The wind provides an important effect on the trajectory which has to be taken in account in planning. Figures 27.6 and 27.7 show this effect on the trajectory and controls. We notice that the airship

needs more time to reach the goal point and from a geometrical view, we can say that we have a generalization of Dubins curves.

27.5 Conclusions

This paper presents an analysis of the time optimal trajectories of an airship considering a piecewise constant acceleration. Geometric characterization of the candidate paths satisfying the necessary conditions for time optimality is presented. An obvious generalization of this work is to include dynamics. It will help into energy savings as the wind variations are used as inputs in the trajectory generation for the vehicle motion. Another motivation for determining these elementary pieces is for use as motion primitives in planning and control algorithms that consider obstacles.

References

1. Etkin, B.: Turbulent wind and its effect on flight. J. of Aircraft 18, 327–345 (1981)
2. Bestaoui, Y., Kahale, E.: Analysis of Time Optimal 3D Paths for an Autonomous Aircraft with a Piecewise Constant Acceleration. In: 48th AIAA Aerospace Sciences Meeting, Orlando, Florida, paper AIAA-1352 (2010)
3. Dubins, L.E.: On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents. American J. of Mathematics 79, 497–517 (1957)
4. Boukraa, D., Bestaoui, Y., Azouz, N.: Three Dimensional Trajectory Generation for an Autonomous Plane. Inter. Review of Aerospace Eng. 4, 355–365 (2008)
5. McGee, T., Hedrick, J.K.: Optimal path planning with a kinematic airplane model. AIAA J. of Guidance, Control and Dynamics, 30 (2007)
6. Nelson, R., Barber, B., McLain, T., Beard, R.: Vector Field Path Following for Miniature Air Vehicle. IEEE Trans. on Robotics 23, 519–529 (2007)
7. Rysdyk, R.: Course and heading changes in significant wind. AIAA J. of Guidance, Control and Dynamics 30, 1168–1171 (2007)
8. Jurdjevic, V.: Geometric control theory. Cambridge Studies in Advanced Mathematics (2008)
9. Sussmann, H.J.: Shortest 3-dimensional paths with a prescribed curvature bound. In: 34th IEEE Conf. on Decision and Control, pp. 3306–3312 (1995)

Chapter 28
Formation Flight Control Scheme for Unmanned Aerial Vehicles

Zdzisław Gosiewski and Leszek Ambroziak

Abstract. Cooperative control of Unmanned Aerial Vehicles is currently being researched for a wide range of applications. Applicability of aerial unmanned systems might be increased by formation flight. In this paper, we present an application of a proportional-integral controller for formation flight of three Unmanned Aerial Vehicles (UAVs). A decentralized cooperative control scheme is used with information flow modeled by a leader-follower structure. Control laws, based on velocity and position errors between the vehicles, are defined and derived for each of the three flying objects in formation. The main goal of this work is a simulation study. Three-dimensional motion of the three-object formation was performed and analyzed using a six-degrees-of-freedom state space quadrotor model. The simulation studies presented allow verifying the adopted control structure. Convergence time is used as an indicator of the control system quality and the formation stability. The presented research and diagram of control algorithms tests were done before experimental tests and field trials of formation flight.

28.1 Introduction

Formation flight is defined as intended motion of two or more flying objects, connected by a common control law [1]. Formation flight is a special case of UAV group flight, which is focused on achieving and maintaining a particular structure (shape) of the entire formation, appropriate speed keeping at flight through individual objects in the formation, distance keeping by neighboring vehicles and collision avoidance. To meet all these criteria, the process of designing the control laws needs to be supported by numerous studies of simulation [2, 3]. Formation flying, and problems associated with it are a relatively young field of science and a new theme

Zdzisław Gosiewski · Leszek Ambroziak
Department on Automatics and Robotics, Białystok University of Technology,
ul. Wiejska 45 c, 15-351 Białystok, Poland
e-mail: gosiewski@pb.bialystok.pl, leszek.ambroziak@gmail.com

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 331–340.
springerlink.com

inspired by biology. In 1977 Scholomitsky, Prilutsky, and Rodnin [5] worked on the concept of an infrared interferometer for a few flying objects. This date, according to the literature review related to this subject, is considered the beginning of work on formation flights. These studies were used and continued in the early eighties by Lyberie, Samarie, Schumacher, Stachnik, and Gezari [5], for formation flight of spacecrafts, their refueling and maintenance. In the mid-eighties and early nineties researchers began to focus on the use of formation flight for airplanes [8] and automatic refueling in the air. That was the beginning for research in the nineties and now on flights and flights within a group and formation of unmanned flying objects. There are many works which concern aircraft formation flight strategies such as fuzzy logic control [6], adaptive control [9], or robust control [7], but they are considered only theoretically. Practical formation flight tests are also taken [10, 11, 12], but they are always based on analytical and simulation research. In this paper we present an application of a proportional-integral controller to formation flight of three Unmanned Aerial Vehicles (UAVs). We adopted the leader-follower type pattern of communication between vehicles. Decentralized control laws were used and performed for each one of three UAVs. To check the adopted control system, simulations with six-degrees-of-freedom state space UAV model are presented.

28.2 Vehicle Model Description

In this paper we consider a formation of three identical flying objects which can communicate with one another to achieve the main goal – formation keeping. Every object in the formation has two main control loops. The first feedback control loop (internal) stabilizes the objects locally. The second feedback control loop (external) is a "formation-level" controller and is responsible for formation achieving. For this reason, we would like to present dynamics of the control objects and their local controllers first. We have chosen quadrotors as the control objects. They are vertical take-off and landing (VTOL) objects with very specific dynamics. Quadrotors are often used as a platform for formation flying control framework [4]. They can hover above the desired waypoint and can change the motion direction in place. It is simple to separate navigation from orientation in quadrotor motion. A simplified dynamics model of these objects is described in the next subsection.

28.2.1 Quadrotor Model Dynamics Used in Simulations

The dynamics of a single quadrotor is shown in equations (28.1)–(28.6). The quadrotor dynamical model is obtained from the Euler-Lagrange Equations with external force simplified based on [4]. For the first stage of simulations and to verify the control law quality this dynamics model is sufficient:

$$m\ddot{x} = -u\sin(\theta), \tag{28.1}$$
$$m\ddot{y} = u\cos(\theta)\sin(\varphi), \tag{28.2}$$
$$m\ddot{z} = u\cos(\theta)\cos(\varphi) + mg, \tag{28.3}$$
$$\ddot{\psi} = S_{\psi}, \tag{28.4}$$
$$\ddot{\theta} = S_{\theta}, \tag{28.5}$$
$$\ddot{\varphi} = S_{\varphi}, \tag{28.6}$$

where x and y describe the position of the quadrotor on the horizontal plane, z is the vertical axis, φ is the angle measured around the x axis ("roll"), θ is the angle measured around the y axis ("pitch"), and ψ is the angle measured around the z axis ("yaw"). Moreover, $u = \sum_{i=1}^{4} F_i$ (where F_i – force produced by the i-th motor) is the thrust applied at the center of gravity directed outwards from the bottom of the aircraft, S_{ψ}, S_{θ}, S_{φ} are the angular moments (yawing moment, pitching moment, and rolling moment), $m = 0.88\,\text{kg}$ is the mass of a quadrotor, and $g = 9.81\,\frac{\text{m}}{\text{s}^2}$ is the gravitational acceleration (see Fig. 28.1).

Based on [4], a state space model with 4 inputs and 12 states of the quadrotor is presented below:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \tag{28.7}$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}, \tag{28.8}$$

where

$$\mathbf{A} = \begin{bmatrix} 0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&-g&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&g&0 \\ 0&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \\ \frac{1}{m}&0&0&0 \\ 0&0&0&0 \\ 0&1&0&0 \\ 0&0&0&0 \\ 0&0&1&0 \\ 0&0&0&0 \\ 0&0&0&1 \end{bmatrix}.$$

The state vector looks like:

$$\mathbf{x} = \begin{bmatrix} x\ \dot{x}\ y\ \dot{y}\ z\ \dot{z}\ \psi\ \dot{\psi}\ \theta\ \dot{\theta}\ \varphi\ \dot{\varphi} \end{bmatrix}^T$$

and the input vector $\mathbf{u}$ is given by:

$$\mathbf{u} = \begin{bmatrix} u_1\ u_2\ u_3\ u_4 \end{bmatrix}^T = \begin{bmatrix} u - mg\ S_{\psi}\ S_{\theta}\ S_{\varphi} \end{bmatrix}^T.$$

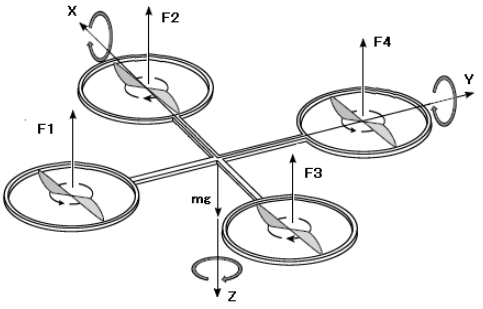


Fig. 28.1 Scheme of four-rotor rotorcraft

28.2.2 Quadrotor Local Controller Development

In order to stabilize system dynamics, a LQR (Linear-Quadratic-Regulator) controller was developed and implemented to ensure local stability for one object. The LQR approach is often used and mentioned to stabilize the four-rotor rotorcraft, but we decided to use this solution because local stability and good quality of quadrotor control will determine stability of a formation flight mission.

The LQR controller uses measurements $\mathbf{y}$ of the object (quadrotor) to generate a control signal $\mathbf{u}$ that controls $\mathbf{y}$. The LQR regulator minimizes the cost function:

$$J(\mathbf{u}) = \int_0^\infty (\mathbf{y}^T \mathbf{Q}\mathbf{y} + \mathbf{u}^T \mathbf{R}\mathbf{u})dt. \tag{28.9}$$

The state control law can be written as follows:

$$\mathbf{u} = -\mathbf{K}\mathbf{x}. \tag{28.10}$$

The closed loop system can be determined with (28.10) as:

$$\dot{\mathbf{x}} = [\mathbf{A} - \mathbf{B}\mathbf{K}]\mathbf{x}. \tag{28.11}$$

In addition to the state-feedback gain $\mathbf{K}$, the LQR returns the solution $\mathbf{S}$ of the associated Riccati equation:

$$\mathbf{A}^T\mathbf{S} + \mathbf{S}\mathbf{A} - (\mathbf{S}\mathbf{B} + \mathbf{N})\mathbf{R}^{-1}(\mathbf{B}^T\mathbf{S} + \mathbf{N}^T) + \mathbf{Q} = 0. \tag{28.12}$$

$\mathbf{K}$ is derived from $\mathbf{S}$ using:

$$\mathbf{K} = \mathbf{R}^{-1}(\mathbf{B}^T\mathbf{S} + \mathbf{N}^T). \tag{28.13}$$

For the LQR controller the Matlab Control Toolbox software was used with the control system toolbox. Matrix $\mathbf{Q}$ contains the restrictions put on state vector $\mathbf{x}$, matrix $\mathbf{R}$ contains the effort signal scales (the first simulations and matrix $\mathbf{Q}$ selection

were done for $\mathbf{R} = \mathrm{diag}(1, 1, 1, 1)$). Finally, the weight matrices are chosen so as to meet the control performances denoted by (28.9) and are equal to:

$$\mathbf{R} = \mathrm{diag}(50, 0.05, 20, 20), \tag{28.14}$$

$$\mathbf{Q} = \mathrm{diag}(0.02, 1, 0.2, 1, 0.05, 20, 0.25, 1, 1000, 25, 1000, 25). \tag{28.15}$$

The derived matrix $\mathbf{K} = [\mathbf{K_1} \quad \mathbf{K_2}]$ is described as:

$$\mathbf{K_1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0.1 & 0.7509 \\ -0.0001 & 0.8165 & 0 & -0.0006 & 0 & 0 \\ -0.0312 & 1.9480 & -0.0050 & -0.0556 & 0 & 0 \\ -0.0050 & 3.4012 & 0.0312 & 0.3213 & 0 & 0 \end{bmatrix},$$

$$\mathbf{K_2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 2.2361 & 4.9469 & -1.4263 & -0.1564 & -0.0225 & 0.0254 \\ 0 & -0.0004 & 10.0323 & 4.4196 & -1.8033 & -1.3349 \\ 0 & -0.0001 & -9.5328 & -1.3349 & 1.3349 & -4.7227 \end{bmatrix}.$$

To verify closed-loop stability of a single object, simulation investigations were done and are shown in Fig. 28.2.

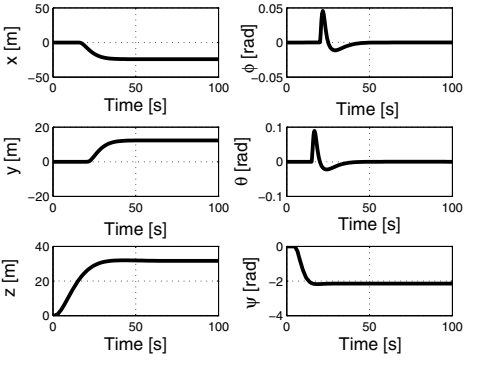


Fig. 28.2 Quadrotor control quality

28.3 Formation Structure Specification

As a communication structure of the formation we have chosen the leader-follower architecture. The leader-follower structure is the most commonly encountered in formation flight of UAVs. Leader-follower is often referred to as chief-deputy, target-chase, or master-slave. It is a hierarchical structure in which control of the objects with a lower status and position in the formation hierarchy comes very often to the tracking process. The leader is called the object running the mission, having

information about the reference trajectory. The follower is an object tracking the trajectory of the leader or moving along the guidelines designated by the leader. The leader-follower structure is often used because its reconfiguration of the formation shape is easy, and it is simple to expand formation of new objects. The structure is resistant to communication errors between the objects and is characterized by high reliability in case of loss of an object (its functions can be quickly taken over by other vehicles) [1]. In our investigations of formation flight we have chosen a leader-follower communication structure for three vehicles with one leader (a single leader-follower system) and two follower objects. The analyzed structure of formation is shown in Fig. 28.3, where three UAVs denoted UAV1, UAV2, and UAV3 can be seen. Communication flow between UAVs is shown by arrows. UAV1 is the leader of the whole formation. It has information about the reference trajectory and sends the information to the other objects. UAV2 and UAV3 receive the information from UAV1. The presented shape of the quadrotor formation is also popularly called a 'V' shape formation.

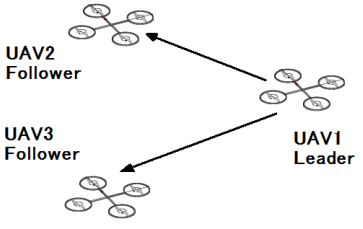


Fig. 28.3 Leader-follower structure

28.4 Control Law Development with PI Controller

The process of designing the control law is focused on the internal stability of the entire formation, which means the ability to achieve and maintain the desired structure and shape of the UAV formation. The control laws, based on the velocity and position errors between the vehicles in the x direction, are defined as follows [1, 3]:

$$\left.\begin{aligned} e_x &= x_r - x_1 - b_{1sep}, \\ \dot{e}_x &= \dot{x}_r - \dot{x}_1, \end{aligned}\right\} \quad UAV1, \tag{28.16}$$

$$\left.\begin{aligned} e_x &= x_1 - x_2 - b_{2sep}, \\ \dot{e}_x &= \dot{x}_1 - \dot{x}_2, \end{aligned}\right\} \quad UAV2, \tag{28.17}$$

$$\left.\begin{aligned} e_x &= x_2 - x_3 - b_{3sep}, \\ \dot{e}_x &= \dot{x}_2 - \dot{x}_3, \end{aligned}\right\} \quad UAV3, \tag{28.18}$$

where e_x is the position error in the x direction, x_r is the reference trajectory information, b is a constant distance between the vehicles. Vehicle UAV1, which is the leader of the whole formation, tracks reference trajectory x_r and stays in relation to this trajectory with some constant separation distance b_{1sep}. These relations are the same for UAV2 and UAV3 and for the y and z directions.

Control inputs u_{ext} (external loop, formation level control) of the three vehicles have to stabilize the error dynamics of the formation with three objects and for position the effort signal can be written in a general form as follows:

$$u_{ext}(t) = K_p e(t) + K_i \int_0^t e(t)dt, \tag{28.19}$$

where K_p is a proportional gain, K_i is an integral gain, and $e(t)$ is a position error signal. Control law (28.19) for velocity looks similarly.

28.5 Simulation Results

The main goal of this work is a simulations study. Simulations were run with Matlab/Simulink software. We ran simulations with three quadrotors flying in formation in a 'V' shape. To stabilize the position and velocity errors between the vehicles in the x, y, and z directions a PI controller described in (28.19) with a linear mixer was developed and implemented (Fig. 28.4) using the equations presented in Section 28.3. The linear mixer calculates the errors between the position and the velocity, reference and actual, during the flight also taking into account the desired separation commands between the vehicles. The errors calculated in the linear mixer block are a signal input to the formation PI controller. The gains for proportional and integral actions are chosen experimentally to provide as short as possible convergence time and non oscillatory character of the system response.

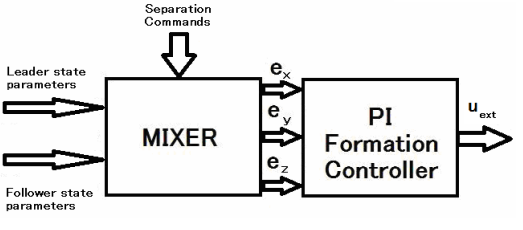


Fig. 28.4 Formation controller

Improper selection of the parameters results in a large overshoot (the desired distances between the vehicles are not maintained). The system is highly sensitive

to changing the controller parameters. For position PI controller gains are shown in Table 28.1.

Table 28.1 Controller gains

Gain	Value
K_{xp}	0.02
K_{xi}	0.015
K_{yp}	0.91
K_{yi}	0.009
K_{zp}	0.77
K_{zi}	0.015

In our simulations we investigated the case of reference trajectory tracking. The first vehicle in formation (the leader) follows a constant velocity reference trajectory. The altitude is constant after the quadrotors take off. Simulation results of the three-quadrotor formation flight and trajectory tracking are presented in Figs. 28.5–28.7.

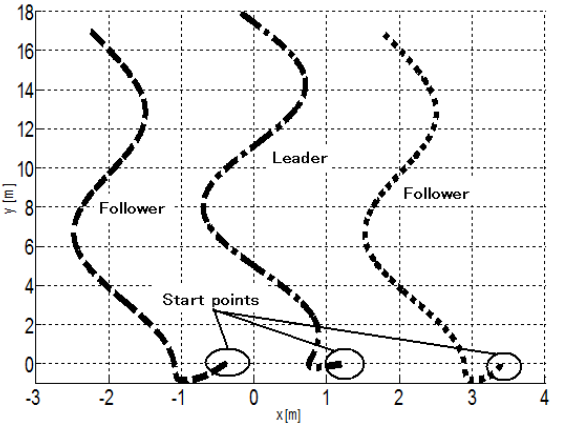


Fig. 28.5 Quadrotors trajectory tracking (*x*-*y* view)

Figure 28.5 shows trajectory tracking of the three quadrotors. The leader of the group has information about the reference trajectory. The objects start flying from the start points marked by a circle. The center line is the leader's trajectory, the left- and right-hand lines describe the followers' motion. The objects try to follow the leader's trajectory fairly well.

The changes of the x direction with time in Fig. 28.6 show the inaccuracies in tracking the leader's trajectory. The smallest inaccuracy occurrs in the z direction. However, influence of the specific quadrotor dynamics on trajectory tracking is noticeable.

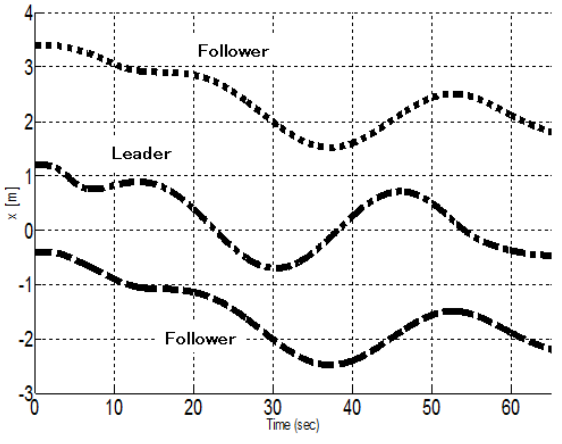


Fig. 28.6 Changes in x direction while trajectory tracking

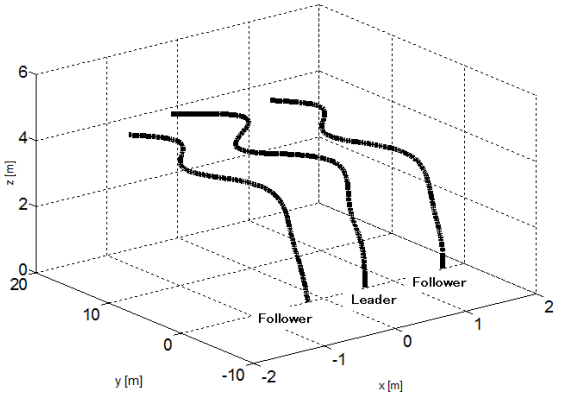


Fig. 28.7 3D quadrotors formation flight with trajectory tracking

28.6 Conclusions

This paper focuses on formation flight control with a leader-follower communication structure designed for three UAVs. The control laws used require information about the position and velocity of the leader and the reference trajectory. A PI quadrotor formation flight controller was developed, analyzed, and presented. Simulation results for a six-degrees-of-freedom UAV model are presented to demonstrate formation convergence and control laws quality. Full state measurement control laws with the PI formation controller give good results and demonstrate applicability of the proposed formation control laws. The control laws adopted are very sensitive to parameter changes.

The next step of our research will be a design of a very accurate model of the quadrotor, including motor transfer functions, measuring device models and disturbances, and studies of this simulation model. Field trials of formation flight will also be performed.

Acknowledgements. The research work is financed by the Polish Ministry of Science and Higher Education as part of the ordered research project No 0059/R/T00/2008/06 conducted in the years 2008-2010.

References

1. Weitz, L.A., Hurtado, J.E., Sinclair, A.J.: Decentralized Cooperative-Control Design for Multivehicle Formations. Journal of Guidance, Control and Dynamics 31(4) (2008)
2. Li, C.: Decentralized Cooperative Control for Multivehicle Without Velocity Measurement. In: 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference, Shanghai, P.R. China (2009)
3. Stipanowic, D.M., Inalhan, G., Teo, R., Tomlin, C.J.: Decentralized Overlapping Control of Formation of Unmanned Aerial Vehicles. Elsevier Journal, Automatica 40, 1285–1296 (2004)
4. Pilz, U., Popov, A.P., Werner, H.: Robust Controller Design for Formation Flight of Quad-Rotor Helicopter. In: 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference, Shanghai, P. R. China (2009)
5. Scharaf, D.P., Hadaeg, F.Y., Ploen, S.R.: A Survey of Spacecraft Formation Flying Guidance and Control (Part II): Control. In: Proceedings of the 2004 American Control Conference, Boston (2004)
6. Li, Y., Li, B., Sun, Z., Weng, L., Zhang, R., Song, D.Y.: Close Formation Flight Control of Multi-UAVs via Fuzzy Logic Technique. In: Advanced Fuzzy Logic Technologies in Industrial Applications, Advances in Industrial Control, pp. 237–247 (2006)
7. Song, Y.D., Li, Y., Bikdash, M., Dong, T.: Cooperative Control of Multiple UAVs in Close Formation Flight via Smooth Robust and Adaptive Approach. In: 4th International Conference on Cooperative Conference Control and Optimization, pp. 759–765 (2003)
8. Buzogany, L.E., Pachter, M., D'Azzo, J.J.: Automated Control of Aircraft in Formation Flight. In: AIAA Guidance, Navigation and Control Conference, pp. 1349–1370 (1993)
9. Boskowic, J.D., Mehra, R.K.: An adaptive reconfigurable formation flight control design. In: American Control Conference, pp. 284 - 289 (2003)
10. Seanor, B., Gu, Z., Neapolitano, M.R., Campa, G., Gururajan, S., Rowe, L.: 3-Aircraft Formation Flight Experiment. In: Control and Automation, pp. 1–6 (2006)
11. Li, N.H., Liu, H.H.: Multiple UAVs Formation Flight Experiments Using Virtual Structure and Motion Synchronization. In: AIAA Guidance, Navigation and Control Conference, Chicago (2009)
12. Li, N.H.M., Liu, H.H.T., Earon, E.J.P., Fulford, C.D., Huq, R., Rabbath, C.A.: Multiple UAVs Autonomous Mission Implementation on COTS Autopilots and Experimental Results. In: AIAA Guidance, Navigation and Control Conference, Chicago (2009)

Chapter 29
Comparison of Two- and Four-Engine Propulsion Structures of Airship

Wojciech Adamski and Przemysław Herman

Abstract. The structure of the propulsion system is an important element of every mobile robot. It has a great impact on controllability and possibility to fight against disturbances. In the case of airships, atmospheric conditions can significantly act on them, and are very difficult (sometimes totally impossible) to predict. In this paper two propulsion structures are described and compared to each other in simulations: first, a two-engine structure on one rotating axis, slightly modified by moving drives away from local plane XZ, to make turning by differential control more effective; second, a four-engine structure with two rotating axes, which gives possibility to better control the pitch angle. A mathematical model and a control algorithm of an airship are also described.

29.1 Introduction

Autonomous airships are objects which use static force: buoyancy to stay in the air. Thanks to it they can carry load with theoretically no costs. In practice control of airships including fight against very important influence of surrounding conditions is not costless, but still energetically cheaper than using aerodynes like airplanes or helicopters. The structure of the propulsion system as part of the control system is an important element of airship design, and is not well analyzed in the literature. In this paper two propulsion structures are briefly described in Sections 29.2 and 29.3, and compared to each other in simulations in Sections 29.6 and 29.7. A mathematical model and a control algorithm of an airship are briefly described in Sections 29.4 and 29.5. More detailed information about these subjects can be found in e.g. [2], [3], [5], [6], [9], [12], [13], and [14].

Wojciech Adamski · Przemysław Herman
Chair of Control and Systems Engineering, Poznań University of Technology,
ul. Piotrowo 3a, 60-965 Poznań, Poland
e-mail: {wojciech.adamski,przemyslaw.herman}@put.poznan.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 341–350.
springerlink.com

29.2 Two-Engine Structure

The classic structure of the airship propulsion consists of two engines located near the gondola and one placed on the tail of the airship. The two engines located at the gondola are used to give speeds in the XZ plane, while the rear engine is used to generate force moment about the Z axis. The authors had at their disposal a model developed for the AS500 airship (by Airspeed Airships), but they decided to modify the structure of the drives, to allow differential control and get rid of the rear engine. Differential control was achieved by moving the drives mounted at the gondola 1 metre away from the XZ plane. It should be noted that the aileron was blocked during the experiments since the authors' aim was to provide control at low speeds, at which aerodynamic control is much less effective than control actuators. Control at low speeds is an important task, because the airships move usually at a low speed during the most difficult maneuvers such as landing, takeoff, or moving at low altitudes in an environment with obstacles.

The impact of the propulsion of the individual engines on the blimp is described by the equations [10]:

$$\left(\Theta_N^R\right)^T \cdot \mathbf{f}_R + \left(\Theta_N^{\mathrm{L}}\right)^T \cdot \mathbf{f}_L = \tau, \tag{29.1}$$

$$\mathbf{f}_{N\,6\times1} = \left(\Theta_N^A\right)^T_{6\times6} \cdot \mathbf{f}_{A\,6\times1}, \tag{29.2}$$

$$\Theta_N^A = X_N^A \cdot \Phi_N^A, \tag{29.3}$$

$$X_N^A = \begin{bmatrix} \mathbf{1} & \mathbf{0} \\ S(A,N) & \mathbf{1} \end{bmatrix}, \tag{29.4}$$

$$\Phi_N^A = \begin{bmatrix} R_N^A & \mathbf{0} \\ \mathbf{0} & R_N^A \end{bmatrix}, \tag{29.5}$$

where τ is the control input, $\mathbf{f}_R$, $\mathbf{f}_L$, $\mathbf{f}_{A\,6\times1}$ are the spatial vectors of moments of forces and forces in engine frames R, L, A, and $\mathbf{f}_{N\,6\times1}$ is the same vector expressed in main local frame N. Matrix R_N^A is the rotation matrix which transforms frame N into frame A, while $S(A,N)$ is the skew-symmetric matrix of vector of distance between the origins of frames N and A which satisfies $S(A,N)\cdot x = \overrightarrow{\mathbf{NA}} \times x$.

However, taking into account the assumptions that the actuators are placed symmetrically to plane XZ and can act only in this plane, the equations of forces that the actuators have to generate in order to fulfill in the maximum way the task posed by the control algorithm have the following form:

$$F_{RZ} = (\tau(5)\cdot z + \tau(6)\cdot y + \tau(1))/2y, \tag{29.6}$$

$$F_{RX} = (-\tau(4)\cdot y - \tau(5)\cdot x + \tau(3))/-2y, \tag{29.7}$$

$$F_{LZ} = (-\tau(5)\cdot z + \tau(6)\cdot y - \tau(1))/2y, \tag{29.8}$$

$$F_{LX} = (-\tau(4)\cdot y + \tau(5)\cdot x - \tau(3))/-2y, \tag{29.9}$$

where F_{RZ}, F_{RX}, F_{LZ}, F_{LX}, denote the components of the force applied to the origins of local frames of engines R and L, which act along axes Z and X. The distances x, y, z, are the absolute values of the position of the engines in frame N.

Eqs. (29.6)–(29.10) have been derived by appropriate transformations of Eq. (29.1). In addition, considering the underactuated nature of the system, we obtain the equation of realizability of control signals, which expresses the error ξ between the control signal, and the vector of forces and force moments possible to generate:

$$\xi = -\tau(4) \cdot z + \tau(6) \cdot x + \tau(2). \tag{29.10}$$

29.3 Four-Engine Structure

The four-actuator structure presented on the right-hand side in Fig. 29.1 enables better control of the orientation about Y axis, thus theoretically increasing the possibility of keeping the airship in the orientation parallel to the XY plane, but also it allows controlling the angle of attack, which has a large impact on the aerodynamic properties.

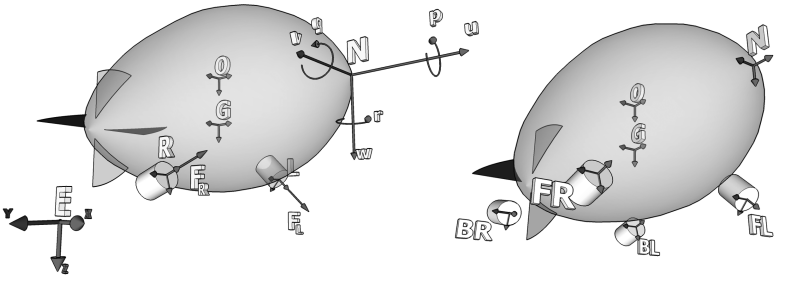


Fig. 29.1 Schematic models of the airship with two- and four-engine propulsion structure and description of their coordinate systems

The four engines rotating about their Y axis give eight control inputs, which in contrast to the solution from Section 29.2 results in a redundant system. To determine the values of forces that need to be generated by the individual engines in order to produce the resultant vector of forces and force moments in the center of gravity of the solid, which correspond to the control signals, the right-sided pseudoinversion method [10] was used.

Naturally, such a solution improves safety of flight because in case of failure of one engine controllability of the system remains unchanged.

29.4 Airship Model

29.4.1 Kinematics

The left-hand side in Fig. 29.1 shows the schematic model of the airship with two actuators moved from the plane XZ, together with the coordinate frames describing the system: the inertial frame E and the main local frame N, located at the nose of the envelope, which does not change its position with respect to the structure, and whereby the aerodynamic effects are presented. Besides, there is frame G in the center of gravity, frame O placed in the centre of the volume in which the buoyancy is expressed, and the local actuator frames, placed at the engine mounting points, which are the points of acting propulsion forces to the body of the airship. Considering the conclusions of research [11], the airship is treated as a rigid body.

The equation of motion has the following form:

$$\dot{\eta} = \begin{bmatrix} J(\eta) & 0_{3\times 3} \\ 0_{3\times 3} & R_N^E \end{bmatrix} \mathbf{v}, \tag{29.11}$$

where $\dot{\eta}_{6\times 1} = \begin{bmatrix} \dot{\varphi} & \dot{\theta} & \dot{\psi} & \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T$ is the vector of angular and linear velocities in frame E, $\mathbf{v}_{6\times 1} = [p\ q\ r\ u\ v\ w]^T$ is the vector of velocities in the main local frame N, R_N^E is the rotation matrix between frames N and E, and $J(\eta)$ denotes the transformation matrix between the angular velocities described in frame N and the global frame E.

29.4.2 Dynamics

The dynamics equation has the following form:

$$M\dot{\mathbf{v}} + C(\mathbf{v})\mathbf{v} + F^{ext} = \tau, \tag{29.12}$$

$$M, C(\mathbf{v}) \in \mathbb{R}^{6\times 6}, \tag{29.13}$$

$$\mathbf{v}, \tau \in \mathbb{R}^{6\times 1}, \tag{29.14}$$

where $\dot{\mathbf{v}} \in \mathbb{R}^{6\times 1}$ denotes the vector of the angular and linear accelerations expressed in the local frame N, M is the mass matrix, and C is the matrix of Coriolis and centrifugal forces. Equation (29.12) takes into account the added mass effect which is important in the case of bodies whose volume is large with respect to their weight, for example, mobile objects using buoyancy such as ships, submarines, and airships. In the equation of dynamics the impact of this effect is expressed in the modification of matrices M and C. The method of determining the parts added to these matrices is described for example in [7]. Element F^{ext} contains other aerodynamic forces acting on the object such as a drag dependent on the shape and the angle of attack, noise generated by texture of the envelope material, buoyancy, gravity, and disturbances. The mathematical model together with the aerodynamic coefficients defined in a wind tunnel were taken from [4]. It should be noticed that the modifications of the actuator structure can change the values of the particular coefficients [8], despite the fact that they relate only to the modification of the gondola.

29.5 Control

The control system used in the simulations realizes the task of path tracking and consists of a reference trajectory generator and control rules. Furthermore, the control signal is transmitted to the available propulsion system with the rules described in Sections 29.2 and 29.3, and then acts on the object taking into account physical limitations. None of these structures ensures full realization of the control signal. The most important constraint is the lack of possibility to actuate the airship in the direction of the Y axis, which makes the system nonholonomic. Additionally, due to such procedure it becomes possible to analyze the needs of the real engines, which is an important element in designing the real airship.

29.5.1 Trajectory Generation

The first element of the control system generates the reference trajectory in the form of a 3rd order polynomial, taking into account the method of the error tunnels. In this method, the main task should be designated in the form of a time-independent path in three-dimensional space, and the maximum position error, in other words the acceptable distance from the path. The temporary trajectory is in the form of a 3rd order polynomial and is generated on the basis of the actual state of the airship and the temporary destination point, which is always located on the main trajectory, however, depending on the circumstances at different points. Thanks to this in case of exceeding the acceptable tracking error of the main trajectory, the trajectory generator orders to return to the last point on the main trajectory from which the distance was less than the acceptable error. The actual reference trajectory, which is the part of the input of the control algorithm is the temporary trajectory. Such a solution makes it possible to guarantee the execution with certain error practical tasks such as monitoring flood barriers, or pipelines, even in cases where disturbances temporarily exceed the capabilities of the propulsion system (strong wind gusts). Figure 29.2 shows examples of generated trajectories. More information about this method can be found in [1].

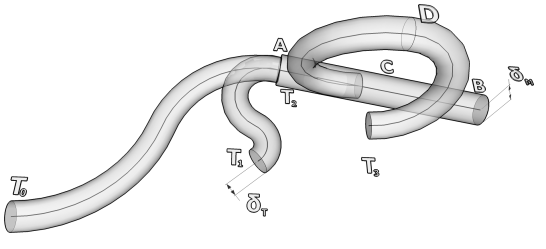


Fig. 29.2 Examples of trajectories with error tunnels; section $\overline{\mathbf{AB}}$ is main trajectory, all other are temporary trajectories

29.5.2 Control Algorithm

The second element of the control system is the control algorithm. In this case, a simple PD algorithm in the following form is used:

$$\tau = \mathbf{K_P} \cdot \mathbf{e} + \mathbf{K_D} \cdot \dot{\mathbf{e}}, \tag{29.15}$$

$$\mathbf{e} = \left[\mathbf{e}_1^T \ \mathbf{e}_2^T\right]^T, \tag{29.16}$$

$$\mathbf{e}_1 = R_E^N \cdot (\eta_{d1} - \eta_1), \tag{29.17}$$

$$\mathbf{e}_2 = R_E^N \cdot (\eta_{d2} - \eta_2), \tag{29.18}$$

where $\mathbf{K_P}$ and $\mathbf{K_D}$ are diagonal matrices of the controller parameters of size 6×6, while the vectors $\eta_d = \left[\eta_{d1}^T \ \eta_{d2}^T\right]^T$ and $\eta = \left[\eta_1^T \ \eta_2^T\right]^T$ denote the desired and actual orientation and position.

This algorithm has been selected as one of the first test algorithms due to large universality and simplicity of possible implementation. The biggest problem in an application is the selection of appropriate parameters, in this case 12. For this purpose a multiobjective genetic algorithm is used. There are five criteria for optimizing all of them: the output parameters of the simulation executed for a limited period of time for the same initial conditions and the same main trajectory. Specifically, they are the three average position errors, the number of the temporary trajectories generated during the simulation and the distance between the airship and the end of the main trajectory after the expiration of the time of the simulation. A detailed description of the procedure for selection of the parameters is in [1].

29.6 Results

The simulations were made for the path following task with the main goal different than in the procedure of selecting the control parameters. The path used for the presentation of the results is quite easy to realize but good to show the differences in the discussed solutions.

The procedure of obtaining the control parameters by using the genetic algorithm, after about 25 thousand evaluations of each function for the maximum velocity equal to 3 m/s gives sets of pareto-optimal solutions for both the two-engine and the four-engine propulsion structures. From these sets collections of parameters for the two cases are selected. The values of the optimized function for these parameters are presented in Table 29.1.

The data presented in Tab. 29.1 show important improvement in the results of the optimization function for the four-engine structure. The position error in axis X, the remaining distance, and the number of temporary trajectories are much lower. The most important advance is the last one, because when a new temporary trajectory is generated, all positional errors are reset. The results obtained in the simulations for the two-engine structure are presented in Figs. 29.3–29.6 and for the four-engine structure in Figs. 29.7–29.10.

Table 29.1 The values of the optimization function for the selected parameters

Propulsion Structure	X-error	Y-error	Z-error	Remaining Distance	NoTT[a]
2 Engines	5.7833	2.1386	6.6254	140.8124	7
4 Engines	2.2725	2.3771	6.0245	105.4867	2

[a] Number of Temporary Trajectories.

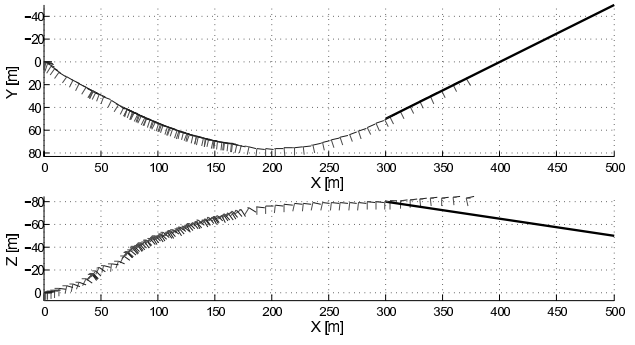


Fig. 29.3 The projections in planes XY and XZ of the movement visualization of the airship with two-engine propulsion

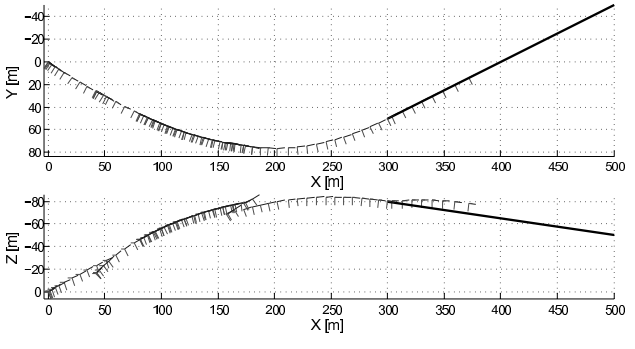


Fig. 29.4 The projections in planes XY and XZ of the trajectory visualization of the airship with two-engine propulsion

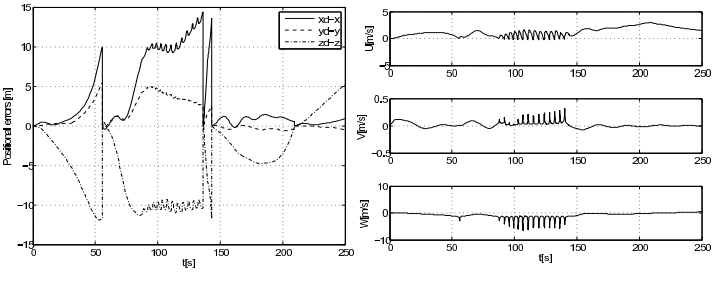


Fig. 29.5 The position errors and the velocities of the airship with two-engine propulsion

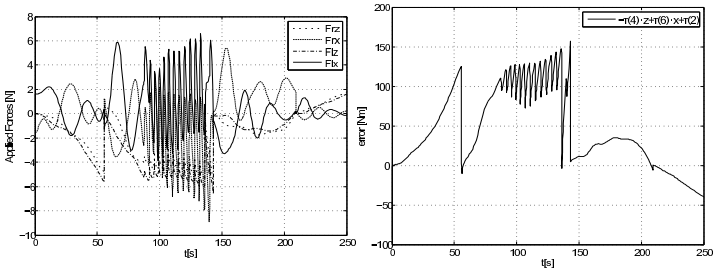


Fig. 29.6 The forces generated by the engines of the airship with two-engine propulsion and the value of ξ in Eq. (29.10)

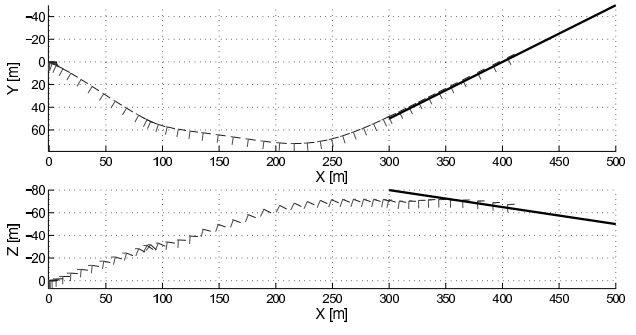


Fig. 29.7 The projections in planes XY and XZ of the movement visualization of the airship with four-engine propulsion

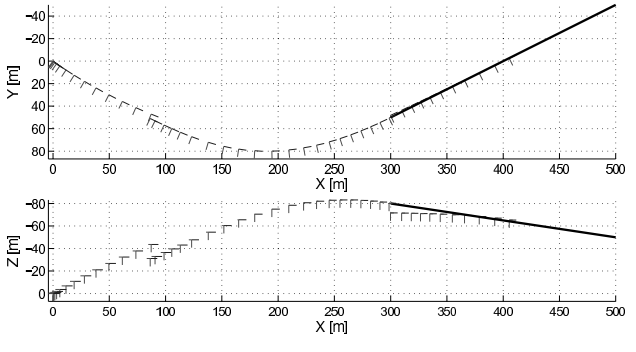


Fig. 29.8 The projections in planes XY and XZ of the trajectory visualization of the airship with four-engine propulsion

The level of realization of the given task is good, because the position errors are relatively small (Fig. 29.5). Unfortunately there are oscillations in orientation about axis Y between about 80 and 150 seconds (Figs. 29.5–29.6); they occur because in this structure of the engines there is no possibility to control pitch independently. This effect is not observed for lower velocities [1].

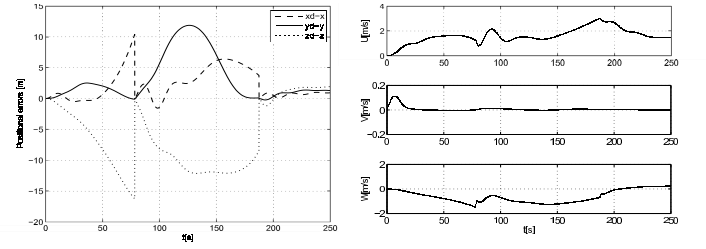


Fig. 29.9 The position errors and the velocities of the airship with two-engine propulsion

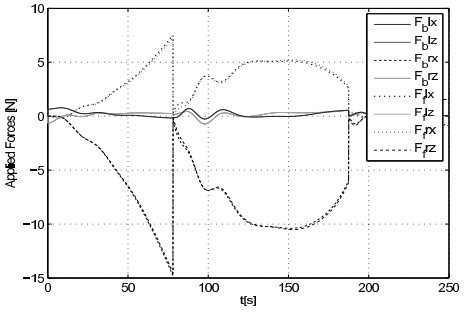


Fig. 29.10 The forces generated by the engines of the airship with four-engine propulsion

Because of the fact that four engines on two axes give possibility to control pitch, the trajectory used in the simulations has its pitch angle equal to zero (Fig. 29.8). The simulated airship moves more smoothly (Figs. 29.9–29.10) and achieves a further point on the main trajectory (Fig. 29.7).

29.7 Conclusions

The propulsion structure is an important part of the airship design, especially when ailerons are not used in control. The simulation experiments presented in this work show that it is possible to control airship movement by using the propulsion system alone, at least at lower velocities, which are important in maneuvers such as landing. Using four engines on two rotating axes gives a possibility to better control the pitch angle, which helps in fighting with oscillations about axis Y.

In this work a simple PD type controller was used, however with a more sophisticated algorithm it is probable to eliminate oscillations in the two-engine case. But the authors' intention was to compare engine structures for the same type of algorithm, which operates on values associated with the airship body. Differences in propulsion structures are invisible for the control algorithm, so it is legitimate to assume that for a better control regime the advantages of the four-engine system occur as well.

Of course using four engines instead of two has weaknesses. The main one is usually a greater weight of the system, but in the case of airships where lift is relatively "cheap", this is not most important.

References

1. Adamski, W., Herman, P., Bestaoui, Y., Kozłowski, K.: Control of airship in case of unpredictable environment conditions. In: Conference on Control and Fault-Tolerant System - SysTol 2010, Nice, France, pp. 843–848 (2010), doi:10.1109/SYSTOL.2010.5675976
2. Ashraf, Z., Choudhry, M.A.: Dynamic modeling of the airship using analytical aerodynamic model. In: International Conference on Emerging Technologies, ICET 2009, Islamabad, pp. 188–193 (2009), doi:10.1109/ICET.2009.5353178
3. Azinheira, J.R., Moutinho, A., De Paiva, E.C.: A backstepping controller for path-tracking of an underactuated autonomous airship. International Journal of Robust and Nonlinear Control 19(4), 418–441 (2009)
4. Bestaoui, Y.: In: Unpublished report (2007)
5. Bestaoui, Y., Kuhlmann, H.: A Newton Euler approach to modeling of a quad-rotor autonomous airship – preliminary results. In: 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition. American Institute of Aeronautics and Astronautics (2010)
6. Bestaouui, Y., Kuhlmann, H.: A Newton Euler approach to modeling of a quad-rotor autonomous airship – preliminary results. In: 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, Orlando, Florida (2010)
7. Fossen, T.I.: Guidance and Control of Ocean Vehicles. Wiley (1994)
8. Khoury, G.A., Gillett, J.D.: Airship Technology (Cambridge Aerospace Series). Cambridge University Press (2004)
9. Kornienko, A.: System identification approach for determining flight dynamical characteristics of an airship from flight data. Ph.D. thesis, Stuttgart (2006)
10. Kozłowski, K., Wróblewski, W., Dutkiewicz, P.: Modelowanie i sterowanie robotów. Wydawnictwo Naukowe PWN, Warszawa (2003) (in Polish)
11. Li, Y.: Dynamics modeling and simulation of flexible airships. Ph.D. thesis, Montreal (2008)
12. Moutinho, A.: Modeling and nonlinear control for airship autonomous flight. Ph.D. thesis, Lisbona (2007)
13. Solaque, L., Pinzon, Z., Duque, M.: Nonlinear control of the airship cruise flight phase with dynamical decoupling. In: Electronics, Robotics and Automotive Mechanics Conference, CERMA 2008, Morelos, pp. 472–477 (2008), doi:10.1109/CERMA.2008.53
14. Thomasson, P.G.: Equations of motion of a vehicle in a moving fluid. Journal of Aircraft 37(4), 630–639 (2000)

Chapter 30
Dynamic Simulations of Free-Floating Space Robots

Tomasz Rybus, Karol Seweryn, Marek Banaszkiewicz, Krystyna Macioszek, Bernd Mädiger, and Josef Sommer

Abstract. This paper focuses on the dynamics of a 6-dof manipulator mounted on a free-flying servicer satellite during final part of an on-orbit rendezvous maneuver. Determination of reaction torques induced by the manipulator on the servicer satellite is critical for the development of the Guidance, Navigation and Control (GNC) subsystem. Presented in this paper is a path planning algorithm for capturing a tumbling target satellite, as well as simulation results of the capture maneuver and folding of the manipulator with the attached target satellite. The second part of this paper is focused on the presentation of our work leading to the construction of a planar air-bering test-bed for space manipulators.

30.1 Introduction

The lifetime of every Earth-orbiting spacecraft is limited by the amount of fuel on-board. The malfunctioning of satellite's components may additionally reduce their operational period. Because of the increasing complexity and costs of satellites, missions that could extend satellites' lifetime are being considered ([23]). Due to economical and safety issues it seems obvious that these missions should be unmanned. Advances in automation and robotics allow for development of autonomous space robots. In recent years many studies showed that unmanned servicing missions might be economically justified ([8], [3]). Therefore specific business plans are presented ([14]), concepts of various possible servicing missions and strategies are developed (e.g. [25]), and some commercial enterprises begin to

Tomasz Rybus · Karol Seweryn · Marek Banaszkiewicz · Krystyna Macioszek
Space Research Centre, Polish Academy of Sciences,
ul. Bartycka 18a, 00-716 Warsaw, Poland
e-mail: {trybus,kseweryn,marekb}@cbk.waw.pl

Bernd Mädiger · Josef Sommer
ASTRIUM Space Transportation GmbH, Bremen, Germany
e-mail: {bernd.maediger,josef.sommer}@astrium.eads.net

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 351–361.
springerlink.com

emerge (e.g. Orbital Life Extension Vehicle – OLEV). A number of different tasks could be realized by servicing missions, e.g.: on-orbit refueling of Low Earth Orbit (LEO) and Geostationary Earth Orbit (GEO) satellites or taking over attitude and orbit control for live extension; management services (e.g. towing, de-orbiting).

The unmanned servicing satellites capable of capturing non-cooperative (uncontrolled) targets may also be used for another important task: utilization of large space debris (i.e. defunct, defective satellites). Computer simulations show that current debris population in LEO is likely to increase due to random collisions between existing on-orbit debris, even if all of the new satellites were launched with some built-in disposal system ([11]), and that active removal of existing intact objects (several per year) can prevent this growth of the existing LEO population of space debris ([10]). The mission concepts for removal of specific large debris which would rely on the capture of a target satellite by a manipulator are proposed (e.g. [15]).

Demonstrative missions that tested parts of technology required for on-orbit servicing were conducted (e.g. DART in 2005, Orbital Express in 2007, ROKVISS on ISS from 2005 ([9])) and some others are planned (e.g. Deutsche Orbitale Servicing Mission – DEOS ([22], [17])). An overview of the servicing missions can be found in [16]. In parallel, theoretical investigations of space manipulator dynamics were performed by various authors (e.g. [1]). Work performed at the Space Research Centre of the Polish Academy of Sciences ([19], [20]) formed the basis for a joint project between SRC PAS and Astrium ST. The results presented in this paper show two different issues: (i) a simulation tool of space robots, which is available for non-commercial use on request, and (ii) simulation results showing the level of interactions between the manipulator control subsystem and the Guidance, Navigation and Control (GNC) subsystem for the case which is currently under development. Both together give information of the current status of work performed in SRC PAS.

The paper is divided into six sections, including the introduction. In the second section path a planning algorithm of the manipulator arm is presented. In the ideal fixed base case it allows moving manipulator EE from the initial point to the final one and fulfill the desired path of position and velocity. In the next paragraph a simulation tool for simulating the orbital motion of both service and target satellites is described. It allows testing the path planning algorithm and gives insight into the interactions between the satellite GNC and the manipulator's motion which are presented in section four. In the fifth section the approach to validate space manipulator behavior is shown. All results and plans for future investigations are summarized in the last section.

30.2 Path Planning Algorithm for Capture of Tumbling Target Satellite

In our study a system consisting of a client (target) satellite and a servicer (chaser) satellite equipped with a 6-dof manipulator is considered. This system is presented in Fig. 30.1. Both satellites are treated as free floating bodies in orbital motion. The servicer satellite is equipped with GNC, which exploits thrusters for attitude control.

The GNC algorithm (developed by Astrium ST) uses information on orientation and angular velocity of the satellite to generate torques acting on the servicer's center of gravity (CG) in order to keep the desired attitude regardless of external torques disturbing the satellite's orientation. These torques are generated by the motions of the manipulator. A joint controller, based on a linear-quadratic regulation, to control the manipulator's arm motion was developed and tested at SRC PAS. In our previous studies we had presented an algorithm for free-floating path planning ([19]). In the current investigations a simplification of that algorithm with the assumption of a fixed manipulator base is used. It allows investigating the interactions between the manipulator and GNC (the manipulator mounted on a satellite equipped with ideal GNC, keeping the desired attitude and position, corresponds to the case of a fixed base manipulator and, therefore, using the fixed-base algorithm is justified in this case). The generalized equation of motion for a system consisting of a free-floating satellite equipped with a 6-dof robotic arm was used in our simulations:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\dot{\mathbf{q}},\mathbf{q})\dot{\mathbf{q}} = \begin{bmatrix} 0_{6\times 1} \\ \mathbf{Q}_{fix} \end{bmatrix} + \begin{bmatrix} 0_{6\times 6} & 0_{6\times 12} \\ 0_{6\times 6} & \mathbf{K} \end{bmatrix} \begin{bmatrix} 0_{6\times 1} \\ \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix} + \begin{bmatrix} 0_{3\times 1} \\ \mathbf{Q}_{GNC} \\ 0_{6\times 1} \end{bmatrix}, \qquad (30.1)$$

where $\mathbf{q} = [x_s, y_s, z_s, \varphi_s, \theta_s, \psi_s, \theta_1 \ldots, \theta_6]^T$ is the 12-dimensional state vector consisting of the satellite's position, orientation (Euler angles) and a set of 6 generalized coordinates (i.e. joints angles), $\mathbf{M}$ is the manipulator mass matrix, $\mathbf{C}$ is the Coriolis matrix, $\mathbf{Q}_{fix}$ is a vector of torques acting on the manipulator's joints (computed with the fixed-base trajectory planning algorithm), $\mathbf{Q}_{GNC}$ is a vector of attitude control torques acting on the satellite (computed by the GNC controller) and $\mathbf{K}$ is a gain matrix for the joint controller.

The algorithm used for computation of matrices $\mathbf{M}$ and $\mathbf{C}$ is based on the approach presented in [2]. If the desired trajectory of manipulator EE is defined in quasi coordinates (positions of the manipulator's joints) $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$, and $\ddot{\mathbf{q}}(t)$. Equation (30.1) is used directly, but for the purpose of trajectory planning in Cartesian coordinates the inverse kinematics problem is solved and values of $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$, and $\ddot{\mathbf{q}}(t)$ are computed from the trajectory given in terms of linear position, linear velocity, linear acceleration, orientation, angular velocity, and angular acceleration of the EE ($\mathbf{p_{ee}}(t), \mathbf{v_{ee}}(t), \mathbf{a_{ee}}(t), \mathbf{T_{ee}}(t), \omega_{\mathbf{ee}}(t)$, and $\varepsilon_{\mathbf{ee}}(t)$ respectively). The quasi coordinates are used for simple maneuvers (e.g. unfolding the manipulator from the stowed position), while the Cartesian coordinates are used in the main part of the rendezvous, i.e. the final approach of the EE and capture of the target satellite.

For the worst case scenario it is assumed that the target satellite is non-cooperative, which means that it is uncontrolled and tumbling (such behavior of many defunct satellites was confirmed by ground observations [7]).

In the final part of the maneuver, when both satellites are close enough and there is no relative motion between them, the EE is moved to a point from which the final approach to the docking port is conducted. For this last part of the maneuver the straight-line (in the docking port coordinate system (CS)) trajectory was selected. This maneuver assures obstacle avoidance once the EE reaches the proximity of

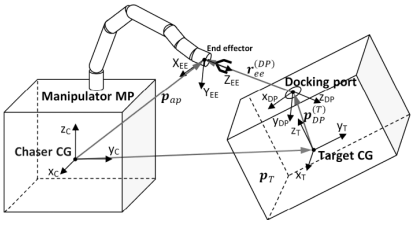


Fig. 30.1 Schematic view of the chaser and target satellites during the final part of the rendezvous maneuver

the target. Trajectory planning is constrained by the final state of the EE which should be the same as the state of the docking port at the moment of capture. The equations describing this trajectory in the Cartesian CS located at the servicer's CG are presented below:

$$\mathbf{p_{ee}} = \mathbf{p_T} + \mathbf{T_T^{-1}} \cdot \mathbf{r_{ee}^{(T)}}, \tag{30.2}$$

$$\mathbf{V_{ee}} = \omega_\mathbf{T} \times \left(\mathbf{T_T^{-1}} \cdot \mathbf{r_{ee}^{(T)}} \right) + \mathbf{T_T^{-1}} \cdot \mathbf{T_{DP}^{(T)^{-1}}} \cdot \dot{\mathbf{r}}_{\mathbf{ee}}^{\mathbf{(DP)}}, \tag{30.3}$$

$$\begin{aligned} \mathbf{a_{ee}} = {} & \omega_\mathbf{T} \times \left(\omega_\mathbf{T} \times \mathbf{T_T^{-1}} \cdot \mathbf{r_{ee}^{(T)}} \right) + \varepsilon_\mathbf{T} \times \mathbf{T_T^{-1}} \cdot \mathbf{r_{ee}^{(T)}} + \\ & + \mathbf{T_T^{-1}} \cdot \mathbf{T_{DP}^{(T)^{-1}}} \cdot \ddot{\mathbf{r}}_{\mathbf{ee}}^{\mathbf{(DP)}} + 2 \cdot \omega_\mathbf{T} \times \mathbf{T_T^{-1}} \cdot \mathbf{T_{DP}^{(T)^{-1}}} \cdot \dot{\mathbf{r}}_{\mathbf{ee}}^{\mathbf{(DP)}}, \end{aligned} \tag{30.4}$$

$$\mathbf{T_{ee}} = \mathbf{T_T} \mathbf{T_{DP}^{(T)}}, \tag{30.5}$$

$$\omega_\mathbf{ee} = \omega_\mathbf{T}, \tag{30.6}$$

$$\varepsilon_\mathbf{ee} = \varepsilon_\mathbf{T}, \tag{30.7}$$

where $\mathbf{r_{ee}^{(T)}} = \mathbf{p_{DP}^{(T)}} + \mathbf{T_{DP}^{(T)^{-1}}} \cdot \mathbf{r_{ee}^{(DP)}}$; $\mathbf{p_T}$, $\omega_\mathbf{T}$ and $\varepsilon_\mathbf{T}$ are the position, angular velocity, and angular acceleration vectors of the target satellite; $\mathbf{T_T}$ is the orientation matrix of the target satellite; $\mathbf{p_{DP}}$ is the position of the docking port; $\mathbf{T_{DP}}$ is the orientation matrix of the docking port and $\mathbf{r_{ee}}$ is the position vector from the docking port to the EE. The CS are denoted by superscripts (T) and (DP) for the target satellite CS (located at its CG) and the docking port CS, respectively. The quantities defined in the chaser satellite CS (located at its CG) are not denoted by any superscript.

The distance (in the docking port CS) between the EE and the docking port of the target satellite during the final phase of the rendezvous maneuver could be defined by any time-dependant class C^2 function: $\mathbf{r_{ee}^{(DP)}} = f(t)$. The results presented in Section 30.5 were obtained for the EE approach divided into three stages: acceleration, motion with constant velocity (with respect to the docking port), and breaking. The trajectory could be alternatively planned with non-zero initial relative velocity, which might be even more beneficial for the minimization of energy used by the joints' motors, because there is no need to stop the EE at the starting point of the final approach.

30.3 Simulation Tool

Our work is based on a model developed in SimMechanics. The SimMechanics software is based on the Simscape, the platform product for the Matlab/Simulink. This platform allowed us to incorporate controllers (GNC and LQR joint controller) and system dynamics in one environment. The Simscape solver works autonomously and creates the physical network from the block model of the system. Then the system of equations for the model is constructed and solved ([5], [24]).

The satellite and manipulator links are modeled as rigid bodies, with inertial and geometrical parameters of the manipulator taken from a CAD model of the robotic arm. Coulomb and viscous friction in the joints are modeled, as well as the backlash effect. Specific use of the SimMechanics built-in block of a 6-dof joint allowed us to simulate tumbling motion of a satellite and its orbital motion around the Earth. The simulations are divided into two parts: at the beginning a trajectory of the manipulator is planned by the algorithm implemented in MATLAB function. It is assumed that the inertial parameters and the parameters of motion of the target satellite are known at the beginning of the rendezvous maneuver (during a space mission these parameters could be assessed by an optical system, e.g. [6], [4]). The motion of the target satellite is computed in advance by solving the Euler equations of rigid body motion from the initial parameters and then computation of the docking port trajectory (position and orientation) takes place. Using this data, the torques for the joints are computed from the numerical solution of the manipulator dynamics. The gain matrix for the joint controller depends on the configuration of the manipulator and is computed for every step of the simulation – for the purpose of the linear-quadratic regulation the non-linear system of the manipulator has to be linearized in every step ([21]). After the phase of trajectory planning, the actual simulation in Simulink begins. This serves as a natural verification of the trajectory planning algorithms as it is only necessary to compare if the trajectory realized by the SimMechanics model of the system is consistent with the planned one. Our simulation tool allows us to simulate fixed base and free floating cases (also with orbital motion). Specific use of the 6-dof joint in SimMechanics allows us to set the initial rate of this joint, which is usually unavailable in commercial software.

30.4 Results of Simulations

The simulations of the final part of the rendezvous maneuver were performed for different initial parameters (i.e. the orientation and angular velocity of the tumbling target satellite) in order to assess the performance of GNC and to estimate the maximum values of GNC-generated torques required for the attitude control during the maneuver. Below the results of simulations for fixed base manipulator are presented. In the analysed scenario the mass of the target and chaser satellites is 340 kg and 750 kg, respectively. The moving mass of the manipulator is 35 kg and its total length is 3 m. The motion of the EE is presented in Fig. 30.2. In the left panel of Fig. 30.3 the orientation (Euler angles in 3-2-1 convention) of the tumbling target

satellite during the maneuver is shown. At the beginning of the final approach the EE is 0.5 m from the docking port and at the end its state is the same as the state of the docking port. The final approach was planned for 20 seconds, with 8 seconds of acceleration and 8 seconds of breaking (in the docking port CS).

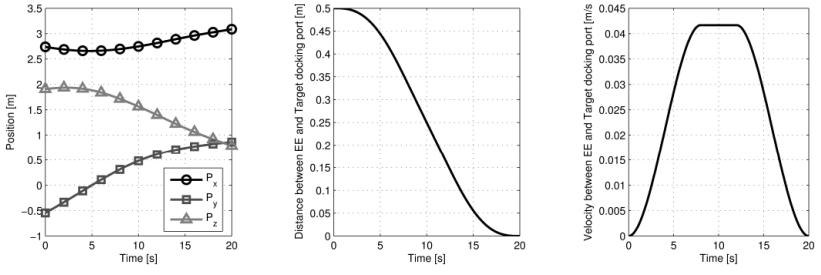


Fig. 30.2 EE position in the chaser satellite CS (left panel) and EE position and velocity in the docking port CS (only one component has non-zero values and thus two other components are not plotted)

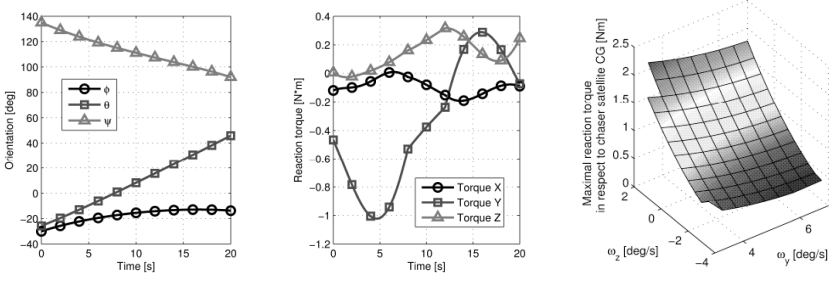


Fig. 30.3 Target satellite orientation (left panel), manipulator reaction torques with respect to the chaser satellite CG (center), and maximum value of the reaction torque during the capture maneuver for the target satellite angular velocity $\omega_x = 1$deg/s as a function of the other two components of its angular velocity (right panel)

The rather complicated motion of the manipulator, necessary for this linear approach to the docking port, results in the reaction torques and reaction forces acting on the manipulator base. These torques are presented in Fig. 30.3 with respect to the servicer CG. In the free floating case the ideal GNC (attitude and position keeping) would have to generate the opposite torques in order to compensate for the manipulator's reactions.

The trajectory of the EE's linear approach depends on the relative position and orientation of both satellites and on the parameters of target satellite's motion. The maximum value of the reaction torque during the capture maneuver for the x-axis angular velocity (in the body reference frame) of the target satellite tumbling motion $\omega_x = 1$ deg/s as a function of the other two components of its angular velocity is

presented in the right panel of Fig. 30.3. It is clearly seen that the increase of the ω_y component of the angular velocity does not necessarily cause the increase of the maximum reaction torque during the maneuver. As far as the case presented on this plot is concerned, the increase of this component actually means that the docking port could be reached by the EE with smaller and slower motion of the manipulator.

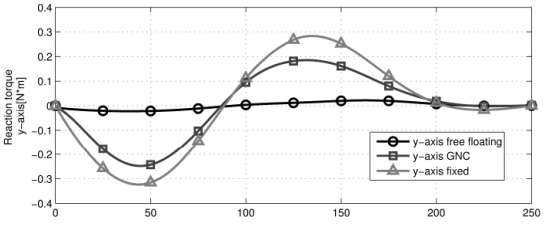


Fig. 30.4 Manipulator reaction torques with respect to the chaser satellite CG during folding of the manipulator

The second part of our analyses is devoted to the behavior of the system during folding the manipulator with the target satellite attached to the EE (after capture and de-tumbling of the target). For folding the manipulator the trajectory is defined in quasi-coordinates (positions of the joints). During motion the joints accelerate to a specific velocity, then move with this constant velocity and brake at the end of motion. Although this maneuver can be performed with very low velocity, the resulting reaction torques may be substantial because the mass of the target satellite attached to the last link of the manipulator is significant in comparison to the mass of the servicer satellite. The Y-axis reaction torque is presented in Fig. 30.4 for three different cases: (1) maneuver without attitude control of the servicer, (2) maneuver with attitude control, and (3) maneuver for the fixed base manipulator as a reference which corresponds to the ideal attitude and position keeping. The LQR joint controller was used in cases (1) and (2) to assure appropriate following the trajectory defined by the quasi-coordinates. When the maneuver is performed without attitude control and the servicer satellite is allowed to rotate freely, the joint controller assures proper relative motion of the target satellite, while the resulting reaction torques are very small. The difference between cases (2) and (3) is due to the linear motion of the servicer CG.

30.5 Verification of Space Robotic Simulations

In order to verify the control algorithms and numerical simulations, tests on a real manipulator should be performed. Designing and building space missions that could demonstrate the technology of satellite servicing is very expensive. This is a reason why it is necessary to build test-beds in laboratories on the Earth. It must be taken

into account that space robots are free-floating, as they work in zero-gravity conditions, which means they move with almost no friction and that their base has additional 6-dof as opposed to robots working on the Earth. Fulfilling these conditions on the Earth is very difficult, however there are several solutions ([13], [18]). The most complicated task is to provide frictionless motion in all three dimensions. It can be performed in water conditions or during parabolic flights ([12]). At the SRC PAS it was decided to build a test-bed providing frictionless planar motion. The requirements of the test-bed are as follows: the compensation of the gravity influence, the provision of a free planar motion and the possibility to record the manipulator's movements. Assuring only the free planar motion (2D) gives possibility to verify in some cases control algorithms simulations for free-floating robots with 6-dof.

The first two requirements can be obtained using air-bearings. They generate a thin film of pressurized air and slide on it. This film is of the magnitude of several microns and depends on the load – the higher the load, the smaller the air gap. The flat round air-bearings based on porous media technology are used in the test-bed which is being built at SRC PAS. Pressurized air is supplied through a hole on a side of the air-bearing and then runs out through special porous media (a schematic view of the air-bearing is presented in Fig. 30.5). Owing to this technology the air under the air-bearing is distributed very evenly, giving very smooth motion and high stiffness of the bearing. Moreover, any scratches on their surface do not cause damages at all, contrary to classic air-bearings where the air is distributed through many small orifices. In their case any damage causes significant decrease in the bearing's performance.

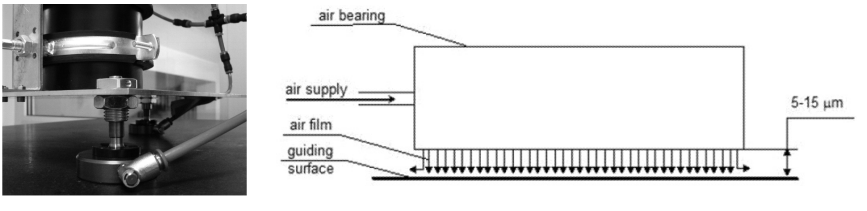


Fig. 30.5 Air-bearings mounted on a test model of a satellite simulator (left panel) and schematic view of the air bearing (right panel)

These air-bearings provide almost frictionless motion – according to the producer their coefficient of friction is of the magnitude of 10^{-5}. This means that the guiding surface on which they move should be leveled very precisely, as in case of any inclinations the bearings would simply slide down. Owing to a very small air film, the guiding surface must have a very low roughness and high flatness. It was observed that even very small inaccuracies in the surface cause stopping of the air-bearing.

In the test-bed a manipulator with two links with rotational joints will be used. The manipulator will be moving on the air-bearings supplied with compressed air from the gas cylinders. The data about the manipulator's movements will be obtained from a special vision system and from the encoders in each joint. This data

will be compared to the values derived from simulations performed by our Simulink tool. In the future the same test-bed can be used to test different manipulators. Until now tests were performed with the sole base of a manipulator supported with three air-bearings. This allowed us to specify requirements that should be met when designing the final manipulator.

30.6 Conclusions

A flexible simulation tool that takes into account interactions between separate components of a servicer satellite (GNC, joint controller, manipulator) was created in the Simulink environment. This tool was used for simulations of the final part of the orbital rendezvous during a servicing mission in order to compute the reaction torques induced by motions of the manipulator on the servicer satellite. The path planning algorithm that can be used on a satellite with active attitude control and position keeping for capturing the tumbling target satellite was developed and tested as a standalone application. This path planning algorithm for a fixed base manipulator was used in simulations performed with different initial parameters of the target satellite motion in order to assess the performance of GNC and to estimate the maximum values of GNC-generated torques required for the attitude control of the free floating satellite during the capture maneuver. The performed work shows that it is necessary to simulate many different situations in order to assess these values, as there is no general solution and every case must be analysed separately. The appropriate functioning of the joint controller based on linear-quadratic regulation used on a highly non-linear system was demonstrated in simulations of folding the manipulator with the attached target satellite. In such simulations the joint controller was able to compensate differences between joint control torques generated for the fixed base case and control torques required for appropriate trajectory following in the case of free floating satellite. In the second part of this paper an approach was presented for verification of space robotics simulations on the planar air-bearing table, which could be used as a simulator of microgravity conditions for free-floating robots. Further work is planned to incorporate the free-floating path planning algorithm in our simulation tool and tests of the simple manipulator mounted on a simulator of a satellite is planned on the air-bearing test bed. The authors of this paper are also proposing a space robots experiment inside the Columbus module of the ISS (SPACE CAT).

Acknowledgements. The analyzed scenario is a part of common project between SRC PAS and ASTRIUM ST. The details about the manipulator and satellite were taken from the INVERITAS project and used as a simulation example. The INVERITAS project is a joint project between EADS Astrium, Jena-Optronik, and the RICDFKI Bremen, supported by the German Aerospace Center (DLR), funded by the Federal Ministry of Economics and Technology.

References

1. Aghili, F.: Optimal Control for Robotic Capturing and Passivation of a Tumbling Satellite with Unknown Dynamics. In: AIAA GNC Conference and Exhibit. Honolulu, Hawaii, USA (2008)
2. Angeles, J.: Fundamentals of Robotic Mechanical Systems: Theory, Methods and Algorithms. Springer, New York (2003)
3. Cougnet, C., Gerber, B., Heemskerk, C., et al.: On-Orbit Servicing System of a GEO Satellite Fleet. In: ASTRA 2006, ESTEC, Noordwijk, The Netherlands (2006)
4. Dionnet, F., Marchand, E.: Robust model-based tracking with multiple cameras for spatian applications. In: ASTRA 2006, ESTEC, Noordwijk, The Netherlands (2006)
5. Haug, E.: Computer Aided Kinematics and Dynamics of Mechanical Systems. In: Basic Methods, vol. 1. Allyn and Bacon, London (1989)
6. Kapellos, K., Chaumette, F., et al.: VIMANCO: ASTRA 2006, ESTEC, Noordwijk, The Netherlands (2006)
7. Kawamoto, S., Nishida, S., Kibe, S.: Research on a Space Debris Removal System. NAL Res. Prog. 2002/2003 (2003)
8. Kreisel, J.: On-Orbit Servicing of Satellites (OOS): Its Potential Market & Impact. In: ASTRA 2002, ESTEC, Noordwijk, The Netherlands (2002)
9. Landzettel, K., Albu-Schaffer, A., et al.: Robotic On-Orbit Servicing – DLR's Experience and Perspective. In: Proceedings of the 2006 IEEE/RSJ, IROS, Beijing, China (2006)
10. Lewis, H., Swinerd, G., et al.: Active Removal Study for On-Orbit Debris Using DAMAGE. In: Proc. 5th European Conference on Space Debris, Darmstadt, Germany (2009)
11. Liou, J.-C., Johnson, N.L.: Risks in Space from Orbiting Debris. Science (311) (2006)
12. Menon, C., Aboudan, A., et al.: Free-Flying Robot Tested on Parabolic Flights: Kinematic Control. Journal of Guidance, Control and Dynamics 28(4) (2005)
13. Menon, C., Busolo, S., et al.: Issues and solutions for testing free-flying robots. Acta Astronautica 60 (2007)
14. Mueller, R., Kreisel, J.: On-Orbit Servicing (OOS) Concept Trades Based on the S@tMax System. In: ASTRA 2006, ESTEC, Noordwijk, The Netherlands (2006)
15. Rebele, B., Krenn, R., Schäfer, B.: Grasping Strategies and Dynamic Aspects in Satellite Capturing by Robotic Manipulator. In: ASTRA 2002, ESTEC, Noordwijk, The Netherlands (2002)
16. Rekleitis, I., Martin, E., et al.: Autonomous Capture of a Tumbling Satellite. Journal of Field Robotics 24(4), 1–XXXX (2007)
17. Rupp, T., Boge, T., et al.: Flight Dynamics Challenges of the German On-Orbit Servicing Mission DEOS. In: 21st International Symposium on Space Flight Dynamics, Toulouse, France (2009)
18. Schwarz, J., Peck, M., Hall, C.: Historical Review of Air-Bearing Spacecraft Simulators. Journal of Guidance, Control and Dynamics 26(4) (2003)
19. Seweryn, K., Banaszkiewicz, M.: Optimization of the Trajectory of a General Free-flying Manipulator During the Rendezvous Maneuver. In: AIAA Guidance, Navigation and Control Conference and Exhibit, Honolulu, Hawaii, USA (2008)
20. Seweryn, K., Banaszkiewicz, M.: Trajectories of a Manipulator During the Satellite Rendezvous Maneuver. In: 1st ESA Workshop on Multibody Dynamics for Space Applications, ESTEC, Noordwijk, The Netherlands (2010)

21. Seweryn, K., Rybus, T.: Interaction of satellites during the final phase of the rendezvous maneuver: ROBOT. Technical report part II, Space Research Centre PAS (2010)
22. Sellmaier, F., Boge, T., et al.: On-orbit Servicing Missions: Challenges and Solutions for Spacecraft Operations. In: SpaceOps, Conference, Huntsville, Alabama, USA (2010)
23. Waltz, D.M.: On-orbit Servicing of Space Systems. Kriger Publishing Co., Malabar (1993)
24. Wood, G., Kennedy, D.: Simulating Mechanical Systems in Simulink with SimMechanics. Technical report, The MathWorks, Inc., Natick, USA (2003)
25. Yasaka, T., Ashford, W.: GSV: An Approach Toward Space System Servicing. Earth Space Review 5(2) (1996)

Part VI
Motion Planning and Control of Manipulators

Chapter 31
Planning Motion of Manipulators with Local Manipulability Optimization

Ignacy Dulęba and Iwona Karcz-Dulęba

Abstract. An original algorithm of locally optimal (w.r.t. the manipulability criterion) motion planning for manipulators is presented. It takes advantage of the Singular Value Decomposition algorithm to decompose the Jacobian matrix of a manipulator and to transform a local motion planning problem into a task of finding a tangent point of a given manipulability ellipsoid with a family of spheres. This task supported by an analytic geometry technique generates one-dimensional optimization task. Its solution determines a short term motion at a current configuration. Performance of the algorithm was illustrated on a 3D planar pendulum manipulator.

31.1 Introduction

Manipulability is one of the basic properties of robotic systems [2, 3]. Introduced by Yoshikawa [6], it is defined as the square root of the determinant of the manipulability matrix (the matrix is a product of the Jacobian matrix and its transposition, while the Jacobi matrix is derived from appropriately defined kinematics k). Manipulability is popular due to its nice physical interpretation, analytical soundness, and practical importance. Manipulability can also be defined in terms of singular values of the manipulability matrix. In practice it is desirable to maximize manipulability to exploit motion abilities of a manipulator, to save energy on moving its joints and to avoid singular configurations where manipulability attains its minimal, equal to zero, value and the Newton algorithm of motion planning [5] fails to work. Manipulability can be assigned to each robotic system with an appropriately defined "kinematic" transformation. Primarily, this property was defined for manipulators and

Ignacy Dulęba · Iwona Karcz-Dulęba
Institute of Computer Engineering, Control, and Robotics,
Wrocław University of Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
e-mail: {ignacy.duleba,iwona.duleba}@pwr.wroc.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 365–375.
springerlink.com

their kinematics, then it was extended to systems like mobile platforms [7], nonholonomic manipulators, compound robotic systems (tree-like multi-effector manipulators) and finally also to systems with dynamic characteristics [1]. Originally, the manipulability characterized a single configuration but it can be extended easily to evaluate trajectories by taking an integral along the trajectory. Usually a main concern of motion planning in a collision-free environment is to reach a given goal $\boldsymbol{x}_f$ in the taskspace starting at a configuration $\boldsymbol{q}_0$. Let the task be to find a trajectory $\boldsymbol{q}(\cdot)$ joining $\boldsymbol{q}_0$ with some $\boldsymbol{q}^\star$, such that $k(\boldsymbol{q}^\star) = \boldsymbol{x}_f$, $\|k(\boldsymbol{q}(t)) - \boldsymbol{x}_f\|$ decreases monotonically as t increases and the total manipulability along the trajectory is maximized. To find the globally optimal solution for the task seems to be very difficult. Therefore locally optimal solutions are preferred. The local optimization paradigm assumes that a global trajectory is composed of pieces of trajectories resulting from local (much simpler than global) optimizations around a current configuration. The first current configuration is $\boldsymbol{q}_0$ and (iteratively) the end-configuration of previous local planning is set for the next current configuration. At least two approaches are possible. One approach relies on using a Newton algorithm with a secondary function (manipulability) optimization in the null space of the Jacobi matrix [5]. However, in this case, motion towards the goal is performed along a straight line in the taskspace, which is not necessarily the best strategy to maximize manipulability. Therefore, in this paper, a different and original approach to optimize manipulability is introduced. It assumes that at a given configuration any motion decreasing the Euclidean distance towards $\boldsymbol{x}_f$ is admissible. Thus, a more numerous set of directions is used to search for the locally optimal motion with respect to the manipulability criterion function. To maximize locally the manipulability, we take advantage of the Singular Value Decomposition algorithm [4] to transform this task into a geometric domain where a task of finding the intersection of a family of spheres centered at $\boldsymbol{x}_f$ with a manipulability ellipsoid corresponding to a current configuration has to be solved. Fortunatelly, it is a simple task in the analytic geometry and its solution can be found almost analytically.

The question whether it is desirable to optimize manipulability along a trajectory rather than at its final configuration should be posed. Optimization of the manipulability at the final configuration (corresponding to $\boldsymbol{x}_f$) may not cause significant improvement of the manipulability, as a restrictive condition of not going away from the reached goal should be respected. Moreover, more ellegant and shorter motion towards the goal can be generated with permanent optimization of the manipulability.

This paper is organized into five sections. In Section 31.2 some issues on manipulability and related terms are briefly recalled. Also a locally optimal (w.r.t. the manipulability function) algorithm of motion planning is presented. In Section 31.3 an almost analytical solution is presented for a key task of the algorithm, namely a search for a tangent point of an ellipsoid and a family of spheres. In Section 31.4 some simulation results are presented to illustrate the algorithm. Section 31.5 concludes the paper.

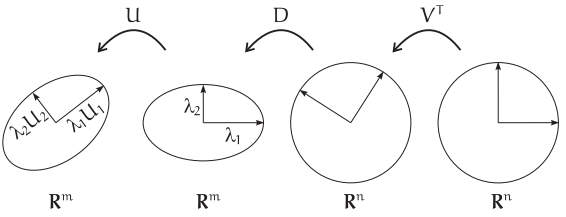


Fig. 31.1 Phases of velocity transformation from the configuration space to the taskspace

31.2 An Algorithm of Locally Optimal Motion Planning

The manipulator kinematics is a mapping from a configuration space Q into a taskspace X

$$k(\boldsymbol{q} \in Q) = \boldsymbol{x} \in X \subset SE(3), \qquad \dim Q = n, \qquad \dim X = m, \tag{31.1}$$

that assigns to each configuration $\boldsymbol{q}$ a generalized position $\boldsymbol{x}$ (i.e. position and/or orientation) of the end-effector. Manipulability is based on the Jacobi matrix $J(\boldsymbol{q})$ which transforms velocities from the configuration space to the taskspace $\dot{\boldsymbol{x}} = \frac{\partial k}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} = J(\boldsymbol{q})\dot{\boldsymbol{q}}$. To describe the geometry of motion at a given configuration $\boldsymbol{q}$, the Jacobi matrix J, $(m \times n)$, is split with the Singular Value Decomposition (SVD) algorithm [4] into a product of three matrices

$$J(\boldsymbol{q}) = U(\boldsymbol{q})\, D(\boldsymbol{q})\, V^T(\boldsymbol{q}), \tag{31.2}$$

where U is an $(m \times m)$ rotation matrix, $D = diag(\lambda_1, \ldots, \lambda_m)$ is an $(m \times n)$ diagonal matrix with singular values $\lambda_i, i = 1, \ldots, m$ on its main diagonal, and V^T is an $(n \times n)$ rotation matrix. Formula (31.2) admits a nice physical interpretation, cf. Fig. 31.1. Unit-length velocity vectors $\|\dot{\boldsymbol{q}}\| = 1$, living in a (tangent) configuration space are rotated in this space, then transformed into a taskspace R^m. Some of the vectors are transformed into vectors collinear with canonical basis in R^m and their lengths are equal to $D_{ii} = \lambda_i$, $i = 1, \ldots, m$. Finally, the vectors are once more rotated, this time in the taskspace. The definition of manipulability and its relation with singular values follows

$$\mathrm{man}(\boldsymbol{q}) = \sqrt{\det(J(\boldsymbol{q})J^T(\boldsymbol{q}))} = \prod_{i=1}^{m} D_{ii}(\boldsymbol{q}) = \prod_{i=1}^{m} \lambda_i(\boldsymbol{q}). \tag{31.3}$$

A manipulability ellipsoid with semi-axes equal to λ_i describes geometric displacements in the taskspace corresponding to unit-length shifts $\Delta\boldsymbol{q}$ in the joint space applied at a current configuration. A basic algorithm optimizing manipulability that guarantees (when a singular configuration is not met on the way) straight-line motion towards the goal $\boldsymbol{x}_f$ in the taskspace is given by the Newton scheme [5]:

$$\boldsymbol{q}_{i+1} = \boldsymbol{q}_i + \alpha J^{\#}(\boldsymbol{q}_i)\left(\boldsymbol{x}_f - k(\boldsymbol{q}_i)\right) + \left(I - J^{\#}(\boldsymbol{q}_i)J(\boldsymbol{q}_i)\right) \frac{\partial \mathrm{man}(\boldsymbol{q})}{\partial \boldsymbol{q}}\bigg|_{\boldsymbol{q}_i}, \tag{31.4}$$

where an initial configuration $\boldsymbol{q}_0$ is given together with a positive scalar α effecting the convergence property of the algorithm and i denotes the iteration counter. $J^{\#}(\boldsymbol{q})$ in Eq. (31.4) stands for the Moore-Penrose pseudo-inverse of the matrix $J(\boldsymbol{q})$ and it can be expressed in terms of SVD matrices as follows:

$$J^{\#} = V(D^T)^{-1}U^T, \tag{31.5}$$

where $(D^T)^{-1}$ is a matrix $(n \times m)$ composed of transposed D with elements on its main diagonal D_{ii} inverted.

The original algorithm admitting not only the straight-line motion toward the goal goes as follows (the symbol $\leftarrow$ is reserved for substitution and $\|\cdot\|$ denotes the Euclidean norm):

Step 1 Input data: the initial configuration $\boldsymbol{q}_0$, forward kinematics $k(\boldsymbol{q})$, the goal point in the taskspace $\boldsymbol{x}_f$, ε - accuracy of reaching the goal. Set the current configuration $\boldsymbol{q}_c \leftarrow \boldsymbol{q}_0$ and add it to the resulting trajectory.

Step 2 Check the stop condition: if $\|\boldsymbol{x}_f - k(\boldsymbol{q}_c)\| < \varepsilon$ then complete the algorithm and output the resulting trajectory. Otherwise progress with Step 3.

Step 3 Using the SVD algorithm split $J(\boldsymbol{q}_c) = U(\boldsymbol{q}_c)D(\boldsymbol{q}_c)V^T(\boldsymbol{q}_c)$, and compute a modified direction towards the goal $\Delta\boldsymbol{x} = U^T(\boldsymbol{q}_c)(\boldsymbol{x}_f - k(\boldsymbol{q}_c))$, cf. Fig. 31.1. The purpose of this step is to express a real direction to the goal $\boldsymbol{x}_f - k(\boldsymbol{q}_c)$ at $\boldsymbol{q}_c$ in such a frame that the semi-axes of the manipulability ellipsoid coincide with the axes of the frame.

Step 4 To facilitate computations, it is desirable to transform $\Delta\boldsymbol{x} \in R^m$ with some negative coordinates possible into a frame with only nonnegative coordinates $\Delta\boldsymbol{y} \in R^m$. Therefore a square non-singular diagonal matrix $K = diag(k_1, \ldots, k_m)$ is determined with the element k_i equal to -1 if $sgn(\Delta\boldsymbol{x}_i) < 0$ and $+1$ otherwise, which satisfies $K\Delta\boldsymbol{x} = \Delta\boldsymbol{y}$.

Step 5 Manipulability ellipsoid is constructed for $\|\Delta\boldsymbol{q}\| = 1$. The shift can be too large (and accuracy of motion based on the Jacobian matrix deteriorates) or too small (and too many iterations should be performed) for generating desired shift $\Delta\boldsymbol{y}$. Therefore a one-dimensional optimization procedure is applied to determine the optimal value $\xi^\star$ of a coefficient $\xi \geq 0$ scaling the ellipsoid (with varied semi-axes $(\xi\lambda_1, \ldots, \xi\lambda_m)^T$). The procedure initialized with $\xi^\star = 0$, and $\Delta\boldsymbol{q}(\xi^\star) = 0$), goes as follows:

a) For a fixed value of ξ, find a tangent point $\boldsymbol{y}(\xi)$ of a family of spheres, with the radius varied, centered at $\Delta\boldsymbol{y}$ (a transformed goal) and a scaled ellipsoid with the semi-axes equal to $(\xi\lambda_1, \ldots, \xi\lambda_m)^T$. This sub-task is described in details in Section 31.3. Compute

$$\Delta\boldsymbol{q}(\xi) = V(D^T)^{-1}K^{-1}\boldsymbol{y}(\xi).$$

The last transformation K^{-1} moves back the target $\boldsymbol{y}(\xi)$ to the Δx space while the two others, $V(D^T)^{-1}$, transform it into displacements in the configuration space, cf. Fig. 31.1.

b) Check whether the solution is admissible (in the configuration and/or taskspace), i.e. not too extensive motions are applied to disrupt reliability of the Jacobian matrix around $\boldsymbol{q}_c$

$$\|\Delta\boldsymbol{q}(\xi)\| < \varepsilon_q, \qquad \|\boldsymbol{y}(\xi)\| < \varepsilon_x, \tag{31.6}$$

where $\varepsilon_q, \varepsilon_x$ are given constants. Continue with Step 5c if condition (31.6) is satisfied; otherwise go to Step 5d.

c) Check whether the solution $\Delta\boldsymbol{q}(\xi)$ is better than the current best one $\Delta\boldsymbol{q}(\xi^\star)$. Two variants to select $\Delta\boldsymbol{q}(\xi)$ are possible:
1: either if

$$\|\boldsymbol{x}_f - k(\boldsymbol{q}_c + \Delta\boldsymbol{q}(\xi))\| < \|\boldsymbol{x}_f - k(\boldsymbol{q}_c + \Delta\boldsymbol{q}(\xi^\star))\|, \tag{31.7}$$

2: or if

$$\|\boldsymbol{x}_f - k(\boldsymbol{q}_c + \Delta\boldsymbol{q}(\xi))\| < \|\boldsymbol{x}_f - k(\boldsymbol{q}_c)\| \tag{31.8}$$

and

$$\frac{\mathrm{man}(\boldsymbol{q}_c) + \mathrm{man}(\boldsymbol{q}_c + \Delta\boldsymbol{q}(\xi))}{\|\boldsymbol{x}_f - k(\boldsymbol{q}_c + \Delta\boldsymbol{q}(\xi))\|} > \frac{\mathrm{man}(\boldsymbol{q}_c) + \mathrm{man}(\boldsymbol{q}_c + \Delta\boldsymbol{q}(\xi^\star)))}{\|\boldsymbol{x}_f - k(\boldsymbol{q}_c + \Delta\boldsymbol{q}(\xi^\star)\|} \tag{31.9}$$

then $\xi^\star = \xi$, $\Delta\boldsymbol{q}(\xi^\star) = \Delta\boldsymbol{q}(\xi)$.

d) If the optimization process is completed, go to Step 6, otherwise select a next trial point by setting a new value of ξ and go back to Step 5a.

Step 6 Set the current next configuration $\boldsymbol{q}_c \leftarrow \boldsymbol{q}_c + \Delta\boldsymbol{q}(\xi^\star)$, and add this configuration to the resulting trajectory. Go to Step 2.

Some remarks concerning the algorithm follow:

1. In steps 5a–d) the user has to develop rules to generate trial points for ξ and to decide whether the optimization process is completed (for example ξ can be generated within a given range with a fixed step and the algorithm is stopped when the distance to the goal does not decrease reasonably fast).
2. The algorithm is based on the Jacobi matrix (i.e. linearization of kinematics at a current configuration $\boldsymbol{q}_c$). While moving the trial configuration far from the current one, the reliability of linearization decreases and convergence of the algorithm can be lost. To prevent this, Condition (31.6) was introduced.
3. In Step 5b it is decided whether $\Delta\boldsymbol{q}(\xi)$ (obtained via solving an elementary sphere-ellipsoid optimization task) is better or not than found previously. Condition (31.7) requires only that the current trial configuration $\boldsymbol{q}_c + \Delta\boldsymbol{q}(\xi)$ approaches the goal more effectively than the current best trail configuration. This condition guarantees the convergence property of the algorithm as the checking is performed on real shifts in the taskspace (based on kinematics) rather than on virtual ones based on the Jacobi matrix whose reliability deteriorates as the distance of the trial configuration from the current one $\boldsymbol{q}_c$ increases.
4. Condition (31.7) does not provide full information about manipulability along the path between $\boldsymbol{q}_c$ and $\boldsymbol{q}_c + \Delta\boldsymbol{q}(\xi)$ because it is based only on the first configuration

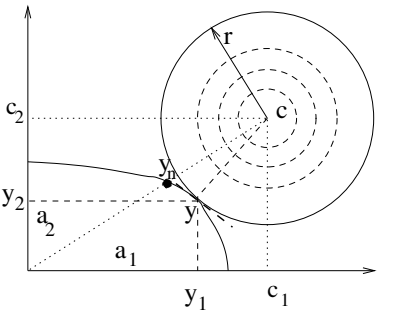


Fig. 31.2 A tangent point $\boldsymbol{y}$ of the ellipse-sphere pair and a point $\boldsymbol{y}_n$ being the goal for a basic Newton algorithm, $m = 2$

$\boldsymbol{q}_c$ of the segment. To some extent, this drawback is not present in the second, more computationally involved, variant given by Conditions (31.8), (31.9). While selecting the optimal configuration shift $\Delta\boldsymbol{q}(\xi)$ the averaged manipulability from $\boldsymbol{q}_c$ and $\boldsymbol{q}_c + \Delta\boldsymbol{q}(\xi)$ is taken into account and once again averaged (normalized) by dividing this value by the decrease of the distance to the goal.

31.3 A Tangent Point of an Ellipsoid and Spheres

The key sub-task of the algorithm, Step 5a, is to find a tangent point $\boldsymbol{y}$ of an m-dimensional ellipsoid (with semi-axes $a_i = \xi\lambda_i$, $i = 1,\ldots,m$, given)

$$e(\boldsymbol{y}) = \sum_{i=1}^{m} \frac{y_i^2}{a_i^2} - 1 = 0 \tag{31.10}$$

with a sphere of the minimal radius r

$$\sum_{i=1}^{m} (y_i - c_i)^2 = r^2$$

centered at a given point $\boldsymbol{c} = (c_1,\ldots,c_m)^T = \Delta\boldsymbol{y}(\xi)$, cf. Fig. 31.2, with $\forall_i c_i \geq 0$ (cf. Step 4). The existence and uniqueness of a solution is mathematically obvious due to convexity of an ellipsoid and a family of spheres (with the radius varied). The aforementioned formulation, although quite natural, does not facilitate finding the tangent point.

Therefore we prefer to search for $\boldsymbol{y} = (y_1,\ldots,y_m)^T$ lying on an ellipsoid where a normal line to the ellipsoid passes through the point $\boldsymbol{c}$. In this approach two geometric facts are used: 1) if two convex surfaces touch each other at a point $\boldsymbol{y}$, then normal lines to the surfaces are collinear there; 2) a normal line to any point placed on a sphere passes through its center. Using these facts, we get

$$c_i = y_i + \left.\frac{\partial e}{\partial y_i}\right|_{\boldsymbol{y}} \times t = y_i \left(1 + \frac{2}{a_i^2} \times t\right), \quad i = 1,\ldots,m, \tag{31.11}$$

where t determining r is searched for. Until now $\boldsymbol{c}$ satisfied $\forall_i c_i \geq 0$. To facilitate notations and not to lose generality, it will be assumed that

$$\forall_i c_i > 0. \tag{31.12}$$

Condition (31.12) seems to be restrictive but in fact is not. At first the case of $\forall_i c_i = 0$ is excluded as no motion is required to reach the goal. Using geometric arguments, it is easy to show that if some (but not all) coordinates c_i vanish, so do the corresponding coordinates of the tangent point $\boldsymbol{y}$. Consequently, the original task can be projected into a smaller dimensional problem with all non-zero coordinates. Let us go back to the task (31.11) with constraints (31.12). Further on, it is assumed that $\boldsymbol{c}$ does not belong to the ellipsoid because in this case the solution is trivial, $\boldsymbol{y} = \boldsymbol{c}$, $r = 0$. From Eq. (31.11) one can compute y_i, $i = 1, \ldots, m$ and substitute to Eq. (31.10) to get

$$F(t) = \sum_{i=1}^{m} \frac{a_i^2 c_i^2}{(a_i^2 + 2t)^2} - 1 = 0, \tag{31.13}$$

with only one variable t to determine. Obviously, when $\boldsymbol{c}$ is placed on the surface of the ellipsoid then $t = 0$. If $\boldsymbol{c}$ is placed outside/inside the ellipsoid then $t > 0 (t < 0)$. Equation (31.13) cannot be solved analytically, therefore a numerical approach should be used instead. First a domain of t will be determined. Because $\forall_i y_i \geq 0$ (due to $\forall_i c_i > 0$), from Eq. (31.11) the constraints on t can be obtained:

$$t \geq \max_{i=1}^{m} \left(-a_i^2/2\right). \tag{31.14}$$

On the other hand, at a tangent point the condition $\forall_i y_i \leq a_i$ has to be satisfied. This condition generates another constraint:

$$t \geq \max_{i=1}^{m} a_i \frac{c_i - a_i}{2} = \frac{1}{2} \max_{i=1}^{m} (-a_i^2 + c_i a_i) = t_{min}. \tag{31.15}$$

As $a_i, c_i > 0$, condition (31.15) is more restrictive than the previous one and it sets constraints on the lower bound $t_{\min}$ at which $F(t_{\min}) > 0$. Now the upper bound on t will be established. For $t \to \infty : F(t) \to -1$. So such value of t_0 is looked for that $F(t > t_0) < 0$. This value will be determined by the extra requirement that each expression under the sum in Eq. (31.13) is not larger than $1/m$. In this case the whole expression on $F(t)$ will not be larger than 0. This generates the following condition:

$$t \leq \max_{i=1}^{m} a_i \frac{\sqrt{m} c_i - a_i}{2} = t_{\max}. \tag{31.16}$$

Constraints (31.15), (31.16) (with auxiliary $\forall_i a_i, c_i > 0$) generate the domain of t, and its value satisfying Eq. (31.13) is determined numerically.

Finally, the case when the ellipsoid is degenerated, $\exists_i a_i = 0$ should be considered. In this case the manipulator is at a singular configuration and the Jacobian matrix needs to be modified to be non-singular. A well-known robust Jacobian matrix [5] replaces the Jacobian matrix at this particular configuration.

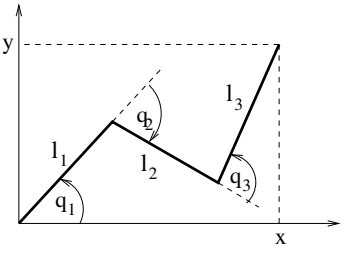


Fig. 31.3 3D planar pendulum

31.4 Simulations

Simulations of running the algorithm of locally optimal motion planning were carried out on the 3D planar pendulum shown in Fig. 31.3 with kinematics given by

$$x = l_1c_1 + l_2c_{12} + l_3c_{123}, \quad y = l_1s_1 + l_2s_{12} + l_3s_{123}. \tag{31.17}$$

The taskspace of the manipulator is a subset of the plane (x,y) $(m = 2)$, and the configuration is described by the vector $\boldsymbol{q} = (q_1,q_2,q_3)^T$ $(n = 3)$. In Eq. (31.17) a standard robotic convention was used to denote trigonometric functions, i.e. c_{12} stands for $\cos(q_1 + q_2)$, while s_{123} for $\sin(q_1 + q_2 + q_3)$. Manipulability (31.3) for the 3D planar pendulum (31.17) is given by expression

$$\mathrm{man}(\boldsymbol{q}) = \sqrt{l_1^2(l_2s_2 + l_3s_{23})^2 + l_3^2(l_1s_{23} + l_2s_3)^2 + l_2^2l_3^2s_3^2}. \tag{31.18}$$

An acceptable accuracy of reaching the goal was set to $\varepsilon = 0.003$, the upper bound on admissible configuration change in a single iteration, cf. Eq. (31.6), was set to $\Delta\boldsymbol{q} = 4^\circ$ and the geometric parameters (lengths of links) equal to $l_1 = l_2 = l_3 = 1$. Three tasks were considered: Task 1: $\boldsymbol{q}_0 = (0^\circ,0^\circ,45^\circ)^T$, $\boldsymbol{x}_f = (-1.71,0.29)^T$; Task 2: $\boldsymbol{q}_0 = (0^\circ,0^\circ,45^\circ)^T, \boldsymbol{x}_f = (-2.86,0.49)^T$; Task 3: $\boldsymbol{q}_0 = (10^\circ,20^\circ,20^\circ)^T, \boldsymbol{x}_f = (-2,-1)^T$.

In Figs. 31.4 and 31.5 the simulation results are presented. The independent variable $s \in [0,1]$ (cf. Figs. 31.4, 31.5a,b) was changed linearly in the interval $[0,1]$ as the manipulator progressed along the path from 0 to $len = \sum_{i=1}^{iter} \|\Delta\boldsymbol{q}_i\|$, where $iter$ is the total number of iterations. For Task 1, the basic Newton algorithm (31.4) generated a straight line segment in the taskspace and the length of the trajectory was equal to $len = 5.1$ [rd] (this value is correlated with the total energy spend on motion). For the same data the locally optimal algorithm generated a trajectory of the length $len = 2.72$ [rd] and its image, via forward kinematics, did not resemble a straight line segment, Fig. 31.4c1. The local manipulability optimization algorithm prefers shorter trajectories, Fig. 31.4a, although it had to leave areas with high manipulability values just to reach the goal point of the planning. Identical observations were made for Tasks 2 and 3 as well.

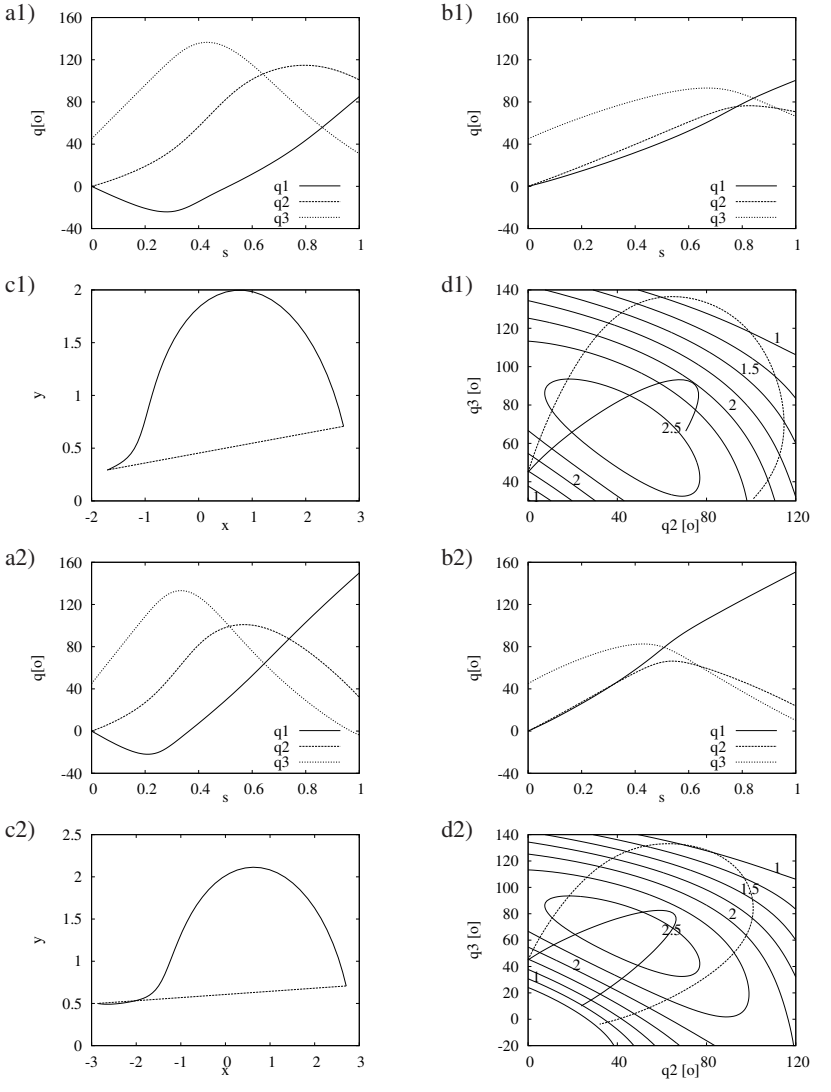


Fig. 31.4 Trajectories obtained with the basic Newton algorithm (a), and the locally optimal algorithm (b); c) a path on the *xy* plane drawn by the image of the trajectory generated by the locally optimal algorithm and the basic Newton algorithm (straight-line), d) the trajectory plot in q_2q_3 plane with layers of the manipulability function drawn. Task 1: (a1–d1), Task 2: (a2–d2)

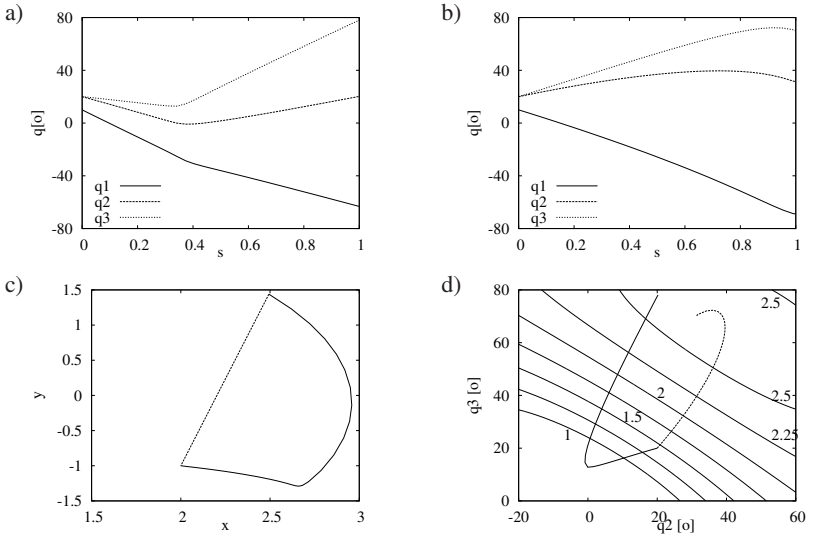


Fig. 31.5 Task 3: trajectories obtained with the basic Newton algorithm (a), and the locally optimal algorithm (b); c) a path on the *xy* plane drawn by the image of the trajectory generated by the locally optimal algorithm and the basic Newton algorithm (straight-line), d) the trajectory plot in q_2q_3 plane with layers of the manipulability function drawn

31.5 Conclusions

In this paper an original locally optimal motion planning w.r.t. the manipulability criterion function was presented. Although, due to the SVD procedure used, the algorithm is more computationally involved than the basic Newton algorithm, still it is computationally feasible as mostly analytical calculations are needed to establish the direction of motion at each configuration. The optimization provides significantly shorter trajectories (equivalently energy-efficient trajectories) as compared to the basic Newton algorithm of straight-line motion planning.

References

1. Bicchi, A., Prattichizzo, D., Melchiorri, C.: Force and dynamic manipulability for cooperating robots. In: Proc. Conf. on Intelligent Robotic Systems, pp. 1479–1484 (1991)
2. Barcio, B.T., Walker, I.D.: Impact ellipsoids and measures for robot manipulators. In: Proc. IEEE Conf. on Robotics and Automation, pp. 1588–1594 (1994)
3. Klein, C.A., Blaho, B.E.: Dexterity measures for the design and control of kinematically redundant manipulators. Int. Journ. of Robotic Research 6(2), 72–83 (1987)

4. Maciejewski, A.A., Klein, C.A.: The Singular Value Decomposition: Computation and Applications to Robotics. Int. Journ. of Robotic Research 8(6), 63–79 (1989)
5. Nakamura, Y.: Advanced Robotics: Redundancy and Optimization. Addison Wesley, New York (1991)
6. Yoshikawa, T.: Manipulability of robotic mechanisms. Int. Journ. of Robotic Research 4(2), 3–9 (1985)
7. Muszyński, R., Tchoń, K.: Dexterity ellipsoid for mobile robots. In: Proc. Conf. on Methods and Models in Automation and Robotics, Międzyzdroje, Poland, vol. 2, pp. 665–670 (2000)

Chapter 32
A Comparison Study of Discontinuous Control Algorithms for a Three-Link Nonholonomic Manipulator

Dariusz Pazderski, Bartłomiej Krysiak, and Krzysztof R. Kozłowski

Abstract. The paper is focused on selected discontinuous methods which are adapted to control of a three-link nonholonomic manipulator equipped with ball gears. The considered control solutions defined at the kinematic level are based on discontinuous transformations using the so-called sigma process or polar representation. In order to improve robustness to measurement noise a hybrid technique is introduced. The comparison of the algorithms is based on simulation results and takes into account their robustness to measurement errors, convergence rate as well as control effort.

32.1 Introduction

In this paper we deal with a control problem of a three-link manipulator (NHM3) equipped with ball gears (cf. [8]) which are used to transfer velocities via three separate joints. The manipulator belongs to a class of nonholonomic mechanical systems. It reveals strong nonlinear properties. In particular it is difficult to find some symmetry for the system – as a result it cannot be considered on a Lie group. Following Sørdalen *et al.* [8] we overcome this difficulty by transforming NHM3 kinematics to a first-order chained system (which is a system on Lie group). To extend the domain of NHM3 feasible configurations we define two alternative coordinate maps.

To solve the regulation control problem for the manipulator and to overcome Brockett's well-known obstruction we refer to controllers which are discontinuous at the desired configuration. From among different algorithms proposed so far two alternative solutions are chosen. The first algorithm is based on the idea proposed

Dariusz Pazderski · Bartłomiej Krysiak · Krzysztof Kozłowski
Chair of Control and Systems Engineering, Poznań University of Technology,
ul. Piotrowo 3a, 60-965 Poznań, Poland
email: {dariusz.pazderski,bartlomiej.krysiak,
krzysztof.kozlowski}@put.poznan.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 377–389.
springerlink.com

by Astolfi [2]. He showed that a nonholonomic system can be controlled with an asymptotic convergence by a proper choice of the coordinate system by using the sigma-process transformation which transforms the original continuous system to a discontinuous one. The second control approach is somehow related to Astolfi's concept, however it is based on the polar representation, which is singular at zero. In [1] Aicardi and others designed a controller which ensures asymptotic and fast error convergence to zero for a unicycle robot. Later this idea was used by Pazderski *et al.* in [4, 5] to control a first-order chained system and NHM3.

In our work we take into account the main disadvantage of discontinuous controllers at the desired point, namely their sensitivity to unmodeled dynamics and measurement noise. In fact these algorithms do not ensure stability in the sense of Lyapunov – this problem becomes crucial particularly in practice and should not be ignored. This issue was investigated by Lizárraga and others who formally proved unrobustness of some class of non-Lipshitz "stabilizers" to a particular type of disturbance [3]. To deal with this obstruction some hybrid techniques can be invoked – cf. [6], [7].

In this paper we propose a simple idea of a hybrid time-invariant controller taking into account the convergence phase and the stabilization phase near the desired point. Additionally we use gain scheduling for the algorithm based on the sigma-process to exclude the singularity at the origin. Next, we compare properties of selected discontinuous algorithms with respect to their sensitivities near the desired point in the configuration space. The theoretical statements are supported by simulation results which illustrate the performance of the controllers.

The novelty of the paper is mainly related to the new proposition of coordinate maps and the provided solutions of improving robustness of discontinuous control algorithms adapted for the nonholonomic manipulator.

32.2 Nonholonomic Manipulator Properties and Kinematic Transformation

We consider a three-link nonholonomic manipulator with the following kinematics [5]:

$$\Sigma_{NHM3}: \ \dot{q} = \underbrace{\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}}_{X_1} u_1 + \underbrace{\begin{bmatrix} 0 \\ \kappa_1 \sin q_1 \\ \kappa_2 \cos q_1 \cos q_2 \end{bmatrix}}_{X_2(q)} u_2, \tag{32.1}$$

where $q = [q_1 \ q_2 \ q_3]^{\mathrm{T}} \in \mathcal{Q}$ describes the independent coordinates (namely the angles of the joints), u_1 and u_2 denote the control inputs, X_1, X_2 define vector fields (vf.) with κ_1 and $\kappa_2 > 0$ being the gear ratios.

Referring to the Lie Algebra Rank Condition (LARC) it can be proved that an n-link nonholonomic manipulator equipped with ball gears can be controllable in the whole configuration space [8]. Considering the particular kinematics (32.1) and

recalling the formal analysis discussed in [5] one can find that the degree of non-holonomy for this system is not constant. Namely, in order to show the LARC for $\cos q_2 \neq 0$ it is sufficient to use vf. X_1, X_2 along with the first-order Lie bracket $[X_1, X_2]$, however for $\cos q_2 = 0$ the second-order Lie bracket $[X_2, [X_1, X_2]]$ should be involved instead of $[X_1, X_2]$.

Moreover, taking into account the control Lie algebra defined over $\mathbb{R}$ and generated by vf. X_1 and X_2 one can show that, in the given coordinates, its dimension is infinite. As a result any Lie group associated with the control Lie algebra can be defined. Consequently it is hard to find some symmetry for the control system Σ_{NHM3} which can be used for control design. However, the structure of the NHM3 kinematics (two input driftless affine system) and its controllability properties suggest that it can be locally (excluding configurations for which $\cos q_2 = 0$) considered as an equivalent control system to the first-order chained system (CS3) defined by

$$\Sigma_{CS3} : \dot{x} = \begin{bmatrix} v_1 \\ v_2 \\ x_1 v_2 \end{bmatrix}, \tag{32.2}$$

where $x = [x_1\ x_2\ x_3]^{\mathrm{T}}$ is a state and v_1, v_2 are input signals. In order to cover the largest set of $\mathscr{Q}$ we consider two different local coordinate transformations $x := F(q)$ with different domains. They are defined as follows:

$$F(q) = [F_1(q)\ F_2(q)\ F_3(q)]^{\mathrm{T}} := \begin{cases} F^I(q) & \text{for } q \in \mathscr{Q}^I \\ F^{II}(q) & \text{for } q \in \mathscr{Q}^{II} \end{cases}, \tag{32.3}$$

with

$$F^I(q) := \left[\frac{\kappa_1 \tan q_1}{\kappa_2 \cos^3 q_2}\ q_3\ \tan q_2\right]^{\mathrm{T}},\ F^{II}(q) := \left[\frac{\kappa_2 \cos^3 q_2}{\kappa_1 \tan q_1}\ \tan q_2\ q_3\right]^{\mathrm{T}} \tag{32.4}$$

and

$$\mathscr{Q}^I := \left\{q \in \mathscr{Q} :\ q_1 \in \left(-\pi, -\tfrac{\pi}{2}\right) \cup \left(-\tfrac{\pi}{2}, \tfrac{\pi}{2}\right) \cup \left(\tfrac{\pi}{2}, \pi\right),\ q_2 \in \left(-\tfrac{\pi}{2}, \tfrac{\pi}{2}\right),\ q_3 \in \mathbb{S}^1\right\}, \tag{32.5}$$

$$\mathscr{Q}^{II} := \left\{q \in \mathscr{Q} :\ q_1 \in (-\pi, 0) \cup (0, \pi),\ q_2 \in \left(-\tfrac{\pi}{2}, \tfrac{\pi}{2}\right),\ q_3 \in \mathbb{S}^1\right\} \tag{32.6}$$

being overlapped sets which denote restricted configuration. Considering the inverse maps of F^I and F^{II} one can prove that for the bounded $\|x\|$ configuration q does not escape from set $\mathscr{Q}^I$ or $\mathscr{Q}^{II}$ (assuming that initially $\cos q_2 \neq 0$).

Taking into account map (32.3) one can obtain the following associated input transformation:

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \frac{\partial F_1}{\partial q} X_1 & \frac{\partial F_1}{\partial q} X_2(q) \\ \frac{\partial F_2}{\partial q} X_1 & \frac{\partial F_2}{\partial q} X_2(q) \end{bmatrix}^{-1} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \tag{32.7}$$

which is well defined for the particular domain $\mathscr{Q}^I$ or $\mathscr{Q}^{II}$.

32.3 Control Algorithms

32.3.1 Problem Formulation

In this paper the following control problem is investigated.

Problem 32.1. Find bounded controls $u_1(t)$ and $u_2(t)$ such that for the initial configuration $q(0) \in \mathcal{Q}^{\chi}$ and the desired configuration $q_d \in \mathcal{Q}^{\chi}$ ($\chi = I$ or II) the regulation error $e := q - q_d$ remains bounded and converges to an assumed neighborhood of zero such that $\lim_{t\to\infty} \|e(t)\| \leq \varepsilon$, where $\varepsilon > 0$ is an arbitrary small constant.

Remark 32.1. It is assumed that the initial and desired configurations have to be in the same restricted set. However, even if this assumption is not satisfied one can take advantage of the fact that $\mathcal{Q}^I \cap \mathcal{Q}^{II} \neq \emptyset$ and define an additional desired point $\bar{q}_d$ such that $\bar{q}_d \in \mathcal{Q}^I \cap \mathcal{Q}^{II}$. This gives the possibility to achieve any point $q_d \in \mathcal{Q}^I \cup \mathcal{Q}^{II}$.

Problem 1 is formally defined with respect to the *deterministic case* assuming that full knowledge of the manipulator dynamics (kinematics) and the feedback structure is given in the continuous time-domain. Then, theoretically, the configuration error in the steady state can be made arbitrary small. However, motivated by practical issues we additionally require that the algorithm should guarantee some robustness to the small measurement noise which affects the input feedback signal (in practice it is not possible to determine the real configuration q with any precision). Then one has to use signal $q^{\#} := q + r$ instead of q to determine the control feedback, with $r \in \mathbb{R}^3$ being additive bounded measurement noise. In this case it is assumed that the configuration error converges to some neighborhood of zero with the radius dependent on the noise magnitude (namely the radius ε cannot be made arbitrary small).

The control algorithms for the NHM3 considered in this paper take advantage of the coordinate and input transformations defined by Eqs. (32.3) and (32.7). Hence, the control solutions are directly designed for the CS3. In order to do that we introduce a transformed desired state $x_d = [x_{d1}\ x_{d2}\ x_{d3}]^{\mathrm{T}} := F(q_d)$.

In order to design the controller we refer to the following *left-invariant* group operation:

$$y_1 \circ y_2 = \begin{bmatrix} y_{11} + y_{21} \\ y_{12} + y_{22} \\ y_{13} + y_{23} + y_{11} y_{22} \end{bmatrix}, \tag{32.8}$$

with $y_i = [y_{i1}\ y_{i2}\ y_{i3}]^{\mathrm{T}}$ being an element of the Lie group, where $i = 1, 2$. Utilizing the inverse element of the group, the auxiliary control error can be defined by

$$\tilde{x} := \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ \tilde{x}_3 \end{bmatrix} = \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}^* \end{bmatrix} = x_d^{-1} \circ x = \begin{bmatrix} x_1 - x_{d1} \\ x_2 - x_{d2} \\ x_3 - x_{d3} + x_{d1}(x_{d2} - x_2) \end{bmatrix}. \tag{32.9}$$

As a result of the group symmetry, the open-loop error dynamics has a similar form to the original system Σ_{CS3} independent on chosen x_d, namely:

$$\Sigma^* : \dot{\tilde{x}} = \begin{bmatrix} v_1 \\ v_2 \\ \tilde{x}_1 v_2 \end{bmatrix}. \tag{32.10}$$

It is straightforward to verify that $(\tilde{x} \to 0) \Rightarrow ((x - x_d) \to 0)$. Consequently, assuming that $q, q_d \in \mathcal{Q}^\chi$ implies that $((x - x_d) \to 0) \Rightarrow (e = 0)$.

In this paper we deal with discontinuous controllers in order to overcome Brockett's obstruction which affects the regulation control problem of any nonholonomic system. Considering the problem of stabilization we take into account the following stages of control:

- *convergence (transient) stage* – it is required that the error trajectory converges to some vicinity of zero with good dynamic quality,
- *local stabilization stage* – the configuration is locally stabilized with some robustness to the noise.

We assume that the first stage is realized by the algorithm with singularity at the desired point, however the second stage is supported by a smooth local controller.

32.3.2 Discontinuous Algorithms for Convergence Stage

32.3.2.1 Algorithm A1 – Almost-Stabilizer Based on Sigma-Process

Let us assume that $\tilde{x}_1 \neq 0$ and consider a non-smooth change of coordinates given by (compare [2])

$$z := \left[\tilde{x}_1 \; \tilde{x}_2 \; \frac{\tilde{x}_3}{\tilde{x}_1}\right]^{\mathrm{T}}. \tag{32.11}$$

Taking the time derivative of (32.11) gives

$$\dot{z} = \begin{bmatrix} v_1 \\ v_2 \\ v_2 - z_3 \frac{v_1}{z_1} \end{bmatrix}. \tag{32.12}$$

Next, assuming that

$$v_1 := \mu z_1, \tag{32.13}$$

with $\mu \in \mathbb{R}$ being some non-zero gain and substituting (32.13) in (32.12) gives the following linear system

$$\tilde{\Sigma}^1 : \dot{z}_1 = \mu z_1, \tag{32.14}$$

$$\tilde{\Sigma}^* : \dot{z}^* = \begin{bmatrix} 0 & 0 \\ 0 & -\mu \end{bmatrix} z^* + \begin{bmatrix} 1 \\ 1 \end{bmatrix} v_2, \tag{32.15}$$

where $z^* := [z_2 \; z_3]^{\mathrm{T}}$.

In order to achieve exponential convergence of z^* to zero the following proposition can be considered.

Proposition 32.1. *Applying the static state linear feedback given by*

$$v_2 := Kz^*, \tag{32.16}$$

where $K := [k_2\ k_3]$ *with* $k_2 = -\lambda_0^2/\mu$ *and* $k_3 = \mu - 2\lambda_0 + \lambda_0^2/\mu$ *being gain coefficients while* $\lambda_0 > 0$ *determines the convergence rate and* $\mu \neq 0$*, to the dynamic system* $\tilde{\Sigma}^*$ *ensures that trajectory* $z^*(t)$ *exponentially approaches zero.*

Proof. The closed loop dynamics of system $\tilde{\Sigma}^*$ can be written as follows:

$$\dot{z}^* = Hz^*, \tag{32.17}$$

where $H := \begin{bmatrix} k_2 & k_3 \\ k_2 & k_3 - \mu \end{bmatrix}$. It can be shown that using gains according to Proposition 32.1 implies that H is a Hurwitz-stable matrix. As a result one can find the Lyapunov function given by

$$V := z^{*\mathrm{T}} P z^*, \tag{32.18}$$

where $P \in \mathbb{R}^{2\times 2}$ is some positive definite matrix. □

Stabilization of system $\tilde{\Sigma}^1$ is trivial – in view of (32.14) one can select $\mu < 0$ in order to achieve exponential stability. However, considering the singularity of the algorithm this choice does not become obvious. Basically, when z_1 approaches zero the algorithm approaches singularity as a result of the non-smooth coordinate change (32.11). Then, one can be aware of the following implication:

$$(\forall \varepsilon > 0, |\tilde{x}_3| \leq \varepsilon, \tilde{x}_1 \to 0) \Rightarrow (z_3 \to \pm\infty) \Rightarrow |v_2| \notin \mathscr{L}_\infty. \tag{32.19}$$

This property can be seen as a serious weakness of the considered approach. In order to overcome the singularity at $\tilde{x}_1 = 0$ we introduce *gain scheduling* by appropriately decreasing the gains to zero. Accordingly, we define a continuous switching scalar function $\sigma : \mathbb{R} \to [0,\ 1)$ given by $\sigma := \frac{|\tilde{x}_1|}{|\tilde{x}_1|+\delta}$, where $\delta > 0$ is a small positive parameter. Next, we use function σ for gain scaling given as follows:

$$v_1 = \sigma \mu z_1, \tag{32.20}$$

$$v_2 = \sigma K z^*. \tag{32.21}$$

Taking into account that $\sigma k_3 z_3 = \frac{|\tilde{x}_1|}{|\tilde{x}_1|+\delta} k_3 z_3 = \frac{\tilde{x}_3 \mathrm{sgn}(\tilde{x}_1)}{|\tilde{x}_1|+\delta} k_3$ the singularity at $\tilde{x}_1 = 0$ is overcome. The side-effect of restoring the Lipschitz continuity of the algorithm is related to the appearance of a local minimum at $z_1 = 0$. Since $z_1 = 0$ implies $\sigma = 0$ and $v_2 = 0$ (system $\tilde{\Sigma}^*$ is no longer controllable for $\mu = 0$), states z_2 and z_3 are not stabilized. Hence, the necessary condition for the asymptotic convergence of z^* is $z_1(0) \neq 0$. In order to satisfy this requirement one can use the following switching algorithm.

Proposition 32.2. *Let ε_1 and ε_2 be positive parameters such that $\varepsilon_1 < \varepsilon_2$. Assume that $\bar{V}$ is the Lyapunov function for the closed-loop system (32.17) derived for $\mu = -k < 0$, where $k > 0$ is a positive defined parameter. Then, for $\bar{V} > \varepsilon_1$ use controls (32.13) and (32.21) with $\mu = k > 0$ until $V > \varepsilon_2$. Otherwise use (32.20) and (32.21) with $\mu = -k < 0$.*

Proof. Assume that initially $\bar{V}(0) > \varepsilon_1$. Then one can show that $\|z^*\|$ decreases (however, not monotonically) while z_1 increases. Consequently, at some time $t = t_1 > 0$ function $\bar{V}$ becomes less than the assumed constant ε_2. At the second stage (with $\mu = -k < 0$) $\bar{V}$ and $|z_1|$ are strictly decreasing and converge to zero (however as a result of gain scheduling the convergence of function is no longer $\bar{V}$ exponential). □

32.3.2.2 Algorithm A2 – Almost-Stabilizer Based on Polar Representation

This control algorithm is based on the concept given by Aicardi *et al.* [1]. Later this idea was discussed by Pazderski and others [4, 5]. Because of the restriction on the paper length we omit most of the details concerning this control scheme.

Let us define polar coordinates as $\alpha := \operatorname{atan2}(\tilde{x}_3, \tilde{x}_2)$ and $\rho := \|\tilde{x}^*\|$. The control law which we consider is similar to that presented in [4]. However, in comparison to the original proposition we scale the control input by $\cos^2\alpha$ to avoid signal unboundedness at the point for which $\tilde{x}_2 = 0$.

Proposition 32.3. *Assuming that $|\alpha(0)| \neq \frac{\pi}{2}$ and $\rho > 0$, the control defined as*

$$v_1(t) = -k_2 \cos^2\alpha e_\gamma - k_1 \cos\gamma \left(\sin\gamma - \frac{1}{h} \sin\alpha \cos\beta \right), \tag{32.22}$$

$$v_2(t) = -k_1 \cos\gamma \cos\beta \cos^2\alpha \cdot \rho, \tag{32.23}$$

where $e_\gamma := \frac{\sin\gamma}{\cos\alpha\cos\beta}$, $\beta := \operatorname{atan}(\tilde{x}_1) \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, $\gamma := \beta - \alpha \in \left(-\frac{3}{2}\pi, \frac{3}{2}\pi\right)$ and k_1, k_2, and $h > 0$ are design parameters, ensures convergence of $\rho(t)$ and $\beta(t)$ to zero.

The proof of the convergence can be obtained based on [4] and it is not recalled here.

It is worth noting that this control law is well defined only for $\rho > 0$. At $\rho = 0$ the polar representation is singular and it is not possible to determine the value of α. However, considering Eq. (32.22) it can be proved that $|v_1|$ and $|v_2| \in \mathscr{L}_\infty$ for any ρ and α. It is a clear advantage over the discontinuous transformation based on the sigma-process.

Additionally one should exclude the initial point such that $\tilde{x}_2(0) = 0$. At such an initial condition a local minimum appears. This problem can be relatively easy solved using an idea similar to that presented in proposition 32.2.

32.3.3 Hybrid Controller

32.3.3.1 Local Smooth Controller

Locally the control problem can be interpreted as the requirement of error boundedness. This means that if the error is in some neighborhood of zero it should not escape from it. To be more specific, we require that norm $\|\tilde{x}^*\|$ should not increase (however it may decrease).

Proposition 32.4. *Applying the control law given by*

$$v_1 = -\bar{k}_1 \tilde{x}_1,\ v_2 = -\bar{k}_2 \left[1\ \tilde{x}_1\right] \tilde{x}^*, \tag{32.24}$$

where $\bar{k}_1$ and $\bar{k}_2 > 0$ are positive gains, to the chained system (32.10) ensures that

$$\forall \tilde{x}(0) \in \mathbb{R}^3 \ \lim_{t\to\infty} \tilde{x}_1(t) = 0 \text{ and } \lim_{t\to\infty} \|\tilde{x}^*(t)\| \leq \|\tilde{x}^*(0)\|.$$

Proof. The exponential convergence with respect to variable $\tilde{x}_1$ is clear. In order to show that $\|\tilde{x}^*(t)\|$ does not increase we consider the following positive definite quadratic form: $V := \frac{1}{2}\tilde{x}^{*\mathrm{T}}\tilde{x}^*$. Taking the time derivative of V and using the closed-loop control system with controls (32.24) we have: $\dot{V} = -\bar{k}_2 \tilde{x}^{*\mathrm{T}} P \tilde{x}^*$ with P being a positive semi-definite matrix. As a result $\dot{V} \leq 0$, which implies that V is a non-increasing function. □

32.3.3.2 Overall Algorithm

The hybrid controller proposed here can be formulated as follows.

Proposition 32.5. *The control law defined as follows:*

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{cases} (32.24) & \text{for } \|\tilde{x}^*\| \leq \tilde{\varepsilon} \\ \text{use algorithm A1 or A2} & \text{for } \|\tilde{x}^*\| > \tilde{\varepsilon} \end{cases}, \tag{32.25}$$

where $\tilde{\varepsilon} > 0$, ensures that control problem 32.1 is solved.

Proof. The stability proof of the hybrid algorithm is considered for the deterministic case assuming that measurement noise $r \equiv 0$. It is assumed that $\|\tilde{x}^*(0)\| > \tilde{\varepsilon}$. Taking into account that the discontinuous algorithms A1 and A2 ensure that the Euclidean norm of the trajectory error $\tilde{x}(t)$ converges to zero, there exists a finite time $t_1 > 0$ such that $\|\tilde{x}^*(t_1)\| = \tilde{\varepsilon}$. Then the local controller is selected. It ensures that $\forall t > t_1 \ \|\tilde{x}^*(t)\| \leq \tilde{\varepsilon}$. Referring to the inverse map of (32.3) we have $\forall t > t_1 \ \|e(t)\| \leq \varepsilon$. □

It should be noted that for the deterministic case switching between the convergence and stabilization stages appears only once. However, taking into account the measurement noise, the switching phenomenon may occur several times. Then one should choose $\tilde{\varepsilon}$ to be high enough with respect to the noise level and the desired point (notice that ε and $\tilde{\varepsilon}$ are scaled via coordinate transformation (32.3)).

32.4 Simulation Results

In this section we present results of numerical simulations assuming two different initial and desired configurations of the manipulator:

P1: $q(0), q_d \in \mathcal{Q}^I$: $q(0) = [10\ -80\ -90]^{\mathrm{T}}$ deg, $q_d = [-60\ 60\ 120]^{\mathrm{T}}$ deg,
P2: $q(0), q_d \in \mathcal{Q}^{II}$: $q(0) = [10\ -80\ -90]^{\mathrm{T}}$ deg, $q_d = [90\ 60\ 120]^{\mathrm{T}}$ deg.

Transformation F^I is used for posture P1 and transformation F^{II} is used for posture P2. We will use notation A1 for the algorithm presented in Section 32.3.2.1 and A2 for the algorithm presented in Section 32.3.2.2. The control feedback is realized based on the simulated measurement signal $q^{\#}$ affected by uniformly distributed white noise (with zero mean value) with variance 10^{-6} rad^2. The control inputs are saturated to meet the following constraints: $|u_1|, |u_2| \leq 10$ rad/s (the nominal control input generated by the controller is scaled to satisfy the constraints).

The parameters of the discontinuous algorithms A1 and A2 are selected as

A1: $\lambda_0 = 2, k = 1, \delta = 0.1$,
A2: $k_1 = 1.5, k_2 = 4, h = 0.5, m = 3$,

while the local controller is parametrized by $\bar{k} = 1$ and $\bar{k}_2 = 2$. The condition of switching is chosen by $\tilde{\varepsilon} = 0.05$.

Simulations were conducted taking into account the following three scenarios:

S1: without the hybrid controller in a long period of time, posture P1 is used,
S2: with the use of the hybrid controller, posture P1 is used,
S2: with the use of the hybrid controller, posture P2 is used.

The results of simulations S1 for which the local controller was not used are presented in Figs. 32.1–32.2. It can be concluded that the configuration errors do not converge to zero, however they are bounded to some neighborhood of zero. Taking into account the time plots of the control input one can observe permanent chattering phenomena (the plots were generated for a long period of time to show that the chattering does not vanish). This comes out from the fact that the discontinuous stabilizers A1 and A2 are not real stabilizers and they are extremely prone to measurement noise. Moreover, it can be seen that for algorithm A1 control input u_2 is predominant, while for algorithm A2 input u_1 is mainly responsible for convergence. This observation corresponds to the design assumptions taken for both algorithms.

In the second simulation (S2) the hybrid controller is used – the results are presented in Figs. 32.3–32.5. In this case when the configuration errors are less than the assumed constant, the control law is switched from algorithm A1 (or A2) to the local continuous controller. The configuration errors are bounded to values smaller than 10^{-3} rad (for simulation S1 they were bounded to values smaller than 10^{-1} rad). Comparing the time plots of the control signal it can be seen that in this case the sensitivity to noise is significantly reduced.

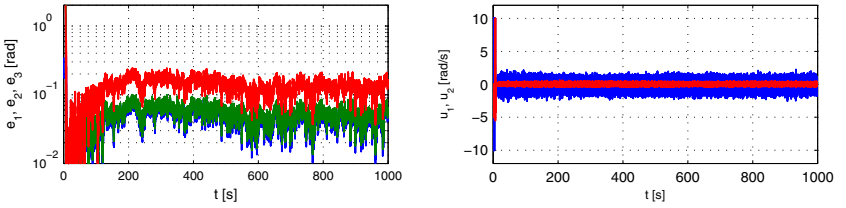


Fig. 32.1 S1: algorithm A1 – time plots of errors (●e_1, ●e_2, ●e_3) (on the left) and control inputs (●u_1, ●u_2) (on the right)

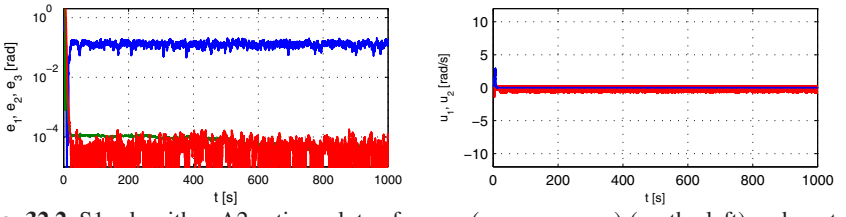


Fig. 32.2 S1: algorithm A2 – time plots of errors (●e_1, ●e_2, ●e_3) (on the left) and control inputs (●u_1, ●u_2) (on the right)

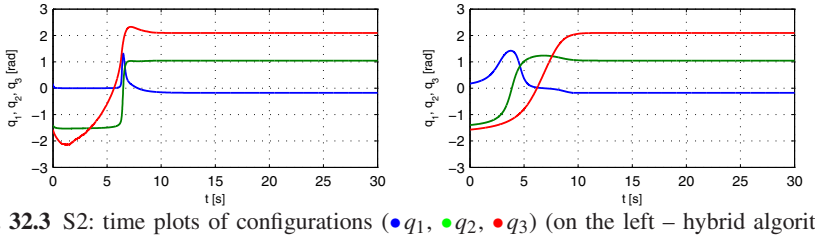


Fig. 32.3 S2: time plots of configurations (●q_1, ●q_2, ●q_3) (on the left – hybrid algorithm using control A1, on the right – algorithm A2)

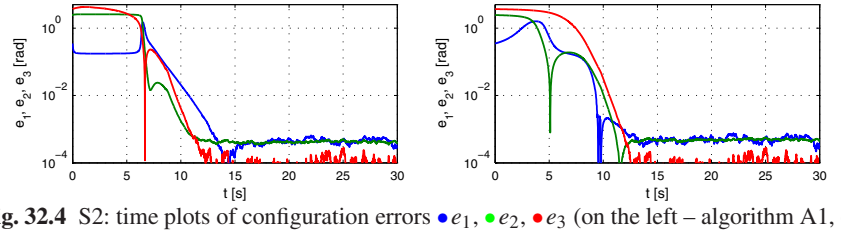


Fig. 32.4 S2: time plots of configuration errors ●e_1, ●e_2, ●e_3 (on the left – algorithm A1, on the right – algorithm A2)

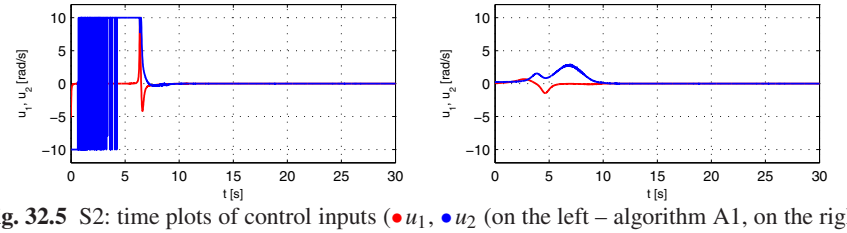


Fig. 32.5 S2: time plots of control inputs (●u_1, ●u_2 (on the left – algorithm A1, on the right – algorithm A2)

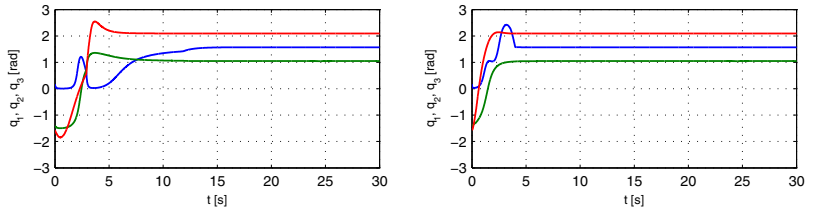


Fig. 32.6 S3: time plots of configurations $\bullet\, q_1$, $\bullet\, q_2$, $\bullet\, q_3$ (on the left – hybrid algorithm using control A1, on the right – algorithm A2)

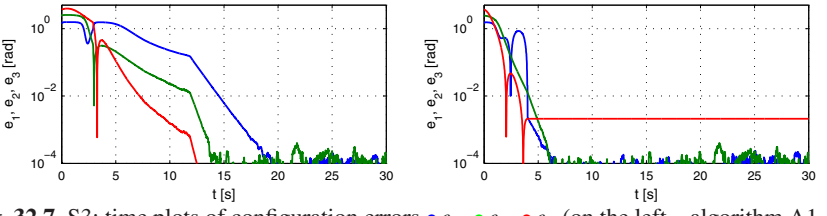


Fig. 32.7 S3: time plots of configuration errors $\bullet\, e_1$, $\bullet\, e_2$, $\bullet\, e_3$ (on the left – algorithm A1, on the right – algorithm A2)

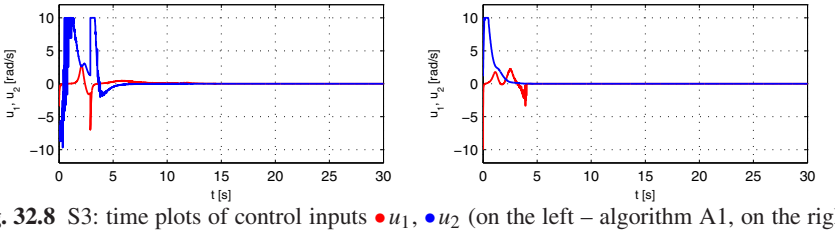


Fig. 32.8 S3: time plots of control inputs $\bullet\, u_1$, $\bullet\, u_2$ (on the left – algorithm A1, on the right – algorithm A2)

Simulations S1 and S2 were carried out for posture P1 with the use of the transformation described by $F^I(q)$. In simulation S3 posture P2 is used with the transformation given by $F^{II}(q)$ – the results are given in Figs. 32.6–32.8. Comparing the time plots obtained in simulation S2 and S3 we can state that the regulation time is smaller in case S3. However, this observation is local and it is strictly related to the chosen desired points.

It should be stated that during the simulation research a significant influence of the initial and desired configuration on the stabilization process was recorded and it is hard to conclude which transformation map (F_I or F_{II}) provides better convergence rate of the configuration error. Moreover, choosing the desired configuration near the boundary of sets $\mathcal{Q}^I$ or $\mathcal{Q}^{II}$ can decrease the convergence rate and robustness to noise (notice that near the boundary of $\mathcal{Q}^\chi$ the nonlinearity of the coordinate transformations F^I and F^{II} significantly increases and the Euclidean norm of the transformed state may achieve high magnitudes). This subject needs further investigation.

Comparing the results provided by algorithms A1 and A2 we can conclude that the controller based on the polar representation in general provides better results

and is less sensitive to noise (in particular it is relatively easy to guarantee that the control input is bounded). It should be emphasized that the nominal version of controller A1 (for $\delta = 0$) cannot be used alone (without the local controller), as a result of the singularity present at the origin.

32.5 Conclusion

The problem of robust stabilization of nonholonomic systems becomes an important issue in practice. In this paper we consider adaptation of discontinuous techniques in order to stabilize the configuration of a nonholonomic manipulator with ball gears. To extend the subset of feasible configurations two coordinate transformations have been defined (it is an extension of the result given by Sørdalen and others [8]).

To improve robustness to measurement noise a hybrid control solution has been considered and gain scheduling used in order to overcome the singularity at the desired point. Numerical simulation results confirm the theoretical expectation.

It has been verified that the properties of the controller are strictly related to the chosen desired points as a result of the highly nonlinear transformation. This issue is a side-effect of transforming the NHM3 kinematics to the chained form.

Still an open theoretical research problem of control of an NHM3 system is related to stabilization of the configuration at the points where $\cos q_2 = 0$. Another important issue is possibility of controlling the NHM3 in the original coordinates (some results have been reported in [5]).

Acknowledgements. This work was supported under university grant No. 93/193/11 DS-MK and the Ministry of Science and Higher Education grant No. N N514 299735.

References

1. Aicardi, M., Casalino, G., Bicchi, A., Balestrino, A.: Closed loop steering of unicycle-like vehicles via Lyapunov techniques. IEEE Robotics and Automation Magazine 2, 27–35 (1995)
2. Astolfi, A.: Discontinuous control of nonholonomic systems. System & Control Letters 27, 37–45 (1996)
3. Lizárraga, D., Morin, P., Samson, C.: Non-robustness of continuous homogeneous stabilizers for affine control systems. In: Proc. of Conference on Decision and Control, Phoenix, USA, pp. 855–860 (1999)
4. Pazderski, D., Kozłowski, Tar, J.: Discontinuous stabilizer of the first order chained system using polar-like coordinates transformation. In: Proceedings of European Control Conference, Budapest, pp. 2751–2756 (2009)
5. Pazderski, D., Kozłowski, K., Krysiak, K.: Nonsmooth stabilizer for three link nonholonomic manipulator using polar-like coordinate representation. LNCIS, pp. 35–44. Springer, Heidelberg (2009)

6. Pomet, J.B., Thuilot, B., Bastin, G., Campion, G.: A hybrid strategy for the feedback stabilization of nonholonomic mobile robots. In: Proceedings of the 1992 IEEE International Conference on Robotics and Automation, Nice, France, pp. 129–134 (1992)
7. Prieur, C., Astolfi, A.: Robust stabilization of chained systems via hybrid control. IEEE Transactions on Automatic Control 48(10), 1768–1772 (2003)
8. Sørdalen, O.J., Nakamura, Y., Chung, W.: Design of a nonholonomic manipulator. In: Proc. of the IEEE Int. Conf. on Robotics and Automation, pp. 8–13 (1994)

Part VII
Rehabilitation Robotics

Chapter 33
Development of Knee Joint Robot with Flexion, Extension and Rotation Movements – Experiments on Imitation of Knee Joint Movement of Healthy and Disable Persons

Yoshifumi Morita, Yusuke Hayashi, Tatsuya Hirano, Hiroyuki Ukai, Kouji Sanaka, and Keiko Takao

Abstract. We have developed a knee joint robot as an educational simulation tool for students becoming physical therapists or occupational therapists. The knee joint robot (Knee Robo) has two degrees-of-freedom to simulate both flexion/extension movement and rotation movement of a human knee joint. This paper presents knee joint models of healthy and disabled persons for the Knee Robo. The Knee Robo can simulate screw home movement (SHM) in a human knee joint. Moreover, the Knee Robo can simulate knee joint movements of not only a healthy person but also a patient with knee joint troubles, such as range of motion (ROM) trouble, contracture, rigidity, spasticity and so on. The effectiveness of the knee joint models and the control algorithms have been verified experimentally.

33.1 Introduction

In Japan the number of schools for students becoming a physical therapist (PT) or an occupational therapist (OT) is increasing. However, several problems concerning the educational effect have been pointed out. One of them is the shortage of experience of clinical training in medical institutions. During clinical training, the students learn patient troubles, manual training/testing techniques and so on.

Before clinical training, in the schools the students learn manual training/testing techniques, such as manual muscle training, manual muscle testing, range of motion testing and so on. In order to deepen the understanding the students play the role of

Yoshifumi Morita · Yusuke Hayashi · Tatsuya Hirano · Hiroyuki Ukai
Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya, Aichi 4668555, Japan
e-mail: {morita,ukai.hiroyuki}@nitech.ac.jp

Kouji Sanaka
Biological Mechanics Laboratory, Aichi, Japan

Keiko Takao
Harvest Medical Welfare College, Kobe, Japan

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 393–402.
springerlink.com

a therapist and a patient by turns mutually, and repeat the skill training of manual training/testing techniques to a healthy student instead of a patient. However, the students cannot experience patients' troubles before clinical training. Therefore, a patient robot imitating patients' troubles is necessary for the students to experience patient's troubles virtually and to learn the manual training/testing techniques before clinical training.

In a previous work of another research group, Masutani et al. have developed a patient leg robot as an educational training tool for students becoming PT or OT. By introducing the robot to the school, it is shown that the students' motivation went up [1]. However, the mechanism of the robot has only one degree-of-freedom in the knee joint. This mechanism is not enough to imitate actual troubles of knee joints. In [2] a leg-robot for demonstration of spastic movements of brain-injured patients has been presented.

We have developed a knee joint robot as an educational simulation tool of human knee joint movement [3, 4]. The mechanism is designed on the basis of the idea that the human knee joint movement consists of three kinds of movement, namely "sliding", "rolling", and "coming off". We have designed the optimal arrangement of four pulleys in the wire drive system by introducing performance indices [4]. In addition we have designed control algorithms and knee joint models to imitate three kinds of human knee joint troubles by using only the flexion/extension motion mechanism of the Knee Robo, and verified the effectiveness by fundamental experiments.

In this paper we present the design of knee joint models and control algorithms in which not only the flexion/extension motion mechanism but also the rotation motion mechanism of the Knee Robo are considered. The models can imitate screw home movement, range of motion (ROM) trouble, lead pipe rigidity, cogwheel rigidity, spasticity, contracture and so on. The effectiveness is verified experimentally.

33.2 Knee Joint Robot [2, 3]

We have developed the knee joint robot (Knee Robo) as shown in Fig. 33.1. The Knee Robo is used as a simulator to feel resistance of knee joint passive movement. Then the Knee Robo does not move automatically. Only a subject can move the lower leg of the Knee Robo. When a subject holds the femur of the Knee Robo and extends and retracts the lower leg of the Knee Robo, as shown in Fig. 33.2, the subject feels various resistance of normal and disabled knee joint movements.

A lower limb of a human being consists of a femur, a lower leg, and a knee joint. Human knee joint movement consists of flexion/extension movement and rotation movement. When a human being extends his/her lower leg from the bended position to the full extended position on the seating posture, the tibia rotates outward slightly at the vicinity of the full extended position. This movement is called screw home movement (SHM). Therefore, students should pay attention to this movement in manual testing and training on patients.

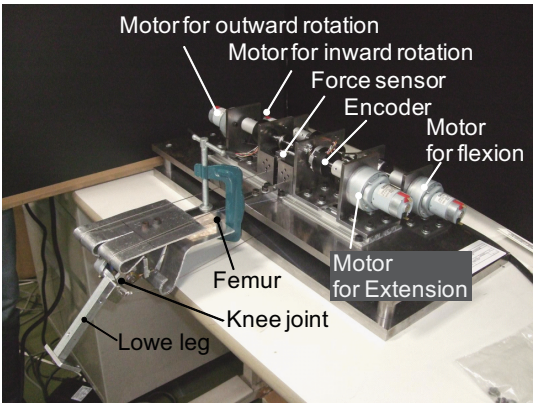


Fig. 33.1 Knee Robo

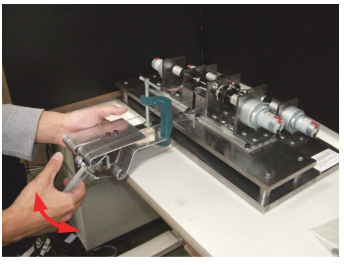


Fig. 33.2 Training scene of manual testing technique

The Knee Robo consists of a link corresponding to a femur, a link corresponding to a tibia, and a knee joint. The Knee Robo simulates a left leg. The Knee Robo is three fourths of sizes compared with a standard adult male.

The Knee Robo is driven by four motors and four wires corresponding to muscles. The four motors are used for outward rotation, inward rotation, flexion, and extension. The motor angles are measured with rotary encoders, and the wire tensions with force sensors. Four guide pulleys are used for each wire. It can be noted in Fig. 33.1 that guide plates rather than guide pulleys are used. Since the Knee Robo has two degrees of freedom, various passive movements of a human knee joint can be imitated, which is our original idea and advantage.

33.3 Models of Knee Joint Movement

33.3.1 Educational Effect of Knee Robo

The structure and movement of a knee joint are very complicated. By introducing the Knee Robo to schools improvement of educational effects is expected. The educational effects are as follows:

1. Students learn the structure and movement of normal and disabled knee joints.
2. Students learn the prohibited operation in leg passive movement in manual testing and training.
3. Student learn the manual training/testing technique for knee joint troubles.

The Knee Robo can imitate resistance of a knee joint during passive leg movement. In the Knee Robo, ROM trouble, lead pipe rigidity, cogwheel rigidity, spasticity and contracture can be simulated as reproducible knee joint troubles.

33.3.2 Healthy Knee Joint Movement

Models of a healthy human knee joint movement during passive leg movement are derived for the Knee Robo. Let $\theta_{FE}(t)$ and $\theta_R(t)$ denote the flexion/extension angle and the rotation angle. Let $\tau_E(t)$, $\tau_F(t)$, $\tau_{OR}(t)$, and $\tau_{IR}(t)$ denote the extension torque, the flexion torque, the outward rotation torque, and the inward rotation torque, respectively. Let $\theta_{shm}(t)$ denote the desired rotation angle to imitate SHM. We assume that in the full extended knee position, the flexion/extension angle $\theta_{FE}(t)$ is equal to zero, and the rotation angle $\theta_R(t)$ is equal to θ_{shmb}. When the flexion/extension angle $\theta_{FE}(t)$ is equal to θ_{shm0}, the rotation angle $\theta_R(t)$ is equal to zero. The knee joint model of a healthy person including SHM is represented as follows:

$$\tau_E(t) = \tau_{basic} + \tau_g + M_{FE}\ddot{\theta}_{FE}(t) + D_E\dot{\theta}_{FE}(t) + K_E(\theta_{FEmin} - \theta_{FE}(t)), \quad (33.1)$$

$$\tau_F(t) = \tau_{basic} + \tau_g + M_{FE}\ddot{\theta}_{FE}(t) + D_F\dot{\theta}_{FE}(t) + K_F(\theta_{FE}(t) - \theta_{FEmax}), \quad (33.2)$$

$$\tau_{OR}(t) = \begin{cases} \tau_{basic} + M_R\ddot{\theta}_R(t) + D_{OR}\dot{\theta}_R(t) + K_{OR}\theta_R(t) & (\theta_{FE}(t) > \theta_{shm0}), \\ \tau_{basic} + \hat{K}_{Rshm}(\theta_{shm}(t) - \theta_R(t)) & (\theta_{FE}(t) \le \theta_{shm0}), \end{cases} \quad (33.3)$$

$$\tau_{IR}(t) = \begin{cases} \tau_{basic} + M_R\ddot{\theta}_R(t) + D_{IR}\dot{\theta}_R(t) + K_{IR}\theta_R(t) & (\theta_{FE}(t) > \theta_{shm0}), \\ \tau_{basic} + \hat{K}_{Rshm}(\theta_{shm}(t) - \theta_R(t)) & (\theta_{FE}(t) \le \theta_{shm0}), \end{cases} \quad (33.4)$$

$$\theta_{shm}(t) = \begin{cases} \theta_{shmb}(1 - \frac{1}{\theta_{shm0}}\theta_{FE}(t)) & (\theta_{FE}(t) \le \theta_{shm0}), \\ 0 & (\theta_{FE}(t) > \theta_{shm0}), \end{cases} \quad (33.5)$$

$$K_E = \begin{cases} 0 & (\theta_{FE}(t) \ge \theta_{FEmin}), \\ K_{Erom} & (\theta_{FE}(t) < \theta_{FEmin}), \end{cases} \quad K_F = \begin{cases} 0 & (\theta_{FE}(t) \le \theta_{FEmax}), \\ K_{From} & (\theta_{FE}(t) > \theta_{FEmax}), \end{cases} \quad (33.6)$$

where τ_{basic} is the constant wire tension for not sagging, τ_g is the gravitational torque of the lower leg, (M_*, D_*, K_*) are the impedance parameters, θ_{FEmin} and θ_{FEmax} are the minimum and maximum values of the ROM, θ_{shm0} is the starting flexion/extension angle of SHM, θ_{shmb} is the maximum value of the rotational angle by SHM, and $\hat{K}_{Rshm}$ is the proportional control gain. When $\theta_{FE}(t) \le \theta_{shm0}$, proportional angle control is used so that the rotation angle $\theta_R(t)$ becomes the desired rotation angle $\theta_{shm}(t)$ as shown in the upper equations of Eqs. (33.3) and (33.4). When $\theta_{FE}(t) > \theta_{shm0}$, impedance control is used to realize the desired outer and inner rotation torques as shown in the lower equations of Eqs. (33.3) and (33.4).

33.3.3 Disabled Knee Joint Movement

In order to imitate troubles in a knee joint in the Knee Robo, we modify the impedance parameters and the parameters of the ROM in Eqs. (33.1)–(33.6).

33.3.3.1 Range of Motion Trouble

In hospitals, therapists perform range of motion (ROM) testing on patients. A patient with a ROM trouble in his/her knee joint can move his/her lower leg only in a limited range. When the therapist moves the patient's lower leg, the therapist feels resistance at the end of the patient's ROM. The resistance, called the end feel, is very important for the therapist in the ROM testing. When the ROM for flexion is 0 to θ_{rom}, the ROM trouble can be imitated in the Knee Robo by replacing θ_{FEmin} and θ_{FEmax} with 0 and θ_{rom}, respectively.

33.3.3.2 Lead Pipe Rigidity and Cogwheel Rigidity

Rigidity and spasticity are troubles of knee joints caused by disorder of the central nervous system. In both troubles, when the therapist moves the patient's lower leg, the therapist feels resistance of the knee joint during passive leg movement. There are two types of rigidity, namely lead pipe rigidity and cogwheel rigidity. In this paper we assume that rigidity occurs only during passive flexion.

In the case of lead pipe rigidity the resistance force is constant during passive leg movement. In order to generate a constant torque τ_{rig}, the damping coefficient D_E in Eq. (33.1) is replaced with the following equation:

$$D_E = \begin{cases} \dfrac{\tau_{rig}}{\dot{\theta}_{FE}(t)} & (\dot{\theta}_{FE}(t) \geq v_{rig}), \\ 0 & (\dot{\theta}_{FE}(t) < v_{rig}), \end{cases} \tag{33.7}$$

where v_{rig} is a threshold value, which is introduced so as to avoid division by zero.

In the case of cogwheel rigidity, the resistance force is generated intermittently during passive leg movement. The damping coefficient of cogwheel rigidity is denoted by multiplying the damping coefficient of lead pipe rigidity of Eq. (33.7) and a square wave function $X(\theta_{FE}(t))$ depending on the flexion/extension angle as follows:

$$D_E = \begin{cases} \dfrac{\tau_{rig}}{\dot{\theta}_{FE}(t)} X(\theta_{FE}(t)) & (\dot{\theta}_{FE}(t) \geq v_{rig}), \\ 0 & (\dot{\theta}_{FE}(t) < v_{rig}), \end{cases} \tag{33.8}$$

$$X(\theta_{FE}) = \begin{cases} 1 & (n\theta_{gr} \leq \theta_{FE}(t) \leq n\theta_{gr} + \Delta\theta_{gr}) \\ 0 & ((n\theta_{gr} + \Delta\theta_{gr} < \theta_{FE}(t) \leq (n+1)\theta_{gr}) \end{cases} \quad (n = 0, 1, 2, \cdots, N), \tag{33.9}$$

where θ_{gr} and $\Delta\theta_{gr}$ are the parameters of the square wave function.

33.3.3.3 Spasticity

In the case of spasticity the resistance force depends on the flexion/extension angular velocity of the knee joint. In order to generate the resistance force for spasticity,

the damping coefficient D_E in Eq. (33.1) is adjusted. Moreover clasp-knife spasticity is caused by rigidity of the extensor muscles of the knee joint. At the beginning of the passive flexion of the patient with clasp-knife spasticity, the therapist feels resistance. During the passive flexion the therapist does not feel resistance suddenly. This implies that the knee joint gives resistance to passive flexion, but suddenly does not give resistance and allows easy flexion. It is assumed that clasp-knife spasticity occurs when $\theta_{FE}(t) = \theta_{ck}$. Then in order to imitate clasp-knife spasticity the damping coefficient in Eq. (33.1) is replaced with the following equation:

$$D_E = \begin{cases} D_{spa} & (\dot{\theta}_{FE}(t) \geq 0, \theta_{FE}(t) < \theta_{ck}), \\ 0 & (\text{else}), \end{cases} \tag{33.10}$$

where D_{spa} is the damping coefficient for spasticity.

33.3.3.4 Contracture

Contracture is one kind of troubles in knee joints. A patient suffers from contracture when the patient does not move his/her body after onset of the disease, such as cerebral apoplexy. Then it will become harder to move his/her body. In hospitals therapists perform a stretching program for patients with contracture. The therapist rotates the lower leg outward and inward repeatedly, and then retracts the lower leg. The Knee Robo can imitate such passive movement.

33.4 Experiments

The basic experiments are performed to verify the effectiveness of the proposed knee joint models of healthy and disabled persons. In order to imitate resistance of a knee joint during passive leg movement, impedance control is applied to the Knee Robo as shown in Fig. 33.3. The desired knee joint torques for impedance control are calculated by using the knee joint torques of Eqs. (33.1)–(33.4). The parameters of the proposed models are determined from the therapists' opinion in the demonstration of the Knee Robo.

33.4.1 Knee Joint Movement of Healthy Person

In order to imitate knee joint movement of a healthy person we use the parameters as follows: $\theta_{shmb} = 4$ deg, $\theta_{shm0} = 10$ deg. The experimental results are shown in Fig. 33.4. It can be seen that the rotation movement due to SHM is realized in the robot.

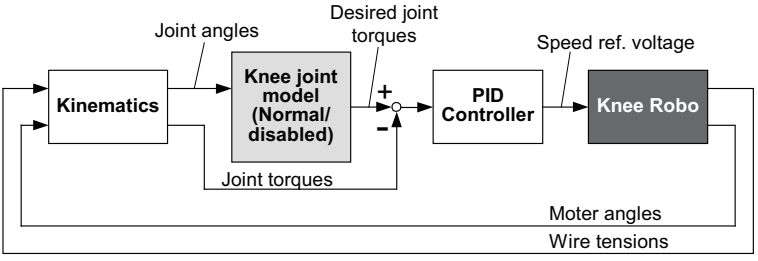


Fig. 33.3 Block diagram of control system of Knee Robo

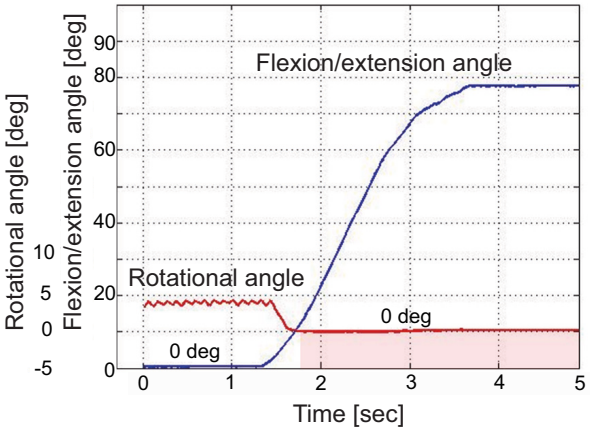


Fig. 33.4 Experimental results of imitation of knee joint movement of healthy person

33.4.2 Knee Joint Movement of Range of Motion Trouble

In order to imitate range of motion trouble we use the parameters as follows: $\theta_{rom} = 45\pi/180$ rad ($= 45$ deg) and $K_{Erom} = 0.2$ Nm/rad. Two kinds of end-feel are imitated at the end of the ROM by using $K_{From} = 0.1$ Nm/rad, and $K_{From} = 0.01$ Nm/rad. The simulation results are shown in Fig. 33.5. It can be seen in the case of $K_{From} = 0.1$ Nm/rad in Fig. 33.5 that the knee joint angle is held about 45 deg, although the knee joint torque is increasing by external human force.

33.4.3 Knee Joint Movement of Lead Pipe Rigidity and Cogwheel Rigidity

In order to imitate lead pipe rigidity and cogwheel rigidity we use the parameters as follows: $\tau_{rig} = 0.08$ Nm, $\theta_{gr} = 7$ deg, and $\Delta\theta_{gr} = 2$ deg. The imitation of cogwheel

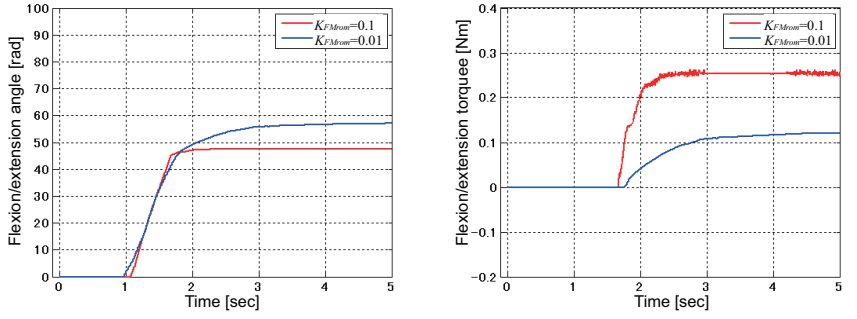


Fig. 33.5 Experimental results of imitation of ROM trouble

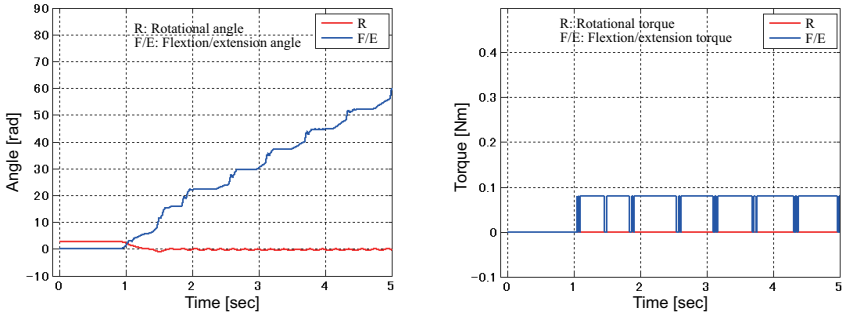


Fig. 33.6 Experimental results of imitation of cogwheel rigidity

rigidity is shown in Fig. 33.6. The periodical torque is seen during the passive leg movement.

33.4.4 Knee Joint Movement of Clasp-Knife Spasticity

In order to imitate clasp-knife spasticity we use the parameters as follows: $\theta_{rom} = 90\pi/180\,\text{rad}\ (= 90\,\text{deg})$ and $\theta_{ck} = 45\pi/180\,\text{rad}\ (= 45\,\text{deg})$. The imitation of clasp-knife spasticity is shown in Fig. 33.7. When the knee joint angle is less than 45 deg, the knee joint torque is generated. This implies that the subject feels resistance. After 45 deg the knee joint torque is decreasing, although the knee joint angle increases. This implies that the subject does not feel resistance. It can be seen from Fig. 33.7 that clasp-knife spasticity is imitated in the Knee Robo.

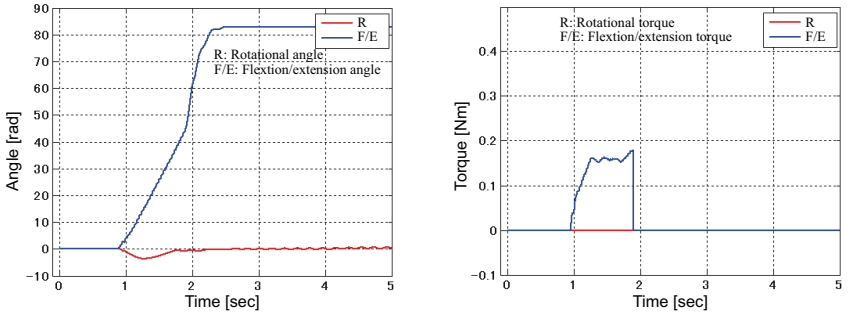


Fig. 33.7 Experimental results of imitation of clasp-knife spasticity

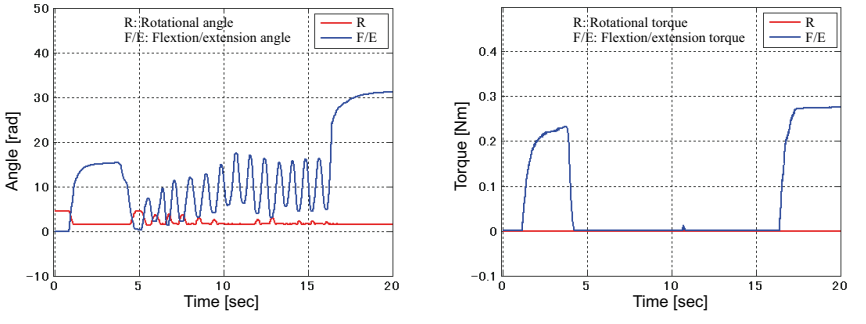


Fig. 33.8 Experimental results of imitation of training for contracture

33.4.5 Knee Joint Movement during Training for Contracture

A subject can experience training for contracture by using the Knee Robo. In the training for contracture there are traction treatment and repetitive movement treatment. In this paper the repetitive movement treatment is considered. In general after repetitive movement treatment the ROM becomes larger. For this purpose the ROM which is improved by repetitive movement treatment is modeled by the following equation:

$$\theta_{FEmax}(t) = \theta_{rom\_init} + K_{rom\_train} \int_0^t |\dot{\theta}_{FE}(t)| dt. \tag{33.11}$$

$\theta_{FEmax}(t)$ increases according to the repetitive passive movement of the lower leg. θ_{rom_init} =15 deg and $K_{rom\_train} = 0.5$ are used in the Knee Robo. The imitation results of the training for contracture are shown in Fig. 33.8. It is shown that the ROM is 15 deg at the beginning of the treatment, and the ROM becomes larger, namely about 30 deg, after the treatment by the subject.

33.5 Conclusions

We have proposed knee joint models of healthy and disabled persons on the basis of flexion/extension movement and rotation movement. These models can imitate the human knee joint movements, namely SHM, range of motion trouble, lead pipe rigidity, cogwheel rigidity, clasp-knife spasticity, and contracture. Consequently, it has been found from the fundamental experiments that the Knee Robo enables us to learn the knee joint movement of healthy and disabled persons by feeling resistance of the knee joint.

The future work is to introduce the Knee Robo to PT/OT training schools, and to improve the knee joint models and the control algorithms on the basis of the opinions from therapists, educational staffs and students.

Acknowledgements. This research receives support of grants-in-aid for scientific research (base research (C) 205200480).

References

1. Uraoka, S., Makita, S., Masutani, Y., Nishimura, A.: Development of Leg Robot for Physical Therapist Training. In: Procs. of the 24th Annual Conf. of the Robotics Society of Japan, p. 1134 (2006) (in Japanese)
2. Kikuchi, T., Oda, K., Furusho, J.: Leg-Robot for demonstration of spastic movements of brain-injured patients with compact MR fluid clutch. Advanced Robotics 24, 671–686 (2010)
3. Morita, Y., Kawai, Y., Ukai, H., Sanaka, K., Nakamuta, H., Takao, K.: Development of Leg Robot for Physical Therapy Training - Proposal of Knee Joint Mechanism with Rolling, Sliding and Coming Off -. In: Procs. of 2009 Int. Conf. on Mechatronics and Information Technolog, ICMIT 2009, pp. 333–334 (2009)
4. Morita, Y., Kawai, Y., Hayashi, Y., Hirano, T., Ukai, H., Sanaka, K., Nakamuta, H., Takao, K.: Development of Knee Joint Robot for Students Becoming Therapist, - Design of Prototype and Fundamental Experiments -. In: Procs. of Int. Conf. on Control, Automation and Systems 2010, ICCAS 2010, pp. 151–155 (2010)

Chapter 34
Exercise Programming and Control System of the Leg Rehabilitation Robot RRH1

Marcin Kaczmarski and Grzegorz Granosik

Abstract. A robot for rehabilitation of the lower extremities has been developed at the Technical University of Łódź (TUL) to allow early treatment of patients after injury or in coma. The device is designed to exercise patients lying in their beds and can fit most of hospital appliances. The main advantage over existing similar solutions is that it provides simultaneous two-plane motion exercises for the knee and the hip. One of the methods for programming the exercises is following the therapist's movements and recording trajectories. Compliance control applied to each axis allows detecting the patient's force counteraction and muscle spasticity. Additionally, various protection systems that allow the robot to be used for rehabilitation therapy of persons with locomotive disabilities are presented.

34.1 Introduction

Modern neurorehabilitation is based on the neuroplasticity of the human brain, with therapists utilizing methods that require repetitive motion to relearn any lost functionality of damaged brain regions [22]. Robot-assisted devices are very useful for such tasks, due to their uninterruptible work, lack of fatigue, and high accuracy of generated movements.

Presently, there are only a few commercial robots for rehabilitation of people with central nervous system disorders. The Swiss Hocoma is one of the market leaders that has developed and commercialized a set of three rehabilitation robots. For the rehabilitation of the lower limbs it offers the Lokomat, a mechanical orthosis which is used in assistance and reeducation of walking for people after strokes and spinal cord injuries [5, 15].

Marcin Kaczmarski · Grzegorz Granosik
Institute of Automatic Control, Technical University of Łódź,
ul. Stefanowskiego 18/22, 90-924 Łódź, Poland
e-mail: {marcin.kaczmarski,granosik}@p.lodz.pl

K.R. Kozłowski (Ed.): Robot Motion and Control 2011, LNCIS 422, pp. 403–412.
springerlink.com

Researchers all over the world work on similar structures, however, they are still in the prototype stage or undergoing clinical trials. These devices may be divided into two subgroups: systems that are based on anthropomorphic structures and those in which the movement of a patient's limb is generated by the robot's end effectors. The anthropomorphic constructions mostly use electrical drives. This group includes the mechanical orthoses ALEX (Active Leg Exoskeleton) [2, 3] and LOPES (Lower Extremities Powered Exoskeleton) [6], with designs very similar to the commercially available Lokomat. They are characterized by a larger number of degrees of freedom. The ALEX system is an evolution of the Gravity Balancing Orthosis. This passive mechanical orthosis does not have any electric motors, yet it can unload the leg joints of the gravity force in all ranges of motion [1].

Aside from electrical motors, pneumatic or hydraulic actuators are also used for rehabilitation devices. At the University of Michigan Human Neuromechanics Laboratory, an ankle joint orthosis powered by pneumatic McKibben muscles [4, 21] has been developed. Myoelectric signals are used for control of the pneumatic muscles, effectively increasing the strength of the wearer [7]. Dong et al. in [20] present an optimal design of magnetorheological fluid dampers for a variable resistance exercise device in the form of a knee brace. The device provides both isometric and isokinetic strength training for the knee and an intelligent supervisory control for regulating the resistive force or torque of the knee brace.

At the moment, the most advanced design among non-anthropomorphic systems is the HapticWalker created by German scientists from the Technical University in Berlin and a group of engineers from Fraunhofer IPK [18, 19]. The mechanism has the form of a cubic frame, where the patient is suspended in a special harness over the robotic arms, that push or pull the patient's feet, as it is done in the Continuous Passive Motion rails [17]. Generating leg movements by acting on the feet provides a possibility to simulate a natural stride and involve the whole body in the therapy. However, acting only on the feet without additional constrains on knees and hips may lead to a situation where pathological compensatory movements develop. The same idea for leg rehabilitation with an actuated rail ended with a foot plate is also applied in the Polish robot Renus-II, created in PIAP and presented in [14]

Another design of a robotic system for rehabilitation of the lower extremities with moving foot plates is the robot developed by a team from Gyeongsang University in Korea [16]. This robot can simulate walking over a flat surface, as well as generate trajectories corresponding to ascending or descending stairs. In the Korean design, an additional powered mechanical system is used to engage the upper limbs in the therapy process. The patient holds a pair of handles that are driven by DC motors, according to a programmed pattern. A similar structure that can simulate an omnidirectional uneven surface is Iwata's GaitMaster from University of Tsukuba [11].

In 2002 an advanced wire driven manipulation system for rehabilitation of the lower extremities was presented [9, 10]. In this robot assisted rehabilitation device the patient's legs are attached to harnesses, which are connected by wires to electric motors mounted on an external frame. A system of cables and pulleys mounted on the frame on both sides of the patient allows for manipulation of the patient's leg.

34.2 Construction of the RRH1 Rehabilitation Robot

The rehabilitation robot for lower extremities created at the Institute of Automatic Control (TUL) is presented in Fig. 34.1. It consists of two rigid arms equipped with harnesses fitted to the patient's knee and ankle, an adjustable column and a wheeled base which allows the robot to be relocated between bedridden patients and provide the therapy even for an unconscious person. The main components of the robot arms and the driving system, and also possible movements (marked with arrows) are shown in Fig. 34.2. More details of the construction of the RRH1 robot can be found in [12, 13].

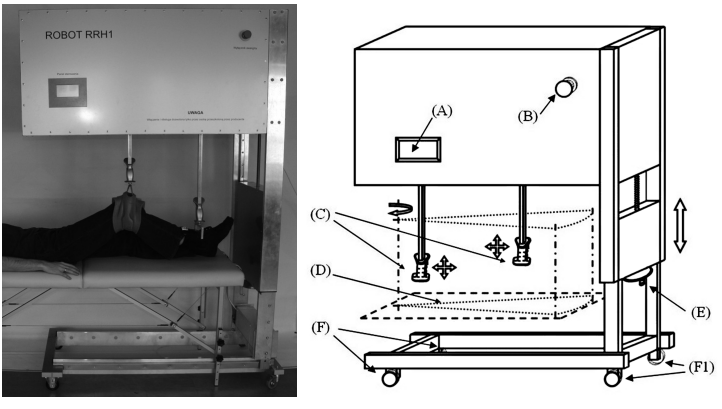


Fig. 34.1 Rehabilitation robot RRH1: A – chassis with LCD display and touch panel, B – emergency stop, C – robot arms with handles, D – robot working space, E – adjustable column, F – base with castor wheels [12, 13]

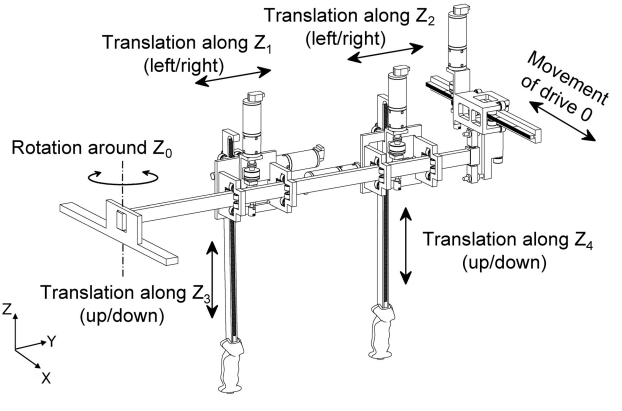


Fig. 34.2 Internal structure of the driving mechanism of the RRH1 rehabilitation robot (names of the joints according to the Denavit-Hartenberg notation)

34.3 Structure of the Control System

We have used the idea of the Distributed Control System (DCS) to control the RRH1 robot. The DCS contains the main controller that cooperates with the Human-Machine Interface (HMI) and communicates via the CAN bus with the local controllers that actuate the robot's joints, as schematically shown in Fig. 34.3.

The main controller is based on the 8-bit single-chip AT90CAN128 microcontroller from Atmel. It has 128kB of built-in flash memory and 4kB of RAM memory. Additionally, it is equipped with 32kB of external SRAM memory for storing trajectories recorded during the teaching process. The CAN bus controller is an internal part of the processor. An additional USB port connects the main controller and an external PC for monitoring purposes. The controller sends information about the position, motor torque, and emergency events during the therapy process.

The Human-Machine Interface helps the therapist communicate with the robot. It contains an LCD graphics display with the resolution of 240×128 pixels and a resistive touch pad layer covering the entire screen area. All information about the current operation, the state of the robot, and any errors or emergency events are shown on the screen. The therapist can confirm operations, choose options or stop the robot by simply pressing appropriate icons on the screen.

The local controller is based on the miControl® Digital Servo Amplifier (DSA), which can drive DC or stepper motors and use an incremental encoder or a Hall sensor for position and velocity feedback. It can generate a trapezoid profile of velocity and interpolate a position profile between two points. It can also work in position, velocity, or current (torque) mode. The DSA contains a single bipolar analog input (range ± 10 V), a single digital output (24 V, 500 mA), and four digital inputs (24 V) with optical separation. In the case of RRH1 robot, the local controller directly drives each DC motor, uses the position feedback from the incremental encoder and potentiometer, measures the motor current, and additionally controls the electromagnetic clutches. For communication with the main controller it uses a built-in CAN interface and the CANopen protocol.

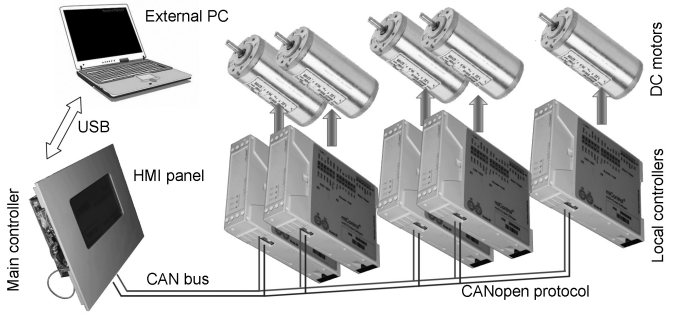


Fig. 34.3 Distributed Control System of the RRH1 rehabilitation robot

34.4 Exercise Programming

The operation of the RRH1 rehabilitation robot begins with a test of communication with all the local controllers. Even if only one of the DSAs does not report back, a warning about the communication error is displayed on the operator's panel and the robot stops its operation immediately. The process of recording and repeating an exercise proceeds in a few subsequent steps, as shown in Fig. 34.4.

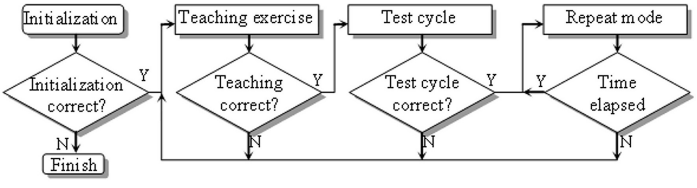


Fig. 34.4 Steps for programming and repeating recorded exercise

Initialization is the first step, where the therapist attaches a lower limb of the patient to specially adapted harnesses. They are connected to ergonomically shaped handles equipped with button triggers. Pressing the buttons gives a signal to the control system to release the electromagnetic clutches, so that the robot's arms can comply with the therapist's movements. Therefore the patient's leg can be freely positioned in a pose convenient to start an exercise. Releasing the buttons locks the robot's arms in their positions. The robot is ready to start the teaching phase.

Teaching the exercise starts when the therapist pressed the appropriate icon on the HMI. Recording the trajectory begins after pressing one of the buttons on the robot arms and continues until both are released. During the teaching phase the measurements of the robot parameters are repeated every 100ms; during that period the main controller records into memory the absolute position of each joint, read from the potentiometer.

The Test cycle is executed using the position control mode in the local controllers; therefore the previously recorded trajectory is replayed. The most important part of this phase is registration of all torques generated by the motors, which reflect forces and torques counteracting on the robot's arms. The measurements of the motor currents (a good equivalent of the torque in a DC motor) are repeated every 100 ms. If the therapist notices that the programmed exercise requires corrections, he or she can stop the robot arm at any time and repeat the teaching phase. After the correct Test cycle, the RRH1 robot is ready to repeatedly execute the exercise using compliance control explained in the next section.

In the Repeat mode the therapist can increase/decrease the speed of motion and change the therapy time. Speeds are defined as the percentage of the nominal recorded value, and they can be set to the following defined levels: 100%, 50%,

33%, 25%, and 20%. Selecting any of the speeds which are less than 100% causes recalculation of all via positions in the trajectory using linear interpolation.

34.5 Compliance Control in the RRH1 Robot

The rehabilitation robot works in direct interaction with the human body, hence safety is an important aspect of the system. The kinematic structure of the robot remains rigid, which may possibly cause injury or pain during execution of any undesired movement. Therefore, the robot uses compliance control (according to the admittance control algorithm [8]) during the Repeat phase of the therapy.

As we recall from the previous section, the main controller recorded position trajectories and motor current trajectories of each joint of the robot during the Teaching and Test phases, respectively. These data are the reference values during execution of the exercise. When the robot works in the Repeat mode, the scaled values of the motor current are used as the force feedback. They show the force interaction between the robot arm and the lower limb of the patient. Additionally, the difference between the reference motor current trajectory – $I_d(t)$ and the actual motor current values – $I_{act}(t)$ indicate changes of the counteraction of the patient, as shown in Fig. 34.5. We can see the rising value of this difference associated with large errors in the position trajectory following. Furthermore, when this difference comes close to the threshold value (called the motor current limit – I_{limit}) it should be treated as a warning in the control algorithm.

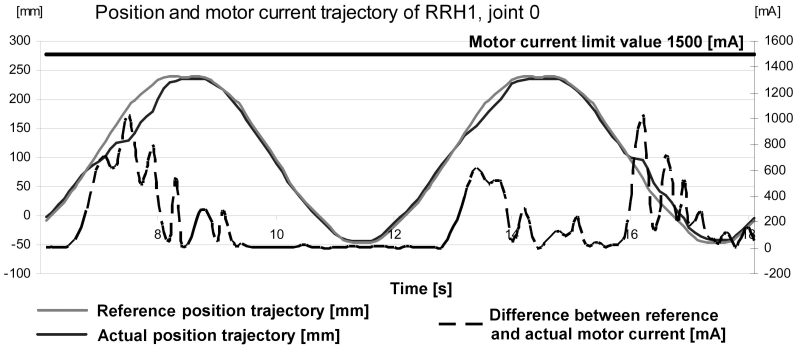


Fig. 34.5 Position and motor current trajectories with presence of force counteraction

The absolute value of the difference between the reference motor current trajectory and the actual motor current is subtracted from the motor current limit and related to it according to the following equation

$$k_{velocity}(t) = \frac{I_{limit} - \big||I_d(t)| - |I_{act}(t)|\big|}{I_{limit}}. \tag{34.1}$$

The motor current limit value is programmed by the therapist and it determines the sensitivity of the compliance control – or it determines the stiffness of the robot drives (seen as the closed loop servo systems). The motor current limit value should be chosen larger than the expected counteracting forces. The value of $k_{velocity}$ calculated according to (34.1) represents a scaling factor for the corrected velocity of the robot joint (a reference signal for the velocity controller in the DSA) expressed by

$$u(t) = k_{velocity}(t) \times e_{\dot{q}(t)}, \tag{34.2}$$

where $k_{velocity}$ – compliance factor, $u(t)$ – corrected velocity of the robot joint, $e_{\dot{q}(t)}$ – velocity error. The compliance control algorithm applied in the RRH1 robot is schematically shown in Fig. 34.6. It can be called the compliance factor and ranges from 0 up to 1.

When the difference between the motor current values reaches I_{limit}, the corrected velocity is reduced to 0. The robot stops its operation and a message about force overloading is displayed. The therapist is given a choice to continue the robot movement along the recorded trajectory, but with reduced speed, or to cancel the program and reenter an exercise.

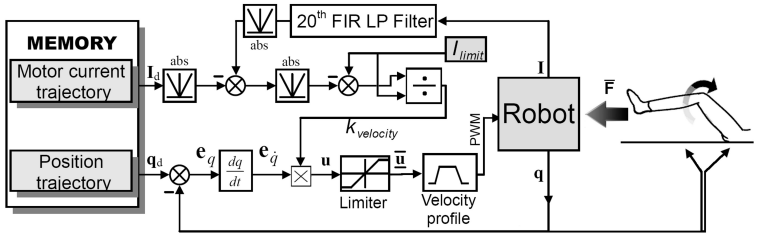


Fig. 34.6 Block diagram of compliance control algorithm used in RRH1 rehabilitation robot: **I**, $\mathbf{I}_d$ vector of actual and reference motor current trajectories for robot's joints, respectively; **q**, $\mathbf{q}_d$ vectors of actual and reference position trajectories, respectively; $\mathbf{e}_{\dot{q}}$, $\mathbf{e}_{\dot{q}_d}$ vectors of position and velocity errors, respectively; **u** vector of corrected velocities; I_{limit} compliance factor programmed by therapist

In the following figures we show the results of the experiment comparing position and compliance control in the RRH1. During the Repeat mode the external force was applied to the right carriage along axis Z_2 (see Fig. 34.2) between the 10-th and 12-th seconds and between the 28-th and 38-th seconds of the experiment. The value of this force was below the physical limit of the drive and therefore it could not stop the motor. This experiment was repeated using the position control algorithm (when the controller follows the trajectory using only position feedback) and with our compliance control. Figure 34.7 shows the influence of the counteracting force on the movement of joint Z_2. We can see the reference position trajectory of the joint and the actual trajectory in two cases: when the robot works with position control and with compliance control. In position mode the robot arm follows the recorded

position trajectory despite the acting force. Compliance control allows stopping the single joint if the measured force differs from the value recorded in the Test cycle. Moreover, after removing the external force the actual joint's position goes toward the reference trajectory.

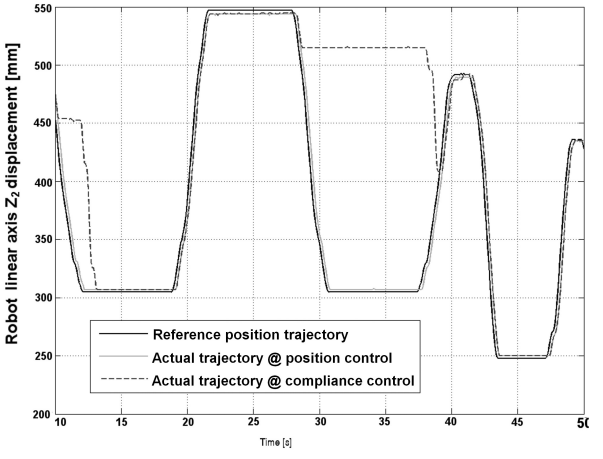


Fig. 34.7 Position of joint Z_2 of the RRH1 robot during the experiment with different control algorithms, under presence of the external force

Figure 34.8 shows the space trajectory of the robot's right handle (refer to Fig. 34.2) during the same experiment – but actually only the part of this experiment between the 22-nd and 40-th seconds is shown in the picture. We observe that the external force strongly reduced motion only along the Y axis, while the robot's arm could freely move in other directions. We can see that the shape of the trajectory in the places indicated on the picture is the same except the trajectory is shifted rightward. In the other parts of the picture trajectory following is almost exact.

34.6 Hardware Protection Features

The robot for rehabilitation of the lower extremities is a device supporting the work of the therapist, who must always be present during the rehabilitation exercises. The device can be stopped at any time by pressing the emergency stop button which cuts off the power supply from the motor drivers (part of the local controller). Additionally, pressing the emergency stop button is signaled on the HMI panel. Returning to the Repeat mode is only possible after removing the cause of the fault and releasing the emergency stop button. In the case of a power supply failure, the robot uses a UPS power supply system to immediately stop the motion and to hold power on the electromagnetic clutches. Information about the power failure is also displayed on the HMI panel.

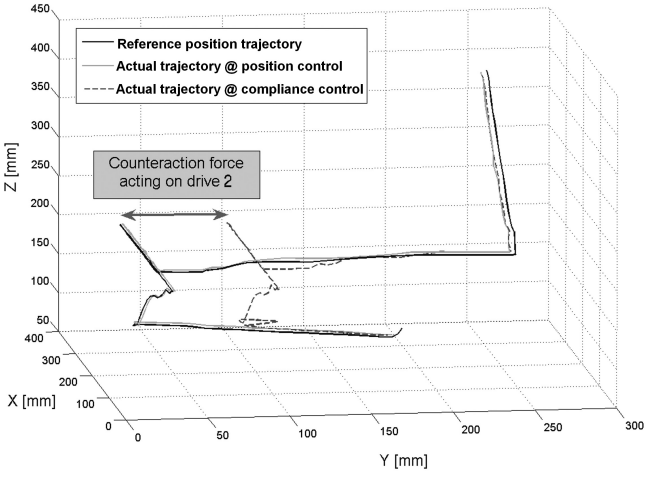


Fig. 34.8 3D view of the position trajectory of the right handle during the same experiment

34.7 Conclusions

The article presents the key aspects of the exercise programming and the main features of the control system of the rehabilitation robot for the lower extremities. The RRH1 device is designed for passive movement treatment for unconscious patients that lie in beds. Applying indirect measurement of forces and compliance control algorithm allow detecting muscle spasticity and external forces counteracting the robot arms. Additionally, the therapist is able to program exercises that enable patients to increase the range of motion in their leg's joints. Since June 2010 the design of the presented RRH1 robot has been under patent rights procedures. The robot is prepared for trial tests which take place in the Clinic of Rehabilitation and Physical Medicine at the Medical University of Łódź.

Acknowledgements. The rehabilitation robot RRH1 was created at the Institute of Automatic Control (TUL) in collaboration with Mr Maciej Czapiewski – the owner of the LEDMEN company and Dr Krzysztof Mianowski from the Institute of Aeronautics and Applied Mechanics, Warsaw University of Technology, who provided valuable consultation on mechanical solutions.

References

1. Agrawal, S.K., Banala, S.K., Mankala, K., Sangwan, V., Scholz, J.P., Krishnamoorthy, V., Hsu, W.-L.: Exoskeletons for gait assistance and training of the motor-impaired. In: Proc. IEEE 10th Int. Conf. Rehabilitation Robotics ICORR 2007, pp. 1108–1113 (2007)
2. Banala, S.K., Agrawal, S.K., Scholz, J.P.: Active leg exoskeleton (alex) for gait rehabilitation of motor-impaired patients. In: Proc. IEEE 10th Int. Conf. Rehabilitation Robotics ICORR 2007, pp. 401–407 (2007)

3. Banala, S.K., Kim, S.H., Agrawal, S.K., Scholz, J.P.: Robot assisted gait training with active leg exoskeleton (alex) 17(1), 2–8 (2009)
4. Chou, C.-P., Hannaford, B.: Measurement and modeling of mckibben pneumatic artificial muscles 12(1), 90–102 (1996)
5. Colombo, G., Jorg, M., Dietz, V.: Driven gait orthosis to do locomotor training of paraplegic patients. In: Proc. 22nd Annual Int. Engineering in Medicine and Biology Society Conf. of the IEEE, vol. 4, pp. 3159–3163 (2000)
6. Ekkelenkamp, R., Veneman, J., van der Kooij, H.: Lopes: a lower extremity powered exoskeleton. In: Proc. IEEE Int. Robotics and Automation Conf., pp. 3132–3133 (2007)
7. Ferris, D.P., Lewis, C.L.: Robotic lower limb exoskeletons using proportional myoelectric control. In: Proc. Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society EMBC 2009, pp. 2119–2124 (2009)
8. Hogan, N.: Impedance control: An approach to manipulation. In: Proc. American Control Conf., pp. 304–313 (1984)
9. Homma, K., Fukuda, O., Nagata, Y.: Study of a wire-driven leg rehabilitation system. In: Proc. IEEE/RSJ Int. Intelligent Robots and Systems Conf., vol. 2, pp. 1451–1456 (2002)
10. Homma, K., Fukuda, O., Sugawara, J., Nagata, Y., Usuba, M.: A wire-driven leg rehabilitation system: development of a 4-dof experimental system. In: Proc. IEEE/ASME Int. Conf. Advanced Intelligent Mechatronics AIM 2003, vol. 2, pp. 908–913 (2003)
11. Iwata, H., Yano, H., Nakaizumi, F.: Gait master: a versatile locomotion interface for uneven virtual terrain. In: Proc. IEEE Virtual Reality, pp. 131–137 (2001)
12. Kaczmarski, M., Czapiewski, M., Mianowski, K., Granosik, G.: Robot rehabilitacyjny RRH1. In: Proceedings of XI Krajowa Konferencja Robotyki (2010) (in Polish)
13. Kaczmarski, M., Granosik, G.: Rehabilitation robot RRH1. In: The Archive of Mechanical Engineering, vol. LVIII, pp. 103–113 (2011)
14. Klimasara, W.J., Dunaj, J., Stempniak, P., Pilat, Z.: Zrobotyzowane systemy renus-1 oraz renus-2 do wspomagania rehabilitacji ruchowej po udarach mózgu. In: Proceedings of XI Krajowa Konferencja Robotyki (2010) (in Polish)
15. Lunenburger, L., Colombo, G., Riener, R., Dietz, V.: Biofeedback in gait training with the robotic orthosis lokomat. In: Proc. 26th Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society IEMBS 2004, vol. 2, pp. 4888–4891 (2004)
16. Novandy, B., Yoon, J., Manurung, A.: Interaction control of a programmable footpad-type gait rehabilitation robot for active walking on various terrains. In: Proc. IEEE Int. Conf. Rehabilitation Robotics, ICORR 2009, pp. 372–377 (2009)
17. Salter, R.: The biologic concept of continuous passive motion of synovial joints: The first 18 years of basic research and its clinical application. Clinical Orthopaedics and Related Research 12(2), 242 (1989)
18. Schmidt, H., Sorowka, D., Hesse, S., Bernhardt, R.: Robotic walking simulator for neurological gait rehabilitation. In: Proc. Second Joint (Engineering in Medicine and Biology 24th Annual Conf. and the Annual Fall Meeting of the Biomedical Engineering Society) EMBS/BMES Conf., vol. 3, pp. 2356–2357 (2002)
19. Schmidt, H., Volkmar, M., Werner, C., Helmich, I., Piorko, F., Kruger, J., Hesse, S.: Muscle activation patterns of healthy subjects during floor walking and stair climbing on an end-effector-based gait rehabilitation robot. In: Proc. IEEE 10th Int. Conf. Rehabilitation Robotics ICORR 2007, pp. 1077–1084 (2007)
20. Sun, J.Q., Rudolph, K., Dong, S., Lu, K.-Q.: Rehabilitation device with variable resistance and intelligent control. Medical Engineering & Physics, 249–255 (2005)
21. Tondu, B., Lopez, P.: Modeling and control of mckibben artificial muscle robot actuators 20(2), 15–38 (2000)
22. Young, J.A., Tolentino, M.: Neuroplasticity and its applications for rehabilitation. American Journal of Therapeutics, December 29 (2010)

Author Index

Index

Lecture Notes in Control and Information Sciences

Edited by M. Thoma, F. Allgöwer, M. Morari

Further volumes of this series can be found on our homepage:
springer.com

Vol. 422: Kozłowski, K.R. (Ed.)
Robot Motion and Control 2011
418 p. 2012 [978-1-4471-2342-2]

Vol. 420: Trinh, H.; Fernando, T.:
Functional Observers for Dynamical Systems
218 p. 2012 [978-3-642-24063-8]

Vol. 419: Samy, I.; Gu, D.-W.:
Fault Detection and Flight Data Measurement
170 p. 2012 [978-3-642-24051-5]

Vol. 418: Alberer, D.; Hjalmarsson, H.; del Re, L.:
Identification for Automotive Systems
348 p. 2012 [978-1-4471-2220-3]

Vol. 417: Johansson, R.; Rantzer A.:
Distributed Decision Making and Control
412 p. 2012 [978-1-4471-2264-7]

Vol. 416: Varga, A.; Hansson, A.; Puyou, G.:
Optimization Based Clearance of Flight Control Laws
451 p. 2012 [978-3-642-22626-7]

Vol. 415: Chesi, G.:
Domain of Attraction
283 p. 2011 [978-0-85729-958-1]

Vol. 414: Imine, H.; Fridman, L.; Shraim, H.; Djemai, M.:
Sliding Mode Based Analysis and Identification of Vehicle Dynamics
127 p. 2011 [978-3-642-22223-8]

Vol. 413: Eleftheriou, E.; Reza Moheimani, S.O.:
Control Technologies for Emerging Micro and Nanoscale Systems
289 p. 2011 [978-3-642-22172-9]

Vol. 412: Fridman, L.; Moreno, J.; Iriarte, R.:
Sliding Modes after the first Decade of the 21st Century
XXX p. 2011 [978-3-642-22163-7]

Vol. 411: Kaczorek, T.;
Selected Problems of Fractional Systems Theory
344 p. 2011 [978-3-642-20501-9]

Vol. 410: Bourlès, H., Marinescu, B.:
Linear Time-Varying Systems
637 p. 2011 [978-3-642-19726-0]

Vol. 409: Xia, Y., Fu, M., Liu, G.-P.:
Analysis and Synthesis of Networked Control Systems
198 p. 2011 [978-3-642-17924-2]

Vol. 408: Richter, J.H.;
Reconfigurable Control of Nonlinear Dynamical Systems
291 p. 2011 [978-3-642-17627-2]

Vol. 407: Lévine, J., Müllhaupt, P.:
Advances in the Theory of Control, Signals and Systems with Physical Modeling
380 p. 2010 [978-3-642-16134-6]

Vol. 406: Bemporad, A., Heemels, M., Johansson, M.:
Networked Control Systems
appro. 371 p. 2010 [978-0-85729-032-8]

Vol. 405: Stefanovic, M., Safonov, M.G.:
Safe Adaptive Control
appro. 153 p. 2010 [978-1-84996-452-4]

Vol. 404: Giri, F.; Bai, E.-W. (Eds.):
Block-oriented Nonlinear System Identification
425 p. 2010 [978-1-84996-512-5]

Vol. 403: Tóth, R.;
Modeling and Identification of Linear Parameter-Varying Systems
319 p. 2010 [978-3-642-13811-9]

Vol. 402: del Re, L.; Allgöwer, F.; Glielmo, L.; Guardiola, C.; Kolmanovsky, I. (Eds.):
Automotive Model Predictive Control
284 p. 2010 [978-1-84996-070-0]

Vol. 401: Chesi, G.; Hashimoto, K. (Eds.):
Visual Servoing via Advanced Numerical Methods
393 p. 2010 [978-1-84996-088-5]

Vol. 400: Tomás-Rodríguez, M.; Banks, S.P.:
Linear, Time-varying Approximations to Nonlinear Dynamical Systems
298 p. 2010 [978-1-84996-100-4]

Vol. 399: Edwards, C.; Lombaerts, T.; Smaili, H. (Eds.):
Fault Tolerant Flight Control
appro. 350 p. 2010 [978-3-642-11689-6]

Vol. 398: Hara, S.; Ohta, Y.; Willems, J.C.; Hisaya, F. (Eds.):
Perspectives in Mathematical System Theory, Control, and Signal Processing
appro. 370 p. 2010 [978-3-540-93917-7]

Vol. 397: Yang, H.; Jiang, B.; Cocquempot, V.:
Fault Tolerant Control Design for Hybrid Systems
191 p. 2010 [978-3-642-10680-4]

Vol. 396: Kozlowski, K. (Ed.):
Robot Motion and Control 2009
475 p. 2009 [978-1-84882-984-8]

Vol. 395: Talebi, H.A.; Abdollahi, F.; Patel, R.V.; Khorasani, K.:
Neural Network-Based State Estimation of Nonlinear Systems
appro. 175 p. 2010 [978-1-4419-1437-8]

Vol. 394: Pipeleers, G.; Demeulenaere, B.; Swevers, J.:
Optimal Linear Controller Design for Periodic Inputs
177 p. 2009 [978-1-84882-974-9]

Vol. 393: Ghosh, B.K.; Martin, C.F.; Zhou, Y.:
Emergent Problems in Nonlinear Systems and Control
285 p. 2009 [978-3-642-03626-2]

Vol. 392: Bandyopadhyay, B.; Deepak, F.; Kim, K.-S.:
Sliding Mode Control Using Novel Sliding Surfaces
137 p. 2009 [978-3-642-03447-3]

Vol. 391: Khaki-Sedigh, A.; Moaveni, B.:
Control Configuration Selection for Multivariable Plants
232 p. 2009 [978-3-642-03192-2]

Vol. 390: Chesi, G.; Garulli, A.; Tesi, A.; Vicino, A.:
Homogeneous Polynomial Forms for Robustness Analysis of Uncertain Systems
197 p. 2009 [978-1-84882-780-6]

Vol. 389: Bru, R.; Romero-Vivó, S. (Eds.):
Positive Systems
398 p. 2009 [978-3-642-02893-9]

Vol. 388: Jacques Loiseau, J.; Michiels, W.; Niculescu, S-I.; Sipahi, R. (Eds.):
Topics in Time Delay Systems
418 p. 2009 [978-3-642-02896-0]

Vol. 387: Xia, Y.; Fu, M.; Shi, P.:
Analysis and Synthesis of Dynamical Systems with Time-Delays
283 p. 2009 [978-3-642-02695-9]

Vol. 386: Huang, D.; Nguang, S.K.:
Robust Control for Uncertain Networked Control Systems with Random Delays
159 p. 2009 [978-1-84882-677-9]

Vol. 385: Jungers, R.:
The Joint Spectral Radius
144 p. 2009 [978-3-540-95979-3]

Vol. 384: Magni, L.; Raimondo, D.M.; Allgöwer, F. (Eds.):
Nonlinear Model Predictive Control
572 p. 2009 [978-3-642-01093-4]

Vol. 383: Sobhani-Tehrani E.; Khorasani K.:
Fault Diagnosis of Nonlinear Systems Using a Hybrid Approach
360 p. 2009 [978-0-387-92906-4]

Vol. 382: Bartoszewicz A.; Nowacka-Leverton A.:
Time-Varying Sliding Modes for Second and Third Order Systems
192 p. 2009 [978-3-540-92216-2]

Vol. 381: Hirsch M.J.; Commander C.W.; Pardalos P.M.; Murphey R. (Eds.)
Optimization and Cooperative Control Strategies: Proceedings of the 8th International Conference on Cooperative Control and Optimization
459 p. 2009 [978-3-540-88062-2]

Vol. 380: Basin M.
New Trends in Optimal Filtering and Control for Polynomial and Time-Delay Systems
206 p. 2008 [978-3-540-70802-5]

Vol. 379: Mellodge P.; Kachroo P.;
Model Abstraction in Dynamical Systems: Application to Mobile Robot Control
116 p. 2008 [978-3-540-70792-9]

Vol. 378: Femat R.; Solis-Perales G.;
Robust Synchronization of Chaotic Systems Via Feedback
199 p. 2008 [978-3-540-69306-2]

Vol. 377: Patan K.
Artificial Neural Networks for the Modelling and Fault Diagnosis of Technical Processes
206 p. 2008 [978-3-540-79871-2]

Vol. 376: Hasegawa Y.
Approximate and Noisy Realization of Discrete-Time Dynamical Systems
245 p. 2008 [978-3-540-79433-2]