Springer Tracts in Advanced Robotics

Volume 4

Springer

Berlin
Heidelberg
New York
Hong Kong
London
Milan
Paris
Tokyo

Engineering ONLINE LIBRARY

http://www.springer.de/engine/

Antonio Bicchi · Henrik I. Christensen ·
Domenico Prattichizzo (Eds.)

Control Problems in Robotics

With 119 Figures

Springer

Professor Bruno Siciliano, Dipartimento di Informatica e Sistemistica, Università degli Studi di Napoli Federico II, Via Claudio 21, 80125 Napoli, Italy, email: siciliano@unina.it

Professor Oussama Khatib, Robotics Laboratory, Department of Computer Science, Stanford University, Stanford, CA 94305-9010, USA, email: khatib@cs.stanford.edu

Professor Frans Groen, Department of Computer Science, Universiteit van Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands, email: groen@science.uva.nl

STAR (Springer Tracts in Advanced Robotics) has been promoted under the auspices of EURON (European Robotics Research Network)

Editors

Prof. Antonio Bicchi
Centro Interdipartimentale
di Ricerca, "Enrico Piaggio"
Università di Pisa
Via Diotisalvi, 2
56100 Pisa, Italy

Prof. Henrik Iskov Christensen
Centre for Autonomous Systems
CVAP/NADA
Kungliga Tekniska Högskolan
10044 Stockholm, Sweden

Prof. Domenico Prattichizzo
Dipartimento di Ingegneria
dell'Informazione
Facoltà di Ingegnerici
Università di Siena
Via Roma, 56
53100 Siena, Italy

ISSN 1610-7438

ISBN 3-540-00251-0 Springer-Verlag Berlin Heidelberg New York

Cataloging-in-Publication Data applied for
Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at <http://dnb.ddb.de>.

Springer-Verlag Berlin Heidelberg New York
a member of BertelsmannSpringer Science + Business Media GmbH

http://www.springer.de

Printed in Germany

Typesetting: Digital data supplied by author. Data-conversion by PTP-Berlin e.k., Stefan Sossna
Cover-Design: design & production GmbH, Heidelberg
Printed on acid-free paper SPIN 10903889 62/3020 Rw - 5 4 3 2 1 0

Foreword

The field of robotics continues to flourish and develop. In common with general scientific investigation, new ideas and implementations emerge quite spontaneously and these are discussed, used, discarded or subsumed at conferences, in the reference journals, as well as through the Internet. After a little more maturity has been acquired by the new concepts, then archival publication as a scientific or engineering monograph may occur.

The goal of the Springer Tracts in Advanced Robotics is to publish new developments and advances in the fields of robotics research – rapidly and informally but with a high quality. It is hoped that prospective authors will welcome the opportunity to publish a structured presentation of some of the emerging robotics methodologies and technologies.

The edited volume by Antonio Bicchi, Henrik Christensen and Domenico Prattichizzo is the outcome of the second edition of a workshop jointly sponsored by the IEEE Control Systems Society and the IEEE Robotics and Automation Society. Noticeably, the previous volume was published in the Springer Lecture Notes on Control and Information Sciences.

The authors are recognised as leading scholars internationally. A number of challenging control problems on the forefront of today's research in robotics and automation are covered, with special emphasis on vision, sensory-feedback control, human-centered robotics, manipulation, planning, flexible and cooperative robots, assembly systems.

Besides the theoretical advancement, most contributions survey the state-of-the-art in the field, report a number of practical applications to real systems, and discuss possible future developments. A fine addition to the series!

Bruno Siciliano
STAR Editor

Preface

This book aims at reporting on some of the most exciting problems of control theoretic nature raised by robotics applications. It is a long-time tradition of the two communities, represented within IEEE by the Control Systems Society (CSS) and by the Robotics and Automation Society (RAS) respectively, to lead the effort to identify innovative applications and related technology demands from our society, as well as in providing the enabling technical solutions.

To further such tradition of cross-disciplinary fertilization and discussion, IEEE CSS and RAS joined with EURON (the European Union Network of Excellence in Robotics) and RSJ (the Robotics Society of Japan) in organizing the *Second International Workshop on Control Problems in Robotics and Automation*, held in Las Vegas, Nevada, on December 14, 2002.

Chapters in this book are edited versions of papers that were selected for presentation at the Workshop. An International Steering Committee, composed of renowned scientists such as H. Arai, J. Bailleul, R. Bajcsy, J. Burdick, R. Chatila, E. Feron, G. Hirzinger, J. Hollerbach, D. Koditscheck, K. Kosuge, V. Kumar, Y. Nakamura, F. Park, S. Sastry, B. Siciliano, M. Spong, and T. Yoshikawa, helped in choosing promising areas and researchers to be invited to participate. Innovative and problematic contributions have been promoted, as well as young and energetic investigators. Unsolicited submissions were also encouraged. All contributions have gone through a thorough review process before reaching their final form.

The organization of the material in the book is as follows. A first group of chapters is concerned with trajectory generation and optimization for classes of complex mechanical systems as they are encountered in advanced robotics applications. These include kinematic and dynamic nonholonomic systems, as e.g. is the case with robot vehicles and under-actuated manipulators, respectively. Two contributions on kinematic nonholonomic systems are *Path optimization for nonholonomic systems: application to reactive obstacle avoidance and path planning* by F. Lamiraux, D. Bonnafous, and C. Van Geem, and *From dynamic programming to RRTs: algorithmic design of feasible trajectories*, by S. La Valle. Two chapters, *Control of nonprehensile manipulation*, by K. M. Lynch, T. D. Murphey, and *Motion planning and control problems for underactuated robots* by S. Martinez, J. Cortes, and F. Bullo, address planning and control for underactuated systems. The chapter *Motion description languages for multi-modal control in robotics*, by M. Egerstedt, addresses some more general questions about how a general language for describing motions of robotic systems could be conceived

A second group of contributions deals with robotic systems designed to interact directly with human operators. Humanoids and human-friendly robots,

about which recently there is much talking even in the popular press, are topics of great applicative potential that involve extremely challenging control problems. The chapter *Polynomial design of dynamics-based information processing system*, by M. Okada and Y. Nakamura, describes one of the problems in designing the dynamic motion of such a complex system as a humanoid, where the number of degrees-of-freedom renders traditional approaches simply unfeasible. Robots interacting with human operators in a safe and dependable way are often designed (sometimes with a biomechanical inspiration) with a lightweight and flexible arm structure. Three chapters in this book deal with how such arms can be designed and controlled: these are *Control challenges of low impedance actuators for human-friendly robotics*, by M. Zinn, O. Khatib, B. Roth, J. K. Salisbury; *Control of a Flexible Manipulator with Noncollocated Feedback: Time Domain Passivity Approach*, by J.-H. Ryu, D.-S. Kwon and B. Hannaford; and *Cartesian compliant control strategies for light-weight, flexible joint robots*, by A. Albu-Schaeffer and G. Hirzinger.

Another prominent avenue of research being currently explored in robotics is the study of "group robotics", i.e. systems formed by many autonomous or semi-autonomous systems cooperating in certain tasks. The chapter *Toward the control of self assembling systems with autonomous mesoscale parts* by E. Klavins explores the possibilities offered by systems that can assembly themselves in more complex structures, while *Towards abstraction and control for large groups of robots* by C. Belta and V. Kumar introduces a formalization of group behaviours and their control.

A relatively large number of chapters in this book is devoted to the application of artificial vision to robotics. This is so because the topic has attracted, in the last few years, a widespread attention from both practitioners and scholars, due to its potentials (think e.g. of replacing costly localization apparatuses in an automated factory with cheap onboard cameras) and to its technical challenges. Chapters contributed to the field include *Omnidirectional sensing for robot control*, by K. Daniilidis, C. Geyer, V. Isler and A. Makadia; *A passivity approach to vision-based dynamic control of robots with nonlinear observer*, by H. Kawai, S. Izoe and M. Fujita; *Visual servoing along epipoles*, by J. Piazzi, D. Prattichizzo, and A. Vicino; *Toward geometric visual servoing*, by N. J. Cowan and D. E. Chang; *Vision based online trajectory generation and its application to catching*, by A. Namiki and M. Ishikawa; and *Stability analysis of invariant visual servoing and robustness to parametric uncertainties*, by E. Malis.

October, 2002

Antonio Bicchi
Henrik I. Christensen
Domenico Prattichizzo

Contents

Path Optimization for Nonholonomic Systems: Application to Reactive Obstacle Avoidance and Path Planning

Florent Lamiraux[1], David Bonnafous[1], and Carl Van Geem[2]

[1] LAAS-CNRS, 7 avenue du Colonel Roche 31077 Toulouse cedex 4, France
email: florent@laas.fr,
WWW home page: http://www.laas.fr/florent
[2] Kineo-CAM, 7 avenue du Colonel Roche 31077 Toulouse cedex 4, France
email: cvangeem@kineocam.com,
WWW home page: http://www.kineocam.com

Abstract. This paper presents a method of path optimization for nonholonomic systems. This method consists in iteratively modifying a given feasible path in order to make a cost related to the path decrease. The paper presents two main applications of this method: the first one is an algorithm that solves the problem of path planning for complex kinematic systems (*i. e.* trucks with two trailers) in extremely constrained environments. The second one is an application in mobile robotics and addresses the problem of reactive trajectory deformation for nonholonomic mobile robots (*i. e.* a cart towing a trailer) in order to avoid unexpected obstacles, and cope with map uncertainty and localization errors.

1 Introduction

Nonholonomic systems have raised interesting issues in robotics as well as in control theory for more than twenty years. These systems first appeared in robotics in the 70's when the first wheeled mobile robots were built. The kinematic constraint imposed by wheels raised successive problems, ranging from controllability, to motion planning and closed loop motion control that tied links between mobile robotics and control theory. Since then, kinematically more complex systems like cars or trucks towing trailers have made these links even stronger.

Joint works in control theory and in robotics have led to practical solutions in the domains of path planning in the presence of obstacles and of motion control for nonholonomic systems. The work presented in this paper is related to both issues and aims at providing new solutions for situations where common approaches generally fail.

The first such situation, in which classical path planning methods are inefficient is the problem of planning a path for a truck towing several trailers on a road when

- the trailer connections are not on the wheel axis of the truck or of the former trailer,

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 1–18, 2003

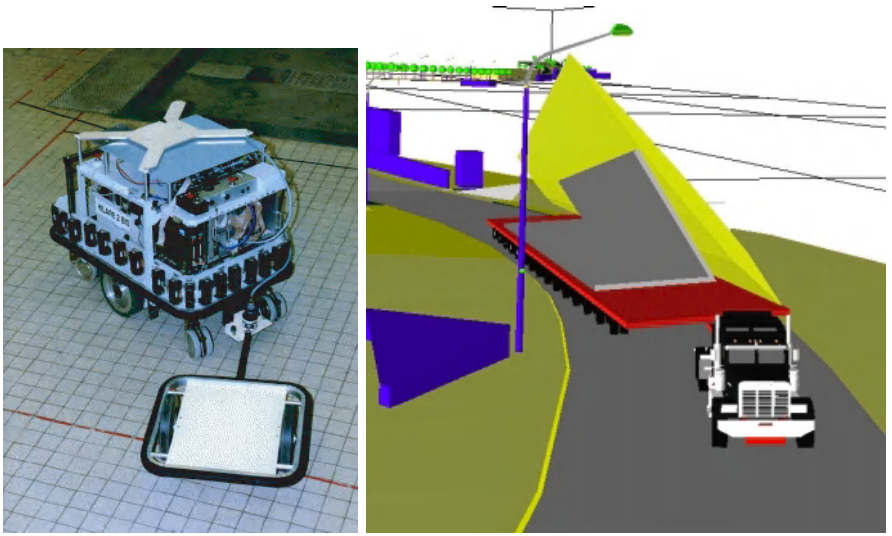


Fig. 1. Two nonholonomic systems: Hilare 2 towing a trailer (dimension 4) and a truck towing a trailer carrying a wing of the Airbus 380 (dimension 6).

- the truck is very large and the road is very narrow.

Classical path planning methods in this situation are inefficient for several reasons. First, this type of kinematic systems are not well understood in control theory : it is not differentially flat nor is it nilpotent. In other words, there is no known steering method for this type of vehicles and thus path planning methods using a steering method [2,5,6] are difficult to use. Other classical path planning methods explore the configuration space by expanding a tree [1,3]. Nodes of the tree are generated by applying constant inputs to the system over small intervals of time, from an existing node (configuration). These methods suffer a major drawback in our context: they require some parameters to be correctly tuned. These parameters are mainly the number of nodes expanded from each leaf of the tree and the time interval over which constant inputs are applied. Too small time intervals and too many nodes expanded from each leaf of the tree lead to huge data structures and time of computation, whereas too large intervals of time and not enough nodes expanded may lead to failure in finding a solution for very constrained problems. Let us moreover notice that computation time and data-structure size grow with the dimension of the configuration space. In examples on which we apply our method, the dimension of the configuration space is 6 and paths are approximately 2km long and have only a few tens of centimeters clearance. In this case, tree-based motion planning methods are not applicable.

The second situation in which the method described in this paper is helpful is when a nonholonomic mobile robot tries to execute a planned

motion. Usually the map of the environment used by the path planning algorithm does not fit the real world, localization of the robot is never exact and unexpected obstacles may lie in the volume swept by the planned motion. For these reasons, executing a planned motion as such usually leads to collisions.

The method we propose in this paper provides an efficient solution to the problems of path planning and of motion control described above. This method takes as input a discretized feasible path, and iteratively compute modifications of this path in order to get away from obstacles and to keep the nonholonomic constraints satisfied. The output of the method is a collision-free feasible path. The method is described in Section 2. Sections 3 and 4 present two different applications of the method.

2 Nonholonomic Systems and Path Deformation

The principle of our method is the same in both applications. A first feasible path, possibly in collision being given, the method iteratively modifies this path in such a way that the path moves away from obstacles. The deformation applied at each iteration makes a potential function related to the path decrease. This potential function decreases when the distance of the path to obstacles increases. Our method is an extension to nonholonomic systems of the elastic strip algorithm proposed in [4]. This later algorithm applies to each configuration $\mathbf{q}(s)$ along the initial path a deformation vector $\eta(s)$ (Figure 2) in order to make the path move away from obstacles. If the robot is not subject to any kinematic constraint, any path is feasible and thus any deformation outputs a feasible path. In our case, the systems we work with are nonholonomic. This means that any path is not necessarily feasible and that any deformation of a feasible path may output a non-feasible path. The seminal idea of our method consists in choosing deformations that keep the nonholonomic constraints of the system satisfied. These deformations are those obtained by perturbing the input functions of the initial path as explained later.

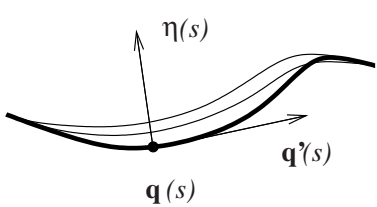


Fig. 2. Current path $\mathbf{q}(s)$ (in bold) and a deformation $\eta(s)$ along this path.

In the rest of this section, we describe one iteration of our deformation algorithm.

2.1 Nonholonomic Systems

A nonholonomic system of dimension n is characterized by a set of $k < n$ vector fields $X_1(\mathbf{q})$,...,$X_k(\mathbf{q})$, where $\mathbf{q} \in \mathcal{C} = \mathbf{R}^n$ is the configuration of the system. For each configuration $\mathbf{q}$, the admissible velocities of the system are the linear combinations of the $X_i(q)$'s. Let us define $n-k$ additional vector fields $X_{k+1}(\mathbf{q})$,...,$X_n(\mathbf{q})$ in such a way that $(X_1, ..., X_n)$ is a basis of $\mathbf{R}^n$ at each configuration. Equivalently, a path $\mathbf{q}(s)$ defined over an interval $[0, S]$ is a feasible path if and only if

$$\forall s \in [0, S] \quad \mathbf{q}'(s) = \sum_{i=1}^{n} u_i(s) X_i(\mathbf{q}) \quad \text{and} \tag{1}$$

$$u_i(s) = 0 \quad \text{for} \quad k+1 \leq i \leq n \tag{2}$$

where $\mathbf{q}'(s)$ is the derivative of $\mathbf{q}(s)$. The reader will understand in Section 2.5 the reason for these additional vector fields. Up to this point, we invite him to forget these vector fields even though they appear in the following equations multiplied by zero functions $u_{k+1}(s)$,..., $u_n(s)$.

2.2 Infinitesimal Path Deformation

To deform a given path we only need to perturb the input functions $u_1(s)$, ..., $u_k(s)$ of the initial path $\mathbf{q}(s)$. For that, we define n real functions $v_1(s)$,...,$v_n(s)$ called *input perturbations*, a real number h and we denote by $\mathbf{q}(s, h)$ the path obtained by plugging $u_i(s) + hv_i(s)$ as input to system (1). (Again, we consider that $v_i(s)$=0 for $k+1 \leq i \leq n$). As a result,

$$\frac{\partial \mathbf{q}}{\partial s}(s, h) = \sum_{i=1}^{n} (u_i(s) + h\ v_i(s)) X_i(\mathbf{q}(s, h))$$

Let us differentiate this equation w.r.t. h:

$$\frac{\partial^2 \mathbf{q}}{\partial s \partial h}(s, h) = \sum_{i=1}^{n} v_i(s) X_i(\mathbf{q}(s, h))$$
$$+(u_i(s) + hv_i(s)) \frac{\partial X_i}{\partial \mathbf{q}}(\mathbf{q}(s, h)) \frac{\partial \mathbf{q}}{\partial h}(s, h)$$

If we denote now by $\eta(s) = \frac{\partial \mathbf{q}}{\partial h}(s, 0)$ the *infinitesimal deformation* and by $\eta'(s) = \frac{\partial \eta}{\partial s}(s)$ the derivative w.r.t. the path parameter s, the above equation becomes for $h = 0$:

$$\eta'(s) = \sum_{i=1}^{n} v_i(s) X_i(\mathbf{q}(s)) + u_i(s) \frac{\partial X_i}{\partial \mathbf{q}}(\mathbf{q}(s)) \eta(s) \tag{3}$$

$$= A(s)\eta(s) + B(s)\mathbf{v}(s) \tag{4}$$

where $A(s) = \sum_{i=1}^{n} u_i(s) \frac{\partial X_i}{\partial \mathbf{q}}(\mathbf{q}(s))$, $B(s)$ is the $n \times n$ matrix the columns of which are the $X_i(\mathbf{q}(s))$ and $\mathbf{v}(s)$ is the n dimensional vector composed of the $v_i(s)$. Let us notice that $A(s)$ and $B(s)$ depend only on the current

path $\mathbf{q}(s)$. (4) is a linear control system, the state and input of which are respectively $\eta(s)$ and $\mathbf{v}(s)$. This system gives the relation between the first order variation of the inputs $u_i(s)$'s and the first order variation $\eta(s)$ of the path $\mathbf{q}(s)$. This system is in fact the tangent linearized system of (1) about the initial trajectory $\mathbf{q}(s)$. We can integrate System (4) to get the following expression:

$$\eta(s) = H(s) \int_0^s H^{-1}(\tau) B(\tau) \mathbf{v}(\tau) d\tau \tag{5}$$

where $H(s)$ is the $n \times n$-matrix-valued function that satisfies:

$$H(0) = I_n \tag{6}$$

$$H'(s) = A(s)H(s) \tag{7}$$

I_n is the identity matrix of order n. Given the current path $\mathbf{q}(s)$ and obstacles, we need to choose at each step, functions $v_1(s)$,...,$v_k(s)$ and a deformation step h in order to make the new path $\mathbf{q}(s,h)$ move away from obstacles. This is the topic of the next section.

2.3 Obstacles and Infinitesimal Path Deformation

Given a set of obstacles in the workspace, we define a potential field $U(\mathbf{q})$ in the configuration space in such a way that the value of the potential increases when the robot gets closer to obstacles. There are different ways to design such a potential field. We will give details about this construction in sections 3.3 and 4.3.

From the potential field in the configuration space, we define the potential of a path by summing $U(\mathbf{q})$ along the path:

$$V(h) = \int_0^S U(\mathbf{q}(s,h)) ds$$

To make the path go away from obstacles, we only need to find functions $\mathbf{v}(s) = (v_1(s), ..., v_k(s), 0, ..., 0)$ such that the first-order variation of the path potential:

$$\frac{\partial V}{\partial h}(0) = \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\mathbf{q}(s))^T \eta(s) ds \tag{8}$$

is negative. The set of smooth functions defined over an interval is an infinite-dimensional space. In order to be able to represent function $\mathbf{v}(s)$ by a finite-dimensional vector, we restrict the choice of $\mathbf{v}(s)$ to a finite-dimensional subspace spanned by a set of test functions $(\mathbf{e}_1(s), ..., \mathbf{e}_p(s))$, where p is an integer, defined over $[0, S]$ into $\mathbf{R}^n$ with the last $n-k$ components uniformly equal to 0. Different solutions are available for this finite-dimensional subspace : polynomials, truncated Fourier series for instance. We thus impose

$$\mathbf{v}(s) = \sum_{l=1}^{p} \lambda_l \mathbf{e}_l(s) \tag{9}$$

where the vector $\lambda = (\lambda_1, ..., \lambda_p)$ is the vector of coefficients of the linear combination of $\mathbf{e}_i(s)$'s. Plugging this expression into (5), we get an expression of $\eta(s)$ w.r.t. the coefficients λ_l.

$$\eta(s) = \sum_{l=1}^{p} \lambda_l \mathbf{E}_l(s) \tag{10}$$

where

$$\mathbf{E}_l(s) = H(s) \int_0^s H^{-1}(\tau) B(\tau) \mathbf{e}_l(\tau) d\tau \tag{11}$$

are the elementary infinitesimal deformations defined over $[0, S]$, solution of System (4) for each input perturbation $\mathbf{e}_l(s)$.

We now need to choose the coefficients λ_l's in such a way that the variation of the path potential V is negative. Let us express this variation w.r.t. these coefficients. We replace $\eta(s)$ by expression (10) in (8) and we get:

$$\frac{\partial V}{\partial h}(0) = \sum_{l=0}^{p} \lambda_l \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\mathbf{q}(s))^T \mathbf{E}_l(s) ds$$

Let us notice that the first order variation of the path potential V is linear w.r.t. the λ_l's. To make the potential decrease, we choose the λ_l's as follows:

$$\lambda_l = -\int_0^S \mathbf{E}_l(s)^T \frac{\partial U}{\partial \mathbf{q}}(\mathbf{q}(s)) ds \tag{12}$$

Thus $\frac{\partial V}{\partial h}(0) = -\sum_{l=1}^{p} \lambda_l^2 < 0$.

2.4 Boundary Conditions

Once the vector $\lambda = (\lambda_1, ..., \lambda_p)$ has been computed as explained above, and once an iteration step h has been chosen, we could plug the new input functions $u_i(s) + hv_i(s)$ as input to system (1) and get as output a new trajectory $\mathbf{q}(s, h)$. However, the end configuration of the deformation interval $\mathbf{q}(S)$ would be modified. If the deformation interval $[0, S]$ is a sub-interval of the initial path interval of definition, after deformation, the path would lose continuity. We need to impose $\mathbf{q}(S, h) = \mathbf{q}(S)$. Let us notice that

1. this boundary condition is non-linear over the vector of coefficients λ,
2. the vector λ we have computed in the previous section does not satisfy this constraint in general.

By approximating the deformed trajectory $\mathbf{q}(s, h)$ by

$$\mathbf{q}(s, h) \approx \mathbf{q}(s) + h\eta(s) \tag{13}$$

the above boundary conditions becomes $\eta(S) = 0$ and according to (10), this constraint is linear over vector λ:

$$\eta(S) = \sum_{l=1}^{p} \lambda_l \mathbf{E}_l(S) = 0 \tag{14}$$

To get a vector that satisfies linear constraints from a vector that does not, we can project the latter vector over the linear subspace defined by the constraints. The constraint matrix L is the matrix the columns of which are the vectors $\mathbf{E}_l(S)$'s:

$$L = (\mathbf{E}_1(S)...\mathbf{E}_p(S))$$

The linear subspace defined by constraint (14) becomes $\{\nu \in \mathbf{R}^p, L\nu = 0\}$ and the projection of vector λ defined by (12) is the following:

$$\bar{\lambda} = (I - L^+L)\lambda \tag{15}$$

where $L^+ = L^T(LL^T)^{-1}$ is the pseudo-inverse of L. Replacing λ by $\bar{\lambda}$ in (10) results in an infinitesimal deformation that satisfies the boundary condition $\eta(S) = 0$. It can easily be verified that this new infinitesimal deformation still makes the path potential decrease.

2.5 Correction of Nonholonomic Deviation

The approximation (13) we made in order to make the boundary condition a linear constraint induces a side effect: this approximation implies a slight nonholonomic deviation: in equation (1), constraints (2) are not satisfied anymore. The velocity $\mathbf{q}'(s)$ along the path has small but non zero components $u_{k+1}(s), ..., u_n(s)$ along vectors $X_{k+1}(\mathbf{q}(s)), ..., X_n(\mathbf{q}(s))$. If we do not correct this deviation, it tends to get amplified after a few iterations. In this section, we explain how to keep these undesirable components close to zero.

In the previous sections, we applied input perturbations to the input functions $u_1(s)...u_k(s)$ corresponding to the vector fields of the system. In this section, we apply input perturbation to the components $u_{k+1}(s)...u_n(s)$ of the velocity $\mathbf{q}'(s)$ along the additional vector fields $X_{k+1}, ..., X_n$ in order to keep these components close to 0. The following input perturbation

$$v_i(s) = -\mu u_i(s) \qquad k+1 \leq i \leq n$$

with $0 < \mu < 1/h$ corresponds to a proportional closed-loop regulation since after deformation and up to approximation (13), the input functions become $u_i(s) \leftarrow u_i(s) + hv_i(s) = (1 - h\mu)u_i(s)$. This regulation corrects the noise introduced by approximation (13). The deformation is now computed according to the following steps.

1. We project the velocity of the current path $\mathbf{q}'(s)$ over vector fields X_i's $(1 \leq i \leq n)$ to get input functions $u_i(s)$'s over interval $[0, S]$,
2. we compute $\eta_{nhd}(s)$ from Equation (5) with

$$\mathbf{v}(s) = (0, ..., 0, -\mu u_{k+1}(s), ..., -\mu u_n(s)),$$

3. we compute vector λ from the obstacle potential field as described in Section 2.3 and we denote by $\eta_{obst}(s) = \sum_{l=1}^{p} \lambda_l \mathbf{E}_l(s)$ the corresponding infinitesimal deformation.

4. If we set $\eta(s) = \eta_{nhd}(s) + \eta_{obst}(s)$, the limit condition $\eta(S) = 0$ becomes $\eta_{obst}(S) = -\eta_{nhd}(S)$. This constraint is affine over the vector of coefficients λ:

$$L\lambda = -\eta_{nhd}(S)$$

As previously, we project vector λ obtained from (12) over the affine set of coefficients satisfying the above equation:

$$\bar{\lambda} = -L^{+}\eta_{nhd}(S) + (I - L^{+}L)\lambda$$

to get an infinitesimal deformation that gets away from obstacles and that corrects nonholonomic deviation.

2.6 Stability Issues for Trailer Systems

Applying the path deformation method described above to truck-trailer systems raises stability issues as explained now.

Truck trailer systems are well-known for being stable when driving forward and unstable when driving backward. this stability (resp. unstability) implies exponential stability (resp. unstability) for the tangent linearized system (4). As a result, the singular values of matrix $H(s)$ decrease exponentially for forward motions and increase exponentially for backward motions. Similarly, the singular values of $H^{-1}(s)$ increase exponentially for forward motions and decrease exponentially for backward motions. This fact implies undesirable numerical effects. Formula (5) indeed combines matrices H and H^{-1}. These matrices are composed of very small and very large coefficients so that numerical errors become of the same order of magnitude as values computed, making the algorithm fail.

This serious limitation can be overcome by combining the control vector fields of the system to get a kinematically equivalent system without the above exponential unstability. We do not provide generic solutions to this problem in this paper. For our two systems, we propose specific solutions described in sections 3 and 4.

3 Application to the Mobile Robot Hilare Towing a Trailer

In this section we describe the application of our method to the mobile robot Hilare 2 towing a trailer (Figure 1 left). For this system, a configuration is represented by the vector $\mathbf{q} = (x, y, \theta, \varphi)$ where (x, y) and θ are the position and orientation of the robot and φ is the angle of the trailer w.r.t. the robot. The control vector fields for this system are

$$Y_1 = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \\ -\frac{1}{l_t}\sin\varphi \end{pmatrix} \quad Y_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ -1 - \frac{l_r}{l_t}\cos\varphi \end{pmatrix}$$

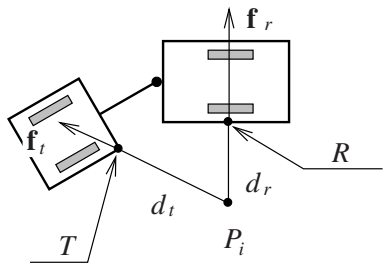


Fig. 3. Configuration space potential field generated by each obstacle point P_i.

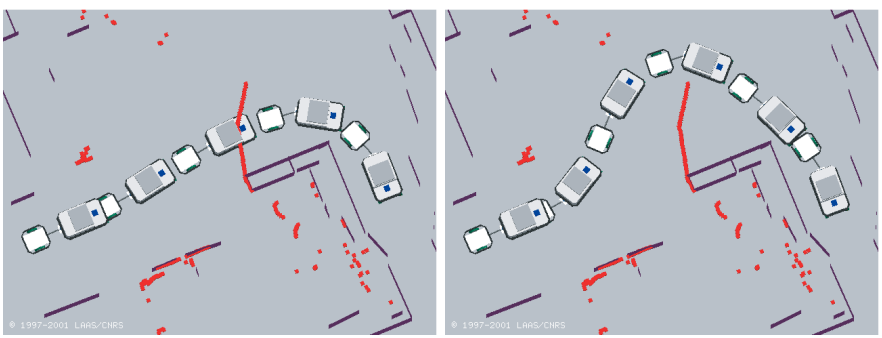


Fig. 4. On the left, a first trajectory computed by the motion planning platform Move3D. The obstacles of the map are in purple. An obstacle lies in the way of the robot. Red dots represent points detected by the laser range finder. On the right, the trajectory after deformation. On this example, the robot stays at the beginning of the path (bottom left) during deformation.

where l_r (resp. l_t) is the distance between the center of the robot (resp. the trailer) and the trailer connection. The inputs of the system are u_1 and u_2 the linear and angular velocities of the robot.

3.1 Stability Analysis

As mentioned in Section 2.6, the exponential (un)stability of trailer systems is inherited by the tangent linearized system. As an example, let us consider a straight line motion for the robot : $\mathbf{q}(s) = (s, 0, 0, 0)$, $(u_1(s), u_2(s)) = (1, 0)$. matrices $A(s)$ in Equation 4, $H(s)$ and $H^{-1}(s)$ are is in this case:

$$A(s) = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1/l_t \end{pmatrix} \quad H(s) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & s & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{-s/l_t} \end{pmatrix} \quad H^{-1}(s) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & -s & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{s/l_t} \end{pmatrix}$$

We notice that $H(s)$ grows linearly while $H^{-1}(s)$ grows exponentially. The exponential term in $H(s)$ means that if we start the straight line motion with

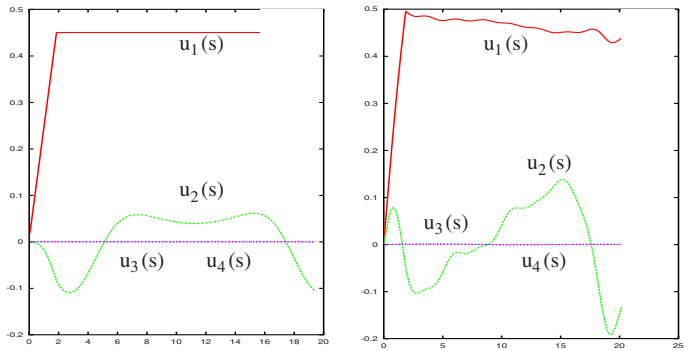


Fig. 5. Input functions $u_1(s)$, $u_2(s)$ along control vector fields and $u_3(s)$, $u_4(s)$ along additional vector fields, before (left) and after (right) deformation. Notice that, as expected, after deformation, the components of the velocity along additional vector fields stay very close to 0.

a small angle φ, applying the same input functions $(u_1(s), u_2(s)) = (1, 0)$ exponentially brings φ back to its initial path value 0.

To get rid of this exponential mode, we replace Y_1 and Y_2 by the following linear combinations:

$$X_1 = Y_1 - \frac{\sin\varphi}{l_t + l_r\cos\varphi} Y_2 = \begin{pmatrix} \cos\theta \\ \sin\theta \\ -\frac{\sin\varphi}{l_t + l_r\cos\varphi} \\ 0 \end{pmatrix}$$

$$X_2 = Y_2$$

Vector field X_1 corresponds to circular motions with constant angle between the robot and the trailer. Along these motions, there is no exponential (un)stability. The reader can indeed check that along circular motions,

1. the tangent linearized system (4) is constant w.r.t. s,
2. matrix $A(s) = A$ has 0 as unique eigenvalue and therefore,
3. $H(s)$ and $H^{-1}(s)$ grow no faster than a polynomial.

Of course, the relative stability along circular motions is not a proof of stability along any trajectory. From a practical point of view, however, we notice that the algorithm behaves much better with these new vector fields, and especially when the interval of deformation gets larger.

3.2 Additional Vector Fields

We define two additional vector fields to get a basis at each configuration:

$$X_3 = \begin{pmatrix} -\sin\theta \\ \cos\theta \\ 0 \\ 0 \end{pmatrix} X_4 = \begin{pmatrix} -\sin(\theta+\varphi) \\ \cos(\theta+\varphi) \\ -l_t - l_r\cos\varphi \\ -l_t \end{pmatrix}$$

We assume that a first path has been computed by a path planner using a map of the environment. This path is sampled at a sample step δs. Then two tasks are executed at the same time: first, the robot starts following the path, detecting obstacles with a laser scanner and secondly the current path is deformed on parts where collisions are detected. The velocity of the robot along the path decreases when the first collision along the current path is close. If s_0 is the abscissa of the first collision, an interval $[s_0 - h_0, s_0 + h_1]$ is defined over which the current path is deformed. At each step, this interval corresponds to the interval $[0, S]$ of the previous section. Along each sample position of this interval, the deformation task computes $A(s)$ and $B(s)$:

$$A(s) = \begin{pmatrix} 0\,0 & -u_1\, s\theta & 0 \\ 0\,0 & u_1\, c\theta & 0 \\ 0\,0 & 0 & -u_1 \frac{l_r + l_t c\varphi}{(l_t + l_r c\varphi)^2} \\ 0\,0 & 0 & \frac{u_2\, l_r s\varphi}{l_t} \end{pmatrix} \quad B(s) = \begin{pmatrix} c\theta & 0 & -s\theta & -s\psi \\ s\theta & 0 & c\theta & c\psi \\ -\frac{\sin\varphi}{l_t + l_r \cos\varphi} & 1 & 0 & -l_t - l_r c\varphi \\ 0 & -1 - \frac{l_r}{l_t} c\varphi & 0 & -l_t \end{pmatrix}$$

where to make notation shorter, $c\theta = \cos\theta$, $s\theta = \sin\theta$, $c\varphi = \cos\varphi$, $s\varphi = \sin\varphi$, $c\psi = \cos(\theta+\varphi)$, $s\psi = \sin(\theta+\varphi)$. Then $H(s)$ is computed using expressions (6-7). For each sample position along the path,

$$H(s + \delta s) = H(s) + A(s)H(s)\delta s$$

Once the matrices $H(s)$'s and their inverses have been computed, we compute the $\mathbf{E}_l(s)$ using (11) where the test functions $\mathbf{e}_l$ are the vector-valued functions with all components equal to 0 except one which is equal to $\cos(\frac{2m\pi s}{S})$ or $\sin(\frac{2m\pi s}{mS})$ where m is an integer between 0 and M: a user defined maximal order. The input perturbations are thus the truncated Fourier series of order less than $M + 1$.

The last step of the deformation computation requires the expression of the configuration space potential field. We describe this potential field in the next section.

3.3 Potential Field in $\mathcal{C}$

Each obstacle point P_i detected by the on-board sensors ot the robot produces a potential field in the plane defined as follows. If M is a point in the plane at distance d from P_i,

$$\begin{array}{ll} u_i(M) = \frac{1}{(d+d_0)^2} - \frac{1}{(d_1+d_0)^2} & \text{if } 0 \le d \le d_1 \\ u_i(M) = 0 & \text{if } d > d_1 \end{array}$$

$d_0 < d_1$ are constant distances. Let $\mathbf{f}_i(M) = -\nabla u_i(M)$ be the force in the plane deriving from this potential. Let R and T be the closest points to P_i on the robot and on the trailer. The configuration space potential field implied by P_i is defined by evaluating the plane potential field at R and T (Figure 3):

$$U_i(\mathbf{q}) = u_i(R) + u_i(T) \tag{16}$$

If P_i is inside the robot or inside the trailer the corresponding term in U_i is set to 0.

The configuration space potential field is defined as the sum of the potential fields relative to each obstacle point:

$$U(\mathbf{q}) = \sum_i U_i(\mathbf{q})$$

The gradient of the potential field is obtained by differentiating (16) w.r.t. the configuration variables (x, y, θ, φ).

$$\begin{aligned}\frac{\partial U_i}{\partial \mathbf{q}}(\mathbf{q}) &= \nabla u_i(R)\frac{\partial R}{\partial \mathbf{q}} + \nabla u_i(T)\frac{\partial T}{\partial \mathbf{q}} \\ &= -\mathbf{f}_r\frac{\partial R}{\partial \mathbf{q}} - \mathbf{f}_t\frac{\partial T}{\partial \mathbf{q}}\end{aligned}$$

where $\mathbf{f}_r = \mathbf{f}_i(R)$ and $\mathbf{f}_t = \mathbf{f}_i(T)$ are the values of the plane force field induced by u_i at R and at T.

3.4 Experimental Results:

Figure 4 (left) shows a path planned for the mobile robot Hilare 2 towing a trailer using Move3D the path planning platform developed at LAAS [7]. We put an obstacle on this path and run our method on-board. After approximately 10 seconds of deformation, the path was clear of collision as shown on Figure 4 (right). Figure 5 shows how the correction of the nonholonomic deviation keeps forbidden velocities close to 0. Without this correction, $u_3(s)$ and $u_4(s)$ grow and become quickly of the same order of magnitude as $u_1(s)$ and $u_2(s)$.

4 Application to Path Planning for Trucks and Trailers

Another field of application for our method has been opened by a project between Airbus Transportation, LAAS, the French Department of Transportation and Kineo-CAM, a start-up company created by researchers from LAAS. The goal of this project is to validate the itinerary of convoys carrying huge parts of the future Airbus 380 aircraft. Sizes of the freights do not allow the convoys to take the highway since existing bridges are too low. Therefore, the convoys need to take secondary roads through villages (Figure 1 right). The problem raised in this project is thus a path planning problem. Given the map of the road, of buildings along the road, given the modeling of the convoys, is it possible to cross the village without collision? Beside path planning, there is a problem of optimization since the convoys have to stay as far as possible from obstacles.

The path planning problem raised here is quite different from the classical formulation however. First, previous work on path planning for nonholonomic

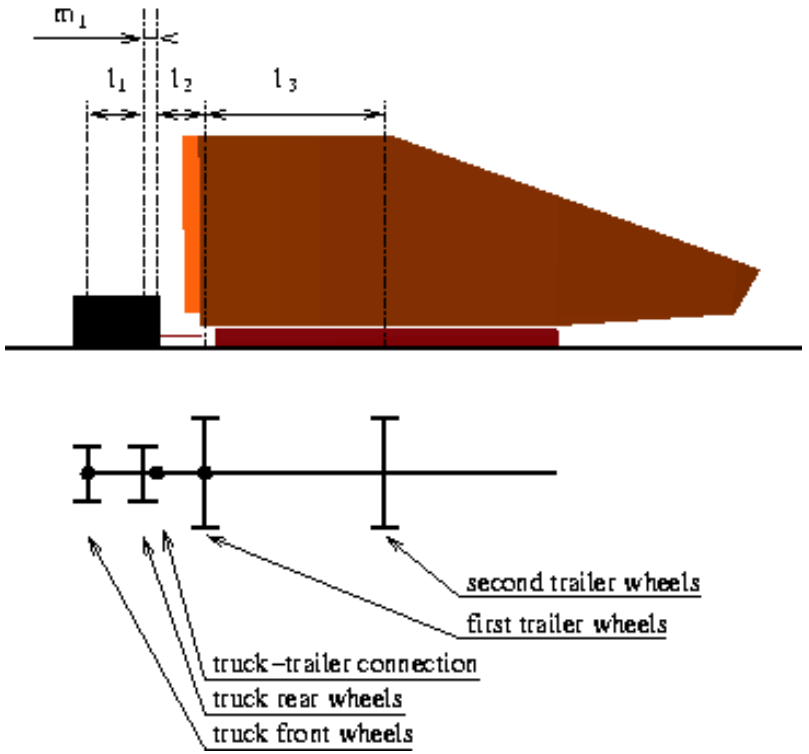


Fig. 6. Truck towing a trailer with a pole. This system is equivalent to a truck towing two trailers. The black box is the bounding box of the truck. The red volume on the trailer is a bounding volume for the wing. The vehicle is 50m long, 12.5m high and 7.7m wide.

systems often have taken advantage of the property of small-time local controllability to design complete path planner. In our application, convoys are not allowed to drive backward and thus lose this controllability property. This constraint breaks down the completeness property of most path planning algorithms. Secondly, the class of homotopy of the solution path is known in advance: the convoy has to follow the road. For these reasons, but also because the distance margins are extremely tight, our path deformation method is the perfect tool for this problem. The planner we developed takes as input an initial trajectory on the road, possibly with collisions and returns a trajectory that locally maximizes the distance to obstacles.

4.1 Modeling of the Convoys

Different truck-trailer systems have been investigated in the project. We describe here only the most interesting for us, that is the most complex and the largest. This system, described on Figure 6 is composed of a truck and

of a trailer connected to the truck by a pole. The trailer is equipped with 24 orientable wheels. The orientation of each of these wheels is controlled by the angle between the pole and the trailer, in such a way that the axis of the wheel passes by the center of rotation of the trailer. From a control point of view, this system is equivalent to a truck towing two trailers, the first trailer being connected behind the wheel axis of the truck (Figure 6 bottom). This system is neither differentially flat nor nilpotent. In other words, no exact steering method is known up to now for this system.

The usual vector fields associated to this system, corresponding respectively to the linear velocity of the truck and to the steering angle time derivative are:

$$Y_1 = \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \\ \frac{\sin\varphi_0}{l_1\cos\varphi_0} \\ 0 \\ -\frac{l_2\sin\varphi_0+\sin\varphi_1 l_1\cos\varphi_0+m_1\cos\varphi_1\sin\varphi_0}{\cos\varphi_0 l_1 l_2} \\ \frac{\sin\varphi_1 l_3-l_2\cos\varphi_1\sin\varphi_2}{l_2 l_3} + \frac{m_1\sin\varphi_0(\cos\varphi_1 l_3-\sin\varphi_1\sin\varphi_2 l_2)}{l_2 l_3 l_1\cos\varphi_0} \end{pmatrix} Y_2 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

where configurations are parameterized by vector $\mathbf{q} = (x, y, \theta, \varphi_0, \varphi_1, \varphi_2)$, (x, y) being the position of the middle point of the truck rear axis, θ the orientation of the truck, φ_0, the steering angle, φ_1 the orientation of the first trailer w.r.t. the truck and φ_2 the orientation of the second trailer w.r.t. the first trailer (Figure 7).

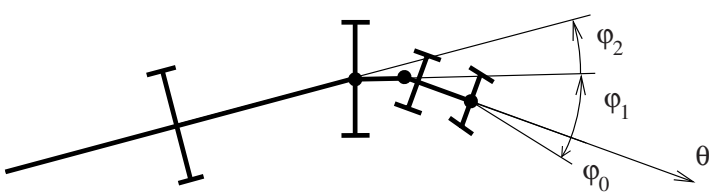


Fig. 7. Parameterization of configurations of a truck towing two trailers.

4.2 Stability Analysis

As for Hilare and its trailer the tangent linearized system of the above system produces matrices $H^{-1}(s)$ that grow exponentially and numerical troubles make the algorithm behave poorly when the interval of deformation becomes large.

Once again the solution we propose is based on intuition and we only observed that this solution makes the algorithm behave much better, without any further proof.

The idea of this solution is basically the same as for Hilare with trailer and consists in finding a combination of the control vector fields

$$X_1 = Y_1 + \alpha(\mathbf{q})Y_2$$
$$X_2 = Y_2$$

such that matrix $A(s)$ of the tangent linearized system (4) about integral curves of X_1 has 0 as unique eigenvalue. To design such a vector field, we set $\alpha(\mathbf{q}) = a_0\varphi_0 + a_1\varphi_1 + a_2\varphi_2$. $\alpha(\mathbf{q})$ does not depend on x, y, θ in order to keep symmetries of the system. Thus

$$\frac{\partial X_1}{\partial \mathbf{q}}(\mathbf{q}) = \frac{\partial Y_1}{\partial \mathbf{q}} + \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & a_0 & a_1 & a_2 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Then we impose the characteristic polynomial of $\frac{\partial X_1}{\partial \mathbf{q}}(\mathbf{q})$ to be equal to s^6. Let us denote by $P(s) = s^6 + \alpha_5 s^5 + \alpha_4 s^4 + \alpha_3 s^3 + \alpha_2 s^2 + \alpha_1 s + \alpha_0$, this characteristic polynomial, where the α_i's are expressions containing a_0, a_1, a_2, φ_0, φ_1 and φ_2. Among equations $\alpha_i = 0$, $i \in \{0, ..., 5\}$, 3 are trivial $(0 = 0)$ due to the structure of matrix $\frac{\partial X_1}{\partial \mathbf{q}}(\mathbf{q})$. By solving the 3 remaining equations, we get expressions of a_0, a_1, a_2 w.r.t. φ_0, φ_1 and φ_2. These expressions are too long to be reported in this paper. They can be easily obtained using any good symbolic computation software.

4.3 Potential Field in $\mathcal{C}$

The interaction between the environment and the different bodies of the vehicle is slightly different from the Hilare-trailer case. The configuration space potential field is now a sum of three terms: one related to the truck and two related to the trailer. In order to make interactions between the vehicle and the environment smoother, the trailer is divided into two symmetric parts, defining two separate bodies, as explained on Figure 8.

$$u(\mathbf{q}) = u_{truck}(\mathbf{q}) + u_{trailer-right}(\mathbf{q}) + u_{trailer-left}(\mathbf{q})$$

The potential of each body in a given configuration depends on the distance d_{body} between the body and the obstacles:

$$u_{body}(\mathbf{q}) = \frac{1}{(d_{body}+d_0)^2} - \frac{1}{(d_1+d_0)^2} \text{ if } 0 \leq d \leq d_1$$
$$u_{body}(\mathbf{q}) = 0 \qquad \text{if } d > d_1$$

similarly as in Section 3.3.

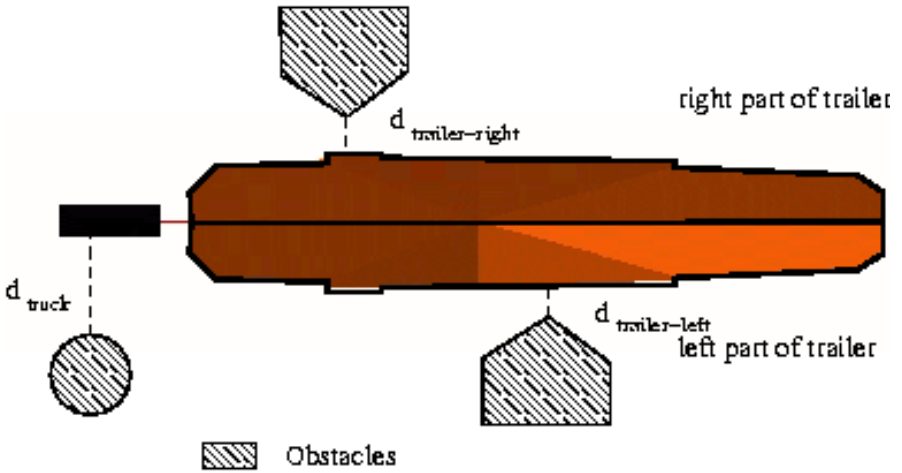


Fig. 8. The potential of a configuration of the truck is the sum of three terms, each one related to a body of the vehicle: truck, left and right side of the trailer. The trailer is far wider than the truck and thus usually much closer to obstacles. If the trailer was considered as a unique body, for configurations of the trailer approximately equidistant from two obstacles, only the closer would interact, making the deformation process oscillate.

4.4 Experimental Results

The algorithm described in this section has been integrated in a software designed by Kineo-CAM and used to optimize trajectories for different vehicles in two villages in the Southwest of France: Lévignac and Gimont. Figure 9 shows the truck towing a trailer carrying a wing (left) and a path computed in Lévignac (right). Figure 10 shows a path in a curve. As for Hilare towing a trailer (Figure 5), the components of the velocity along forbidden vector fields is kept very close to 0.

5 Conclusion and Future Work

We have described in this paper a nonholonomic path optimization method. We have applied this method to two different systems in two different contexts. The method has been initially designed for Hilare and its trailer, a system of dimension 4 and later applied to path planning for more complex systems. This extension has been possible because of the genericity of the approach: we basically only need to know the control vector fields of the system to be able to apply this method.

From a practical point of view however, in order to make the method efficient, adequate combinations of vector fields have to be found to avoid exponential unstability. We have proposed specific solutions for each of our

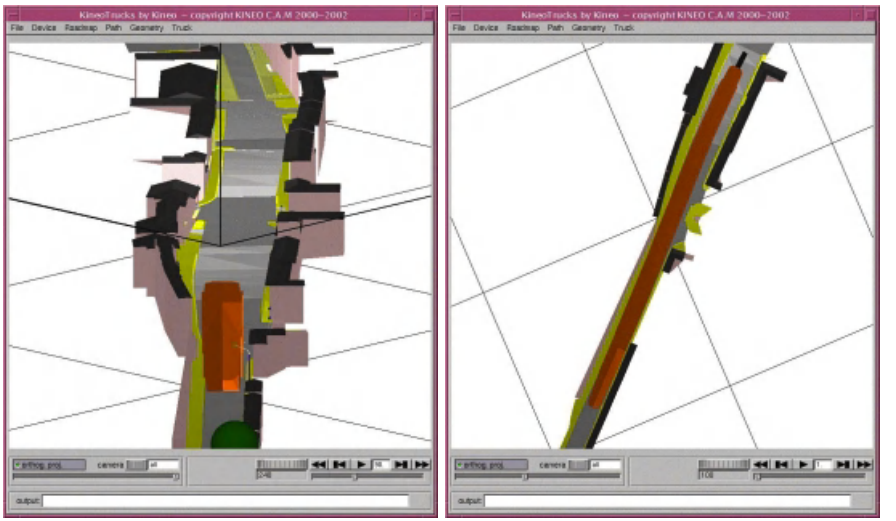


Fig. 9. Truck with trailer in Lévignac. Although the problem seems easy since the road is almost straight, the path clearance here is of only 40cm. On the right, the volume swept by the trailer is in red

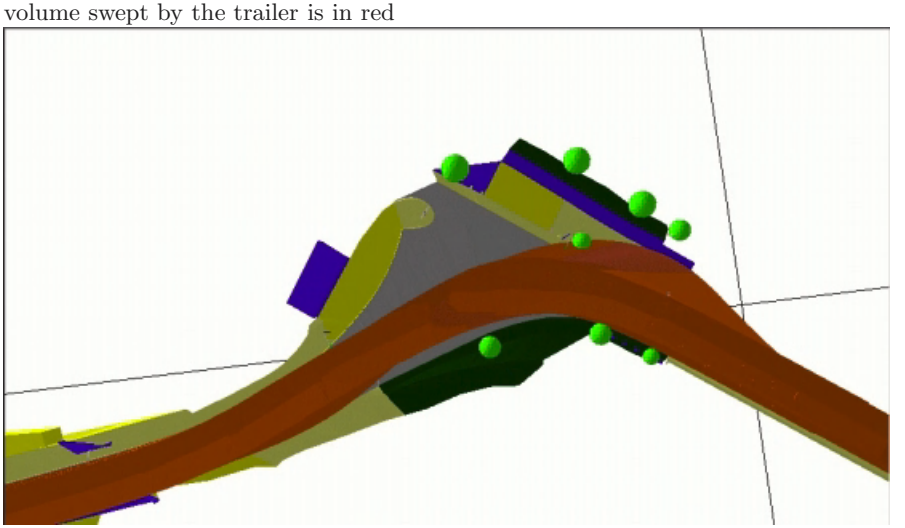


Fig. 10. Volume swept by the trailer in a large curve through Gimont. Trees are not taken into account as obstacles.

applications. More work needs to be done to find generic solutions to this issue.

Two other issues need to be addressed to make our method really efficient for an autonomous mobile robot.

1. As most algorithms, a lot of parameters have to be tuned to make the method work in different situations. Autonomy requires that the robot finds itself the right parameters for each situation.
2. When collision cannot be removed by a simple path deformation, replanning is necessary. The method has to detect these situations and return failure to inform the upper level that replanning is necessary.

We are currently working on both issues.

Acknowledgements

This work has been conducted in the framework of a common project involving Airbus Transportation and the Direction Régionale de l'Équipement of the French Department of Transportation.

References

1. J. Barraquand and J.-C. Latombe. Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles. *Algorithmica*, 10:121–155, 1993.
2. L. E. Kavraki, P. **S**vestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for fast path planning in high dimensional configuration spaces. *IEEE Tr. on Rob. and Autom.*, 12:566–580, 1996.
3. J. Kuffner and S. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *International Conference on Robotics and Automation*, San Francisco, CA, April 2000. IEEE.
4. S. Quinlan. *Real-Time Modification of Collision Free Paths*. PhD thesis, Stanford University, 1995.
5. S. Sekhavat, P. **S**vestka, J.-P. Laumond, and M. Overmars. Multi-level path planning for nonholonomic robots using semi-holonomic subsystems. *International Journal on Robotics Research*, 17:840–857, 1998.
6. T. Simeon, J.-P. Laumond, and C. Nissoux. Visibility-based probabilistic roadmaps for motion planning. *Journal of Advanced Robotics*, 14(6):477–494, 2000.
7. T. Siméon, JP. Laumond, C. Van Geem, and J. Cortés. Computer aided motion: Move3d within molog. In *International Conference on Robotics and Automation*, Seoul KR, May 2001. IEEE.

From Dynamic Programming to RRTs: Algorithmic Design of Feasible Trajectories

Steven M. LaValle

Department of Computer Science, University of Illinois, Urbana, IL 61801, USA

Abstract. This paper summarizes our recent development of algorithms that construct feasible trajectories for problems that involve both differential constraints (typically in the form of an underactuated nonlinear system), and global constraints (typically arising from robot collisions). Dynamic programming approaches are described that produce approximately-optimal solutions for low-dimensional problems. Rapidly-exploring Random Tree (RRT) approaches are described that can find feasible, non-optimal solutions for higher-dimensional problems. Several key issues for future research are discussed.

1 Introduction

In robotics, we frequently study the interaction between two elements: *local constraints*, e.g., a nonlinear, underactuated, mechanical system and *global constraints*, e.g., state space constraints that arise from a complicated environment that is often modeled with thousands of piecewise-linear or piecewise-algebraic constraints. Even considering one of these elements in isolation, designing motions automatically that bring a system from an initial state to a goal region represents a formidable research challenge, even if optimality is abandoned. Whether appearing in control literature or algorithms (computer science, robotics) literature, the problem is generally considered as a form of *motion planning*. To emphasize the consideration of both nonlinear systems and complicated environments, this paper refers to the problem as *trajectory design*. If algorithms can be designed that are able to efficiently find and possibly optimize feasible trajectories for broad classes of problems, many application areas would be greatly impacted, including robotics, aeronautics, automotive design, and computer graphics. The successful development of such algorithms will most likely depend on a culmination of ideas from both the algorithms and control communities.

In the algorithms community, focus has been primarily on global constraints, such as computing collision free paths in the presence of complicated, global constraints on the configuration space of one or more movable bodies. In this case, $\dot{x} = u$, and the configuration, x, simply represents the set of all rigid or articulated body transformations. It is widely known that the class of problems is NP-hard [50], which has caused the focus of research in this area to move from exact, complete algorithms to sampling-based algorithms that can solve many challenging high-dimensional problems efficiently at the

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 19–37, 2003

expense of completeness. Within the past decade, sampling-based versions of earlier ideas were developed. The classic notion of a *roadmap* [12,29,48], is a network of collision-free paths that captures the configuration-space topology, and is generated by preprocessing the configuration space independently of any initial-goal query. The most popular sampling-based variation is termed a probabilistic roadmap (PRM) [24], which is formed by selecting numerous configurations at random, and generating a network of paths by attempting to connect nearby points. In contrast to roadmaps, classical *incremental search* ideas are based heavily on a particular initial-goal query, and include methods such as dynamic programming, A^* search, or bidirectional search. Randomized approaches, such as the randomized potential field approach [3], Ariadne's clew algorithm [44], the planner in [23][1], and Rapidly-exploring Random Trees (RRTs) [26], were introduced to handle higher-dimensional planning problems through clever sampling.

A separate but extremely challenging problem is overcoming local constraints, such as the design of open-loop controls that bring an underactuated nonlinear system from an initial state to a goal state or region, even with no global constraints on the state space (e.g., [10,46]). Much of this work is surveyed in [31].

This paper considers the problem of designing trajectories under both local and global constraints on the state space. These problems are often referred to as *nonholonomic planning* [4,30,31] or in the case of systems with drift, *kinodynamic planning* [11,15,16,19,18,20,25,36]. Many of these methods, such as those in [4,18], follow closely the incremental search paradigm because it is generally more challenging to design a roadmap-based algorithm due to the increased difficulty of connecting numerous pairs of states in the presence of differential constraints (often referred to as the steering problem [31]).

Section 2 formulated the problem. Sections 3 and 4 summarize recent approaches to which I have contributed, based on dynamic programming and Rapidly-exploring Random Trees, respectively. Section 5 helps identify interesting directions for future research.

2 Generic Problem Formulation

The problem formulation given in this section is intended to provide the general idea; however, additional definitions will be needed in each of Sections 3 and 4.

1. **State Space:** A bounded manifold, $X \subset \Re^n$
2. **Boundary Values:** $x_{init} \in X$ and $X_{goal} \subset X$

[1] We note that the method introduced here is termed a PRM by the authors. Since the method is based on incremental search, it is included here to help categorize the methods based on their conceptual similarities.

3. **Constraint Satisfaction:** A function, $D : X \rightarrow \{true, false\}$, that determines whether global constraints are satisfied from state x. This could alternatively be a real-valued function that indicates distance from the constraint boundary. Let $X_{free} \subseteq X$ denote the set of states, x, such that $D(x) = true$ (i.e., the constraints are satisfied).
4. **Inputs:** A set, $U(x) \subset \Re^m$, for each $x \in X$, which specifies the set of controls. This set may be finite, as in the case of a quantized system [17,42], or a compact subset of $\Re^m$, for some $m \leq n$.
5. **State Transition Equation:** $\dot{x} = f(x, u)$.
6. **Incremental Simulator:** Given the current state, $x(t)$, and inputs applied over a time interval, $\{u(t')|t \leq t' \leq t + \Delta t\}$, the incremental simulator yields $x(t + \Delta t)$ through numerical integration of f for a fixed input.

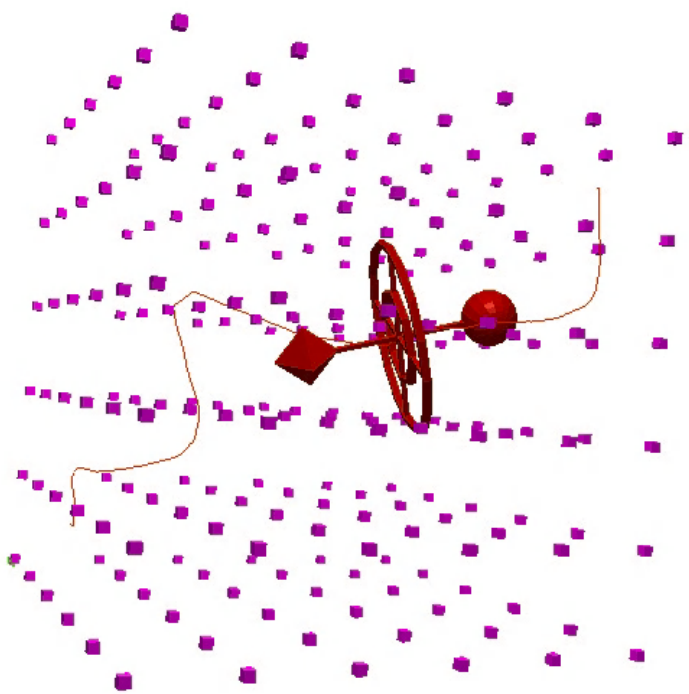


Fig. 1. A challenging example: firing three off-center thrusters to bring a spacecraft through an obstacle course.

An example that illustrates the problem formulation is shown in Fig. 1 (this was solved in [14]). The robot is a multiply-connected spacecraft

modeled with 869 triangles, and the environment is a collection of 216 small squares, arranged in a grid formation. The robot is considered as a free-floating rigid body in a vacuum. There are three thrusters on the robot, each of which can provide a impulse thrust which yields a force with direction that is not through the center of mass. The task is to compute a sequence of firings for the thrusters that brings the robot from an initial state at rest to goal state at rest, on the other side of the grid.

3 Dynamic Programming

Whether in control theory or the design of algorithms, Bellman's principle of optimality [5] has represented a powerful constraint on the set of candidate solutions to an optimization problem. Assume that a discrete-time approximation is made to the original problem described in Section 2. Let k refer to a *stage* or *time step*, and let x_k and u_k refer to the state and input at stage k. Let $K+1$ denote the final stage (as described later, this does not have to be explicitly chosen).

Define a *cost functional* of the form

$$L(x_1, \ldots, x_{K+1}, u_1, \ldots, u_K) = \sum_{k=1}^{K} l(x_k, u_k) + l_{K+1}(x_{K+1}), \tag{1}$$

in which $l_{K+1}(x_{K+1}) = 0$ if $x_{K+1} \in X_{goal}$, and $l_{K+1}(x_{K+1}) = \infty$ otherwise. Furthermore, l is a nonnegative real-valued function such that $l(x_k, u_k) = 0$ if and only if $x_k \in X_{goal}$ (one might also require that u_k represents an input that expends no energy in this case). Assume that once the state reaches X_{goal}, it remains there until stage $K+1$ without any further cost (other variants are possible, of course).

3.1 Classical Cost-to-go Iterations

Classical numerical dynamic programming iterations [5–7,27,28] can numerically solve the problems of interest in this paper. Let the *cost-to-go* function $L_k^* : X_{free} \rightarrow [0, \infty]$ represent the cost if the optimal trajectory is executed from stage k until stage $K+1$,

$$L_k^*(x_k) = \min_{u_k, \ldots, u_K} \left\{ \sum_{i=k}^{K} l(x_i, u_i) + l_{K+1}(x_{K+1}) \right\}. \tag{2}$$

The approach computes representations of the cost-to-go functions iteratively from stage $K+1$ to 1. In each iteration, L_k^* is computed using the representation of L_{k+1}^*, by using the following dynamic programming equation, which involves a local optimization over the inputs:

$$L_k^*(x_k) = \min_{u_k} \left\{ l(x_k, u_k) + L_{k+1}^*(x_{K+1}) \right\}. \tag{3}$$

Note that $L^*_{K+1} = l_{K+1}$ from (1), which implies that the final stage-dependent cost-to-go function is immediately determined. If K is sufficiently large, then for reasonably-behaved planning problems there exists an $i < K$ such that $L^*_k = L^*_i$ for all k satisfying $k > i$. This will hold for problems in which: 1) all optimal trajectories that reach the goal arrive within a bounded amount of time; 2) infinite cost is obtained for a trajectory that fails to reach the goal; 3) no cost accumulates while the robot "waits" in the goal region; 4) the environment and system are stationary.

The cost-to-go values are computed over a finite set, $P \subset X_{free}$, of *sample points* on which the cost-to-go function L^* is defined through interpolation. In d-dimensions, the complexity of classical interpolation is $O(2^d)$; however, in [35], we presented a method based on barycentric subdivision that reduces this to $O(n \lg n)$. The dynamic programming equation (3) is applied at each sample point to compute the next cost-to-go function, $\tilde{L}^*_K$. Each subsequent cost-to-go function is similarly computed. Consider the computation of $\tilde{L}^*_k$. For a finite-input model, the right side of (3) is an optimization over all inputs $u_k \in U$. The values $\tilde{L}^*_{k+1}(x_{k+1})$ are computed by using the incremental simulator and interpolation. For the compact input model, a set of sample points are defined in U, and the right side of (3) is an optimization over all inputs $u_k \in U$. When the inputs are tried, the global constraints (e.g., collision checking) can be directly evaluated each time to determine whether each x_{k+1} lies in X_{free}.

Note that L^*_K represents the cost of the optimal one-stage strategy from each state x_K. More generally, L^*_{K-i} represents the cost of the optimal $(i+1)$-stage strategy from each state x_{K-i}. It was assumed that all optimal trajectories require no more than a bounded number of stages before terminating in X_{goal}. For a state x, let $I(x) \subset P$ denote the set of sample points that are used to compute the cost-to-go for x by interpolation. For the algorithm to succeed, the resolution must be set so that $I(x_{k+1}) \subset X_{goal}$ for sample points near the goal region; otherwise, interpolation with infinity will be attempted, and the algorithm will fail to progress. For a small, positive δ the dynamic programming iterations are terminated when $|\tilde{L}^*_k(x_k) - \tilde{L}^*_{k+1}(x_{k+1})| < \delta$ for all sample points. Note that no original choice of K was necessary because termination occurs when the cost values have stabilized. The resulting stabilized cost-to-go function, $\tilde{L}^*_1$, can be considered as a representation of the optimal feedback strategy, and is simply denoted as $\tilde{L}^*$.

¿From any state, x, the optimal input in this strategy is obtained by selecting u_k to minimize

$$\tilde{L}^*(x) = \min_{u_k} \left\{ l(x, u_k) + \tilde{L}^*(x') \right\}, \tag{4}$$

in which x' is obtained by applying the incremental simulator to x and u_k. In the compact input case, only the inputs that are sample points in $U(x)$ are considered. Starting from any initial state, $x_1 \in X_{free}$, a trajectory can be computed by iteratively applying (4) to compute and apply inputs

until termination in X_{goal} is achieved. This results in a sequence of inputs, $u_1, u_2, \ldots, u_k$, and a sequence of states, $x_1, x_2, \ldots, x_k$, in which $x_k \in X_{goal}$. The cost functional (1) can be used to compute the cost of this trajectory. Without numerical error, the cost would be $L^*(x_1)$. Let $\hat{L}^*(x_1)$ denote the actual computed cost by applying $\tilde{L}^*$ to guide the state from x_1 to X_{goal}.

3.2 Improved Algorithms

In each pass over the state space in the method above, most of the values remain unchanged because they either have not been reached, or the optimal value is already known. In [35], we introduced three improved variations that focus the computation only on the active portion of the state space, much in the same way as Dijkstra's algorithm on a graph. The first of the three is summarized here.

For a sample point, $p \in P$, consider the values $\tilde{L}^*_{k+1}(p)$ and $\tilde{L}^*_k(p)$, which are the cost-to-go values at p from iteration $k+1$ to iteration k of the classical algorithm. Note that for any $k \in \{2, \ldots, K+1\}$ and any $p \in P$, the cost-to-go is monotonically nonincreasing, $\tilde{L}^*_k(p) \leq \tilde{L}^*_{k+1}(p)$. If $\tilde{L}^*_k(p) = \tilde{L}^*_{k+1}(p)$, then there are two possible interpretations: 1) $\tilde{L}^*_k(p)$ is infinite,[2] which implies that no trajectories exist which can reach X_g from p in stages k to $K+1$; 2) $\tilde{L}^*_k(p)$ is finite, which implies that the cost-to-go has been correctly computed for p, and it will not decrease further in subsequent iterations. For each of these two cases, the costly evaluation of (3) performs no useful work. Furthermore, if p belongs to the second case, it never needs to be considered in future iterations. Let P_f be called the *finalized set*, which is the set of all sample points for which the second condition is satisfied. Let P_u denote the *unreached set*, which is the set of all sample points for which the first condition is satisfied. One more situation remains. If $\tilde{L}^*_k(p) < \tilde{L}^*_{k+1}(p)$, then in iteration k the evaluation of (3) is useful because it reduces the estimate of the true cost-to-go at p. Let P_a denote the *active set*, which denotes these remaining sample points. Note that P_f, P_u, and P_a define a partition of P. We assume that P_a is small relative to P in each iteration.

For a set of sample points, P_1, let $R(P_1) \subseteq X_{free}$ denote the set of all states, x, such that $I(x) \subseteq P_1$. For any subset $C \subset X_{free}$, let $Pre(C)$ denote a *preimage*, which is the set of all $x_{k+1} \in X_{free}$ such that there exists some $u_k \in U$ with x_{k+1} obtained by integration of f over Δt, starting with $x_k \in C$ and applying input u_k. In other words, $Pre(C)$ gives the set of states from which the set C is reachable in a single stage.

Figure 2 shows an algorithm based on preimages that avoids most of the wasted computations of the classical algorithm. Steps 2 to 4 perform a cost-

[2] In practice, a large positive floating point number represents this cost. In this case, the cost-to-go actually increases in each iteration. This does not pose a problem, however, because this is the only case in which an increase can occur, and it is correctly interpreted.

```
1  P_a ← {}; P_f ← P ∩ X_goal
2  for each p ∈ Pre(R(P_f)) \ P_f
3        Compute lub(p)
4        INSERT(p, P_a)
5  while P_a ≠ ∅ do
6        for each p ∈ P_a
7              Recompute lub(p)
8              if lub(p) is unchanged
9                    DELETE(p, P_a)
10             if lub(p) is unchanged and finite
11                   INSERT(p, P_f)
12       P_a = Pre(R(P_a)) \ P_f
```

Fig. 2. This algorithm computes the optimal navigation function while avoiding most of the wasted computations of the classical algorithm.

to-go computation for every sample point outside of P_f that can reach the finalized region in one stage. Step 3 computes and stores the cost-to-go for a sample point using interpolation; this value is referred to as $lub(p)$, which indicates that it represents lowest upper bound on the optimal cost-to-go. Over time, the value is repeatedly updated until $lub(p) = \tilde{L}^*(p)$. Steps 5-12 generate a loop that terminates when P_a is empty. Within each iteration, an updated $lub(p)$ is computed for each $p \in P_a$. If the dynamic programming computation does not change, then one of two possibilities exists: p is finalized or p is unreached. In either case, it should not belong to P_a, and is therefore deleted.

The second dynamic programming variation in [35] additionally assumes that the state, x_{k+1}, obtained through integration of the state transition equation from x_k over Δt, always lies in a different interpolation region (the convex hull of $I(x)$, for a state, x) as x_k. In this case, a Dijkstra-like algorithm results, which is able to compute the optimal cost-to-go in a single pass over the state space. The third variation in [35] additionally assumes time optimal solutions are requested, which results in an efficient wavefront propagation algorithm.

3.3 Convergence Conditions

One advantage of numerical dynamic programming is that convergence to the optimal solution can be guaranteed for some suitable choice of constants. This section gives conditions such that both $\tilde{L}^*(x)$ and $\hat{L}(x)$ converge to $L^*(x)$ for all $X \in X_{free}$, which are proved in [35], based on extending earlier analysis from [7]. For the *finite input model*, $U(x)$ is finite for all $x \in X_{free}$. Furthermore, it is assumed that there exist positive constants α_1 and α_2, such that for all $x, x' \in X_{free}$, and for all $u \in U(x) \cap U(x')$,

$$\|f(x,u) - f(x',u)\| \leq \alpha_1 \|x - x'\|$$

and

$$\|l(x,u) - l(x',u)\| \leq \alpha_2 \|x - x'\|.$$

These represent Lipschitz conditions which are needed to establish convergence to the optimal solution in the algorithms.

For the *compact input model*, $U(x)$, is a compact subset of $\Re^m$ for some $m \leq n$. Furthermore,

$$U = \bigcup_{x \in X_{free}} U(x)$$

is compact. It is assumed that there exist positive constants α_1 and α_2, such that for all $x, x' \in X$, there exists a positive constant β such that

$$U(x) \subset U(x') + \{u \mid \|u\| \leq \beta \|x - x'\|\}.$$

It is also assumed that for all $x, x' \in X_{free}$, and for all $u, u' \in U(x)$,

$$\|f(x,u) - f(x',u)\| \leq \alpha_1 (\|x - x'\| + \|u - u'\|)$$

and

$$\|l(x,u) - l(x',u)\| \leq \alpha_2 (\|x - x'\| + \|u - u'\|).$$

Let d_x denote the *dispersion* of the set, P, of sample points over X_{free}, which is defined as:

$$d_x = \max_{x \in X_{free}} \min_{p \in P} \|x - p\|.$$

Intuitively, the dispersion measures the furthest distance possible in which a state can be placed away from the nearest sample point. For the compact input model, a dispersion, d_u, can similarly be defined for a set of samples defined over U.

As the number of samples increases, the dispersion becomes smaller. The convergence result is therefore stated in terms of dispersion [35]:

Proposition 1. *For classical dynamic programming and the three variants mentioned in this section, there exists a positive constant $\epsilon > 0$ such that*

$$\|L^*(x) - \tilde{L}^*(x)\| < \epsilon \, (d_x + d_u) \quad \forall x \in X_{free}$$

and

$$\|L^*(x) - \hat{L}^*(x)\| < \epsilon \, (d_x + d_u) \quad \forall x \in X_{free}.$$

Under the finite input model, the proposition holds true by setting $d_u = 0$.

3.4 Barraquand and Latombe's Method

To ease the transition to the topic of Section 4, it is interesting to consider a dynamic programming variant introduced in [4]. In contrast to maintaining cost-to-go functions over X_{free}, as in the methods above, the approach in [4] is to grow a tree of trajectories. Each vertex of the tree represents a state, and each edge represents a piece of the trajectory over which an input is applied.

The state space, X, is partitioned into a rectangular grid in which each element is called a *cell*, to which one of three labels may be applied:

- OBST: The cell contains points in $X \setminus X_{free}$. These are usually precomputed.
- FREE: The cell has not yet been visited by the algorithm, and it lies entirely in X_{free}.
- VISITED: The cell has been visited, and it lies entirely in X_{free}.

Initially, all cells are labeled either FREE or OBST.

Let Q represent a priority queue in which the elements are states, sorted in increasing order according to L, which represents the cost accumulated along the path constructed so far from x_{init} to x.

```
FORWARD_DYNAMIC_PROGRAMMING(x_init, x_goal)
1  Q.insert(x_init, L);
2  T.init(x_init);
3  while Q ≠ ∅ and FREE(x_goal)
4        x_cur → Q.pop();
5        for each x ∈ NBHD(x_cur)
6              if FREE(x)
7                    Q.insert(x, L);
8                    T.add_vertex(x);
9                    T.add_edge(x_cur, x);
10                   Label cell that contains x as VISITED;
11 Return G;
```

The algorithm iteratively grows a tree, $\mathcal{T}$, which it rooted at x_{init}. The NHBD function tries a finite collection of quantized inputs, and returns a set of states that can be reached in time Δt. For each of these states, if the cell that contains it is FREE, then $\mathcal{T}$ is extended. At any given time, there is at most one vertex per cell. The algorithm terminates when the cell that contains the goal has been reached. Convergence properties of the algorithm are established in [4] for car-like robot systems; however, its convergence for general systems remains to be established.

4 Rapidly-Exploring Random Trees

The dynamic programming-based methods are limited to state spaces of only a few dimensions. The Rapidly-exploring Random Tree (RRT) was introduced in [33] as an exploration algorithm for quickly searching high-dimensional spaces that have both global constraints (arising from workspace obstacles and velocity bounds) and differential constraints (arising from kinematics and dynamics). The key idea is to bias the exploration toward unexplored portions of the space by randomly sampling points in the state space, and incrementally "pulling" the search tree toward them. Based on RRTs, a randomized, sampling-based approach to trajectory design was introduced in [36]. In that paper, RRTs were applied to trajectory design problems for

```
BUILD_RRT(x_init)
1 T.init(x_init);
2 for k = 1 to K do
3       x_rand ← RANDOM_STATE();
4       EXTEND(T, x_rand);
5 Return T
```

```
EXTEND(T, x)
1 x_near ← NEAREST_NEIGHBOR(x, T);
2 if NEW_STATE(x, x_near, x_new, u_new) then
3       T.add_vertex(x_new);
4       T.add_edge(x_near, x_new, u_new);
5       if x_new = x then
6             Return Reached;
7       else
8             Return Advanced;
9 Return Trapped;
```

Fig. 3. The basic RRT construction algorithm.

hovercrafts and rigid spacecrafts that move in a cluttered 2D or 3D environment. Theoretical performance bounds are presented in [37]. In [21], RRTs were applied to the design of collision-free trajectories for a helicopter in a cluttered 3D environment. In [53], RRTs were applied to the design of trajectories for underactuated vehicles. After RRTs, another tree-based randomized approach to trajectory design was proposed in [25].

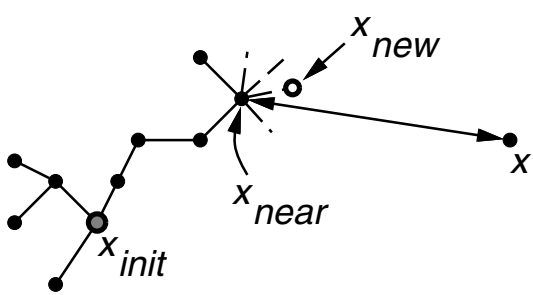


Fig. 4. The EXTEND operation.

The basic RRT construction algorithm is given in Fig. 3. A simple iteration is performed in which each step attempts to extend the RRT by adding a new vertex that is biased by a randomly-selected state, $x \in X$. The EXTEND function, illustrated in Fig. 4, selects the nearest vertex already in the

RRT to x. The "nearest" vertex is chosen according to some metric, ρ. This can be accomplished naively in linear time, or algorithms that perform near logarithmic time can be applied (see [1] for a discussion of these issues in the context of RRTs).

The function NEW_STATE makes a motion toward x by applying an input $u \in U$ for some time increment Δt. The input, u, can be chosen at random, or selected by trying all possible inputs and choosing the one that yields a new state as close as possible to the sample, x (if U is infinite, then a finite approximation or analytical technique can be used). NEW_STATE implicitly checks whether the new state (and all intermediate states) satisfies the global constraints. For many problems, this can be performed quickly ("almost constant time") using incremental distance computation algorithms [22,39,45] by storing the relevant invariants with each of the RRT vertices. If NEW_STATE is successful, the new state and input are represented in x_{new} and u_{new}, respectively. The first row of Fig. 5 shows an RRT grown from the center of a square region in the plane. In this example, there are no differential constraints (motion in any direction is possible from any point). The incremental construction method biases the RRT to rapidly explore in the beginning, and then converge to a uniform coverage of the space [37]. Note that the exploration is naturally biased towards vertices that have larger Voronoi regions. This causes the exploration to occur mostly in the unexplored portion of the state space.

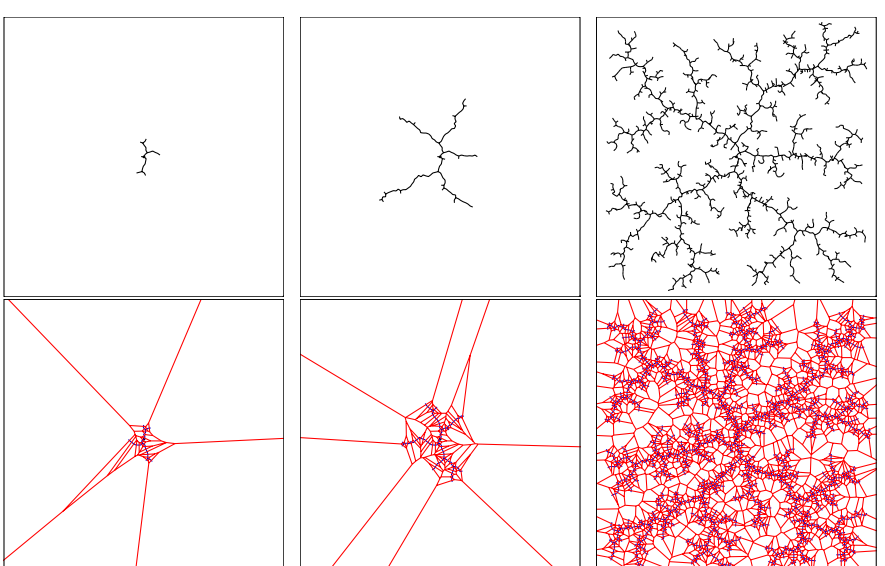


Fig. 5. The RRT rapidly explores in the beginning, before converging to the sampling distribution. Below each frame, the corresponding Voronoi regions are shown to indicate the exploration bias.

4.1 RRT-Based Trajectory Design Algorithms

In principle, the basic RRT can be used in isolation as a path planner because its vertices will eventually cover a connected component of X_{free}, coming arbitrarily close to any specified x_{goal}. The problem is that without any bias toward the goal, convergence might be slow. This can be overcome by altering the sampling strategy to concentrate some samples at or near x_{goal}.

```
RRT_BIDIR(x_init, x_goal)
1 T_a.init(x_init); T_b.init(x_goal);
2 for k = 1 to K do
3       x_rand ← RANDOM_STATE();
4       if (EXTEND(T_a, x_rand) = Trapped) then
5             if (EXTEND(T_b, x_new) = Reached) then
6                   Return PATH(T_a, T_b);
7       SWAP(T_a, T_b);
8 Return Failure
```

Fig. 6. A bidirectional RRT-based planner.

The best performance has been obtained so far by conducting a bidirectional search [49] using two RRTs, one from x_{init} and the other from x_{goal}; a solution is found if the two RRTs meet. Figure 6 shows the RRT_BIDIR algorithm, which may be compared to the BUILD_RRT algorithm of Fig. 3. RRT_BIDIR divides the computation time between two processes: 1) exploring the state space; 2) trying to grow the trees into each other. Two trees, $\mathcal{T}_a$ and $\mathcal{T}_b$ are maintained at all times until they become connected and a solution is found. In each iteration, one tree is extended, and an attempt is made to connect the nearest vertex of the other tree to the new vertex. Then, the roles are reversed by swapping the two trees.

Through extensive experimentation over a wide variety of problems with state spaces up to twelve dimensions, we have concluded that, when applicable, the bidirectional approach is much more efficient than a single RRT approach. One shortcoming of using the bidirectional approach for nonholonomic and kinodynamic planning problems is the need to make a connection between a pair of vertices, one from each RRT. For a planning problem that involves reaching a goal region from an initial state, no connections are necessary using a single-RRT approach. The gaps between the two trajectories can be closed in practice by applying steering methods [31], if possible, or classical shooting methods [8], which are often used for two-point boundary value problems.

5 Research Challenges

This section compares the algorithms presented in Sections 3 and 4, and identifies key directions for future research. Dynamic programming and RRTs offer complementary advantages, and each has its drawbacks.

5.1 Optimality vs. Feasibility

An advantage of the dynamic programming methods is that approximately-optimal solutions are obtained. Even though the dynamic programming equation provides a powerful constraint that significantly reduces the amount of work required to find optimal solutions, it is sometimes much easier to find feasible, but non-optimal solutions. Although much of modern control theory is concerned with optimal decision-making, this requirement is not necessary in many robotics applications. It is certainly desirable in many instances; however, the additional computation cost might outweigh the benefits. Given the complexity of the global constraints considered due to obstacles in the environment, feasibility becomes challenging enough. This philosophy has been followed through most classical algorithmic motion planning work due to the difficulty of finding shortest paths (see the methods in [29]). Algorithms such as RRTs provide solutions that only guarantee feasibility, but they can solve problems that would be too challenging for dynamic programming. In many cases, the solutions can be used in practice after performing some gradient-based optimization techniques (such as those in [8]) on the result, to at least obtain a locally-optimal solution. It remains to be seen whether it is worthwhile to pay the additional cost that appears to be required to obtain optimal solutions, or if there there are ways to avoid this cost.

5.2 Random vs. Deterministic Sampling

All of the methods presented in this paper require some form of sampling on the state space. One important issue is whether randomization offers any advantages when applied to sampling strategies. Over the past decade of algorithmic motion planning work, it has been argued by many that randomization is a key to overcoming the curse of dimensionality; however, in related work [9,34], my co-authors and I have shown that deterministic sampling offers some performance advantages over random sampling. Excellent overviews of deterministic sampling methods include [43,47]. The key idea is to view sampling as an optimization problem in which a set of points is chosen to optimize some criterion of uniformity (as opposed to a statistical test). One of the most popular measures is discrepancy. Let $\mu(R)$ denote the Lebesgue measure of subset R. If the samples in P are uniform in some ideal sense, then it seems reasonable that the fraction of these samples that lie in any subset R should be roughly $\mu(R)$ (divided by $\mu(X)$, which is simply

one). Suppose $X = [0,1]^n$. Define the *discrepancy* [54] to measure how far from ideal the point set P is:

$$D(P, \mathcal{R}) = \sup_{R \in \mathcal{R}} \left| \frac{|P \cap R|}{N} - \mu(R) \right| \tag{5}$$

in which $|\cdot|$ applied to a finite set denotes its cardinality, and $\mathcal{R}$ denotes the set of all axis-aligned rectangular subsets of X (or some other suitably-chosen range space).

At first glance, the progression from deterministic to randomized, and then back to deterministic might appear absurd; thus, some explanation is required. There appear to be two prevailing reasons for the preference of randomized methods over classical deterministic techniques: 1) they fight the curse of dimensionality by allowing a problem to be solved without prior, systematic exploration of all alternatives; 2) if the "problem maker" is viewed as an opponent in a game, then one can often avoid defeat by employing a random strategy (imagine defeating a deterministic strategy by designing a problem that causes worst-case performance).

The first reason is often motivated by considering that a grid with a fixed number of points per axis will require a number of points that is exponential in the dimension of the space. However, this result is not the fault of grids or even deterministic sampling. It was proved long ago by Sukharev [51] that *any* sampling method that constructs a good covering of the space requires an exponential number of samples. The "goodness" is in terms of point dispersion, as defined in Section 3.3, but using an ℓ^∞ metric. We believe the explanation for good performance of path planners in solving challenging high-dimensional problems is that they are able to either exploit some greedy heuristics and/or find solutions to easier problems early by using low-resolution sampling. These benefits are independent of the issue of randomization versus determinism. Thus, multiresolution, deterministic sampling methods are certainly worth exploring in this context. We have already taken steps in this direction in [40]. Randomization appears to be useful in the RRT because it avoids the expensive computation of Voronoi region volumes; however, it may be possible to construct a practical derandomized version.

The second reason (defeating an opponent) might be valid in the case of "true" random numbers; however, any machine implementation generates a deterministic sequence of pseudo-random numbers. These numbers are designed to meet performance criteria that are based on uniform probability densities; however, once it is understood that these numbers are deterministic and being used to solve a particular task, why not design a deterministic sequence that can solve the task more efficiently, instead of worrying about statistical closeness to a uniform probability density? Even if we suppose that true random numbers exist, it seems unlikely that practical examples drawn from applications will contain state space constraints that are designed to break a specific deterministic sampling strategy. Furthermore, randomization can even be introduced back into a deterministic sampling strategy for

precisely the reason of fooling an adversary while still maintaining quasi-random sample distribution properties that are superior to pseudo-random sampling [43].

5.3 Convergence Conditions

For the algorithms presented in this paper, there are many interesting issues regarding conditions that guarantee convergence to a solution, if a solution exists. In the case of dynamic programming, it is additionally important to show that approximate optimality is obtained. When considering dynamic programming as an approximation of the cost-to-go function, it is possible to guarantee convergence by specifying Lipschitz conditions. In the case of the dynamic programming algorithm by Barraquand and Latombe from Section 3.4, it is not clear whether convergence can be guaranteed if the algorithm is applied to nonlinear systems other than those intended by the authors. In general, for convergence of the algorithms, it appears that it would be useful to have multiple, independent conditions of convergence for a given algorithm. For example, suppose a trajectory design algorithm is known to converge if either a Lipschitz condition is met, or if the system is controllable. It might be possible to specify different rates of convergence depending on which of the two conditions is met.

5.4 Designing Metrics

The primary drawback with the RRT-based methods is the sensitivity of the performance on the choice of a metric, ρ, which is not a requirement for dynamic programming. In [13], an RRT variant was introduced that exhibits less sensitivity, but the problem still remains in many instances. All of the results presented [38] were obtained by assigning a simple, weighted Euclidean metric for each model (the same metric was used for different collections of obstacles). Nevertheless, we observed that the computation time varies dramatically for some problems as the metric is varied. This behavior warrants careful investigation into the effects of metrics.

In general, we can characterize the ideal choice of a metric (technically this could be called a pseudometric due to the violation of some metric properties). Consider a cost functional, L, defined as

$$L = \int_0^T l(x(t), u(t))dt + l_f(x(T)).$$

As examples, this could correspond to the distance traveled, the energy consumed, or the time elapsed during the execution of a trajectory. The optimal cost to go from x to x' can be expressed as

$$\rho^*(x, x') = \min_{u(t)} \left\{ \int_0^T l(x(t), u(t))dt + l_f(x(T)) \right\}.$$

Ideally, ρ^* would make an ideal metric because it indicates "closeness" as the ability to bring the state from x to x' while incurring little cost. The ideal metric has appeared in similar contexts as the nonholonomic metric (see [31]), the value function [52], and the cost-to-go function [2,32]. Of course, computing ρ^* is as difficult as solving the original planning problem! It is generally useful, however, to consider ρ^* because the performance of RRT-based planners seems to generally degrade as ρ and ρ^* diverge. An effort to make a crude approximation to ρ^*, even if obstacles are neglected, often leads to great improvements in performance. For classical path planning, nearby states in terms of an l^p metric are easy to reach, but for trajectory design problems, it is more challenging to design a good metric. One good example of work that yields excellent performance through the design of a better RRT metric appears in [21], in which helicopter trajectories are designed by using the cost-to-go for a hybrid system based on trim trajectories as the RRT metric.

5.5 Constructing Motion Primitives

One can always consider more complicated problems. In robotics, the limits are being pushed by rapidly-increasing interest in the design of motions of humanoid robots. These problems typically have state spaces of more than 100 dimensions. One reasonable way to deal with the complexity of these systems is to build a family of higher-level motion primitives. A single primitive might be applicable over a large region of the state space due to Lie group symmetries. As an example, the work in [21] develops a hybrid system for designing helicopter trajectories by sequencing trim trajectories. Each trim trajectory is an input function that would typically be applied by a pilot, and can be considered as a mode of the system. If certain conditions are met, it is possible to execute a transition into another mode. When constructing higher-level primitives, it is important to ensure that as much as possible of the expressiveness of the original system is preserved. In the case of [21], it was established that the resulting system is controllable (although not in the presence of global constraints on X). In related computer graphics work, researchers have begun to develop libraries of motion primitives for human characters, and are designing search algorithms that sequence these motions [41]. These efforts represent a promising direction of research for handling larger systems that arise in robotics and other fields.

5.6 Conclusions

A summary of our approaches to the trajectory design problem in robotics was presented. This problem considers both local and global constraints on the state space. Approximately-optimal dynamic programming approaches

and heuristic-based Rapidly-exploring Random Tree approaches were discussed. Although there have been several successes, many interesting and challenging questions remain in this area.

Acknowledgments. This work was funded in part by NSF CAREER IRI-9875304 and NSF IIS-0118146. I am grateful for the help of Michael Branicky, Peng Cheng, Prashanth Konkimalla, James Kuffner, Andrew Olson, and Libo Yang.

References

1. A. Atramentov and S. M. LaValle. Efficient nearest neighbor searching for motion planning. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 632–637, 2002.
2. T. Başar and G. J. Olsder. *Dynamic Noncooperative Game Theory.* Academic Press, London, 1982.
3. J. Barraquand, B. Langlois, and J. C. Latombe. Numerical potential field techniques for robot path planning. *IEEE Trans. Syst., Man, Cybern.*, 22(2):224–241, 1992.
4. J. Barraquand and J.-C. Latombe. Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles. *Algorithmica*, 10:121–155, 1993.
5. R. E. Bellman. *Dynamic Programming.* Princeton University Press, Princeton, NJ, 1957.
6. R. E. Bellman and S. E. Dreyfus. *Applied Dynamic Programming.* Princeton University Press, Princeton, NJ, 1962.
7. D. P. Bertsekas. Convergence in discretization procedures in dynamic programming. *IEEE Trans. Autom. Control*, 20(3):415–419, June 1975.
8. J. T. Betts. Survey of numerical methods for trajectory optimization. *J. of Guidance, Control, and Dynamics*, 21(2):193–207, March-April 1998.
9. M. Branicky, S. M. LaValle, K. Olsen, and L. Yang. Quasi-randomized path planning. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 1481–1487, 2001.
10. F. Bullo. Series expansions for the evolution of mechanical control systems. *SIAM J. Control and Optimization*, 40(1):166–190, 2001.
11. J. Canny, A. Rege, and J. Reif. An exact algorithm for kinodynamic planning in the plane. *Discrete and Computational Geometry*, 6:461–484, 1991.
12. J. F. Canny. *The Complexity of Robot Motion Planning.* MIT Press, Cambridge, MA, 1988.
13. P. Cheng and S. M. LaValle. Reducing metric sensitivity in randomized trajectory design. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, pages 43–48, 2001.
14. P. Cheng and S. M. LaValle. Resolution complete rapidly-exploring random trees. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 267–272, 2002.
15. M. Cherif. Kinodynamic motion planning for all-terrain wheeled vehicles. In *IEEE Int. Conf. Robot. & Autom.*, 1999.

16. C. Connolly, R. Grupen, and K. Souccar. A Hamiltonian framework for kinodynamic planning. In *Proc. of the IEEE International Conf. on Robotics and Automation (ICRA'95)*, Nagoya, Japan, 1995.
17. D. F. Delchamps. Stabilizing a linear system with quantized output record. *IEEE Trans. Autom. Control*, 35(8):916–926, 1990.
18. B. Donald and P. Xavier. Provably good approximation algorithms for optimal kinodynamic planning: Robots with decoupled dynamics bounds. *Algorithmica*, 14(6):443–479, 1995.
19. B. R. Donald, P. G. Xavier, J. Canny, and J. Reif. Kinodynamic planning. *Journal of the ACM*, 40:1048–66, November 1993.
20. Th. Fraichard and C. Laugier. Kinodynamic planning in a structured and time-varying 2d workspace. In *IEEE Int. Conf. Robot. & Autom.*, pages 2: 1500–1505, 1992.
21. E. Frazzoli, M. A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance and Control*, 25(1):116–129, 2002.
22. L. J. Guibas, D. Hsu, and L. Zhang. H-Walk: Hierarchical distance computation for moving convex bodies. In *Proc. ACM Symposium on Computational Geometry*, pages 265–273, 1999.
23. D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. *Int. J. Comput. Geom. & Appl.*, 4:495–512, 1999.
24. L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. & Autom.*, 12(4):566–580, June 1996.
25. R. Kindel, D. Hsu, J.-C. Latombe, and S. Rock. Kinodynamic motion planning amidst moving obstacles. In *IEEE Int. Conf. Robot. & Autom.*, 2000.
26. J. J. Kuffner and S. M. LaValle. RRT-connect: An efficient approach to single-query path planning. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 995–1001, 2000.
27. R. E. Larson. A survey of dynamic programming computational procedures. *IEEE Trans. Autom. Control*, 12(6):767–774, December 1967.
28. R. E. Larson and J. L. Casti. *Principles of Dynamic Programming, Part II*. Dekker, New York, NY, 1982.
29. J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.
30. J.-P. Laumond. Finding collision-free smooth trajectories for a non-holonomic mobile robot. In *Proc. Int. Joint Conf. on Artif. Intell.*, pages 1120–1123, 1987.
31. J. P. Laumond, S. Sekhavat, and F. Lamiraux. Guidelines in nonholonomic motion planning for mobile robots. In J.-P. Laumond, editor, *Robot Motion Plannning and Control*, pages 1–53. Springer-Verlag, Berlin, 1998.
32. S. M. LaValle. *A Game-Theoretic Framework for Robot Motion Planning*. PhD thesis, University of Illinois, Urbana, IL, July 1995.
33. S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. TR 98-11, Computer Science Dept., Iowa State University, Oct. 1998.
34. S. M. LaValle and M. S. Branicky. On the relationship between classical grid search and probabilistic roadmaps. In *Proc. Workshop on the Algorithmic Foundations of Robotics (to appear)*, December 2002.
35. S. M. LaValle and P. Konkimalla. Algorithms for computing numerical optimal feedback motion strategies. *International Journal of Robotics Research*, 20(9):729–752, September 2001.

36. S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 473–479, 1999.
37. S. M. LaValle and J. J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In *Workshop on the Algorithmic Foundations of Robotics*, 2000.
38. S. M. LaValle and J. J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In B. R. Donald, K. M. Lynch, and D. Rus, editors, *Algorithmic and Computational Robotics: New Directions*, pages 293–308. A K Peters, Wellesley, MA, 2001.
39. M. C. Lin and J. F. Canny. Efficient algorithms for incremental distance computation. In *IEEE Int. Conf. Robot. & Autom.*, 1991.
40. S. R. Lindemann and S. M. LaValle. Incremental low-discrepancy lattice methods for motion planning. In *Submitted to IEEE International Conference on Robotics and Automation*, 2003.
41. C. K. Liu and Z. Popovic. Synthesis of complex dynamic character motion from simple animations. In *SIGGRAPH*, 2002.
42. A. Marigo, B. Piccoli, and A. Bicchi. Reachability analysis for a class of quantized control systems. In *Proc. IEEE Conf. on Decision and Control*, 2000.
43. J. Matousek. *Geometric Discrepancy.* Springer-Verlag, Berlin, 1999.
44. E. Mazer, G. Talbi, J. M. Ahuactzin, and P. Bessière. The Ariadne's clew algorithm. In *Proc. Int. Conf. of Society of Adaptive Behavior*, Honolulu, 1992.
45. B. Mirtich. V-Clip: Fast and robust polyhedral collision detection. Technical Report TR97-05, Mitsubishi Electronics Research Laboratory, 1997.
46. R. M. Murray and S. Sastry. Nonholonomic motion planning: Steering using sinusoids. *Trans. Automatic Control*, 38(5):700–716, 1993.
47. H. Niederreiter. *Random Number Generation and Quasi-Monte-Carlo Methods.* Society for Industrial and Applied Mathematics, Philadelphia, USA, 1992.
48. C. O'Dunlaing and C. K. Yap. A retraction method for planning the motion of a disc. *Journal of Algorithms*, 6:104–111, 1982.
49. I. Pohl. Bi-directional and heuristic search in path problems. Technical report, Stanford Linear Accelerator Center, 1969.
50. J. H. Reif. Complexity of the mover's problem and generalizations. In *Proc. of IEEE Symp. on Foundat. of Comp. Sci.*, pages 421–427, 1979.
51. A. G. Sukharev. Optimal strategies of the search for an extremum. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 11(4), 1971. Translated from Russian, *Zh. Vychisl. Mat. i Mat. Fiz.*, 11, 4, 910-924, 1971.
52. S. Sundar and Z. Shiller. Optimal obstacle avoidance based on the Hamilton-Jacobi-Bellman equation. *IEEE Trans. Robot. & Autom.*, 13(2):305–310, April 1997.
53. G. J. Toussaint, T. Başar, and F. Bullo. Motion planning for nonlinear underactuated vehicles using hinfinity techniques. Coordinated Science Lab, University of Illinois, September 2000.
54. H. Weyl. Über die Gleichverteilung von Zahlen mod Eins. *Math. Ann.*, 77:313–352, 1916.

Control of Nonprehensile Manipulation

Kevin M. Lynch and Todd D. Murphey

Mechanical Engineering Dept.
Northwestern University
Evanston, IL 60208 USA

Abstract. *Nonprehensile manipulation* is the process of manipulating a part without a form- or force-closure grasp. Without such a grasp, the part is free to roll, slide, or break contact with the robot(s) manipulating it. Controlling the motion of a part using nonprehensile manipulation becomes the challenging problem of controlling a dynamic system with equations of motion incorporating the robot and part geometry, friction and restitution laws, and changing dynamics due to changing contact states. Drawing from the work of others and our own previous work, in this paper we pose several open problems in the control of nonprehensile manipulation.

1 Introduction

Nonprehensile manipulation is the process of manipulating a part without a form- or force-closure grasp. Without such a grasp, the part is free to roll, slide, or break contact with the robot(s) manipulating it. Examples of nonprehensile manipulation are shown in Fig. 1.

Many nonprehensile manipulation tasks share common characteristics. First, nonprehensile manipulation often involves *unilateral controls*. The robot manipulates the part by contact forces. The controls may be unilateral because the robot can only push the part, not pull it. There are no kinematic equality constraints guaranteeing the actuators' connection to the part at all times. Indeed, this is part of the power of nonprehensile manipulation — the nature of the connection of the actuators to the part can be changed. Second, nonprehensile manipulation exhibits *nonsmooth dynamics*. The robot manipulates the part through a set of contacts. When one or more of the contacts changes its mode, say from sticking contact to slipping, or from slipping to no contact, the dynamics of the system changes in a nonsmooth manner. Some of these switches can be controlled, as when a finger pushing a part moves away and recontacts it at another location or when an impact is used for the purpose of control. Other switches are uncontrolled, as when the friction force between an object and a surface it is sliding on changes due to indeterminacy in the support force distribution. This kind of uncertainty is inherent in many manipulation systems. Finally, both *part geometry* and *robot geometry* are important. Since forces are applied to the part through contact, the set of available contacts afforded by the geometry of the part is critical in determining the controllability of the part and in devising a control law. The

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 39–57, 2003

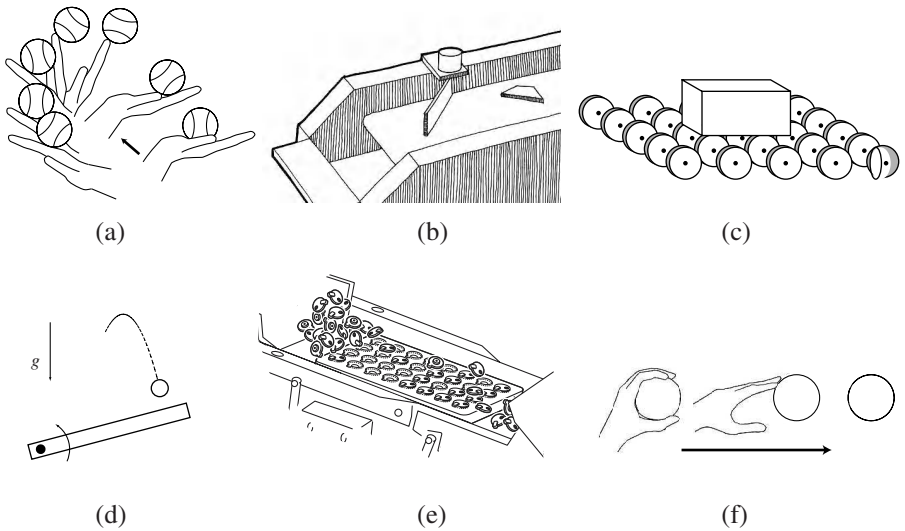


Fig. 1. Examples of nonprehensile manipulation. (a) A human juggler performs a "butterfly," dynamically rolling a ball from the palm to the back of the hand. (b) A part is carried on a constant-speed conveyor, and a rotating fence pushes the part back upstream while rotating it. By a series of pushing motions and conveyor drift, the part can be positioned and oriented. (c) A box is carried on an array of rolling wheels. The wheels may change orientation and speed. (d) A single-joint robot bat-juggles a puck in a gravity field. (e) The Sony APOS parts feeder orients parts in a tray of part-shaped depression. Parts wash over the vibrating tray, and parts which fall in a hole in the right orientation are trapped, while parts in a wrong orientation bounce out and continue into a recirculation bin. The vibration and hole shapes are designed to "juggle" parts into place in an open-loop fashion. (Figure taken from [33] and used with permission from MIT Press.) (f) Throwing a ball. Note that the initial carry phase is prehensile.

robot's geometry (and kinematics) is also important, as it determines how the robot can contact the part.

Some nonprehensile manipulation systems are *underactuated* and others are *overactuated*. In the former case, there are fewer robot actuators than part degrees-of-freedom. Examples include robot pushing and juggling. The ability to break contact and recontact leads to favorable controllability properties despite the underactuation. The latter case includes distributed manipulation by actuator arrays, where many actuators interact with a single part. In this case, the dimension of the input space can be quite large, while the dimension of the output space is small (typically three or six). For such a system the mechanics may be nonsmooth due to stick-slip phenomena associated with friction. Additionally, the mechanics of these systems are typically not well posed, in the sense that for a given set of inputs, solutions are not unique.

Control problems in nonprehensile manipulation include:

- *Defining sensible and testable notions of controllability.* The state space of a manipulation system naturally decomposes into a robot state space and a part state space, and we are typically interested in the local and global controllability (or "feedability" in the case of parts feeding) properties of the part. Some progress has been made for stratified systems in [15]. Ideally we would be able to derive tests for controllability in terms of the kinematics and dynamics of the robot, the geometry and mass properties of the part, and friction and restitution coefficients.
- *Reduction of quasistatic and dynamic nonprehensile manipulation control systems to kinematic systems for motion planning purposes.* Recent progress has been made in characterizing when (smooth) mechanical control systems can be treated as kinematic systems in [9,10,19]. Initial results are given in [28,34] for manipulation systems with uncertainty.
- *Trajectory generation.* Given the nonsmooth dynamics of the manipulation system, the problem is to find a set of controls to take the system to the goal state. This includes finding a sequence of contact states as well as continuous motion plans within each contact state [29,43].
- *Stabilizing a planned trajectory or equilibrium.* The problem of positioning and orienting a part (parts feeding) is to find a control law to stabilize an equilibrium.

Another important issue is reducing the sensing requirements needed to implement a successful control law. Full state sensing may not be required if we can identify equivalence classes of states where the same control will take the system closer to the goal state [14].

In this paper, we will consider a few different nonprehensile manipulation systems and list some open questions in control related to the issues raised above. The formulation in this paper is heavily influenced by previous work on rolling manipulation [5,12,17,31,32], juggling [6–8,11,27,39,40], pushing [1,26,28], control of nonsmooth manipulation systems [15,29,34,36,43], and distributed manipulation [3,22,34,36].

2 Definitions

In this section we propose some common definitions for nonprehensile systems. Nonprehensile manipulation systems come in so many shapes and flavors, however, that the utility of these definitions is limited. Nonetheless, these definitions provide a starting point for our discussion.

A nonprehensile manipulation system consists of a robot R and a part P. (The definition can easily be extended to multiple parts.) The configuration of the robot is written $q_R \in \mathcal{Q}_R$, and its dimension is m_R. Let z_R be the state of the robot, where $z_R = (q_R, \dot{q}_R)$ if the system is dynamic, and $z_R = q_R$ if the

system is quasistatic. Let $\mathcal{Z}_R$ be the robot state space, and $\dim(\mathcal{Z}_R) = n_R$. The control applied to the robot is $u \in \mathcal{U}$.

Analogously, we can define the configuration of the part $q_P \in \mathcal{Q}_P$, its dimension $m_P = \dim(\mathcal{Q}_P)$, and its state $z_P \in \mathcal{Z}_P$ of dimension n_P. The configuration of the entire nonprehensile manipulation system is then written $q = (q_R, q_P) \in \mathcal{Q} = \mathcal{Q}_R \times \mathcal{Q}_P$, and the state is written $z = (z_R, z_P) \in \mathcal{Z} = \mathcal{Z}_R \times \mathcal{Z}_P$, where $\dim(\mathcal{Z}) = n_R + n_P = n$. Although the meaning of a "part" is intuitively obvious, we distinguish it from unactuated degrees-of-freedom of an underactuated robot by requiring an n-dimensional subset of the system state space where the control u has no influence on $\dot{z}_P$ (e.g., the robot can break contact with the part).

We call the system an *underactuated nonprehensile system* if $m_R < m_P$, and an *overactuated nonprehensile system* if $m_R > m_P$ and the evolution of z_P is affected by more than m_P actuators (i.e., the actuators are somehow connected to the part).

We will assume that a "part" is a rigid body. A frame $\mathcal{F}^P$ is attached to the center of mass of the rigid body part and aligned with the principal axes of inertia. The configuration of a spatial part is the matrix $g \in SE(3)$ representing the displacement of $\mathcal{F}^P$ relative to an inertial frame $\mathcal{F}^w$, where

$$g = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix},$$

or simply $g = (p, R)$ for short, where $R \in SO(3)$ is a 3×3 rotation matrix and $p \in \mathbb{R}^3$ is the position of the origin of $\mathcal{F}^P$ in $\mathcal{F}^w$. The velocity of the part is an element ξ of $se(3) = \mathbb{R}^6$, the Lie algebra associated to $SE(3)$. We write $\xi = (v, \omega)$, where $\omega \in \mathbb{R}^3$ is the angular velocity and $v \in \mathbb{R}^3$ is the linear velocity of the body written in $\mathcal{F}^P$. The kinematic equations are

$$\dot{p} = Rv, \quad \dot{R} = R\widehat{\omega}, \quad \text{where } \widehat{\omega} = \widehat{\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}. \tag{1}$$

Let f_P be the force applied by the robot to the part, written in the part frame $\mathcal{F}^P$. Let r_P be the vector to a point on the line of action of f_P, written in $\mathcal{F}^P$, so that the torque acting on the part is $\tau_P = r_P \times f_P$. Together, (f_P, τ_P) is called a generalized force or *wrench*. Let m be the part mass, $\mathbb{M} = \text{diag}(m, m, m)$, and $\mathbb{J} = \text{diag}(I_1, I_2, I_3)$ be the body-fixed inertia tensor.

The evolution of a nonprehensile system is of the form

$$h(q) \geq 0 \tag{2}$$

$$f_P = c(q, \dot{q}, u), \quad f_P h(q) = 0 \tag{3}$$

$$M(q_R)\ddot{q}_R + \dot{q}_R \Gamma(q_R)\dot{q}_R + G(q_R) = T(q_R)u - J^T(q) f_P \tag{4}$$

$$\begin{bmatrix} \dot{p} \\ \dot{R} \end{bmatrix} = \begin{bmatrix} Rv \\ R\widehat{\omega} \end{bmatrix}, \quad \begin{bmatrix} \mathbb{M} & 0 \\ 0 & \mathbb{J} \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} \omega \times mv \\ \omega \times \mathbb{J}\omega \end{bmatrix} = \begin{bmatrix} f_P \\ \tau_P \end{bmatrix} + \begin{bmatrix} f_E(q_P, \dot{q}_P) \\ \tau_E(q_P, \dot{q}_P) \end{bmatrix} \tag{5}$$

$$\text{when } h(q) = 0, \dot{h}(q) < 0: \qquad (q^+, \dot{q}^+) = (q, \eta(q, \dot{q})), \tag{6}$$

where $h(q) \in \mathbb{R}$ is a "distance" function between the part and the robot, describing the unilateral configuration constraints that the part cannot penetrate the robot; $c(q, \dot{q}, u) \in \mathbb{R}^3$ gives the force applied to the part as a function of the system state (including the contact geometry), the control, and the friction law (the contact force is zero if the distance between the part and the robot is nonzero); (4) denotes the robot subsystem dynamics, and $J^T(q)$ describes how the force from the part affects the robot; (5) denotes the kinematics and dynamics of the part subsystem, where $(f_E, \tau_E) \in \mathbb{R}^6$ indicate any environmental forces such as gravity or frictional forces as a part slides over a stationary surface; and (6) denotes the impact law, which leaves the configuration of the system unchanged but changes the velocity discontinuously. Impacts between the part and the environment could also be modeled.

The coupling between the robot and part subsystems comes through the contact law c and the impact law η. Controllability and control of nonprehensile manipulation systems depends intimately on the friction and impact laws, and the vast majority of the complexity of controlling nonprehensile systems comes from them. In particular, the contact law c is nonsmooth due to geometry (changes in robot-part contact geometry as contacts are established and broken) and the friction law (switches from slipping to sticking contact). Each contact mode, which includes the set of contacts and their current state (slipping or sticking), leads to its own set of dynamic equations, and a switch in the contact mode results in a change in the equations. Some of these switches can be controlled, while others cannot. Thus nonprehensile systems inherit the difficulties of general hybrid control systems.

In Sections 3 and 4, we specialize the equations of motion to systems with much more structure.

2.1 Controllability Definitions

Let $R^{\mathcal{V}}(z)$ denote the set of reachable states from the initial state z by feasible trajectories remaining in $\mathcal{V} \subseteq \mathcal{Z}$. (In this definition, note that we are not concerned with time.) It is possible to define the standard controllability terms for the nonprehensile system on $\mathcal{Z}$, such as accessibility, controllability, etc. However, we are interested more in what can be done with the part, i.e., the subsystem (5), than the entire system. Let $\mathcal{V}_P \subset \mathcal{Z}_P$ be a neighborhood of z_P, and let $R_P^{\mathcal{V}_P}(z)$ be the reachable states of the part starting from the initial system state z by trajectories of the part confined to $\mathcal{V}_P$.

Definition 1. The part is *locally-locally accessible* from z if $\dim(R_P^{\mathcal{V}_P}(z)) = n_P$ for any $\mathcal{V}_P$ containing z_P in its interior.

Definition 2. The part is (globally) *controllable* from z if $R_P^{\mathcal{Z}_P}(z) = \mathcal{Z}_P$. For the rest of the paper, the term "controllable" refers to this global concept.

Definition 3. The part is *locally-locally controllable* from z if $z_P \in \text{int}(R_P^{\mathcal{V}_P}(z))$ for all neighborhoods $\mathcal{V}_P$ of z_P.

Definition 4. Given a set of initial states $\mathcal{W} \subset \mathcal{Z}$, we say the part is *controllable to* z_P from $\mathcal{W}$ if $z_P \in R_P^{\mathcal{Z}_P}(w)$ for all $w \in \mathcal{W}$. We will sometimes say the part is *feedable* to z_P from $\mathcal{W}$.

Since the definitions are not concerned with time, we borrow the "local-local" modifiers from Haynes and Hermes [16]. We could also define notions of configuration controllability and equilibrium controllability for a part [20].

3 Dynamic Underactuated Nonprehensile Manipulation

In this section we discuss the control of a rigid-body part by contact forces on its surface. The shape of the part, as well as the contact friction coefficient, determines the set of forces that can be applied to the part. Unlike grasping manipulation, where the robot makes multiple contacts with the part to establish a grasp which is capable of resisting all external wrenches, in this section we will focus on the case where the robot makes a single point of contact with the part at any given time, and study the resulting set of possible part trajectories. Let $d = 2, 3$ be the dimension of the part's configuration space. Since $m_P = \dim(SE(d)) = 6$ (3) for a spatial (planar) part and a point in space (the plane) has only three (two) degrees-of-freedom, the nonprehensile manipulation is necessarily underactuated.

We will first consider the part by itself, assuming a point robot that moves freely around the part. This is the subsystem (5) with (f_P, τ_P) as the control. We will then consider more realistic models of robot interaction with the part.

3.1 Part Geometry Only

Let ∂P be the surface (boundary) of the part P. Let r_P be the vector to a point on ∂P, and let $\hat{n}$ be the inward-pointing unit normal at this point, measured in $\mathcal{F}^P$. (We will assume that $\hat{n}$ is well-defined at each $r_P \in \partial P$, although the definition could be generalized to handle nondifferentiable points such as vertices.) Then, in the absence of friction, a contact at this point will give rise to a wrench on the part

$$f_P = k\hat{n}, \quad k \geq 0 \tag{7}$$

$$\tau_P = r_P \times f_P, \tag{8}$$

where $k \geq 0$ is the force magnitude (encoding the unilateral constraint) and f_P and τ_P are the force and torque measured in $\mathcal{F}^P$. In the presence of dry Coulomb friction with a friction coefficient μ, (7) becomes the constraint

$$f_P = k(\hat{n} + \gamma\hat{n}^\perp), \quad k \geq 0, \quad 0 \leq \gamma \leq \mu \tag{9}$$

where $\hat{n}^\perp$ is any unit vector satisfying $\hat{n}^T\hat{n}^\perp = 0$ (i.e., in the contact tangent plane). The constraint (9) defines a *friction cone*, where $\alpha = \tan^{-1}\mu$ is the friction cone half-angle (Fig. 2). If the contact is slipping, then

$$f_P = k(\hat{n}+\gamma\hat{n}^\perp), \quad k \geq 0, \quad \gamma = \mu \tag{10}$$

and $\hat{n}^\perp$ is opposite the relative slip direction in the tangent plane.

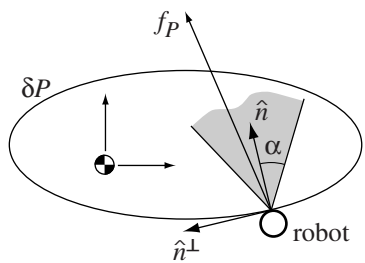


Fig. 2. The contact friction cone for a planar part. The contact force f_P must lie somewhere in the cone of half-angle $\alpha = \tan^{-1}\mu$.

For a point $r_P \in \partial P$, let $W(r_P)$ denote the set of wrenches that can be applied satisfying the friction cone constraint (9). Let $W(\partial P) = \bigcup_{r_P \in \partial P} W(r_P)$ be the set of all wrenches that can be applied to the part by point contact. Define the control system $(*)$ to be the part subsystem (5) with $(f_E, \tau_E) = 0$ and the control $(f_P(t), \tau_P(t)) : \mathbb{R} \to W(\partial P)$.

Theorem 1. *The control system* $(*)$ *is locally-locally controllable at zero velocity states* $(q_P, 0)$ *for* $d = 2$ *unless* $\mu = 0$ *and the part is a disk centered at its center of mass.*

In other words, the planar part is locally-locally controllable (Definition 3) if it can be pushed at all points along its boundary.

Now define the system $(\dagger)$ to be the part subsystem (5) with $(f_E, \tau_E) = 0$ and the control $(f_P(t), \tau_P(t)) : \mathbb{R} \to W(r_P)$ for a fixed point of contact r_P.

Theorem 2. *For any planar part* $(d = 2)$ *and any* $\mu > 0$*, there exists a* $r_P \in \partial P$ *such that the control system* $(\dagger)$ *is controllable.*

In other words, any planar part is controllable (Definition 2), but not locally-locally controllable, by contact forces through a single point if the part is not frictionless. A sufficient condition for the choice of r_P to render $(\dagger)$ controllable for $d = 2$ is that the contact normal $\hat{n}$ pass through the center of mass.

Proofs for these theorems can be found in [25,29]. Both rely in part on the geometry of ∂P and the implied $W(\partial P)$. Theorem 1 is also satisfied by a finite

set of contact points, i.e., $(f_P, \tau_P) \in W(r_{P_1}) \cup W(r_{P_2}) \cup \ldots \cup W(r_{P_k})$, and both theorems are satisfied by choosing a discrete set of wrenches belonging to each $W(r_{P_i})$ for the finite set of contacts $\{r_{P_1}, \ldots, r_{P_k}\}$. Therefore, the discrete set of contact forces can be considered unilateral jet thrusters arranged along the boundary of the part, with the added constraint that only one of the thrusters can be active at a time. Unilateral thruster control of a 2D or 3D spacecraft may be more intuitive than pushing a part on its boundary.

These results indicate the possibility of controlling a rigid-body part by point contact along its boundary, ignoring how these contacts may be established by a robot. Some open questions include:

Problem 1. Complete characterization of controllability and local-local controllability for the systems $(*)$ and $(\dagger)$ for spatial bodies $(d = 3)$. A related problem is finding minimum unilateral thruster configurations for controlling spatial rigid bodies.

Problem 2. The proof of (global) controllability for the system $(\dagger)$ is a laborious near-constructive proof. On the other hand, it is not difficult to prove global controllability for systems (a) satisfying the Lie algebra rank condition (LARC), (b) subject to a phase-volume-conserving drift vector field, and (c) evolving on a compact configuration space (see, e.g., [18,21]). For nonprehensile (or jet thruster) manipulation of a rigid body, it is easy to establish the LARC, and the drift field is phase-volume-conserving. The configuration space is not compact, however; $SE(d)$ is a semi-direct product of a noncompact space $\mathbb{R}^d$ and a compact space $SO(d)$. We are not aware of general (global) controllability results for such systems, but we expect that the interaction between the compact and non-compact subsystems can be exploited in a systematic manner.

Problem 3. The trajectory generation problem is to find thrusts (wrenches) as a function of time yielding a feasible trajectory between the initial state and the goal state.

Problem 4. Imagine a controllable body which is not locally-locally controllable. What are the closed orbits achievable by this system? Although it is not possible to stabilize an equilibrium, is it possible to stabilize an orbit? Describe the control law. This question is relevant to trajectory generation and feedback control of severely underactuated space and underwater vehicles, with or without unilateral thrust constraints.

Another version of the control problem of the part subsystem (5) is pushing a planar body over a frictional support surface. In this case, the environmental forces (f_E, τ_E) are significant and uncertain due to the unknown distribution of support over the area of the body. A quasistatic version of this problem is studied in [28].

3.2 With Manipulator Constraints

Instead of manipulating a planar part at arbitrary points along its perimeter, imagine manipulating the part with a robot arm with a single revolute joint, as shown in Fig. 3. For the purposes of the example, let the part be a polygon. If the part remains rigidly attached to the robot, its accessible state space is two-dimensional and is parameterized by the robot state $(\theta, \dot{\theta})$. If the part can also roll relative to the robot with sticking at the contact point, it can achieve two more motion directions in its state space, parameterized by $(\phi, \dot{\phi})$. If the contact can also independently slip on the robot, the part can achieve two more motion directions, parameterized by the contact point on the robot arm and its rate of change. Thus it may be possible for the robot to transfer the part to an open subset of its six-dimensional state space $\mathcal{Z}_P$ without even breaking contact with the part. The robot can then throw the part, increasing the volume of the reachable state space. More interestingly, the robot could break and recontact the part to increase the reachable state space, or bat-juggle the part (Section 3.3).

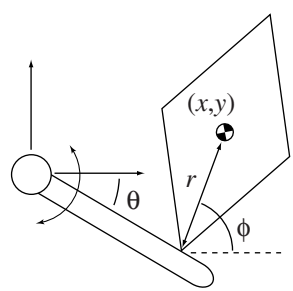


Fig. 3. A one-joint arm manipulating a polygon at a vertex.

Let's study how the system of Fig. 3 can be expressed in our canonical form (2 – 6). We can express four different systems depending on the contact mode: the robot either breaks contact or stays in contact with the part, and if in contact, the contact either sticks, slips left, or slips right. Which of these systems actually occurs can be determined by examining the current state and control and incorporating the friction law (except in cases of ambiguity and inconsistency, which may occur due to the Coulomb friction law [33]). Sticking contact implies a kinematic equality constraint, while slipping contact implies a force equality constraint (the force must lie on the appropriate edge of the friction cone).

We will assume that the robot moves so as to stay in contact with the vertex of the polygon, and we will assume zero friction at the contact. A slipping contact model with dry Coulomb friction is similar, except in this case the contact force lies along a friction cone edge, not the contact normal.

Let the mass of the part be m, its inertia about the center of mass be I_P, and the robot inertia about the pivot be I_R. We have $q_R = \theta$, $q_P = (x, y, \phi)$, and $q = (q_R, q_P)$. For simplicity in this example, we will break with the notation of (2 – 6) by expressing the contact wrench (f_P, τ_P) and the part velocity and acceleration in a frame fixed to the part center of mass but always aligned with the inertial frame fixed to the pivot of the robot. Except for this, Equations (11 – 14) correspond directly to Equations (2 – 5) in the canonical form:

$$h(q) = \cos\theta(y - r\sin\phi) + \sin\theta(r\cos\phi - x) \geq 0 \tag{11}$$

$$f_P = \begin{bmatrix} f_{Px} \\ f_{Py} \end{bmatrix} = k \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix}, \quad \tau_{Pz} = -f_{Py} r\cos\phi + f_{Px} r\sin\phi \tag{12}$$

$$I_R\ddot{\theta} = u - J^T(q) f_P, \quad J^T(q) = (r\sin\phi - y, x - r\cos\phi) \tag{13}$$

$$\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I_P \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} f_{Px} \\ f_{Py} \\ \tau_{Pz} \end{bmatrix}. \tag{14}$$

Since we assume that contact is maintained, we have $h(q) = \dot{h}(q) = \ddot{h}(q) = 0$. It is the last of these that is used in solving the equations of motion. If in solving the equations of motion we find that the contact force magnitude k becomes negative, we know that the "maintain contact" assumption has been violated.

The exact equations of motion for even this simple system are complex, so we omit them (see [29]). The robot control torque u and the contact force magnitude k are related by

$$u = \alpha_1(z) + k\alpha_2(q),$$

where $\alpha_1(z)$ is the torque needed for the robot to stay in contact with the moving rod and $\alpha_2(q)$ is the extra torque needed to provide one unit of contact force to the rod. With a feedback transformation to use k as the control, the part dynamics can be written in the form

$$\dot{z}_P = g_0(z_P) + k g_1(z_P),$$

where $g_0(z_P) = (\dot{x}, \dot{y}, \dot{\phi}, 0, 0, 0)^T$, $g_1(z_P) = (0, 0, 0, -\frac{\sin\theta}{m}, \frac{\cos\theta}{m}, -\frac{r\cos(\phi-\theta)}{I_P})^T$, and $\theta = \text{atan2}(y - r\sin\phi, x - r\cos\phi)$. An examination of the Lie algebra of $\{g_0, g_1\}$ readily shows that the part is locally-locally accessible (Definition 1) at generic states when the contact point is not at the robot's pivot. In other words, the robot can manipulate the part to a full six-dimensional subset of its state space. This is a desirable property, but a limited description of the capability of the system. Some relevant open problems are listed below.

Problem 5. Controllability of rolling manipulation has been heavily studied when the rolling velocities are directly controlled (a kinematic system). Controllability of dynamic nonprehensile rolling systems, where the body rolls on

a single actuated surface, has received less attention (but see, e.g., [12,17,30]). If one body is free to roll on an actuated surface, what are the conditions for local and global controllability of the body? How much actuation does the surface need? How do we do motion planning? Can we derive feedback laws to stabilize equilibria or trajectories? Stabilizing rolling motion is inherently easier than stabilizing slipping motions, as slip depends critically on the friction coefficient, which may be unknown or varying.

Problem 6. In general, a part may go through a sequence of rolling, slipping, and free flight phases during dynamic nonprehensile manipulation. What are the reachable states for the part from a given initial state? Can we find a sequence of phases, and smooth motion plans within each phase satisfying the phase transition conditions at the switches, that will transfer the part to a desired state? Some early work in this direction is reported in [29,43].

Problem 7. The "dexterous kinematic workspace" of a robot arm is usually defined as the set of configurations a gripper can reach with any orientation. A "dynamic workspace" for a robot-part system might describe the set of reachable part states from a given initial system state. In particular, the robot can send the part to points outside the kinematic workspace by throwing.

3.3 Juggling

Bat-juggling is the process of manipulating a part by a series of impacts with the part moving in ballistic flight in between (e.g., Fig. 1(d)). A typical goal of a juggling system is to stabilize a desired limit cycle. Brogliato and Zavala-Rio [6] have defined the term "controllable through the impacts" to describe juggling controllability of a part, and Spong [41] considered the controllability of a batted air hockey puck. Control laws for juggling planar pucks and balls in space, and experimental implementations, are described in [7,8,11,27,39,40]. Brogliato and Zavala-Rio [6] outline a general framework for studying juggling systems. Here we simply mention some relevant problems.

Problem 8. Most work on juggling has focused on parts modeled as point masses. What about parts with non-trivial geometry? Are such parts controllable? Can we stabilize limit cycles? How much feedback is needed to do so? How do we plan a sequence of impacts to take the part to a desired state? What is the effect of different friction and impact laws?

Problem 9. For a given part of non-trivial geometry, is it possible to design an open-loop robot motion, as well as the robot shape, so that the impacts will cause the part to converge from a large basin of attraction to a desired resting state relative to the robot? This is essentially the APOS parts feeder (Fig. 1(e)) design problem for a single part.

4 Distributed Manipulation and Open Problems

Distributed manipulation is an area of robotics dedicated to the consideration of systems with many inputs affecting a part. One of the simplest examples, and the one we will consider here, is that of a planar array of actuators, like the one depicted in Fig. 1(c). If one places a part upon such an array, each of these actuators can exert a force on the part, together creating a net force and torque on the part. Therefore, using many actuators, one can control the location of the part in $SE(2)$. Hence, we can consider this system to be *overactuated*, in the sense that there are many more inputs than outputs. It can be difficult to model these systems, partly because the underlying mechanics can be both nonsmooth and nondeterministic. There have been several attempts to address the issue of overactuation. The work of [4,13,23,24,42] presents a methodology where the forces produced by the individual actuators are idealized as a force field acting on the part. A different modeling methodology used in previous work by the second author in [34,36,37] has suggested both analytically and experimentally that a "quasistatic" approach may work well, and has the advantage of being first order. This is the approach we will describe here.

We should comment here that in some special cases, the feedback control problem for distributed manipulation is very straightforward. Consider the case where the distributed manipulation system is fully actuated, in the sense that every actuator can exert a force of an arbitrary magnitude in an arbitrary direction while all satisfying the kinematic constraints associated with contact between the actuators and the part. In the example in Fig. 1(c), these kinematic constraints are associated with the nonslip constraints between the wheels and the part. In this case, perhaps not surprisingly, one can effectively reduce such a system from an overactuated system to a system with three inputs and three outputs by enforcing kinematic compatibility constraints between the actuators. (This functions in precisely the same manner as the differential on an automobile, which ensures that both front wheels satisfy the same kinematic constraints to reduce wear on tires due to slipping.) By doing this, one gains direct control over the velocity of the part. This method additionally provides desirable robustness properties with respect to actuator uncertainties. For more details on this method, see [34]. This control law is smooth, which we will see cannot necessarily be expected for overactuated systems. Moreover, it can only be used for fully actuated systems, which requires that every actuator in the plane have two degrees of freedom. Many interesting examples do not satisfy this requirement, such as micro-electromechanical system (MEMS) arrays which typically only have a one degree of freedom actuator at every location. We will consider a similar system with only one degree of freedom actuators shortly as an example to illustrate the questions we are interested in.

Let us consider the dynamic representation (in 2-6) of distributed manipulation. As before, let q_R and q_P denote the configuration of the array/part

system, consisting of the part's planar location, and the variables that describe the configuration of each actuator array element. We treat the part and the array element contact as a rigid body contact system. We assume point contacts between the part and the array elements, and that the part's contact with the manipulating surface is governed by the Coulomb friction law at each contact point. When the part is slipping against the i^{th} actuator we get a force f_P^i which is proportional to the slipping velocity between the actuator and the part. When it is not slipping we get a force which is just the actuator input u^i scaled to account for the actuator and part inertias. The robot dynamics (in (4)) are simply the inputs, again scaled to account for actuator and part inertias. This leaves the part dynamics, as in (5). The difficulty lies in the fact that as the actuators stick and slip against the part, the dynamics change discontinuously. Because the dynamics are different for every combination of stick and slip for each actuator, there are 2^n dynamics for n actuators. Rather than attempt to analyze such a complicated system, we instead choose a quasistatic approach, described next.

If $\omega_i(q)$ is the one form describing the kinematic constraint between the part and the i^{th} actuator, then $\omega_i(q)\dot{q}$ describes the relative motion of the contact between the part and the actuator. If $\omega_i(q)\dot{q} = 0$, the contact is not slipping, while if $\omega_i(q)\dot{q} \neq 0$, then $\omega_i(q)\dot{q}$ describes the slipping velocity. In general, the moving part will be in contact with the actuator array at many points. From kinematic considerations, one or more of the contact points must be in a slipping state, thereby dissipating energy. The *power dissipation function* measures the part's total energy dissipation due to contact slippage.

Definition 5. The *Dissipation* or *Friction Functional* for an n-contact state is defined to be

$$\mathcal{D} = \sum_{i=1}^{n} \mu_i N_i \mid \omega_i(q)\dot{q} \mid \tag{15}$$

where μ_i and N_i are the Coulomb friction coefficient and normal force at the i^{th} contact, which are assumed known.

Since there will generally not exist a motion where all of the contacts can be simultaneously slipless (although our example of a fully actuated system is a case when such a condition exists), we are led to the following concept for finding the governing equations.

Power Dissipation Principle: With $\dot{q}$ small, a part's motion at any given instant is the one that minimizes $\mathcal{D}$.

The *power dissipation method (PDM)* assumes that the part's motion at each instant is the one that instantaneously minimizes power dissipation due to contact slippage. This method is adapted from the work of [2] on wheeled vehicles. An interesting topic to explore in this context is finding a

concrete correspondence between solutions to the PDM and solutions to the dynamics as in (2-6). In particular, one would like to know if for every choice of inputs for a control system on $\mathcal{Q}$ there exists a corresponding choice of inputs for a control system on $T\mathcal{Q}$ such that the trajectory on $\mathcal{Q}$ is merely the projection of the other trajectory on $T\mathcal{Q}$. Although we do not discuss them here, preliminary results have been made in [34] based on results in [19]. In [35], we showed that the power dissipation approach generically leads to multiple model systems defined next. The intuition behind this is based on the fact that both μ_i and N_i typically depend on the configuration, $q \in \mathcal{Q}$. Therefore, as the set of supports, coefficient of friction, and center of mass location all change, a different set of contacts are slipping. For each one of these choices of slipping or not slipping the model is different. Moreover, the transitions between these models can be modeled as discontinuous jumps from one model to the next. In the case of distributed manipulation, this switching corresponds to the switching among different contact states (i.e., different sets of slipping contacts) due to variations in contact geometry and surface friction properties. Other examples can be found in [38]. Now we proceed to our formal description of such systems.

Definition 6. A control system Σ is said to be a *multiple model driftless affine system (MMDA)* if it can be expressed in the form

$$\Sigma: \qquad \dot{q} = g_1(q)u_1 + g_2(q)u_2 + \cdots + g_m(q)u_m \tag{16}$$

where for any q and t, $g_i \in \{\gamma_{\alpha_i} | \alpha_i \in I_i\}$, with I_i an index set, γ_{α_i} analytic in (q,t) for all α_i, and the controls $u_i \in \mathbb{R}$ piecewise constant and bounded for all i. Moreover, letting σ_i denote the "switching signals" associated with g_i (which will be referred to as "MMDA maps"),

$$\begin{aligned} \sigma_i : \mathcal{Q} \times \mathcal{R} &\longrightarrow \mathbb{N} \\ (q,t) &\longrightarrow \alpha_i \end{aligned}$$

then the σ_i are measurable in (q,t).

An MMDA system is a driftless affine nonlinear control system where each control vector fields may "switch" back and forth between different elements of a finite set. The σ_i which regulate this switching may not be known, so we have no guarantees about the nature of the switching except that it is, by assumption, measurable. In the context of control, we now have a synthesis problem. For instance, if the array is such that each array element can only exert a force in one direction (in $\mathbb{R}^2$), then the system generally *must* have some slipping, thereby leading to nonsmooth effects as the model describing the dynamics changes discontinuously.

Now we proceed to an example. Figure 4 shows on the left a photograph of an experiment at Caltech which has been used previously to test algorithms for distributed manipulation. In the photograph we see four wheels all oriented towards the origin. These wheels are not allowed to change direction–hence each actuator is a one degree of freedom actuator. We use a

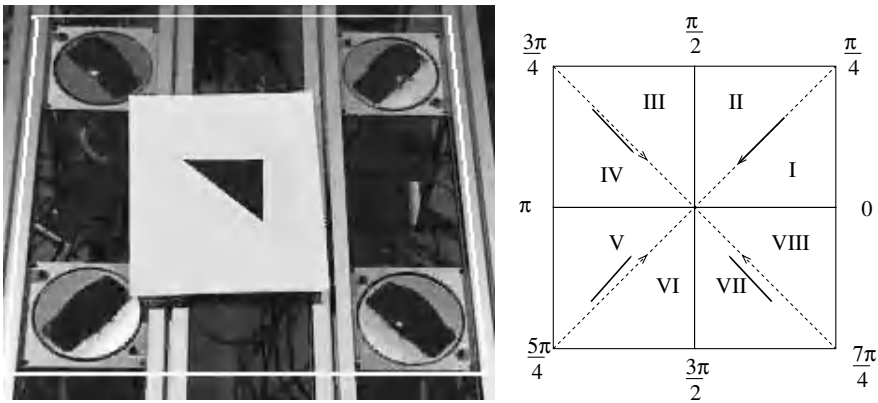


Fig. 4. Photograph and cartoon of 4 cell distributed manipulator.

piece of plexiglass (for purposes of visualization) on top of the four wheels. The white line seen in the photograph indicates the outline of the plexiglass. The goal, then, is to control the center of mass to the origin in $\mathbb{R}^2$ with a desired orientation of $\theta = 0$. To do this, we obtain feedback of the plexiglass' configuration by affixing a piece of paper with a black triangle (also seen in the photo) whose right angle corner coincides with the plexiglass' center of mass. Using this, we obtain the position and orientation of the plexiglass through visual feedback. The right hand side of Fig. 4 is a cartoon of the experiment, where the four arrows correspond to actuators and the regions denoted by **I-VIII** and **0-**$\frac{7\pi}{4}$ will be important in our subsequent description of the kinematics described by the PDM.

Note that this system thus described is overactuated because there are four inputs and only three outputs. Now we describe the kinematics of this system. Assume the coefficient of friction is the same for all four actuators. Then the model describing the kinematics will only change as the center of mass moves across the array. The solution to minimizing $\mathcal{D}$ always consists of choosing the constraints $\omega_i(q)$ that have the most dissipation and finding the kinematics that satisfy those constraints (see [2,35]). Therefore, the actuator wheel nearest to the center of mass will have both its "rolling" constraint and its "sideways" slip constraint satisfied. The actuator wheel second closest to the center of mass will have one of its two constraints satisfied. In the case of the wheels shown in the figure, it will be the rolling constraint. For details on this analysis, see [36]. Denote the actuator input associated with the closest actuator by u_i and the actuator input associated with the second closest actuator by u_j. Then these considerations lead to kinematics of the form:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = g_1 u_i + g_2 u_j \tag{17}$$

where

$$g_1 \in \begin{bmatrix} \frac{-y_i}{(x_j-x_i)\sin(\theta_j)+(y_i-y_j)\cos(\theta_j)} \\ \frac{x_i}{(x_j-x_i)\sin(\theta_j)+(y_i-y_j)\cos(\theta_j)} \\ \frac{u_j}{(x_i-x_j)\sin(\theta_j)+(y_j-y_i)\cos(\theta_j)} \end{bmatrix} \tag{18}$$

$$g_2 \in \begin{bmatrix} \frac{\sin(\theta_j)((x_i-x_j)\cos(\theta_i)+y_i\sin(\theta_i))+\cos(\theta_i)\cos(\theta_j)y_j}{(x_j-x_i)\sin(\theta_j)+(y_i-y_j)\cos(\theta_j)} \\ \frac{-\cos(\theta_i)\cos(\theta_j)x_i-\sin(\theta_i)(x_j\sin(\theta_j)-(y_i-y_j)\cos(\theta_j))}{(x_j-x_i)\sin(\theta_j)+(y_i-y_j)\cos(\theta_j)} \\ \frac{-\cos(\theta_i-\theta_j)}{(x_i-x_j)\sin(\theta_j)+(y_j-y_i)\cos(\theta_j)} \end{bmatrix} \tag{19}$$

In these equations x_i, y_i, and θ_i refer to the planar coordinates and orientation of the i^{th} actuator. The set-valued notation of (18) and (19) refers to the fact that at a transition between actuators i and j being the two closest actuators to actuators k and l being the closest the kinematics are discontinuous. Therefore, at these points we must allow multi-valued differentials in order to guarantee existence of solutions to the differential equation in (17). See [34] for more details. It should be noted that here the index notation should be thought of as mapping (i, j) pairs to equations of motion in some neighborhood (not necessarily small) around the i^{th} and j^{th} actuator. In each region $I - VIII$ the kinematics are smooth, but when a trajectory crosses a boundary $\mathbf{0\text{-}\frac{7\pi}{4}}$, there is a discontinuity in the kinematics. It is possible to obtain point stabilization to $(x, y, \theta) = (0, 0, 0)$ from any initial condition using discontinuous control laws based on the kinematics and knowing the current model (see [34] for details of this control design). However, there are many questions relevant to this system which remain unanswered.

Problem 10. Is the system in (17) locally controllable near the origin, or anywhere else?

Problem 11. We know that point stabilization to the origin can be achieved using discontinuous control laws, but does there exist a continuous control law which will do so? More generally how do we test to see if a given nonsmooth system (such as Definition 6) has a smooth feedback law for the purposes of point stabilization?

Problem 12. How can we characterize the set of points in $\mathbb{R}^2$ to which the system in (17) can be stabilized?

Problem 13. What kind of trajectories can this system follow? Most importantly, if we want to move a part to an arbitrary point in $SE(2)$, how close can we get it to the desired point?

With the advent of more and more actuators being built cheaply at very small scales, massively overactuated systems will become prevalent in industry. It is therefore important to develop systematic tools for the analysis and design of control laws for these systems. Sometimes, by necessity, this

will involve complicated control laws based on the full nonsmooth system. In such a case, we will need the tools to answer questions such as those posed in Problems 10-13 before we will be able to move parts efficiently and reliably at the MEMS scale, or in any other distributed manipulation system with limited actuator degrees of freedom.

References

1. S. Akella, W. Huang, K. M. Lynch, and M. T. Mason. Parts feeding on a conveyor with a one joint robot. *Algorithmica*, 26(3):313–344, March-April 2000.
2. J. C. Alexander and J. H. Maddocks. On the kinematics of wheeled vehicles. *The International Journal of Robotics Research*, 8(5):15–27, October 1989.
3. K. F. Böhringer and H. Choset, editors. *Distributed Manipulation*. Kluwer, 2000.
4. K. F. Böhringer, B. R. Donald, L. E. Kavraki, and F. Lamiraux. A distributed, universal device for planar parts feeding: unique part orientation in programmable force fields. In *Distributed Manipulation*, pages 1–28. Kluwer, 2000.
5. R. W. Brockett and L. Dai. Nonholonomic kinematics and the role of elliptic functions in constructive controllability. In Z. Li and J. Canny, editors, *Nonholonomic Motion Planning*. Kluwer Academic, 1993.
6. B. Brogliato and A. Zavala-Rio. On the control of complementary-slackness juggling mechanical systems. *IEEE Transactions on Automatic Control*, 45(2):235–246, February 2000.
7. M. Bühler and D. E. Koditschek. From stable to chaotic juggling: Theory, simulation, and experiments. In *IEEE International Conference on Robotics and Automation*, pages 1976–1981, Cincinnati, OH, 1990.
8. M. Bühler, D. E. Koditschek, and P. J. Kindlmann. Planning and control of a juggling robot. *International Journal of Robotics Research*, 13(2):101–118, 1994.
9. F. Bullo, A. D. Lewis, and K. M. Lynch. Controllable kinematic reductions for mechanical systems: Concepts, computational tools, and examples. In *2002 International Symposium on the Mathematical Theory of Networks and Systems*, August 2002.
10. F. Bullo and K. M. Lynch. Kinematic controllability for decoupled trajectory planning of underactuated mechanical systems. *IEEE Transactions on Robotics and Automation*, 17(4):402–412, August 2001.
11. R. R. Burridge, A. A. Rizzi, and D. E. Koditschek. Toward a dynamical pick and place. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2: 292–297, 1995.
12. P. Choudhury and K. M. Lynch. Rolling manipulation with a single control. *International Journal of Robotics Research*, 2002. To appear.
13. B. R. Donald, J. Jennings, and D. Rus. *Algorithmic Foundations of Robotics (WAFR)*, chapter Information invariants for distributed manipulation, pages 431–459. A.K. Peters, Ltd, Wellesley, MA, 1995.
14. M. A. Erdmann. Understanding action and sensing by designing action-based sensors. *International Journal of Robotics Research*, 14(5):483–509, October 1995.

15. B. Goodwine and J. W. Burdick. Controllability of kinematic control systems on stratified configuration spaces. *IEEE Trans. on Automatic Control*, 46(3):358–368, 2000.
16. G. W. Haynes and H. Hermes. Nonlinear controllability via Lie theory. *SIAM Journal on Control*, 8(4):450–460, November 1970.
17. Y. Jia and M. Erdmann. Observing pose and motion through contact. In *IEEE International Conference on Robotics and Automation*, pages 723–729, 1998.
18. V. Jurdjevic. *Geometric Control Theory*. Cambridge University Press, 1997.
19. A. D. Lewis. When is a mechanical control system kinematic? In *IEEE Conference on Decision and Control*, pages 1162–1167, December 1999.
20. A. D. Lewis and R. M. Murray. Configuration controllability of simple mechanical control systems. *SIAM Journal on Control and Optimization*, 35(3):766–790, May 1997.
21. K. Lian, L. Wang, and L. Fu. Controllability of spacecraft systems in a central gravitational field. *IEEE Transactions on Automatic Control*, 39(12):2426–2440, December 1994.
22. J. Luntz and W. Messner. Closed-loop stability of distributed manipulation. In *Proc. American Control Conference (ACC)*, 2000.
23. J. Luntz, W. Messner, and H Choset. Velocity field design for parcel manipulation on the modular distributed manipulation system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1999.
24. J. Luntz, W. Messner, and H. Choset. Closed-loop distributed manipulation using discrete actuator arrays. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2000.
25. K. M. Lynch. Controllability of a planar body with unilateral thrusters. *IEEE Transactions on Automatic Control*, 44(6):1206–1211, June 1999.
26. K. M. Lynch. Locally controllable manipulation by stable pushing. *IEEE Transactions on Robotics and Automation*, 15(2):318–327, April 1999.
27. K. M. Lynch and C. K. Black. Recurrence, controllability, and stabilization of juggling. *IEEE Transactions on Robotics and Automation*, 17(2):113–124, April 2001.
28. K. M. Lynch and M. T. Mason. Stable pushing: Mechanics, controllability, and planning. *International Journal of Robotics Research*, 15(6):533–556, December 1996.
29. K. M. Lynch and M. T. Mason. Dynamic nonprehensile manipulation: Controllability, planning, and experiments. *International Journal of Robotics Research*, 18(1):64–92, January 1999.
30. K. M. Lynch, N. Shiroma, H. Arai, and K. Tanie. The roles of shape an motion in dynamic manipulation: The butterfly example. In *IEEE International Conference on Robotics and Automation*, pages 927–932, 1998.
31. A. Marigo and A. Bicchi. Rolling bodies with regular surface: Controllability theory and applications. *IEEE Transactions on Automatic Control*, 45(9):1586–1599, September 2000.
32. A. Marigo, M. Ceccarelli, S. Piccinocchi, and A. Bicchi. Planning motions of polyhedral parts by rolling. *Algorithmica*, 26:560–576, 2000.
33. M. T. Mason. *Mechanics of Robotic Manipulation*. MIT Press, 2001.
34. T. D. Murphey. *Control of Multiple Model Systems*. PhD thesis, California Institute of Technology, May 2002.

35. T. D. Murphey and J. W. Burdick. Issues in controllability and motion planning for overconstrained wheeled vehicles. In *Proc. Int. Conf. Math. Theory of Networks and Systems (MTNS)*, Perpignan, France, 2000.
36. T. D. Murphey and J. W. Burdick. On the stability and design of distributed systems. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Seoul, Korea, 2001.
37. T. D. Murphey and J. W. Burdick. Global exponential stabilizability for distributed manipulation. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Washington D.C., 2002.
38. F. Pfeiffer and C. Glocker. *Multibody Dynamics with Unilateral Controls*. Wiley, Chichester, 1996.
39. A. A. Rizzi and D. E. Koditschek. Progress in spatial robot juggling. In *IEEE International Conference on Robotics and Automation*, pages 775–780, Nice, France, 1992.
40. A. A. Rizzi and D. E. Koditschek. Further progress in robot juggling: The spatial two-juggle. In *IEEE International Conference on Robotics and Automation*, pages 3:919–924, Atlanta, GA, 1993.
41. M. W. Spong. Impact controllability of an air hockey puck. *Systems and Control Letters*, 42:333–345, 2001.
42. A. Sudsang and L. Kavraki. A geometric approach to designing a programmable force field with a unique stable equilibrium for parts in the plane. In *Proc. IEEE Int. Conf. Robotics and Automation*, Seoul, Korea, 2001.
43. M. Zefran and J. W. Burdick. Stabilization of systems with changing dynamics. In *Workshop on Hybrid systems: Computation and control*, Berkeley, CA, 1998.

Motion Planning and Control Problems for Underactuated Robots

Sonia Martínez[1], Jorge Cortés[2], and Francesco Bullo[2]

[1] Escuela Universitaria Politécnica
Universidad Politécnica de Cataluña
Av. V. Balaguer s/n
Vilanova i la Geltrú 08800, Spain
[2] Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
1308 W. Main St
Urbana, IL 61801, USA

Abstract. Motion planning and control are key problems in a collection of robotic applications including the design of autonomous agile vehicles and of minimalist manipulators. These problems can be accurately formalized within the language of affine connections and of geometric control theory. In this paper we overview recent results on kinematic controllability and on oscillatory controls. Furthermore, we discuss theoretical and practical open problems as well as we suggest control theoretical approaches to them.

1 Motivating Problems from a Variety of Robotic Applications

The research in Robotics is continuously exploring the design of novel, more reliable and agile systems that can provide more efficient tools in current applications such as factory automation systems, material handling, and autonomous robotic applications, and can make possible their progressive use in areas such as medical and social assistance applications.

Mobile Robotics, primarily motivated by the development of tasks in unreachable environments, is giving way to new generations of autonomous robots in its search for new and "better adapted" systems of locomotion. For example, traditional wheeled platforms have evolved into articulated devices endowed with various types of wheels and suspension systems that maximize their traction and the robot's ability to move over rough terrain or even climb obstacles. The types of wheels that are being employed include passive and powered castors, ball-wheels or omni-directional wheels that allow a high accuracy in positioning and yet retain the versatility, flexibility and other properties of wheels. A rich and active literature includes (i) various vehicle designs [38,41,44,46], (ii) the automated guided vehicle "OmniMate" [2], (iii) the roller-walker [15] and other dexterous systems [17] that change their internal shape and constraints in response to the required motion sequence, and (iv) the omni-directional platform in [19].

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 59–74, 2003

Other types of remotely controlled autonomous vehicles that are increasingly being employed in space, air and underwater applications include submersibles, blimps, helicopters, and other crafts. More often than not they rely on innovative ideas to affect their motion instead of on classic design ideas. For example, in underwater vehicle applications, innovative propulsion systems such as shape changes, internal masses, and momentum wheels are being investigated. Fault tolerance, agility, and maneuverability in low velocity regimes, as in the previous example systems, are some of the desired capabilities.

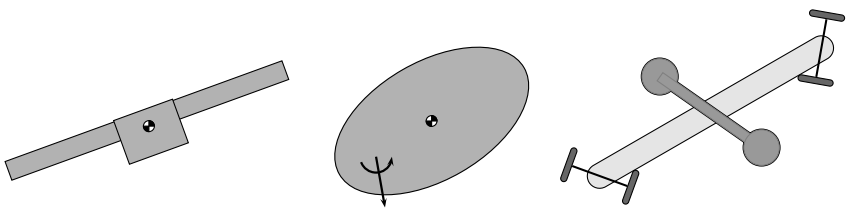


Fig. 1. Underactuated robots appear in a variety of environments. From left to right, a planar vertical take-off and landing (PVTOL) aircraft model, a horizontal model of a blimp and the snakeboard.

A growing field in Mobile Robotics is that of *biomimetics*. The idea of this approach is to obtain some of the robustness and adaptability that biological systems have refined through evolution. In particular, biomimetic locomotion studies the periodic movement patterns or *gaits* that biological systems undergo during locomotion and then takes them as reference for the design of the mechanical counterpart. In other cases, the design of robots without physical counterpart is inspired by similar principles. Robotic locomotion systems include the classic *bipeds and multi-legged robots* as well as swimming snake-like robots and flying robots. These systems find potential applications in harsh or hazardous environments, such as under deep or shallow water, on rough terrain (with stairs), along vertical walls or pipes and other environments difficult to access for wheeled robots. Specific examples in the literature include hyper-redundant robots [13,16], the snakeboard [32,40], the G-snakes and roller racer models in [26,27], fish robots [23,25], eel robots [21,36], and passive and hopping robots [18,35,42].

All this set of emerging robotic applications have special characteristics that pose new challenges in motion planning. Among them, we highlight:

Underactuation. This could be owned to a design choice: nowadays low weight and fewer actuators must perform the task of former more expensive systems. For example, consider a manufacturing environment where robotic devices perform material handling and manipulation tasks: automatic planning algorithms might be able to cope with failures without interrupting the manufacturing process. Another reason why these systems are underactuated is

because of an unavoidable limited control authority: in some locomotion systems it is not possible to actuate all the directions of motion. For example, consider a robot operating in a hazardous or remote environment (e.g., aerospace or underwater), an important concern is its ability to operate faced with a component failure, since retrieval or repair is not always possible.

Complex dynamics. In these control systems, the drift plays a key role. Dynamic effects must necessarily be taken into account, since kinematic models are no longer available in a wide range of current applications. Examples include lift and drag effects in underwater vehicles, the generation of momentum by means of the coupling of internal shape changes with the environment in the eel robot and the snakeboard, the dynamic stability properties of walking machines and nonholonomic wheeled platforms, etc.

Current limitations of motion algorithms. Most of the work on motion planning has relied on assumptions that are no longer valid in the present applications. For example, one of these is that (wheeled) robots are kinematic systems and, therefore, controlled by velocity inputs. This type of models allows one to design a control to reach a desired point and then immediately stop by setting the inputs to zero. This is obviously not the case when dealing with complex dynamic models.

Another common assumption is the one of fully actuation that allows to decouple the motion planning problem into path planning (computational geometry) and then tracking. For underactuated systems, this may be not possible because we may be obtaining motions in the path planning stage that the system can not perform in the tracking step because of its dynamic limitations.

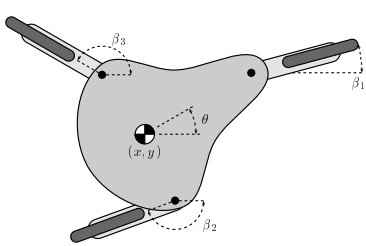


Fig. 2. Vertical view of an omni-directional robotic platform with 6 degrees of freedom and 3 nonholonomic constraints [12,19]. This device is capable of highly accurate positioning, high payloads, and high speed motion. In its fully actuated configuration, the robot is endowed with 6 motors at the three wheels and at the three joints $(\beta_1, \beta_2, \beta_3)$. However, underactuated configurations can arise because of failures or intentional design.

Furthermore, motion planning and optimization problems for these systems are nonlinear, non-convex problems with exponential complexity in the dimension of the model. These issues have become increasingly important due to the high dimensionality of many current mechanical systems, including flexible structures, compliant manipulators and multibody systems undergoing reconfiguration in space.

Benefits that would result from better motion planning algorithms for underactuated systems. From a practical perspective, there are at least two advantages to designing controllers for underactuated robotic manipulators and vehicles. First, a fully actuated system requires more control inputs than an underactuated system, which means there will have to be more devices to generate the necessary forces. The additional controlling devices add to the cost and weight of the system. Finding a way to control an underactuated version of the system would improve the overall performance or reduce the cost. The second practical reason for studying underactuated vehicles is that underactuation provides a backup control technique for a fully actuated system. If a fully actuated system is damaged and a controller for an underactuated system is available, then we may be able to recover gracefully from the failure. The underactuated controller may be able to salvage a system that would otherwise be uncontrollable.

2 Mathematical Unifying Approach to the Modeling of Robotic Systems

Most of the robotic devices we have mentioned so far can be characterized by their special Lagrangian structure. They usually exhibit symmetries and their motion is constrained by the environment where they operate. In the following, we introduce a general modeling language for underactuated robotic systems.

Let $q = (q^1, \ldots, q^n) \in Q$ be the configuration of the mechanical system and consider the control equations:

$$\ddot{q}^i + \Gamma^i_{jk}(q)\dot{q}^j\dot{q}^k = -M^{ij}\frac{\partial V}{\partial q^j} + k^i_j(q)\dot{q}^j + Y^i_1(q)u_1 + \ldots + Y^i_m(q)u_m\,, \quad (1)$$

where the summation convention is in place for the indices j, k that run from 1 to n, and

(i) $V : Q \to \mathbb{R}$ corresponds to potential energy, and $k^i_j(q)\dot{q}^j$ corresponds to damping forces,

(ii) $\{\Gamma^i_{jk} : i, j, k = 1, \ldots, n\}$ are n^3 Christoffel symbols, derived from $M(q)$, the inertia matrix defining the kinetic energy, according to

$$\Gamma^k_{ij} = \frac{1}{2}M^{mk}\left(\frac{\partial M_{mj}}{\partial q^i} + \frac{\partial M_{mi}}{\partial q^j} - \frac{\partial M_{ij}}{\partial q^m}\right),$$

where M^{mk} is the (m, k) component of M^{-1}, and,

(iii) $\{F_a : a = 1, \ldots, m\}$ are the m input co-vector fields, and $\{Y_a = M^{-1}F_a : a = 1, \ldots, m\}$ are the m input vector fields.

Underactuated systems have fewer control actuators, m, than degrees of freedom $n > m$. Other limitations on the control signals u_a might be present,

e.g., actuators might have magnitude and rate limits, or they might only generate unilateral or binary signals (e.g., thrusters in satellites).

The notion of affine connection provides a coordinate-free means of describing the dynamics of robotic systems. Given two vector fields X, Y, the *covariant derivative* of Y with respect to X is the third vector field $\nabla_X Y$ defined via

$$(\nabla_X Y)^i = \frac{\partial Y^i}{\partial q^j} X^j + \Gamma^i_{jk} X^j Y^k. \tag{2}$$

The operator ∇ is called the *affine connection* for the mechanical system in equation (1). We write the Euler-Lagrange equations for a system subject to a conservative force Y_0, a damping force $k(q)\dot{q}$ and m input forces as:

$$\nabla_{\dot{q}} \dot{q} = Y_0(q) + k(q)(\dot{q}) + \sum_{a=1}^{m} Y_a(q) u_a(t). \tag{3}$$

Equation (3) is a coordinate-free version of equation (1). A crucial observation is the fact that systems subject to nonholonomic constraints can also be modeled by means of affine connections. In the interest of brevity, we refer to [10,30] for the exposition of this result and the explicit expression of the Christoffel symbols corresponding to the Lagrange-d'Alembert equations.

The homogeneous structure of mechanical systems. The fundamental structure of the control system in equation (3) is the polynomial dependence of the various vector fields on the velocity variable $\dot{q}$. This structure affects the Lie bracket computations involving input and drift vector fields. The system (3) is written in first order differential equation form as

$$\frac{d}{dt}\begin{bmatrix} q \\ \dot{q} \end{bmatrix} = \begin{bmatrix} \dot{q} \\ -\Gamma(q,\dot{q}) + Y_0(q) + k(q)(\dot{q}) \end{bmatrix} + \sum_{a=1}^{m} \begin{bmatrix} 0 \\ Y_a \end{bmatrix} u_a(t)$$

where $\Gamma(q,\dot{q})$ is the vector with ith component $\Gamma^i_{jk}(q)\dot{q}^j\dot{q}^k$. Also, if $x = (q,\dot{q})$,

$$Z(x) = \begin{bmatrix} \dot{q} \\ -\Gamma(q,\dot{q}) \end{bmatrix}, \quad Y_a^{\text{lift}}(x) \triangleq \begin{bmatrix} 0 \\ Y_a(q) \end{bmatrix}, \quad \text{and} \quad k^{\text{lift}}(x) \triangleq \begin{bmatrix} 0 \\ k(q)(\dot{q}) \end{bmatrix},$$

the control system is rewritten as

$$\dot{x} = Z(x) + Y_0^{\text{lift}}(x) + k^{\text{lift}}(x) + \sum_{a=1}^{m} Y_a^{\text{lift}}(x) u_a(t)\,.$$

Let $h_i(q,\dot{q})$ be the set of scalar functions on $\mathbb{R}^{2n}$ which are arbitrary functions of q and homogeneous polynomials in $\{\dot{q}^1, \ldots, \dot{q}^n\}$ of degree i. Let $\mathcal{P}_i$ be the set of vector fields on $\mathbb{R}^{2n}$ whose first n components belong to h_i

and whose second n components belong to h_{i+1}. We note that these notions can also be defined on a general manifold, see [7].

We are now ready to present two simple ideas. First, all the previous vector fields are homogeneous polynomial vector fields for some specific value of i. Indeed, $Z \in \mathcal{P}_1$, $k^{\text{lift}} \in \mathcal{P}_0$, and $Y_a^{\text{lift}} \in \mathcal{P}_{-1}$. Second, since the Lie bracket between a vector field in $\mathcal{P}_i$ and a vector field in $\mathcal{P}_j$ belongs to $\mathcal{P}_{i+j}$, any Lie bracket of the given relevant vector fields remains a homogeneous polynomial. In other words, the set of homogeneous vector fields is closed under the operation of Lie bracket.

A consequence of this analysis is the definition of symmetric product of vector fields. We define the *symmetric product* between Y_b and Y_a as the vector field $\langle Y_a : Y_b \rangle = \langle Y_b : Y_a \rangle$ given by

$$\langle Y_b : Y_a \rangle^i = \langle Y_a : Y_b \rangle^i = \frac{\partial Y_a^i}{\partial q^j} Y_b^j + \frac{\partial Y_b^i}{\partial q^j} Y_a^j + \Gamma_{jk}^i \left(Y_a^j Y_b^k + Y_a^k Y_b^j \right).$$

Straightforward computations show that $\langle Y_a : Y_b \rangle^{\text{lift}} = [Y_b^{\text{lift}}, [Z_g, Y_a^{\text{lift}}]]$. This operation plays a key role in nearly all the control problems associated with this class of systems: nonlinear controllability [14,31], optimal control [11,29], dynamic feedback linearization [43], algorithms for motion planning and stabilization [6,33,39], etc.

A series expansion for the forced evolution starting from rest. The homogeneous structure of the mechanical control system (3), together with the symmetric product, set the basis to establish the following description of the evolution of the system trajectories starting with zero initial velocity [3,14]. Assume no potential or damping forces are present in the system. Let $Y(q,t) = \sum_{a=1}^m Y_a(q) u_a(t)$. Define recursively the vector fields V_k by

$$V_1(q,t) = \int_0^t Y(q,s) ds\,, \quad V_k(q,t) = -\frac{1}{2} \sum_{j=1}^{k-1} \int_0^t \Big\langle V_j(q,s) :\ V_{k-j}(q,s) \Big\rangle ds.$$

Then, the solution $q(t)$ of equation (3) satisfies

$$\dot{q}(t) = \sum_{k=1}^{+\infty} V_k(q(t),t), \tag{4}$$

where the series converges absolutely and uniformly in a neighborhood of q_0 and over a fixed time interval $t \in [0,T]$. This series expansion provides a means of describing the open-loop response of the system to any specific forcing. As we will see below, it plays a key role in several motion planning and control strategies for underactuated robots.

3 Existing Results on Planning for Underactuated Systems

To design planning algorithms for underactuated robotic systems, we advocate an integrated approach based on modeling, system design, controllability analysis, dexterity, manipulability, and singularities. These analysis concepts are fundamental for robust planning algorithms that do not solely rely on randomization or nonlinear programming. We do not suggest closed-form planning algorithms, rather we envision methods that combine the best features of formal analysis and of numerical algorithms.

For reasons of space, we cannot present a detailed account of all existing results on motion planning for underactuated systems, and not even of the results obtained within the modeling approach proposed in Section 2. Therefore, we focus on two specific control methodologies for motion planning: decoupled planning algorithms for kinematically controllable systems, and approximate inversion algorithms based on oscillatory controls.

Section 3.1 reviews *decoupled planning algorithms* that exploit certain differential geometric properties to reduce the complexity of the motion planning problem (still to be solved via numerical algorithms). The notion of *kinematic controllability* is extremely effective: trajectory planning decouples from being a problem on a $2n$ dimensional space to an n dimensional space. Furthermore, various state constraints can be neglected in the reduced space. For systems that are not kinematically controllable and that require oscillatory controls to locomote, Section 3.2 presents motion planning algorithms based on *approximate inversion*. Both design methods are closely related to recent results on nonlinear controllability [9,31], power series expansions [3,14], two time-scales coordinate-free averaging [4,34], and nonlinear inversion algorithms [6,33].

The strengths of this methodology are as follows. Both methodologies provide solutions to the corresponding problems, i.e., point to point and trajectory planning. These analytic results do not rely on non-generic assumptions such as feedback linearization, nilpotency or flatness. The results are coordinate-free and hence widely applicable, e.g., to aerospace or underwater robotics settings. Both methods are *consistent*, *complete* and constructive (consistent planners recover the known solutions available for linear and nilpotent systems, and complete planners are guaranteed to find a local solution for any nonlinearly controllable system).

3.1 Kinematic controllability for underactuated robots

The following decoupling methodology was proposed in [9] to reduce the complexity of the motion planning problem. The method is constructive (only quadratic equations and no PDEs are involved) and physically intuitive.

We consider as a motivating example a common pick-&-place manipulator: Fig. 3 shows a vertical view of a three-revolute-joints device. We investigate planning schemes for this system when one of its three motors is either

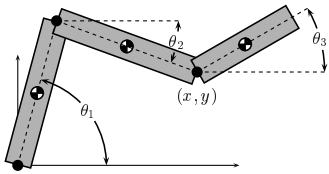


Fig. 3. A three-revolute-joints device. It can be proven [9] that any two-actuator configuration of this system is kinematically controllable, i.e., one can always find two decoupling vector fields whose involutive closure is full-rank.

failed or missing. We present a decoupling idea to reduce the complexity of the problem: instead of searching for feasible trajectories of a dynamic system in $\mathbb{R}^6$, we show how it suffices to search for paths of a simpler, kinematic (i.e., driftless) system in $\mathbb{R}^3$.

A curve $\gamma : [0, T] \mapsto Q$ is a controlled solution to equation (1) if there exist inputs $u_a : [0, T] \to \mathbb{R}$ for which γ solves (1). To avoid the difficult task of characterizing all controlled solutions of the system (1), we focus on curves satisfying $\dot{\gamma} = \dot{s}(t)X(\gamma)$, where X is a vector field on Q, and where the map $s : [0, T] \to [0, 1]$ is a "time-scaling" parameterization of γ. Such curves are called *kinematic motions*.

We call V a *decoupling vector field* if all curves γ satisfying $\dot{\gamma} = \dot{s}(t)V(\gamma)$ for any time scaling s, are kinematic motions. This definition is useful for three reasons. First, V is decoupling if and only if V and $\nabla_V V$ are linear combinations in $\{Y_1, \ldots, Y_m\}$. Second, decoupling vector fields can be computed by solving $(n - m)$ quadratic equations. Third, if enough decoupling vector fields, say $V_1, \ldots, V_p$, are available to satisfy the LARC, we call the system *kinematically controllable*. In the latter case, we can plan motions for the kinematic system $\dot{q} = \sum_{a=1}^{p} w_a(t)V_a(q)$, and they will automatically be controlled curves for the original system (1).

3.2 Approximate inversion via small amplitude and oscillatory controls

As in the previous section, the objective is to design motion planning and stabilization schemes for underactuated systems. We propose perturbation and inversion methods as widely applicable approaches to solve point to point and trajectory planning problems. Let us regard the flow map Φ of equation (3) over a finite time interval as a map from the input functions $u_i : [0, T] \to \mathbb{R}$ to the target state $x(T)$. The ideal algorithm for point-to-point planning computes an exact (right) inverse Φ^{-1} of Φ. Unfortunately, closed form expressions for Φ^{-1} are available only assuming *non-generic* differential geometric conditions (e.g., the system needs to be feedback linearizable, differentially flat, or nilpotent). Instead of aiming at "exact" solutions, we focus on computing an *approximate inverse map* using perturbation methods such as power series expansions and averaging theory. Although these tools are only approximate, the resulting algorithms are consistent and complete.

Oscillatory (high frequency, high amplitude) controls for trajectory planning. We present the approach in three steps and refer to [34] for all the details. As first step, we present a recent coordinate-free averaging result. Let $0 < \epsilon \ll 1$. Assume the control inputs are of the form

$$u_i = \frac{1}{\epsilon} u_i \left(\frac{t}{\epsilon}, t \right),$$

and assume they are T-periodic and zero-mean in the first variable. Define the averaged multinomial iterated integrals of $u_1, \ldots, u_m$ as

$$\overline{\mathcal{U}}_{k_1,\ldots,k_m}(t) = \frac{T^{-1}}{k_1! \ldots k_m!} \int_0^T \left(\int_0^s u_1(\tau, t) d\tau \right)^{k_1} \ldots \left(\int_0^s u_m(\tau, t) d\tau \right)^{k_m} ds\,.$$

Let a, b, c take value in $\{1, \ldots, m\}$. Let $\boldsymbol{k}_a$ (resp. $\boldsymbol{k}_{ab}$) denote the tuple $(k_1, \ldots, k_m)$ with $k_c = \delta_{ca}$ (resp. $k_c = \delta_{ca} + \delta_{cb}$). Then, over a finite time $q(t) = r(t) + O(\epsilon)$, as $\epsilon \to 0$, where $r(t)$ satisfies

$$\begin{aligned} \nabla_{\dot r} \ddot r = Y_0(r) + k(r)(\dot r) + \sum_{a=1}^m \left(\frac{1}{2} \overline{\mathcal{U}}^2_{\boldsymbol{k}_a}(t) - \overline{\mathcal{U}}_{\boldsymbol{k}_{aa}}(t) \right) \langle Y_a : Y_a \rangle(r) \\ + \sum_{a<b} \left(\overline{\mathcal{U}}_{\boldsymbol{k}_a}(t) \overline{\mathcal{U}}_{\boldsymbol{k}_b}(t) - \overline{\mathcal{U}}_{\boldsymbol{k}_{ab}}(t) \right) \langle Y_a : Y_b \rangle(r)\,. \end{aligned} \tag{5}$$

As a second step, given $z_a(t)$, $z_{bc}(t)$ arbitrary functions of time, we propose the following inversion procedure

(i) define the scalar functions $\psi_{N(a,b)}(t) = \sqrt{2}\, N(a,b) \cos(N(a,b)\, t)$, where $(a,b) \mapsto N(a,b) \in \{1, \ldots, N\}$ is an enumeration of the pairs of integers (a,b), with $a < b$.

(ii) select the following controls in (3),

$$\begin{aligned} u_a(t,q) &= v_a(t,q) + \frac{1}{\epsilon} w_a \left(\frac{t}{\epsilon}, t \right), \\ w_a(\tau, t) &= -\sum_{c=1}^{a-1} \psi_{N(c,a)}(\tau) + \sum_{c=a+1}^{m} z_{ac}(t) \psi_{N(a,c)}(\tau)\,, \end{aligned}$$

where $v_a(t,q)$ are still to be chosen.

After computing the averaged iterated integrals of the oscillatory inputs $w_a(t/\epsilon, t)$, equation (5) for the averaged system becomes

$$\begin{aligned} \nabla_{\dot r} \ddot r = Y_0(r) + k(r)(\dot r) + \sum_{a=1}^m v_a(t,r) Y_a(r) \\ - \sum_{a=1}^m \overline{\mathcal{U}}_{\boldsymbol{k}_{aa}}(t) \langle Y_a : Y_a \rangle(r) + \sum_{a<b} z_{ab}(t) \langle Y_a : Y_b \rangle(r)\,. \end{aligned}$$

As a third and final step, assume that all the vector fields of the form $\langle Y_b : Y_b \rangle$ belong to $\text{span}\{Y_a\}$. Let $\alpha_{ab} : Q \to \mathbb{R}$ be such that $\langle Y_a : Y_a \rangle(q) = \sum_b \alpha_{ab}(q) Y_b(q)$, $q \in Q$. Select

$$v_a(t,q) = z_a(t) + \frac{1}{2} \sum_{b=1}^{m} \alpha_{ba}(q) \left(b - 1 + \sum_{c=b+1}^{m} (z_{bc}(t))^2 \right) .$$

Then, we have

$$\sum_{a=1}^{m} v_a(t,r) Y_a(r) = \sum_{a=1}^{m} z_a^d(t) Y_a(r) + \sum_{a=1}^{m} \overline{\mathcal{U}}_{\boldsymbol{k}_{aa}}(t) \langle Y_a : Y_a \rangle(r) ,$$

which implies that eq. (5) takes the final form,

$$\nabla_{\dot{r}} \dot{r} = Y_0(r) + k(r)(\dot{r}) + \sum_{a=1}^{m} z_a(t) Y_a(r) + \sum_{b<c} z_{bc}(t) \langle Y_b : Y_c \rangle(r) ,$$

The averaged system now has more available control inputs than the original one. If the input distribution $\mathcal{I} = \text{span}\{Y_a , \langle Y_b : Y_c \rangle\}$ is full rank, then the latter system is fully actuated (i.e., one control input is available for each degree of freedom). If the input distribution $\mathcal{I}$ contains a sufficient number of decoupling vector fields, then the system is kinematically controllable. In both cases, we have reduced the complexity of the motion planning problem.

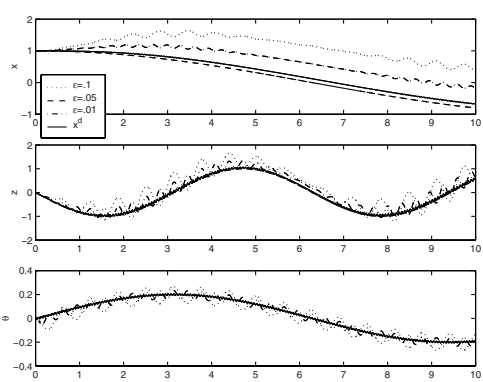


Fig. 4. Approximate trajectory tracking for an underactuated PVTOL model by means of oscillatory controls. The curve to be tracked is shown solid, and the various oscillating curves correspond to different values of the parameter ϵ

Remark 1 (Small amplitude algorithms based on series expansions). A related approach to motion planning relies on small amplitude periodic forcing; see [6,33]. The planning problem is solved by approximately inverting the series expansion describing the evolution of the control system (cf. Section 2). This inversion procedure is very similar to the one presented above. Based on it, one can establish two simple primitives of motion to change and

maintain velocity, while keeping track of the changes in the configuration. These primitives can then be used as the building blocks to design high-level motion algorithms that solve the point-to-point reconfiguration problem, the static interpolation problem and the local exponential stabilization problem. Fig. 5 shows two examples of the execution of these algorithms.

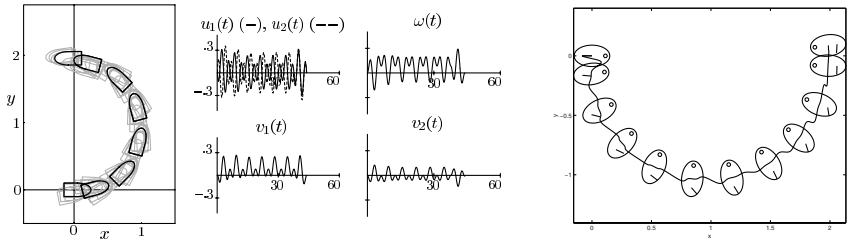


Fig. 5. Illustration of the motion planning algorithms via small amplitude periodic forcing for a simple planar body (left) and the blimp model (right). The errors in the final configuration are within the same order of magnitude of the input employed.

4 Open Problems and Possible Approaches

Immediate open questions arising from the above-presented results are the following:

Kinematic modeling and control. The current limitations are as follows: the design problem is now reduced to planning for a kinematic system with the additional constraint of zero-velocity transitions between feasible motions. This additional constraint leads to poor performance when coupled with current randomized planners [20,24,28] that switch frequently between the available motions. The zero-velocity switches also create problems for trajectory tracking controllers based on linearization, since the system loses linear controllability at zero-velocity. Finally, there is no notion of time-optimality for these kinematic motions and there is no way of dealing with systems where oscillatory inputs are needed for locomotion (see below for a discussion on this point). Motivated by this analysis, we identify the following open issues:

(i) Develop a catalog of kinematically controllable systems, including planar manipulators with revolute as well as prismatic joints, parallel manipulator, manipulators in three dimensional space and in aerospace and underwater environments (accounting for the different dynamics in such settings). Some preliminary work in this direction can be found in [8]. Analyze and classify the singularities that these vector fields possess as a prerequisite step for planning purposes.

(ii) A (left) group action is a map $\psi : G \times Q \to Q$ such that $\psi(e, q) = q$, for all $q \in Q$, where e denotes the identity element in G, and $\psi(g, \psi(h, q)) = \psi(gh, q)$, for all $g, h \in G$, $q \in Q$. Usually $G \subset SE(n)$, and then the action describes a rigid displacement of some components of the robot. An interesting problem would be to identify conditions under which decoupling vector fields can be found which are invariant under such group actions. When this is the case, motion plans can be designed exploiting established "inverse kinematics" methods; see [37, Chapter 3]. This simplification eliminates the need for any numerical procedure if the robot moves in an un-obstructed environment, or further reduces the dimensionality and complexity of the resulting search problem in complex environments.

(iii) To tackle the difficulties inherent with zero-velocity transitions, it would be appropriate to develop randomized planners which require as few switches between decoupling vector fields as possible, and to develop trajectory tracking controllers for these systems able to adequately perform through the singularities.

(iv) Another interesting idea would consist of switching between decoupling vector fields without stopping. In some sense, this is also related to the problem of developing transitions between relative equilibria. *Relative equilibria* are "steady trajectories" that the system admits as feasible solutions. This family of trajectories is of great interest in theory and applications as they provide a rich family of motions with the simplifying property of having constant body-fixed velocity. Relative equilibria for systems in three dimensional Euclidean space include straight lines, circles, and helices. Despite partial results, no method is currently available to design provably stable switching maneuvers from one relative equilibrium to another (or from one decoupling vector field to another without stopping). A necessary preliminary step toward this objective is to analyze the controllability properties of underactuated systems moving along a relative equilibrium or along a decoupling vector field.

Small-amplitude and high-frequency controls. The current limitations are as follows. The implementation of the small amplitude approach requires the computation and manipulation of high order tensors, and the approach has a limited region of convergence. The implementation of the oscillatory control approach presents difficulties in most physical settings because of the required high frequency, high amplitude inputs. Motivated by this analysis, we think that the following are interesting issues to explore:

(i) For the small amplitude controls formulation, open questions include (a) investigate tight estimates for the region of validity of the truncations (simulation studies suggest that there are better bounds than the conservative ones currently available), (b) design base functions optimal with regards to region of convergence and appropriate cost criteria, (c) design

inversion algorithms for systems that are not linearly controllable. The latter setting is equivalent to a non-definite quadratic programming problem, i.e., to the problem of finding sufficient conditions for a vector-valued quadratic form to be surjective (see [5] for a discussion on this subject).

(ii) For the oscillatory controls formulation, standing problems are (a) investigate the use of high-frequency bounded amplitude controls, (b) characterize approximate kinematic controllability and differential flatness via oscillations, (c) investigate physical settings in which oscillatory controls are natural control means, e.g., micro-electromechanical robots, (d) investigate extensions of this coordinate-free perturbation theory to discrete-time nonlinear systems, and to distributed parameter systems and partial differential equations.

(iii) An ambitious program would consist of developing schemes that combine the proposed analytic methods with iterative numerical algorithms. One approach is via homotopy and level set methods [1,45] as schemes that overcome the limitations induced by the small parameter (small convergence region or high amplitude high frequency). A second direction is to use the planner based on small amplitude controls as a local planner inside a global search algorithm based on randomization; see [22] for some preliminary results on local/global planners.

Acknowledgments

This research was partially funded by NSF grants CMS-0100162 and IIS-0118146. The authors would like to thank Andrew Lewis, Kevin Lynch, Milos Zefran, Todd Cerven, and Timur Karatas.

References

1. E. L. Allgower and K. Georg. *Numerical Continuation Methods: An Introduction*. Springer Verlag, New York, NY, 1990.
2. J. Borenstein. The OmniMate: a guidewire- and beacon-free AGV for highly reconfigurable applications. *International Journal of Production Research*, 38(9):1993–2010, 2000.
3. F. Bullo. Series expansions for the evolution of mechanical control systems. *SIAM J Control Optim*, 40(1):166–190, 2001.
4. F. Bullo. Averaging and vibrational control of mechanical systems. *SIAM J Control Optim*, 41(2):542–562, 2002.
5. F. Bullo, J. Cortés, A. D. Lewis, and S. Martínez. Vector-valued quadratic forms in control theory. In *Proc MTNS*, Notre Dame, IN, August 2002. Workshop on Open Problems in Systems Theory.
6. F. Bullo, N. Ehrich Leonard, and A. D. Lewis. Controllability and motion algorithms for underactuated Lagrangian systems on Lie groups. *IEEE Trans Automat Control*, 45(8):1437–1454, 2000.

7. F. Bullo and A. D. Lewis. On the homogeneity of the affine connection model for mechanical control systems. In *Proc CDC*, pages 1260–1265, Sydney, Australia, December 2000.
8. F. Bullo, A. D. Lewis, and K. M. Lynch. Controllable kinematic reductions for mechanical systems: concepts, computational tools, and examples. In *Proc MTNS*, Notre Dame, IN, August 2002.
9. F. Bullo and K. M. Lynch. Kinematic controllability for decoupled trajectory planning in underactuated mechanical systems. *IEEE Trans Robotics Automat*, 17(4):402–412, 2001.
10. F. Bullo and M. Zefran. On mechanical control systems with nonholonomic constraints and symmetries. *Syst & Control Lett*, 45(2):133–143, 2002.
11. M. Camariña, F. Silva Leite, and P. E. Crouch. Splines of class C^k on non-Euclidean spaces. *IMA Journal of Mathematical Control & Information*, 12:399–410, 1995.
12. G. Campion, G. Bastin, and B. D'Andrea-Novel. Structural properties and classification of kinematic and dynamic models of wheeled mobile robots. *IEEE Trans Robotics Automat*, 12(1):47–62, 1996.
13. G. Chirikjian and J. W. Burdick. Kinematics of hyperredundant locomotion. *IEEE Trans Robotics Automat*, 11(6):781–793, 1994.
14. J. Cortés, S. Martínez, and F. Bullo. On nonlinear controllability and series expansions for Lagrangian systems with dissipative forces. *IEEE Trans Automat Control*, 47(8):1396–1401, 2002.
15. G. Endo and S. Hirose. Study on roller-walker (system integration and basic experiments). In *Proc ICRA*, pages 2032–7, Detroit, MI, May 1999.
16. S. Hirose. *Biologically inspired robots: snake-like locomotors and manipulators*. Oxford University Press, Oxford, UK, 1993.
17. S. Hirose. Variable constraint mechanism and its application for design of mobile robots. *Int J Robotics Research*, 19(11):1126–1138, 2000.
18. J. K. Hodgins and M. H. Raibert. Biped gymnastics. *Int J Robotics Research*, 9(2):115–132, 1990.
19. R. Holmberg and O. Khatib. Development and control of a holonomic mobile robot for mobile manipulation tasks. *Int J Robotics Research*, 19(11):1066–1074, 2000.
20. D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. *International Journal of Computational Geometry and Applications*, 9(4):495–512, 1999.
21. J. C. Jalbert, S. Kashin, and J. Ayers. Design considerations and experiments of a biologically based undulatory lamprey AUV. In *Ninth International Symposium on Unmanned Untethered Submersible Technology*, Durham, NH, 1995.
22. T. Karatas and F. Bullo. Randomized searches and nonlinear programming in trajectory planning. In *Proc CDC*, pages 5032–5037, Orlando, FL, December 2001.
23. N. Kato and T. Inaba. Guidance and control of fish robot with apparatus of pectoral fin motion. In *Proc ICRA*, pages 446–451, Leuven, Belgium, May 1998.
24. L. E. Kavraki, P. **S**vestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional space. *IEEE Trans Robotics Automat*, 12(4):566–580, 1996.
25. S. D. Kelly, R. J. Mason, C. T. Anhalt, R. M. Murray, and J. W. Burdick. Modelling and experimental investigation of carangiform locomotion for control. In *Proc ACC*, pages 1271–1276, Philadelphia, PA, 1998.

26. P. S. Krishnaprasad and D. P. Tsakiris. G-snakes: Nonholonomic kinetic chains on Lie groups. In *Proc CDC*, pages 2955–2960, Lake Buena Vista, FL, December 1994.
27. P. S. Krishnaprasad and D. P. Tsakiris. Oscillations, $SE(2)$-snakes and motion control: a study of the roller racer. *Dynamics and Stability of Systems*, 16(4):347–397, 2001.
28. S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *Int J Robotics Research*, 20(5):378–400, 2001.
29. A. D. Lewis. The geometry of the maximum principle for affine connection control systems. Preprint available at: `http://penelope.mast.queensu.ca/~andrew`, 2000.
30. A. D. Lewis. Simple mechanical control systems with constraints. *IEEE Trans Automat Control*, 45(8):1420–1436, 2000.
31. A. D. Lewis and R. M. Murray. Configuration controllability of simple mechanical control systems. *SIAM J Control Optim*, 35(3):766–790, 1997.
32. A. D. Lewis, J. P. Ostrowski, R. M. Murray, and J. W. Burdick. Nonholonomic mechanics and locomotion: the snakeboard example. In *Proc ICRA*, pages 2391–2400, San Diego, CA, May 1994.
33. S. Martínez and J. Cortés. Motion control algorithms for simple mechanical systems with symmetry. *Acta Applicandae Mathematicae*, 2002. To appear.
34. S. Martínez, J. Cortés, and F. Bullo. Analysis and design of oscillatory controls systems. *IEEE Trans Automat Control*, June 2001. Submitted. Available electronically at `http://motion.csl.uiuc.edu`.
35. T. McGeer. Passive dynamic walking. *Int J Robotics Research*, 9(2):62–82, 1990.
36. K. A. Mclsaac and J. P. Ostrowski. A geometric approach to anguilliform locomotion: modelling of an underwater eel robot. In *Proc ICRA*, pages 2843–8, Detroit, MI, May 1999.
37. R. M. Murray, Z. X. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Boca Raton, FL, 1994.
38. S. Ostrovskaya, J. Angeles, and R. Spiteri. Dynamics of a mobile robot with three ball-wheels. *Int J Robotics Research*, 19(4):383–393, 2000.
39. J. P. Ostrowski. Steering for a class of dynamic nonholonomic systems. *IEEE Trans Automat Control*, 45(8):1492–1497, 2000.
40. J. P. Ostrowski and J. W. Burdick. The geometric mechanics of undulatory robotic locomotion. *Int J Robotics Research*, 17(7):683–701, 1998.
41. F. G. Pin and S. M. Killough. A new family of omnidirectional and holonomic wheeled platforms for mobile robots. *IEEE Trans Robotics Automat*, 10(4):480–9, 1994.
42. M. H. Raibert. *Legged Robots that Balance*. MIT Press, Cambridge, MA, 1986.
43. M. Rathinam and R. M. Murray. Configuration flatness of Lagrangian systems underactuated by one control. *SIAM J Control Optim*, 36(1):164–179, 1998.
44. S. Saha, J. Angeles, and J. Darcovich. The design of kinematically isotropic rolling robots with omnidirectional wheels. *Mechanism and Machine Theory*, 30(8):1127–1137, 1995.
45. J. A. Sethian. *Level set methods: Evolving interfaces in geometry, fluid mechanics, computer vision, and materials science*. Cambridge University Press, New York, NY, 1996.

46. M. West and H. Asada. Design of ball wheel mechanisms for omnidirectional vehicles with full mobility and invariant kinematics. *ASME Journal of Mechanical Design*, 119(2):153–161, 1997.

Motion Description Languages for Multi-Modal Control in Robotics

Magnus Egerstedt

magnus@ece.gatech.edu
Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332

Abstract. In this paper we outline how motion description languages provide useful tools when designing multi-modal control laws in robotics. Of particular importance is the introduction of the description length as a measure of how complicated a given control procedure is. This measure corresponds to the number of bits needed for coding the input string. Description length arguments can furthermore be invoked for selecting sensors and actuators in a given robotics application, thus providing a unified framework in which a number of major areas of robotics research can coexist.

1 Introduction

When humans instruct each other how to carry out particular tasks, only a limited number of tokenized instructions are used. In contrast to this, classic control theory specifies a control action to be carried out at each time instant. But, in a number of applications, such as semi-autonomous service robots for industrial and domestic use, intelligent appliances, and communication constrained embedded and/or teleoperated devices, the control procedures have a natural, linguistic flavor. In this paper we take the point of view that such tokenized instructions are useful, not only in particular robotics applications, but also for understanding how computer generated inputs to a robotics system should be defined, selected, and coded in order to minimize the number of bits transmitted from the computer to the robot. This should be achieved while guaranteeing that the system meets its specifications. To this end, an information theoretic approach to control theory will be developed, serving as a useful tool not only for source coding of control signals, but also for describing how symbolic instructions should be interpreted and operated on by continuous systems. Questions concerning what sensors and actuators to use in a given robotics application can be addressed quite elegantly within this framework as well.

In order to understand the interactions between these two heterogeneous components, i.e. between the symbolic computer programs and the continuous device dynamics, different *hybrid architectures*, serving as abstractions between continuous and discrete control, have been suggested. In [6] a general

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 75–89, 2003

model for such hybrid systems is proposed:

$$\begin{aligned}\dot{x} &= f(x, y, v(\lfloor p \rfloor)) \\ \dot{p} &= g(x, y, v(\lfloor p \rfloor)) \\ y &= h(x, v(\lfloor p \rfloor)),\end{aligned}$$

where x is the continuous state of the system and y is the measured output signal. Moreover, v is the symbolic input string from the motor control program, and the evolution of the scalar p triggers the reading of the string v. Here $\lfloor \cdot \rfloor$ denotes the floor operator, and g is assumed to be nonnegative for all arguments.

In this paper we model the way linguistic control signals affect mechanical devices on this form, and we will combine this model with the notion of a *motion description language* (MDL), which refers to a framework for device control, as proposed for example in [5,14,21]. By combining these two ideas we will construct interpreter mechanisms for generating meaningful control commands from symbolic inputs, and the outline of this paper is as follows: In Section 2, the main objects of study, i.e. the motion description languages, will be introduced for representing strings of idealized motions. Section 3 will consequently investigate how to design control laws using the MDL formalism. A cost criterion for evaluating the control laws will be introduced, corresponding to the description lengths of the control procedures. This cost can be interpreted as the number of bits needed for uniquely coding a given control procedure. In Section 4 we will invoke a complexity argument, similar to that in [22], that provides guidelines for how to choose sensors and actuators for a given robotics application.

2 Motion Description Languages

Given a finite set, or alphabet, A, by $A^\star$ we understand the set of all strings of finite length over A. There is a naturally defined binary operation on this set, namely the concatenation of strings, denoted by $\mathbf{a}_1 \cdot \mathbf{a}_2$, i.e. $\mathbf{a}_1 \cdot \mathbf{a}_2 \in A^\star$ if $\mathbf{a}_1, \mathbf{a}_2 \in A^\star$. Relative to this operation, $A^\star$ is a semigroup. If we include the empty string in $A^\star$ it becomes a monoid, i.e. a semigroup with an identity, and a *formal language* is a subset of a free monoid over a finite alphabet.

Now, by a *motion alphabet* we mean a possibly infinite set of symbols representing different control actions that, when applied to a specific robot, define segments of motion. A MDL is thus given by a set of symbolic strings that represent idealized motions, i.e. a MDL is a subset of a free monoid over a given motion alphabet. Particular choices of MDLs become meaningful only when the languages are defined relative to the physical robot that is to be controlled. In this paper, we let the robot dynamics be given by

$$\begin{aligned}\dot{x} &= F(x, u) \\ y &= H(x),\end{aligned}$$

where $u \in U$ is a control signal, $x \in X \subset \Re^n$ is the state of the system, and H is a measurable mapping from X to Y assigning an output value to each point $x \in X$.[1]

Now, it is quite standard, when structuring the navigation task for autonomous mobile robots, to decompose the task into basic building blocks, e.g. *behaviors* [2,7], and in this paper we refer to such building blocks as *modes*. Each mode should contain a description of the preplanned setpoints (generated by the high-level mission planner), of the way the robot should react to state and environmental changes (reactive control), and of when a new mode of operation should guide the behavior of the robot (e.g. specify the transition or arbitration rules). If we formalize these observations, following the development in [5,14], we say that a mode is given by a triple (u, k, ξ), where $u \in U$ is an *open-loop component*, $k : Y \times U \to U$ is a *closed-loop component*, and $\xi : Y \to \{0, 1\}$ is an *interrupt*.

If, at time t_0, the robot receives the input string $(u_1, k_1, \xi_1), \ldots, (u_p, k_p, \xi_q)$, then x evolves according to

$$\begin{aligned} \dot{x} &= F(x, k_1(y, u_1)); \quad t_0 \le t < T_1 \\ &\vdots \qquad\qquad\quad \vdots \\ \dot{x} &= F(x, k_q(y, u_q)); \; T_{q-1} \le t < T_q, \end{aligned}$$

where T_i denotes the time at which the interrupt ξ_i changes from 0 to 1.

The model of a trigger based hybrid system, as described in the introduction, can moreover reproduce this behavior if symbols are interpreted and operated on as follows: Let the input string to the trigger based hybrid system be such that

$$v(i) = (u_i, k_i, \xi_i), \; i \in \mathbb{Z}^+,$$

and let

$$\begin{aligned} \dot{x} &= f(x, y, v(\lfloor p \rfloor)) = f(x, y, (u_{\lfloor p \rfloor}, k_{\lfloor p \rfloor}, \xi_{\lfloor p \rfloor})) = F(x, k_{\lfloor p \rfloor}(y, u_{\lfloor p \rfloor})) \\ y &= h(x, v(\lfloor p \rfloor)) = h(x, (u_{\lfloor p \rfloor}, k_{\lfloor p \rfloor}, \xi_{\lfloor p \rfloor})) = H(x) \\ \dot{p} &= g(x, y, v(\lfloor p \rfloor)) = g(x, y, (u_{\lfloor p \rfloor}, k_{\lfloor p \rfloor}, \xi_{\lfloor p \rfloor})) = \begin{cases} 0 & \text{if } \xi_{\lfloor p \rfloor}(y) = 0 \\ \delta & \text{if } \xi_{\lfloor p \rfloor}(y) = 1, \end{cases} \end{aligned}$$

where δ is a unit impulse, $x(0) = x_0$, and $p(0) = 1$.

It is clear that we now have a construction that allows continuous machines to operate on linguistic inputs in a way that can be given a meaningful control theoretic interpretation.

2.1 Navigation Example

In order to make matters somewhat concrete, we illustrate these ideas with an example, found in [12]. What makes the control of mobile robots particularly

[1] At this point we will not specify U and Y further but, as will be shown later, different choices of input-output sets have a direct impact on how the control procedures should be selected and coded.

challenging is the fact that the robots operate in unknown, or partially unknown environments. Any attempt to model such a system must take this fact into account. We achieve this by letting the robot make certain observations about the environment, and we let the robot dynamics be given by

$$\dot{x} = v,\ x, v \in \Re^2$$
$$y_1 = x,\ y_2 = c_f(x),\ y_1, y_2 \in \Re^2,$$

where c_f is the contact force from the environment. The contact force could either be generated by tactile sensors in contact with the obstacle or by range sensors such as sonars, lasers, or IR-sensors.

Relative to this robot it is now possible to define the following two MDLs for distinguishing between the open-loop and the closed-loop cases.

Definition 1 (Open-Loop and Closed-Loop MDLs). Let the Open-Loop MDL, $\mathcal{L}_{ol}$, be given by the free monoid over the set

$$\big\{(u, k, \xi) \mid u \in \Re^2,\ k(y_1, y_2, u) = u,\ \xi : \Re^4 \to \{0, 1\}\big\},$$

where ξ is any measurable mapping from $\Re^4$ to $\{0, 1\}$. Consequently, let the Closed-Loop MDL, $\mathcal{L}_{cl}$, be given by the free monoid over the set

$$\big\{(u, k, \xi) \mid u = x_F,\ k(y_1, y_2, u) \in \{\kappa(u - y_1), Dy_2\},\ \xi : \Re^4 \to \{0, 1\}\big\},$$

where $\kappa > 0$ is a given constant, D is any linear mapping from $\Re^2$ to $\Re^2$, and x_F is the given, desired robot position.

By a point-to-point navigation task we understand the problem of moving the robot between given initial and final states in a safe way, and one straightforward question to investigate is how to design short strings of modes in $\mathcal{L}_{ol}$ and $\mathcal{L}_{cl}$ that achieve this? The reason why this is an interesting question is that the paths generated by $\mathcal{L}_{ol}$ on the robot are identical to those paths that are considered in the literature on the complexity of minimum time or shortest path algorithms for robot motion planning in dynamic environments [9].

This problem was studied in [12] and the following two theorems were proved:

Theorem 1 (Open-Loop Complexity). *(Egerstedt [12]) In a convex environment populated by N convex, polygonal obstacles with $M \geq 3$ vertices each, the length of the shortest string (in the worst case) in $\mathcal{L}_{ol}$ that drives the robot between x_0 and x_F is of order $\mathcal{O}(NM)$.*

The proof of this theorem consists of establishing tight bounds on the number of segments necessary for producing a piecewise linear path between x_0 and x_F. This path should furthermore not intersect the interior of any obstacle. To find these bounds is equivalent to finding the shortest string in $\mathcal{L}_{ol}$, since the only paths that can be generated on the particular robot under investigation are piecewise linear when using words in $\mathcal{L}_{ol}$.

In [12] it was furthermore shown how to construct a closed-loop control strategy that requires a lower number of instructions than $\mathcal{O}(NM)$, under the assumption that the contact force from an obstacle in contact with the robot is parallel to the outward normal of the surface of the obstacle.[2]

Theorem 2 (Closed-Loop Complexity). *(Egerstedt [12]) In a convex environment populated by N convex, polygonal obstacles, an upper bound on the length of the shortest (in the worst case) string in $\mathcal{L}_{cl}$ is of order $\mathcal{O}(N)$.*

The proof of Theorem 2 is constructive. When the robot is not in contact with an obstacle, we let $k(y_1, y_2, u) = \kappa(u - y_1)$, where $\kappa > 0$, as in Definition 1. On the other hand, when the robot is in contact with an obstacle it seems reasonable to follow the contour of that obstacle, as suggested in [17]. The control strategy that we propose for this guarantees that the robot reaches the unique global minimum (the point closest to x_F on the obstacle), while committing to a clockwise or counter-clockwise obstacle negotiation, before it leaves the contour of the obstacle. The multi-modal control sequence is thus an element in the set $\sigma_{GA} \cdot (\sigma_{OA} \cdot \sigma_{GA})^\star \subset \mathcal{L}_{cl}$, where GA and OA denotes "goal-attraction" and "obstacle-avoidance" respectively, and where $a^\star = \{\emptyset, a, aa, aaa, ...\}$. The individual modes $\sigma_{GA} = (u_{GA}, k_{GA}, \xi_{GA})$ and $\sigma_{OA} = (u_{OA}, k_{OA}, \xi_{OA})$ are furthermore given by

$$
\begin{cases}
u_{GA} = x_F \\
k_{GA}(y_1, y_2, u) = \kappa(u - y_1) \\
\xi_{GA}(y_1, y_2) = \begin{cases} 0 \text{ if } \langle y_2, x_F - y_1 \rangle \geq 0 \\ 1 \text{ otherwise} \end{cases}
\end{cases}
$$

$$
\begin{cases}
u_{OA} = x_F \\
k_{OA}(y_1, y_2, u) = cR(-\pi/2)y_2 \\
\xi_{OA}(y_1, y_2) = \begin{cases} 0 \text{ if } \langle y_2, x_F - y_1 \rangle < 0 \text{ or } \angle(x_F - y_1, y_2) < 0 \\ 1 \text{ otherwise.} \end{cases}
\end{cases}
$$

Here $R(\theta)$ is a 2×2 rotation matrix, $c > 0$, and $\angle(\alpha, \beta)$ denotes the angle between the vectors α and β.

An example of using this multi-modal control sequence is shown in Figure 1, and what this result means is that when the sensory information available to us is sufficiently abundant, fewer instructions are necessary in the feedback case than in the open-loop case. However, this way of measuring the complexity of input strings in terms of their *string lengths* is not a very natural complexity measure for the following two reasons:

(i) The feedback instructions rely on sensory data, but string lengths do not capture the complexity associated with making these measurements in any meaningful way.

[2] Since no unique normal vector exists when x belongs to a vertex on the boundary on P, we let the output, y_2, take on any value in the *normal cone*, i.e. $y_2 \in N_P(x) = \{h \in \Re^2 \mid \langle h, y - x \rangle \leq 0, \ \forall y \in P\}$.

(ii) If the input strings were to be transmitted over a communication channel, the number of bits needed for coding the instructions should affect the complexity of the input strings, which is not reflected by the string lengths.

Based on these two observations, we instead introduce the *description length*, i.e. the number of bits needed for describing a given input string, as a natural complexity measure when designing multi-modal control procedures over motion description languages.

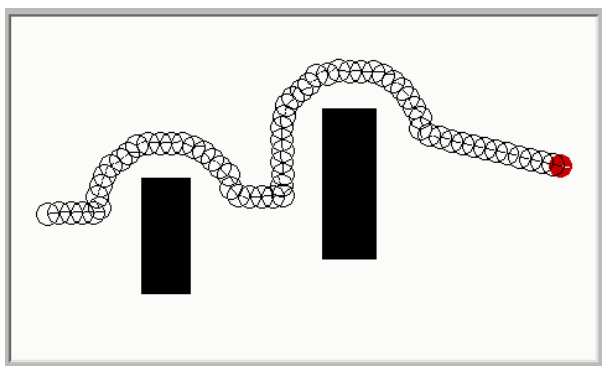


Fig. 1. A multi-modal input string in $\mathcal{L}_{cl}$ is used for negotiating two rectangular obstacles. Depicted is a simulation of a Nomadic Scout in the Nomadic `Nserver` environment.

3 Description Lengths

If we allow the control modes to contain mappings to or from sets of infinite cardinality, it is clear that the description lengths would be infinite as well. Therefore it is necessary to focus our attention on finitely describable control procedures. However, this is not really a restriction. Recall that the robot dynamics was given by $\dot{x}(t) = F(x(t), u(t))$, where u is the control signal and $x \in X \subset \Re^n$ is the state of the system. Based on the choice of actuators, u belongs to some set U. Since all real, physical stepper motors have finite range and resolution, and since the control signals are computer generated, U has finite cardinality, which is the case in the emerging area of *quantized control* as well [3,11,15]. It is also conceivable, as pointed out in [1], that we would like to further restrict the control signal, e.g. $u \in \{u_{\min}, u_{\max}\} = U$, resulting in somewhat more tractable design and verification processes. This would limit the cardinality of U further, and for analogous reasons, we let the observations made by the robot take on values in the finite set Y. This

implies that the input strings take on values in the set $\Sigma^\star$, where Σ is the finite set $\Sigma = U \times U^{Y \times U} \times \{0,1\}^Y$.

Now, assume that we are given a string of modes $\boldsymbol{\sigma} \in \Sigma^\star$. Then the number of bits needed for describing $\boldsymbol{\sigma}$ uniquely is given by the following definition [10,22]:

Definition 2 (Description Length). Consider the finite set Σ. We say that a word $\boldsymbol{\sigma} \in \Sigma^\star$ has description length

$$\mathcal{D}(\boldsymbol{\sigma}, \Sigma) = |\boldsymbol{\sigma}| \log_2(card(\Sigma)),$$

where $|\boldsymbol{\sigma}|$ denotes the length of $\boldsymbol{\sigma}$, i.e. the number of modes in the string, and $card(\cdot)$ denotes cardinality.

The description length thus tells us how complicated $\boldsymbol{\sigma}$ is, i.e. how many bits we need for describing it, and since $\Sigma = U \times U^{Y \times U} \times \{0,1\}^Y$ we directly see that a higher resolution sensor (e.g. laser-scanners) result in a larger Y than what is the case for a lower resolution sensor (e.g. sonars). A better sensor might thus make the control procedures significantly more complicated while only providing marginally better performance. This trade-off between complexity and performance is something that can be capitalized on when designing control laws as well as when choosing what sensors and actuators to use. The idea is simply to pick $(U, Y, \boldsymbol{\sigma} \in \Sigma^\star)$ in such a way as to make the robot behave satisfactory, while minimizing the description lengths of the control procedures.

3.1 Free-Running, Feedback Automata

In order to illustrate how the description length measure can be a useful tool in multi-modal control design, we focus our attention on the finitely describable aspects of finite state machines.

If we let X, U be finite sets, and let $\delta \in X^{X \times U}$, then we can identify (X, U, δ) with a *finite automaton* (see for example [16]), whose operation is given by $x_{k+1} = \delta(x_k, u_k)$. If we add another finite set Y and a mapping $\gamma \in Y^X$ to the definition, we get an *output* automaton $(X, Y, U, \delta, \gamma)$, where $x_{k+1} = \delta(x_k, u_k)$ and $y_k = \gamma(x_k)$.

However, in order to let finite automata read strings of control modes, the model must be modified in such a way that instruction processing is akin to the way in which differential equations "process" piecewise constant inputs. For this, a dynamical system called a *free-running, feedback automaton* (FRF-automaton), was introduced in [14]. The idea is to let such an automaton read an input from a given alphabet, and then advance the state of the automaton repeatedly (free-running property) without reading any new inputs until an interrupt is triggered. Additional structure is furthermore imposed on the input set to allow for feedback signals to be used. Hence a FRF-automaton is a free-running automaton whose input alphabet admits the structure $\Sigma =$

$U \times K \times \Xi$, where, as before, U is a finite set, $K = U^{Y \times U}$, and $\Xi = \{0,1\}^Y$. Hence, the input to a FRF-automaton is a triple (u, k, ξ), where $u \in U$, $k : Y \times U \to U$, and $\xi : Y \to \{0,1\}$.

Definition 3 (Free-Running, Feedback Automaton). Let X, Y, U be finite sets and let $\delta : X \times U \to X$, $\gamma : X \to Y$ be given functions. Let $\Sigma = U \times K \times \Xi$, where U is a finite set, $K = U^{Y \times U}$, and $\Xi = \{0,1\}^Y$. We say that $(X, \Sigma, Y, \delta, \gamma)$ is a free-running, feedback automaton whose evolution equation is

$$x_{k+1} = \delta(x_k, k_{l_k}(y_k, u_{l_k})), \;\; y_k = \gamma(x_k)$$
$$l_{k+1} = l_k + \xi_{l_k}(y_k),$$

given the input string $(u_1, k_1, \xi_1) \cdots (u_p, k_p, \xi_p) \in \Sigma^\star$.

It should be noted that the free-running property of the FRF-automata implies that they can, in general, be guided along a path using fewer instructions than the classical finite automata. However, since the input set to a finite automaton is simply the finite set U, while the input set to the corresponding FRF-automaton is of the form $U \times K \times \Xi$, the input set has a higher cardinality in the latter of these cases. As already pointed out, any reasonable measure of the complexity of a control procedure must take the size of the input space into account since the number of bits required to code a word over a given alphabet typically depends logarithmically on the size of the alphabet. (See for example [10].) This dependency is captured in a natural way if we define the specification complexity of a control task as the description length of the input sequence.

Definition 4 (Specification Complexity). Consider a FRF-automaton, A, with state space X and input set Σ. Let $\boldsymbol{\sigma}$ be the word of minimal description length over Σ that drives the automaton between two given states $x_0, x_f \in X$. We then say that the task of driving A between x_0 and x_f has specification complexity $\mathcal{C}(A, x_0, x_f) = \mathcal{D}(\boldsymbol{\sigma}, \Sigma)$.

3.2 Feedback Can Reduce the Specification Complexity

We here review the main results from [12] and [14] in order to see how the complexity of the instructions can be reduced when using landmark-based navigation. Since the cardinality of the input set depends on the size of the domain of the feedback mapping, a smaller domain can be expected to reduce the complexity. In order to make this observation rigorous, we need to introduce the notions of ballistic reachability and control-invariant reachability: A set $X_s \subset X$ is *ballistically reachable from* x if there exists a $u \in U$ such that $\delta(x, u^q) \in X_s$ for some $q \in \mathbb{Z}^+$. Furthermore, X_s is ballistically reachable from $X_t \subset X$ if there exists a $u \in U$ such that for all $x \in X_t$ it holds that $\delta(x, u^{q(x)}) \in X_s$ for some $q(x) \in \mathbb{Z}^+$. An element $x \in X_s \subset X$ is said to be *control-invariantly reachable in* X_s if it can be reached from all states in X_s without the trajectory leaving X_s.

Now, in order to compare purely open-loop control, i.e. when no observations are made, with a situation where sensory information is available, we must be able to generate open-loop motions on the FRF-automata. It is clear that the input sequence $\boldsymbol{\sigma}_{ol} = (u_1, k_{ol}, \xi_{ol}) \cdots (u_q, k_{ol}, \xi_{ol}) \in \Sigma^\star$, where $k_{ol}(u, y) = u \ \forall u \in U, \ y \in Y$, $\xi_{ol}(y) = 1 \ \forall y \in Y$ achieves this. However, this word has length q, and it is drawn from the input alphabet $\Sigma = U \times U^{Y \times U} \times \{0, 1\}^Y$, and thus the description length is $\mathcal{D}(\boldsymbol{\sigma}_{ol}, \Sigma) = q \log_2(card(\Sigma))$. But, this is clearly not the result we would like to have. Instead we can restrict the input alphabet to be $\Sigma_{ol} = U \times \{k_{ol}\} \times \{\xi_{ol}\}$, which has cardinality $card(U)$. The description length of $\boldsymbol{\sigma}_{ol}$ is now $\mathcal{D}(\boldsymbol{\sigma}_{ol}, \Sigma_{ol}) = q \log_2(card(U))$, relative to the smaller input set Σ_{ol}.

Consider the connected, classical, finite automaton $A = (X, U, \delta)$. We recall that the *backwards eccentricity* of a state, $ecc(A, x)$, denotes the minimum number of instructions necessary for driving the automaton from any other state to x. (See for example [8].) We furthermore let the *radius* of A be given by

$$radius(A) = \min_{x \in X} ecc(A, x).$$

Now, consider the FRF-automaton $\tilde{A}$. If we let

$$\mathcal{C}(\tilde{A}, x) = \max_{x_0 \in X} \mathcal{C}(\tilde{A}, x_0, x),$$

then we directly get that

$$\mathcal{C}(A_{ol}, x) \geq radius(A) \log_2(card(U)),$$

where A_{ol} is the FRF-automaton $(X, Y, \Sigma_{ol}, \delta, \gamma)$, and A is the classical automaton (X, U, δ).

Theorem 3. *(Egerstedt and Brockett [14]) Assume that $card(U) \geq 2$. Suppose that $x_f \in X_f$, where X_f is an observable subset for the finite automaton A, i.e. it is possible to construct an observer that converges in a finite number of steps on the subset X_f. Assume that $card(\gamma(X_f)) < card(X_f)$ and $\gamma(X_f) \cap \gamma(X \backslash X_f) = \emptyset$. If X_f is ballistically reachable from $X \backslash X_f$, and x_f is control-invariantly reachable in X_f, then there exists a FRF-automaton $A_{FRF} = (X, Y, \Sigma', \delta, \gamma)$ such that*

$$\frac{\mathcal{C}(A_{FRF}, x_f)}{\mathcal{C}(A_{ol}, x_f)} \leq \frac{4 card(X_f)}{radius(A)}.$$

The proof of Theorem 3 consists of constructing one closed-loop instruction, consisting of two parts. The first part is given by a ballistic motion, i.e. a long open-loop motion, followed by an observer-based feedback instruction defined on the reduced set X_f, as seen in Figure 2(a). However, goals are seldom final goals. More often they tend to be intermediate goals in a grander scheme. This is for instance the case when mobile robots are navigating using landmarks.

3.3 Navigation Using Landmarks

It is clear that the premises on which the previous theorem is based are too restrictive to capture the *chained* structure that intermediary goals give rise to. Instead we need to extend the trajectories through a series of goal states. This can be achieved by assuming that we work with an automaton where subset-observers can be designed around different states, i.e. the intermediate goals. We also assume that the sets on which the observers are defined are ballistically reachable from each other. We could then use open-loop control for driving the system between these sets on the parts of the state space where the lack of sensory information prevents effective use of feedback. We compliment this with feedback controllers on the subsets where subset-observers can be constructed, as seen in Figure 2(b).

Theorem 4 (Navigation Using Landmarks). *(Egerstedt and Brockett [14]) Assume that* $card(U) \geq 2$. *Let the sets* $X_1, \ldots, X_n$ *be disjoint, observable subsets with cardinality less than or equal to* C, *where* $card(\gamma(X_i)) < C$, $i = 1, \ldots, n$, $\gamma(X_i) \cap \gamma(X \backslash X_i) = \emptyset$, $\gamma(X_i) \cap \gamma(X_j) = \emptyset$, $i \neq j$. *Let* $x_f \in X_n$ *be control-invariantly reachable in* X_n *and let* X_1 *be ballistically reachable from* x_0. *Assume that there exists intermediary goals* $x_i \in X_i$, $i = 1, \ldots, n-1$ *such that* x_i *is control-invariantly reachable in* X_i *and* X_{i+1} *is ballistically reachable from* x_i. *Then there exists a FRF-automaton* $A_{FRF} = (X, Y, \Sigma', \delta, \gamma)$ *such that*

$$\frac{\mathcal{C}(A_{FRF}, x_f)}{\mathcal{C}(A_{ol}, x_f)} \leq \frac{4nC}{radius(A)}.$$

One conclusion to be drawn from Theorem 4 is that the increase in description length, caused by the summation over many intermediate goals, can be counter-acted by making the sets where feedback is effective small. In the mobile robot case, this would correspond to using many easily detectable landmarks as a basis for the navigation system.

4 A Unified Approach to Control and Hardware Design

Another issue that can be tackled quite elegantly within the specification complexity context is how sensor selection for mobile robots affects the description lengths of the control procedures. In other words, we want to be able to determine, in a rigorous manner, which observations to make. But, we furthermore want to do this while simultaneously addressing a number of other potentially interesting questions in robotics. By scanning the proceedings from the *IEEE Conference on Robotics and Automation*, a rough taxonomy over five broad (not necessarily disjoint) areas of research can be identified:

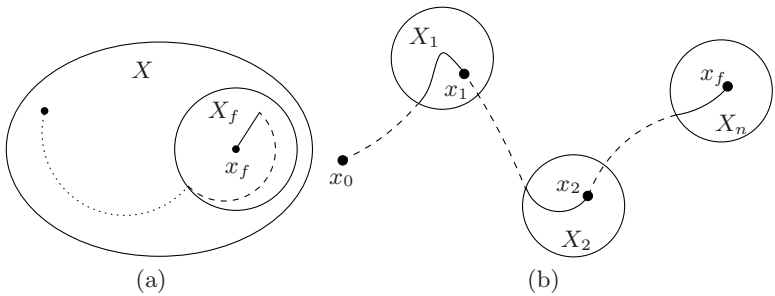


Fig. 2. In the left figure, the observer-based, single instruction, goal-finding procedure in Theorem 3 is depicted. The dotted line is the open-loop part of the evolution, the dashed line defines the part where the observer is converging, and the solid line is the last part of the evolution. The right figure shows the chained extension of this instruction over a collection of landmarks, as in Theorem 4.

i. Electro-mechanical design of robot platforms, sensors, and actuators;
ii. The development of rational methods for collecting, compressing, and analyzing sensory data;
iii. Deliberative, high-level mission planning;
iv. Control design for different classes of dynamical systems; and
v. The construction of appropriate software architectures and data structures for internal representations.

Since these areas of research have been developed somewhat in parallel, questions about software-hardware co-design have not been easy to address (or even to raise). What is novel in this paper is thus a unified model that incorporates some aspects of all of these five research areas and allows us to ask questions simultaneously about hardware design, mission planning, and robot dynamics.

Let us assume that we have access to a collection of sensors $\mathcal{Y}$ (with a corresponding $H : X \to Y$ for each $Y \in \mathcal{Y}$), and a set of possible actuators $\mathcal{U}$. One could thus ask the following question

$$(P): \min_{U \in \mathcal{U}, Y \in \mathcal{Y}} \left\{ \min_{\boldsymbol{\sigma} \in \Sigma^\star} |\boldsymbol{\sigma}| \log_2\big(card(\Sigma)\big) \right\},$$

subject to the constraint that $\boldsymbol{\sigma}$ makes the robot meet the given specifications. By solving P we would thus solve sensor and actuator selection as well as control design on both a low level ("What should the individual σ's look like?") and at a high level ("How should the strings of modes be chosen?") in a unified way. It should be noted that the solution is dependent on the dynamics of the robot as well as on the particular task the robot is asked to carry out. It is furthermore clear that by solving P we directly address questions residing in research areas **i** and **iv** in the previously established

taxonomy. Area **iii** is also touched upon since the open-loop component in the mode description can be thought of as corresponding to setpoints, or landmark locations, and by forming strings over Σ we thus provide a list of landmarks for the robot to move between. Area **ii** is also addressed by solving P through the construction of the output function $H : X \rightarrow Y$. Sensor fusion, virtual sensors, and feature extraction all correspond to designing different output functions, and the only remaining area untouched by P is thus area **v**. It can be argued that the use of motion description languages has implications for the software architecture as well [18] (the programmer should specify control triples directly), but we will let this aspect fall outside the scope of this paper.

5 Preliminary Results

At the present, we do not have the general solution to P. However, initial work has been conducted along these lines, and the findings are summarized in the following paragraphs:

As already shown in Section 3, the use of the description length as a measure of how complicated it is to specify control procedures can be put to work for answering a question of fundamental significance in control theory, namely that concerning the computational benefits of feedback. The reason why this work is promising is twofold: First, the result can be thought of as a special case of the sensor selection problem since it tells us whether or not sensors should be used at all. Secondly, the many visible and successful applications of feedback mechanisms at work testify to its effectiveness and over the years a variety of arguments have been advanced showing why, in particular settings, it is useful. The models commonly used bring to the fore considerations of sensitivity, uncertainty, etc, and to this list we have now added an argument for the usefulness of feedback in terms of the effect it has on reducing the description lengths of the control procedures.

In [13] it was investigated how to choose the resolution and the scope of the range sensors in order to manage the navigation task successfully at the same time as the description lengths should be kept as short as possible. The main theorem in that work states that if the robot dynamics is evolving on a bounded lattice, where "k-sensors" can detect neighboring states of a distance less than or equal to k from the present state of the system, then there is an optimal choice, $k^\star$, of such a range-sensor in terms of the number of bits needed for coding the control strategy. Furthermore, the number of bits needed for coding the control procedure was found to be convex in k (over the real numbers) which simplifies the design of optimization algorithms for finding $k^\star$ (over the positive integers) significantly, as shown in Figure 3.

This result sets the tone for the development of sensor selection algorithms in more realistic settings. For instance, different choices of k can be thought of as different range sensors in robotics applications. A larger k, e.g. a laser

scanner, can thus be compared to short-range sensors such as infra-red sensors, in terms of the specification complexity. The work also suggests that it should be possible to optimize not only over k, but also over some parameter that describes the accuracy and resolution of the actuators.

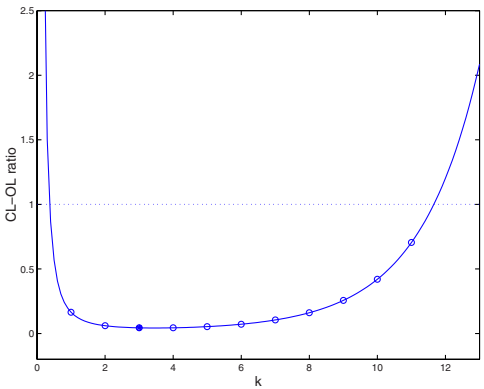


Fig. 3. The ratio between the closed-loop and open-loop specification complexity is shown as a function of k.

6 Further Issues

Previously we have seen that given $\boldsymbol{\sigma} \in \Sigma^\star$, a total number of $|\boldsymbol{\sigma}| \log_2(card(\Sigma))$ bits are needed for specifying $\boldsymbol{\sigma}$. Now, if we assume that we have been able to establish a probability distribution over Σ we can use optimal coding schemes, such as the Huffman code [19], for finding the shortest expected number of bits $l^\star(\Sigma)$ needed for coding an element drawn at random from Σ. Shannon's classic source coding theorem [23] tells us that $\mathcal{H}(\Sigma) \leq l^\star(\Sigma) < \mathcal{H}(\Sigma) + 1$, where the *entropy* $\mathcal{H}(\Sigma)$ is given by

$$\mathcal{H}(\Sigma) = -\sum_{i=1}^{card(\Sigma)} p_i \log_2 p_i.$$

Here the interpretation is that the control triple $\sigma_i \in \Sigma$ occurs with probability p_i, and it should be noted that a probability distribution over Σ corresponds to a specification of what modes are potentially useful.

But, to establish such a probability distribution over a structured set, such as the set of modes, is not a trivial task, and four possible routes can be identified:

(i) Derive potentially useful modes directly using hybrid control synthesis tools [4,20];

(ii) Draw inspiration from existing, man-made multi-modal systems, e.g. autopilots for unmanned autonomous vehicles [24], or behaviors in behavior-based robotics [2,7];
(iii) Observe how human operators instruct mobile robots, or how biological systems are structured, and reconstruct the modes from experimental data; and
(iv) Apply reinforcement learning techniques in order to find potentially useful modes.

Once an appropriate model has been established for a given robot platform equipped with a collection of sensors and actuators, and a coding strategy has been decided on, we are ready to synthesize multi-modal control laws. As already argued for extensively in this document, we want to keep the mode descriptions short. In other words, assume that we have established a probability distribution over Σ and let $l_C(\boldsymbol{\sigma})$ denote the number of bits used for coding $\boldsymbol{\sigma} \in \Sigma^\star$ under coding strategy C. What one wants to achieve is thus to find $\hat{\boldsymbol{\sigma}} \in \Sigma^\star$ that minimizes $l_C(\boldsymbol{\sigma})$, while making sure that the system meet the specifications, such as driving the robot between given points while avoiding obstacles.

Acknowledgments.
The support from NSF through the program EHS NSF-01-161 (grant # 0207411) is gratefully acknowledged. The author also wishes to thank Roger Brockett for many insightful comments in the emerging area of linguistic control.

References

1. R. Alur, J. Esposito, M. Kim, V. Kumar, and I. Lee. Formal Modeling and Analysis of Hybrid Systems: A Case Study in Multirobot Coordination. *Proceedings of the World Congress on Formal Methods*, LNCS 1708, pp. 212–232, Springer, 1999.
2. R.C. Arkin. *Behavior Based Robotics.* The MIT Press, Cambridge, MA, 1998.
3. A. Bicci, A. Margio, and B. Piccoli. On the Reachability of Quantized Control Systems. *IEEE Transactions on Automatic Control*, Vol. 47, No. 4, April 2002.
4. M.S. Branicky. Multiple Lyapunov Functions and Other Analysis Tools for Switched and Hybrid Systems. *IEEE Transactions on Automatic Control*, Vol. 43, No. 4, pp. 475–482, April 1998.
5. R.W. Brockett. On the Computer Control of Movement. In the *Proceedings of the 1988 IEEE Conference on Robotics and Automation*, pp. 534–540, New York, April 1988.
6. R.W. Brockett. Hybrid Models for Motion Control Systems. In *Perspectives in Control*, Eds. H. Trentelman and J.C. Willems, pp. 29–54, Birkhäuser, Boston, 1993.

7. R. Brooks. A Robust Layered Control System for Mobile Robots. *IEEE Journal of Robotics and Automation*, Vol. RA-2, No. 1, pp. 14–23, 1986.
8. F. Buckley and F. Harary. *Distance in Graphs.* Addison-Wesley Publishing Company, New York, 1990.
9. J. Canny. *The Complexity of Robot Motion Planning.* The MIT Press, Cambridge, MA, 1987.
10. T.M. Cover and J.A. Thomas. *Elements of Information Theory*, John Wiley & Sons, Inc., New York, 1991.
11. D.F. Delchamps. The Stabilization of Linear Systems with Quantized Feedback. *Proceedings of the IEEE Conference on Decision and Control*, Austin, TX, Dec. 1988.
12. M. Egerstedt. Linguistic Control of Mobile Robots. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Maui, Hawaii, Oct. 2001.
13. M. Egerstedt. Some Complexity Aspects of the Control of Mobile Robots. *American Control Conference*, Anchorage, Alaska, May, 2002.
14. M. Egerstedt and R.W. Brockett. Feedback Can Reduce the Specification Complexity of Motor Programs. To appear in *IEEE Transactions on Automatic Control*, 2002.
15. N. Elia and S.K. Mitter. Stabilization of Linear Systems With Limited Information. *IEEE Transactions on Automatic Control*, Vol. 46, No. 9, Sept. 2001.
16. A. Ginzburg. *Algebraic Theory of Automata.* ACM Monograph Series, Academic Press, New York, 1968.
17. J.E. Hopcroft and G. Wilfong. Motion of Objects in Contact. *The International Journal of Robotics Research*, Vol. 4, No. 4, pp. 32–46, 1986.
18. D. Hristu and S. Andersson. Directed Graphs and Motion Description Languages for Robot Navigation and Control. *IEEE Conference on Robotics and Automation*, May 2002.
19. D.A. Huffman. A Method for the Construction of Minimum Redundancy Codes. *Proceedings of IRE*, Vol. 40, pp. 1098–1101, 1952.
20. M. Johansson and A. Rantzer. Computation of Piecewise Quadratic Lyapunov Functions for Hybrid Systems. *IEEE Transactions on Automatic Control*, Vol. 43, No. 4, pp. 555–559, April 1998.
21. V. Manikonda, P.S. Krishnaprasad, and J. Hendler. Languages, Behaviors, Hybrid Architectures and Motion Control. In *Mathematical Control Theory*, Eds. Willems and Baillieul, pp. 199–226, Springer-Verlag, 1998.
22. J. Rissanen. *Stochastic Complexity in Statistical Inquiry.* World Scientific Publishing Company, River Edge, NJ, 1989.
23. C.E. Shannon. A Mathematical Theory of Communication. *Bell Systems Technical Journal*, Vol. 27, pp. 379–423, 1948.
24. C. Tomlin, G.J. Pappas, and S. Sastry. Conflict Resolution for Air Traffic Management: A Study in Multi-Agent Hybrid Systems. *IEEE Transactions on Automatic Control*, Vol. 43, No. 4, April 1998.

Polynomial Design of Dynamics-based Information Processing System

Masafumi Okada and Yoshihiko Nakamura

Univ. of Tokyo, 7-3-1 Hongo Bunkyo-ku, JAPAN

Abstract. For the development of the intelligent robot with many degree-of-freedom, the reduction of the whole body motion and the implementation of the brain-like information system is necessary. In this paper, we propose the reduction method of the whole body motion based on the singular value decomposition and design method of the brain-like information processing system using the nonlinear dynamics with polynomial configuration. By using the proposed method, we design the humanoid whole body motion that is caused by the input sensor signals.

1 Introduction

For the robot intelligence such as learning and searching, Neural Network and Generic Algorithm are mainly used as powerful tools. In these methods, a cost function is set and optimized. The focuses of conventional researches are how to define a cost function and how to optimize the cost function that defines robot motions based on sensor signals. The information processing in these methods is represented as the following sequence of modules.

(i) Recognition of environment using sensor signal.
(ii) Selection of the robot motion.
(iii) Realization of the motion

Because this procedure is suitable for the recent computer (Neumann computer), the development of the faster processor enables the more sensing, selection and the faster calculation. And so far, many effective methods have been proposed. However, because this procedure reaches to the flame problem in the end, a new approach to the robot intelligence is expected.

Recently, the approach that the intelligence is the result of cooperation between the robot body dynamics and the environments has received attention. For example, in Subsumption Architecture[1], some modules are connected each other, which yields robot motion based on the sensor feedback through the environment. This method aims at producing the robot intelligence using interaction of the robot body and environment.

Since Freeman showed the dynamical phenomenon in the rabbit olfactory such as entrainment and chaos phenomenon[2–4], some researchers have tried to realize the robot intelligence using dynamical phenomena. Nakamura and

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 91–104, 2003

Sekiguchi developed the information processing system using chaotic dynamics[5,6]. They designed a control algorithm for a mobile robot using entrainment and synchronization of the robot dynamics and environment dynamics based on Arnold differential equation.

In this paper, we propose the dynamics-based brain-like information processing system using dynamical phenomena such as entrainment and detrainment to attractors, and give a design strategy of the dynamics-based information processing system that handles humanoid whole body motions. The nonlinear dynamics with polynomial representation memorizes, generates and transits humanoid whole body motions based on the entrainment and detrainment phenomenon . Some design methods of the nonlinear dynamics that has an attractor using the Lyapunov function have been proposed [7]∼[9]. In our method, the dynamics is defined as the vector field in N dimensional space and it has an attractor to some closed curved lines. By changing the vector field configuration, the dynamics behavior is changeable.

On the other hand, because the whole body motions of humanoid robots consist of many joint angles data, it requires much computational quantity to deal with those motions. In this paper, we propose a reduction and symbolization method of the whole body motion based on the principal component analysis[10] using singular value decomposition.

2 Dynamics and Whole Body Motion

The human does not remember the time sequence pattern of the joint angle respect to a whole body motion. The motions are symbolized and selected appropriately based on the internal state and input sensor signals. We show this process and phenomenon corresponding to the dynamics. Consider the following discrete time dynamical equation.

$$\boldsymbol{x}[k+1] = \boldsymbol{x}[k] + \boldsymbol{g}(\boldsymbol{u}[k], \boldsymbol{x}[k]) \tag{1}$$

$\boldsymbol{x} \in \boldsymbol{R}^N$ is the state vector and $\boldsymbol{u} \in \boldsymbol{R}^L$ is the input signal. $\boldsymbol{x}[k]$ moves in the N dimensional space. If this dynamics has attractor to one closed curved line C, $\boldsymbol{x}[k]$ is entrained to C according to $k \to \infty$ with an initial condition $\boldsymbol{x}_0$ and $\boldsymbol{u}[k] = 0$. Suppose that the C represents the time sequence data of a motion M. The time sequence data of $\boldsymbol{x}[k]$ yields that of humanoid joint angles, which means the dynamics memorizes and generates the humanoid whole body motion.

Suppose the dynamics in equation (1) has some attractors and transits to each attractors by the input signal $\boldsymbol{u}[k]$, which means the motion transition of the humanoid robot based on the input sensor signal.

In this paper, we design the brain-like information processing by using nonlinear dynamics that memorizes and generate the whole body motion as an attractor.

3 Motion Reduction and Symbolization

For the design of a dynamics that handles a humanoid whole body motion, a reduction is necessary to decrease the calculation. By the mapping function and its inverse function, the whole body motion is reduced to the symbol (reduced dimensional data) and restored from symbol. Consider the humanoid robot with n degrees-of-freedom. The humanoid motion data Y is defined as follows using time sequence data $y_i[k]$.

$$Y = \begin{bmatrix} y_1[1] & y_1[2] & \cdots & y_1[m] \\ y_2[1] & y_2[2] & \cdots & y_2[m] \\ \vdots & \vdots & & \vdots \\ y_n[1] & y_n[2] & \cdots & y_n[m] \end{bmatrix} \tag{2}$$

For example, $y_i[k]$ means angles of each joint. By the singular value decomposition of Y

$$Y = USV^T \tag{3}$$

$$U = \left[U_1 \middle| U_2 \right] \tag{4}$$

$$S = \left[\begin{array}{c|c} S_1 & \\ \hline & S_2 \end{array} \right] \tag{5}$$

$$S_1 = \mathrm{diag} \left\{ s_1 \ s_2 \ \cdots \ s_r \right\} \tag{6}$$

$$S_2 = \mathrm{diag} \left\{ s_{r+1} \ s_{r+2} \ \cdots \ s_n \right\} \tag{7}$$

$$V^T = \left[\begin{array}{c} V_1^T \\ \hline V_2^T \end{array} \right] \tag{8}$$

if $s_r \gg s_{r+1}$ is satisfied, Y is reduced to r dimensional motion V_1^T as follows.

$$Y = FV_1^T \tag{9}$$

$$F = U_1 S_1 \tag{10}$$

Here, $U_1 \in \boldsymbol{R}^{n \times r}$, $V_1^T \in \boldsymbol{R}^{r \times m}$. F is the mapping function from the symbol to whole body motion. This result shows that

(i) The whole body motion Y that represents a curved line in n dimensional space, is symbolized by a reduced dimensional curved line V_1^T in r dimensional space. If $y_i[k]$ $(i = 1, \ \cdots \ n, \ k = 1, \ \cdots \ m)$ is the periodic sequence, Y and V_1^T mean the closed curved lines M and C respectively.
(ii) By checking the singular value s_i $(i = 1, 2, \cdots, n)$, the appropriate r can be selected.
(iii) The first column vector of V_1 is the first principal component of the motion Y, the second column vector is the second principal component.
(iv) The inverse function of F is $S_1^{-1} U_1^T$

4 Design of Dynamics-Based Information Processing System

4.1 Design of the nonlinear dynamics

Consider the reduced whole body motion in N dimensional space. The time sequence data of each degree-of-freedom is set as $\xi_i[k]$ $(i = 1, \cdots N,\ k = 1, \cdots m)$. Assuming the time sequence data is periodic, $\boldsymbol{\xi}[k]$

$$\boldsymbol{\xi}[k] = \left[\, \xi_1[k]\ \xi_2[k] \cdots\ \xi_N[k] \,\right]^T \tag{11}$$

consists the closed curved line Ξ in the N dimensional space.

$$\Xi = \left[\, \boldsymbol{\xi}[1]\ \boldsymbol{\xi}[2] \cdots\ \boldsymbol{\xi}[m]\ \boldsymbol{\xi}[1] \,\right] \tag{12}$$

Figure 1 shows the simple example of $N = 2$. Two time sequences $\xi_1[k],\ \xi_2[k]$

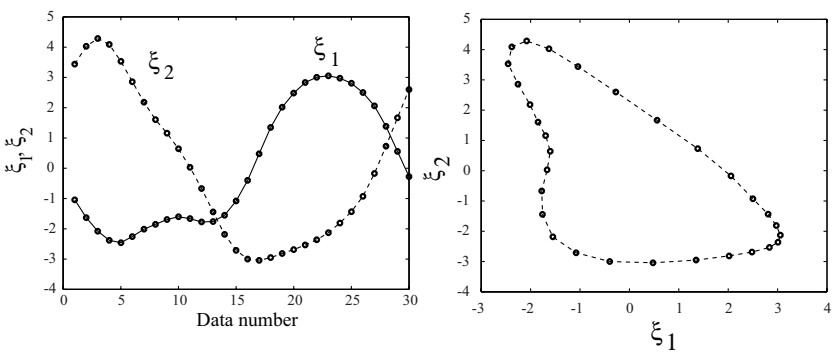


Fig. 1. Time sequence data and closed curved line

construct the closed curved line in the 2 dimensional space. In this section, we design a nonlinear dynamics $\mathcal{D}$ that has an attracter to this closed curved line with the following formulation.

$$\mathcal{D}\ :\ \boldsymbol{x}[k+1] = \boldsymbol{x}[k] + \boldsymbol{f}(\boldsymbol{x}[k]) \tag{13}$$

The design algorism is as follows.

Step 1 Draw the closed curved line C in equation (12) in the N dimensional space.

Step 2 Define the vector field $\boldsymbol{f}(\boldsymbol{\eta}_i)$ on the point $\boldsymbol{\eta}_i$ which is in domain D in the N dimensional space according to the following algorism (refer to figure 2).

$$\boldsymbol{f}(\boldsymbol{\eta}_i) = (\boldsymbol{\xi}_\eta[k+1] - \boldsymbol{\xi}_\eta[k]) + \boldsymbol{\gamma}_i[k] \tag{14}$$

$$\boldsymbol{\xi}_\eta[k] = \arg\min_{\boldsymbol{\xi}[k]} \|\boldsymbol{\eta}_i - \boldsymbol{\xi}[k]\| \tag{15}$$

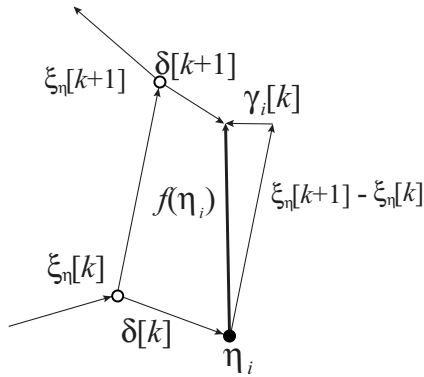


Fig. 2. Vector definition

Because $\boldsymbol{\eta}_i = \boldsymbol{\xi}_\eta[k] + \boldsymbol{\delta}[k]$, the sufficiency of that the closed curved line becomes the attractor is to satisfy the following inequarity.

$$\|\boldsymbol{\delta}[k] + \boldsymbol{\gamma}_i[k]\| < \|\boldsymbol{\delta}[k+1]\| \tag{16}$$

By satisfing this condition, $\boldsymbol{\delta}[k] \to 0$ at $k \to \infty$.

Step 3 On the points $\boldsymbol{\eta}_1,\ \boldsymbol{\eta}_2,\ \cdots,\ \boldsymbol{\eta}_m$ in the domain D, define the vectors $\boldsymbol{f}(\boldsymbol{\eta}_1),\ \boldsymbol{f}(\boldsymbol{\eta}_2),\ \cdots,\ \boldsymbol{f}(\boldsymbol{\eta}_m)$ shown in figure 3

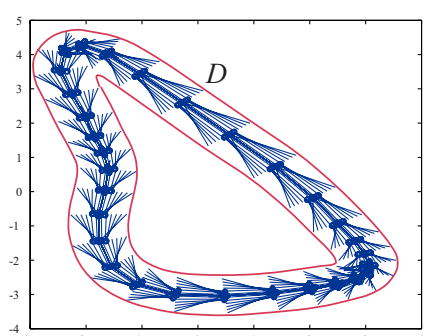


Fig. 3. Definition of the vector field

Step 4 Obtain the nonlinear function $\boldsymbol{f}(\boldsymbol{x}[k])$ that approximate $\boldsymbol{f}(\boldsymbol{\eta}_i)$ by using the polynomial approximation of $\boldsymbol{x}_i$ as follows.

$$\boldsymbol{f}(\boldsymbol{\eta}_i) = \sum_{P=0}^{l} \sum_{\substack{p_1,\cdots,p_n \\ \sum p_k = P \\ p_k : \text{positive integer}}} a_{(p_1 p_2 \cdots p_n)} \prod_{j=1}^{n} \eta_{ij}^{p_j} \tag{17}$$

$$\boldsymbol{\eta}_i = \begin{bmatrix} \eta_{i1} & \eta_{i2} & \cdots & \eta_{iN} \end{bmatrix}^T \tag{18}$$

a_{ij}^{pq} are constants. In the case of N=2 and ℓ=3, equation (17) is represented as follows.

$$\begin{aligned} \boldsymbol{f}(\boldsymbol{\eta}_i) = {} & a_{(30)}\eta_{i1}^3 + a_{(21)}\eta_{i1}^2\eta_{i2} + a_{(12)}\eta_{i1}\eta_{i2}^2 + a_{(03)}\eta_{i2}^3 \\ & + a_{(20)}\eta_{i1}^2 + a_{(11)}\eta_{i1}\eta_{i2} + a_{(02)}\eta_{i2}^2 \\ & + a_{(10)}\eta_{i1} + a_{(01)}\eta_{i2} + a_{(00)} \end{aligned} \tag{19}$$

It is easy to calculate $\boldsymbol{f}(\boldsymbol{x}[k])$ by the least square method as follows.

$$\Phi(a_{pq}^{ij}) = FH^{\#} \tag{20}$$

$$F = \begin{bmatrix} \boldsymbol{f}(\boldsymbol{\eta}_1) & \boldsymbol{f}(\boldsymbol{\eta}_2) & \cdots & \boldsymbol{f}(\boldsymbol{\eta}_m) \end{bmatrix} \tag{21}$$

$$H = \begin{bmatrix} \eta_{11}^{\ell} & \eta_{21}^{\ell} & \cdots & \eta_{m1}^{\ell} \\ \vdots & & & \\ \eta_{1N}^{\ell} & \eta_{2N}^{\ell} & \cdots & \eta_{mN}^{\ell} \\ \eta_{11}^{\ell-1}\eta_{12} & \eta_{21}^{\ell-1}\eta_{22} & \cdots & \eta_{m1}^{\ell-1}\eta_{m2} \\ \vdots & & & \\ 1 & 1 & \cdots & 1 \end{bmatrix} \tag{22}$$

Φ is a constant matrix. The domain D is the basin, which means the state $\boldsymbol{x}[k]$ inside D is entrained to C at $k \to \infty$. If Φ gives a right approximation of the defined vector field, we obtain the nonlinear dynamics that has an attractor to the closed curved line C. Figure 4 shows the motion of the designed nonlinear dynamics. '+' means the initial condition of $\boldsymbol{x}[0]$. It is entrained to C. This result shows that the designed dynamics memorizes the time sequence data $\boldsymbol{\xi}[k]$ $(k = 1,\ 2,\ \cdots\ m)$ and generate the whole body motion.

4.2 Set of the input signal

The designed dynamics $\mathcal{D}$ in equation (13) moves autonomously. From the initial state condition $\boldsymbol{x}[0]$ in the domain D, it converges to C. In this section, we set the input signal to the designed dynamics. By the input signal, the nonlinear dynamics is entrained or detrained to the attractor, which means this dynamics has the following functions.

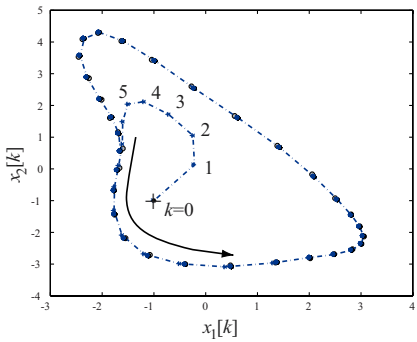


Fig. 4. Motion of the nonlinear dynamics

Association of whole body motion By observing a part of whole body motion (for example, motion of arm), the robot associate the rest of motions (for example, motion of leg and body).

Module component The dynamics-based information processing system becomes a mdule component with input and output signal, which enables to design the dynamics network.

Recognition of self motion By the comparison of input signal and its estimation, the robot recognizes the observed motion or self motion. This is discussed in the later section.

By adding another time sequence $\boldsymbol{u}[k]$ ($\in \boldsymbol{R}^L$, $k = 1, 2, \cdots, M$), the dynamics is represented as follows.

$$\widehat{\mathcal{D}} \ : \ \boldsymbol{x}[k+1] = \boldsymbol{x}[k] + \boldsymbol{g}\left(\boldsymbol{u}[k], \boldsymbol{x}[k]\right) \tag{23}$$

By changing Ξ in equation (12) as

$$\Xi = \begin{bmatrix} \boldsymbol{\mu}[1] & \boldsymbol{\mu}[2] & \cdots & \boldsymbol{\mu}[M] & \boldsymbol{\mu}[1] \\ \boldsymbol{\xi}[1] & \boldsymbol{\xi}[2] & \cdots & \boldsymbol{\xi}[M] & \boldsymbol{\xi}[1] \end{bmatrix} \tag{24}$$

$\boldsymbol{\mu}[k]$: given

$\boldsymbol{g}(\boldsymbol{u}[k], \boldsymbol{x}[k])$ is calculated by the same algorism as $\boldsymbol{f}(\boldsymbol{x}[k])$. Figure 5 shows the motion of the dynamics with 1 dimensional input $u[k]$. The input $u[k]$ is changed from one signal to $\mu[k]$ in time $k = \widehat{k}$. The dynamics is entrained to C after going around in the space according to the change of input. This result means that the input signal changes the structure of the nonlinear dynamics by changing the vector field $\boldsymbol{g}(\boldsymbol{u}[k], \boldsymbol{x}[k])$, and entrains the dynamics to C.

4.3 Dynamics with multi attractors

We modify the dynamics $\widehat{D}$ so that it has some attractors and transits to each attractors. The nonlinear dynamics is re-written as follows.

$$\widehat{\mathcal{D}}^w \ : \ \boldsymbol{x}[k+1] = \boldsymbol{x}[k] + w(\boldsymbol{x}[k])\boldsymbol{g}\left(\boldsymbol{u}[k], \boldsymbol{x}[k]\right) \tag{25}$$

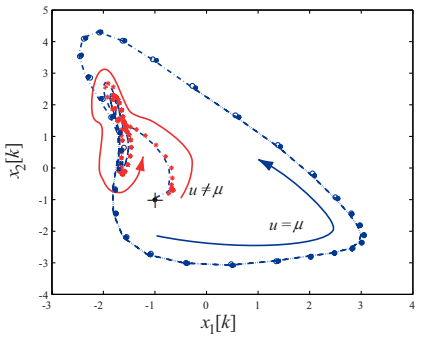


Fig. 5. Motion of the nonlinear dynamics with input signal

$w(\boldsymbol{x}[k])$ is the weighting function defined as follows.

$$w(\boldsymbol{x}[k]) = 1 - \frac{1}{1 + \exp\{a(\omega(\boldsymbol{x}[k]) - 1)\}} \tag{26}$$

$$\omega(\boldsymbol{x}[k]) = (\boldsymbol{x}^T[k] - X_0^T)Q(\boldsymbol{x}[k] - X_0) \tag{27}$$

a is a constant. Q and X_0 define the following ellipsoid E.

$$(\boldsymbol{x}^T[k] - X_0^T)Q(\boldsymbol{x}^T[k] - X_0) = 1 \tag{28}$$

X_0 is the center of the ellipsoid, Q is the positive definite matrix. Equation (26) means that if $\boldsymbol{x}[k]$ is inside the ellipsoid in equation (28), the weighting function is 1. Based on the dynamics $\widehat{\mathcal{D}}_i^w$ $(i = 1, 2, \cdots)$, we design the nonlinear dynamics $\widetilde{\mathcal{D}}$ which has some attractors as follows.

$$\widetilde{\mathcal{D}} \ : \ \boldsymbol{x}[k+1] = \boldsymbol{x}[k] + \sum_i w_i(\boldsymbol{x}[k])\boldsymbol{g}_i(\boldsymbol{u}[k], \boldsymbol{x}[k]) \tag{29}$$

This configuration means that the vector field that defines an attractor is effective only in the ellipsoid. One vector field defined by $\boldsymbol{g}_i(\boldsymbol{u}[k], \boldsymbol{x}[k])$ dose not have influence to other attractors. Because the vector fields is defined by the sum of $\boldsymbol{g}_i(\boldsymbol{u}[k], \boldsymbol{x}[k])$ surrounded by the ellipsoid E_i, it is easy to add the new attractor. The state $\boldsymbol{x}[k]$ moves to some attractors in term of the input signal $\boldsymbol{u}[k]$. Figure 6 shows the designed dynamics that has three attractors in the 2 dimensional space. By the 1 dimensional input $\boldsymbol{u}[k]$, the dynamics moves in the space attracted to the closed curved lines.

4.4 Recognition of self motion

The designed dynamics in equation (23) is modified as follows.

$$\bar{\mathcal{D}} \ : \ \begin{bmatrix} \widehat{\boldsymbol{u}}[k+1] \\ \boldsymbol{x}[k+1] \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}[k] \\ \boldsymbol{x}[k] \end{bmatrix} + \boldsymbol{h}(X[k]) \tag{30}$$

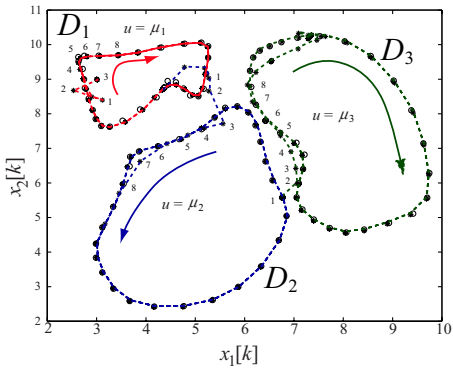


Fig. 6. Designed three attractors and dynamics

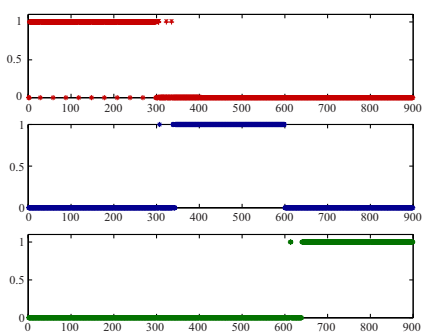


Fig. 7. Recognition of the input signal

The vector field $\boldsymbol{h}$ is calculate by the same ways as $\boldsymbol{f}$ and $\boldsymbol{g}$. Because $\widehat{\boldsymbol{u}}[k+1]$ is the prediction of the $\boldsymbol{u}[k+1]$, this dynamics recognizes which closed curved line the state vector is attracted by comparing $\boldsymbol{u}[k+1]$ and $\widehat{\boldsymbol{u}}[k+1]$. Figure 7 shows the recognition index of the attraction. The vertical axis is the recognition index (RI) that is defined

$$RI_i = \begin{cases} 1 \ (\|\widehat{\boldsymbol{u}}[k+1] - \boldsymbol{u}[k+1]\| \geq \alpha) \\ 0 \ (\|\widehat{\boldsymbol{u}}[k+1] - \boldsymbol{u}[k+1]\| < \alpha) \end{cases} \tag{31}$$

The upper figure, the middle figure and the lower figure show the recognition of $\boldsymbol{u}_i[k]$ (correspond to $\mathcal{D}_i$, $i = 1,\ 2,\ 3$) respectively. By checking RI_i, the humanoid recognizes which motion it takes, which means the recognition of self motion.

5 Generation of the Whole Body Motion

5.1 Whole body motion of the humanoid robot

In this section, we design the humanoid whole body motion using the dynamics based information processing system. Figure 8 shows the humanoid robot (FUJITSU Humanoid HOAP-1) that has 20 degree of freedom. We design

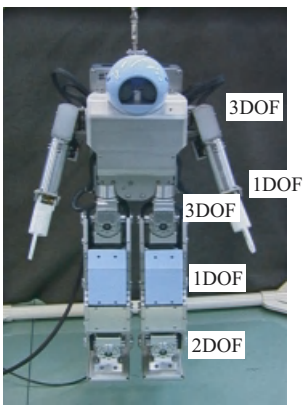


Fig. 8. Humanoid robot HOAP-1

the "walk" motion and "squat" motion. Figure 9 shows the original motion. This humanoid robot is not grounded. Because the dynamic based information processing system yields only the time sequence of the joint angle, the feedback controller that stabilizes each motions should be implemented.

5.2 Design of the dynamics

From the "walk" motion $Y_w \in \boldsymbol{R}^{20}$ and "squat" motion $Y_s \in \boldsymbol{R}^{20}$, we obtain the reduced motions $V_w^T \in \boldsymbol{R}^3$, $V_s^T \in \boldsymbol{R}^3$ in three dimensional space.

Based on the reduced motion, we design the dynamics based on the following equation

$$\boldsymbol{x}[k+1] = \boldsymbol{x}[k] + \sum_{i=w,s} w_i(\boldsymbol{x}[k])\boldsymbol{f}_i(\boldsymbol{x}[k]) + \sum_{i=w,s} K_i O_i(\boldsymbol{x}[k]) \tag{32}$$

$$O_i(\boldsymbol{x}[k]) = \delta(X_i^c - \boldsymbol{x}[k]) \tag{33}$$

By changing K_w and K_s, the humanoid transits its motion. δ is constant. X_w^c and X_s^c mean the center of reduced closed curved line "walk" and "squat"

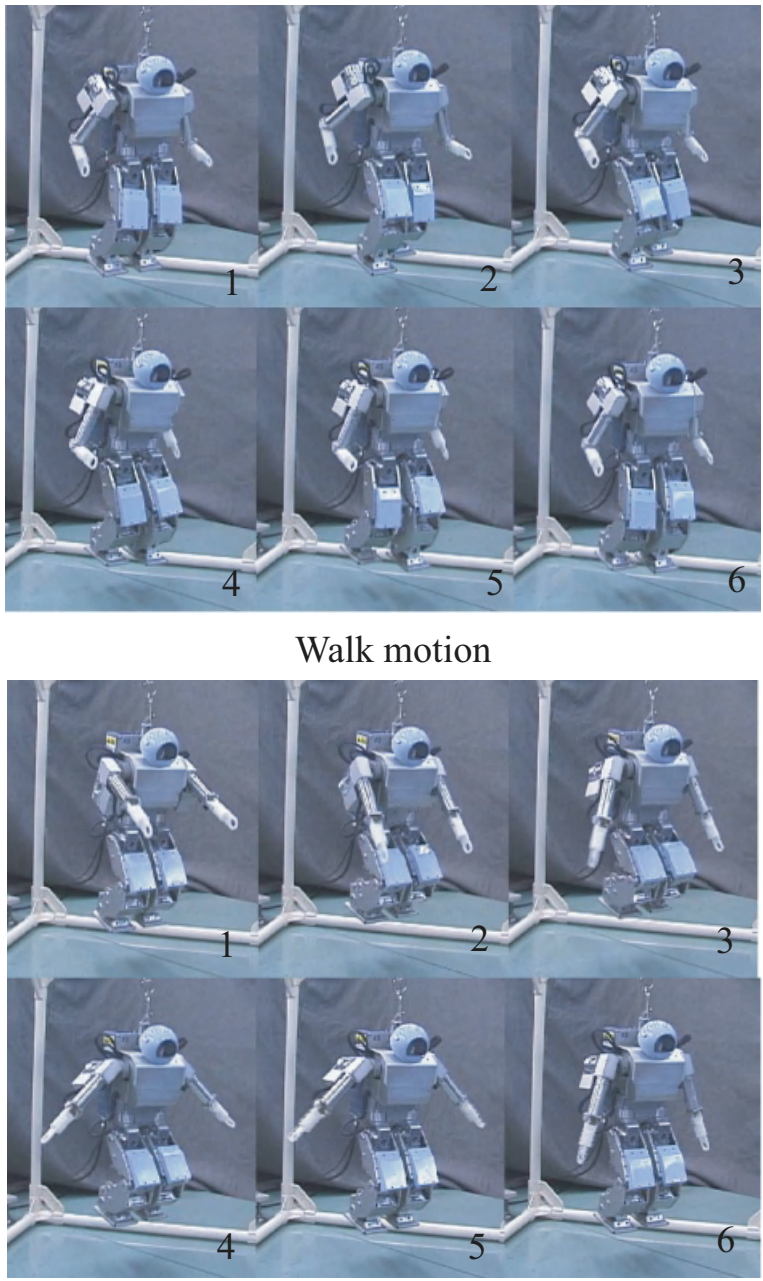


Fig. 9. Humanoid motion

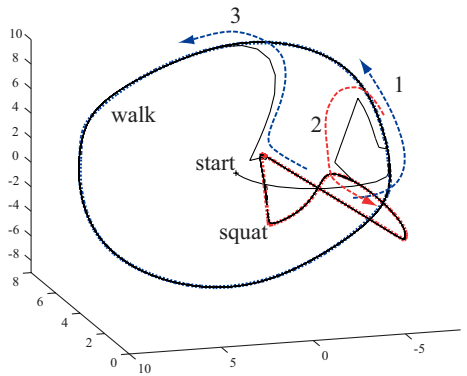


Fig. 10. Motion of the dynamics

respectively. Figure 10 shows the motion of the dynamics. From the initial position, the dynamics is entrained to the walk motion (arrow 1), entrained to squat motion (arrow 2) and finally entrained to walk motion again (arrow 3).

5.3 Motion of the humanoid robot

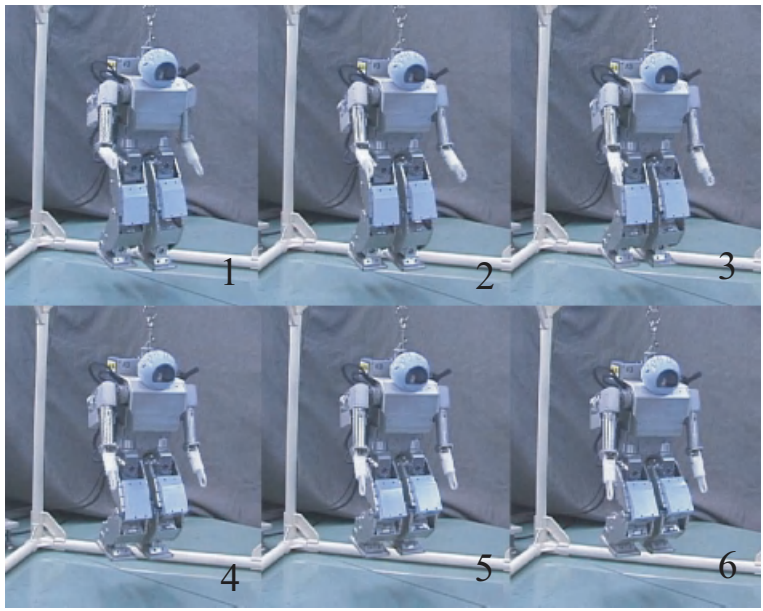
Figure 11 shows the generated humanoid motion. Because while the dynamics is attracted to the closed curved line, the humanoid motion is as same as figure 9, and only the transition motions are shown. The continuous transition is generated.

6 Conclusion

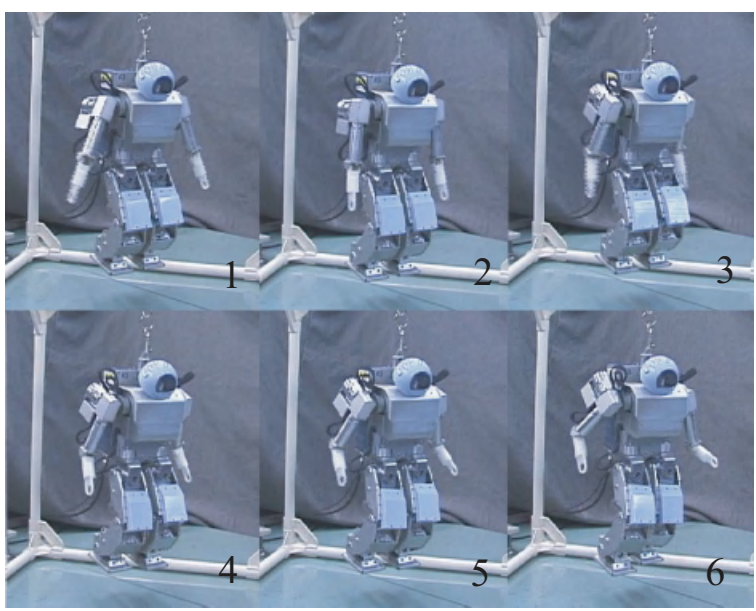
In this paper, we propose the motion reduction method and the brain-like information processing that realizes the memorization and generation of the humanoid whole body motion using the nonlinear dynamics with the polynomial configuration. The results of this paper are as follows.

(i) We propose the motion reduction method using the principal component analysis based on singular value decomposition.
(ii) We propose the design method of the nonlinear dynamics that has an attractor to the N dimensional closed curved line with polynomial configuration.
(iii) Using the proposed method, the whole body humanoid motion is generated.

The proposed method is a design method of the nonlinear dynamics that has some attractors to any curved line in N dimensional space. The



Walk to squat



Squat to walk

Fig. 11. Motion transintion of the humanoid robot

hierarchical structure and dynamics network gives a new approach to the robot intelligence. How to design and how to connect the network of the dynamics are future problems.

Acknowledgments.
This research is supported by the "Robot Brain Project" under the Core Research for Evolutional Science and Technology (CREST program) of the Japan Science and Technology Corporation. We use the C-language library developed by Mr. Sugihara (in Univ. of Tokyo) for the creation of the humanoid robot motion.

References

1. Brooks R.A. (1991) How to build complete creatures rather than isolated cognitive simulators, Architectures for Intelligence (K. VanLehn (ed.)), pp. 225-239
2. Freeman W.J. and Schneider W. (1982) Changes in Spatial Patterns of Rabbit Olfactory EEG with Conditioning to Odors, Psychophysiology, Vol.19, pp.44–56
3. Freeman W.J. (1987) Simulation of Chaotic EEG Patterns, Nonlinear Dynamic model of the Olfactory Systems, Biological Cybanetics, Vol.56, pp.139–150
4. Yao Y. and Freeman W.J. (1990) Model of Biological Pattern Recognition with Spatially Chaotic Dynamics, Neural Networks, Vol.3, pp.153–160
5. Sekiguchi A. and Nakamura Y. (2000) The Chaotic Mobile Robot, Proc. of Systemics, Cyberenetics and Informatics 2000, Vol.9, pp.463–468
6. Sekiguchi A. and Nakamura Y. (2001) Behavior Control of Robot Using Orbits of Nonlinear Dynamics, Proc. of IEEE International Conference on Robotics and Automation, pp.1647–1652
7. Hirai K. (1971) Inverse stability problem and its applications, Int. J. Control, Vol.13, No.6, pp.1073–1081
8. Hirai K. and Chinen H. (1982) A synthesis of the nonlinear discrete-time system having a periodic solution, IEEE Trans. on Circuits and Systems, Vol.CAS-29, No.8, pp.574–577
9. Green D. (1984) Synthesis of systems with periodic solution satisfying $V(x) = 0$, IEEE Trans. on Circuits and systems, Vol.CAS-31, No.4, pp.317–326
10. Moore B.C. (1981) Principal Component Analysis in Linear Systems: Controllability, Observability and Model Reduction, IEEE Trans., AC-37, No.1, pp17–32

Actuation Methods For Human-Centered Robotics and Associated Control Challenges

Michael Zinn[1], Oussama Khatib[2], Bernard Roth[1], and J. Kenneth Salisbury[2]

[1] Department of Mechanical Engineering, Design Division
[2] Department of Computer Science, Robotics Laboratory
Stanford University, Stanford CA, USA

Abstract. In recent years, many successful robotic manipulator designs have been introduced. However, there remains the challenge of designing a manipulator that possesses the inherent safety characteristics necessary for human-centered robotics. In this paper, we describe recent developments in low impedance actuation that have allowed for improvements in the safety characteristics of human-centered manipulators. In addition, the control challenges unique to the use of low impedance actuation are discussed along with possible control strategies for their successful implementation.

1 Introduction

In recent years, there has been great interest generated in the emerging field of human-centered robotics. Human-centered robotics involves the close interaction between robotic manipulation systems and human beings, including direct human-manipulator contact. In such applications, traditional figures of merit such as bandwidth, maximum force and torque capability, and reachable workspace, do not fully encompass the range of metrics which define the requirements of such systems. Specifically, human-centered robotic systems must consider the requirements of safety in addition to the the traditional metrics of performance. The question arises as to whether it is possible to successfully integrate the competing requirements of safety and performance in a single system. To answer this question we must first understand why some robotic systems are unsafe and, alternatively, why some systems have low performance.

1.1 Why Are Some Manipulators Unsafe?

Manipulator safety is dependent on a manipulator's mechanical, electrical, and software design characteristics. However, the biggest danger present when working in close proximity with robotic manipulators is the potential for large impact loads resulting from the large effective inertia (or more generally effective impedance) of many robotic manipulators. The addition of a compliant

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 105–119, 2003

covering can reduce impact loading by an order of magnitude or more. However, the amount of compliant material required to reduce loads to a safe level can be substantial and does not address the root cause of high impact loads - namely the large effective impedance of most modern robotic manipulators [1]. If inherent safety is to be achieved, we must design manipulators that have naturally low impedance. Unfortunately, most modern robotic manipulators have high effective impedance stemming from their requirements for high performance. The payload requirements and high bandwidth control necessitate the use of high inertia gear-head actuators and stiff, bulky structure which drive up the weight and impedance of these systems to unsafe levels.

1.2 Why Do Some Manipulators Have Low Performance?

Some types of robotic manipulators, notably those utilizing compliant actuation, such as pneumatic actuators, or those employing compliant drive trains, such as a cable driven manipulators, do not produce the large impact loads associated with high impedance designs. We can understand this by examining a simple mass-spring model of an actuator-link system with drive train compliance (see Fig. 1a). At low frequencies, the effective impedance

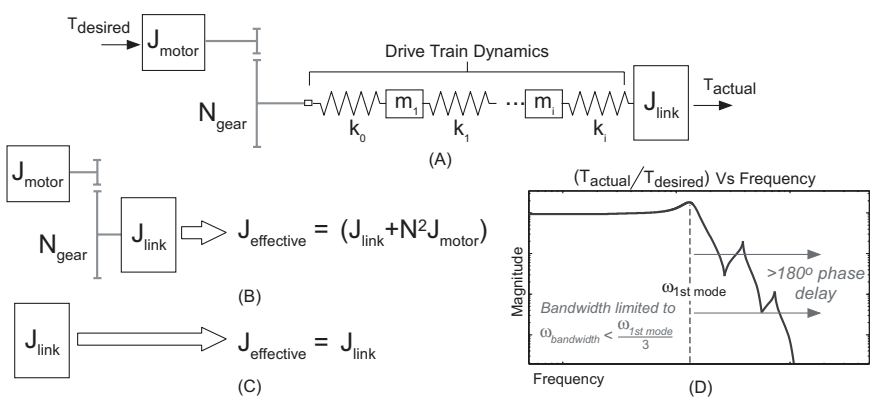


Fig. 1. (**a**) Robotic manipulator compliant drive-train mass-spring model (**b**) Low frequency effective inertia approximation (**c**) High frequency effective inertia approximation (**d**) Open-loop T_{actual} /$T_{desired}$ magnitude vs frequency

at the link can be approximated as the sum of the link and reflected actuator impedance (see Fig. 1b). However, at high frequencies, which produce the bulk of impact load energy, the effective impedance is reduced to the link inertia only (see Fig. 1c). For many manipulator systems, the actuator reflected inertia, with the N^2 amplification due to gear reduction, is much larger than the link inertia. The attenuation of the actuator reflected iner-

tia through the compliant drive train can significantly reduced impact loads, improving safety characteristics.

While a compliant actuator or drive train can enhance safety characteristics, the performance of such systems is limited. The flexible modes of the compliant system prevents control bandwidths greater than about 1/3 of the fundamental resonant frequency. In addition, attenuation of flexible mode oscillations excited by disturbances can be difficult to achieve. This results from the phase delay introduced above the first mode frequency (see Fig. 1d). With the resonant frequencies of many cable driven manipulators in the range of 10 Hz or less, high performance control is difficult if not impossible.

1.3 Actuator Characteristics - Obstacle Toward Achieving Safety and Performance

So why is it so difficult to simultaneously achieve safety and performance characteristics in a single manipulator design? The limitations of current actuation technology and the manner in which these actuators are implemented in manipulator designs are to blame. To understand why, we must examine the characteristics of existing actuation technology.

Currently, only electro-magnetic, hydraulic, and pneumatic actuators have the power and torque capabilities required for robotic manipulation tasks. Unfortunately, all of these actuation methods have serious deficiencies, limiting their inherent safety and/or performance characteristics.

Hydraulic actuators, which have the highest torque and power density characteristics of any of the actuation methods, are capable of performing tasks which involve the application of thousands of Newton-meters of torque and many kilowatts of power output. However, their very high output stiffness characteristics, which make the hydraulic actuator essentially a pure position source, can render it very dangerous. The output impedance, as compared to the driven manipulator and environment, is virtually infinite, generating very high impact loads during collisions. Typically these actuators are employed at the joint or through a rigid linkage further increasing the effective inertia of the manipulator. Thus, manipulators that employ hydraulic actuators have very poor inherent safety characteristics.

Pneumatic actuators on the other hand can be made very compliant. Due to the near zero inductance of the compressible gas, their output impedance is low over a wide frequency range, reducing uncontrolled impact loads to potentially safe levels. However, pneumatic actuators have very low bandwidth capabilities. Even when pressure control is implemented (as opposed to conventional flow control), control bandwidths are limited to less than 20 Hz which is insufficient for high performance tasks [2]. Making matters worse, the slow bandwidth capabilities render the large amount of stored potential energy in the compressible gas a serious hazard. Thus, while the natural compliance of pneumatic actuation reduces its effective inertia, its low band-

width characteristics limit the performance characteristics of manipulators which use them for the same reasons described in section 1.2.

Primarily as a result of the limitations of pneumatic and hydraulic actuators, many current human-centered research efforts use manipulation devices that employ electromagnetic actuation as their primary torque source. The primary limitation of electromagnetic motors is their relatively low torque and power density. The use of electromagnetic motors without a torque magnifying reducer is limited to direct drive systems which must employ large DC torque motors which are heavy and inefficient. To increase the torque output to useful levels, gear reducers are almost universally employed when using electromagnetic actuators. Unfortunately, the increase in torque and power density that results must be traded off against the large increase in reflected inertia which increases with the square of the gear reduction. Reduction ratios employed in most systems more than double the effective inertia of the manipulator, trading off safety for improved performance.

2 New Actuation Approaches

Recently, new actuation approaches have been developed to overcome the safety and performance limitations of existing systems. Chief among these are the joint torque control approach [3,4], series elastic actuation [5,6], and more recently, parallel-distributed actuation [1].

2.1 Joint Torque Controlled Actuation

Joint torque control was developed to eliminate the deleterious effects of nonlinearities and friction inherent in the actuator-transmission systems generally found in industrial robots. Initial implementations were successful in substantially reducing joint friction effects but wide joint actuation bandwidth was difficult to achieve without actually reducing the friction and nonlinearities in the actuator-transmission system [3].

In response, joint torque control systems employ high performance actuator and transmission designs with integrated torque sensors to achieve the performance levels desired. Perhaps the most successful of these has been the new DLR lightweight arm design [4]. The implementation of joint torque control allows for near zero low frequency impedance, which gives the DLR arm excellent force control characteristics. However, above the control bandwidth, joint torque control is ineffective at reducing the impedance of the manipulator. The open loop characteristics of the manipulator and reflected actuator inertia dominate. Thus, the magnitude of impact loads, which are determined by the high frequency impedance of the contacting surfaces, are not attenuated.

While the joint torque control has been successful in improving the force and impedance control of robotic manipulators, their fundamental open-loop

characteristics make inherent safety difficult to achieve and thus do not satisfy the human-centered robotic requirements of both performance and safety.

2.2 Series Elastic Actuation

Recently a class of actuators, known as series elastic actuators (SEA), has been developed to address the problems of high impedance actuators [5,6]. The SEA approach seeks to mitigate the limitations of high impedance actuators, such as conventional gear-head electromagnetic or hydraulic actuators, by placing an elastic element between the output of the actuator and the robotic link. The elastic element limits the high frequency impedance of the actuator to the stiffness of the elastic coupling. To limit the low frequency impedance, and thus transform the actuator into an approximate pure torque source, a linear feedback system is implemented to regulate the output torque of the actuator-spring system. (see Fig. 2).

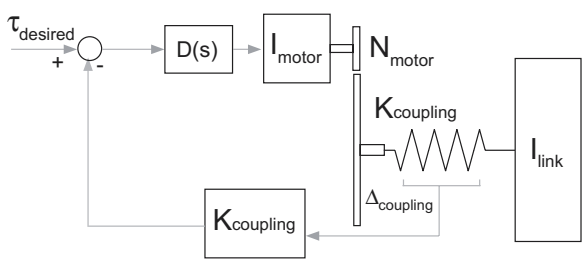


Fig. 2. Series elastic actuation (SEA) topology

The main advantage of the SEA topology is that it provides low output impedance across the frequency spectrum. As shown in [5,6], the SEA topology reduces the output impedance of the SEA actuator in proportion with the stiffness of the elastic coupling (1). At frequencies below the closed loop bandwidth of the SEA controller, the output impedance is reduced as a function of the control gains. Impedance reduction of 10x-100x is common and is only limited by the maximum obtainable bandwidth. At frequencies above the closed loop bandwidth, the output impedance reduces to the stiffness of the elastic coupling.

$$\frac{F(s)}{X(s)} = \frac{s^2(N_{motor})^2 I_{motor}}{\frac{s^2(N_{motor})^2 I_{motor}}{K_{coupling}} + 1 + N_{motor} D(s)} \tag{1}$$

This is in contrast to other approaches, such as joint torque control discussed in section 2.1, which have good low frequency impedance but suffer from large high frequency impedance.

There are trade offs with using the SEA actuators. Due to velocity and torque saturation of the SEA actuator, the maximum output torque above the open loop mode of the system[1] falls off as $1/\omega$ regardless of the control loop controller bandwidth [6]. This behavior is an open loop characteristic of the SEA actuator topology and represents a fundamental physical limitation of the actuator. The choice of the elastic coupling stiffness (in relation to the manipulator and motor reflected inertia) determines the open loop mode frequency. A stiffer coupling improves the high frequency torque performance but adversely affects the desirable closed and open loop impedance characteristics.

The use of a compliant coupling and the closed loop control of the SEA output torque limits the bandwidth of any task which relies on a series elastic actuator as its only torque source. This limitation derives from the use of the SEA closed loop system within a larger, task-specific control loop. As shown in Fig. 3, the design and resulting stability of the task-specific control loop is dependent on the interaction between the inner SEA closed loop system and the outer task-specific control loop. If the outer loop bandwidth approaches the bandwidth of the inner loop, instability is likely to occur. As a result, the task specific control loop cannot be closed at a rate faster than the inner loop.

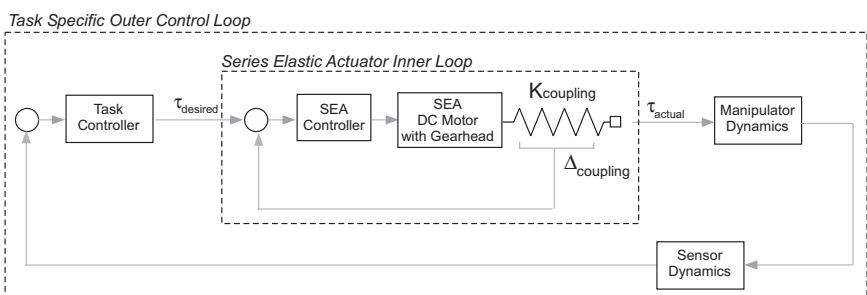


Fig. 3. Nested series elastic actuation and outer task control loops

Tasks such as position control and end-effector impedance control are limited to a bandwidth that is significantly below the closed loop bandwidth of the SEA actuator. This is not a major consideration for manipulation systems which do not require fast dynamics such as walking robots for which the series elastic actuators were originally developed. However, for tasks requiring high bandwidth control such as high speed trajectory tracking or high frequency disturbance rejection, the limitations of the series elastic actuators are prohibitive.

[1] SEA open loop mode: unforced coupled motion of actuator and manipulator link inertias through the compliant coupling

2.3 Parallel-Distributed Actuation

Recently, a new actuation approach, referred to as parallel-distributed actuation, has been developed to overcome the safety limitations of joint torque control and the performance limitations of series elastic actuation [1]. As the name implies, the parallel-distributed approach employs a pair of actuators, connected in parallel and distributed to different locations on the manipulator. The overall approach is shown in Fig. 4

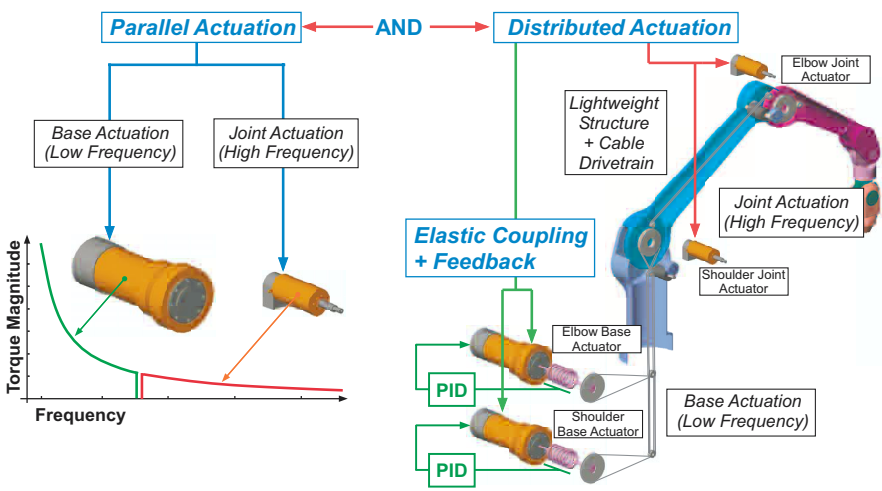


Fig. 4. Parallel-Distributed actuation approach

The first part of the parallel-distributed actuation approach is to divide the torque generation into separate low and high frequency actuators whose torque sum in parallel. The effectiveness of this approach can be seen clearly when one considers that most manipulation tasks involve position or force control which are dominated by low frequency trajectory tracking or DC load torques. High frequency torques are almost exclusively used for disturbance rejection. Even haptic device torque profiles, which might require rapid changes approximating a square wave input, have a torque magnitude versus frequency curve that falls off with increasing frequency by $1/\omega$. This partition is even more compelling when one considers power requirements vs frequency. Using the square wave example above, power versus frequency falls off with $1/\omega^2$. This power versus frequency profile is ideally fit using a large output, low frequency actuator coupled with a high frequency small torque motor.

In order for the parallel-distributed approach to work properly, both the high and low frequency actuators must have zero or near zero impedance. This is due to the fact that during power transfer the actuator torques will add non-destructively only if their respective impedance is zero. In particular, each actuator must not have significant impedance within the fre-

quency range of the opposing actuator. Only if this condition is true will the parallel-distributed concept work. For the high frequency actuation, very low impedance is achieved by using a small low inertia torque motor connected to the manipulator through a low friction, low reduction cable transmission. For the low frequency actuation, we achieve low impedance by using a series elastic actuator (see section 2.2). Because the parallel-distributed approach does not require that the base actuator be capable of supplying high frequency torques, the bandwidth limitations of SEA actuators do not pose a difficulty.

The second part of the parallel-distributed actuation approach, which differs from previous attempts at coupled actuation [7], is to distribute the low and high frequency actuators to locations on the manipulator where their effect on contact impedance is minimized while their contribution to control bandwidth is maximized. This is achieved by locating the low frequency series elastic actuator remotely from the actuated joint. This is particularly advantageous as the low frequency components of most manipulation tasks are considerably larger in magnitude than the high frequency components and consequently require a relatively large actuator. Locating the large SEA actuator at the base significantly reduces the weight and inertia of the manipulator. The high frequency actuators are located at the manipulator joints and connected through a stiff, low friction transmission, providing the high frequency torque components that the low frequency base actuators cannot. The high frequency torque actuator must be connected to the joint inertia through a connection which produces a high primary mode vibration frequency. By locating the actuator at the joint and by using a low inertia servomotor, we can achieve this high bandwidth connection with a minimum amount of weight and complexity.

Preliminary experimental and simulation results have demonstrated the effectiveness of the parallel-distributed approach. The reduction in impact loading by an order of magnitude, as compared to conventional joint actuated manipulators, substantially improves the inherent safety of the manipulator. In the case of a two-axis prototype developed at Stanford, the effective joint inertia was reduced by almost a factor of ten [1]. In combination with a light weight structure and compliant covering, this new actuation approach can be used to design a manipulator that reduces impact loads substantially, thus ensuring inherent safety.

In addition to safety, the parallel-distributed approach, with the introduction of the high frequency joint actuator, has been shown experimentally to improve manipulator performance. Initial experiments demonstrated a position control bandwidth of approximately 5 Hz as compared to a 2 Hz bandwidth using the base series elastic actuator alone (see Fig. 5), reducing the position tracking error by more than a factor of ten. Further improvements in performance are expected, as the primary limitation of our two axis testbed was structural resonance in the supporting test stand, which was not a func-

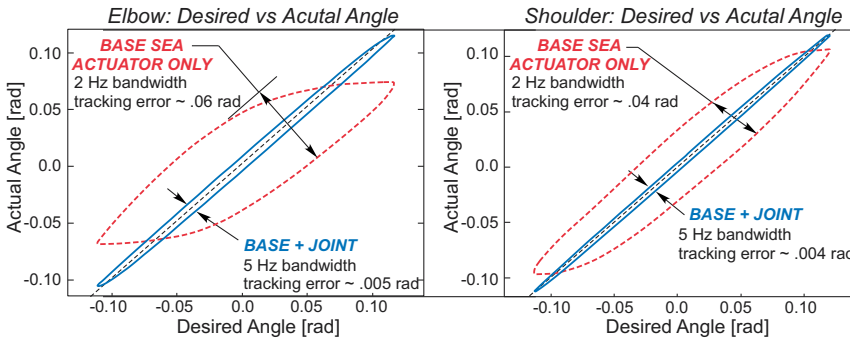


Fig. 5. Position tracking performance: Comparison of series elastic actuation to parallel-distributed actuation

tion of the actuation concept. However, some the the performance limitations of the parallel-distributed approach can be attributed to unique control challenges associated with implementing low impedance actuation in general. A discussion of the control approach and implementation issues is discussed in the following section.

Parallel-Distributed Actuation Control Approach Our control approach seeks to exploit the parallel-distributed actuation's unique characteristics to construct a near perfect torque source. The characteristics of a perfect torque source, consisting of zero output impedance and infinite control bandwidth, would enable a manipulator to possess the characteristics necessary for both inherent safety and high performance tasks. While a perfect torque source is impossible to achieve, a near perfect torque source, with low output impedance relative to the driving load and high bandwidth torque capability offers much of the same advantages.

Near Perfect Torque Source The control structure, shown in Fig. 6a, utilizes the low frequency base actuator's low pass filter characteristics to partition the control torques into low and high frequency components. By using the actual measured torque output from the low frequency base actuators in combination with the desired torque, we automatically compensate for the non-ideal behavior of the base actuators. Assuming that the smaller joint actuators can produce this torque, the combined torques sum is a perfect realization of the desired torque. The frequency partitioning can be clearly seen if we rearrange the structure in Fig. 6a into a pure parallel structure, as shown in Fig. 6b. As seen in Fig. 6b, the equivalent base actuator falls off at high frequency while the equivalent joint actuator approximates a double lead filter, which adds phase to the combined system and attenuates the DC and low frequency components commanded to the high frequency actuator.

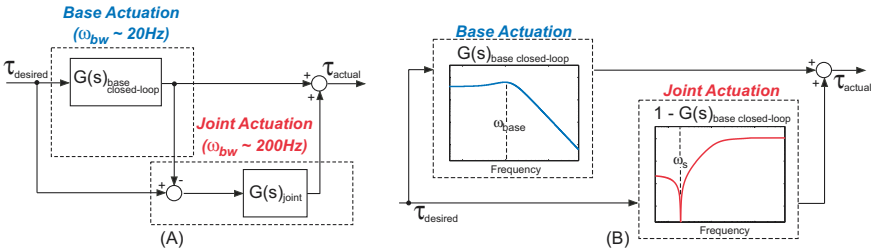


Fig. 6. (**a**) Parallel-distributed actuation control structure (**b**) Equivalent parallel structure

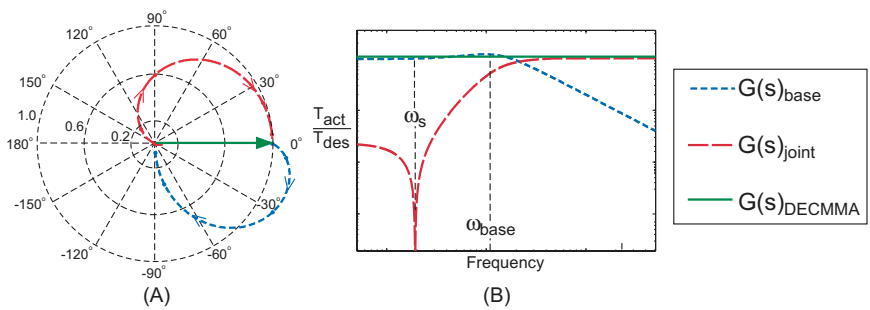


Fig. 7. (**a**) Perfect torque source: Base, joint, and combined parallel-distributed actuator torque magnitude vs phase polar plot (**b**) Near perfect torque source: Base, joint, and combined parallel-distributed actuator torque magnitude vs frequency

The combined actuator control structure creates a perfect torque source in the linear sense, where the torques sum to unity magnitude and zero phase, up to the first resonance mode frequency (ω_{joint}) as seen in Fig. 7a and 7b.

Effects Of Saturation Parallel-distributed actuator torque saturation represents the threshold above which the joint actuator can no longer compensate for the phase and magnitude error of the low frequency base actuator. Commanded torques which force the high frequency joint actuator to saturate will cause both magnitude errors and phase lag to occur, invalidating the perfect torque source characteristics of the combined parallel actuation. This effect is illustrated in Fig. 8a and 8b. In Fig. 8a and 8b, the frequency response of the base series elastic actuator, the joint actuator, and the combined parallel actuator is shown on a polar plot of magnitude versus frequency (Fig. 8a) and as a bode plot (Fig. 8b). The effect of saturation can be seen as both magnitude and phase errors in the resulting parallel actuation response. As the joint actuator approaches complete saturation, the combined parallel actuator's response approaches that of the single base series elastic actuator with its lower bandwidth constraints. This ia particularly problematic in that a task control loop, such as position tracking, which under normal conditions is

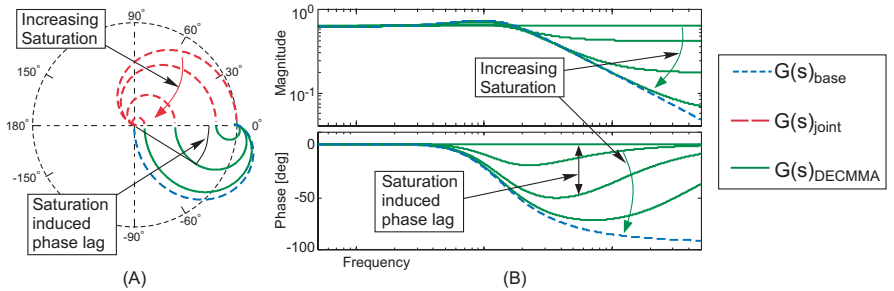


Fig. 8. Breakdown of perfect torque source due to saturation (**a**) Base, joint (with saturation), and resulting parallel-distributed actuator torque magnitude vs phase polar plot (**b**) Bode plot of parallel-distributed actuator torque with joint actuator saturation

stable, can become unstable as a result of a torque command which exceeds the capabilities of the smaller joint actuator.

Effects Of Large Inertia Mismatch. For systems employing low impedance actuation, such as parallel-distributed actuation, the ratio of actuator reflected inertia to driven link inertia is typically 1:10 or less. These systems can have a problem, sometimes referred to as peaking [8], which occurs when position or velocity feedback is introduced. We can understand this by examining the open-loop transfer function of a simple mass-spring model of an actuator-link system. Fig. 9a and (2a) show the assumed model and its uncompensated open-loop transfer function. In many servo-systems, includ-

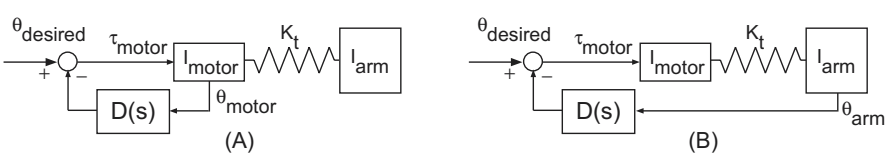


Fig. 9. (Spring-mass model of actuator and driven link Inertias (**a**) Collocated control (**b**) Non-collocated control

ing robotics, the actuator and link inertias are matched or nearly matched to achieve optimum power and acceleration transfer from motor to load. In this situation, the poles and zeros of the transfer function, given by (2c), are approximately equal in frequency.

$$\frac{\theta_{motor}(s)}{\tau_{motor}(s)} = \frac{s^2 I_{arm} + K_t}{s^2(s^2 I_{arm} I_{motor} + K_t(I_{arm} + I_{motor}))} \tag{2a}$$

$$\frac{\theta_{arm}(s)}{\tau_{motor}(s)} = \frac{K_t}{s^2(s^2 I_{arm} I_{motor} + K_t(I_{arm} + I_{motor}))} \tag{2b}$$

$$\omega_{zero} = \sqrt{\frac{K_t}{I_{arm}}} \quad \text{and} \quad \omega_{pole} = \sqrt{\frac{K_t(I_{motor} + I_{arm})}{I_{arm} I_{motor}}} \tag{2c}$$

However, in a system employing low impedance actuation, the zero's frequency can be an order of magnitude below the frequency of the flexible mode pole. This large separation amplifies the flexible mode peak by a factor approximately equal to the ratio of drive link to motor inertias (see Fig. 10). This effect severely limits the the achievable closed loop bandwidth and thus

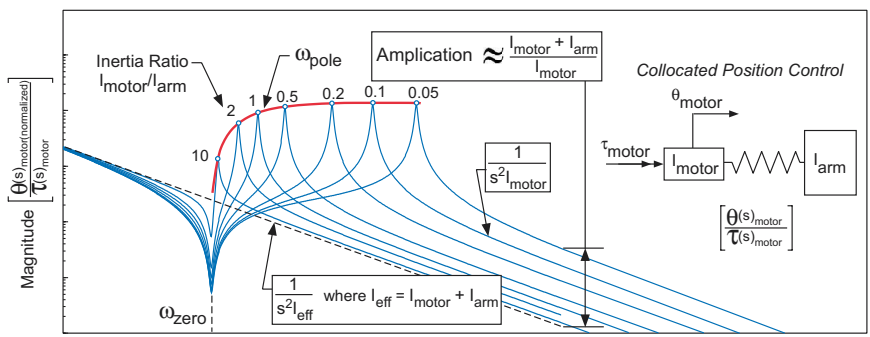


Fig. 10. Open loop transfer function of collocated motor position control: Amplification of oscillatory pole due to mismatched actuator-link inertia

performance in general. The effect can be quite puzzling considering that the flexible mode frequency can be very high - an order of magnitude or more above the open loop crossover frequency - and still cause excessive oscillations in the closed loop response. Only when one considers the zero, whose frequency is affected by the much larger drive link inertia, does it become clear why the problem exists.

Another way to analyze the problem is to examine the root locus of the system shown in Fig. 9a. When the inertia ratio, I_{motor}/I_{arm}, is close to 1:1, the oscillatory poles are drawn toward the transmission zeros as the gain is increased, reducing their residues which reduces the magnitude of oscillations and allows for larger closed loop gains. However, when the motor inertia, I_{motor}, is much less than the arm inertia, I_{arm}, the transmission zeros are located too far from the oscillatory poles to have a stabilizing effect and instead attract the dominant second order poles. This phenomenon can be clearly seen if we look at the symmetric root locus for the transfer function

in (2a). As seen in Fig. 11, when the motor inertia, I_{motor}, is smaller than

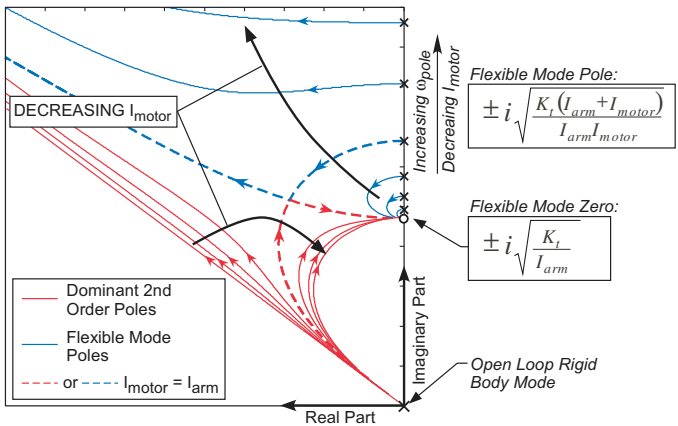


Fig. 11. (Symmetric root locus of collocated position control system with shaft compliance

the arm inertia, I_{arm}, the optimal control gains drive the dominant poles toward the zeros, indicating that a large amount of control effort would be required to modify the system behavior away from the low frequency zeros. As a result, achieving high bandwidth closed loop system is very difficult.

There a various methods available to mitigate the deleterious effect that low actuator inertia has on closed loop performance [8], including filtering and acceleration feedback. These methods allow for closed loop bandwidths which approach the frequency of the transmission zero. A complementary approach is to seek to modify the plant dynamics. Increasing the stiffness of the coupling between the motor inertia and link inertia will increase both the frequency of the oscillatory poles and the transmission zeros, allowing for a higher crossover frequency. In some cases, intentionally increasing the inertia of the motor can have a beneficial effect by reducing the frequency of the oscillatory poles to the frequency of the zeros. However, this approach is only useful when the motor and link inertias differ by less than approximately a factor of 2. Otherwise, the required increase in motor inertia is excessively large and severely reduces the acceleration capability of the system. Regardless, in the case of low impedance actuation, a large increase in actuator inertia would substantially increase the reflected inertia of the actuator, adversely effecting its safety characteristics and thus can not be considered for human-centered robotic systems.

Another, somewhat surprising method to deal with the peaking problem of low impedance actuation is to change the control topology from collocated to non-collocated control. We can understand this by examining the open-loop transfer function of a simple mass-spring model of an actuator-link system

which employs non-collocated control. Fig. 9b and (2b) show the assumed model and its associated transfer function. At first glance, this seems counter intuitive since in most cases the stabilizing effect of the zeros associated with collocated control are beneficial and allow for more aggressive gains. However, in the case of large inertia mismatch, the collocated control zero is the main cause of the problem. A comparison of peaking amplitude (see Fig. 12) shows that for large mismatches the non-collocated control may be better than a collocated approach. Of course, this doesn't take into account the tendency of the oscillatory poles to become unstable, and special care must be taken to insure their stability, such as using of a notch filter or a gain stabilizing lag network. With this consideration, we can conservatively assume that when using non-collocated control we can achieve a cross-over frequency as high as 1/5 of the flexible mode frequency. With this assumption, we can see from Fig. 12 that for inertia ratios above approximately $I_{arm}/I_{motor} > 10$ the use of non-collocated control allows for a higher closed loop bandwidth than collocated control. This, in fact, has been shown to be the case on a two axis testbed, where the motor-link inertia ratios range from 50:1 to movr than 100:1.

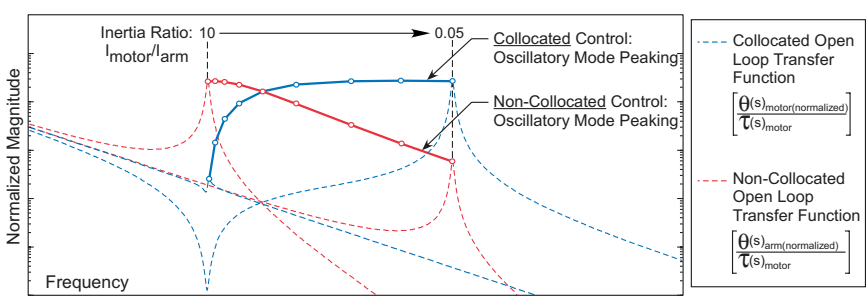


Fig. 12. Variation of peaking amplitude for collocated and non-collocated position control for varying motor to load inertia ratios, I_{motor}/I_{arm}

In addition to actuator saturation and oscillatory peaking, there are other challenges when implementing low impedance actuation. Low actuator and manipulator impedance make achieving high static stiffness and general disturbance rejection more difficult. In addition, the low friction characteristics associated with low impedance actuation can introduce unwanted limit cycling. These and the issues discussed above pose unique challenges when compared to their high impedance counterparts. While we have presented some useful approaches to address these problems there still exists much work to be done to properly utilize these systems.

3 Conclusion

To achieve inherent safety a manipulator must have low open-loop impedance to reduce uncontrolled impact loads to safe levels. Recent developments in low impedance actuation has allowed for improvements in the safety characteristics of human-centered manipulators. However, the use of low impedance actuation, and in particular the use of distributed actuation to augment high frequency characteristics, creates additional control challenges not normally encountered in traditional servo-systems. The effects of actuator saturation on system stability and the phenomenon of peaking are of particular concern given that they can adversely effect the stability and limit the achievable closed loop bandwidth. While some progress has been made in mitigating these effects additional efforts will be required to fully realize the benefits of low impedance actuation for human-center robotics.

Acknowledgments.
The authors would like to thank Gunter Neimeyer[1], Ken Waldron[1], and Gene Duval for their helpful insights and discussion in preparing this paper. The financial support of NSF grant EIA-9977717 is gratefully acknowledged.

References

1. Zinn, M., Khatib, O., Roth, B., Salisbury, J.K. (2002) A New Actuation Approach for Human Friendly Robot Design, Proceedings of Int. Symposium on Experimental Robotics, Sant'Angelo d'Ischia, Italy, 2002
2. Hollerbach, J., Hunter, I., Ballantyne, J. (1991) A Comparative Analysis of Actuator Technologies for Robotics, Robotics Review 2, 299–342, MIT Press 1991
3. Vischer, D., Khatib, O. (1995) Design and Development of High-Performance Torque-Controlled Joints,IEEE Trans on Robotics and Automation, v11, n4
4. Hirzinger, G., Albu-Schäffer, A., Hähnle, M., Schaefer, I., Sporer, N., (2001) On a New Generation of Torque Controlled Light-Weight Robots, ICRA, Seoul, Korea, May 21-26, 2001
5. Pratt, G., Williamson, M. (1995) Series Elastic Actuators, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 399–406
6. Robinson, D. (2000) Design and Analysis of Series Elasticity in Closed-loop Actuator Force Control, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts
7. Morrel, J.B. (1996)Parallel Coupled Micro-Macro Actuators, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts
8. Ellis, G., Lorenz, R.D., (2000) Resonant Load Control Methods for Industrial Servo Drives, Proceedings of IEEE IAS (Rome), 2000

Control of a Flexible Manipulator with Noncollocated Feedback: Time Domain Passivity Approach

Jee-Hwan Ryu[1], Dong-Soo Kwon[2], and Blake Hannaford[1]

[1] Department of Electrical Engineering University of Washington Box 352500, Seattle, WA 98195-2500
[2] Department of Mechanical Engineering Korea Advanced Institute of Science and Technology Daejon, Republic of Korea, 305-701

Abstract. A new method to control a flexible manipulator with noncollocated feedback is proposed. We introduce a method to implement the time domain passivity control approach to a flexible manipulator with noncollocated feedback, which could not be treated with the previous time domain passivity control framework due to a possibly active transfer function from the collocated output to the noncollocated output. The developed method is simulated with the model of a single link flexible manipulator and we obtained a good control performance.

1 Introduction

Flexible manipulators are finding their way into industrial and space robotics applications due to their lighter weight and faster response time compared to rigid manipulators. Control of flexible manipulators has been studied extensively for more than a decade by several researchers [2], [3], [8], [13], [15], [17]. Despite their results, control of flexible manipulators has proven to be rather complicated.

It is well known that stabilization of a flexible manipulator can be greatly simplified by collocating the sensors and the actuator, in which the input-output mapping is passive [18], and a stable controller can be easily devised independent of the structure details. However, the performance of this collocated feedback turns out to be not satisfactory due to a week control of the vibrations of the link [4]. This initiated finding other noncollocated output measurements like the position of the end-point of the link to increase the control performance [3]. However, if the end-point is chosen as the output and the joint torque is chosen as the input, the system becomes a nonminimum phase one, and may possibly behave actively. As a result, a small increment of feedback controller gains can easily make the closed-loop system unstable. This led many researchers to seek other outputs which have the passivity property.

Wang and Vidyasagar proposed the so-called reflected tip position as such an output [18]. This corresponds to the rigid body deflection minus the deflection at the tip of the flexible manipulator. Pota and Vidyasagar used

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 121–134, 2003

the same output to show that in the limit, for a non uniform link, the transfer function from the input torque to the derivative of the reflected tip position is passive whenever the ratio of the link inertia to the hub inertia is sufficiently small [9]. Chodavarapu and Spong considered the virtual angle of rotation, which consists of the hub angle of rotation augmented with a weighted value of the slope of the link at its tip [4]. They showed that the transfer function with this output is minimum phase and that the zero dynamics are stable.

Despite the fact that these previous efforts have succeeded in numerous kinds of applications, the critical drawback was that these are model-based approaches requiring the system parameters or the dynamic structure information at the least. However, interesting systems are uncertain and it is usually hard to obtain the exact dynamic parameters and structure information.

In this paper, we introduce a different way of treating noncollocated control systems without any model information. A recently developed stability guaranteed control method based on time-domain passivity control [6], [11] [12] is applied. First the new control method is reviewed, and then several issues for implementing with noncollocated control systems are studied. Performance of the proposed controller is then investigated through numerical simulations with a flexible manipulator.

2 Review of Time Domain Passivity Approach

2.1 Network Model

In our previous paper [12], the traditional control system view could be analyzed in terms of energy flow by representing it in a network point of view. Energy here was defined as the integral of the inner product between the conjugate input and output, which may or may not correspond to a physical energy. We partition the traditional control system into three elements, the trajectory generator, the control element (consisting of the controller, actuator and sensors) and the plant. The connection between the controller element and the plant is a physical interface at which suitable conjugate variables define the physical energy flow. The connection between trajectory generator and controller, which traditionally consists of a one-way command information flow, is modified by the addition of a virtual feedback of the conjugate variable. For a motion control system, the trajectory generator output would be a desired velocity (v_d), and the virtual feedback would be equal to the controller output (τ) (Fig. 1).

To show that this consideration is generally possible for motion control systems, we physically interpret these energy flows. We consider a general tracking control system with a position PID and feed forward controller for moving a mass (M) on the floor with a desired velocity (v_d). The control system can be described by a physical analogy with Fig. 2. The position PD

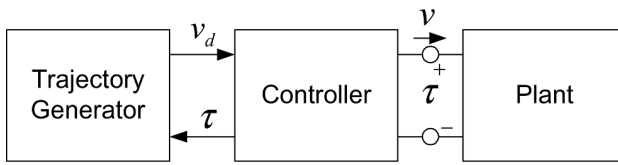


Fig. 1. Network view of a motion control system

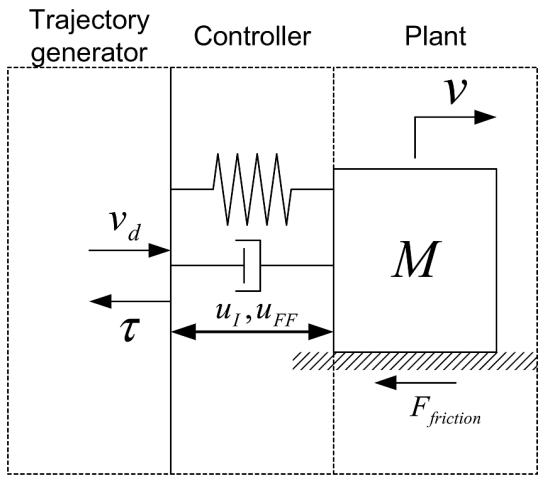


Fig. 2. Physical analogy of a motion control system

controller is physically equivalent to a virtual spring and damper whose reference position is moving with a desired velocity (v_d). In addition, the Integral Controller (u_I) and the feed forward controller (u_{FF}) can be regarded as internal force sources. Since the mass and the reference position are connected with the virtual spring and damper, we can obtain the desired motion of the mass by moving the reference position with the desired velocity. The important point is that if we want to move the reference position with the desired velocity (v_d), force is required. This force is determined by the impedance of the controller and the plant. Physically this force is equivalent to the controller (PID and feed forward) output (τ). As a result, the conjugate pair (v_d and τ) simulates the flow of virtual input energy from the trajectory generator, and the conjugate pair (v and τ) measures the flow of real output energy to the plant. Through the above physical interpretation, we can construct a network model for general tracking control systems (Fig. 1), and this network model is equivalently described with Fig. 3 whose trajectory generator is a current (or velocity) source with electrical-mechanical analogy. Note that electrical-mechanical analog networks enforce equivalent relationships between effort and flow. For the mechanical systems, forces replace voltages

in representing effort, while velocities representing currents in representing flow.

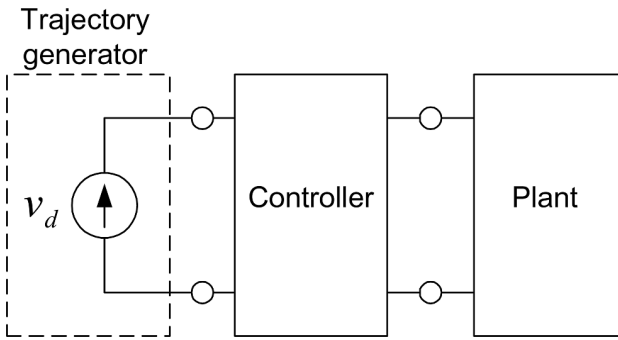


Fig. 3. Network view of general motion control systems

2.2 Stability Concept

From the circuit representation (Fig. 3), we find that the virtual input energy from the trajectory generator depends on the impedance of the connected controller and plant. We assume that the plant is passive. Thus if we can make the controller network passive, we can guarantee the stability of the system since passivity of each block is a sufficient condition for stability.

2.3 Time Domain Passivity Approach

In this section, we briefly review time-domain passivity control. First, we define the sign convention for all forces and velocities so that their product is positive when power enters the system port (Fig. 4). Also, the system is assumed to have initial stored energy at $t = 0$ of $E(0)$. The following widely known definition of passivity is used.

Definition 1 : The two-port network, N, with initial energy storage $E(0)$ is *passive* if and only if,

$$\int_0^t f_1(\tau)\dot{x}_1(\tau)+f_2(\tau)\dot{x}_2(\tau)d\tau+E(0) \geq 0, \quad \forall t \geq 0 \tag{1}$$

for forces (f_1, f_2) and velocities $(\dot{x}_1, \dot{x}_1)$. Eqn(1) states that the energy supplied to a passive network must be greater than negative $E(0)$ for all time [14], [1], [5], [16].

The conjugate variables that define power flow in such a computer system are discrete-time values, and the analysis is confined to systems having a

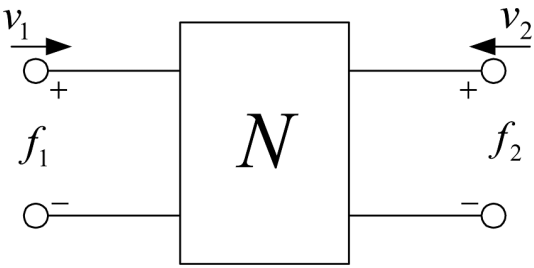


Fig. 4. Two-port network

sampling rate substantially faster than the dynamics of the system. We assume that there is no change in force and velocity during one sample time. Thus, we can easily "instrument" one or more blocks in the system with the following "Passivity Observer," (PO) for a two-port network to check the passivity (1).

$$E_{obsv}(n) = \Delta T \sum_{k=0}^{n} (f_1(k)v_1(k) + f_2(k)v_2(k)) + E(0) \tag{2}$$

where ΔT is the sampling period. If $E_{obsv}(n) \geq 0$ for every n , this means the system dissipates energy. If there is an instance when $E_{obsv}(n) < 0$, this means the system generates energy and the amount of generated energy is $-E_{obsv}(n)$. For the case when the sampling rate is not fast enough, recently, we propose more accurate time domain PO with removes the constant force or velocity assumption during one sampling time [10].

Consider a two-port system which may be active. Depending on operating conditions and the specifics of the two-port element's dynamics, the PO may or may not be negative at a particular time. However, if it is negative at any time, we know that the two-port may then be contributing to instability. Moreover, we know the exact amount of energy generated and we can design a time-varying element to dissipate only the required amount of energy. We call this element a "Passivity Controller" (PC). The PC takes the form of a dissipative element in a series or parallel configuration depending on the input causality [6].

For a 2-port network with impedence causality at each port, we can design two series PCs (Fig. 5) in real time as follows. Please see [11], [12] for more detail about two-port time domain passivity control approach.

(i) $v_1(n) = v_2(n)$ and $v_3(n) = v_4(n)$ are inputs.
(ii) $f_2(n)$ and $f_3(n)$ are the outputs of the system
(iii) $W(n) = W(n-1) + f_2(n)v_2(n) + f_3(n)v_3(n) + \alpha_1(n-1)v_2(n-1)^2 + \alpha_2(n-1)v_3(n-1)^2$ is the PO

Two series PC can be designed for several cases

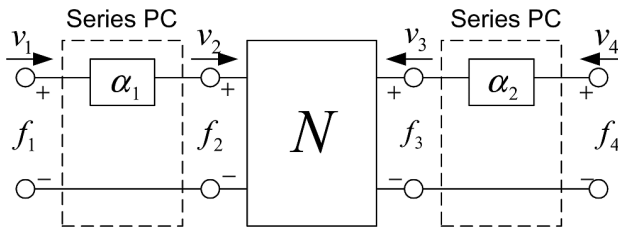


Fig. 5. Configuration of series PC for 2-port network

(iv)

$$\alpha_1(n) = \begin{cases} \frac{-W(n)}{v_2(n)^2}, & \text{if case 2, 4.2} \\ \frac{-f_2(n)v_2(n)}{v_2(n)^2}, & \text{if case 4.1} \\ 0 & \text{if case 1,3} \end{cases} \tag{3}$$

(v)

$$\alpha_2(n) = \begin{cases} \frac{-W(n)}{v_3(n)^2}, & \text{if case 3} \\ \frac{-(W(n-1)+f_3(n)v_3(n))}{v_3(n)^2}, & \text{if case 4.1} \\ 0 & \text{if case 1, 2, 4.2} \end{cases} \tag{4}$$

(vi) $f_1(n) = f_2(n) + \alpha_1 v_2(n) \Rightarrow output$
(vii) $f_4(n) = f_3(n) + \alpha_2 v_3(n) \Rightarrow output$

where each case is as follows
Case 1: energy does not flow out
$W(n) \geq 0$

Case 2: energy flows out from the left port
$W(n) < 0, f_2(n)v_2(n) < 0, f_3(n)v_3(n) \geq 0$

Case 3: energy flows out from the right port
$W(n) < 0, f_2(n)v_2(n) \geq 0, f_3(n)v_3(n) < 0$

Case 4: energy flows out from the both ports: as we mentioned above, in this paper, we divide it into two cases. The first case is when the produced energy from the right port is greater than the previously dissipated energy:

4.1 $W(n) < 0, f_2(n)v_2(n) < 0, f_3(n)v_3(n) < 0, W(n-1) + f_3(n)v_3(n) < 0$
in this case, we only have to dissipate the net generation energy of the right port as the second line in Eq. (4). The second case is when the produced energy from the right port is less than the previously dissipated energy:

4.2 $W(n) < 0, f_2(n)v_2(n) < 0, f_3(n)v_3(n) < 0, W(n-1) + f_3(n)v_3(n) \geq 0$ in this case we don't need to activate the right port PC, and also reduce the conservatism of the left port PC as the fist line of Eq. (3).

3 Implementation Issues

This section addresses how to implement the time domain passivity control approach to flexible manipulator with noncollocated feedback. Consider a single link flexible manipulator having a planar motion, as detailed in Fig. 6. v_e is the end-point velocity, v_a is the velocity of the actuating position, and τ is the control torque at the joint.

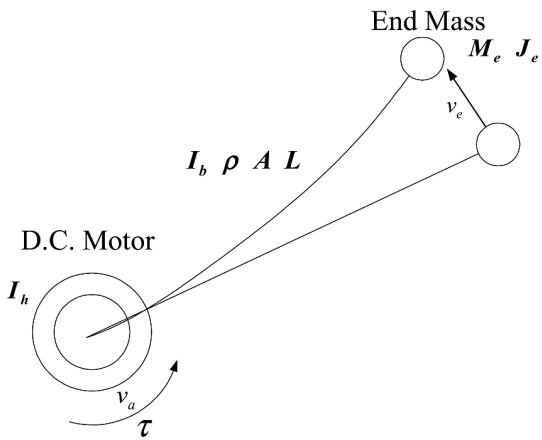


Fig. 6. A Single-link flexible manipulator

3.1 Network Modeling

When we feedback end-point position to control the motion of the flexible manipulator, a network model (including causality) of the overall control system can be drived (Fig. 7). v_{ed} means a desired velocity of the end-point. In this case, we have to consider one important thing. If the input-output relation of the plant is active, the time domain passivity control scheme cannot be applied, since the time domain passivity control scheme has been developed in the framework that the input-output relation of the plant is passive. If the end-point position is a plant output and joint torque is a plant input, the input-output relation of the plant is possibly active. Thus the overall control system may not be passive even though the controller remains passive.

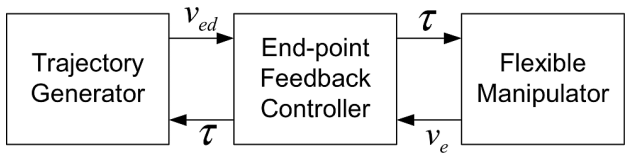


Fig. 7. A network model of flexible manipulator with end-point feedback

To solve this problem, we make the above network model suitable to our framework. The important physical fact is that the conjugate input-output pair (v_e, τ) is not simulating physical output energy from the controller to the flexible manipulator, the energy flows into the flexible manipulator through only the place where the actuator is attached. Even though the controller uses non-collocated sensor information to generate its output, the actual physical energy that is transmitted to the flexible manipulator is determined by the conjugate pair at the actuating position. Therefore, we can extract link dynamics from the joint velocity to the end-point velocity from the flexible manipulator (which has noncollocated feedback) and include it in the controller block (Fig. 8). The noncollocated (possibly active) system is then separated into the collocated (passive) system and a dynamics from the collocated output (joint velocity) to the noncollocated output (end-point velocity). As a result, if it is possible, and it generally is, to use the velocity information measured at the actuating position, we can construct the network model (controller and passive plant) that is suitable to our framework as in Fig. 8 by including the link dynamics that cause the noncollocation problem into the controller.

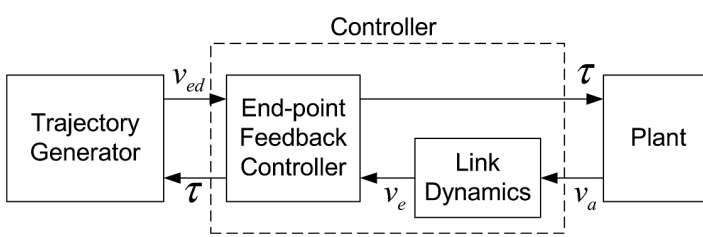


Fig. 8. A modified network model of flexible manipulator with end-point feedback

3.2 Designing the PO/PC

First, for designing the PO, it is necessary to check the real-time availability of the conjugate signal pairs at each port of the controller. The conjugate pair at the port that is connected with the trajectory generator is usually

available since the desired trajectory (v_{ed}) is given and the controller output (τ) is calculated in real-time. Furthermore, the conjugate pair is generally available for the other port that is connected to the plant since the same controller output (τ) is used, and the output velocity of the actuating position (v_a) is measured in real-time. Thus, the PO is designed as

$$E_{obsv}(n) = \Delta T \sum_{k=0}^{n} (\tau(k)v_{ed}(k) - \tau(k)v_a(k)) + E(0) = \Delta T \cdot W(n) \tag{5}$$

After designing the PO, the causality of each port of the controller should be determined in order to choose the type of PC for implementation. In a noncollocated flexible manipulator control system, the output of the trajectory generator is the desired velocity (v_{ed}) of the end-point, and the controller output (τ) is feedback to the trajectory generator. Thus, the port that is connected with the trajectory generator has impedance causality. Also, the other port of the controller has impedance causality because a motion controlled flexible manipulator usually has admittance causality (torque input, τ and joint velocity output, v_a). Thus, two series PCs have to be placed at each port to guarantee the passivity of the controller (Fig. 9).

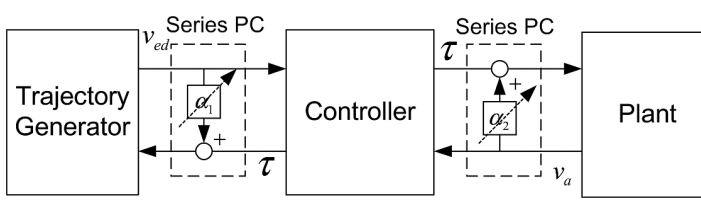


Fig. 9. Configuration of PC for a flexible manipulator with end-point feedback

From the result in [12], the initial energy of the controller is as follows:

$$E(0) = \frac{1}{2} K_p e(0)^2 \tag{6}$$

where K_p is a proportional gain and $e(0)$ is the position error of the end-point at the starting time.

4 Simulation Examples

Many researchers have used a flexible manipulator for testing newly developed control methods due to its significant control challenges. In this section, the proposed stability guaranteed control scheme for noncollocated control systems is tested for feasibility with a simulated flexible link manipulator.

Table 1. Physical properties of a single-link flexible manipulator

Link	Tip mass	Hub
Stiffness (EI): $11.85Nm^2$	Mass (M_e): $0.5867Kg$	Rotational inertia (I_h): $0.016Kgm^2$
Thickness (H): $47.63e-4m$	Rotational inertia (J_h): $0.2787Kgm^2$	
Unit length mass (ρA): $0.2457Kg/m$		
Length (L): $1.1938m$		

The experimentally verified single link flexible manipulator model [7] is employed in this paper. A single link flexible manipulator having a planar motion is detailed in Fig. 6. The rotational inertia of the servo motor, the tachometer, and the clamping hub are modeled as a single hub inertia I_h. The payload is modeled as an end mass M_e and a rotational inertia J_e. The joint friction is included in the damping matrix. The system parameters in Fig. 6 are given in Table 1. The closed form dynamic equation is derived using the assumed mode method. For the system dynamic model, the flexible mode is modeled up to the third mode, that is, an 8th order system is considered. The following PD controller gain is used, $K_P = 30, K_D = 0.8$. In this noncollocated feedback system, the hub angle and joint torque can be considered as a conjugate pair to calculate physical energy output flow into the flexible manipulator (see Section 3.1).

Without the PC turned on, tip-position tracking control was simulated (Fig. 10). The desired tip-position trajectory was $x_d(t) = 0.1sin(t)$. The tip-position could not follow the desired trajectory and tip position and control input have oscillation which increases with time (Fig. 10a,b). The PO (Fig. 10c) grew to more and more negative values.

Stable tip-position tracking is achieved with the PC turned on. Tip-position tracks the desired trajectory very well (Fig. 11a), and the PO is constrained to positive values (Fig. 11c). The PC at the both side is briefly active only when these are required, and dissipates only the amount of energy actually generated (Fig. 11d)

5 Discussion

In this paper, we propose a stability guaranteed control scheme for noncollocated feedback control systems without any model information. The main contribution of this research is proposing a method to implement the PO/PC for a possibily active plant due to the noncollocated feedback. Without needing model information, we separate the active plant into a passive plant and a possibly active transfer function from the collocated output to the noncollocated output which allows the control system to be fit to the PO/PC

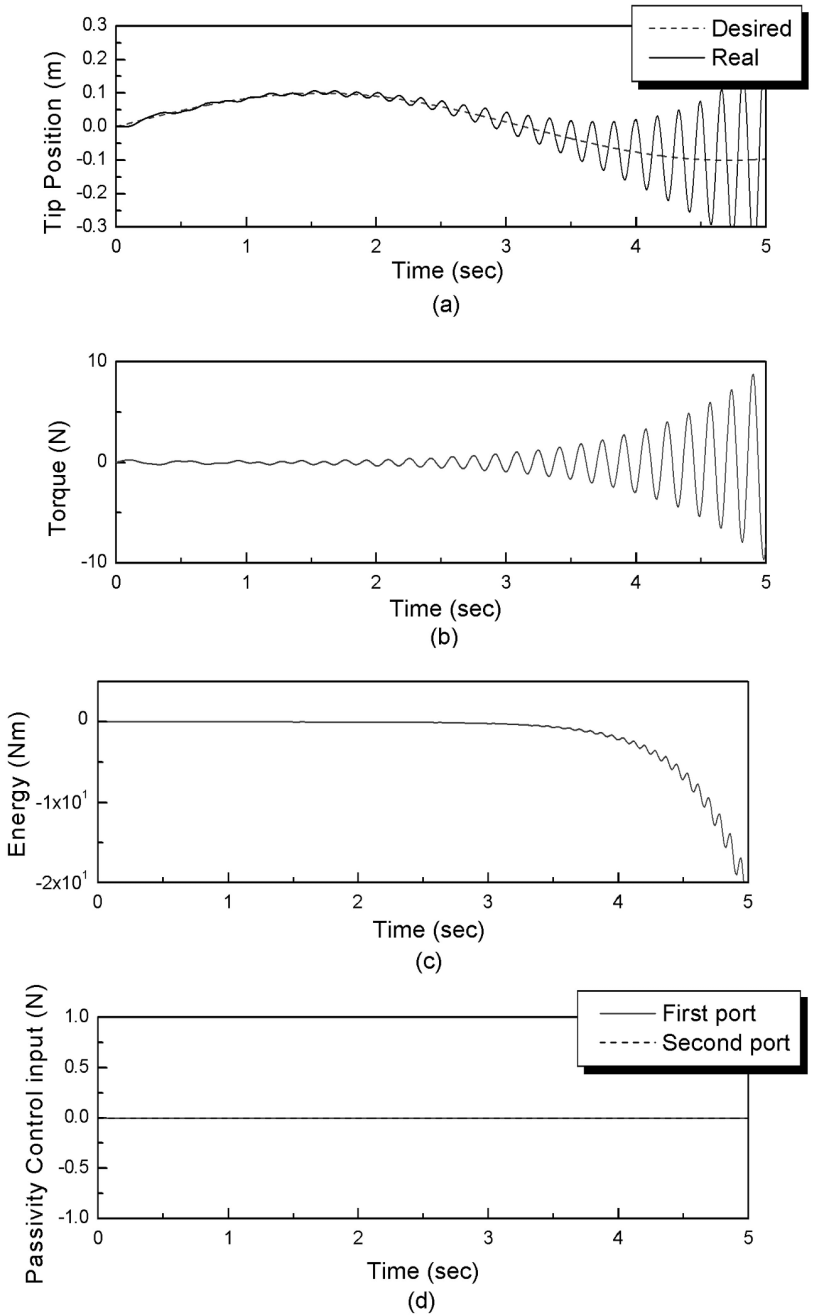


Fig. 10. Tip-position feedback without the PC

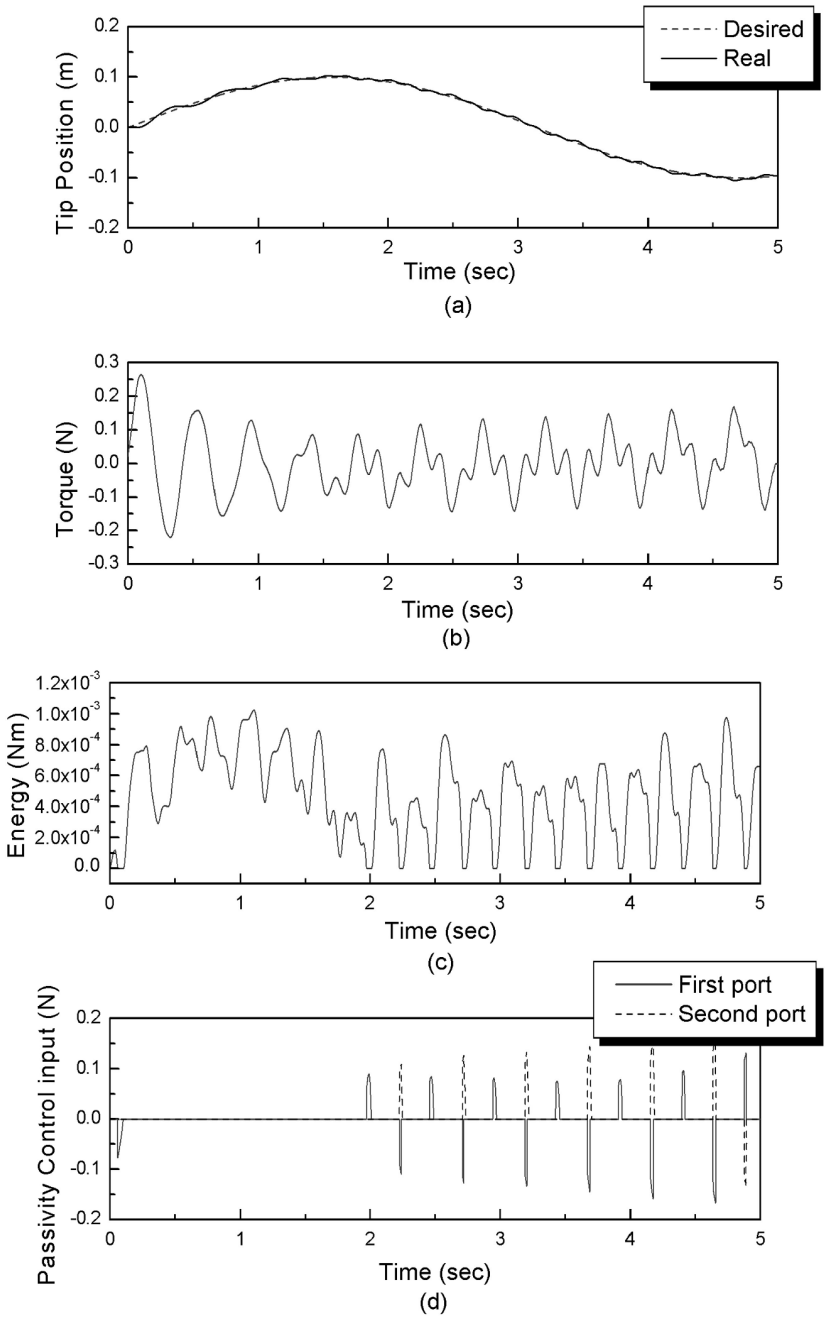


Fig. 11. Tip-position feedback with the PC

framework. Simulation results with an 8th order realistic model demonstrated stable control even for the noncollocated control system.

As a further work, we will prove the feasibility of the proposed PO/PC approach through an experiment with flexible manipulator. One of the expected problem is the performance degradation due to the noise during low values of velocity.

References

1. Adams R. J., and Hannaford B. (1999) "Stable Haptic Interaction with Virtual Environments," *IEEE Trans. Robot. Automat.*, vol. 15, no. 3, 465-474.
2. Book W. J. (1993) "Controlled Motion in an elastic world," *ASME Journal of Dynamic Systems Measurement and Control*, 115(2B), pp. 252-261.
3. Cannon R. H., and Schmitz E. (1984) "Initial Experiments on the End-point Control of a Flexible One-link Robot," *Int. Journal of Robotics Research*, vol. 3, no. 3, pp. 62-75.
4. Chudavarapu P. A., and Spong M. W. (1996) "On Noncollocated Control of a Single Flexible Link," *IEEE Int. Conf. Robotics and Automation*, MN, April.
5. Desoer C. A., and Vidyasagar M. (1975) Feedback Systems: Input-Output Properties, New York: Academic.
6. Hannaford B., and Ryu J. H. (2002) "Time Domain Passivity Control of Haptic Interfaces," *IEEE Trans. On Robotics and Automation*, Vol. 18, No. 1, pp. 1-10.
7. Kwon D. S., and Book W. J. (1994) "Time-domain inverse dynamic tracking control of a single-link flexible manipulator," *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, Vol. 116, No. 2, pp. 193-200.
8. De Luca A., and Siciliano B. (1989) "Trajectory Control of a Non-linear One-link Flexible Arm," *Int. Journal of Control*, vol. 50, no. 5, pp. 1699-1715.
9. Pota H. R., and Vidyasagar M. (1991) "Passivity of Flexible Beam Transfer Function with Modified Outputs," *IEEE Int. Conf. Robotics and Automation*, pp. 2826-2831.
10. Ryu J. H., Kim Y. S., and Hannaford B. (2002) "Sampled and Continuous Time Passivity and Stability of Virtual Environments," *Submitted to the 2003 IEEE Int. Conf. on Robotics and Automation.*
11. Ryu J. H., Kwon D. S., and Hannaford B. (2002) "Stable Teleoperation with Time Domain Passivity Control," *Proc. of the 2002 IEEE Int. Conf. on Robotics and Automation*, Washington DC, USA, pp1863-1869.
12. Ryu J. H., Kwon D. S., and Hannaford B. (2002) "Stability Guaranteed Control: Time Domain Passivity Approach," *will be published in IROS2002.*
13. Sciliano B., and Book W. J. (1988) "A Singular Perturbation Approach to Control of Lightweight Flexible Manipulators," *Int. Journal of Robotics Research*, vol. 7, no. 4, pp. 79-90.
14. van der Schaft A.J. (2000) "L2-Gain and Passivity Techniques in Nonlinear Control," Springer, Communications and Control Engineering Series.
15. Vidyasagar M., and Anderson B. D. O. (1989) "Approximation and Stabilization of Distributed Systems by Lumped Systems," *Systems and Control Letters*, vol. 12, no. 2, pp. 95-101.

16. Willems J. C. (1972) "Dissipative Dynamical Systems, Part I: General Theory," *Arch. Rat. Mech. An.*, vol. 45, pp. 321-351.
17. Wang W. J., Lu S. S., and Hsu C. F. (1989) "Experiments on the Position Control of a One-link Flexible robot arm," *IEEE Trans. on Robotics and Automation*, vol. 5, no. 3, pp. 373-377.
18. Wang D., and Vidyasagar M. (1990) "Passivity Control of a Single Flexible Link," *IEEE Int. Conf. Robotics and Automation*, pp. 1432-1437.

Cartesian Compliant Control Strategies for Light-Weight, Flexible Joint Robots

Alin Albu-Schäffer and Gerd Hirzinger

DLR Oberpfaffenhofen, German Aerospace Center, Institute of Robotics and Mechatronics

Abstract. The paper focuses on theoretical and experimental aspects related to the Cartesian compliant control of flexible joint robots with joint torque measurement. While the Cartesian impedance control for rigid robots, as well as the joint level control of flexible joint robots have been studied in detail, their combined implementation on robots with six or seven joints still leaves many open questions from a practical point of view. On the other hand, from a theoretical point of view it is not always possible to prove the stability of simpler, practically implementable controller structures. The solutions chosen for the DLR robots, as well as some experimental results are presented.

1 Introduction

The currently growing research interest in application fields such as service robotics, health care, space robotics or force feedback systems has led to an increasing demand for light robot arms with a load to weight ratio comparable to that of human arms. These manipulators should be able to perform compliant manipulation in contact with an unknown environment and guarantee the safety of humans interacting with them. A major problem which is specific to the implementation of light-weight robot concepts is the inherent flexibility introduced into the robot joints. Consequently, the success in the above mentioned robotics fields is strongly dependent on the design and implementation of adequate control strategies which can:

- compensate for the weakly damped elasticity in the robot joints and
- provide the desired Cartesian compliant behaviour of the manipulator.

These two control goals require measurement capabilities which clearly exceed the classical position sensing of industrial robots. The solution chosen in the case of the DLR light-weight robots was to equipe the joints with torque sensors in addition to motor position sensors [8]. Additionally, a 6 DOF force-torque sensor was mounted on the robot wrist.

The following well known model is used for the flexible joint robot [16]:

$$\boldsymbol{\tau}_m = \boldsymbol{J}_0\ddot{\boldsymbol{q}}_1 + \boldsymbol{\tau} + \boldsymbol{D}\boldsymbol{K}^{-1}\dot{\boldsymbol{\tau}}\,, \tag{1}$$

$$\boldsymbol{\tau} + \boldsymbol{D}\boldsymbol{K}^{-1}\dot{\boldsymbol{\tau}} = \boldsymbol{M}(\boldsymbol{q}_2)\ddot{\boldsymbol{q}}_2 + \boldsymbol{v}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2) + \boldsymbol{g}(\boldsymbol{q}_2) + \boldsymbol{\tau}_{\text{ext}}\,, \tag{2}$$

$$\boldsymbol{\tau} = \boldsymbol{K}(\boldsymbol{q}_1 - \boldsymbol{q}_2)\,. \tag{3}$$

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 135–151, 2003

$\boldsymbol{\tau}_m$ is the motor torque vector, $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ are the motor and link positions, respectively, and $\boldsymbol{\tau}$ is the joint torque. $\boldsymbol{J}_0$ is the motor inertia matrix, $\boldsymbol{K}$ and $\boldsymbol{D}$ are the elasticity and damping matrices. These matrices are diagonal and positive definite. The value of the damping is usually small and therefore it can be neglected as required by many of the known control methods. $\boldsymbol{M}$, $\boldsymbol{v}$ and $\boldsymbol{g}$ are the same as for rigid robots: the mass matrix, the Coriolis and centripetal torque vector and the gravity vector. $\boldsymbol{\tau}_{\text{ext}} = \boldsymbol{J}^T(\boldsymbol{q}_2)\boldsymbol{f}_{\text{ext}}$ is the external torque vector, which is produced by a force $\boldsymbol{f}_{\text{ext}}$ acting at the tip of the robot and which is related to this force through the transposed Jacobian $\boldsymbol{J}^T(\boldsymbol{q}_2)$. In the following, the notation

$$\boldsymbol{N}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2) = \boldsymbol{v}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2) + \boldsymbol{g}(\boldsymbol{q}_2) \tag{4}$$

is often used for brevity.

The aim of an impedance controller is to establish a mass-damper-spring relationship between the Cartesian position displacement $\Delta\boldsymbol{x} = \boldsymbol{x}_0 - \boldsymbol{x}$ and the Cartesian force $\boldsymbol{f}_{\text{ext}}$:

$$\boldsymbol{f}_{\text{ext}} = \boldsymbol{M}_{\text{k}}\Delta\ddot{\boldsymbol{x}} + \boldsymbol{D}_{\text{k}}\Delta\dot{\boldsymbol{x}} + \boldsymbol{K}_{\text{k}}\Delta\boldsymbol{x}, \tag{5}$$

where $\boldsymbol{M}_{\text{k}}$, $\boldsymbol{D}_{\text{k}}$ and $\boldsymbol{K}_{\text{k}}$ are positive definite matrices representing the virtual inertia, damping and stiffness of the system. If the focus is on the stiffness and damping properties, there exist various control strategies to implement this behaviour on a real robot system [5,6,9,10,15,18].

The paper is organized as follows: in Sec. 2 three classical approaches to the Cartesian compliant control problem for rigid robots are reviewed and a new method, based on an impedance controller enhanced by local stiffness control is proposed. In Sec. 3, the corresponding controllers on joint level are addressed for each of the Cartesian controllers. Finally, Sec. 4 presents an experiment which compares the performance of the different controllers.

2 Cartesian Compliant Control

Figure 1 shows the three most popular approaches to Cartesian compliance control, as they were implemented on the DLR light-weight robots. By using motor position and joint torque measurements, a position, an impedance, as well as a torque controller can be implemented in a decentralized manner at joint level. Based on these interfaces, Cartesian admittance, stiffness or impedance controller structures can be used to implement the desired Cartesian behaviour. This allows a direct comparison of the different methods in terms of attainable stiffness, geometric accuracy or robustness.

2.1 Impedance control methods

In this section we summarize the three aforementioned control methods, which can be used to obtain the desired compliant behaviour in Cartesian

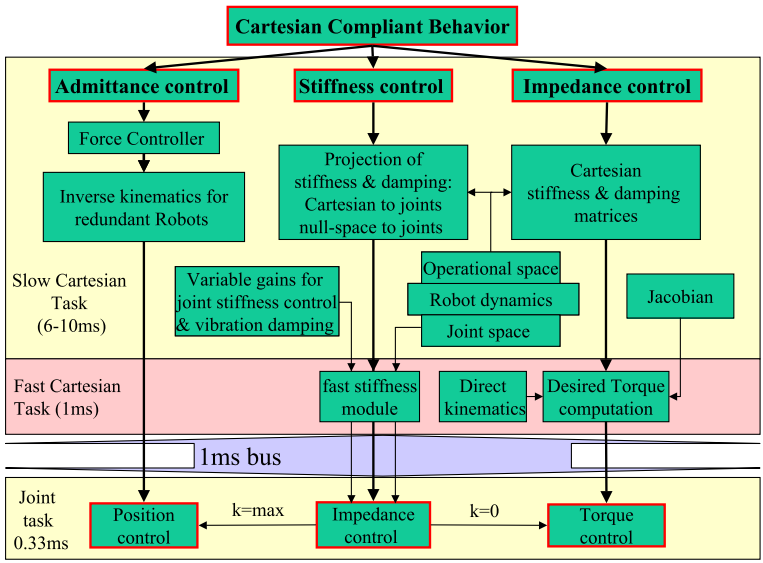


Fig. 1. Controller architecture for DLR's light-weight robots

space. It is useful to keep in mind that, regardless of the control method used, the motor torque is eventually the value commanded to the robot.

Admittance Control. The Cartesian force at the end-effector is measured in the case of admittance control by a 6DOF force-torque sensor. The force vector is used to generate a desired Cartesian position $\boldsymbol{x}_d$. Using the inverse kinematics $\mathcal{K}^{-1}$, this displacement is converted to the desired joint positions $\boldsymbol{q}_{2d}$. The joint position controller $\mathcal{P}_{\mathcal{R}}$ then generates the motor torques:

$$\boldsymbol{x}_d(s) = \boldsymbol{x}_0(s) - \frac{\Delta \boldsymbol{f}(s)}{\boldsymbol{K}_{\mathrm{k}} + \boldsymbol{D}_{\mathrm{k}} s} \rightarrow \boldsymbol{q}_{2d} = \mathcal{K}^{-1}\{\boldsymbol{x}_d\} \rightarrow \boldsymbol{\tau}_m = \mathcal{P}_{\mathcal{R}}\{\boldsymbol{q}_{2d}\} \tag{6}$$

where $\boldsymbol{q}_2$ is the link side position of the joints. This method is the most commonly used one, since most robots only have a position interface. The advantages are that the high gain position controllers can compensate for the friction in the joints and that, for the implementation of a high stiffness, low gains are needed in the Cartesian control loop. Hence it is clear that stability problems will appear for low desired stiffness and damping, for which the bandwidth of the Cartesian control loop approaches the joint bandwidth. This problem is even more noticeable for flexible joint robots, since in that case the bandwidth of the joint control is more critical. Further problems arise in the vicinity of singularities, where Cartesian position control may typically lead to fast, destabilizing movements.

Impedance Control. The impedance control approach directly uses equation (5), in which the actual Cartesian position is computed from the position $\boldsymbol{q}_2$ using the forward kinematics $\boldsymbol{x} = \mathcal{K}(\boldsymbol{q}_2)$. Using the transposed Jacobian $\boldsymbol{J}^T(\boldsymbol{q}_2)$, the Cartesian force is transformed into desired joint torques. The joint torque controller $\mathcal{T}_\mathcal{R}$ then generates the motor torque command:

$$\boldsymbol{f}_d = \boldsymbol{K}_\mathrm{k} \Delta\boldsymbol{x} + \boldsymbol{D}_\mathrm{k} \Delta\dot{\boldsymbol{x}} \rightarrow \boldsymbol{\tau}_d = \boldsymbol{J}^T(\boldsymbol{q}_2)\boldsymbol{f}_d \rightarrow \boldsymbol{\tau}_m = \mathcal{T}_\mathcal{R}\{\boldsymbol{\tau}_d\} \tag{7}$$

To obtain good results with this method, a joint torque controller which can overcome the joint friction disturbance is very useful. The impedance controller is, in principle, complementary to the admittance controller. It is well suited for low stiffness and damping, which now require low gains in the Cartesian loop, while the bandwidth of the torque controller is optimally exploited. Conversely, the stability problems will appear for high Cartesian stiffness.

The behaviour at singularities is also different from that of the admittance controller. Components of $\boldsymbol{f}_d$, which act in singular directions, are not mapped to the joint space. The movement in the vicinity of singularities will be stable and smooth, but the resulting stiffness matrix will be distorted.

Stiffness control. From the previous two methods one can see that appropriate performance can be achieved if the gains in the Cartesian control loop are substantially lower than those in the joint controllers. This leads to the idea of converting the desired Cartesian stiffness and damping to corresponding matrices for the joint stiffness $\boldsymbol{K}_j$ and damping $\boldsymbol{D}_j$. Assuming that the desired Cartesian matrices are changing rather slowly, there is no need for very high Cartesian sampling rates. The joint impedance controller $\mathcal{S}_\mathcal{R}$ can then be used to generate the desired motor torque.

$$\{\boldsymbol{K}_j, \boldsymbol{D}_j\} = \mathcal{T}\{\boldsymbol{K}_\mathrm{k}, \boldsymbol{D}_\mathrm{k}\} \rightarrow \boldsymbol{\tau}_m = \mathcal{S}_\mathcal{R}\{\mathcal{K}^{-1}(\boldsymbol{x}_d), \boldsymbol{K}_j, \boldsymbol{D}_j\} \tag{8}$$

Limitations of the stiffness controller. The mapping $\mathcal{T}$ as well as the matrices $\{\boldsymbol{K}_j, \boldsymbol{D}_j\}$ have only a local meaning. $\boldsymbol{K}_j$ and $\boldsymbol{D}_j$ give a local relationship between the joint torque and the joint position or the joint velocities, respectively [1]:

$$\begin{aligned}
\boldsymbol{D}_j &= \frac{\partial \boldsymbol{\tau}}{\partial \dot{\boldsymbol{q}}_2} = \frac{\partial (\boldsymbol{J}(\boldsymbol{q}_2)^T \boldsymbol{D}_\mathrm{k} \Delta\dot{\boldsymbol{x}})}{\partial \dot{\boldsymbol{q}}_2} = \boldsymbol{J}(\boldsymbol{q}_2)^T \boldsymbol{D}_\mathrm{k} \boldsymbol{J}(\boldsymbol{q}_2) \\
\underline{\boldsymbol{K}_j} &= \underline{\frac{\partial \boldsymbol{\tau}}{\partial \boldsymbol{q}_2} = \frac{\partial (\boldsymbol{J}(\boldsymbol{q}_2)^T \boldsymbol{K}_\mathrm{k} \Delta\, \boldsymbol{x})}{\partial \boldsymbol{q}_2}} = \boldsymbol{J}(\boldsymbol{q}_2)^T \boldsymbol{K}_\mathrm{k} \boldsymbol{J}(\boldsymbol{q}_2) + \frac{\partial \boldsymbol{J}(\boldsymbol{q}_2)^T}{\partial \boldsymbol{q}_2} \boldsymbol{K}_\mathrm{k}\, \Delta \boldsymbol{x}
\end{aligned} \tag{9}$$

[1] For sake of simplicity, we used here the somewhat sloppy expression from [9] of the derivative of a matrix by a vector $\frac{\partial (\boldsymbol{J}(\boldsymbol{q}_2))}{\partial \boldsymbol{q}_2}$. Writing the expression for every row of $\boldsymbol{K}_{j_i}$ with $i = 1, \ldots, N$ eliminates the ambiguity. The resulting matrix is of course equivalent to the formulation of the conservative congruence transformation in [6].

In the early work of [15], only the first term of (9) is mentioned. The importance of the second term, which reflects the position dependent change of the Jacobian is pointed out in [9,6]. The idea of mapping the Cartesian stiffness to the joint stiffness is biologically motivated, it enables stable and fast manipulation despite of a time delay of several tens of ms for the communication to the central nervous system. The human arm is able to change its stiffness by contracting antagonistic muscle pairs [9,11]. In the case of robotic manipulators, this can be accomplished only by means of control, the input for the robot is always the motor torque.

The direct implementation of a Cartesian stiffness and damping with the control law:

$$\boldsymbol{\tau}_d = \boldsymbol{K}_j \Delta\boldsymbol{q}_2 - \boldsymbol{D}_j \dot{\boldsymbol{q}}_2 = \boldsymbol{K}_j (\mathcal{K}^{-1}\{\boldsymbol{x}_0\} - \boldsymbol{q}_2) - \boldsymbol{D}_j \dot{\boldsymbol{q}}_2 \tag{10}$$

leads, at least for higher displacements $\Delta\boldsymbol{x}$ from the desired position, to substantial errors in the stiffness matrix. The reason for these errors is the local character of (9). The last term in (9) does not completely eliminate this error, because $\frac{\partial(\boldsymbol{K}_\mathrm{k}\Delta\boldsymbol{x})}{\partial\boldsymbol{q}} = \boldsymbol{K}_\mathrm{k}\boldsymbol{J}(\boldsymbol{q}_2)$ is also valid only locally. Here, the Jacobian is implicitly assumed to be constant, despite of being multiplied by a big displacement $\Delta\boldsymbol{q}_2$.

2.2 Impedance controller enhanced by local stiffness control

In this section, a new controller structure for the implementation of Cartesian stiffness is presented. Based on the remarks in the previous section, the new controller structure is designed with the following considerations in mind:

(A1) $\boldsymbol{J}(\boldsymbol{q}_2)$ and $\boldsymbol{x} = \mathcal{K}(\boldsymbol{q}_2)$ may be computed in a slower Cartesian control loop.
(A2) The robot has a fast joint control loop, for which the time delay is negligible, so that $\boldsymbol{q}_2 \approx \boldsymbol{q}_{2j}$.
(A3) Equation (10) is valid only locally.

Notations: The signals, which are measured or computed in the Cartesian loop are denoted by the index $[]_k$. The index $[]_j$ refers to signals determined in the joint control loop, while the real (instantaneous) values have no index. Furthermore, for the difference signals, the notations $\Delta\boldsymbol{q}_k = \boldsymbol{q}_0 - \boldsymbol{q}_k$ and $\Delta\boldsymbol{q}_j = \boldsymbol{q}_k - \boldsymbol{q}_j$ are used. Considering (A1), it follows that:

$$\Delta\boldsymbol{x} = \boldsymbol{x}_0 - \boldsymbol{x} \approx \boldsymbol{x}_0 - \boldsymbol{x}_k - \left.\frac{\partial\boldsymbol{x}}{\partial\boldsymbol{q}_2^T}\right|_{\boldsymbol{q}_2=\boldsymbol{q}_{2k}} (\boldsymbol{q}_{2j} - \boldsymbol{q}_{2k}) = \Delta\boldsymbol{x}_k + \boldsymbol{J}(\boldsymbol{q}_{2k})\Delta\boldsymbol{q}_{2j} \tag{11}$$

$$\boldsymbol{J}^T(\boldsymbol{q}_2) \approx \boldsymbol{J}^T(\boldsymbol{q}_{2j}) \approx \boldsymbol{J}^T(\boldsymbol{q}_{2k}) + \Delta\boldsymbol{q}_{2j}^T \left.\frac{\partial\boldsymbol{J}^T(\boldsymbol{q}_2)}{\partial\boldsymbol{q}_2}\right|_{\boldsymbol{q}_2=\boldsymbol{q}_{2k}} \tag{12}$$

This leads to the following expression for the stiffness induced component of the desired joint torque:

$$\boldsymbol{\tau}_{dK} = \boldsymbol{J}^T(\boldsymbol{q}_2)\boldsymbol{K}_{\mathrm{k}}\Delta\boldsymbol{x} = \boldsymbol{J}^T(\boldsymbol{q}_{2k})\boldsymbol{K}_{\mathrm{k}}\Delta\boldsymbol{x}_k + \boldsymbol{J}^T(\boldsymbol{q}_{2k})\boldsymbol{K}_{\mathrm{k}}\boldsymbol{J}(\boldsymbol{q}_{2k})\Delta\boldsymbol{q}_{2j} + \\ + \Delta\boldsymbol{q}_{2j}^T \left.\frac{\partial \boldsymbol{J}^T(\boldsymbol{q}_2)}{\partial \boldsymbol{q}_2}\right|_{\boldsymbol{q}_2=\boldsymbol{q}_{2k}} \boldsymbol{K}_{\mathrm{k}}\Delta\boldsymbol{x}_k + \Delta\boldsymbol{q}_{2j}^T \left.\frac{\partial \boldsymbol{J}^T(\boldsymbol{q}_2)}{\partial \boldsymbol{q}_2}\right|_{\boldsymbol{q}_2=\boldsymbol{q}_{2k}} \boldsymbol{K}_{\mathrm{k}}\boldsymbol{J}(\boldsymbol{q}_{2k})\Delta\boldsymbol{q}_{2j} \quad (13)$$

The first term corresponds to the impedance controller at Cartesian level. The second term corresponds to the stiffness controller, as described in [15]. Here, in contrast to (10), the stiffness controller acts only locally, in the vicinity of the last Cartesian position. This helps to overcome the slower Cartesian sampling rate by the faster joint controller.

The third term corresponds to the correction term in (9), which ensures the conservativeness of the mapping. Since the term contains the Cartesian elastic force for the complete displacement $\boldsymbol{K}_{\mathrm{k}}\Delta\boldsymbol{x}_k$, it can not always be ignored, although the variation of the Jacobian within a Cartesian cycle is small. Finally, the fourth term depends on the square of the small displacement $\Delta\boldsymbol{q}_{2j}$ and consequently has no practical significance.

In a similar way one obtains for the damping term:

$$\boldsymbol{\tau}_{dD} = \boldsymbol{J}^T(\boldsymbol{q}_{2k})\boldsymbol{D}_{\mathrm{k}}\boldsymbol{J}(\boldsymbol{q}_{2k})\Delta\dot{\boldsymbol{q}}_{2j} + \Delta\boldsymbol{q}_{2j}^T \left.\frac{\partial \boldsymbol{J}(\boldsymbol{q}_2)^T}{\partial \boldsymbol{q}_2}\right|_{\boldsymbol{q}_2=\boldsymbol{q}_{2k}} \boldsymbol{D}_{\mathrm{k}}\boldsymbol{J}(\boldsymbol{q}_{2k})\Delta\dot{\boldsymbol{q}}_{2j} \quad (14)$$

The second term in this expression may again be ignored.

The desired torque $\boldsymbol{\tau}_d = \boldsymbol{\tau}_{dK} + \boldsymbol{\tau}_{dD}$ can be commanded to the torque controller. To optimally exploit the performance of the joint controller, it is more advantageous to use the joint stiffness controller instead. The term $\boldsymbol{J}^T(\boldsymbol{q}_{2k})\boldsymbol{K}_{\mathrm{k}}\Delta\boldsymbol{x}_k$ is computed in the Cartesian loop and is sent as an additional commanded torque value to the stiffness controller. The two terms $\boldsymbol{J}^T(\boldsymbol{q}_{2k})\boldsymbol{K}_{\mathrm{k}}\boldsymbol{J}(\boldsymbol{q}_{2k})$ and $\left.\frac{\partial \boldsymbol{J}(\boldsymbol{q}_2)^T}{\partial \boldsymbol{q}_2}\right|_{\boldsymbol{q}_2=\boldsymbol{q}_{2k}} \boldsymbol{K}_{\mathrm{k}}\Delta\boldsymbol{x}$ are also computed in every Cartesian cycle and represent a constant desired stiffness, which is commanded to the joint controllers for the duration of this step.

Remark: In the current, decentralized joint controller structure, only the diagonal terms of the stiffness and damping matrices can be implemented using the joint impedance controller. The torques produced by the off-diagonal elements are computed on the central computer in a joint level task (at the bus sampling rate of 1ms) and are added to the desired torques.

3 Control of the flexible joint robot

The relationship between the above mentioned control methods and the low level control of a flexible joint robot is interesting from a practical and a theoretical point of view. The general problems for each of the three controllers will be briefly discussed and the solutions chosen for the DLR robots will be presented.

3.1 Admittance Control

The admittance controller accesses at joint level a position controller, for which well established control methods are available. Numerous powerful theoretical solutions have been proposed for position control of flexible joint robots, which are based, e.g., on feedback linearization, passivity control, backstepping or stability analysis of triangular systems [16,4,13,12]. These methods require the computation of higher order derivatives of the link position, as well as of the rigid robot dynamics terms (mass matrix, gravity term, Coriolis and centripetal terms), and hence are very challenging from an implementation point of view. Therefore experimental results are rarely reported for robots with 6 or 7 joints. Two of the approaches used for the DLR robots are reported below. The first refers to the practical implementation issues of advanced control algorithms, taking as an example the feedback linearization controller. The second method is a simple, decentralized, but very robust and efficient state feedback controller, for which a Lyapunov and passivity based stability proof has been given.

Practical implementation of advanced control strategies Expressed in terms of the link side position $\boldsymbol{q}_2$ and by neglecting the damping ($\boldsymbol{D} = \boldsymbol{0}$) and the external torque $\boldsymbol{\tau}_{\text{ext}}$, the robot dynamics are:

$$\begin{aligned}\boldsymbol{\tau}_m = \boldsymbol{K}^{-1}\boldsymbol{J}_0 \left(\ddot{\boldsymbol{M}}(\boldsymbol{q}_2)\ddot{\boldsymbol{q}}_2 + 2\dot{\boldsymbol{M}}(\boldsymbol{q}_2)\boldsymbol{q}_2^{(3)} + \boldsymbol{M}(\boldsymbol{q}_2)\boldsymbol{q}_2^{(4)} + \ddot{\boldsymbol{N}}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2) + \boldsymbol{K}\ddot{\boldsymbol{q}}_2\right) \\ + \boldsymbol{M}(\boldsymbol{q}_2)\ddot{\boldsymbol{q}}_2 + \boldsymbol{N}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2)\,. \qquad (15)\end{aligned}$$

For the implementation of feedback linearization, one can theoretically use the following control equation:

$$\begin{aligned}\boldsymbol{\tau}_m = \boldsymbol{K}^{-1}\boldsymbol{J}_0 \left(\ddot{\boldsymbol{M}}(\boldsymbol{q}_2)\ddot{\boldsymbol{q}}_2 + 2\dot{\boldsymbol{M}}(\boldsymbol{q}_2)\boldsymbol{q}_2^{(3)} + \boldsymbol{M}(\boldsymbol{q}_2)\boldsymbol{\nu} + \ddot{\boldsymbol{N}}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2) + \boldsymbol{K}\ddot{\boldsymbol{q}}_2\right) \\ + \boldsymbol{M}(\boldsymbol{q}_2)\ddot{\boldsymbol{q}}_2 + \boldsymbol{N}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2)\,, \qquad (16)\end{aligned}$$

$$\boldsymbol{\nu} = \boldsymbol{q}_{2d}^{(4)} + \boldsymbol{C}_4\tilde{\boldsymbol{q}}_2^{(3)} + \boldsymbol{C}_3\tilde{\boldsymbol{q}}_2^{(2)} + \boldsymbol{C}_2\dot{\tilde{\boldsymbol{q}}}_2 + \boldsymbol{C}_1\tilde{\boldsymbol{q}}_2 \qquad (17)$$

in order to obtain the following, decoupled error dynamics:

$$\tilde{\boldsymbol{q}}_2^{(4)} + \boldsymbol{C}_4\tilde{\boldsymbol{q}}_2^{(3)} + \boldsymbol{C}_3\tilde{\boldsymbol{q}}_2^{(2)} + \boldsymbol{C}_2\dot{\tilde{\boldsymbol{q}}}_2 + \boldsymbol{C}_1\tilde{\boldsymbol{q}}_2 = \boldsymbol{0}\,, \qquad (18)$$

where $\tilde{\boldsymbol{q}}_2 = \boldsymbol{q}_{2d} - \boldsymbol{q}_2$ is the link side position error. The matrices $\boldsymbol{C}_{1\ldots4}$ are designed such that the desired error dynamics are imposed. A first major practical problem arises from the computation of the higher derivatives of $\boldsymbol{M}(\boldsymbol{q}_2)$ and $\boldsymbol{N}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2)$. Their symbolical computation exceeds the available computing power and the numerical differentiation would lead to noisy signals. However, for velocity ranges which are typical for service robotics tasks, it turned out from numerical simulations that the contribution of these terms to the closed loop dynamics is not significant and therefore can be neglected [1], leading to a simplified controller:

$$\boldsymbol{\tau}_m = \boldsymbol{K}^{-1}\boldsymbol{J}_0\boldsymbol{M}(\boldsymbol{q}_2)\boldsymbol{\nu} + \big(\boldsymbol{J}_0 + \boldsymbol{M}(\boldsymbol{q}_2)\big)\ddot{\boldsymbol{q}}_2 + \boldsymbol{N}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2)\,. \qquad (19)$$

A second practical problem is the availability of the full state for the controller, which consists of the link side position and its derivatives up to the jerk: $\boldsymbol{x} = \{\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2, \ddot{\boldsymbol{q}}_2, \dddot{\boldsymbol{q}}_2\}$. These signals can be computed using the available motor position and joint torques signals, as well as their derivatives:

$$\boldsymbol{q}_2 = \boldsymbol{q}_1 - \boldsymbol{K}^{-1}\boldsymbol{\tau} \tag{20}$$

$$\dot{\boldsymbol{q}}_2 = \dot{\boldsymbol{q}}_1 - \boldsymbol{K}^{-1}\dot{\boldsymbol{\tau}} \tag{21}$$

$$\ddot{\boldsymbol{q}}_2 = \boldsymbol{M}^{-1}(\boldsymbol{q}_2)\left[\boldsymbol{\tau} - \boldsymbol{N}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2)\right] \tag{22}$$

$$\dddot{\boldsymbol{q}}_2 \approx \boldsymbol{M}^{-1}(\boldsymbol{q}_2)\left[\dot{\boldsymbol{\tau}} - \dot{\boldsymbol{N}}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2)\right] . \tag{23}$$

The same procedure can be applied to implement not only feedback linearization, but also some of the passivity based control methods [1]. Consequently, simplified versions of the previously mentioned, theoretical methods may be implemented using the torque and its derivative to compute the acceleration and the jerk, while neglecting some of the derivatives of the robot dynamic terms. Implementation and evaluation of these types of controllers for the DLR light weight robots is subject of current activity.

State feedback controller with gravity compensation. In practical implementations simpler controllers are often preferred, because of their robustness, decentralized implementation, and low computing power requirements. Singular perturbation based controllers are often used in this context [17], but are valid only for moderate elasticity in the joints. In the case of the DLR light-weight robots, a state feedback controller with gravity compensation (with motor position and velocity, as well as the torque and its derivative as states) is used. For this controller, stability can be proven in the regulation case [2]. The control input is as follows:

$$\boldsymbol{\tau}_m = \boldsymbol{K}_P\tilde{\boldsymbol{q}}_1 - \boldsymbol{K}_D\dot{\boldsymbol{q}}_1 - \boldsymbol{K}_T\boldsymbol{K}^{-1}\boldsymbol{\tau} - \boldsymbol{K}_S\boldsymbol{K}^{-1}\dot{\boldsymbol{\tau}} + (\boldsymbol{K}+\boldsymbol{K}_T)\boldsymbol{K}^{-1}\boldsymbol{g}(\boldsymbol{q}_{2d}) \tag{24}$$

$\boldsymbol{K}_P, \boldsymbol{K}_D, \boldsymbol{K}_S, \boldsymbol{K}_T$ are diagonal gain matrices. $\boldsymbol{q}_{2d}$ is the link side desired trajectory and $\tilde{\boldsymbol{q}}_1 = \boldsymbol{q}_{1d} - \boldsymbol{q}_1$ is the motor position error.

In this section we use the passivity theory to sketch a systematic way of deriving a Lyapunov function for the stability proof. A complete Lyapunov proof is given in [2]. It can be shown that the flexible joint robot controlled with (24) can be regarded as a parallel and feedback connection of passive systems (Fig. 2), and hence it is itself a passive system.

Let us consider the actuator side equation of the robot dynamics (1) together with (3) and the controller (24) for each joint as a system with the input $u = -\dot{q}_2$ and the output $y = \tau_a$, with $\tau_a = k(q_1 - q_2) + d(\dot{q}_1 - \dot{q}_2)$. We intend to write this system in the power form:

$$y^T(t)u(t) = \dot{V}(t) + D_p(t) \tag{25}$$

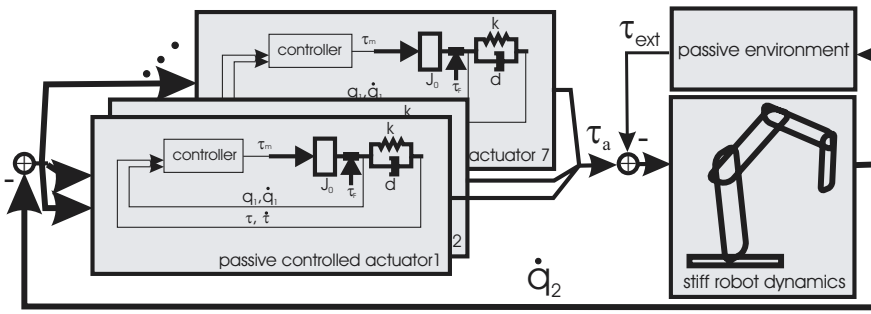


Fig. 2. Representation of the robot as a connection of passive blocks

with the stored energy V, the dissipated power D_p and the external power exchange $y^T u$. We have

$$\begin{aligned} -\tau_a \dot{q}_2 &= -k(q_1 - q_2)\dot{q}_2 - d(\dot{q}_1 - \dot{q}_2)\dot{q}_2 \\ &= \frac{d}{dt}\left(\frac{k}{2}(q_1 - q_2)^2\right) - k(q_1 - q_2)\dot{q}_1 - d(\dot{q}_1 - \dot{q}_2)\dot{q}_2 \end{aligned} \tag{26}$$

From (1),(3) and (24), by ignoring the gravity, we get

$$k(q_1 - q_2) = -\frac{k}{k + k_T}\left(J_0\ddot{q}_1 - k_P\tilde{q}_1 + k_D\dot{q}_1 + (d + k_S)(\dot{q}_1 - \dot{q}_2)\right) \tag{27}$$

Substituting (27) into (26) we finally obtain

$$\begin{aligned} -\tau_a \dot{q}_2 = \frac{d}{dt}\underbrace{\left(\frac{k_P k}{2(k + k_T)}\tilde{q}_1^2 + \frac{k}{2(k + k_T)}J_0\dot{q}_1^2 + \frac{1}{2}k(q_1 - q_2)^2\right)}_{V_i} + \\ + \underbrace{\frac{k_D k}{(k + k_T)}\dot{q}_1^2 - \frac{k_T d - k k_S}{k + k_T}(\dot{q}_1 - \dot{q}_2)\dot{q}_1 + d(\dot{q}_1 - \dot{q}_2)^2}_{D_{p_i}} \end{aligned} \tag{28}$$

The energy expression V_i is always positive and D_{p_i} is a quadratic function of $\{\dot{q}_1, \dot{q}_2\}$, which can be rewritten as:

$$\begin{aligned} D_{p_i} = \frac{1}{k + k_T}\big((k_D + k_S + d)k\dot{q}_1^2 \\ -((k_S + 2d)k + dk_T)\dot{q}_1\dot{q}_2 + (dk_T + dk)\dot{q}_2^2\big) \end{aligned} \tag{29}$$

It follows that the condition for the passivity of the system is $D_{p_i} > 0$. The condition is fulfilled for:

$$\begin{cases} k + k_T > 0 \\ k_D > \frac{(k_S k - k_T d)^2}{4kd(k + k_T)} \end{cases} \tag{30}$$

The Lyapunov function for the whole robot can be obtained as the sum of the energies V_i for each joint and the energy of the rigid robot

$$V_{\text{rigid}} = \frac{1}{2}\dot{\boldsymbol{q}}_2^T \boldsymbol{M}(\boldsymbol{q}_2)\dot{\boldsymbol{q}}_2 + U_G(\boldsymbol{q}_2)\,, \tag{31}$$

with $U_G(\boldsymbol{q}_2)$ being the potential energy of the gravity forces. The stability is preserved also in contact with any passive environment.

An advantage of this controller structure over the PD control [20] is that the joint elasticity is included in the same passive block as the controller, providing the possibility to compensate for the effects of joint deformation. It should be noticed that the controller structure (24) can implement by a proper parameterization a position, a stiffness or a torque controller, while, as long as condition (30) is fulfilled, its passivity property is preserved.

An intuitive extension of the state feedback controller presented above is a controller with variable controller gains of the form:

$$\boldsymbol{\tau}_m = \boldsymbol{K}_P(\boldsymbol{\theta})\tilde{\boldsymbol{q}}_1 + \boldsymbol{K}_D(\boldsymbol{\theta})\dot{\tilde{\boldsymbol{q}}}_1 - \boldsymbol{K}_T(\boldsymbol{\theta})\boldsymbol{K}^{-1}\tilde{\boldsymbol{\tau}} - \boldsymbol{K}_S(\boldsymbol{\theta})\boldsymbol{K}^{-1}\dot{\tilde{\boldsymbol{\tau}}} + \boldsymbol{N}(\dot{\boldsymbol{q}}_2, \boldsymbol{q}_2) + \boldsymbol{\tau}_{\text{ff}} \tag{32}$$

with

$$\tilde{\boldsymbol{\tau}} = \boldsymbol{\tau} - \boldsymbol{N}(\dot{\boldsymbol{q}}_2, \boldsymbol{q}_2) \tag{33}$$

and the feed forward term $\boldsymbol{\tau}_{\text{ff}}$, which contains terms that are dependent on $\boldsymbol{q}_{2d}$ and its higher derivatives. This control law leads to (24) by setting $\boldsymbol{\tau}_{\text{ff}} = \boldsymbol{0}$, $\boldsymbol{N}(\dot{\boldsymbol{q}}_2, \boldsymbol{q}_2) \approx \boldsymbol{g}(\boldsymbol{q}_{2d})$, $\dot{\boldsymbol{N}}(\dot{\boldsymbol{q}}_2, \boldsymbol{q}_2) \approx \boldsymbol{0}$ and $\dot{\boldsymbol{q}}_{1d} \approx \boldsymbol{0}$. The gains $\boldsymbol{K}_D, \boldsymbol{K}_P, \boldsymbol{K}_S, \boldsymbol{K}_T$ of the state feedback controller are then computed according to the instantaneous value of the mass matrix ($\theta_{\text{i}} = m_{\text{ii}}$), which is assumed to be quasi-stationary.

3.2 Stiffness Control

The problem of joint impedance control for flexible joint robots was very rarely addressed in literature. For the DLR light-weight robots, the state feedback controller with variable parameters (32) was used to implement a joint controller with variable stiffness and damping [1]. In this case, the controller gains $\boldsymbol{K}_{\text{Ri}} = [K_{\text{Dii}}, K_{\text{Pii}}, K_{\text{Sii}}, K_{\text{Tii}}]$ are computed as a function of the desired stiffness and damping of the joint ($\boldsymbol{\theta}_{\text{i}} = [K_{j\,\text{ii}}, D_{j\,\text{ii}}, m_{\text{ii}}]$).

According to the flexible joint model, the transfer function

$$H_{\text{i}}(s) = \frac{\tau_{\text{ext}\,\text{i}}(s)}{q_{2\text{i}}(s)} = \frac{N_{\text{i}}(s)}{Z_{\text{i}}(s)} \tag{34}$$

for one joint i from the torque $\tau_{\text{ext}\,\text{i}}$ to the link position has a fourth order denominator $Z_{\text{i}}(s)$. Hence, an approximation of the second order impedance

dynamics by the fourth order system (34) has to be found. A reasonable choice is to define the stiffness of the controlled joint as:

$$K_{\mathrm{j_{ii}}} = \left.\frac{\tau_{\mathrm{ext\,i}}(s)}{q_{\mathrm{2i}}(s)}\right|_{s=0} \tag{35}$$

and to design the feedback gains such that the characteristic polynom in (34) is

$$Z_{\mathrm{i_{des}}} = (s^2+2\xi_1\omega_{1\mathrm{s}}+\omega_1^2)(s^2+2\xi_2\omega_{2\mathrm{s}}+\omega_2^2)\,. \tag{36}$$

The desired damping $D_{\mathrm{j_{ii}}}$ can then be directly mapped to the parameters ξ_1 and ξ_2. The denominator $Z_{\mathrm{i}}(s) = Z_{\mathrm{i}}(s, \boldsymbol{\theta}_{0\mathrm{i}}, m_{\mathrm{ii}}, \boldsymbol{K}_{\mathrm{Ri}})$ is a function of the constant physical parameters of the joint $\boldsymbol{\theta}_{0\mathrm{i}}$, of the actual inertia m_{ii} and of the control gain vector $\boldsymbol{K}_{\mathrm{Ri}}$. The values of the parameters ω_1 and ω_2 in $Z_{\mathrm{i_{des}}}(s)$, and consequently the control gains $\boldsymbol{K}_{\mathrm{Ri}}$ in $Z_{\mathrm{i}}(s)$ are then chosen to simultaneously satisfy the following criteria:

- To implement the desired joint stiffness (35) and the desired damping;
- To maximize the bandwidth of the joint control, for the given instantaneous values of the inertia m_{ii};
- To provide, in case of $D_{\mathrm{j_{ii}}} = 1$ an active vibration damping of the flexible joint structure.

The stability proof for the Cartesian compliance controller based on stiffness control for flexible joint robots is an open theoretical problem.

3.3 Impedance Control

The stability of the impedance controller from Fig. 1 is based on a singular perturbation argument, with an inner torque control loop and an outer Cartesian impedance control loop.

Rewriting the model equations without damping only in terms of the link side position and the torque, leads to:

$$\boldsymbol{M}(\boldsymbol{q}_2)\ddot{\boldsymbol{q}}_2 = -\boldsymbol{N}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2) + \boldsymbol{\tau} \tag{37}$$

$$\begin{aligned}\boldsymbol{K}^{-1}\ddot{\boldsymbol{\tau}} = &-\big(\boldsymbol{M}(\boldsymbol{q}_2)^{-1} + \boldsymbol{J}_0^{-1}\big)\boldsymbol{\tau} \\ &+\boldsymbol{M}(\boldsymbol{q}_2)^{-1}\big(\boldsymbol{N}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2) + \boldsymbol{\tau}_{\mathrm{ext}}\big) + \boldsymbol{J}_0^{-1}\boldsymbol{\tau}_m\end{aligned} \tag{38}$$

The model is singularly perturbed for sufficiently high values of the stiffness $\boldsymbol{K}$. The system (37), (38) is reduced in the case of very stiff joints ($\boldsymbol{K} \to \infty$) to the well known rigid robot dynamics:

$$\big(\boldsymbol{M}(\boldsymbol{q}_2)+\boldsymbol{J}_0\big)\ddot{\boldsymbol{q}}_2+\boldsymbol{N}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2)+\boldsymbol{\tau}_{\mathrm{ext}} = \boldsymbol{\tau}_m\,. \tag{39}$$

Following the separation principle of the singular perturbation theory, the control input can be chosen as:

$$\boldsymbol{\tau}_m = \boldsymbol{\tau}_{ms} + \boldsymbol{\tau}_{mf} \,, \tag{40}$$

with a slow term $\boldsymbol{\tau}_{ms}$ and a fast term $\boldsymbol{\tau}_{mf}$. The equation (37) corresponds to the rigid robot dynamics, but the input is now the torque inside the spring instead of the motor torque. For the control of this subsystem, the impedance control law for rigid robots (7) with additional gravity compensation can be used:

$$\boldsymbol{\tau}_d = \boldsymbol{J}^T(\boldsymbol{q}_2)(\boldsymbol{K}_\mathrm{k}\Delta\boldsymbol{x} + \boldsymbol{D}_\mathrm{k}\Delta\dot{\boldsymbol{x}}) + \boldsymbol{g}(\boldsymbol{q}_2) \tag{41}$$

The slow control input $\boldsymbol{\tau}_{ms}$ follows from (38) for $\boldsymbol{K} \to \infty$ and $\boldsymbol{\tau} = \boldsymbol{\tau}_d$:

$$\boldsymbol{\tau}_{ms} = \boldsymbol{J}_0 \left[\left(\boldsymbol{M}(\boldsymbol{q}_2)^{-1} + \boldsymbol{J}_0^{-1} \right) \boldsymbol{\tau}_d - \boldsymbol{M}(\boldsymbol{q}_2)^{-1} \left(\boldsymbol{N}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2) + \boldsymbol{\tau}_\mathrm{ext} \right) \right] . \tag{42}$$

This leads to the following error dynamics of the slow system (39):

$$\begin{aligned} &(\boldsymbol{J}_0 \boldsymbol{M}(\boldsymbol{q}_2)^{-1} + \boldsymbol{I}) \boldsymbol{J}^T(\boldsymbol{q}_2) \left(\boldsymbol{K}_\mathrm{k}\Delta\boldsymbol{x} + \boldsymbol{D}_\mathrm{k}\Delta\dot{\boldsymbol{x}} \right) = \\ &\quad = (\boldsymbol{M}(\boldsymbol{q}_2) + \boldsymbol{J}_0)\ddot{\boldsymbol{q}}_2 + (\boldsymbol{J}_0 \boldsymbol{M}(\boldsymbol{q}_2)^{-1} + \boldsymbol{I}) \boldsymbol{v}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2) \,. \end{aligned} \tag{43}$$

Expressed only in terms of Cartesian quantities, this is a stable dynamics equation for constant $\boldsymbol{x}_0$:

$$\boldsymbol{\Lambda}\ddot{\boldsymbol{x}} - \boldsymbol{D}_\mathrm{k}\Delta\dot{\boldsymbol{x}} - \boldsymbol{K}_\mathrm{k}\Delta\boldsymbol{x} + \bar{\boldsymbol{v}}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2) = \boldsymbol{f}_\mathrm{ext} \,, \tag{44}$$

with the Cartesian mass matrix $\boldsymbol{\Lambda}$ and the correspondingly modified expression $\bar{\boldsymbol{v}}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2)$ for the Coriolis and centripetal terms [10].

If $\boldsymbol{\tau}_{mf}$ is chosen as:

$$\boldsymbol{\tau}_{mf} = -\boldsymbol{J}_0 \boldsymbol{K}^{-1/2} \boldsymbol{c}_1 \dot{\boldsymbol{\tau}} - \left[\left(\boldsymbol{I} + \boldsymbol{J}_0 \boldsymbol{M}(\boldsymbol{q}_2)^{-1} \right) - \boldsymbol{J}_0 \boldsymbol{c}_2 \right] \boldsymbol{e}_t \tag{45}$$

with $\boldsymbol{e}_t = \boldsymbol{\tau}_d - \boldsymbol{\tau}$, then from (38), by substituting (42) and (45), it follows that

$$\boldsymbol{K}^{-1}(\ddot{\boldsymbol{e}}_t - \ddot{\boldsymbol{\tau}}_d) + \boldsymbol{K}^{-1/2} \boldsymbol{c}_1 (\dot{\boldsymbol{e}}_t - \dot{\boldsymbol{\tau}}_d) + \boldsymbol{c}_2 \boldsymbol{e}_t = \boldsymbol{0} \,. \tag{46}$$

The dynamics of the boundary layer system is obtained by neglecting $\ddot{\boldsymbol{\tau}}_d$ and $\dot{\boldsymbol{\tau}}_d$:

$$\boldsymbol{K}^{-1}\ddot{\boldsymbol{e}}_t + \boldsymbol{K}^{-1/2} \boldsymbol{c}_1 \dot{\boldsymbol{e}}_t + \boldsymbol{c}_2 \boldsymbol{e}_t = \boldsymbol{0} \,. \tag{47}$$

A stable error dynamics can also be obtained without cancellation of the term containing the mass matrix and of $\boldsymbol{\tau}_\mathrm{ext}$ in (45), with a slight modification of (42) [14,3]. The state feedback controller (24) can then be parametrized to implement this torque controller.

4 Experiments

The controller structures discussed in Sec. 2.1 and Sec. 2.2, have been implemented on the DLR light-weight robots. In this section an experiment is described, which compares the performance of the various controllers. During the experiment, the desired position is kept constant. A very low stiffness is commanded in one translational direction (t_2), while the desired stiffness in the other Cartesian directions is high. Regardless of the forces applied by the user hand at the end-effector, the robot should consequently be able to move only in the t_2-direction. The controllers are compared in terms of the minimal and maximal stiffness that can be achieved, as well as in terms of the geometric accuracy for high displacements. The commanded values for the stiffness are summarized in table 1.

Table 1. Commanded values for the diagonal Cartesian stiffness matrix

t_1	t_2	t_3	roll [2]	pitch	yaw
3000 N/m	100 N/m	3000 N/m	200 Nm/rad	200 Nm/rad	200 Nm/rad

In a first design stage, the fast Cartesian task from Fig. 1 was not available, and therefore the Cartesian controller was implemented with a sampling rate of 6 ms. In this case, the pure impedance controller showed the poorest performance in terms of the attainable stiffness range.

The stiffness controller implemented with (10),(9) shows very good stiffness and damping range performances (Fig. 3). An important drawback of the method is, however, the poor performance in terms of geometric accuracy for high displacements.

The other two curves in Fig. 3 present the results for the impedance controller enhanced by local stiffness control (Sec. 2.2), with conservative and non-conservative congruence transformations. Due to the use of the impedance controller, the geometric accuracy is considerably improved, while the stability problems with the impedance controller are eliminated by the fast joint stiffness control. The advantages of a high stiffness range and a high bandwidth are preserved.

From Fig. 4 it can be seen that with respect to geometric accuracy, the admittance controller still has the best performance. Nevertheless, it also has some drawbacks compared to the controller from Sec. 2.2, namely the disturbed behaviour in the vicinity of singularities and the limitations for low stiffness and damping.

[2] Since the focus of this work is on the structures of the impedance controller, the values for the rotational stiffness were deliberately chosen to be high, and therefore the particular representation of orientations has no significant effects on the results, as pointed out in [5,18].

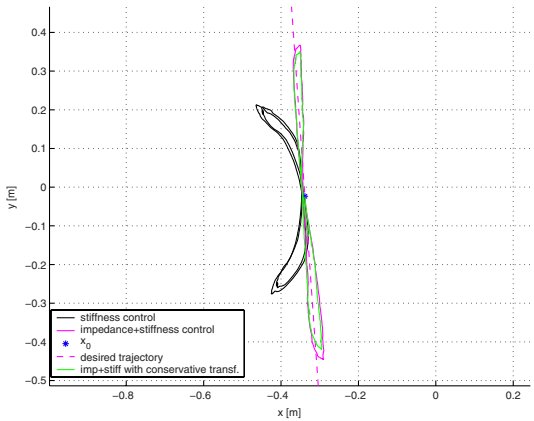


Fig. 3. Measurements for stiffness control and impedance control enhanced by local stiffness control: x-y plot.

A significant improvement to the impedance controller has been achieved by computing the direct kinematics and by evaluating (7) within a fast Cartesian cycle of 1ms, as shown in Fig. 1. Under these conditions, the impedance controller reaches a geometric accuracy which is comparable to that of the admittance controller (Fig. 5) , while the performance for low stiffness is significantly better.

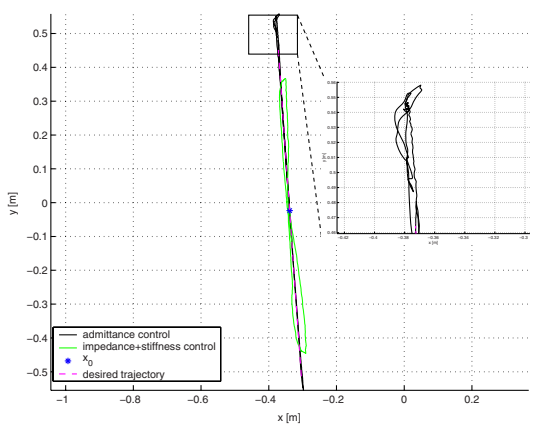


Fig. 4. Measurements for admittance control and impedance control enhanced by local stiffness control: x-y plot. The zoomed region corresponds to the singularity

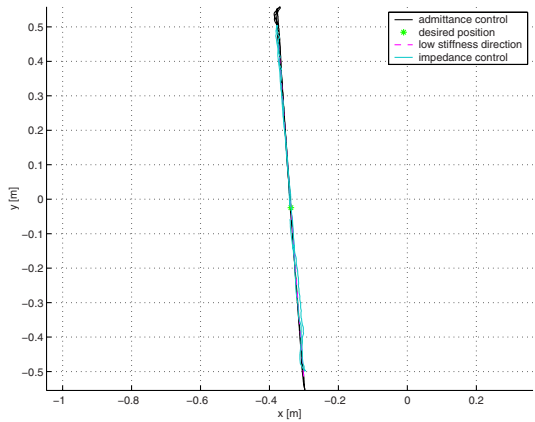


Fig. 5. Measurements for admittance control and impedance control with 1ms cycle: x-y plot

5 Discussion

The field of human friendly robotics faces, from the control point of view, the problem of implementing a Cartesian compliant behaviour on flexible joint robots. This involves cascaded structures of two complex controllers: an inner controller for the flexible joint structure and an outer Cartesian compliance controller. Due to the complexity of the task, there is currently a gap between simple controller structures, which can be practically implemented but often do not admit general stability proofs on one side, and complex theoretical approaches, which require vast sensor and model information (e.g., link position up to the third derivative) on the other side. Especially for the admittance and the stiffness controllers, practically efficient methods, which allow a stability proof for the general tracking case in contact with passive environments, do not exist. An attempt in this direction for the impedance controller was done in [7].

New hardware developments may put the problem in a new light. In this paper, solutions using joint torque sensors were discussed. Link acceleration sensors, which became commercially available, together with link position sensors could be an interesting alternative. New approaches for the mechanical design of human friendly robots [19,21], with variable mechanical stiffness or distributed, elastically coupled actuators may lead to new solutions also from the control point of view.

6 Conclusion

The paper gives an overview of the theoretical and implementation problems encountered with the Cartesian compliant control of flexible joint robots.

Some open theoretical and practical questions are pointed out. The practical insights gained with the DLR light-weight robots, as well as some theoretical contributions which emerged from this experience, are presented. Three different approaches are compared on the way towards control methods which are both practically feasible (regarding implementability, robustness and performance) and also allow a stability analysis for the complete flexible joint robot with Cartesian compliance control.

References

1. A. Albu-Schäffer. *Regelung von Robotern mit elastischen Gelenken am Beispiel der DLR-Leichtbauarme*. PhD thesis, Technical University Munich, april 2002.
2. A. Albu-Schäffer and G. Hirzinger. A globally stable state-feedback controller for flexible joint robots. *Journal of Advanced Robotics, Special Issue: Selected Papers from IROS 2000*, 15(8):799–814, 2001.
3. A. Albu-Schäffer, C. Ott, , U. Frese, and G. Hirzinger. Cartesian impedance control of redundant robots: Recent results with the dlr-light-weight-arms. *submitted to ICRA*, 2003.
4. B. Brogliato, R. Ortega, and R. Lozano. Global tracking controllers for flexible-joint manipulators: a comparative study. *Automatica*, 31(7):941–956, 1995.
5. F. Caccavale, C. Natale, B. Siciliano, and L. Villani. Six-dof impedance control based on angle/axis representations. *IEEE Transactions on Robotics and Automation*, 15(2):289–299, 1999.
6. S. Chen and I. Kao. Theory of stiffness control in robotics using the conservative congruence transformation. *International Symposium of Robotics Research*, pages 7–14, 1999.
7. C.Ott, A. Albu-Schäffer, A. Kugi, and G. Hirzinger. Impedance control for flexible joint robots. *Submitted to ICRA 2003*, 2002.
8. G. Hirzinger, A. Albu-Schäffer, M. Hähnle, I. Schaefer, and N. Sporer. On a new generation of torque controlled light-weight robots. *IEEE International Conference of Robotics and Automation*, pages 3356–3363, 2001.
9. N. Hogan. Mechanical impedance of single- and multi- articular systems. In J.M. Winters and S. Woo, editors, *Multiple Muscle Systems: Biomechanics and Muscle Organization*, pages 149–163. Springer-Verlag, New York, 1990.
10. O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *RA*, RA-3:43–53, 1987.
11. R. Koeppe and G. Hirzinger. From human arms to a new generation of manipulators: Control and design principles. *ASME Int. Mechanical Engineering Congress*, 2001.
12. T. Lin and A.A. Goldenberg. Robust adaptive control of flexible joint robots with joint torque feedback. *IEEE International Conference of Robotics and Automation*, RA-3(4):1229–1234, 1995.
13. A. De Luca. Feedforward/feedback laws for the control of flexible robots. *IEEE International Conference of Robotics and Automation*, pages 233–240, 2000.
14. C. Ott, A. Albu-Schäffer, and G. Hirzinger. Comparison of adaptive and nonadaptive tracking control laws for a flexible joint manipulator. *IROS*, 2002.
15. J. K. Salisbury. Active stiffness control of a manipulator in cartesian coordinates. *19th IEEE Conference on Decision and Control*, pages 83–88, 1980.

16. M. Spong. Modeling and control of elastic joint robots. *IEEE Journal of Robotics and Automation*, RA-3(4):291–300, 1987.
17. M. Spong. Adaptive control of flexible joint manipulators: Comments on two papers. *Automatica*, 31(4):585–590, 1995.
18. S. Stramigioli and H. Bruyninckx. Geometry and screw theory for constrained and unconstrained robot. *Tutorial at ICRA*, 2001.
19. S. Sugano. Human-robot symbiosis. *Workshop on Human-Robot Interaction, ICRA*, 2002.
20. P. Tomei. A simple PD controller for robots with elastic joints. *IEEE Transactions on Automatic Control*, 36(10):1208–1213, 1991.
21. M. Zinn, O. Khatib, B. Roth, and J.K. Salisbury. A new actuation approach for human friendly robot design. *Int. Symp. on Experimental Robotics, Ischia*, 2002.

Toward the Control of Self-Assembling Systems

Eric Klavins

California Institute of Technology, Pasadena, CA 91125, USA

Abstract. In recent work [4], capillary forces between tiles floating on a liquid-liquid interface are used to direct a self-assembly process. By carefully arranging the wetabilities of the edges of the tiles, regular arrays of various shapes spontaneously form when the tiles are gently shaken. It is difficult, however, to avoid flaws in the assembled aggregates and to assemble terminating and asymmetric structures. In this paper, we suppose that the wetability properties of the tiles, and therefore the capillary forces , can be controlled. In particular, we introduce a simple model of a "waterbug" shaped tile and derive the equations of motion for a system of such tiles from a model of the lateral forces between two floating colloidal particles. We then explore the possibilities for control in this setting and present some initial forays into addressing the above difficulties.

1 Introduction

Self-assembly plays a crucial role in many phenomena in chemistry, physics and cell biology. Self assembling systems occur when many similar parts (molecules, colloids, tiles) are placed in an environment that thermodynamically favors their forming regular arrays. Recent work on mesoscale self-assembly [4] has employed capillary forces between millimeter scale tiles floating on a liquid-liquid interface to direct the assembly process. By carefully arranging the wetabilities of the edges of the tiles, regular arrays of various shapes spontaneously form as the system is gently shaken. The straightforward construction of such tile systems allows researchers to more easily observe the assembly process than in chemical systems. Mesoscale self-assembly may also have practical applications such as the assembly of three dimensional memory chips [5] or of computer displays [24].

Two main difficulties arise in self-assembling systems. First, it is difficult to avoid flaws in the assembled aggregates due to malformed parts or, more interestingly, local minima in the energy landscape. Second, it can be difficult to assemble terminating and asymmetric structures for arbitrary part morphologies. For example, square parts with all hydrophobic edges form square lattices that terminate only when all parts are incorporated. Designing an open loop system that terminates at, for example, multiple 10×10 arrays of such parts is not possible.

One way to address these problems is to suppose that the parts themselves have some control over in which binding interactions they participate. In a

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 153–167, 2003

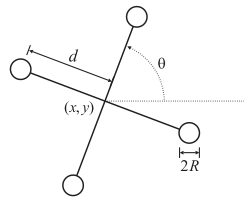


Fig. 1. The *waterbug* tile model: Two lightweight beams of equal length joined at their centers at right angles. To the ends of each beam are attached buoyant particles whose wetability (see Figure 3) can be controlled. When the tile floats on the surface of the water, the meniscus formed by the "feet" results in attraction to or repulsion from the feet of other tiles.

biochemical environment, we might suppose that this control is mediated by organelles in the cell. In mesoscale self-assembly, we might suppose that tiles have control over the wetabilities of their edges and also some idea of their local state.

Of course, other models of self-actuated, robotic parts using magnets or latches are readily imagined. In this paper, however, we propose a model of a tile, dubbed the *waterbug* (Figure 1), we believe will be useful in developing an initial analytical understanding of the dynamics and control of self-assembling systems. The waterbug consists of two beams held at right angles with buoyant particles (its feet) at the ends of the beams. We suppose that the wetabilities of the feet can be varied [22,13] so that their interactions with other feet can be controlled. This model is conceptually simple in that we can, using what is known about particles floating on liquid-fluid interfaces (Section 3.1 which summarizes [16]), derive fairly accurate equations to describe its motion and its interaction with other tiles (Section 3.2).

Remark: We emphasize that the details of the waterbug model are not important. The forces that attract particles together could equally well be magnetic, electrostatic, even gravitational, and the qualitative behavior of the system would be the same: The tiles readily form aggregates that are, qualitatively, quite similar to those observed in [4] as illustrated in Figure 2.

The opportunities for new modeling and control techniques in self-assembly are substantial. In general, the main difficulties are the estimation of global state from local information and distributed control based on local and incomplete information. After introducing the waterbug model, we suggest some initial ideas for how to address these difficulties. In Sections 3.3 and 3.4, we examine the dynamics of the assembly model we propose and our initial understanding of what kinds of aggregates are possible. In Section 4.1, we demonstrate that by controlling the wetabilities of the feet of the waterbug tile, we can avoid certain undesirable subassemblies that would have been energetically favorable if the wetabilities of the feet were static. In Section

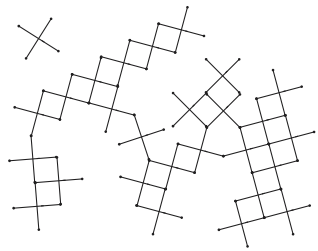


Fig. 2. An example intermediate assembly of waterbug tiles. Without control, the tiles arrange themselves into quasi-regular arrays that terminate only when all tiles are incorporated. Defects, corresponding to undesirable stable configurations (see for example Figure 6(c)), are also possible.

4.2 we review [9,10] and suggest how the methods used there can be adapted to the present model.

2 Related Work

Research in self-assembly has traditionally been the realm of supramolecular chemistry [14] wherein the formation of regular molecular aggregates by non-covalent binding interactions (such as hydrogen bonding) is studied. Such aggregates form spontaneously due to thermodynamics and chance collisions and are not mediated by chemical reactions. Many examples of these aggregates and their assembly dynamics, from dendrimer formation to chemical contaminant recognition, can be found in the recent issue of the PNAS dedicated to the subject [8]. Molecular self-assembly systems are quite flexible and in fact, in [2,23], arbitrary computations are studied using DNA tiles, reinforcing the view put forth in [14] that self-assembly is as much about structure as it is about information.

Research into aggregates of small particles suggests alternative systems for which interesting self-assembling systems can be easily constructed and observed. According to [6], Perrin was the first (in 1909) to report the phenomenon wherein particles in fluid are attracted to each other by capillary forces. The idea is extended significantly in [4] to systems of tiles with various shapes floating on a liquid-liquid interface. By carefully designing the wetabilities of the edges of the tiles with respect to one of the liquids, a wide variety of arrays of tiles can be made. This work was further extended [20] to reproduce, with tiles and capillary forces, the computational systems investigated in [23]. Chemistry and colloid physics are combined in [15] wherein single strands of DNA are attached to small gold balls which then assemble according to the interactions between complementary strands of DNA. Saitou [21] has explored the logic of *conformational switching* whereing the

binding of a part to a subassembly changes the "shape" and therefore the future binding properties of the parts involved.

The idea of controlling the wetability of surfaces is used in [13] to move a liquid droplet across the surface of a microchip (see also [1]) and in [22] for protein patterning. More generally, capillary forces are being used to a greater and greater extent in the MEMs community for microfluidic manipulation and assembly [7].

In robotics and control, Koditschek was the first (in unrelated work) to propose the idea of assembly as a "game of its pieces" [19]. In this work, parts can move themselves – or they can direct external manipulators to move them – based on artificial potential functions [12]. The robotics community, however, has mainly latched onto this work for its application to robot navigation. Initial forays have also been made toward decentralized versions of this idea [18] that resemble self-assembling systems, but for which analysis is apparently quite difficult.

Despite the innovation of considering parts as autonomous, the approach in [19] is problematic in that the programs that achieve a final desired configuration (assembly) are global. We are concerned with systems where global information is not readily available if for no other reason than that the size of the system is enormous, so that the communication complexity of sharing global information [11] is prohibitively high. To overcome this we have examined decentralized schemes for assembly [9,10], which require only local knowledge and which, statistically, seem to work. The main idea is to *compile* programs for each autonomous part from a specification of the desired final assembly, which in general may be terminating and asymmetric – unlike most uncontrolled tile systems [4]. When the programs are run by each part, copies of the desired assembly form. The method relies on local sensing and control: We augment the natural *thermodynamic programs* of the tiles with internal state and control. The work in [9,10] assumes a certain highly *idealized* part. One motivation for the the present work is to investigate the feasibility of various physically realistic models for autonomous parts that may be able to execute the programs compiled in [9,10].

3 Modeling

3.1 Lateral Capillary Forces Between Floating Particles

Consider the system depicted in Figure 3. Two particles float on a liquid-fluid interface (e.g. water-air). Each particle k forms a meniscus which has slope Ψ_k based on the *wetability* of the particle. If the particle is *hydrophilic*, then $\Psi_k > 0$. Otherwise the particle is *hydrophobic* and $\Psi_k < 0$. The two particles will experience a force along the line connecting them due to the thermodynamic tendency for the interfacial free energy between the liquid and fluid to be minimized. Using the Laplace-Young [17] equation, which describes the shape of the meniscus around the particles, Paunov et al. [16]

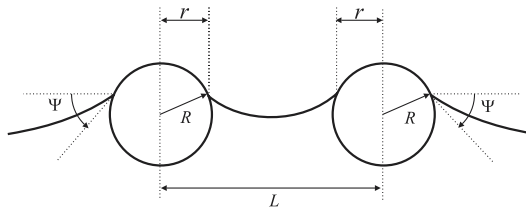


Fig. 3. The model examined in [16]: Two particles floating on a liquid-fluid interface form a meniscus that results in their mutual attraction or repulsion, depending on their respective wetabilities. The waterbug model in Figure 1 uses such particles for its "feet".

determined that the lateral force between two particles at distance L from each other is approximated by

$$F(L) = 2\pi\gamma Q_1 Q_2 \rho K_1(\rho L)[1+\mathrm{O}(\rho^2 R^2)], \tag{1}$$

where γ is a parameter describing the surface tension of the interface; ρ is a parameter (related to γ, the mass densities of the liquid and fluid, and the force of gravity) describing the capillary length; R is the radius of particles; K_1 is the modified Bessel function of the first order; and

$$Q_k \triangleq r_k \sin\Psi_k \tag{2}$$

where r_k is the distance from the center-line of the particle to the interface (see Figure 3). We will assume that $\rho^2 R^2 \ll 1$ so that this approximation is valid. However, we will be noncommittal about the rest of these parameters in this paper.

Using the fact that $K_1(\rho L) > 0$ and the definition of Q_k, we see that two particles of similar wetabilities (both hydrophobic, or both hydrophilic) will be attracted to each other, whereas two particles of differing wetabilities will be repelled from each other.

To simplify notation, define $c_{1,2} \triangleq 2\pi\gamma Q_1 Q_2$. Using the fact that

$$\frac{\partial}{\partial x} K_0(x) = -K_1(x)$$

where K_0 is the modified Bessel function of zeroth order, we see that (1) is a gradient field with potential energy function

$$U_{pre}(L) = -c_{1,2} K_0(\rho L). \tag{3}$$

This function is depicted (with two variations of it) in Figure 4. Note in particular that $U_{pre}(L)$ is singular at $L = 0$.

3.2 The Waterbug Model

We are in general interested in systems of tiles, such as those examined in [4], whose *edges* have certain wetability properties. To understand such systems using what we know about particle interactions, we introduce a model "waterbug" tile (shown in Figure 1) that is similar to the tiles studied experimentally in, for example, [4], with respect to its possible interactions with other tiles. The waterbug model consists of two beams of negligible mass and of length $2d$ joined at their centers at right angles. To the ends of each beam are attached buoyant particles (which we will call *feet*) of radius R, mass m and with wetabilities described by Ψ_1, Ψ_2, Ψ_3 and Ψ_4.

We suppose that the Ψ_j are control inputs to the system; that is, we suppose that a tile is able, somehow [13,1,22], to vary the wetabilities of its feet so that they will be attracted to or repelled by the feet of other tiles according to (1).

Now consider a system consisting of n waterbug tiles. Denote by $q_i = (x_i, y_i, \theta_i)$ the generalized position and orientation of the ith tile and by $w_{i,j} = (u_{i,j}, v_{i,j})$ the actual position of the jth foot of tile i (e.g. $(u_{i,1}, v_{i,1}) = (x_i + d\cos\theta_i, y_i + d\sin\theta_i)$). We denote the full state of the system $(q_1, ..., q_n)$ by $\mathbf{q}$. The potential energy corresponding to two different tiles i and k is

$$U_{tiles}(q_i, q_k) = -\sum_{j=1}^{4}\sum_{l=1}^{4} c_{i,j,k,l} K_0(q||w_{i,j} - w_{k,l}||)$$

where $c_{i,j,k,l} = 2\pi\gamma Q_{i,j} Q_{k,l}$ and $Q_{i,j}$ is the wetability coefficient of the jth foot of the ith tile. We have assumed that no two feet are touching (i.e. that $||w_{i,j} - w_{k,l}|| > 2R$ for all i, j, k and l). The full potential energy of the system[1] is then

$$U(\mathbf{q}) = \sum_{1 \le i \ne k \le n} U_w(q_i, q_k).$$

The kinetic energy of a single tile is $K_i = 4m(\dot{x}_i + \dot{y}_i + d^2\dot{\theta}_i)$, and of the system is $K = \sum_{i=1}^{n} K_i$. We assume that each foot is subject to the force of viscous friction which we model simply by $-k_f \dot{w}_{i,j}$ where $k_f > 0$ is a constant. Setting $L = K - U$ and setting the Lagrangian equal to the sum of the frictional forces, we have that the dynamics of the system are described by

$$\begin{aligned}
8m\ddot{x}_i + \sum_{k=1,k\ne i}^{n} \frac{\partial}{\partial x_i} U_w(q_i, q_k) &= -4k_f \dot{x}_i \\
8m\ddot{y}_i + \sum_{k=1,k\ne i}^{n} \frac{\partial}{\partial y_i} U_w(q_i, q_k) &= -4k_f \dot{y}_i \\
8md^2\ddot{\theta}_i + \sum_{k=1,k\ne i}^{n} \frac{\partial}{\partial \theta_i} U_w(q_i, q_k) &= -4k_f d^2 \dot{x}_i.
\end{aligned} \tag{4}$$

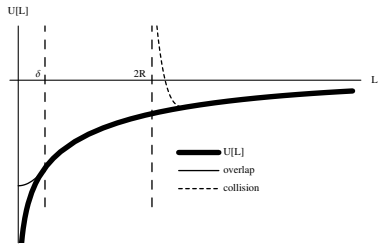


Fig. 4. The potential energy $U_{pre}(L)$ between two particles distance L apart and the modified potentials used to model contact situations. The collision potential, if used, is active when $L < 2R$. The smoothing of the singularity, if used, is active when $L < \delta \ll R$. In both cases a polynomial of appropriate degree is used so that the resulting potential is C^2.

3.3 The Hybrid Dynamics of Assembly Systems

The equations (4) are valid under the assumption that $||w_{i,j} - w_{k,l}|| > 2R$ for all i, j, k and l. When two feet make contact, however, the system dynamics change. There are three obvious ways to describe the system wherein some number of feet are touching.

First, we can define a hybrid automaton [3] with states corresponding to each of the $4n(4n-4)$ possible contact configurations. In each state the dynamics are described by (4) subject to the constraints $||w_{i,j} - w_{k,l}|| = 2R$ for each pair $((i,j),(k,l))$ of contacting feet. Transitions occur between states when new pairs come into contact or when the force (1) holding two feet together is overcome by other forces. This model is appealing because the finite state part mirrors nicely the space of all possible assembly sequences. It is unappealing for analytical and simulation purposes because of possible chattering between states and the neutral stability of equilibria due to the fact that $||w_{i,j} - w_{k,l}|| = 2R$ defines a circle and not a point.

Second, we can modify the potential function U_{pre} in (3) so that a (large) force preventing two particles from overlapping takes affect when $||w_{i,j} - w_{k,l}|| < 2R + \delta$ as in Figure 4. This is very appealing for simulation purposes as it eliminates chatter and is fairly realistic: It has been observed [4] that a thin layer of liquid separates two tiles that are essentially in contact. However, the analysis of stable configurations (possible final aggregate shapes) (Section 3.4) is difficult with this model due to the piecewise and *ad hoc* nature of the modified potential.

Third, we can suppose that two feet can overlap (occupy the same place). If $R \ll d$, then two feet in contact is essentially the same as two feet in the same place. For purposes of simulation, we smooth out the singularity in U_{pre}

[1] We assume that the potentials are simply additive as, for example, with particle systems under the influence of gravity. It is not known whether this assumption can be shown from first principles from the Laplace-Young Equation.

with an appropriate polynomial potential that is in effect when $||w_{i,j} - w_{k,l}|| < \delta \ll 2R$ (see Figure 4). In this manner we are able to numerically simulate systems of up to 40 tiles in a few hours. Further supposing that U_{tiles} is in effect only when tiles are "close by" results in efficient simulation of even larger systems.

For analytical purposes, we proceed much as in the first case and consider a system of waterbug tiles linked by certain feet to form a two dimensional mechanical linkage of tiles (as in Figures 5 and 6). In particular, let $V = \{1, ..., n\} \times \{1, ..., 4\}$ correspond to the set of all $4n$ feet and let E be an equivalence relation over V that specifies which feet are collocated. The submanifold corresponding to the constraints imposed by E is

$$M_E \triangleq \{\mathbf{q} \ : \ [(i,j),(k,l)] \in E \Rightarrow w_{i,j}(q_i) = w_{k,l}(q_k)\}.$$

(It may be that $M_E = \emptyset$). The equations of motion are then similar to (4) except projected onto M_E. To obtain them, however, we can not use $U(\mathbf{q})$ since it is singular if E is nontrivial. Thus, define the set of non-collocated feet by

$$\mathcal{F}_E \triangleq \{(i,j,k,l) \ : \ [(i,j),(k,l)] \notin E\}$$

and set

$$U_E(\mathbf{q}) \triangleq - \sum_{(i,j,k,l) \in \mathcal{F}_E} c_{i,j,k,l} K_0(q||w_{i,j}(q_i) - w_{k,l}(q_k)||).$$

The equations of motion of the partially assembled system given by E are then

$$\frac{d}{dt}\left(\frac{\partial(K - U_E)}{\partial \dot{\mathbf{q}}}\right) - \left(\frac{\partial(K - U_E)}{\partial \mathbf{q}}\right) = F_{friction}. \tag{5}$$

subject to the constraints

$$[(i,j),(k,l)] \in E \Rightarrow w_{i,j}(q_i) = w_{k,l}(q_k) \tag{6}$$

for all i, j, k and l.

3.4 Stable Assemblies

Given an assembly specified by E, any configuration $\mathbf{q}^*$ for which $-\nabla U_E(\mathbf{q}^*)$ is normal to M_E at $\mathbf{q}^*$ is an *equilibrium* of (5) subject to (6). We have:

Proposition 1. *If all points in M_E are equilibria of (5) subject to (6), then the assembly given by E is stable (modulo rotation and translation).*

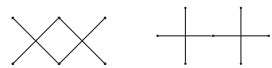


Fig. 5. The two possible assembled configurations of two waterbug tiles (up to symmetry and excluding overlapping crossbars). (a) is stable, whereas (b) is unstable but converges to (a).

Fig. 6. The three possible stable assembled configurations of three waterbug tiles (up to symmetry and excluding overlapping crossbars). In similar tile systems, configurations like (a) and (b) are desirable, while (c) may not be: If a system with a larger number of tiles were "seeded" with (c), irregular aggregates with holes would form (as in Figure 2).

In particular, for all $\Upsilon_{i,j}$ positive, the criterion in Proposition 1 is simply another way of stating that the mechanical linkage described by E is in fact rigid. Non-rigid assemblies can have unstable at equilibrium points. For example, the assemblies in Figures 5(a) and 6(a)–(c) are stable. The assembly in Figure 5(b) is an unstable equilibrium (with respect to $\theta_1 - \theta_2$).

It is also possible that an equilibrium of a non-stable assembly is itself stable. In our simulations, such configurations seem to occur exclusively when the beams of two different tiles overlap – although this remains to be confirmed.

Given an assembly specified by E, the system described by (5) and (6) will have singularities at any point $\mathbf{q}^* \in M_E$ such that, for some i, j, k, and l, $w_{i,j}(q_i^*) = w_{k,l}(q_k^*)$ since

$$\lim_{L \to 0} U_p(L) = -\infty.$$

Thus, at such a point, the equations of motion are not locally Lipschitz and solution trajectories are not guaranteed to be unique. Nevertheless, such configurations can be shown to be locally attracting using a variation of Lyapunov's direct method. For example, configuration 5(b) has two attracting configurations isomorphic to 5(a).

In [4], the observation is made that some configurations are *less stable* than others. They use this property to their advantage by adding enough energy to the system (by gently shaking it) so that undesirable assemblies are energetically unfavorable while desirable ones are. We have not yet developed analytical means for determining the energy required to break up various classes of waterbug assemblies – although we have observed in our simulations

that, for example, the assembly in Figure 6(c) is less stable (in the presence of disturbances resulting from collisions with other tiles and assemblies) than the assemblies in Figures 6(a) or (b).

A complete catalog of possible waterbug assemblies is not known. More generally, given a tile morphology, an automatic method for determining the stable assemblies of a soup of such tiles (and their energies) is not known.

4 Discussion

The essential control task for an assembly system, in its most basic form, is to bring the system from an initial assembly E_0 to a final assembly E_N that is one of some prespecified family of final (stable) assemblies $\mathcal{E}_{final}$. Thus, we desire a sequence $E_0, ..., E_N$ where E_0 is the identity relation, $E_N \in \mathcal{E}_{final}$ for any t, $E_t \subseteq E_{t+1}$. Furthermore, there may be some set $\mathcal{B}$ of undesirable configurations (an *obstacle* in a sense) so that we require that $E_t \notin \mathcal{B}$ for any $t \in \{0, ..., N\}$ (see Figure 7).

We have so far supposed that the actuation model for these systems is the variable wetabilities of the tile feet. Of course, less direct (and possibly more practical) actuation models exist (such as shaking the system and thereby keeping its energy at a certain level unfavorable to the formation of assemblies in $\mathcal{B}$). For the rest of this section though, we assume the first sort of actuation. Of the many control tasks one might imagine for these self-assembling systems, we briefly consider two: eliminating defects and obtaining terminating structures.

4.1 Eliminating Defects

Figure 2 shows an intermediate assembly containing the (possibly undesirable) sub-configuration shown in Figure 6(c). For a system of three tiles, the domain of attraction of the configuration in Figure 6(c) is small but nevertheless measurable. For example, for tiles with all-like wetabilities, the initial configuration in Figure 7(a) leads to Figure 6(c). However, an open loop controller defined by

$$\Psi_{i,2}(t) = \Psi_{i,4}(t) = \begin{cases} 0 \text{ if } t < t_1 \\ \Psi \text{ otherwise} \end{cases}$$

and $\Psi_{i,1}(t) = \Psi_{i,3}(t) = \Psi$ for all $i \in \{1, 2, 3\}$ where $\Psi > 0$ is a constant, results in a system that avoids Figure 6(c). Essentially, with $t < t_1$, Figure 6(c) is not an attracting configuration. Thus, with the wetabilities arranged as they are with $t < t_1$, the configuration in Figure 7(c) is obtained. Once this is arrived at, there is no (energetically decreasing) way to get to 6(c) and therefore, with all wetabilities positive (for $t \geq t_1$), Figure 7(e) is obtained.

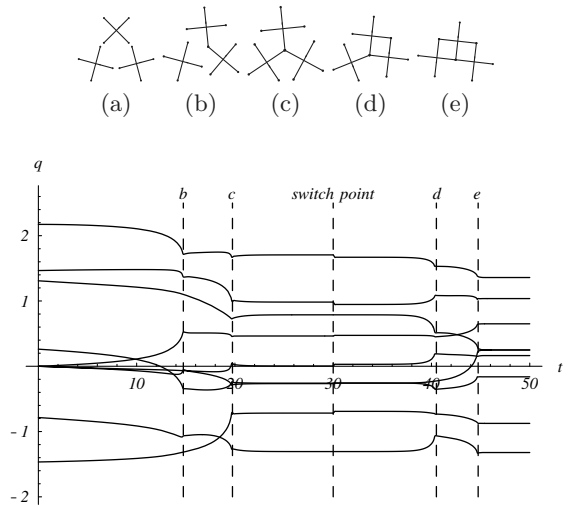


Fig. 7. Trajectory ($x_1(t)$, $y_1(t)$, ...) of a three tile system with initial conditions (a) very close to 6(c), controlled to avoid this configuration. From time $t = 0$ to time $t = 30$, feet $\Psi_{i,2} = \Psi_{i,4} = 0$ and $\Psi_{i,1} > 0$ and $\Psi_{i,3} > 0$ for $i \in \{1, 2, 3\}$. For $t > 30$, all $\Psi_{i,j} > 0$ for all i and j. Times (b)-(d) show intermediate configurations.

4.2 Terminating Structures

In general a given (uncontrolled) tile shape will result in a certain family of structures that can be assembled. This family is the closure of the operation of adding a new part to a given assembly, if possible. The waterbug tile with all hydrophobic feet has an infinite family of assemblies: a new tile can always be added onto any stable assembly to make a new stable assembly. The problem of generating terminating structures is to add control algorithms to the tiles so that the assembly family formed consists only of the desired finite structures and their subassemblies (as, for example, in Figure 8).

In [9,10] we describe a method for assembling such terminating structures out of 2D, self-actuated disk-shaped parts. In particular, any tree-shaped structure (directed, acyclic graph) can be assembled and the local rules directing the assembly can be automatically and efficiently synthesized. The method is shown to be correct under certain assumptions about the assembly dynamics. It requires that each tile be able to sense the discrete state (defined below) of tiles near it and that it be able to change the wetabilities of its feet. In the rest of this section, we outline how a simple variant of this method can be used to assemble a certain tree-shaped assembly: number 6 in Figure 8. Although the controller described here is constructed by hand for a particular assembly, the method in [9,10] can easily be adapted to automatic synthesis of controllers for any tree-shaped assembly.

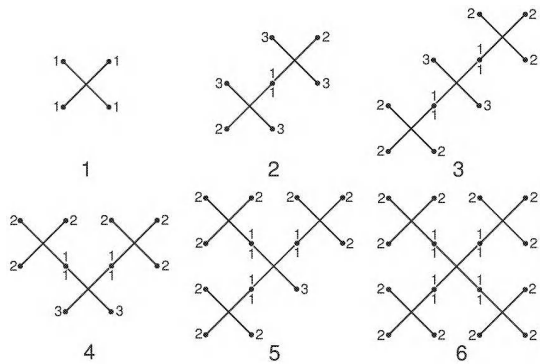


Fig. 8. An example terminating structure (number 6) and its subassemblies up to isomorphism. Each subassembly is given an index (1–6) and each foot of each tile is given a *role* identifier. Thus, each foot in a subassembly has a discrete state corresponding to its subassembly-role pair. The state is used to determine which binding interactions between pairs of tiles should occur.

A desired assembly of n tiles is specified by an equivalence relation E_{spec} over $1, ..., n \times \{1, ..., 4\}$ as in Section 3.3. The first step in generating rules that result in the assembly of copies of E_{spec} is to form a list $\mathcal{E}$ of its subassemblies (including E_{spec}). For the present example, this list is shown in Figure 8 as subassemblies 1 through 6. The next step is to identify the *role* of each foot of each tile in each subassembly. These are given by the small numbers located next to the feet in the figure.

Given $\mathcal{E}$ and an assignment of roles, the discrete state of a waterbug foot is defined to be its *subassembly-role* pair (s, r). For example, any tile that is not joined to another tile has subassembly-role pair $(1, 1)$. A foot in a two-tile assembly will have subassembly-role pair $(2, 1)$, $(2, 2)$ or $(2, 3)$. We now construct an algebra of subassemblies.

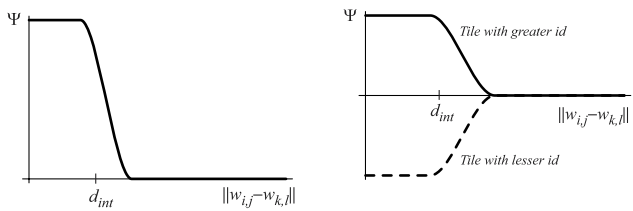
Definition 1. Given (s_1, r_1) and (s_2, r_2), we define their *join* $(s_1, r_1) \oplus (s_2, r_2)$ to be s if the action of joining subassemblies s_1 and s_2 at feet with roles r_1 and r_2 results in a subassembly isomorphic to a subassembly in $\mathcal{E}$ with index s. Otherwise, $(s_1, r_1) \oplus (s_2, r_2) = \perp$. If $(s_1, r_1) \oplus (s_2, r_2) \neq \perp$, then the join is called *valid*.

In our running example, the valid joins are given by

$$(1,1) \oplus (2,2) = 3 \qquad (1,1) \oplus (2,3) = 4 \qquad (1,1) \oplus (3,3) = 5$$
$$(1,1) \oplus (4,3) = 5 \qquad (1,1) \oplus (5,3) = 6.$$

Note that because of the simplicity of the example, $(1, 1)$ is a part of every valid join although, in general, this is not the case.

Next we define discrete update rules for each possible join. For example, if two tiles join via feet with states $(1, 1)$ and $(2, 2)$, then the roles of the



(a) Attraction assignment (b) Repulsion assignment

Fig. 9. Wetability assignments used by two waterbug tile feet as a function of their distance. If the two tiles are to attract (according to the rules described in Section 4.2), then they use the assignment function in (a). Otherwise they use the functions in (b) where the foot whose tile has the greater id forms a positive meniscus and the foot whose tile has the lesser id forms a negative meniscus, resulting in a repulsive force between the two feet (as described in Section 3.1).

subassemblies involved change according to:

$$(1,1),(2,2) \xrightarrow{(1,1)\oplus(2,2)} \begin{cases} (3,1) \text{ if interacting foot} \\ (3,2) \text{ else} \end{cases}$$

$$(2,1) \xrightarrow{(1,1)\oplus(2,2)} (3,1)$$

$$(2,3) \xrightarrow{(1,1)\oplus(2,2)} \begin{cases} (3,3) \text{ if interacting tile} \\ (3,2) \text{ else} \end{cases}$$

With the join algebra and the update rules, we can define control laws for each foot to run. We assume that the tiles are being gently shaken to encourage interactions between feet. If two feet (i, j) and (k, l) come in close proximity, they change their wetabilities to either attract or repel according to whether their join is valid (see Figure 9). If two feet join, the subassembly-role pairs of the feet in the subassemblies involved are updated according to the update rules just described. Waterbug feet may also need to repel other feet in the same subassembly to which they are not directly attached. If no interaction is desired (there are no nearby feet), the wetability assignment is set to 0 (no meniscus).

The method in [9,10] also accounts for deadlock situations arising either geometrically (some tiles are blocking other tiles from useful interaction sites) or logically (for example, all subassemblies with index 1 are used up before any subassemblies of index 6 form). The deadlock mechanism is simple: If no interactions have occurred with a non-final subassembly for longer than some predetermined time, the assembly breaks up (using the repulsion assignment in Figure 9) and tiles formerly joined "ignore" each other (so that different interactions may occur). These rules are proved to assemble a maximum number of final assemblies, in a *perfect model of interaction* (i.e. different from the model described here), in [9].

We are presently exploring possible implementations of the assembly algorithm described above – with tiles or robots – and hope to report on our results in forthcoming publications.

5 Conclusion

We have introduced a model of a self-assembling tile system based on the physics of capillary forces. The tiles in the model are amenable to control actions via changing their wetabilities. We have demonstrated a simple open loop controller that can be used to avoid defects in the assembly process and have suggested a way to use previous work [9,10] to direct the assembly process even further to build arbitrary terminating structures.

There are many other avenues and opportunities for the control of assembly systems based on different interaction and binding mechanisms, alternate actuation paradigms, and so on. We hope that the model we propose here will serve as a good starting point for such investigations.

Acknowledgments.
The author would like to thank Richard Murray, Dan Koditschek and Karl Böhringer for their advice and suggestions regarding this paper and my research in self-assembly in general.

This research is supported in part by DARPA grant number F33615-98-C-3613 and by AFOSR grant number F49620-01-1-0361.

References

1. N. L. Abbott, C. B. Gorman, and G. M. Whitesides. Active control of wetting using applied electrical potentials and self-assembled monolayers. *Langmuir*, 11(1):16–18, 1995.
2. L. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266:1021–1024, 1194.
3. R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In *Hybrid Systems I*, volume 763 of *LNCS*, pages 209–229. Springer, 1993.
4. N. Bowden, A. Terfort, J. Carbeck, and G. M. Whitesides. Self-assembly of mesoscale objects into ordered two-dimensional arrays. *Science*, 276(11):233–235, April 1997.
5. T. L. Breen, J. Tien, S. R. J. Oliver, T. Hadzic, and G. M. Whitesides. Design and self-assembly of open, regular, 3D mesostructures. *Science*, 284:948–951, 1999.
6. N. D. Denkov, O. D. Velev, P. A. Kralchevsky, I. B. Ivanov, H. Yoshimura, and K. Nagayama. Mechanism of formation of two-dimensional crystals from latex particles on substrates. *Langmuir*, 8:3183–3190, 1992.

7. A. Greiner, J. Lienemann, J. G. Korvink, X. Xiong, Y. Hanein, and K. F. Böhringer. Capillary forces in micro-fluidic self-assembly. In *Fifth International Conference on Modeling and Simulation of Microsystems (MSM'02)*, pages 22–25, April 2002.
8. J. Halpern, editor. *Proceedings of the National Academy of Sciences: Special Issue on Supramolecular Chemistry and Self-Assembly*, volume 99:8. National Academy of Sciences, April 2002.
9. E. Klavins. Automatically synthesized controllers for distributed assembly: Partial correctness. In *Proceedings of the Conference on Cooperative Control and Optimization*. Kluwer, Gainsville, FL, November 2001.
10. E. Klavins. Automatic synthesis of controllers for assembly and formation forming. In *International Conference on Robotics and Automation*, Washington DC, May 2002.
11. E. Klavins. Communication complexity of multi-robot systems. In *Workshop on the Algorithmic Foundations of Robotics*, Nice, France, December 2002.
12. D.E. Koditschek. An approach to autonomous robot assembly. *Robotica*, 12:137–155, 1994.
13. J. Lee and C.-J. Kim. Surface-tension-driven microactuation based on continuous electrowetting. *Journal of Microelectricalmechanical Systems*, 9(2):171–180, 2000.
14. J.-M. Lehn. *Supramolecular Chemistry: Concepts and Perspectives*. Wiley, 1995.
15. R. C. Mucic, J. J. Storhoff, C. A. Mirkin, and R. L. Letsinger. DNA-directed synthesis of binary nanoparticle network materials. *Journal of the American Chemical Society*, 120:12674–12675, 1998.
16. V. N. Paunov, P. A. Kralchevsky, N. D. Denkov, and K. Nagayama. Lateral capillary forces between floating submillimeter particles. *Journal of Colloid and Interface Science*, 157:100–112, 1993.
17. H. M. Princen. Wetability and contact angles. In E. Matijević, editor, *Surface and Colloid Science*, volume 2, pages 1–153. Wiley, 1969.
18. H. Reif and H. Wang. Social potential fields: A distributed behavioral control for autonomous robots. In *Proceedings of the 1994 Workshop on the Algorithmic Foundations of Robotics*. A.K.Peters, Boston, MA, 1995.
19. E. Rimon and D.E. Koditschek. Exact robot navigation using artificial potential fields. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, October 1992.
20. W. K. Rothemund. Using lateral capillary forces to compute by self-assembly. *PNAS*, 97(3):984–989, 2000.
21. K. Saitou. Conformational switching in self-assembling mechanical systems. *IEEE Transactions on Robotics and Automation*, 15(3):510–520, June 1999.
22. Y. Wang, X. Cheng, Y. Hanein, B. D. Ratner, and K. F. Böhringer. Protein patterning with programmable surface chemistry chips. In *Proceedings of the Sixth International Symposium on Micro Total Analysis System (μTAS)*, Nara, Japan, November 2002.
23. E. Winfree. Algorithmic self-assembly of DNA: Theoretical motivations and 2D assembly experiments. *Journal of Biomolecular Structure and Dynamics*, 11(2):263–270, May 2000.
24. H.-J.J. Yeh and J.S. Smith. Fluidic assembly for the integration of GaAs light-emitting diodes on Si substrates. *IEEE Photonics Technology Letters*, 6:706–708, 1994.

Towards Abstraction and Control for Large Groups of Robots

Calin Belta and Vijay Kumar

University of Pennsylvania, GRASP Laboratory, 3401 Walnut St., Philadelphia, PA 19104, USA

Abstract. This paper addresses the problem of controlling a large number of robots required to accomplish a task as a group. We propose an abstraction based on the definition of a map from the configuration space of the robots to a lower dimensional manifold, whose dimension does not scale with the number of robots. The task to be accomplished by the team suggests a natural feedback control system on the group manifold. We show that, if mean and covariance matrix are chosen as group variables for fully actuated robots, it is possible to design decoupling control laws, *i.e.,* the feedback control for a robot is only dependent on the state of the robot and the state of the group, therefore the communication necessary to accomplish the task is kept to a minimum.

1 Introduction

There has been a lot of interest in cooperative robotics in the last few years, triggered mainly by the technological advances in control techniques for single vehicles and the explosion in computation and communication capabilities. The research in the field of control and coordination for multiple robots is currently progressing in areas like automated highway systems, formation flight control, unmanned underwater vehicles, satellite clustering, exploration, surveillance, search and rescue, mapping of unknown or partially known environments, distributed manipulation, and transportation of large objects.

In this paper, we consider the problem of controlling a large number of robots required to accomplish a task as a group. For instance, consider the problem of moving 100 planar robots with arbitrary initial positions through a tunnel while staying grouped so that the distance between each pair does not exceed a certain value. The simplest solution, generating reference trajectories and control laws for each robot to stay on the designed trajectory, is obviously not feasible from a computational viewpoint. It is desired to have a certain level of abstraction: the motion generation/control problem should be solved in a lower dimensional space which captures the behavior of the group and the nature of the task.

For example, the robots can be required to form a *virtual structure*. In this case, the motion planning problem is reduced to a left invariant control system on $SE(3)$ (or $SE(2)$ in the planar case), and the individual trajectories are

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 169–182, 2003

$SE(3)$ ($SE(2)$) - orbits [1]. The literature on stabilization and control of virtual structures is rather extensive. Most of the recent works model formations using *formation graphs*, which are graphs whose nodes capture the individual agent kinematics or dynamics, and whose edges represent inter-agent constraints that must be satisfied [2,8,9,7]. Characterizations of rigid formations can be found in [3,1]. The controllers guaranteeing local asymptotic stability of a given rigid formation are derived using Lyapunov energy-type functions [7]. More flexibility is added to the formation if *virtual leaders* are defined [5]. It is interesting to note that in all these works on rigid formations local asymptotic stability can be achieved by decentralized controllers using local information. Moreover, the equilibria exhibit $SE(l)$, $l = 1, 2, 3$ symmetry and also expansion/contraction symmetries. These symmetries can be used to decouple the mission control problem into a formation keeping subproblem and a formation maneuver subproblem as in [6].

In many applications, like swarming, the virtual structure constraint might be too much, or simply not appropriate. The abstraction we propose in this paper involves the definition of a map from the configuration space of the robots to a lower dimensional *group manifold*. We require that the dimension of the group manifold do not scale with the number of robots and an arbitrary element of the group manifold captures the behaviour of the ensemble as a group, *i.e.*, it has a *behavioral* significance. The next step is to equip the group manifold with a vector field, which is built based on two types of restrictions. First, its flow lines should define the desired time evolution of the group, in accordance with a given task. Second, the robots should be able to move as a group in the desired fashion given the possible underactuation / nonholonomy constraints. If these two problems are solved , we propose that the possible remaining degrees of freedom from the individual control laws be used to achieve two more goals. First, it is desired to keep the amount of inter - robot communication in the overall control architecture to a minimum by use of *partial state feedback*. Ideally, we want to achieve *decoupled* architectures, *i.e.,* the control law of a robot only depends on its own state and the low dimensional state of the team from the group manifold. Second, the energy spent by the group to achieve the given task should be kept to a minimum.

In this paper we only consider fully-actuated planar robots abstracted to mean and covariance of the positions with respect to some reference frame. We prove that in this case the *decoupling* control vectors are also *minimum energy* controls. Illustrative examples of trajectory tracking on the group manifold and globally asymptotic stabilization to a point on the group manifols are included.

2 Definitions and Problem Formulation

In this section, we reformulate the ideas presented in Section 1 in a mathematical form.

Consider N robots with states q_i belonging to manifold Q_i. q_i will be interpreted as both a generic element on an abstract manifold or as coordinates of the element, depending on the context. The kinematics of each robot are defined by drift free control distributions on Q_i:

$$\Delta_i : Q_i \times U_i \to TQ_i \tag{1}$$

where U_i is the control space and TQ_i is the tangent bundle of Q_i.

Collect all robot states q_i on a single manifold $Q = q = [q_1, \ldots, q_N]^T$ and equip it with a control distribution Δ obtained from the individual control distributions through direct sum:

$$Q = \prod_{i=1}^{N} Q_i, \;\; \Delta : Q \times U \to TQ, \;\; \Delta = \bigoplus_{i=1}^{N} \Delta_i \tag{2}$$

where U is the Cartesian product of the control spaces $U = \prod_{i=1}^{N} U_i$. We will refer to (2) as the *product control system*. Allow to recover the states and the control distributions of the individual agents by use of the canonical projection on the ith agent:

$$\pi_i : Q \to Q_i, \;\; \pi_i(q) = q_i, \;\; d\pi_i : TQ \to TQ_i, \;\; d\pi_i(\Delta) = \Delta_i \tag{3}$$

Note that relations of type (1) are general enough to accomodate individual underactuation constraints, which are all captured in (2). Also, the notation $d\pi_i$ from (3) does not necessarily stand for the differential of π_i, it just represents an operator to recover individual control distributions.

We need the following definitions before formulating the problem:

Definition 1 (Group Abstraction). Any surjective submersion

$$\phi : Q \to G, \;\; \phi(q) = g$$

so that the dimension n of the group manifold G is not dependent on the number of the robots N is callled a group abstraction.

Definition 2 (Group Behavior). Any vector field X_G on G is called a group behavior.

Given a set of N robots with control systems (1) and corresponding product control system (2), we want to find solutions to the following problems:

Problem 1 (Group Abstraction). Determine a physically meaningful group abstraction which captures the behaviour of the ensamble of robots as a group, *i.e.*, it has a *behavioral* significance.

If the map ϕ is determined, the next step is to solve the motion generation (control) problem on the small dimensional manifold G so that a given task is accomplished by the robots as a team:

Problem 2 (Group Behavior). Design a group behavior X_G on G, so that the flow lines of X_G define the desired time evolution of the group and there exist $X_Q \in \Delta$ on TQ which are pushed forward to X_G through the map ϕ.

Note that Problems 1 and 2 can actually be seen as an input - output linearization problem [4] for the control system (2) with output $g = \phi(q)$. The total relative degree is $\dim(Q) - n$ since each robot is kinematically controlled. The vector field X_G guarantees some desired behavior of the output (which we call group variable) g, which will, of course, guarantee its boundness. Now the hardest problem, as usual in input - output linearization, is calculating and stabilizing the internal dynamics. This would imply, in general, finding the appropriate coordinate transformation separating the internal dynamics from output dynamics, calculating the corresponding zero dynamics and studying its stability. To avoid this, we try to define the output map so that bounds on output would easily imply bounds on the state, so it will not be necessary to explicitly calculate the internal dynamics.

It is also desired to keep the amount of inter - robot communication in the overall control architecture to a minimum, by use of *partial state feedback*. Ideally, we want to achieve a *decoupled* control architecture, *i.e.,* the control law of a robot only depends on its own state and the low dimensional state of the team from the group manifold:

Problem 3 (Decoupling). From the set of vector fields $X_Q \in \Delta$ on Q that are pushed forward to X_G, we want to identify a subset X_Q^* whose projection on TQ_i, $d\pi_i(X_Q^*)$ $i = 1, \ldots, N$ is only dependent on q_i and g.

Pictorially, the desired control architecture combining abstraction and partial state feedback features is given in Figure 1.

On the solution of Problem 2, the vector fields X_Q and X_G are related by

$$d\phi X_Q = X_G \tag{4}$$

where $d\phi$ is the tangent (differential) of the map ϕ. First, note that, for the problem to be well defined, $\phi : Q \to G$ should be a surjective submersion. This condition, together with (4) guarantees that any vector field X_Q satisfying (4) is ϕ - related to X_G. There are two slightly different approaches to finding a solution to Problem 2. If all the robots are fully actuated, *i.e.,* $\dim(\Delta_i) = \dim(Q_i)$, then any group vector field can be implemented by the individual robots. The general solution of (4) is the affine space

$$\mathcal{A} = \mathrm{Ker} d\phi + X_Q^p \tag{5}$$

where the vector field X_Q^p is a particular solution to (4). The degrees of freedom from $\mathrm{Ker} d\phi$ can be used to solve the decoupling Problem 3.

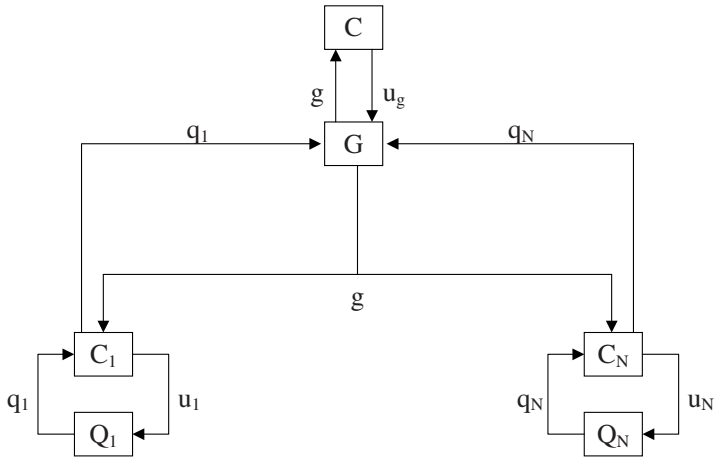


Fig. 1. A decoupled control architecture: the group is controlled on the "abstract" group manifold G; the control law of each robot is only dependent on its own state q_i and the state of the group g.

If some of the robots in the team are underactuated, it is more convenient to start with parameterizing the individual control distributions Δ_i, lift them to Δ and then push forward to G through the map ϕ. The obtained control vector field on G is conveniently parameterized in the robot control variables and exhaustively covers all the possible motions of the group. The individual control parameters can then be used to solve the decoupling Problem 3.

In both fully and under-actuated case, if after decoupling, there are still degrees of freedom left, one can use the minimum energy criterion or some other criteria, depending on the specific application, to choose among the several options.

3 Mean and Covariance Control for Fully Actuated Planar Robots

Assume N fully actuated robots free to move in the plane with position vectors $q_i = [x_i \ y_i]^T \in \mathbb{R}^2$, $i = 1, \ldots, N$ with respect to some reference frame. We will identify all the elements defined in Section 2 and then provide solutions to Problems 1, 2, 3.

Let e_i denote a vector of the standard Euclidean base in some dimension, which will be obvious from the context. Then, each agent is described by the following control system

$$Q_i = \mathbb{R}^2, \ \Delta_i = \text{span}\{e_1, e_2\} = \mathbb{R}^2$$

or, in coordinates,

$$\dot{q}_i = u_i = e_1 u_i^1 + e_2 u_i^2$$

Collecting all the robot states together, we get a $2N$-dimensional product control system

$$Q = \mathbb{R}^{2N},\ \Delta = \mathbb{R}^{2N} = \text{span}\{e_1, \ldots, e_{2N}\}$$

which in coordinates can be written as:

$$Q = \{q = [q_1^T, \ldots q_N^T]^T,\ q_i \in \mathbb{R}^2, i = 1, \ldots, N\},$$
$$\Delta = \{u = [u_1^T, \ldots u_N^T]^T, u_i \in \mathbb{R}^2\} = \text{span}\{e_1, \ldots, e_{2N}\}$$

with the corresponding projections:

$$\pi_i(q) = q_i,\ d\pi_i(u) = u_i$$

3.1 Group Abstraction

To provide an answer to Problem 1, the group variables that we choose in this paper are sample mean $\mu \in \mathbb{R}^2$ and covariance matrix $\Sigma \in \mathbb{R}^{2\times 2}$:

$$\mu = \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix} = \frac{1}{N}\sum_{i=1}^{N} q_i \tag{6}$$

$$\Sigma = \frac{1}{N}\sum_{i=1}^{N}(q_i - \mu)(q_i - \mu)^T = \begin{bmatrix} \sigma_1 & \sigma_2 \\ \sigma_2 & \sigma_3 \end{bmatrix} \tag{7}$$

i.e., we define the map $\phi : Q \to G \equiv \mathbb{R}^5$ by

$$\phi(q) = g = [\mu^T\ \sigma_1\ \sigma_2\ \sigma_3]^T, \tag{8}$$

where μ is given by (6) and σ_1, σ_2, σ_3 are the entries in matrix Σ as in (7). Explicitly,

$$\sigma_1 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_x)^2 \tag{9}$$

$$\sigma_2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y) \tag{10}$$

$$\sigma_3 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \mu_y)^2 \tag{11}$$

Note that, by the Cauchy-Schwartz inequality, $\sigma_1\sigma_3 \geq \sigma_2^2$. The equality holds if and only if all the points q_i are on a line passing through μ. We will call this the *degenerate* case. This includes the situation when all the agents have the same position: $q_i = \mu$, $\sigma_1 = \sigma_2 = \sigma_3 = 0$. Also, non-degeneracy implies $\sigma_1 > 0$ and $\sigma_3 > 0$, because otherwise the agents would be on a

line parallel with the x or y axis. We will assume the non-degenerate case throughout this paper, which will guarantee that the map ϕ is a submersion, as seen from the proof of Proposition 1 in Section 3.2.

An interesting physical interpretation in terms of a subset of the group variable g is given in Section 4 together with some illustrative examples. If the full vector g is available, one can diffeomorphically map G to another 5 - dimensional manifold parameterized by, let's say μ, θ, a, b, where μ is the center, θ the orientation, and a and b the semiaxes of some spanning ellipsoid. Then, the group manifold would have a product structure $G = SE(2) \times S$, where $SE(2)$ is the Euclidean group in two dimensions parameterized by (μ, θ) and S is a *shape* space parameterized by (a, b). The motion planning problem on the group manifold can then be decomposed into motion planning on the Lie group SE(2) (and benefit from the results in this area [1]) and the shape space S. This approach will be presented in a future paper.

3.2 Decoupling

Since we assumed fully actuated robots, as suggested in Section 2, we can design arbitrary group vector fields describing the time evolution of the group. In this section we study the decoupling Problem 3, for an arbitrary given vector field X_G on the group manifold G. The following proposition is the key result of this section:

Proposition 1. *Let X_G be an arbitrary vector field on G. Then the minimum length solution X_Q^* of (4) is also a decoupling control vector as defined in Problem 3,* i.e., *$d\pi_i(X_Q^*)$ is only dependent on q_i and g for $i = 1, \ldots, N$.*

Proof. The proof is rather involved and only a sketch is presented. The tangent map of ϕ defined by (8), (6), (9), (10), and (11) is given at an arbitrary point $q \in Q$ by the following $5 \times (2N)$ matrix:

$$d\phi = \frac{1}{N} \begin{bmatrix} 1 & 0 & \ldots\ldots & 1 & 0 \\ 0 & 1 & \ldots\ldots & 0 & 1 \\ 2(x_1 - \mu_x) & 0 & \ldots\ldots & 2(x_N - \mu_x) & 0 \\ y_1 - \mu_y & x_1 - \mu_x & \ldots\ldots & y_N - \mu_y & x_N - \mu_x \\ 0 & 2(y_1 - \mu_y) & \ldots\ldots & 0 & 2(y_N - \mu_y) \end{bmatrix} \tag{12}$$

It can be shown that

$$\det(d\phi\, d\phi^T) = \frac{16}{N^5}(\sigma_1 + \sigma_3)(-\sigma_2^2 + \sigma_1\sigma_3)$$

from which we conclude that, in the non - degenerate case, $d\phi$ is full row rank, therefore ϕ is a submersion as required in Problem 1. The minimum length solution X_Q^* of (4) can be computed in the form $X_Q^* = d\phi^T\,(d\phi\, d\phi^T)^{-1} X_G$.

After straightforward but rather tedious calculation, we get the projection $\dot{q}_i = u_i = d\pi_i(X_Q^*)$ along the control directions of the i-th robot in the form:

$$\dot{q}_i = \dot{\mu} + \begin{bmatrix} 2(b_{11}\dot{\sigma}_1 + b_{12}\dot{\sigma}_2 + b_{13}\dot{\sigma}_3) & b_{12}\dot{\sigma}_1 + b_{22}\dot{\sigma}_2 + b_{23}\dot{\sigma}_3 \\ b_{12}\dot{\sigma}_1 + b_{22}\dot{\sigma}_2 + b_{23}\dot{\sigma}_3 & 2(b_{13}\dot{\sigma}_1 + b_{23}\dot{\sigma}_2 + b_{33}\dot{\sigma}_3) \end{bmatrix} (q_i - \mu) \tag{13}$$

where

$$b_{11} = \frac{-\sigma_2^2 + \sigma_1\sigma_3 + \sigma_3^2}{4(\sigma_1 + \sigma_3)(-\sigma_2^2 + \sigma_1\sigma_3)} \tag{14}$$

$$b_{12} = \frac{-2\sigma_2\sigma_3}{4(\sigma_1 + \sigma_3)(-\sigma_2^2 + \sigma_1\sigma_3)} \tag{15}$$

$$b_{13} = \frac{\sigma_2^2}{4(\sigma_1 + \sigma_3)(-\sigma_2^2 + \sigma_1\sigma_3)} \tag{16}$$

$$b_{22} = \frac{4\sigma_1\sigma_3}{4(\sigma_1 + \sigma_3)(-\sigma_2^2 + \sigma_1\sigma_3)} \tag{17}$$

$$b_{23} = \frac{-\sigma_1\sigma_2}{4(\sigma_1 + \sigma_3)(-\sigma_2^2 + \sigma_1\sigma_3)} \tag{18}$$

$$b_{33} = \frac{\sigma_1^2 - \sigma_2^2 + \sigma_1\sigma_3}{4(\sigma_1 + \sigma_3)(-\sigma_2^2 + \sigma_1\sigma_3)} \tag{19}$$

It is easy to see that $d\pi_i(X_Q^*)$ is only dependent on q_i and g for $i = 1, \ldots, N$, so the decoupling Problem 3 is solved. In light of (5), X_Q^* can be seen as a particular solution which is orthogonal to the vector space Ker$d\phi$. Therefore, in this case, solving (4) consisted of finding a nice particular solution and putting to zero all the $2N - 5$ control variables that we had on Ker$d\phi$, *i.e.,* annihilate Ker$d\phi$.

Note that under the non-degeneracy assumption the common denominator of all b_{ij} is non-zero. Moreover, $b_{11} > 0$, $b_{22} > 0$, $b_{33} > 0$.

Equation (13) gives the control law which should be implemented by controller C_i as shown in Figure 1 if the output function ϕ is defined as in (8). At each time instant t, the control system on G acquires all the states q_i, updates its own state g in accordance to (6), (9), (10), (11), flows along its designed control vector field X_G and disseminates its state g to all the robots.

3.3 Group Behavior

As suggested in Section 2, in the absence of individual underactuation (nonholonomy) constraints, we can design arbitrary group vector fields X_G describing the time evolution of the group.

Assume the goal is to move the robots from arbitrary initial positions $q_i(0)$ to final rest positions of desired mean μ^d and covariance σ_1^d, σ_2^d, σ_3^d. Also, the control law for each agent should guarantee asymptotic stability of mean and covariance at the desired values under arbitrary position displacements.

An obvious choice of the control vector field $X_G = [\dot{\mu},\ \dot{\sigma}_1,\ \dot{\sigma}_2,\ \dot{\sigma}_3]$ on the group manifold G is

$$\dot{\mu} = K_\mu(\mu^d - \mu) \tag{20}$$
$$\dot{\sigma}_1 = k_{\sigma_1}(\sigma_1^d - \sigma_1) \tag{21}$$
$$\dot{\sigma}_2 = k_{\sigma_2}(\sigma_2^d - \sigma_2) \tag{22}$$
$$\dot{\sigma}_3 = k_{\sigma_3}(\sigma_3^d - \sigma_3) \tag{23}$$

where $K_\mu \in \mathbb{R}^{2\times 2}$ is a positive definite matrix and $k_{\sigma_{1,2,3}} > 0$. The explicit control law for each agent is obtained by substituting (20), (21), (22), (23) and (6), (9), (10), (11) into (13).

More generally, the task might require the robots to follow a desired trajectory $g^d(t) = [\mu^d(t),\ \sigma_1^d(t),\ \sigma_2^d(t),\ \sigma_3^d(t)]$ on the group manifold G. A control vector field on G can be of the form:

$$\dot{\mu} = K_\mu(\mu^d(t) - \mu(t)) + \dot{\mu}^d(t) \tag{24}$$
$$\dot{\sigma}_1 = k_{\sigma_1}(\sigma_1^d(t) - \sigma_1(t)) + \dot{\sigma_1}^d(t) \tag{25}$$
$$\dot{\sigma}_2 = k_{\sigma_2}(\sigma_2^d(t) - \sigma_2(t)) + \dot{\sigma_2}^d(t) \tag{26}$$
$$\dot{\sigma}_3 = k_{\sigma_3}(\sigma_3^d(t) - \sigma_3(t)) + \dot{\sigma_3}^d(t) \tag{27}$$

Note that (20), (21), (22), (23) (in the stabilization case) or (24), (25), (26), (27) (in the trajectory tracking case) only guarantee the desired behavior on the group manifold G. If the imposed trajectory $g^d(t)$ is bounded at all times, it is easy to see that $g(t)$ is bounded. For the problem to be well defined, we still need to make sure that the internal states are bounded. We have:

Proposition 2. *If g is bounded, then so are q_i, $i = 1, \ldots, N$.*

Proof. It is enough to assume boundness of μ, σ_1, and σ_3 to prove boundness of q_i. Assume

$$\|\mu - \mu^d\| \leq M_\mu, \tag{28}$$
$$\|\sigma_1 - \sigma_1^d\| \leq M_{\sigma_1}, \tag{29}$$
$$\|\sigma_3 - \sigma_3^d\| \leq M_{\sigma_3}. \tag{30}$$

Then, from (29) and (9), we have

$$\sum_{i=1}^{N}(x_i - \mu_x)^2 \leq N(\sigma_1^d + M_{\sigma_1})$$

Similarly, from (30) and (11), we derive

$$\sum_{i=1}^{N}(y_i-\mu_y)^2 \leq N(\sigma_3^d+M_{\sigma_3})$$

from which we conclude that

$$\|q_i-\mu\| \leq \sqrt{N(\sigma_1^d+\sigma_3^d+M_{\sigma_1}+M_{\sigma_3})}$$

Finally, using (28), we have

$$\begin{aligned}\|q_i-\mu^d\| &= \|q_i-\mu+\mu-\mu^d\| \leq \|q_i-\mu\|+\|\mu-\mu^d\| \\ &\leq \sqrt{N(\sigma_1^d+\sigma_3^d+M_{\sigma_1}+M_{\sigma_3})}+M_\mu\end{aligned}$$

which concludes the proof.

Remark 1. It is easy to see that, in the assumed non-degenerate case as defined in Section 3.1, the individual velocities $\dot{q}_i$, $i=1,\ldots,N$ are bounded if the velocity $\dot{g}$ on the group manifold is bounded. Future work will focus on adding terms in the individual controls so that degenerate situations cannot occur.

Moreover, from (21) and (23) it is easy to see that if $\sigma_1^d > 0$, $\sigma_3^d > 0$ then $\sigma_1(t) > 0$, $\sigma_3(t) > 0$, $\forall t \geq 0$, which means that if the robots were not coincident at time 0, they will never become coincident if the proposed control is applied.

In the stabilization to a point case, the boundness and globally asymptotic convergence to the desired values of the group variables $g=[\mu^T,\ \sigma_1,\ \sigma_2,\ \sigma_3]^T$ are guaranteed by (20), (21), (22), (23). Proposition 2 proves the boundness of the internal dynamics. We still need to study the equilibria and regions of convergence for each robot. We have the following Proposition:

Proposition 3. *The closed loop system (13), (20), (21), (22), (23),(6), (9), (10), (11) is in equilibrium at each point on the set $\mu=\mu^d$, $\sigma_1=\sigma_1^d$, $\sigma_2=\sigma_2^d$, $\sigma_3=\sigma_3^d$. All solutions of the closed loop system globally asymptotically converge to $\mu=\mu^d$, $\sigma_1=\sigma_1^d$, $\sigma_2=\sigma_2^d$, $\sigma_3=\sigma_3^d$ when $t\to\infty$*

Proof. For the first part, from (13) and the definitions of the group variables, it is easy to see that the group is in equilibrium ($\dot{g}=0$) if and only if each agent is in equilibrium ($\dot{q}_i=0$, $i=1,\ldots,N$). Therefore, the equilibria of the closed loop system are sets described by $\mu=\mu^d$, $\sigma_1=\sigma_1^d$, $\sigma_2=\sigma_2^d$, $\sigma_3=\sigma_3^d$.

For the second part, consider the following Lyapunov function defined on Q:

$$V(q)=\frac{1}{2}\|\mu^d-\mu\|^2+\frac{1}{2}\|\sigma_1^d-\sigma_1\|^2+\frac{1}{2}\|\sigma_2^d-\sigma_2\|^2+\frac{1}{2}\|\sigma_3^d-\sigma_3\|^2 \tag{31}$$

and consider the derivative of V along the vector field on Q:

$$\dot{V}(q) = -K_\mu \|\mu^d - \mu\|^2 - k_{\sigma_1}\|\sigma_1^d - \sigma_1\|^2 - k_{\sigma_2}\|\sigma_2^d - \sigma_2\|^2 - k_{\sigma_3}\|\sigma_3^d - \sigma_3\|^2 \tag{32}$$

Therefore, $\dot{V}(q) \leq 0$, $\forall q \in \mathbb{R}^{2N}$ and $\dot{V} = 0$ if and only if $\mu = \mu^d$, $\sigma_1 = \sigma_1^d$, $\sigma_2 = \sigma_2^d$, and $\sigma_3 = \sigma_3^d$, which is also an invariant set for the closed loop system. According to the Global Invariant Set Theorem (LaSalle), to prove the proposition we only have to prove that $V(q) \to \infty$ as $\|q\| \to \infty$. We prove this by contradiction. Suppose $\|q\| \to \infty$ and there exists some $L > 0$ so that $V(q) < L$. This implies

$$\|\mu - \mu^d\| \leq \sqrt{2L},\ \|\sigma_1 - \sigma_1^d\| \leq \sqrt{2L},$$
$$\|\sigma_2 - \sigma_2^d\| \leq \sqrt{2L},\ \|\sigma_3 - \sigma_3^d\| \leq \sqrt{2L}$$

By an argument similar to the one used in the proof of Proposition 2, we can conclude that

$$\|q_i - \mu^d\| \leq \sqrt{N(\sigma_1^d + \sigma_3^d + 2\sqrt{2L})} + \sqrt{2L}$$

which means that all q_i are bounded. But $\|q\| \to \infty$ implies that, for at least one i, $i = 1, \ldots, N$, $\|q_i\| \to \infty$. Therefore, we reached a contradiction and the theorem is proved.

4 Mean and Variance Control for Fully Actuated Planar Robots

This section is a particular case of the previous Section 3 when the group variable g as defined by (8) is restricted to the 3 - dimensional $\bar{g} = [\mu^T,\ \sigma]^T$ where μ is the mean given by (6) and the variance σ is defined by

$$\sigma = \sigma_1 + \sigma_3 = \frac{1}{N}\sum_{i=1}^{N}(q_i - \mu)^T(q_i - \mu) \tag{33}$$

Then it is easy to see that *each agent $q_i(t)$ is inside a circle centered at $\mu(t)$ and with radius $\sqrt{N\sigma(t)}$*. The proof is obvious by noting that from (33), for each $i = 1, \ldots, N$, we have

$$\|q_i - \mu\|^2 \leq \sum_{j=1}^{N}\|q_j - \mu\|^2 = N\sigma$$

It is straightforward to check that the decoupling, boundness of internal dynamics, and stability results proved above remain valid. The individual control laws assume a much simpler form in this case:

$$\dot{q}_i = u_i = \dot{\mu} + \frac{q_i - \mu}{2\sigma}\dot{\sigma},\ i = 1, \ldots, N \tag{34}$$

Therefore, one can design trajectories on the 3 - dimensional manifold $[\mu^T \ \sigma]^T$, *i.e.,* generate a moving circle with varying radius on the plane, and have the guarantee that, if each agent is applied the corresponding control law (34), stays inside the moving circle. Obvious applications would be clustering, obstacle avoidance, tunnel passing, etc.

For the simplified controllers (34), the following interesting result holds:

Proposition 4. *The decoupling, minimum energy controllers (34) preserve the shape and orientation of the structure formed by the position vectors q_i in the given inertial frame.*

Proof. Let $l_{ij} = \|q_i - q_j\|$, $i \neq j$. Using (34) and $l_{ij}^2 = (q_i - q_j)^T(q_i - q_j)$, it is easy to see by integration that

$$l_{ij}(t) = l_{ij}(0)\sqrt{\frac{\sigma(t)}{\sigma(0)}}, \ \forall t > 0$$

and therefore

$$\frac{l_{ij}(t)}{l_{kl}(t)} = \frac{l_{ij}(0)}{l_{kl}(0)}, \ \forall i, j, k, l = 1, \ldots, N, \ \forall t > 0$$

from which we conclude that the geometric shape is preserved and the scale factor is proportional to $\sqrt{\sigma(t)}$. Also, straightforward calculations show that

$$\frac{d}{dt}\left(\frac{q_i - q_j}{\|q_i - q_j\|}\right) = 0, \ i \neq j$$

proving that the orientation of the structure is also preserved during the motion.

Remark 2. The group abstraction is, in this case, reduced to the position of the centroid and the scale factor of a geometric figure of given shape and orientation determined by the initial positions of the robots.

4.1 Example

Consider the task of controlling $N = 10$ planar fully-actuated robots with arbitrary initial positions so that they pass through a tunnel of given geometry in 2 seconds. See Figure 2. The initial positions are

$$q_1(0) = \begin{bmatrix} 5 \\ 5 \end{bmatrix}, \ q_2(0) = \begin{bmatrix} 5 \\ 4 \end{bmatrix}, \ q_3(0) = \begin{bmatrix} 4 \\ 3 \end{bmatrix}, \ q_4(0) = \begin{bmatrix} 6 \\ 7 \end{bmatrix}, \ q_5(0) = \begin{bmatrix} 9 \\ 6 \end{bmatrix},$$

$$q_6(0) = \begin{bmatrix} 7 \\ 8 \end{bmatrix}, \ q_7(0) = \begin{bmatrix} 3 \\ 4 \end{bmatrix}, \ q_8(0) = \begin{bmatrix} 2 \\ 7 \end{bmatrix}, \ q_9(0) = \begin{bmatrix} 4 \\ 4 \end{bmatrix}, \ q_{10}(0) = \begin{bmatrix} 6 \\ 9 \end{bmatrix}.$$

The corresponding initial group variable is given by

$$\mu(0) = [5.1,\ 5.7]^T, \sigma(0) = 7.3$$

By the simplified abstraction presented above, the problem can be reduced to controlling the 3 - dimensional group variable $\bar{g} = [\mu^T,\ \sigma]^T$ where μ and σ are given by (6) and (33), respectively. From the geometry of the tunnel, it can be seen that an enclosing circle of radius 2 is enough to pass the tunnel. The corresponding variance σ for radius 2 is 0.4 ($\sqrt{10 \cdot 0.4} = 2$).

First, we generate controls so that, in 1 second, the robots gather inside a circle of radius 2 centered at (10,0) in front of the tunnel. This can be simply done by designing constant vector fields on the group manifold as

$$\dot{\sigma} = 0.4 - \sigma(0) = -6.9,\ \dot{\mu} = [10,\ 0]^T - \mu(0) = [4.9,\ -5.7]^T.$$

After 1 second, we need to control the robots so that they pass through the tunnel. This can be done by keeping the radius of the moving circle constant and move the center along the x - axis (the axis of the tunnel) for 1sec until the point (15,0) is reached. The correponding vector fields on the group manifold are constant and given by

$$\dot{\sigma} = 0,\ \dot{\mu} = [15,\ 0]^T - \mu(1) = [5,\ 0]^T$$

Eight snapshots from the generated motion of the team are shown for illustration in Figure 2 together with the enclosing circle.

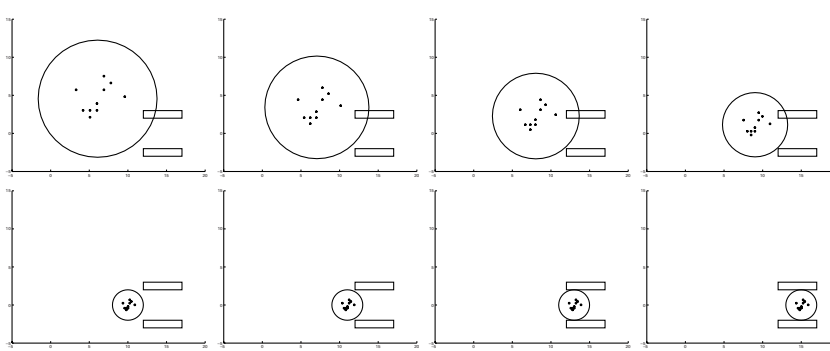


Fig. 2. 10 planar robots moving through a tunnel. The control was designed on the 3 - dimensional $[\mu^T,\ \sigma]$. The individual control laws are given by (34).

5 Conclusion

In this paper we propose a control method for a large number of robots based on an abstraction of the team to a small dimensional group manifold

whose dimension does not scale with the number of robots. The task to be accomplished by the team suggests a natural feedback control system on the group manifold. From the multititude of individual control laws which determine the desired behavior of the group, we select those that only use partial state feedback so that the inter - robot information exchange in the overall control architecture is kept as small as possible. In this paper, we only consider fully-actuated planar robots abstracted to mean and covariance of the positions with respect to some reference frame. We prove that in this case the controllers that minimize the overall energy of the group are also *decoupling* controllers, *i.e.,* the control law of each robot only depends on the state of the robot and the small dimensional state of the group manifold. Illustrative examples of trajectory tracking on the group manifold and globally asymptotic stabilization to a point on the group manifols are included.

References

1. Belta C., Kumar V. (2002) Trajectory design for formations of robots by kinetic energy shaping. In: IEEE International Conference on Robotics and Automation, Washington, DC, 2002
2. Desai J., Ostrowski J.P., Kumar V. (1998) Controlling formations of multiple mobile robots. In: Proc. IEEE Int. Conf. Robot. Automat., Leuven, Belgium, May, 1998. 2864–2869
3. Eren T., Belhumeur P.N., Morse A.S. (2002) Closing ranks in vehicle formations based rigidity. In: IEEE Conference on Decision and Control, Las Vegas, NV, December, 2002
4. Isidori A. (1995) Nonlinear Control Systems, 3rd edn. Springer-Verlag, London
5. Leonard N.E., Fiorelli E. (2001) Virtual leaders, artificial potentials, and coordinated control of groups. In: 40th IEEE Conference on Decision and Control, Orlando, FL, December, 2001. 2968 – 2973
6. Ogren P., Fiorelli E., Leonard N.E. (2002) Formations with a mission: stable coordination of vehicle group maneuvers. In: Proc. Symp. on Mathematical Theory of Networks and Systems, Notre Dame, IN, August, 2002
7. Olfati-Saber R., Murray R.M. (2002) Distributed cooperative control of multiple vehicle formations using structural potential functions. In: IFAC World Congress, Barcelona, Spain, July, 2002
8. Tabuada P., Pappas G.J., Lima P. (2001) Feasible formations of multi-agent systems. In: American Control Conference, Arlington, VA, June, 2001
9. Tanner H., Pappas G.J., Kumar V. (2002) Input-to-state stability on formation graphs. In: Proceedings of the IEEE International Conference on Decision and Control, Las Vegas, NV, December, 2002

Omnidirectional Sensing for Robot Control

Kostas Daniilidis, Christopher Geyer, Volkan Isler, and Ameesh Makadia

GRASP Laboratory and Department of Computer and Information Science
University of Pennsylvania
{kostas,cgeyer,isleri,makadia}@grasp.cis.upenn.edu

Abstract. Most of today's mobile robots are equipped with some kind of omnidirectional camera. The advantages of such sensors in tasks like navigation, homing, appearance-based localization cannot be overlooked. In this paper, we address the basic questions of how to process omnidirectional signals, how to describe the intrinsic geometry of omnidirectional cameras with a single viewpoint, how to infer 3D motion, and how to place omnidirectional sensors efficiently to guarantee complete coverage.

1 Introduction

Visual sensors have always been claimed to be essential for robot control due to their maximum density in spatiotemporal data: no other sensors (contact, force, inertial, sonar, radar, GPS) can provide so many bits in space and time. Today, most robots are equipped with a camera and can successfully accomplish visual tasks: Soccer robots detect color targets, many indoor robots localize themselves based on vertical edges, humanoids react to facial expressions, vehicles can follow traffic lanes. However, to paraphrase Marr, we believe that there is much more to images than linear edges or color patterns. This is unfortunately easy to realize as soon as we deploy a robot on an outdoors natural uneven terrain [14] where no landmarks or urban structures exists. Nevertheless, biological vision systems as observed in birds and insects can solve many vision based control tasks like obstacle avoidance, docking, homing, and map building at ease and in real time. Actually, roboticists do not face this comparison for the first time: they have been already successful in building locomotion systems similar to biological ones [15]. When studying the history of robotics we realize that progress in locomotion and manipulation systems has been tremendous. However, if we look at the vision sensors, we realize that we are 40 years back: we still use cameras originally built for television (IEEE 1394 cameras obey the same principle), built of classical photographic lenses and a photosensitive chip. It is not a coincidence that species navigating easily in natural environments have also the most advanced vision sensors. Based on evidence from biological vision, camera innovation has to be twofold: in the optical level and in the circuitry level. Here, we deal with the optical level and in particularly with the geometric optics (as opposed to the photometric issues).

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 183–197, 2003

One of the first innovations in the geometric optics of cameras was the introduction of omnidirectional cameras, which already became popular in mobile robots. Omnidirectional cameras is one step towards a more efficient sampling of what is called a plenoptic function [1]: A real-valued function which maps every line (ray) in space, wavelength, and time-instant to the apparent intensity of the intersection of this ray with the scene. With omnidirectional cameras we can sample the plenoptic function in all ray directions, though from a constrained set of viewpoints. This helps robots in many aspects: in a bumpy terrain you can keep your target or your home steadily inside the field of view, you are alert towards all directions, and you can keep multiple robot or human agents inside the field of view.

We present a theoretical framework for working with omnidirectional sensors, that covers the entire spectrum from signal processing to sensor planning. In the course of this exposition, we will realize, how, for example, localization and mapping, are solved simpler and more efficiently using such sensors. In particularly, we address the following questions:

- Signal question: How can we process images whose domain is not a perspective plane but a sphere? How could we solve homing and localization directly from that image? How can we conduct signal processing if the plenoptic function is sampled on an arbitrary image manifold?
- Geometry question: How can we solve localization and mapping from multiple views of an omnidirectional camera with a single viewpoint?
- Planning question: What is the complexity of optimally placing omnidirectional cameras in a known polyhedral environment?

Before, addressing the three basic questions, we show a unifying model covering all omnidirectional sensors with a single viewpoint.

2 A Unifying Projection Model

A catadioptric device is an optical system combining reflective (catoptric) and refractive (dioptric) elements. Our motivation to study central catadioptric cameras is to understand the geometric properties of the mappings realized with these sensors. The fact that we are able to choose the parameters of a quadric mirror surface and appropriately mount the camera implicitly encodes information which can be exploited during image interpretation.

Recently, we [8] showed that there is an equivalence between any central catadioptric projection and a composite mapping through the sphere (see Fig. 1 for a sectional drawing of the quivalence on the right). This mapping consists of the projection of a point from the center onto the sphere and a subsequent projection from a point on the axis of the sphere onto a plane perpendicular to that axis. The position of the point on the axis depends on the shape of the mirror. To be more specific we parameterize a conic section with its eccentricity ϵ and its latus rectum $4p$, decoupling thus the shape from the global scale.

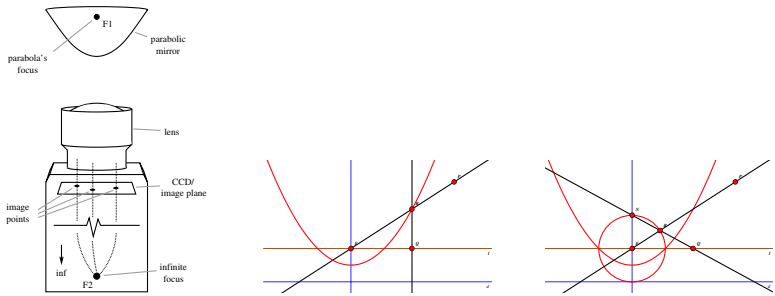


Fig. 1. On the left, a sketch of a parabolic catadioptric camera with the two centers of projection F_1 and F_2. In the middle, the projection of a point P on the parabolic profile with subsequent parallel projection to Q in the catadioptric plane. On the right, the projection of a point P on the sphere with subsequent stereographic projection to Q' on the catadioptric plane, for a proof of $Q = Q'$ see [8].

(i) In case of a hyperbolic mirror with foci at the origin and the point $(0, \frac{4p\epsilon}{\epsilon^2-1})$ and a perspective lens with focal point at the second point of the hyperbola, the equivalent model is a projection to a sphere at the origin followed by a projection from the point $\frac{2\epsilon}{1+\epsilon^2}$ to the plane $y = \frac{2\epsilon(2p-1)}{1+\epsilon^2}$.

(ii) An orthographic camera with a parabolic mirror which is equivalent to the projection on the sphere with a subsequent projection from the north pole to the plane through the equator. This latter projection is well known as the stereographic projection and is a conformal mapping.

(iii) A perspective camera combined with a planar mirror which can be modeled as a projection on the sphere with a subsequent projection from the center to the plane tangent at the south-pole. We realize, that the conventional TV camera, is just the boundary of a continuum of single viewpoint projections.

Given the intermediate projection on the sphere it is very easy to study features in the catadioptric plane. While it is trivial to see that scene points project to points, it is interesting to figure out what are the projections of lines. Lines project onto great circles on the sphere. The subsequent projection of a great circle to the catadioptric plane is a conic section as illustrated in Fig. 2. In case of parabolic projection, the stereographic projection of the second steps projects great circles onto circles in the catadioptric plane. Such circles have the property that they intersect the projection of the sphere equator antipodally.

3 The Signal Question

Consider an omnidirectional image like the one on the left of figure 3 and imagine that we have to perform template matching or filtering for edge detection and flow estimation.

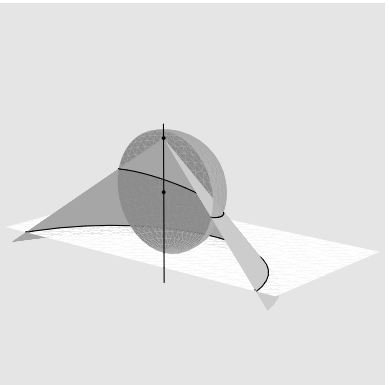


Fig. 2. The projection of a line to the sphere is a great circle; the projection of the great circle is obtained from the intersection of the image plane with a cone containing the great circle and whose vertex is the point of projection.

Obviously, blind application of conventional shift-invariant operators or optical flow estimators yields erroneous results. Based on the unifying model described above, we propose to use the sphere as the underlying domain of image processing in central catadioptric systems. This does not mean that we will warp the catadioptric image into a spherical image (center in Fig. 3). Instead, we will formulate all the operations on the sphere but use the samples from the original catadioptric image $I(u, v)$. Convolution with a kernel $G_s(\theta, \phi)$ will be defined on the spherical image $I_S(\theta, \phi)$ giving a response $R_S(\theta, \phi)$ (Fig. 3, right). We will introduce the Gaussian function on the sphere and compute the derivatives on the sphere by convolving with its spatiotemporal derivatives. We will keep the derivatives on the sphere for the computation of the optical flow, however, sampled on the catadioptric plane. In [5] we have how to compute optical flow in omnidirectional images based on spatiotemporal filtering on the sphere.

Fig. 3. On the left a parabolic catadioptric image; in the center its 1:1 mapping to a hemisphere, on the right the proposed processing scheme.

To perform convolution on the sphere, we define a function at either the north-pole or the south-pole and we rotate the pole to the new position. The reader might have already realized that this cannot be written with

a shift in the θ and ϕ angles. This realization makes the generalization of imaging surfaces to arbitrary imaging manifolds so challenging: We have to consider the group action on the image which in this particular case is the rotation group SO(3). Based on this realization we have to define convolution as well as the Fourier transform on SO(3) acting on the sphere which is the homogeneous space SO(3)/SO(2). Signal analysis on the sphere has been already studied in geophysics and in wavelet transforms [6] and Fourier transforms on groups is the subject of harmonic analysis [4].

In our current work we consider the entire scene as a target and try to estimate the robot's pose directly from the change in the omnidirectional image. If the change is only due to rotations, we can design a computational procedure that makes use of the shift theorem in spherical harmonic transforms in order to extract the rotation parameters. Such global operations can revolutionize visual servoing because they directly overcome the intermediate feature detection and correspondence. A direct application is visual attitude servoing.

4 The Geometry Question

The geometry of perspective views has been extensively studied in the past decade. Two books [10] and [7] contain comprehensive treatments of the subject. At the same time, the need for a larger field of view in surveillance, robotics, and image based rendering motivated us to study whether simultaneous localization and mapping can be solved simpler and more efficiently using omnidirectional cameras. We consider two uncalibrated views obtained with a parabolic catadioptric device whose spherical projection equivalence through stereographic projection has been shown in section 2. In parabolic catadioptric views, images of world points are points but images of world lines are circles. To study epipolar geometry, we need a modeling of the transformations of the catadioptric plane. It is not obvious whether we have something equivalent to affine mappings describing the intrinsic transformation of a camera in the perspective case or even the analogous mapping of a collineation.

4.1 Lifting Points and Circles: Circle Space

A unit sphere centered at the origin has the quadratic form

$$\boldsymbol{Q} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} . \tag{1}$$

Given a point $\boldsymbol{x} = (u, v, 0, 1)$ we wish to find the point $\widetilde{\boldsymbol{x}}$ on the sphere which when stereographically projected from $\boldsymbol{N} = (0, 0, 1, 1)$ would give $\boldsymbol{x}$.

It is easy to verify that the point

$$\widetilde{\boldsymbol{x}} = \left(2u, 2v, u^2 + v^2 - 1, u^2 + v^2 + 1\right)^T \tag{2}$$

lies on the sphere and is collinear with $\boldsymbol{N}$ and $\boldsymbol{x}$. The point $\widetilde{\boldsymbol{p}}$ will be called the **lifting** of the point $\boldsymbol{x}$, whereas $\boldsymbol{x}$ is the stereographic projection of $\widetilde{\boldsymbol{x}}$.

Circles can also be represented in this framework due to the following fact. A a circle in the image plane can be represented with the polar point of the plane containing the lifted image points lying on the circle. Recall from projective geometry that the polar point of a plane is the vertex of the cone tangent to the sphere (or any quadric surface) at the intersection of the plane with the sphere. The polar plane of a point has the reverse relationship. Let $\boldsymbol{\gamma}$ be a circle centered in the image plane at $(c_x, c_y, 1)$ with radius r. We claim that the plane containing the lifted points of $\boldsymbol{\gamma}$ is

$$\boldsymbol{\pi} = \left(2c_x, 2c_y, c_x^2 + c_y^2 - r^2 - 1, -c_x^2 - c_y^2 + r^2 - 1\right)^T .$$

The polar point of this plane $\boldsymbol{\pi}$ will be the point representation $\widetilde{\boldsymbol{\gamma}}$ (Fig. 4 (left)) of the circle $\boldsymbol{\gamma}$, where

$$\widetilde{\boldsymbol{\gamma}} = \boldsymbol{Q}\boldsymbol{\pi} = \left(2c_x, 2c_y, c_x^2 + c_y^2 - r^2 - 1, c_x^2 + c_y^2 - r^2 + 1\right) . \tag{3}$$

Hence, points of the circle space outside the sphere represent circles, points on

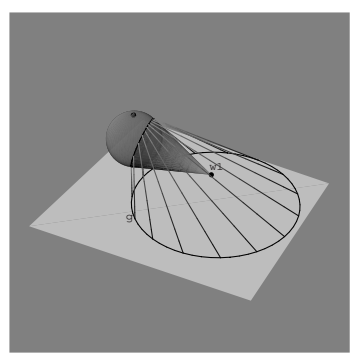


Fig. 4. The lifting of a circle $\boldsymbol{\gamma}$ to the point $\widetilde{\boldsymbol{\gamma}}$ in spherical circle space.

the sphere represent points, and points inside the sphere represent imaginary circles. Among the imaginary circles there is one of particular interest $(u - c_x)^2 + (v - c_y)^2 = -r^2$ called the absolute conic [17] and encoding the entire information on the focal length and the image center. Its lifting lying inside the sphere will be called the lifted absolute conic $\widetilde{\boldsymbol{\omega}}$. If the camera is calibrated, the lifted absolute conic will be at the origin.

Not all circles in a catadioptric plane represent projections of lines, but only the ones whose inverse stereographic projections are great circles. The effect on the catadioptric plane is that all circles which are line projections intersect the equator projection antipodally. The following can be proved [9]:

Proposition 1. *The set of circles intersecting the projection of the equator antipodally, i.e. the set of line images, lie on a plane whose polar point with respect to* $\boldsymbol{Q}$ *is the lifted absolute conic* $\widetilde{\boldsymbol{\omega}}$*.*

4.2 The Lorentz Group and Plane Preserving Subgroups

In a perspective image a natural class of transformations on image points is the set of collineations, projective transformations specified by non-singular 3×3 matrices. We would like to find an equivalent structure for parabolic catadioptric images under the requirement that the transformation operate linearly on the circle space. Therefore this class must consist of some subset of 4×4 matrices. These transformations also should not act in a way which happens to transform a point into a circle or vice versa, for this would inviolate incidence relationships in the image plane. Thus the surface of the sphere must remain invariant under any such transformation. This is the barest of conditions necessary to determine the set of transformations and we therefore investigate the set $\mathcal{L} = \{\boldsymbol{A} : \boldsymbol{A}^T\boldsymbol{Q}\boldsymbol{A} = \boldsymbol{Q}\}$. This is a group since it is closed under multiplication and inversion and contains the identity. As it turns out this is a well known six dimensional[1] Lie group from the study of physics called the Lorentz group [11]. Any transformation from this group preserves angles between circles, for if $\boldsymbol{A} \in \mathcal{L}$ then $\langle \boldsymbol{x}, \boldsymbol{y} \rangle_Q = \langle \boldsymbol{A}\boldsymbol{x}, \boldsymbol{A}\boldsymbol{y} \rangle_Q$. Since two circles can be constructed to form any angle, these transformations must preserve all angles when they transform the image plane. Angles replace the cross ratio as the invariance under these transformations. It also implies that general projective transformations applied to image points that do not preserve angles, such as shearing or change of aspect ratio, can not be represented as a linear transformation of circle space, at least not in a way which preserves incidence relationships.

In the previous section it was said that the set of line images of a given parabolic projection have point representations lying on a plane in circle space. The plane on which they lie is polar to the point representation of the image of the absolute conic, $\widetilde{\boldsymbol{\omega}}$. What is the set of transformations preserving this plane and what meaning does this have? In order that a transformation preserve the plane it must preserve $\widetilde{\boldsymbol{\omega}}$. Therefore $\widetilde{\boldsymbol{\omega}}$ must be an eigenvector of the transformation for any eigenvalue (since $\widetilde{\boldsymbol{\omega}}$ is homogeneous). Let

$$\mathcal{L}_{\widetilde{\omega}} = \{\boldsymbol{A} : \boldsymbol{A}^T\boldsymbol{Q}\boldsymbol{A} = \boldsymbol{Q} \text{ and } A\widetilde{\boldsymbol{\omega}} = \lambda\widetilde{\boldsymbol{\omega}} \text{ for some } \lambda\}\,.$$

This is a group since it is also closed under multiplication and inversion.

We examine two subcases, $\widetilde{\boldsymbol{\omega}} = (0, 0, 0, 1)$ and $\widetilde{\boldsymbol{\omega}} = \boldsymbol{N}$. We will calculate the Lie algebra for the connected component containing the identity. If $\boldsymbol{A}(t)$ is a continuous parameterization of matrices in $\mathcal{L}_{\widetilde{\omega}}$ such that $\boldsymbol{A}(0) = I$, then

[1] The inclusion of scale yields an additional dimension, and then $\boldsymbol{A}^T\boldsymbol{Q}\boldsymbol{A} = \lambda\boldsymbol{Q}$.

the first condition gives

$$\frac{d}{dt}\boldsymbol{A}(t)^T\boldsymbol{Q}\boldsymbol{A}(t)\Big|_{t=0} = \frac{d}{dt}\boldsymbol{Q}\Big|_{t=0} \quad \text{and} \quad \boldsymbol{A}'(0)^T\boldsymbol{Q} + \boldsymbol{Q}\boldsymbol{A}'(0) = 0\,.$$

The second condition is equivalent to the 2×2 sub-determinants of the matrix $(\widetilde{\boldsymbol{\omega}}, \boldsymbol{A}(t)\widetilde{\boldsymbol{\omega}})^T$ being zero. Each of the six equations for the sub-determinants can be differentiated with respect to t and evaluated at $t = 0$ and then one can solve for the entries $\boldsymbol{A}'(0)$. When $\widetilde{\boldsymbol{\omega}} = (0, 0, 0, 1)$, this yields

$$\boldsymbol{A}'(0) = \begin{pmatrix} 0 & a_{12} & a_{13} & 0 \\ -a_{12} & 0 & a_{23} & 0 \\ -a_{13} & -a_{23} & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

which is just the set of matrices which are skew symmetric in the first three rows and columns and zero elsewhere. Therefore $\mathcal{L}_{(0,0,0,1)}$ is the set of rotations in $\mathbb{P}^3$.

If $\widetilde{\boldsymbol{\omega}} = \boldsymbol{N}$, the north pole, then $\boldsymbol{A}'(0)$ must be of the form

$$\boldsymbol{A}'(0) = \begin{pmatrix} 0 & -a_{12} & -a_{13} & a_{13} \\ a_{12} & 0 & -a_{23} & a_{23} \\ a_{13} & a_{23} & 0 & a_{34} \\ a_{13} & a_{23} & a_{34} & 0 \end{pmatrix}.$$

The Lie group generated by this Lie algebra preserves $\boldsymbol{N}$ and therefore it preserves the plane tangent to $\boldsymbol{N}$ on which lie the point representations of lines. They therefore sends lines to lines while also by default preserving angles. Therefore this subgroup corresponds to affine transformations in the plane. An important subcase and reparameterization of $\mathcal{L}_N$ is defined under the following substitutions, $a_{12} = 0, a_{34} = -\log 2f, a_{13} = -c_x\frac{\log 2f}{2f-1}, a_{23} = -c_y\frac{\log 2f}{2f-1}$. Upon exponentiation we have the following matrix dependent on $\widetilde{\boldsymbol{\omega}}$,

$$\boldsymbol{K}_{\widetilde{\omega}} = \begin{pmatrix} 1 & 0 & c_x & -c_x \\ 0 & 1 & c_y & -c_y \\ -\frac{c_x}{2f} & -\frac{c_y}{2f} & \frac{1-c_x^2-c_y^2+4f^2}{2f} & \frac{1+c_x^2+c_y^2-4f^2}{2f} \\ -\frac{c_x}{2f} & -\frac{c_y}{2f} & \frac{1-c_x^2-c_y^2-4f^2}{2f} & \frac{1+c_x^2+c_y^2+4f^2}{2f} \end{pmatrix}. \tag{4}$$

This has the effect on the image points of translating by $(-c_x, -c_y)$ and then scaling by $\frac{1}{2f}$. Also notice that $\boldsymbol{K}_{\widetilde{\omega}}\widetilde{\boldsymbol{\omega}} = \lambda O$ and $\boldsymbol{K}_{\widetilde{\omega}}\widetilde{\boldsymbol{\omega}}' = (0, 0, \lambda, 0)$. We will use this matrix in the next section.

In general when $\widetilde{\boldsymbol{\omega}}$ does not lie on the sphere, the dimension of $\mathcal{L}_{\widetilde{\omega}}$ is three because the sub-determinants give three independent constraints; this Lie group corresponds to rotations about the viewpoint. When $\widetilde{\boldsymbol{\omega}}$ lies on the sphere an additional dimension arises because the number of independent constraints decreases by one; this Lie group leaves the image point corresponding to $\widetilde{\boldsymbol{\omega}}$ invariant. One additional comment, since $\exp A^T = (\exp A)^T$, and since the Lie algebra of $\mathcal{L}$ can be seen to contain $\boldsymbol{A}$ if and only if it contains $\boldsymbol{A}^T$, then $\boldsymbol{B} \in \mathcal{L}$ if and only if $\boldsymbol{B}^T \in \mathcal{L}$.

4.3 3D motion estimation from uncalibrated views

Considering circle space as a threedimensional projective space, it can be shown that we can obtain an epipolar constraint of the form

$$\widetilde{\boldsymbol{x}}_1^T \boldsymbol{F} \widetilde{\boldsymbol{x}}_2 = 0 \quad \text{where} \quad \boldsymbol{F} = \boldsymbol{K}_{\widetilde{\omega}_1}^T \begin{pmatrix} \boldsymbol{E} & 0 \\ 0 & 0 \end{pmatrix} \boldsymbol{K}_{\widetilde{\omega}_2} . \tag{5}$$

A matrix $\boldsymbol{F}$ expressed in this way will be called a catadioptric fundamental matrix. The first thing to note about this new $\boldsymbol{F}$ is that since $K\widetilde{\boldsymbol{\omega}} = \breve{\boldsymbol{\omega}}$, we must have

$$\boldsymbol{F}\widetilde{\boldsymbol{\omega}}_2 = 0 \quad \text{and} \quad \boldsymbol{F}^T \widetilde{\boldsymbol{\omega}}_1 = 0 .$$

Hence, the lifted left and right images of the absolute conic belong to the left and right nullspace of $\boldsymbol{F}$, respectively.

Note that the expression $\widetilde{\boldsymbol{x}}_1^T \boldsymbol{F} \widetilde{\boldsymbol{x}}_2$ is linear in the entries of the matrix $\boldsymbol{F}$. Hence just like in the perspective case, given a set of correspondences a matrix whose entries are the coefficients in the epipolar equation of each entry of $\boldsymbol{F}$ can be constructed whose nullspace contains the matrix $\boldsymbol{F}$ flattened into a single vector in $\mathbb{R}^{16}$. Thus, self-calibration is achieved from two views using singular value decomposition by selecting the vector with the smallest singular value.

5 The Planning Question

We move over to the case of multiple robots equipped with omnidirectional cameras. Imagine a known 3D polyhedral environment where a set of cameras has to be placed in such a way that every point in the environment is visible. The 2D version is known as the art gallery problem [16] and sufficiency results exist for several versions of this problem. For example, $\lfloor \frac{n}{3} \rfloor$ cameras can cover any simple polygon. However, such results are inapplicable in robotic and image-based rendering applications where the environments can be very complex with millions of vertices. A further arising application is placing antennas for line-of-sight broadband communication between flying vehicles.

Here, we consider the problem of minimizing the number of viewpoints without sacrificing the goal of complete visibility. The particular scenarios we are addressing are surveillance, object inspection, and image based rendering. In the case of surveillance, we need a complete coverage at any time so that no event will be missed. This is the reason why coverage with one mobile guard (shortest watchman route - solvable in polynomial time) is not applicable. In case of object inspection, we know the prior geometry of an object, and we need the minimal number of views so that the object will be checked regarding defects. In this scenario, the object might be placed on a turntable and we ask then for the minimal number of rotations. The objects might be medical organs which have to be imaged from very few viewpoints of an endoscope guided by a robot manipulator. In the case of image based

rendering, we have a prior map of the environment but we need to obtain a detailed reconstruction with a range sensor or we need just the appearance of the environment for visualization. This is very important for telepresence and immersive environments: After the environment has been captured and an immersed user changes her viewpoint any hole would cause a break in the sense of presence.

5.1 Vapnik-Chervonenkis-Dimension

A set system is a pair $(X, \mathcal{R})$ where X is a set and $\mathcal{R}$ is a collection of subsets $R \subseteq X$.

Definition 1. Given a set system $(X, \mathcal{R})$, let A be a subset of X. We say A is shattered by $\mathcal{R}$ if $\forall Y \subseteq A \; \exists R \in \mathcal{R}$ such that $R \cap A = Y$. The VC-dimension of $(X, \mathcal{R})$ is the cardinality of the largest set that can be shattered by $\mathcal{R}$ [20].

The VC-dimension, introduced first in supervised learning for pattern classification, is an indicator about the separability of set elements where subsets are pattern classes. It turned out that it also plays an important role in randomized and geometric algorithms [13,2].

Given a set system, the minimum *set cover* problem asks for a minimum cardinality set $\mathcal{S} \subseteq \mathcal{R}$ such that $\bigcup_{R \in \mathcal{S}} R = X$. [3] presented an algorithm which returns $O(\log d)$ approximations to set cover of systems with bounded VC-dimension where d is the optimal set-cover for the system. This is a significant improvement on the previous $\log n$-approximation, when n is large but the optimal is small.

The *hitting set* problem is the dual of set cover and its decision version reads: We are given a set X and a collection of sets $\mathcal{R}$ where each $R \in \mathcal{R}$ is a subset of X. We are also given a number k. The question is whether there is a subset $H \subset X$ such that $|H| \leq k$ and for each $R \in \mathcal{R}$ $R \cap H \neq \emptyset$.

Both problems are known to be NP-complete and can be approximated to within a log factor of the maximum set sizes (in either the primal or the dual system) and not better [18].

5.2 Visibility and set cover

Consider the following formulation of the problem of minimal guard coverage or camera placement. An *instance* of the camera placement problem is: Given a polygon or a polyhedron P and a specification of possible camera locations find a minimum set cover of the system $(P, \{V(c_i)\})$, where $V(c_i)$ is the set of points visible on P from camera c_i and the index i varies over all possible camera locations. The definition of $V(c_i)$ can capture any optical constraints on what a camera can see. We will refer to the specification of possible camera locations as a *setup*. We say a set S of cameras *cover* P if $\bigcup_{c_i \in S} V(c_i) = P$.

Camera placement can also be seen as a particular case of the hitting set problem. The set X is the set of possible camera locations. For each point

p on the boundary of the polyhedron P, there is a set R_p consisting of all camera locations that can see p. The hitting set problem assumes a finite set X and we have to implicitly deal with this issue when we attempt to pose camera placement as such a problem. Typically, we do so by using a sampling technique or by showing that a finite set of extremal points need to be considered. An example of the latter approach is the algorithm *PlaceCircleGuards* to be presented shortly.

We will represent omnidirectional cameras with their projection centers c_i and say that c_i sees the point $p \in P$ if the only intersection of the line segment $\overline{pc_i}$ with P is p. We extend the notion of visibility to sets as follows: We say that a camera sees a set of points ω if it can see all the points in ω. The following notation will be useful for VC-dimension proofs: Let $P_m = \{p_1, \ldots, p_m\}$ be m points. We say that camera c sees the subset $\omega \subseteq P_m$ if c can see all points in ω but no point in $P_m \setminus \omega$.

By the *VC-dimension of a setup*, we will refer to the VC-dimension of the maximum number of points that can be shattered over all instances of the camera placement problem for a specific setup. For example, if there are no restrictions on cameras and we want to cover simple polygons, we would like to find the VC-dimension of the set system $(P, \{V(c_i)\})$ as P varies over the set of all simple polygons. Therefore, in order to give a lower bound m on the VC-dimension of a setup it suffices to present one instance where m points are shattered, but for an upper bound one needs to show that there exists no instance such that $m + 1$ points can be shattered.

5.3 Overview of results

Our results consider 2D and 3D configurations of cameras outside the convex hull of an object. Such setups are practically related to cameras mounted on manipulators for object inspection or image based rendering.

Definition 2. We define 2DCIRCLE as a setup where a set of cameras whose locations are restricted to a circle C are to cover a simple polygon that is contained in C.

Note that not all polygons are completely visible from a circle outside. We restrict ourselves to the points on P that are visible from at least one point on the circle.

Proposition 2. *The VC-dimension of 2DCIRCLE is exactly 2.*

Proof. Consider m points $P_m = \{p_1, \ldots, p_m\}$ that can be shattered. We must identify 2^m points c_ω on the circle corresponding to each subset ω of P_m such that c_ω can see all points in ω but no point in $P_m \setminus \omega$. Let c_i be a camera that sees a point p on P. As we discussed above, for each p_i there is an arc A_i from which p_i is visible.

In general, some of these arcs will be overlapping. The $2m$ endpoints of $A_i,\ i = 1, \ldots, m$ divide C into $2m$ arcs $A'_j,\ j = 1, \ldots, 2m$ such that A'_j are disjoint and within an A'_j only a fixed subset of P_m is visible. Therefore each camera $c_{w_i},\ i = 1, \ldots, 2^m;\ \omega_i \in P_m$ must lie on a different $A'_j,\ j = 1, \ldots, 2m$. But this implies that $2m$ must be greater than or equal to 2^m which is only true for m strictly less than 3. Thus the VC-dimension is upper bounded by 2.

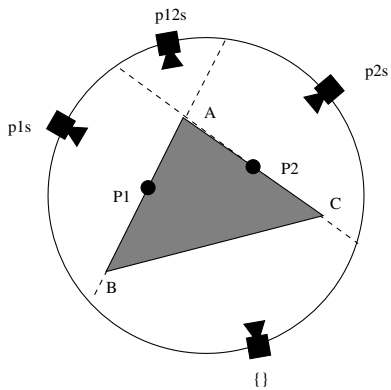


Fig. 5. p_1 and p_2 can be shattered by four cameras. Each camera is labeled with the subset it can see. In this figure the polygon P is $\triangle ABC$.

The lower bound is proven by the example in Figure 5 where the points p_1 and p_2 are shattered by the 4 cameras shown. □

We now relax the restriction on camera locations to be on a circle so that we allow cameras anywhere outside the convex hull of the polygon.

Definition 3. We define 2DCONVEX as a setup where a set of cameras located outside the convex hull of a simple polygon P are to cover P.

Proposition 3. *The VC-dimension of 2DCONVEX is less than or equal to 6.*

Proof. If $P_m = \{p_1, \ldots, p_m\}$ are m points to be shattered on the polygon P, then for each p_i there are two lines $L^L_{p_i}$ and $L^R_{p_i}$ such that p_i is visible from the region lying between $L^L_{p_i}$ and $L^R_{p_i}$ in a clockwise scan. Note that we restricted the cameras to lie outside the convex hull of P, otherwise there could be many lines L_i such that the visibility of a point p changes as we cross L_i. This is illustrated in figure 6.

Consider the arrangement of the $2m$ lines $\{L^L_{p_i}, L^R_{p_i} \mid i = 1, \ldots, m\}$. For contradiction's sake assume that there are two cameras c_ω and $c_{\omega'}$ located in the same face [2] of the arrangement, such that c_ω can see all points in $\omega \subseteq P_m$

[2] The faces of an arrangement of lines are the connected regions on the plane remaining after the removal of the lines from the plane.

but no point in $P_m \setminus \omega$ and similarly for $c_{\omega'}$. If ω and ω' are two different subsets, as we move from c_ω to $c_{\omega'}$ on the line defined by them either a point $q \in \omega \setminus \omega'$ disappears or a point $q \in \omega' \setminus \omega$ becomes visible. But the only reason of this visual event can be crossing L_q^L or L_q^R, which contradicts with the fact that c_ω and $c_{\omega'}$ are in the same face of the arrangement. Therefore, each of the 2^m cameras c_ω, $\omega \subseteq P_m$ must lie in a different face of the arrangement. It is well known that an arrangement of $2m$ lines has at most $\binom{2m+1}{2}+1$ faces therefore we must have $\binom{2m+1}{2}+1 \geq 2^m$ which is only true for m up to 7. Hence the VC-dimension of this system is at most 6. □

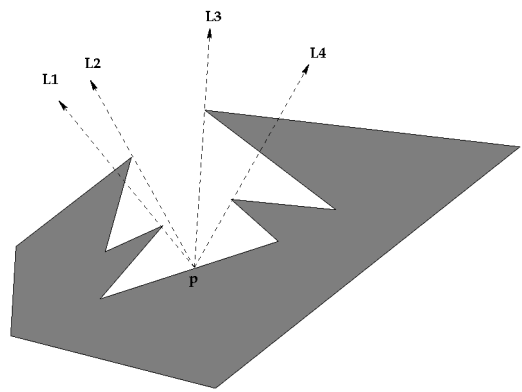


Fig. 6. The effect of restricting the cameras to lie outside the convex hull for the setup 2DCONVEX: Only lines L_2 and L_3 are relevant to the visibility of point P.

Note that relaxing the camera locations from 2DCIRCLE to 2DCONVEX indeed increases the VC-dimension, as we see in the following proposition.

Proposition 4. *The VC-dimension of 2DCONVEX is greater than or equal to 4.*

Proof. See [12].

If we further remove the restriction that the cameras are outside the convex hull then the best known bound is 23 and the reader is referred to [19].

We proceed with the 3D case where we consider a 3D configuration bounded by a sphere. Our main result is a construction that proves that the VC-dimension for visibility in 3D is unbounded.

Definition 4. We define $3DSPHERE$ as a setup where we are given a polyhedron $\mathcal{P}$, and a viewing sphere $\mathcal{S}$ such that $\mathcal{P}$ is totally contained in $\mathcal{S}$.

Proposition 5. *The VC-dimension of 3DSPHERE is* $\Theta(\log n)$.

Proof: See [12].

This result supports our belief that strong geometric constraints for 3D visibility systems do not exist. Therefore, the best one can hope for is the $O(\log n)$-approximation ratio which can be obtained by a simple, greedy algorithm.

6 Future Work

The above mentioned results are just first steps towards a program for exploring space with omnidirectional images and analyzing the information therein. With respect to the geometry question, we study the motion of cameras with rays not intersecting at the same point. In particularly, we are interested in the intrinsic geometry of a camera network which can be regarded as a single collective camera sampling the plenoptic function. In the signal question, which took the least amount of space in this paper, we will employ algorithms for the efficient computation of Fourier transforms on the groups that act on spherical and catadioptric images. In the planning question, we are addressing the exploration of unknown environments and what are the optimal motions that will increase our knowledge of the environment with minimal consumption of computation and travel time.

Acknowledgements.
We thank Professor Sampath Kannan for his contribution to the results in the planning section. We are grateful for support through the following grants: NSF-IIS-0083209, NSF-EIA-0218691, NSF-IIS-0121293, NSF-EIA-9703220, a DARPA/ITO/NGI subcontract to UNC, Chapel Hill, and a Penn Research Foundation grant.

References

1. E.H. Adelson and J.R. Bergen. The plenoptic function and the elements of early vision. In M. Landy and J. A. Movshon, editors, *Computational Models of Visual Processing*. MIT Press, 1991.
2. P.K. Agarwal and S. Sen. Randomized algorithms for geometric optimization problems. *Handbook of Randomization (P. Pardalos, S. Rajasekaran, J. Reif, and J. Rolim, eds.), Kluwer Academic Publishers*, to appear.
3. H. Bronnimann and M.T. Goodrich. Almost optimal set covers in finite VC-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995.
4. G.S. Chirikjian and A.B. Kyatkin. *Engineering Applications of Noncommutative Harmonic Analysis: WIth Emphasis on Rotation and Motion Groups*. CRC Press, 2000.
5. K. Daniilidis, A. Makadia, and T. Bülow. Image processing in catadioptric planes: Spatiotemporal derivatives and optical flow computation. In *Workshop on Omnidirectional Vision, Copenhagen, June 22*, pages 3–12, 2002.

6. J.R. Driscoll and D.M. Healy. Computing fourier transforms and convolutions on the 2-sphere. *Advances in Applied Mathematics*, 15:202–250, 1994.
7. O. Faugeras, Q.-T. Luong, and T. Papadopoulo. *The Geometry of Multiple Images: The Laws That Govern the Formation of Multiple Images of a Scene and Some of Their Applications*. MIT Press, 2001.
8. C. Geyer and K. Daniilidis. Catadioptric projective geometry. *International Journal of Computer Vision*, 43:223–243, 2001.
9. C. Geyer and K. Daniilidis. Properties of the catadioptric fundamental matrix. In *Proc. Seventh European Conference on Computer Vision*, pages 140–154, Copenhagen, Denmark, 2002.
10. R. Hartley and A. Zisserman. *Multiple View Geometry*. Cambridge Univ. Press, 2000.
11. V. Heine. *Group Theory in Quantum Mechanics*. Pergamon Press, Oxford, 1960.
12. I.V. Isler, S. Kannan, and K. Daniilidis. VC-dimension of exterior visibility of polyhedra. Technical Report MS-CIS-01-34, Computer and Information Science, University of Pennsylvania, 2001.
13. J. Matousek. Geometric computation and the art of sampling. In *IEEE Symposium on Foundations of Computer Science*, 1998.
14. L. Matthies. Stereo vision for planetary rovers: Stochastic modeling to near real-time implementation. *International Journal of Computer Vision*, 8:71–91, 1992.
15. K. McIsaac and J. Ostrowski. Motion planning for dynamic eel-like robots. In *IEEE Conf. on Robotics and Automation*, 2000.
16. J. O'Rourke. *Art Gallery Theorems And Algorithms*. Oxford University Press, 1987.
17. J. Semple and G. Kneebone. *Algebraic Projective Geometr*. Oxford University Press, 1979.
18. P. Slavik. A tight analysis of the greedy algorithm for set cover. *Journal of Algorithms*, 25:237–254, 1997.
19. P. Valtr. Guarding galleries where no point sees a small area. *Israel Journal of Mathematics*, 104:1–16, 1998.
20. V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

A Passivity Approach to Vision-based Dynamic Control of Robots with Nonlinear Observer

Hiroyuki Kawai, Shintaro Izoe, and Masayuki Fujita

Department of Electrical and Electronic Engineering
Kanazawa University
Kodatsuno, Kanazawa 920–8667, JAPAN

Abstract. This paper investigates a vision-based robot motion control using position measurements and visual information. Firstly the model of relative rigid body motion (positions and rotations) and the method for estimation of the relative rigid body motion are presented in order to derive the visual feedback system. Secondly we lead a structural passivity-like property of the visual feedback system. Next, we consider the velocity observer and derive the dynamic visual feedback system which contains the manipulator dynamics. Finally the main results with respect to stability and L_2-gain performance analysis for the proposed dynamic visual feedback control are discussed. Our proposed method is based on passivity of the visual feedback system and the manipulator dynamics.

1 Introduction

Vision-based control of robotic systems involves the fusion of robot kinematics, dynamics, and computer vision system to control the position of the robot end-effector in an efficient manner [1]. The combination of mechanical control with visual information, so-called *visual feedback control* or *visual servoing*, should become extremely important, when a mechanical system is operating in an unstructured environment. In this approach, the control objective is to track the target object in a three-dimensional workspace by using image information. An interesting historical review of the visual servoing can be found in [1].

This paper deals with a robot motion control with visual information in the eye-in-hand configuration as depicted in Fig. 1. Classical visual servoing algorithms assume that the manipulator dynamics is negligible and do not interact with the visual feedback loop [2]–[4]. This assumption is invalid for high speed tasks, while it holds for kinematic control problems. Hashimoto and Kimura [5] have proposed a model-based control algorithm which compensated the manipulator dynamics based on the input-output feedback linearization technique. Kelly *et al*. [6] have considered a simple image-based controller which guaranteed local asymptotic stability under an assumption that objects' depths are known. In [7], robust visual servo control of a planar manipulator has been proposed via the robust backstepping approach in the

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 199–213, 2003

fixed camera configuration. In recent papers [8][9], the authors have proposed the 3-D visual feedback control which has guaranteed local asymptotic stability without the known objects' depths from the theoretical standpoint. While several researches [10]–[13] have pointed out a problem of global stability, local asymptotic stability is very important in case of the image feature points exist in the neighborhood of the equilibrium point. In case the initial error is large, it can be sampled and a reference trajectory can be generated by using a path planning approach [14][15]. However, performance measures for the visual feedback systems have not been obtained in the previous works. Since one of the control objective is to track the moving target object, the tracking performance measures are important for the visual feedback systems. In the area of vision and control, it has been expected to investigate not only stability but performance analyses based on theoretical approaches.

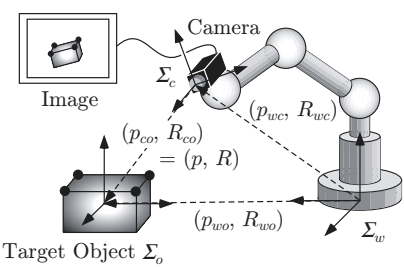


Fig. 1. Eye-in-hand visual feedback system.

In this paper, we propose the vision-based control of robots using position measurements only and discuss stability and L_2-gain performance analysis for the visual feedback system from the theoretical standpoint. The key idea of our proposed method is based on a structural passivity-like property of the visual feedback system. In the nonlinear mechanical systems, an important characteristic of passivity-based control is that the control objective is achieved by reshaping the natural energy of mechanical system [16]. Our previous research [9] has proposed state feedback control which guaranteed local stability and L_2-gain performance analysis based on passivity. This work is a continuation of our previous researches [8][9]. Passivity of the visual feedback system and the design strategy for a velocity observer based on passivity [17] are exploited in our proposed method.

This paper is organized as follows. In Section 2, we consider a model of the relative rigid body motion. Section 3 presents a method for the estimation of the relative rigid body motion and leads the structural passivity-like property of the visual feedback system. Stability and L_2-gain performance analysis for the visual feedback system considering the proposed velocity observer are

discussed based on passivity in Section 4 . Finally, we offer some conclusions in Section 5.

Let a rotation matrix $R_{ab} \in \mathcal{R}^{3\times3}$ represent the change of the principal axes of a frame b relative to a frame a. Then, R_{ab} is known to become orthogonal with unit determinant. Such a matrix belongs to a Lie group of dimension three, called $SO(3) = \{R_{ab} \in \mathcal{R}^{3\times3} | R_{ab}R_{ab}^T = R_{ab}^T R_{ab} = I, \det(R_{ab}) = +1\}$. The configuration space of the rigid body motion is the product space of $\mathcal{R}^3$ with $SO(3)$, which should be denoted as $SE(3)$ throughout this paper (see, e.g. [18]). For any matrix $A(x) = A^T(x) > 0$ and for all x, A_m and A_M represent the minimum and maximum eigenvalue of $A(x)$, respectively. The norm of a vector x and a matrix A are defined as $\|x\| = \sqrt{x^T x}$ and $\|A\| = \sqrt{\lambda_{max}(A^T A)}$, respectively. λ_{max} denotes the maximum eigenvalue.

2 Relative Rigid Body Motion

Let us consider the eye-in-hand system [1] depicted in Fig. 1, where the coordinate frame Σ_w represents the world frame, Σ_c represents the camera (end-effector) frame, and Σ_o represents the object frame, respectively. Let $p_{co} \in \mathcal{R}^3$ and $R_{co} \in \mathcal{R}^{3\times3}$ denote the position vector and the rotation matrix from the camera frame Σ_c to the object frame Σ_o. Then, the relative rigid body motion from Σ_c to Σ_o can be represented by $(p_{co}, R_{co}) \in SE(3)$. Similarly, we will define the rigid body motion (p_{wc}, R_{wc}) from Σ_w to Σ_c, and (p_{wo}, R_{wo}) from Σ_w to Σ_o, respectively, as in Fig. 1.

The objective of the visual feedback control is to bring the actual relative rigid body motion (p_{co}, R_{co}) to a given reference (p_d, R_d) (see, e.g. [1]). The reference (p_d, R_d) for the rigid motion (p_{co}, R_{co}) is assumed to be constant in this paper.

In this section, let us derive a model of the relative rigid body motion. The rigid body motion (p_{wo}, R_{wo}) of the target object, relative to the world frame Σ_w, is given by

$$p_{wo} = p_{wc} + R_{wc}p_{co} \tag{1}$$

$$R_{wo} = R_{wc}R_{co} \tag{2}$$

which is a direct consequence of a transformation of the coordinates [18] in Fig. 1. Using the property of a rotation matrix, i.e. $R^{-1} = R^T$, the rigid body motion (1) and (2) is now rewritten as

$$p_{co} = R_{wc}^T(p_{wo} - p_{wc}) \tag{3}$$

$$R_{co} = R_{wc}^T R_{wo}. \tag{4}$$

The dynamic model of the relative rigid body motion involves the velocity of each rigid body. To this aid, let us consider the velocity of a rigid body [18]. Let $\hat{\omega}_{wc}$ and $\hat{\omega}_{wo}$ denote the instantaneous body angular velocities from

Σ_w to Σ_c, and from Σ_w to Σ_o, respectively. Here the operator '$\wedge$' (wedge), from $\mathcal{R}^3$ to the set of 3×3 skew-symmetric matrices $so(3)$, is defined as

$$\hat{a} = (a)^{\wedge} := \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}, \quad a = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}.$$

The operator '$\vee$' (vee) denotes the inverse operator to '$\wedge$': i.e. $so(3) \to \mathcal{R}^3$. With these, it is possible to specify the velocities of each rigid body as follows [18](Chap.2, (2.55)).

$$\dot{p}_{wc} = R_{wc} v_{wc}, \quad \dot{R}_{wc} = R_{wc} \hat{\omega}_{wc} \tag{5}$$

$$\dot{p}_{wo} = R_{wo} v_{wo}, \quad \dot{R}_{wo} = R_{wo} \hat{\omega}_{wo}. \tag{6}$$

Differentiating (3) and (4) with respect to time, we can obtain

$$\dot{p}_{co} = -v_{wc} + \hat{p}_{co} \omega_{wc} + R_{co} v_{wo} \tag{7}$$

$$\dot{R}_{co} = -\hat{\omega}_{wc} R_{co} + R_{co} \hat{\omega}_{wo}. \tag{8}$$

Now, let us denote the body velocity of the camera relative to the world frame Σ_w as

$$V_{wc} := [v_{wc}^T \ \omega_{wc}^T]^T. \tag{9}$$

Further, the body velocity of the target object relative to Σ_w should be denoted as

$$V_{wo} := [v_{wo}^T \ \omega_{wo}^T]^T. \tag{10}$$

Then we can rearrange (7) and (8) in a matrix form as follows
(Relative Rigid Body Motion : RRBM)

$$\begin{bmatrix} \dot{p} \\ (\dot{R} R^T)^{\vee} \end{bmatrix} = \begin{bmatrix} -I & \hat{p} \\ 0 & -I \end{bmatrix} V_{wc} + \begin{bmatrix} R & 0 \\ 0 & R \end{bmatrix} V_{wo}. \tag{11}$$

Here (p, R) denotes (p_{co}, R_{co}) for short. Equation (11) should be the model of the relative rigid body motion [8].

3 Visual Feedback System

3.1 Estimation of Relative Rigid Body Motion

The visual feedback control task should require the information of the relative rigid body motion (p, R). However, the available information that can be measured in the visual feedback systems is only image information. Hence, let us consider a nonlinear observer which will estimate the relative rigid body motion via image information.

We shall consider the following dynamic model which just comes from the actual relative rigid body motion (11).

$$\begin{bmatrix} \dot{\bar{p}} \\ (\dot{\bar{R}}\bar{R}^T)^\vee \end{bmatrix} = \begin{bmatrix} -I & \hat{\bar{p}} \\ 0 & -I \end{bmatrix} V_{wc} + \begin{bmatrix} I & 0 \\ 0 & \bar{R} \end{bmatrix} u_e \tag{12}$$

where $(\bar{p}, \bar{R})$ is the estimated value of the relative rigid body motion, and a new input u_e for the estimation is to be determined in order to converge the estimated value to the actual relative rigid body motion.

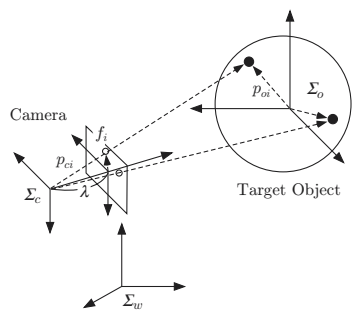


Fig. 2. Pinhole camera model.

Next let us derive a pinhole camera model as shown in Fig. 2. Let λ be a focal length. Let p_{oi} and p_{ci} be coordinates of the target object's i-th feature point relative to Σ_o and Σ_c, respectively. From a transformation of the coordinates, we have

$$p_{ci} = p + Rp_{oi}. \tag{13}$$

The perspective projection of the i-th feature point onto the image plane gives us the image plane coordinate as follows

$$f_i = \frac{\lambda}{z_{ci}} \begin{bmatrix} x_{ci} \\ y_{ci} \end{bmatrix} \tag{14}$$

where $p_{ci} := [x_{ci} \;\; y_{ci} \;\; z_{ci}]^T$. It is straightforward to extend this model to the n feature points case by simply stacking the vectors of the image plane coordinate, i.e. $f := [f_1^T \;\; \cdots \;\; f_n^T]^T = \pi(p, R) \in \mathcal{R}^{2n}$.

Now, we define the estimation error between the estimated value $(\bar{p}, \bar{R})$ and the actual relative rigid motion (p, R) as

$$(p_{ee}, R_{ee}) := (p - \bar{p}, \bar{R}^T R). \tag{15}$$

Note that, if $p = \bar{p}$ and $R = \bar{R}$, then it follows $p_{ee} = 0$ and $R_{ee} = I$. Let the matrix $\mathrm{sk}(R)$ denote $\frac{1}{2}(R - R^T)$ and let

$$e_R(R) := \mathrm{sk}(R)^\vee \tag{16}$$

represent the error vector of the rotation matrix R. Then the vector of the estimation error is given by

$$e_e := \left[p_{ee}^T \; e_R^T(R_{ee}) \right]^T \tag{17}$$

where $e_e = 0$ holds when $p_{ee} = 0$ and $R_{ee} = I$.

Next, we will derive the measurement equation from (13) and (14). Suppose the estimation error is *small* enough that we can let $R_{ee} \simeq I + \mathrm{sk}(R_{ee})$, then (13) becomes

$$p_{ci} = \bar{p}_{ci} - \bar{R}\hat{p}_{oi} e_R(R_{ee}) + p_{ee} \tag{18}$$

where $\bar{p}_{ci} := \bar{p} + \bar{R}p_{oi}$. Using Taylor expansion, (14) can be written as

$$f_i = \bar{f}_i + \begin{bmatrix} \frac{\lambda}{\bar{z}_{ci}} & 0 & -\frac{\lambda \bar{x}_{ci}}{\bar{z}_{ci}^2} \\ 0 & \frac{\lambda}{\bar{z}_{ci}} & -\frac{\lambda \bar{y}_{ci}}{\bar{z}_{ci}^2} \end{bmatrix} (p_{ci} - \bar{p}_{ci}) \tag{19}$$

where $\bar{p}_{ci} = [\bar{x}_{ci} \; \bar{y}_{ci} \; \bar{z}_{ci}]^T$ and $\bar{f}_i := \frac{\lambda}{\bar{z}_{ci}} [\bar{x}_{ci} \; \bar{y}_{ci}]^T$.

An approximation of image information f around the estimated value $(\bar{p}, \bar{R})$ is given by

$$f - \bar{f} = J(\bar{p}, \bar{R}) e_e \tag{20}$$

where the matrix $J(\bar{p}, \bar{R})$ is defined as

$$J(\bar{p}, \bar{R}) := \begin{bmatrix} L(\bar{p}, \bar{R}; p_{o1}) \\ L(\bar{p}, \bar{R}; p_{o2}) \\ \vdots \\ L(\bar{p}, \bar{R}; p_{on}) \end{bmatrix} \tag{21}$$

$$L(\bar{p}, \bar{R}; p_{oi}) := \begin{bmatrix} \frac{\lambda}{\bar{z}_{ci}} & 0 & -\frac{\lambda \bar{x}_{ci}}{\bar{z}_{ci}^2} \\ 0 & \frac{\lambda}{\bar{z}_{ci}} & -\frac{\lambda \bar{y}_{ci}}{\bar{z}_{ci}^2} \end{bmatrix} \begin{bmatrix} I & -\bar{R}\hat{p}_{oi} \end{bmatrix}. \tag{22}$$

Note that the matrix $J(\bar{p}, \bar{R})$ can be interpreted as the image Jacobian. It is known that the appropriate pseudo-inverse of the image Jacobian exists in case of $n \geq 4$ [1].

The above discussion shows that we can derive the vector of the estimation error e_e from image information f and the estimated value of the relative rigid body motion $(\bar{p}, \bar{R})$,

$$e_e = J^\dagger(\bar{p}, \bar{R})(f - \bar{f}) \tag{23}$$

where $\dagger$ denotes the pseudo-inverse.

We consider the state equation of the estimated relative rigid body motion error. Using (11), (12) and (15), the state equation of the estimated RRBM error can be obtained as follows.

$$\begin{bmatrix} \dot{p}_{ee} \\ (\dot{R}_{ee} R_{ee}^T)^\vee \end{bmatrix} = \begin{bmatrix} 0 & \hat{p}_{ee} \\ 0 & 0 \end{bmatrix} V_{wc} - u_e + R_2 V_{wo} \tag{24}$$

where $R_2 = \mathrm{diag}\{R, R_{ee}\}$.

3.2 Passivity of Visual Feedback System

Let us derive a model of the visual feedback system. First, we define the relative rigid body motion error which represents the error between the estimated value $(\bar{p}, \bar{R})$ and the reference of the relative rigid body motion (p_d, R_d) as follows.

$$(p_{ec}, R_{ec}) := (\bar{p} - p_d, \bar{R} R_d^T) \tag{25}$$

It should be remarked that $p_d = \bar{p}$ and $R_d = \bar{R}$ iff $p_{ec} = 0$ and $R_{ec} = I$.

Using the notation $e_R(R)$, the vector of the RRBM error is defined as

$$e_c := \left[p_{ec}^T \; e_R^T(R_{ec}) \right]^T . \tag{26}$$

Note that $e_c = 0$ iff $p_{ec} = 0$ and $R_{ec} = I$.

From (12) and (25), the state equation of the RRBM error can be given by

$$\begin{bmatrix} \dot{p}_{ec} \\ (\dot{R}_{ec} R_{ec}^T)^\vee \end{bmatrix} = -B^T(\bar{p}) u_c + R_1 u_e, \quad B(a) := \begin{bmatrix} I & 0 \\ \hat{a} & I \end{bmatrix} \quad \forall a \in \mathcal{R}^3 \tag{27}$$

where $R_1 = \mathrm{diag}\{I, \bar{R}\}$. Since the camera velocity V_{wc} is considered as an input, the notation u_c is used instead of V_{wc}.

Using (24) and (27), the state equation of the visual feedback system can be derived as

$$\begin{bmatrix} \dot{p}_{ec} \\ (\dot{R}_{ec} R_{ec}^T)^\vee \\ \dot{p}_{ee} \\ (\dot{R}_{ee} R_{ee}^T)^\vee \end{bmatrix} = \begin{bmatrix} -I & \hat{\bar{p}} & I & 0 \\ 0 & -I & 0 & \bar{R} \\ 0 & \hat{p}_{ee} & -I & 0 \\ 0 & 0 & 0 & -I \end{bmatrix} u_r + \begin{bmatrix} 0 \\ R_2 \end{bmatrix} V_{wo} \tag{28}$$

where $u_r := [u_c^T \; u_e^T]^T$ denotes the input of the visual feedback system.

Let us define the error vector of the visual feedback system as

$$e := \left[e_c^T \; e_e^T \right]^T \tag{29}$$

which consists of the RRBM error vector e_c and the estimated RRBM error vector e_e. It should be noted that the actual relative rigid body motion (p, R) tends to the reference (p_d, R_d) when $e \to 0$.

Theorem 1. *[8] If $V_{wo} = 0$ and $e(0) = 0$, then the system (28) satisfies*

$$\int_0^T u_r^T \nu d\tau \geq 0, \quad \forall T > 0 \tag{30}$$

where ν is

$$\nu := \begin{bmatrix} -B(p_d) & 0 \\ R_1^T & -I \end{bmatrix} e. \tag{31}$$

Proof. Consider the following positive definite function

$$W = \frac{1}{2}\|p_{ec}\|^2 + \phi(R_{ec}) + \frac{1}{2}\|p_{ee}\|^2 + \phi(R_{ee}) \tag{32}$$

where ϕ is the error function of the rotation matrix. Let us introduce the notation of the error function

$$\phi(R) := \frac{1}{2}\mathrm{tr}(I - R). \tag{33}$$

The error function ϕ has the following properties [19].

Property 1. Let $R \in SO(3)$. The following properties hold.

(1) $\phi(R) = \phi(R^T) \geq 0$ and $\phi(R) = 0$ iff $R = I$.
(2) $\dot{\phi}(R) = e_R^T(R)(R^T\dot{R})^\vee = e_R^T(R)(\dot{R}R^T)^\vee$.

The positive definiteness of the function W can be given by the property of the error function ϕ. Differentiating (32) with respect to time yields

$$\dot{W} = e^T \begin{bmatrix} \dot{p}_{ec} \\ (\dot{R}_{ec}R_{ec}^T)^\vee \\ \dot{p}_{ee} \\ (\dot{R}_{ee}R_{ee}^T)^\vee \end{bmatrix}. \tag{34}$$

Observing that the skew-symmetry of the matrices $\hat{p}_{ec}$ and $\hat{p}_{ee}$, i.e. $p_{ec}^T\hat{p}_{ec}\omega_{wc} = -p_{ec}^T\hat{\omega}_{wc}p_{ec} = 0$ and $p_{ee}^T\hat{p}_{ee}\omega_{wc} = -p_{ee}^T\hat{\omega}_{wc}p_{ee} = 0$, the above equation along the trajectories of the system (28) can be transformed into

$$\begin{aligned}\dot{W} &= e^T \begin{bmatrix} -I & \hat{\bar{p}} & I & 0 \\ 0 & -I & 0 & \bar{R} \\ 0 & \hat{p}_{ee} & -I & 0 \\ 0 & 0 & 0 & -I \end{bmatrix} u_r = e^T \begin{bmatrix} -B^T(p_d) & R_1 \\ 0 & -I \end{bmatrix} u_r \\ &= u_r^T\nu. \end{aligned} \tag{35}$$

From $e(0) = 0$, $W(e(0)) = 0$ can be derived. Integrating (35) from 0 to T, we can obtain

$$\int_0^T u_r^T\nu d\tau = W(e(T)) - W(e(0)) = W(e(T)) \geq 0. \tag{36}$$

This completes the proof.

Remark 1. In the visual feedback system (28), $p_{ec}^T\hat{\omega}_{wc}p_{ec} = 0$ and $p_{ee}^T\hat{\omega}_{wc}p_{ee} = 0$ hold. This property is analogous to the one of the robot dynamics, i.e. $x^T(\dot{M} - 2C)x = 0$, $\forall x \in \mathcal{R}^m$ (where $M \in \mathcal{R}^{m\times m}$ is the manipulator inertia matrix and $C \in \mathcal{R}^{m\times m}$ is the Coriolis matrix [18]). Moreover, let us take u_r as the input and ν as its output. Then, Theorem 1 would suggest that the system (28) is *passive* from the input u_r to the output ν just formally as in the definition in [20].

It is well known that there is a direct link between passivity and Lyapunov stability. The following control input has been proposed in [8].

$$u_r = -\begin{bmatrix} K_c & 0 \\ 0 & K_e \end{bmatrix} \nu = -\begin{bmatrix} K_c & 0 \\ 0 & K_e \end{bmatrix} \begin{bmatrix} -B(p_d) & 0 \\ R_1^T & -I \end{bmatrix} e \tag{37}$$

where K_c and K_e are 6×6 positive definite matrices. Stability and L_2-gain performance analysis for the visual feedback system have been discussed in [8]. Fig. 3 shows a block diagram of the visual feedback system.

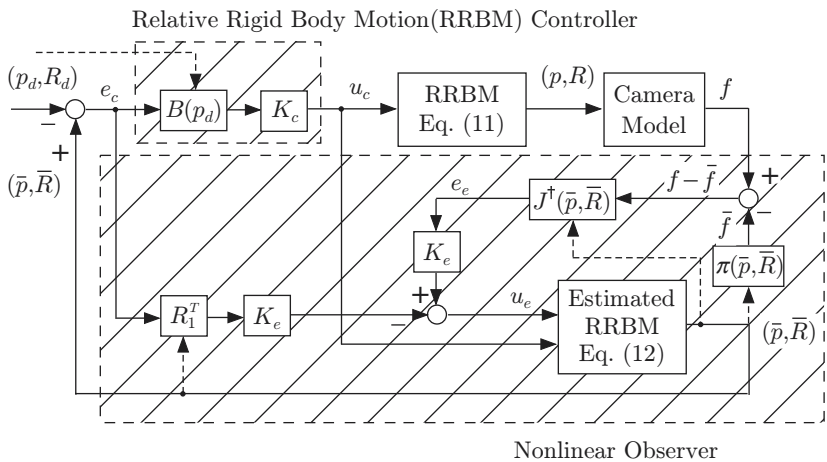


Fig. 3. Block diagram of the visual feedback system.

4 Vision-based Robot Control

4.1 Visual Feedback System with Manipulator Dynamics

The manipulator dynamics can be written as

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau \tag{38}$$

where q, $\dot{q}$ and $\ddot{q}$ are the joint angles, velocities and accelerations, respectively. τ is the vector of the input torques. The matrices $M(q)$ and $C(q,\dot{q})$ have an important property which will be used in the sequel.

Property 2. By defining $C(q,\dot{q})$ using the Christoffel symbols, $\dot{M}(q) - 2C(q,\dot{q})$ is skew-symmetric.

Since the camera is mounted on the end-effector of the manipulator in the eye-in-hand configuration, the body velocity of the camera V_{wc} is given by

$$V_{wc} = J_b(q)\dot{q} \tag{39}$$

where $J_b(q)$ is the manipulator body Jacobian [18]. We assume that the manipulator Jacobian $J_b(q)$ is the nonsingular matrix. Under this assumption, we define the reference of the joint velocities as follows

$$\dot{q}_d := J_b^{-1}(q)u_c \tag{40}$$

where u_c represents an ideal body velocity of the camera.

Let us define the error vector with respect to the joint velocities of the manipulator dynamics as

$$s_1 := \dot{q} - \dot{q}_d. \tag{41}$$

Now, we consider the velocity observer approach [17] in order to estimate the joint velocities. Here we propose the control law as the input torques and the velocity observer for the dynamic visual feedback system as follows.

$$\text{Controller} \quad \begin{cases} \tau = M(q)\ddot{q}_d + C(q,\dot{q}_0)\dot{q}_d + g(q) + u_s \\ \dot{q}_0 = \dot{\tilde{q}} - \Lambda\tilde{q} \end{cases} \tag{42}$$

$$\begin{matrix}\text{Velocity} \\ \text{Observer}\end{matrix} \quad \begin{cases} \dot{\tilde{q}} = z + L_d\tilde{q} \\ \dot{z} = \ddot{q}_d + L_p\tilde{q} \end{cases} \tag{43}$$

where $[\bar{q}^T \ z^T]^T$ is the observer state, $\dot{\tilde{q}}$ denotes the estimated velocities, $\Lambda = \Lambda^T > 0$, $L_d = l_d I + \Lambda > 0$, $L_p = l_d\Lambda > 0$, $l_d > 0$ is scalar. The new input u_s is to be determined in order to achieve the control objectives. Moreover, $\tilde{q}$ is defined as

$$\tilde{q} := q - \bar{q} \tag{44}$$

which represents the error between the actual joint angles and the estimated joint angles. Now we assume that the desired velocities $\dot{q}_d$ are bounded by V_M, i.e. $V_M = \sup_t \|\dot{q}_d(t)\|$.

Here we consider the error dynamics of the manipulator. Substituting (42) into (38) yields

$$M(q)\dot{s}_1 = -C(q,\dot{q})s_1 - C(q,s_2)\dot{q}_d + u_s \tag{45}$$

where s_2 is defined as

$$s_2 := \dot{q} - \dot{q}_0 = \dot{\tilde{q}} + \Lambda\tilde{q}. \tag{46}$$

Next we derive the error dynamics of the velocity observer. From (42) and (43), we have

$$\ddot{q}_0 - \ddot{q}_d = l_d s_2. \tag{47}$$

Hence the error dynamics of the velocity observer can be written as

$$M(q)\dot{s}_2 = -M(q)l_d s_2 + C(q, s_2 - \dot{q})s_1 - C(q, \dot{q})s_2 + u_s. \tag{48}$$

Using (24), (27), (45) and (48), the visual feedback system with manipulator dynamics (we call the dynamic visual feedback system) can be derived as follows

$$M(q)\dot{s}_1 = -C(q, \dot{q})s_1 - C(q, s_2)\dot{q}_d + u_s \tag{49}$$

$$M(q)\dot{s}_2 = -M(q)l_d s_2 + C(q, s_2 - \dot{q})s_1 - C(q, \dot{q})s_2 + u_s \tag{50}$$

$$\begin{bmatrix} \dot{p}_{ec} \\ (\dot{R}_{ec}R_{ec}^T)^\vee \\ \dot{p}_{ee} \\ (\dot{R}_{ee}R_{ee}^T)^\vee \end{bmatrix} = \begin{bmatrix} -I & \hat{\bar{p}} \\ 0 & -I \\ 0 & \hat{p}_{ee} \\ 0 & 0 \end{bmatrix} J_b(q)s_1 + \begin{bmatrix} -I & \hat{\bar{p}} & I & 0 \\ 0 & -I & 0 & \bar{R} \\ 0 & \hat{p}_{ee} & -I & 0 \\ 0 & 0 & 0 & -I \end{bmatrix} \begin{bmatrix} u_c \\ u_e \end{bmatrix} + \begin{bmatrix} 0 \\ R_2 \end{bmatrix} V_{wo}. \tag{51}$$

Equations (49) and (50) represent the manipulator dynamics with the velocity observer. Equation (51) denotes the relative rigid body motion with the nonlinear observer. Fig. 4 shows a block diagram of the visual feedback system with the manipulator dynamics.

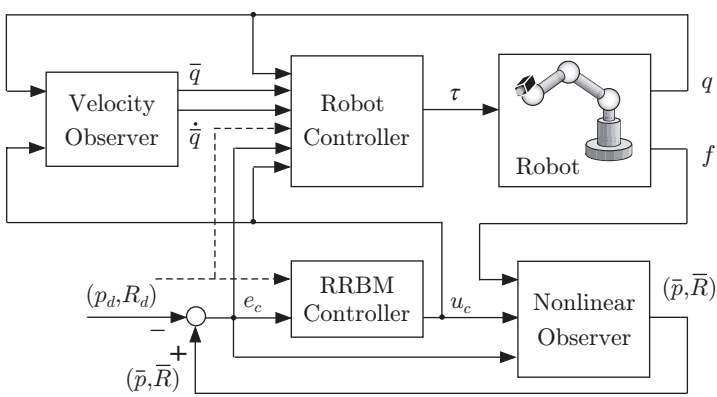


Fig. 4. Block diagram of the dynamic visual feedback system.

4.2 Stability of Vision-based Robot Control

Let us define the error vector of the dynamic visual feedback system as

$$x := \begin{bmatrix} s_1^T & s_2^T & e_c^T & e_e^T \end{bmatrix}^T. \tag{52}$$

Then the dynamic visual feedback control problem can be formulated as follows.

Problem 1. Find a input vector $u = [u_s^T \ u_c^T \ u_e^T]^T$ such that the closed-loop system satisfies the control objectives as follows:
(Internal stability) If the target object is static, i.e. $V_{wo} = 0$, then the equilibrium point $x = 0$ for the closed-loop system is asymptotically stable.
(L_2-gain performance analysis) The closed-loop system has L_2-gain $\leq \gamma$.

We propose the following dynamic visual feedback control law

$$u_s = -K_s(s_1 - s_2) - J_b^T(q)B(p_d)e_c \tag{53}$$

$$\begin{bmatrix} u_c \\ u_e \end{bmatrix} = -\begin{bmatrix} K_c & 0 \\ 0 & K_e \end{bmatrix}\begin{bmatrix} -B(p_d) & 0 \\ R_1^T & -I \end{bmatrix} e \tag{54}$$

where K_s, K_c and K_e are 6×6 positive definite matrices. Note that s_1 and s_2 cannot be realized, whereas the difference $s_1 - s_2$ can be obtained from known signals, i.e., $s_1 - s_2 = \dot{\tilde{q}} - \Lambda\tilde{q} - \dot{q}_d$. Fig. 5 shows a block diagram of the controller for the dynamics visual feedback system.

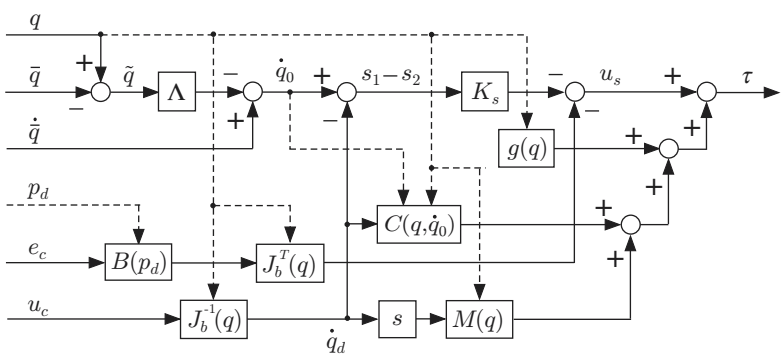


Fig. 5. Block diagram of the controller for the dynamics visual feedback system.

Now, let us define the following matrices.

$$K_{ce} := \begin{bmatrix} B^T(p_d) & -R_1 \\ 0 & I \end{bmatrix}\begin{bmatrix} K_c & 0 \\ 0 & K_e \end{bmatrix}\begin{bmatrix} B(p_d) & 0 \\ -R_1^T & I \end{bmatrix} \tag{55}$$

$$K_J(q) := (J_b^T(q)B(p_d))^T(J_b^T(q)B(p_d)). \tag{56}$$

The result with respect to asymptotic stability for the closed-loop system can be established as follows.

Theorem 2. *Suppose that the following conditions hold.*

$$K_{s,m} > C_M V_M \tag{57}$$

$$l_d > M_m^{-1}(K_{s,M} + \frac{1}{2} + C_M V_M) \tag{58}$$

$$K_{ce,m} > \frac{1}{2}K_{J,M} \tag{59}$$

If $V_{wo} = 0$, then the equilibrium point $x = 0$ for the closed-loop system (49)-(54) is asymptotically stable. Moreover, a region of attraction is given by

$$D_1 = \left\{ x \in \mathcal{R}^{24} \;\middle|\; \frac{2l_d M_m - 2K_{s,M} - 1}{2C_M} - V_M > \|s_1\|, \right. \\ \left. \frac{K_{s,m}}{C_M} - V_M > \|s_2\| \right\}. \tag{60}$$

Proof. Consider the following positive definite function

$$V = \frac{1}{2} s_1^T M s_1 + \frac{1}{2} s_2^T M s_2 + \frac{1}{2}\|p_{ec}\|^2 + \phi(R_{ec}) + \frac{1}{2}\|p_{ee}\|^2 + \phi(R_{ee}). \tag{61}$$

Differentiating (61) with respect to time along the trajectories of the system (49)-(54) yields

$$\dot{V} = -s_1^T K_s s_1 - s_2^T \big(l_d M(q) - K_s - C(q, s_1)\big) s_2 - s_2^T C(q, s_1) s_1 \\ -s_2^T C(q, \dot{q}_d) s_1 - s_1^T C(q, \dot{q}_d) s_2 - s_2^T J_b^T(q) B(p_d) e_c - e^T K_{ce} e \tag{62}$$

where Property 2 is used. Equation (62) can be upper bounded by

$$\dot{V} \le -\big(K_{s,m} - C_M(V_M + \|s_2\|)\big)\|s_1\|^2 - \big(K_{ce,m} - \frac{1}{2} K_{J,M}\big)\|e\|^2 \\ -\big(l_d M_m - K_{s,M} - \frac{1}{2} - C_M(V_M + \|s_1\|)\big)\|s_2\|^2. \tag{63}$$

Hence $\dot{V}$ is negative on $D_1 - \{0\}$. This completes the proof.

In the proof of Theorem 2, the positive definite function V plays the role of a Lyapunov function. In this subsection, we have obtained the rigorous result with respect to stability of the dynamic visual feedback system in a 3-D workspace.

4.3 L_2-Gain Performance Analysis

In case the target object is moving, we consider L_2-gain performance analysis as a tracking performance measure for the dynamic visual feedback system.

Theorem 3. *Given a positive scalar γ and suppose that the following conditions hold.*

$$K_{s,m} > C_M V_M + \frac{1}{2} \tag{64}$$

$$l_d > M_m^{-1}(K_{s,M} + 1 + C_M V_M) \tag{65}$$

$$K_{ce,m} > \frac{1}{2}\left(K_{J,M} + 1 + \frac{1}{\gamma^2}\right) \tag{66}$$

Then the closed-loop system (49)-(54) has L_2-gain $\le \gamma$. Moreover, a region of attraction is given by

$$D_2 = \left\{ x \in \mathcal{R}^{24} \;\middle|\; \frac{l_d M_m - K_{s,M} - 1}{C_M} - V_M > \|s_1\|, \right. \\ \left. \frac{2K_{s,m} - 1}{2C_M} - V_M > \|s_2\| \right\}. \tag{67}$$

Proof. Differentiating the positive definite function V defined in (61) along the trajectory of the closed-loop system and completing the squares yields

$$\begin{aligned}\dot{V} \leq \frac{\gamma^2}{2}\|V_{wo}\|^2 - \frac{\gamma^2}{2}\Big\|V_{wo} - \frac{1}{\gamma^2}\begin{bmatrix}0 & R_2^T\end{bmatrix} e\Big\|^2 + \frac{1}{2\gamma^2}e^T\begin{bmatrix}0 & 0\\ 0 & I\end{bmatrix} e \\ -\big(K_{ce,m} - \frac{1}{2}K_{J,M}\big)\|e\|^2 - \big(K_{s,m} - C_M(V_M + \|s_2\|)\big)\|s_1\|^2 \\ -\big(l_d M_m - K_{s,M} - \frac{1}{2} - C_M(V_M + \|s_1\|)\big)\|s_2\|^2 \qquad (68)\end{aligned}$$

Then the velocity of the target object (in the worst case) should be derived as

$$V_{wo} = \frac{1}{\gamma^2}\begin{bmatrix}0 & R_2^T\end{bmatrix} e. \qquad (69)$$

Hence for any V_{wo} it can be verified that the inequality

$$\begin{aligned}\dot{V} + \frac{1}{2}\|x\|^2 - \frac{\gamma^2}{2}\|V_{wo}\|^2 \leq -\big(K_{s,m} - C_M(V_M + \|s_2\|) - \frac{1}{2}\big)\|s_1\|^2 \\ -\big(l_d M_m - K_{s,M} - 1 - C_M(V_M + \|s_1\|)\big)\|s_2\|^2 \\ -\big(K_{ce,m} - \frac{1}{2}K_{J,M} - \frac{1}{2} - \frac{1}{2\gamma^2}\big)\|e\|^2 \leq 0 \qquad (70)\end{aligned}$$

holds under the conditions (64)–(66). Integrating (70) from 0 to T and noticing $V(x(T)) \geq 0$, we have

$$\int_0^T \|x\|^2 dt \leq \gamma^2 \int_0^T \|V_{wo}\|^2 dt + 2V(x(0)),\ \forall T > 0. \qquad (71)$$

This completes the proof.

L_2-gain performance analysis can be regarded as a tracking performance measure for the moving target object. The positive definite function V plays the role of the storage function for L_2-gain performance analysis.

5 Conclusions

This paper has presented the vision-based control of robots using position measurements only. Stability and L_2-gain performance analysis for the dynamic visual feedback system have been discussed. Our proposed method has been based on the structural passivity-like property of the visual feedback system. The nonlinear observer has been employed in order to estimate the relative rigid body motion. Furthermore, we have proposed the velocity observer with the aim of obtaining the joint velocities.

Our proposed passivity approach will enrich the field of visual servoing, although there are some problems, e.g., global stability, visibility and so on. In particular, motion planning based on optimal control for the proposed framework is an important direction for our future work. Moreover, we expect to systematize the passivity based visual feedback control as well as the theory of the robot control based on the passivity approach.

References

1. Hutchinson S., Hager G.D., Corke P.I. (1996) A Tutorial on Visual Servo Control. IEEE Trans. Robot. Automat. 12:651–670
2. Espiau B., Chaumette F., Rives P. (1992) A New Approach to Visual Servoing in Robotics. IEEE Trans. Robot. Automat. 8:313–326
3. Papanikolopoulos N.P., Khosla P.K., Kanade T. (1993) Visual Tracking of a Moving Target by a Camera Mounted on a Robot: A Combination of Control and Vision IEEE Trans. Robot. Automat. 9:14–35
4. Wilson W.J., Williams Hulls C.C., Bell G.S. (1996) Relative End-Effector Control Using Cartesian Position Based Visual Servoing. IEEE Trans. Robot. Automat. 12:684–696
5. Hashimoto K., Kimura H. (1993) Dynamic Visual Servoing with Nonlinear Model-based Control. Proc. 12th IFAC World Congr. 9:405–408
6. Kelly R., Carelli R., Nasisi O., Kuchen B., Reyes F. (2000) Stable Visual Servoing of Camera-in-Hand Robotic Systems. IEEE/ASME Trans. Mechatron. 5:39–48
7. Zergeroglu E., Dawson D.M., de Queiroz M.S., Behal A. (2001) Vision-Based Nonlinear Tracking Controllers with Uncertain Robot-Camera Parameters. IEEE Trans. Mechat. 6:322–337
8. Maruyama A., Fujita M. (1999) Visual Feedback Control of Rigid Body Motion Base on Dissipation Theoretical Approach. Proc. 38th IEEE Conf. Decision Cont. 4161–4166
9. Maruyama A., Kawai H., Fujita M. (2001) Stability and Tracking Performance of Dynamic Visual Feedback Control for Nonlinear Mechanical Systems. Proc. 40th IEEE Conf. Decision Cont. 4415–4420
10. F. Chaumette (1998) Potential Problems of Stability and Convergence in Visual Servoing. In:Kriegman D.J., Hager G.D., Morse A.S.(Eds) The Confluence of Vision and Control. Springer, London, 68–78
11. Malis E., Chaumette F., Boudet S. (1999) 2-1/2-D Visual Servoing. IEEE Trans. Robot. Automat. 15:238–250
12. Cowan N.J., Koditschek D.E. (1999) Planar Image Based Visual Servoing as a Navigation Problem. Proc. 1999 IEEE Int. Conf. Robot. Automat. 611–617
13. Conticelli F., Allotta B. (2001) Nonlinear Controllability and Stability Analysis of Adaptive Image-Based Systems. IEEE Trans. Robot. Automat. 17:208–214
14. Mezouar Y., Chaumette F. (2000) Path Planning in Image Space for Robust Visual Servoing. Proc. IEEE Int. Conf. Robot. Automat. 2759–2764
15. Ahang H., Ostrwski J.P. (2002) Visual Motion Planning for Mobile Robots. IEEE Trans. Robot. Automat. 18:199–208
16. Arimoto S. (1996) Control Theory of Non-linear Mechanical Systems – A Passivity-based and Circuit-theoretic Approach. Oxford University Press, New York
17. Berghuis H., Nijmeijer H. (1993) A Passivity Approach to Controller-Observer Design for Robots. IEEE Trans. Robot. Automat. 9:740–754
18. Murray R., Li Z., Sastry S.S. (1994) A Mathematical Introduction to Robotic Manipulation. CRC Press
19. Bullo F., Murray R. (1999) Tracking for Fully Actuated Mechanical Systems : a Geometric Framework. Automatica. 35:17–34
20. van der Schaft A. (2000) L_2-Gain and Passivity Techniques in Nonlinear Control, 2nd edn. Springer, London

Visual Servoing Along Epipoles

Jacopo Piazzi, Domenico Prattichizzo, and Antonio Vicino

Dipartimento di Ingegneria dell'Informazione - Università di Siena
Via Roma, 56 - 53100 Siena, Italy {piazzi,prattichizzo,vicino}@dii.unisi.it

Abstract. A visual servoing algorithm for hand-eye robotic system based on epipolar geometry is proposed. The control law is based on the estimation of the epipoles position obtained by points correspondences extracted from the current and target images. The camera-robot motion is computed from the observation of the epipoles coordinates. Only the principal camera point is assumed to be known but not the other intrinsic parameters. Experimental results are reported to validate the visual servoing algorithm proposed.

1 Introduction

This paper proposes a visual servoing algorithm for robotic systems based on epipolar geometry. The control law is based on the estimation of the epipoles position obtained by points correspondences extracted from the current and target images.

The basic idea behind visual servoing [18] consists of compensating the robot position error exploiting a visual feedback. A reference tutorial on visual servoing is [19] where authors discuss the two possible approaches: *image–based* and *position–based* visual servoing. In position–based control, features are extracted from the image and used, together with a model of the 3D structure of the observed scene and of the camera, to estimate the pose of the target with respect to the camera. The relevant features for pose estimation usually are retrieved from CAD-models of the target objects [20]. In the image-based or feature-based approach, visual servoing algorithms make use of object cues whose image plane projections are controlled to desired positions through the visual servoing process. Usually, these cues are distinctive textures, like corners, of objects in the 3D scene. The relationship between the differential changes of image features and robot positions is captured by the *image Jacobian* or *interaction matrix* [11,1,14,17].

An interesting approach for the robot control through visual feedback can be found in [23,12] where authors combine a partial pose estimation with the image-based servoing. The approach known as 2D 1/2 visual servoing does not need any geometric 3D model of the object. Another hybrid method is presented in [10] where authors decouple the z–axis rotational and translational components of the control from the other degrees of freedom.

To avoid stability and convergence problems, a fundamental requirement on visual servoing algorithms is the ability of keeping all the features of

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 215–231, 2003

interest in the field of view during the robot–camera motion [2,8]. Another interesting problem in visual servoing is that of avoiding joint limits. Recently, in [3] authors propose an efficient and iterative method to implement the gradient projection approach to avoid the joint limits in visual servoing.

This paper proposes a visual servoing technique which is based on the epipolar geometry existing between the current and the desired images. The epipolar geometry is the intrinsic projective geometry between two views. It is independent of the scene structure, and only depends on the camera internal parameters and relative pose [16]. In [26] the epipolar geometry is used in a visual homing framework to recover the difference between the position of the cameras. In the present paper an uncalibrated camera is considered. In particular, the internal camera parameters are supposed to be unknown with the exception of the principal camera point. The intrinsic geometry between two views is gathered in the fundamental matrix which can be used to retrieve information on camera pose [22,28].

This paper builds upon previous contributions [7,6] where authors propose a visual servoing algorithm for planar navigation exploiting the epipolar geometry and some special symmetry conditions of the epipoles. This work extends those results to a general 3D motion of an uncalibrated camera with fully actuated six DoFs. Feature point correspondences are used to estimate the fundamental matrix [13] and the epipoles whereby the camera is steered to the desired image position.

This paper takes into account only scenes with static objects. Work is in progress to extend the approach to scenes with moving objects. In order to analyze dynamic scenes in the epipolar geometry setting, a possible approach is that of using the multibody epipolar constraints and its associated multibody fundamental matrix as proposed in [27].

2 Notation

Let $\tilde{\boldsymbol{M}} = [X, Y, Z, 1]^T$ and $\tilde{\boldsymbol{m}} = [u, v, 1]^T$ be the homogeneous coordinates of a 3D point in the world-frame and the 2D point in the CCD-frame, respectively. The pinhole-camera model is used: the relationship between a 3D point $\boldsymbol{M}$ and its image projection $\boldsymbol{m}$ is

$$s\tilde{\boldsymbol{m}} = K[R|t]\tilde{\boldsymbol{M}} \quad \text{with} \quad K = \begin{pmatrix} \alpha_u & c & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{1}$$

where s is an arbitrary scale factor, (R, t) are the extrinsic parameters (the rotation and translation between the world and camera frames), K is the camera intrinsic matrix, (u_0, v_0) are the coordinates of the principal point, $\alpha_u = f\kappa_u$ and $\alpha_v = f\kappa_v$ being f the focal length and κ_u and κ_v the CCD scaling factors and c the parameter describing the skewness of the two image axes. A special role is played by the coordinates of the principal point (u_0, v_0)

which is the intersection of the optical axis with the CCD. In fact parameters u_0 and v_0 are the only calibration matrix parameters which are supposed to be known in order to implement the visual servoing algorithm.

Moreover, let $Tr_x(d)$ and $Rot_x(\theta)$ be the translation of length d along the axis x and a rotation of angle θ about the axis x, respectively. Finally, let $\tilde{m}$ represent the homogeneous representation of vector m.

2.1 Epipoles estimation

Epipolar geometry is here recalled for the reader convenience [13,9,16]. Consider a pair of camera positions with optical centers O_c and O_d, principal axes (X_c, Y_c, Z_c) and (X_d, Y_d, Z_d) and optical axes Z_c and Z_d. In the visual servoing framework, the two camera positions refer to the current or actual position (c) and to the desired or target position (d).

The segment $\overline{O_c O_d}$ is referred to as the *baseline* and its intersections with the image planes define the *epipoles* e_c and e_d. As far as intrinsic camera matrix is concerned, only the coordinates of the principal point (u_0, v_0) are assumed to be known. This allows to refer the epipoles, in pixel coordinate, to the center of the CCD. The image line passing through the epipole and the image center is called the *horizon line*, while any plane containing the baseline is referred to as *epipolar plane* [13].

Given two views of a scene and a set of corresponding image points, $\tilde{m}_c$ and $\tilde{m}_d$, representing the same point in the 3D space in homogeneous coordinates, a matrix exists $F \in \mathcal{R}^{3\times 3}$ of rank 2, defined up to an arbitrary scale factor and referred to as the *fundamental matrix* [13], such that:

$$\tilde{m}_c^T F \tilde{m}_d = 0 \; . \tag{2}$$

For any point $\tilde{m}_c$ ($\tilde{m}_d$) in one view, the equation $l_d = \tilde{m}_c^T F$ ($l_c = \tilde{m}_d^T F^T$) defines a line, called the epipolar line, in the other view such that the corresponding point $\tilde{m}_c$ ($\tilde{m}_d$) belongs to this line. Moreover, the epipole e_d (e_c) is determined by the null right (one dimensional) space of F (F^T).

The geometric distance between an image point $\tilde{m}_c$ and the epipolar line $l_c = F\tilde{m}_d$ is

$$d(\tilde{m}_c, l_c) = \sqrt{\frac{(\tilde{m}_c^T F \tilde{m}_d)^2}{(F\tilde{m}_d)_1^2 + (F\tilde{m}_d)_2^2}} \tag{3}$$

being $(F\tilde{m}_d)_1$ and $(F\tilde{m}_d)_2$ the first and the second components of the epipolar vector $(F\tilde{m}_d)$ [22]. Note that if points $\tilde{m}_c$ and $\tilde{m}_d$ correspond to the same point in the three-dimensional scene, distance (3) is zero.

The proposed visual servoing algorithm involves the estimation of the fundamental matrix whereby the epipoles can be retrieved. Many algorithms have been presented in the literature to estimate this matrix [16,4]. A linear

algorithm based on 8 corresponding points was introduced in [21]. The linear constraints imposed by equation (2) on the elements of F are expressed as

$$A\boldsymbol{f} = 0 \tag{4}$$

where A is an (8×9) matrix containing the coordinates of the corresponding points, and $\boldsymbol{f}$ is the vector of the parameters (the 9 entries of F) [16]. Using more than 8 points leads to an overdetermined set of equations which can be formulated as a minimization problem

$$min\|A\boldsymbol{f}\| \text{ constrained to } \|\boldsymbol{f}\| = 1. \tag{5}$$

The solution to this problem is the eigenvector associated with the smallest singular value of A. Note that, except for very accurate point locations, the F matrix will not have rank 2, so it is useful to replace F by the rank deficient matrix which minimises the Frobenius distance [21]. The error sensitivity of the linear criteria can be reduced by means of an input coordinate transformation, known as "normalization of the input" [16,22].

A different approach is proposed in [4]. It consists of a constrained least-squares technique where the rank condition of the matrix is ensured by the constraint. Thus the singularity of the matrix is imposed a priori instead of forcing it after the minimization procedure as in the linear criterion.

In general the estimation of F, obtained by linear criteria, is used as starting point for most accurate estimation methods involving non linear optimization algorithms. In particular, in this work the Levenberg-Marquardt non–linear minimization procedure [24] has been used. This approach is known in literature as the *gradient criterion* [16].

3 Visual Servoing Algorithm

The visual servoing algorithm proposed in this paper is based on epipole position referred to the principal point (u_0, v_0) which is supposed to be known (Fig. 1). At each step of the visual servoing algorithm, two images of the same scene are available: the current and the desired one. The epipolar geometry describes the intrinsic projective geometry between these two views. More in detail, the visual servoing algorithm steers the current and desired epipoles along trajectories such that the current image converges to the desired one. The control is a function of the epipole positions referred to the principal point which is supposed to be known. Thus at each sample time an epipole estimation phase is needed to implement the algorithm.

3.1 Algorithm overview

Without loss of generality, consider the *target camera* frame as the world frame. All vectors are expressed in the target camera frame unless otherwise

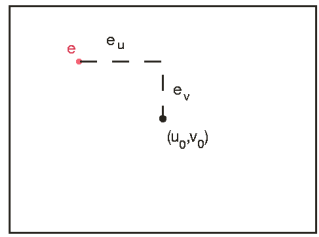


Fig. 1. The epipole position is referred to the principal point (u_0, v_0) which is supposed to be known.

specified. The vector of relative translation T_d^c is the vector from the origin O_d to the center of the *current camera* O_c while the orientation is expressed by the rotation matrix R_d^c. The current and desired epipoles (with respect to the known principal point) are function of the intrinsic camera matrix, current–target camera translation and rotation

$$\tilde{e}_c = f(R_d^c, T_d^c, K); \quad \tilde{e}_d = g(T_d^c, K).$$

The algorithm is based on epipolar geometry in the sense that it is possible to retrieve information on the spatial configuration of the two camera frames needed to move the robot by simply observing the epipoles coordinates in the image planes. For instance, coplanarity of the X_cZ_c plane of the current camera with the X_dZ_d plane of the target camera can be detected by checking that the v–components of the two epipoles are both equal to zero.

The main idea of the algorithm consists in performing a sequence of translations and rotations, depending on the epipole coordinates, in order to reach special 3D configurations of the cameras until the last one, corresponding to $T_d^c = 0$ and $R_d^c = I$, is gained.

The proposed visual servoing consists of 3 phases which allow to home the current camera:

Phase I: Reaching the main plane. Translate, along a suitable direction, the origin of the current camera O_c on the *main plane* defined as the X_dZ_d plane of the camera target.

Phase II: Getting coplanarity. Rotate the current X_cZ_c camera plane to get coplanarity with the X_dZ_d main plane.

Phase III: 2D homing Execute a planar motion on the X_dZ_d main plane to home the current camera.

Fig 2 shows the action of the three phases from their starting configurations to the ending ones. The last phase is performed according to the planar visual servoing algorithm proposed in [7,6].

The following remarks resume some kinematic property of the epipoles subject to motions of the current camera. These are the foundations of

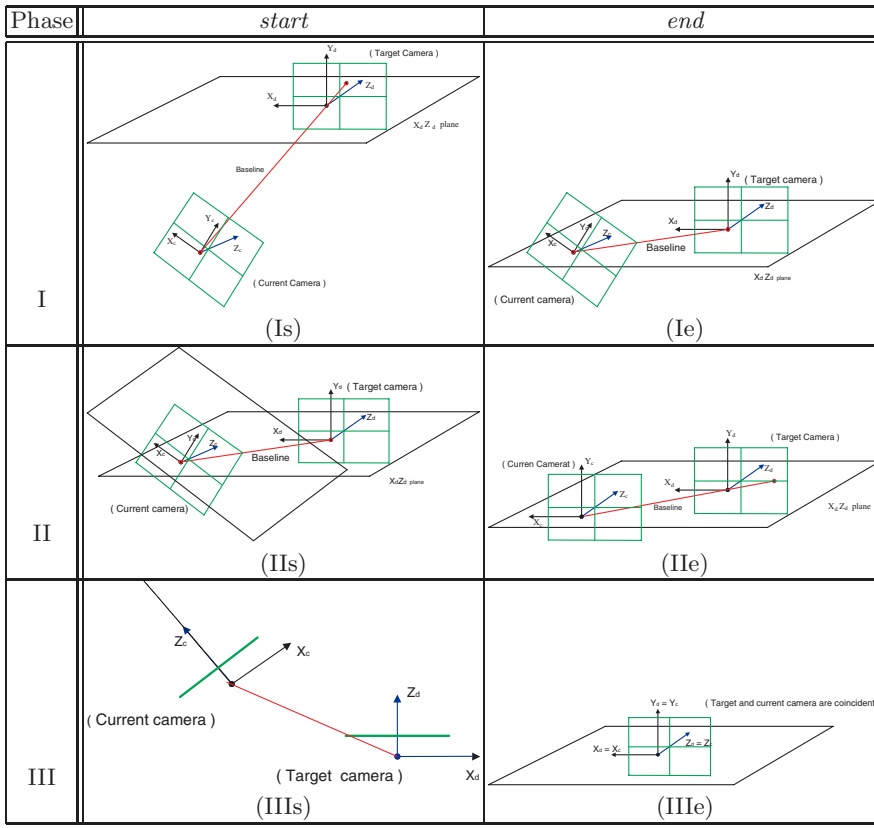


Fig. 2. The visual servoing procedure consists of three phases (rows). The first column describes a possible initial configuration of the three phases while the last column reports the ending configuration of the phases.

the proposed visual servoing algorithm. Without loss of generality, refer the epipole position to the principal point (see Fig. 1).

Property 1. The epipole e_d (e_c) has zero ordinate $e_{(d,v)} = 0$ ($e_{(c,v)} = 0$) if and only if the baseline belongs to the X_dZ_d (X_cZ_c) plane.

Property 2. Refer to two cameras possibly with different internal camera matrices, the two epipoles e_c and e_d have zero v–component ($e_{(c,v)} = 0, e_{(d,v)} = 0$) if and only if the baseline belongs both to the X_cZ_c and X_dZ_d planes.

Property 3. Consider a fixed desired camera position. Then, when the current camera performs a complete rotation about the Z_c axis, the corresponding epipole e_c describes an ellipse centered in the principal point which reduces to a circle if $\alpha_u = \alpha_v$.

In the following, the three phases of the proposed visual servoing algorithm are described.

3.2 Phase I: Reaching the X_dZ_d main plane

This phase is based on Property 1. The origin of the current camera O_c is steered to the desired X_dZ_d plane. The variable to be checked is the ordinate of the desired epipole $e_{(d,v)}$ since it is zero when O_c lies on the X_dZ_d main plane (Property 1). At the end of Phase I, the baseline (and O_c which belongs to the baseline) lies on the X_dZ_d plane. In order to execute this phase two actions are performed.

Action Ia: *Choosing the direction of translation.*

The direction of translation must be chosen in order to minimize the length of the trajectory to steer the optical center O_c on the X_dZ_d plane (see Fig. 3). The direction of translation is chosen within the X_cY_c plane (of the current camera) and in particular the Y_c axis is chosen. The action performs rotations about the optical axis Z_c in order to choose the the direction of translation which maximizes the variation of $e_{(d,v)}$. The non orthogonality of the Z_c axis and the X_dZ_d main plane is assumed.

Note that, rotations about the optical axis have been preferred because these allow to keep the features in the field of view [8]. An example of this action is reported in Fig. 3.

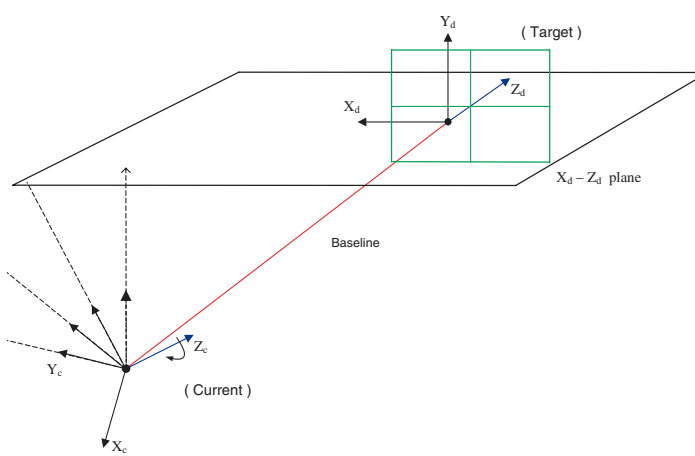


Fig. 3. Choosing the direction of translation which maximizes the epipole ordinate variation.

The i–th step of Action Ia is summarized in the following algorithm.

Algorithm Ia: *Choosing the direction of translation (i–th step).*

(i) Translate along Y_c of a given length β
(ii) Estimate the v–coordinate $e_{(d,v)}$ and compute variation

$$\Delta e_{(d,v)}(i) = e_{(d,v)}(i) - e_{(d,v)}(i-1)$$

(iii) if $\Delta e_{(d,v)}(i)$ is maximum then exit, else
(iv) - translate along Y_c of $-\beta$
- rotate about Z_c of

$$\theta_i = \gamma(\Delta e_{(d,v)}(i) - \Delta e_{(d,v)}(i-1))$$

- translate along Y_c of β
(v) Return to Step (ii)

being γ a proportional coefficient. Note that Step (iv) can be expressed in a compact form as

$$H_i^{i+1}(\theta_i) = Tr_y(-\beta)Rot_z(\theta_i)Tr_y(\beta). \tag{6}$$

being $H_i^{i+1}(\cdot)$ the homogeneous transformation that the robot must perform at the step i to get the position at the step $i+1$.

Action Ib: *Steering O_c on the X_dZ_d plane.*

Once the direction of translation has been selected, the current optical center O_c is translated with a proportional law and control error equal to the v–coordinate, in modulus, of the desired epipole

$$\alpha_i = \lambda |e_{(d,v)_i}| \tag{7}$$

$$H_i^{i+1}(\alpha_i) = Tr_y(\alpha_i) \tag{8}$$

being λ a proportional coefficient.

Action Ia and Ib which have been separately discussed can be simultaneously performed.

3.3 Phase II: Coplanarity of X_cZ_c and X_dZ_d planes

Once the optical center of the current camera has been steered onto the X_dZ_d plane of the desired camera (Fig. 2–(Ie)), the visual servoing algorithm steers the whole current X_cZ_c plane on the X_dZ_d plane which in general are not coplanar (Fig. 2–(IIs)). Properties 3 and 2 are involved in this phase which consists of two actions.

Action IIa: *Steering the baseline on the X_cZ_c plane.*

Note that at the end of Phase II, the baseline already lies on the X_dZ_d plane. This action is performed by rotating the camera about the current optical axis Z_c (Property 3). While in the previous phase, the basic parameter of the epipolar geometry was the ordinate of the desired epipole, here the variable to be checked is the ordinate of the current epipole (Property 2) which goes to zero when the baseline lies on the X_cZ_c plane, see Fig. 4.

In Action IIa rotations about the Z_c axis are controlled by the v–component $e_{(c,v)}$ of the current epipole.

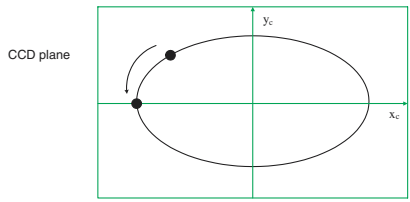


Fig. 4. Current epipole motion under rotations about the optical axis. The ordinate is zero when the baseline belongs to the X_cZ_c plane

Rotation of the current camera at the step i is given by

$$\theta_i = \zeta |e_{(c,v)_i}| \tag{9}$$

$$(H)_i^{i+1} = Rot_z(\theta_i) \tag{10}$$

where ζ is a proportional coefficient.

Recall that at the end of Action IIa the procedure only guarantees that the baseline belongs to both XZ camera planes and not that these are coplanar. Coplanarity will be gained by Action IIb.

Action IIb: *Gaining coplanarity.*

The action allows to get coplanarity of the current camera X_cZ_c plane and the desired camera X_dZ_d plane. Action IIb consists of

- a coplanarity test and
- a motion to get coplanarity.

Coplanarity test. Coplanarity condition is simply checked by performing a test motion, for instance a simple translation along a direction on the X_cY_c plane. If the planes were coplanar, then this test motion would maintain the target epipole ordinate $e_{(d,v)}$ equal to zero (Fig. 5) otherwise, the planes are not coplanar.

Note that, this coplanarity test holds only if the direction of translation is not aligned with the baseline because the epipoles are invariant with respect to translations along the baseline. In this work the coplanarity test translation is chosen along the optical axis which allows to keep the features in the field of view.

Motion to get coplanarity. If the coplanarity test fails, a motion of the current camera is needed to get coplanarity of the X_cZ_c and X_dZ_d planes. For the sake of simplicity assume that the baseline is not aligned neither with the current optical axis Z_c, nor with the X_c axis.

Coplanarity is then gained by performing rotations about X_c (Algorithm 2–a) and Z_c axes.

Algorithm IIa: *Steering Z_c on the plane X_dZ_d (i–th step)*

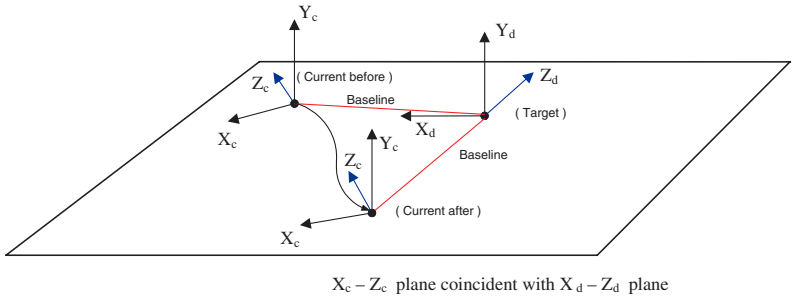


Fig. 5. When the X_cZ_c plane and the X_dZ_d main plane are coplanar, the epipole's ordinate does not change with respect to planar motions on the current X_cZ_c plane.

(i) Translate along Z_c of a given length β,
(ii) Estimate the v–coordinate $e_{(d,v)}$ of the epipole. If it is zero then exit, else
(iii) - translate along Z_c of $-\beta$
- rotate about X_c of an angle

$$\theta_{x,i} = \eta e_{(d,v)}(i).$$

- translate along Z_c of β
(iv) Return to Step (iii).

Note that Step (iii) can be described in a compact form as

$$H_i^{i+1}(\theta_{x,i}) = Tr_z(-\beta)Rot_x(\theta_{x,i})Tr_z(\beta).$$

Translation along the Z_c direction is required to check if Z_c lies on the plane X_dZ_d.

Fig. 6 illustrates the basic motions of Algorithm IIa.

A similar algorithm can be designed to steer the X_c axis on the X_dZ_d plane.

At the end of Phase II, the current and the target cameras differ for a planar roto–translation on the X_dZ_d plane which will be performed by the current camera in the next and last phase.

3.4 Phase III: 2D homing

During the third phase the current camera motion is constrained to lie on the X_cZ_c plane which is coplanar with the X_dZ_d plane. In other terms, in this phase the motion of the current camera keeps the Y_c axis parallel to the Y_d axis of the desired camera frame. In this phase visual servoing involves 3 DoFs only. The problem fits exactly the setting given in [7,6] where an epipole-based visual servoing procedure is proposed to steer a fully-actuated

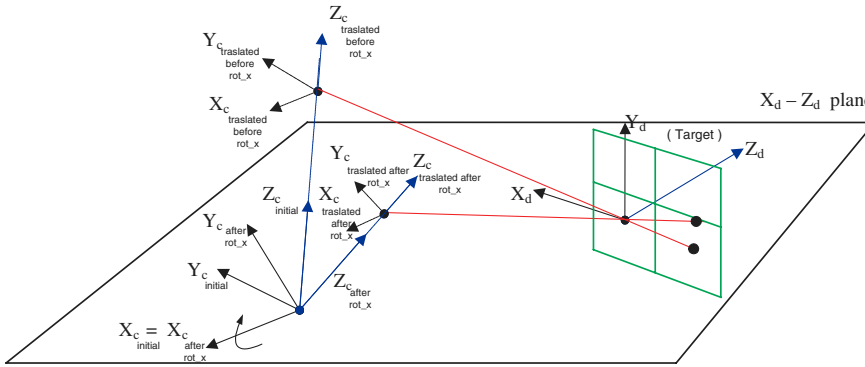


Fig. 6. The Z_c axis is steered on the $X_d Z_d$ main plane.

3 DoFs robot camera to the desired position. For further details the reader is referred to [7,6].

The planar visual servoing algorithm is based on the estimation of a measure of the symmetry of the epipoles and the trajectory consists of a translation along the optical axis to reach a circle where epipoles are symmetric and a motion tracking such circle until convergence of current and target camera frames is gained. A typical trajectory on the plane is that reported in Fig. 7.

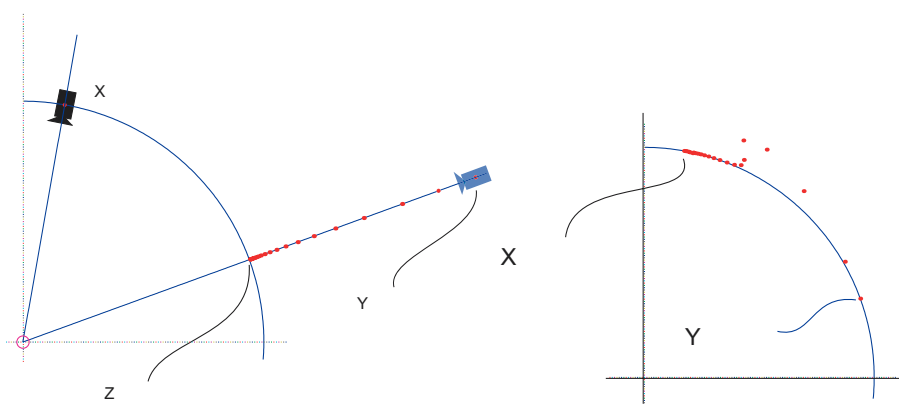


Fig. 7. Motion on the main plane. Translational motion along the optical axis (left). Tracking the circle of epipolar symmetry (right).

Fig. 8. Initial (left) and desired (right) camera-robot configuration.

4 Experiments

The proposed visual servoing algorithm has been experimentally validated. The estimation of the epipolar geometry plays a fundamental role in the overall system performance. For each grabbed image 21 point features have been taken and used for fundamental matrix estimation. The experiment has been designed with the aim of testing the efficiency of the epipolar geometry in visual servoing algorithms. The epipole coordinates have been computed through the fundamental matrix F estimated by the Gradient Criterion [25]. From the feature correspondences, the initial estimation of the fundamental matrix F with linear methods have been obtained. Then a non linear optimization process has been initialized and performed.

The experimental testbed consists of a 6–DoFs robot arm PUMA 560 by Unimation with a CCD camera (HITACHI KP-D50) fixed to the end-effector (Fig. 8). The target image has been grabbed from an unknown camera position. The current image was grabbed each 50msecs, it was properly processed to reduce the distortion, to enhance the contrast and to solve the 21 features correspondences problem.

The initial and (unknown) target camera-robot positions are reported in Fig. 8. The three phases of the proposed visual servoing algorithm are performed. Algorithm parameters are set to $\gamma = 0.4\pi$rad/pixel, $\lambda = 0.3$cm/pixel, $\zeta = 0.45\pi$rad/pixel, $\eta = 0.1\pi$rad/pixel. The zero-threshold for epipoles coordinates is set to 10 pixels.

In Fig. 9, the plot of real data for the desired epipole ordinate $e_{(d,v)}$ is reported. The plot refers to Phase I, where the optical center of the current camera is steered on the (unknown) X_dZ_d plane of the desired camera. Note that, the first 30 steps are used to find the optimal direction of translation as described in Action Ia of Phase I. During the last steps, the camera-robot translates its optical center O_c along the chosen direction until the $e_{(d,v)}$ component is close to zero (less than 10 pixels) which corresponds to have

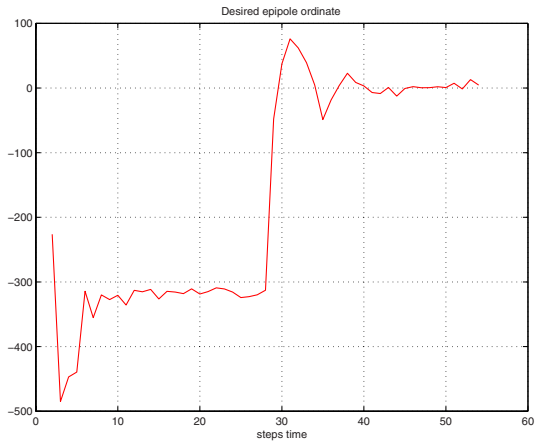


Fig. 9. Desired epipole's ordinate motion in Phase I.

the optical center O_c lying on the X_dZ_d plane. A sketch of this phase is reported in Fig. 3

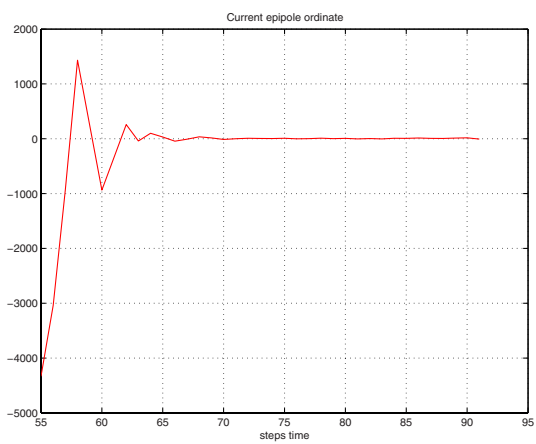


Fig. 10. Actual epipole's ordinate in Phase II

In Fig. 10, the plot of real data for the current epipole v–component $e_{(c,v)}$ is reported. The plot refers to Phase II, where the plane X_cZ_c is rotated, about the Z_c axis, in order to obtain the baseline as intersection of the current and desired planes ($e_{(c,v)} = 0$) as described in Action IIa of Phase II. The convergence to the zero value for the current epipole v–component is gained in about 20 steps. A sketch of this action is reported in Fig. 4.

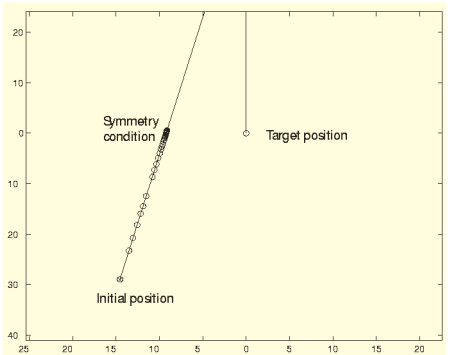


Fig. 11. Reaching the circle of epipolar symmetry.

In Phase III, the current camera motion is constrained to lie on the $X_d Z_d$ main plane. The motion is based on the estimation of a measure of the symmetry of the epipoles and consists of a translation along the optical axis, shown in Fig. 11, to reach the circle with epipolar symmetry and a motion tracking such circle until current and target camera frames are coincident as shown in Fig. 12.

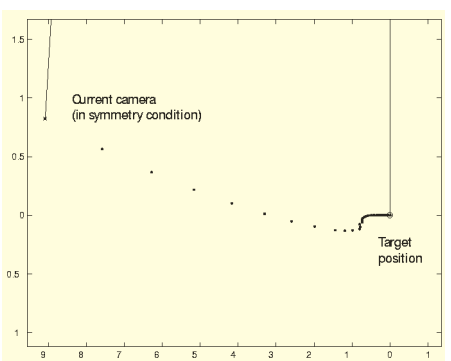


Fig. 12. Reaching the target by tracking the circle of epipolar symmetry.

5 Conclusions and Open Problems

Epipolar geometry was exploited to design an image–based visual servoing algorithm for hand–eye robotic system with uncalibrated camera. The visual servoing algorithm is based on a measure of the epipoles obtained from scene point features.

The main idea is to use only the information gained from the epipoles coordinates to perform the visual servoing. The proposed method relies on extracting epipoles from feature points of the current and target images. This phase must be robustly designed since epipoles estimation is sensitive to image noise.

Designing visual servoing using the epipoles kinematics is certainly a promising idea, however many theoretical and practical problems remain to be solved. The first one concerns the fundamental matrix estimation. The motion of the epipoles is observed from the fundamental matrix whose estimation is based on point correspondences between the current and target images. Although, several methods have been proposed such as non linear methods minimizing distances of points to epipolar lines or between observation and reprojection [16,15,13], further investigation is needed to evaluate performances of epipole estimators in a visual servoing context.

Another interesting research perspective is to consider 3D scenes which do not exhibit any appropriate textures but only smooth surfaces whose main features consist of their apparent contours [9]. In this case the main problem is to estimate the epipoles from apparent contours. Preliminary results have been obtained for the planar case [7]. Work is in progress to extend these results to the 3D fully actuated camera motion. Most of the theory involved in this research are common to the computer vision area dealing with object surface reconstruct using apparent contours and profiles [9,5].

The extension to the contour–based 3D epipolar visual servoing appears as an exciting challenge where the epipole evaluation plays a crucial role. A possible way to enforce the estimation of the epipolar geometry for the contour case is to use more than two images in the visual servoing process. The underlying theory in this case is that of multipleview geometry [16].

References

1. Z. Bien, W. Jang, and J. Park. Characterization and use of featurejacobian matrix for visual servoing. In K. Hashimoto, editor, *Visual servoing: real–time control of robot manipulators based on visual sensory feedback*, Series in Robotics and Automated Systems, pages 317–363. World Scientific, 1993.
2. F. Chaumette. Potential problems of stability and convergence in image-based and positionbased visual servoing. In G. Hager, D. Kriegman, and A. Morse, editors, *The confluence of vision and control*, pages 66–78. Springer-Verlag, 1998.
3. F. Chaumette and E. Marchand. A redundancy–based iterative approach for avoiding joint limints: Application to visual servoing. In *IEEE Transaction on Robotics and Automation*, volume 17(5), Oct. 2001.
4. G. Chesi, A. Garulli, A. Vicino, and R. Cipolla. Estimating the fundamental matrix via constrained least squares: a convex approach. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, volume 24(3), Mar 2002.

5. G. Chesi, E. Malis, and R. Cipolla. Automatic segmentation and matching of planar contours for visual servoing. In *Proc. of IEEE Int. Conf. Rob. Autom.*, San Francisco, CA, 2000.
6. G. Chesi, J. Piazzi, D. Prattichizzo, and A. Vicino. Epipole–based visual servoing using profiles. In *Proc. IFAC'02 World Congress*, Barcellona, Spain, July 2002.
7. G. Chesi, D. Prattichizzo, and A. Vicino. A visual servoing algorithm based on epipolar geometry. In *Proc. IEEE Int. Conf. on Robotics and Automation*, May 2001.
8. G. Chesi, D. Prattichizzo, A. Vicino, and K. Hashimoto. Keeping features in the camera's field of view: a visual servoing strategy. In *International Symposium on Mathematical Theory of Networks and Systems*, pages 2321–2326, Notre Dame, IN, August 12-16 2002.
9. R. Cipolla and P.J. Giblin. *Visual Motion of Curves and Surfaces.* Cambridge University Press, 2000.
10. P. Corke and S. Hutchinson. A new partitioned approach to image-based visual servo control. In *IEEE Transaction on Robotics and Automation*, volume 17, Aug. 2001.
11. B. Espiau, F. Chaumette, and P.Rives. A new approach to visual servoing in robotics. In *IEEE Trans. on Robotics and Automation*, volume 8(3), pages 313–326, June 1992.
12. F. Espiau, E. Malis, and P. Rives. Robust features tracking for robotic applications: towards 2 1/2d visual servoing with natural images. In *Proc. of the 2002 IEEE International Conference on Robotics and Automation*, Washington, DC, May 2002.
13. O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint.* MIT Press, Cambridge, 1993.
14. G. D. Hager. A modular system for positioning using feedback from stereo vision. In *IEEE Trans. on Robotics and Automation*, volume 13, pages 582–595, 1997.
15. R. Hartley. In defence of the 8-point algorithm. In *Proc. of IEEE Int. Conference on Computer Vision*, Cambridge MA,USA, June 1995.
16. R. Hartley and A. Zisserman. *Multiple view in computer vision.* Cambridge University Press, 2000.
17. K. Hashimoto and T. Noritsugu. Performance and sensitivity in visual servoing. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 2321–2326, 1998.
18. J. Hill and W. T. Park. Real time control of a robot with a mobile camera. *International Symposium on Industrial Robots*, pages 233–246, 1979.
19. S. Hutchinson, G.D. Hager, and P.I Corke. Tutorial on visual servo control. *IEEE Trans. Rob. Autom.*, 1996.
20. F. Janabi-Sharifi and W. J. Wilson. Image features for visual servoing. 13(6):890–903, Dec. 1997.
21. H.C. Longuet-Higgins. A computer algorithm for recostructing a scene from two projections. *Nature*, 293:133–135, 1981.
22. Q.-T. Luong and O.D. Faugeras. The fundamental matrix: theory, algorithmsm and stability analysis. *Int. Journal of Computer Vision*, 17(1):43–76, 1996.
23. E. Malis, F. Chaumette, and S.Boudet. 2 1/2d visual servoing. In *IEEE Trans. on Robotics and Automation*, volume 15, pages 238–250, Apr. 1999.
24. J. J. Moré, B. S. Garbow, and K. E. Hillstrom. User guide for minpack-1. In *Argonne National Laboratory Report ANL-80-74, Argonne, Ill*, 1980.

25. O. Faugeras Q. T. Luong, R. Deriche and T. Papadopoulo. On determining the fundamental matrix: analysis of different methods and experimental results. In *INRIA*, volume 1894, Apr. 1994.
26. E. Rivlin R. Basri and I. Shimshoni. Visual homing: Surfing on the epipoles. 33(2):705–710, 1999.
27. R. Vidal, S. Soatto, Y. Ma, and S. Sastry. Segmentation of dynamic scenes from the multibody fundamental matrix. In *ECCV Workshop on Vision and Modelling of Dynamic Scenes*, 2002.
28. Z. Zhang. Determining the epipolar geometry and its uncertainty: a review. In *Int. Journal of Computer Vision*, volume 27, March 1998.

Toward Geometric Visual Servoing

Noah John Cowan[1] and Dong Eui Chang[2]

[1] University of California, Berkeley, CA 94720, USA
[2] University of California, Santa Barbara, CA 93106, USA

Abstract. This paper presents a global diffeomorphism from a suitably defined "visible set" of rigid body configurations — a subset of SE(3) — to an image-space. The mapping between the visible set and the image-space is given by the projection of a set of features of a specially designed visual target. The target is a sphere marked with a feature point and a vector tangent to the sphere at the feature point. We show how the construction of a diffeomorphism to image-space should pave the way for developing global, dynamic visual servoing systems using Navigation Functions.

1 Introduction

There is a large and growing body of algorithms for "visual servoing" (VS) — motion control using visual feedback. Traditionally, VS algorithms generate motor reference velocities to register a camera's current view of a scene with a previously stored view (for a tutorial, see [7]).

We seek to move VS toward a systematic theory by characterizing the geometry of "visible" configurations of a visual target relative to a camera. In particular, for a specific target geometry we present a *diffeomorphism* — a smooth and smoothly invertible transformation — from an appropriately defined visible set of configurations to an image space. We believe this transformation will enable the construction of purely image-based, global dynamic VS algorithms.

1.1 Background

A significant challenge involves representing rigid motions in terms of visually measured quantities. Ideally, such a representation should enable effective encoding of

- **Configuration and State,** e.g. position and velocity or position and momentum for Lagrangian or Hamiltonian systems.
- **Tasks and goals,** e.g. trajectories in the state space or points in the configuration space.
- **Obstacles,** e.g. the edge of the field-of-view (FOV) for VS systems.
- **Uncertainty,** e.g. sensor and actuator noise or parametric error.

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 233–247, 2003

There are several candidate representations of image-based rigid motion to consider from the literature. The classical approach to "2D VS" employs the projection, treated as a vector in $\mathbb{R}^n$, of an arbitrary set of feature points [7]. The redundancy of using extra feature points seems to confer robustness to measurement noise in any one of the feature measurements. However, the movement of features is constrained by the underlying rigid motion, rendering image-based control and motion planning in image space challenging for large deviations from a goal. Notwithstanding those challenges, Corke and Hutchinson [2] created a 2D kinematic algorithm for 6DOF VS that seems (empirically) to have a very large basin of attraction while keeping features in the FOV. Their algorithm employs a clever choice of image features which helped motivate the choice of features used in this paper.

A more recent approach uses partial pose reconstruction: given a sufficient number of feature points, the relative pose, up to a scale in translation, between two views may be determined without exploiting a geometrical model of the points. Using this technique, researchers developed six DOF VS algorithms robust to calibration uncertainty [12,15]. It is worth noting that the methods used require sufficient point correspondences between views to fully reconstruct a geometric model of the visual target [10]. Application of this method to contexts besides full six DOF VS remains a challenge.

Alternatively, one may recover the complete pose of a camera with respect to a target by exploiting a model of the target [11]. Vision-based controllers using full pose reconstruction are often referred to as "3D VS" algorithms. Model based pose reconstruction requires fewer feature points than the model-free approach described above, and has the added advantage of fully recovering feature depth, effectively reducing the camera to a "virtual Cartesian sensor." Representing visibility obstacles, such as the FOV or self-occlusions is less parsimonious, but can be done [5]. Formal results demonstrating parametric robustness of VS systems using this method remain elusive.

Generalized image-based coordinates have proven extremely effective in a few narrow contexts [3,5,16]. Generalized coordinates describe kinematic motion with one variable per mechanical DOF. Lagrange's equations, for example, are usually written using such coordinates. Hence, this approach enables the expression of *dynamical* equations of motion in terms of measured quantities on the image plane. Obstacles such the FOV and self-occlusions often appear as the boundary of a compact manifold in image-space and hence their avoidance may be cast as an instance of dynamical obstacle avoidance [5]. Although quite robust in practice, obtaining formal guarantees of robustness to noise or parametric uncertainty for this framework remains an open problem.

1.2 Contribution

To date, global image-based representations of configuration have been applied only to three DOF systems. This paper builds on previous results in a key way: we present an image-based, geometric representation of six DOF rigid motion. Our development of a global representation of "visible" rigid motions viewed through the projection of a set of features should help pave the way for new global, dynamic VS systems.

Organization. In Section 2, we employ a specific target geometry — a sphere with a few markings — to create a global image-based representation of motion for six DOF VS. Included in our development is a simple, purely image-based representation of the so-called image Jacobian (made possible since, as we show, the image and task spaces are diffeomorphic). In Section 4, we suggest a method for using our diffeomorphism for kinematic or dynamic control, although there is much open work to be done in this endeavor. Finally, we give some concluding remarks in Section 5.

2 Six DOF Diffeomorphism to Image-space

We assume a visual target may be designed to our specifications, so we may explore new image-based representations of rigid motion. In cases in which we have the freedom to design visual targets — for example when designing docking stations for space craft, helicopter landing beacons, or visual targets for a factory setting — this approach may lead to novel target designs that ease the control problem. More generally, it is hoped that the insight drawn from taking this approach may enable us to reinterpret target geometries over which we have have less design freedom.

Consider the problem of moving a rigid target object in six DOF relative to a perspective camera. The rigid target considered is as follows:

(i) **A spherical body.** Consider a spherical body of radius ϱ. As the body moves away from the camera, its projection gets smaller. Roughly speaking, the position and size of the body's image encodes the position of the center of the body relative to the camera.
(ii) **A single point on the body.** Adding a visible point to the body breaks the visual symmetry, allowing us to resolving two rotational DOF's from the location of the feature point on the image.
(iii) **A unit vector tangent to the body.** The final degree of freedom is resolved by considering the orientation on the image of a projected vector attached to our feature point on the body.

Zhang and Ostrowski [16,17] developed the idea of projecting a spherical body to an image plane for VS of a blimp relative to a large ball. Using a "flat" image plane, the resulting image is an ellipse, which they approximate as a circle by assuming that a slice of the spherical body parallel to the image

Table 1. List of symbols.

Symbol	Description
$o, p, b, \ldots \in \mathbb{E}^3$	Euclidean points (Roman)
$\boldsymbol{v}, \boldsymbol{e}, \ldots \in \mathbb{R}^3$	vectors (boldfaced)
$\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots \in \mathbb{R}^3$	standard basis
$\pi : \{\mathbb{E}^3 - o_c\} \to \mathrm{S}^2$	image projection model – spherical panoramic camera
$\mathcal{F} = \{o, \boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}\}$	rigid coordinate frame, $o \in \mathbb{E}^3$ and $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k} \in \mathbb{R}^3$
$\mathcal{F}_c, \mathcal{F}_b$	camera frame and body frame
$p^b, \boldsymbol{v}^b$	point, p, and vector, $\boldsymbol{v}$, with respect to $\mathcal{F}_b$
$H \in \mathrm{SE}(3)$	rigid transformation of $\mathcal{F}_b$, relative to $\mathcal{F}_b$
$R \in \mathrm{SO}(3)$	rotation effected by H, columns $R = \begin{bmatrix}\boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{r}_3\end{bmatrix}$
$\boldsymbol{d} \in \mathbb{R}^3$	translation effected by H
$p^c = Hp^b, \boldsymbol{v}^c = R\boldsymbol{v}^b$	point, p, and vector, $\boldsymbol{v}$, with respect to $\mathcal{F}_c$
$\nu : \mathrm{SE}(3) \to \mathbb{R}$	measure of feature visibility, (5)
$\mathcal{V} \subset \mathrm{SE}(3)$	set of "visible" configurations, $H \in \mathcal{V} \iff \nu(H) > 0$
$\lambda \in (0, 1)$	radius on image sphere of body, (3)
$\boldsymbol{s} \in \mathrm{S}^2$	unit vector pointing toward body centroid, (3)
$Q \in \mathrm{SO}(3)$	image-based rotation, columns $Q = \begin{bmatrix}\boldsymbol{q}_1 & \boldsymbol{q}_2 & \boldsymbol{q}_3\end{bmatrix}$, (7)
$(Q, \lambda, \boldsymbol{s}) = c(H)$	camera map, (8)
$\mathcal{I}$	image feature space, $\mathcal{I} \subset \mathrm{SO}(3) \times (0, 1) \times \mathrm{S}^2$, (9)

plane is projected. The present paper builds on that work, employing a more 'exact' diffeomorphism to the image-space, as well as incorporating additional markings on the body whose projection encodes rotational information.

2.1 Notation and Definitions

At the risk of burdening the reader with formalism, we present the following definitions to enable a precise geometric description of the domain and range of a camera viewing rigid motions.

An affine point $p \in \mathbb{A}^3$ has homogeneous coordinates $p = \begin{bmatrix}p_1 & p_2 & p_3 & 1\end{bmatrix}^T$ with respect to some rigid frame. Note that $T\mathbb{A}^3 = \mathbb{A}^3 \times \mathbb{R}^3$, and that $\mathbb{R}^3$ acts on points to translate them in the usual way, so that if $\boldsymbol{v} = \begin{bmatrix}v_1 & v_2 & v_3\end{bmatrix}^T \in \mathbb{R}^3$ and $p \in \mathbb{A}^3$, then $p + \boldsymbol{v} = \begin{bmatrix}p_1 + v_1 & p_2 + v_2 & p_3 + v_3 & 1\end{bmatrix}^T \in \mathbb{A}^3$. Two points cannot be "added" together, but if $p, b \in \mathbb{A}^3$ then $\boldsymbol{v} = p - b \in \mathbb{R}^3$ is the vector such that $p = b + \boldsymbol{v}$. Adding the usual metric structure to affine space $\mathbb{A}^3$ yields Euclidean space $\mathbb{E}^3$ where the distance between two points is given by the two norm of their difference, $\|p - b\|$ (a measure independent of the choice of rigid frame).

A rigid frame, $\mathcal{F}$, is defined by its origin, $o \in \mathbb{E}^3$, and three mutually orthogonal unit vectors, $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k} \in \mathbb{R}^3$, that create a right-handed frame. Consider a full perspective ("pinhole") camera with frame $\mathcal{F}_c$ such that o_c is located at the pinhole (or optical center), with $\boldsymbol{k}_c$ aligned with the optical axis. The

pinhole camera projects points in the open half space "in front" of the camera to an image-plane pair, given by via the map, $\pi^+\{\mathbb{E}^3 : (p-o_c)\cdot \boldsymbol{k}_c > 0\} \to \mathbb{R}^2$, expressed in camera frame coordinates

$$\pi^+(p) = \frac{f}{p_3}\begin{bmatrix} p_1 \\ p_2 \end{bmatrix} \quad p_3 > 0\,, \tag{1}$$

where f is the camera focal length. The camera observes features of a rigid body, affixed with rigid frame $\mathcal{F}_b$. Let

$$H = \begin{bmatrix} R & \boldsymbol{d} \\ 0^T & 1 \end{bmatrix} \in \mathrm{SE}(3), \quad \text{where} \quad R = \begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_1 & \boldsymbol{r}_3 \end{bmatrix} \in \mathrm{SO}(3),\, \boldsymbol{d} \in \mathbb{R}^3,$$

denote the rigid transformation of $\mathcal{F}_b$ relative to $\mathcal{F}_c$. A point expressed with respect to the body-frame as p^b, appears as $p^c = Hp^b$ with respect to the camera frame. Similarly, if $\boldsymbol{v}^b$ is a vector in the body frame, then $\boldsymbol{v}^c = R\boldsymbol{v}^b$ is the same vector with respect to the camera frame.

Hamel *et. al.* [6] remap the image plane to a sphere to recover some symmetry that is "broken" by a flat image plane. This approach has also been used in the structure from motion (SFM) literature [1]. Let $\boldsymbol{p} = (p - o_c)$ and note that the unit vector, $\boldsymbol{p}/\|\boldsymbol{p}\|$ may be recovered from the image-plane pair in (1) since

$$\frac{\boldsymbol{p}}{\|\boldsymbol{p}\|} = \begin{bmatrix} \pi^+(p) \\ f \end{bmatrix} \Big/ \left\| \begin{bmatrix} \pi^+(p) \\ f \end{bmatrix} \right\|, \quad p_3 > 0$$

with respect to the camera frame. Of course, this assumes that we know the parameter f (or, more generally, all so-called "intrinsic" camera parameters, omitted to simplify the presentation). Motivated by this observation, we consider for convenience a "panoramic" spherical camera

$$\pi \,:\, (\mathbb{E}^3 - \{o_c\}) \to \mathrm{S}^2 \tag{2a}$$

$$:\, p \mapsto \frac{\boldsymbol{p}}{\|\boldsymbol{p}\|} \quad \text{where} \quad \boldsymbol{p} = (p - o_c)\,. \tag{2b}$$

For the purposes of this paper, $\mathrm{S}^2 = \{\boldsymbol{v} \in \mathbb{R}^3 : \boldsymbol{v}\cdot\boldsymbol{v} = 1\} \subset \mathbb{R}^3$. For the camera map, S^2 corresponds to the unit tangent space of $\mathbb{E}^3$ at o_c, namely "the set of unit vectors originating from the camera origin." To keep features within a finite FOV, one may introduce an appropriate image-space "obstacle" into the controller design (see Section 4).

2.2 Image-based Translation

Attach the body frame at the center of the sphere, so that the location of the body relative to the camera origin is given by $o_b - o_c = \boldsymbol{d}$. If $\|\boldsymbol{d}\| > \varrho$ — *i.e.* the body remains bounded away from the camera origin — then the surface

of the body double-covers a topological disc on S^2 via the map π. The edge of the disc, a planar slice of the image-sphere, is a perfect circle of radius

$$\lambda = \frac{\varrho}{\|\boldsymbol{d}\|}, \quad \varrho < \|\boldsymbol{d}\| < \infty.$$

(The circle radius, λ, appears dimensionless because the image-sphere was normalized to unit radius). The center of the circle on the image-sphere is in the direction of

$$\boldsymbol{s} = \frac{\boldsymbol{d}}{\|\boldsymbol{d}\|}$$

and is readily measurable from the projection of the body.

Let $\mathcal{B} := \{\boldsymbol{d} \in \mathbb{R}^3 : \|\boldsymbol{d}\| > \varrho\}$ denote the translations of the body origin that keep it a body radius away from the camera. We now have a diffeomorphism — a smooth and smoothly invertible function — from locations of the body to image measurements, $c_1 : \mathcal{B} \to (0,1) \times \mathrm{S}^2$, given by

$$\boxed{c_1 : \boldsymbol{d} \mapsto (\lambda, \boldsymbol{s})\,.} \tag{3}$$

The inverse of c_1 is given simply by

$$c_1^{-1}(\lambda, \boldsymbol{s}) = \frac{\varrho}{\lambda}\boldsymbol{s}\,. \tag{4}$$

2.3 Image-based Rotation

To break the rotational symmetry of our spherical rigid body, attach a visible feature point, b, to its surface, and a unit vector $\boldsymbol{a}$ tangent to the body at that point. For convenience, align the body frame so that origin coincides with the center of the body, and the unit vector $(b - o_b)/\varrho$ lies along the negative $\boldsymbol{k}_b$ axis. Hence, in the body frame $b^b = \begin{bmatrix}0, 0, -\varrho, 1\end{bmatrix}^T$.

As we will show, the projection of b to the image-sphere, $\boldsymbol{q}_1 = \pi(b)$, encodes two rotational degrees of freedom. We encode the final degree-of-freedom by projecting a unit vector or "arrow", $\boldsymbol{a}$, tangent to the body at the point b. In practice, the vector $\boldsymbol{a}$ may be approximated by two distinguishable points on the surface of the sphere. Again for convenience we assume the vectors body-fixed representation is simply $\boldsymbol{a}^b = \boldsymbol{e}_2$. Let $\boldsymbol{b} = b - o_c$ denote the vector from the camera origin to the body point b. Recalling that the rotation matrix R has columns $(\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3)$, then with respect to the camera frame, we have

$$b^c = \begin{bmatrix}\boldsymbol{b}^c \\ 1\end{bmatrix} = Hb^b, \quad \text{where} \quad \boldsymbol{b}^c = \boldsymbol{d} - \varrho\boldsymbol{r}_3, \quad \text{and} \quad \boldsymbol{a}^c = R\boldsymbol{a}^b = \boldsymbol{r}_2\,.$$

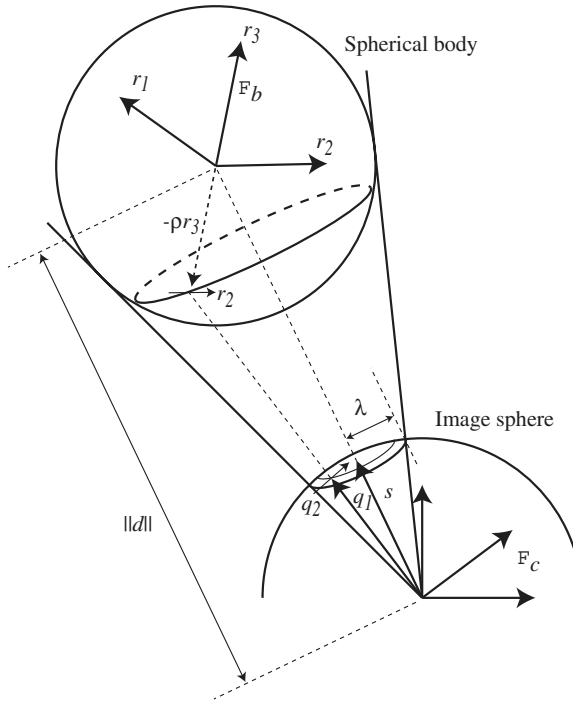


Fig. 1. Projection of a spherical body with a feature point on it to the image-sphere. The image-plane measurement is given by $y = (Q, \lambda, \boldsymbol{s}) = c(H)$.

Note that $(b - o_b) \cdot \boldsymbol{a} = -\varrho\, \boldsymbol{e}_3 \cdot \boldsymbol{e}_2 = 0$.

Some configurations cause the body to occlude the feature point, b. This occurs when $(b - o_c) \cdot (o_b - b)$ becomes negative. Hence, we define a "visibility" function [5], ν, and associated "visible set" of rigid transformations, $\mathcal{V}$ by

$$\nu(H) := (\boldsymbol{d} - \varrho \boldsymbol{r}_3) \cdot \boldsymbol{r}_3 \quad \text{and} \quad \mathcal{V} := \{H \in \mathrm{SE}(3) : \nu(H) > 0\}\,. \tag{5}$$

Note that $\nu(H) > 0 \implies \|\boldsymbol{d}\| > \varrho$, i.e. $\boldsymbol{d} \in \mathcal{B} = \{\boldsymbol{d} \in \mathbb{R}^3 : \|\boldsymbol{d}\| > \varrho\}$.

The projection of $(b, \boldsymbol{a}) \in T\mathbb{E}^3$ to the image sphere is modeled by

$$T\pi : (b, \boldsymbol{a}) \mapsto (\pi(b), T_b\pi \cdot \boldsymbol{a}) \in T\mathrm{S}^2\,.$$

We are not concerned with the length of the projection of $\boldsymbol{a}$, only the direction. Hence, consider the unit tangent map $T^1\pi$ represented in the camera frame by

$$T^1\pi : (b, \boldsymbol{a}) \mapsto (\boldsymbol{q}_1, \boldsymbol{q}_2) \quad \text{where} \tag{6}$$

$$\boldsymbol{q}_1 = \frac{\boldsymbol{b}^c}{\|\boldsymbol{b}^c\|} = \frac{\boldsymbol{d} - \varrho \boldsymbol{r}_3}{\|\boldsymbol{d} - \varrho \boldsymbol{r}_3\|} \quad \text{and} \quad \boldsymbol{q}_2 = \frac{\Gamma_{\boldsymbol{q}_1} \boldsymbol{a}^c}{\|\Gamma_{\boldsymbol{q}_1} \boldsymbol{a}^c\|} = \frac{\Gamma_{\boldsymbol{q}_1} \boldsymbol{r}_2}{\|\Gamma_{\boldsymbol{q}_1} \boldsymbol{r}_2\|}$$

$$\text{where} \quad \Gamma_{\boldsymbol{q}_1} := (I - \boldsymbol{q}_1 \boldsymbol{q}_1{}^T)\,.$$

Geometrically, $\boldsymbol{q}_2$ is a unit vector tangent to the image-sphere at the point $\boldsymbol{q}_1$. The unit vectors $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ are mutually orthogonal. Consider the plane containing the camera origin o_c, the point b, and the vector $\boldsymbol{a}$. The unit vector

$$\boldsymbol{q}_3 = \boldsymbol{q}_1 \times \boldsymbol{q}_2$$

is normal to that plane. Thus, we define a function $c_2 : \mathcal{V} \to \mathrm{SO}(3)$

$$\boxed{c_2 : H \mapsto \begin{bmatrix}\boldsymbol{q}_1 & \boldsymbol{q}_2 & \boldsymbol{q}_3\end{bmatrix} = Q,} \tag{7}$$

identifying $T^1\mathrm{S}^2$ with $\mathrm{SO}(3)$.

2.4 Diffeomorphism to Image-space

Claim. The function $c : \mathcal{V} \to \mathcal{I}$, defined by

$$c(H) := (c_2(H), c_1(\boldsymbol{d})) \text{ , where} \tag{8}$$

$$\mathcal{I} = \left\{(Q, \lambda, \boldsymbol{s}) \in \mathrm{SO}(3) \times (0,1) \times \mathrm{S}^2 : \boldsymbol{q}_1 \cdot \boldsymbol{s} > \sqrt{1-\lambda^2}\right\} \tag{9}$$

$$\text{and } Q = \begin{bmatrix}\boldsymbol{q}_1 & \boldsymbol{q}_2 & \boldsymbol{q}_3\end{bmatrix}$$

is a diffeomorphism, i.e. $\mathcal{V} \simeq \mathcal{I}$. □

The proof is given in the appendix

3 Image Jacobian

To be of practical application to VS we present a representation of the tangent map $Tc : T\mathcal{V} \to T\mathcal{I}$, its inverse Tc^{-1}, and the cotangent map $T^*c : T^*\mathcal{I} \to T^*\mathcal{V}$, with the following commutative diagram in mind:

$$\begin{array}{ccc}
T\mathcal{V} & \underset{Tc^{-1}}{\overset{Tc}{\rightleftarrows}} & T\mathcal{I} \\
\downarrow & & \downarrow \\
\mathcal{V} & \xrightarrow{\quad c \quad} & \mathcal{I} \\
\uparrow & & \uparrow \\
T^*\mathcal{V} & \underset{T^*c}{\overset{T^*c^{-1}}{\rightleftarrows}} & T^*\mathcal{I}
\end{array}$$

We make the following identification of the tangent space $T\mathrm{SE}(3)$ of the Lie group $\mathrm{SE}(3)$:[1]

$$T\mathrm{SE}(3) \simeq \mathrm{SE}(3)\times\mathfrak{se}(3) \simeq \mathrm{SE}(3)\times(\mathbb{R}^3 \circledS \mathbb{R}^3), \tag{10}$$

where $\mathfrak{se}(3)$ is the Lie algebra of $\mathrm{SE}(3)$. The identification occurs via "right translation," i.e.

$$(H, \dot{H}) \mapsto (H, \dot{H}H^{-1}) \mapsto (H, (\boldsymbol{\omega}, \boldsymbol{v})) \tag{11}$$

where

$$H = \begin{bmatrix} R & \boldsymbol{d} \\ 0^T & 1 \end{bmatrix}, \quad \dot{H} = \begin{bmatrix} \dot{R} & \dot{\boldsymbol{d}} \\ 0^T & 0 \end{bmatrix} \quad \text{and} \quad \begin{cases} \boldsymbol{\omega} = \left(\dot{R}R^{-1}\right)^\vee \\ \boldsymbol{v} = -\dot{R}R^{-1}\boldsymbol{d} + \dot{\boldsymbol{d}} \end{cases}$$

and the isomorphism $\mathbb{R}^3 \simeq \mathfrak{so}(3)$ is defined by

$$\widehat{\ }: \begin{bmatrix} \boldsymbol{\omega}_1 \\ \boldsymbol{\omega}_2 \\ \boldsymbol{\omega}_3 \end{bmatrix} \mapsto \begin{bmatrix} 0 & -\boldsymbol{\omega}_3 & \boldsymbol{\omega}_2 \\ \boldsymbol{\omega}_3 & 0 & -\boldsymbol{\omega}_1 \\ -\boldsymbol{\omega}_2 & \boldsymbol{\omega}_1 & 0 \end{bmatrix}, \quad {}^\vee: \begin{bmatrix} 0 & -\boldsymbol{\omega}_3 & \boldsymbol{\omega}_2 \\ \boldsymbol{\omega}_3 & 0 & -\boldsymbol{\omega}_1 \\ -\boldsymbol{\omega}_2 & \boldsymbol{\omega}_1 & 0 \end{bmatrix} \mapsto \begin{bmatrix} \boldsymbol{\omega}_1 \\ \boldsymbol{\omega}_2 \\ \boldsymbol{\omega}_3 \end{bmatrix},$$

where $\mathfrak{so}(3)$ is the Lie algebra of $\mathrm{SO}(3)$. More detail can be found in, for example [13].

Similarly, for each $y = (Q, \lambda, \boldsymbol{s}) = \mathcal{I} \subset \mathrm{SO}(3) \times (0,1) \times \mathrm{S}^2$, we have the following identification

$$T_y\mathcal{I} = T_Q\mathrm{SO}(3)\times T_\lambda(0,1)\times T_{\boldsymbol{s}}\mathrm{S}^2 \simeq \mathbb{R}^3\times\mathbb{R}\times T_{\boldsymbol{s}}\mathrm{S}^2 \tag{12}$$

where we identify $T_Q\mathrm{SO}(3)$ with $\mathfrak{so}(3) \simeq \mathbb{R}^3$, again via right translation

$$(Q, \dot{Q}) \mapsto (Q, \boldsymbol{\xi}) \quad \text{where} \quad \boldsymbol{\xi} = \left(\dot{Q}Q^{-1}\right)^\vee. \tag{13}$$

Hence, to compute T_Hc we find the mapping relating the tangent space identifications made above in (10) and (12), namely

$$(H, (\boldsymbol{\omega}, \boldsymbol{v})) \mapsto \left(y, (\boldsymbol{\xi}, \dot{\lambda}, \dot{\boldsymbol{s}})\right)$$

$$\text{where} \quad y = (Q, \lambda, \boldsymbol{s}) = c(H), \quad \begin{bmatrix} \boldsymbol{\xi} \\ \dot{\lambda} \\ \dot{\boldsymbol{s}} \end{bmatrix} = C(y) \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v} \end{bmatrix}$$

and

$$C(y) := T_Hc|_{H=c^{-1}(\boldsymbol{y})} = \begin{bmatrix} I_{3\times3} & \frac{1}{\beta}(\delta\boldsymbol{q}_1\boldsymbol{q}_3^T - \boldsymbol{q}_2\boldsymbol{q}_3^T + \boldsymbol{q}_3\boldsymbol{q}_2^T) \\ 0_{1\times3} & -\frac{\lambda^2}{\varrho}\boldsymbol{s}^T \\ -\hat{\boldsymbol{s}} & \frac{\lambda}{\varrho}(I_{3\times3} - \boldsymbol{s}\boldsymbol{s}^T) \end{bmatrix}, \tag{14}$$

[1] The Lie algebra $\mathbb{R}^3 \circledS \mathbb{R}^3$ is $\mathbb{R}^3 \times \mathbb{R}^3$ with the Lie bracket structure found in [13].

where

$$\delta = \frac{\boldsymbol{s}\cdot\boldsymbol{q}_2}{\sqrt{\lambda^2-\sin^2\phi}}\,, \quad \beta = \frac{\varrho}{\lambda}\left(\cos\phi - \sqrt{\lambda^2-\sin^2\phi}\right),$$
$$\cos\phi = \boldsymbol{s}\cdot\boldsymbol{q}_1 \quad \text{and} \quad \sin\phi = \sqrt{1-(\boldsymbol{s}\cdot\boldsymbol{q}_1)^2}\,.$$

The construction of C is straight forward. The details are given in [4].

To compute Tc^{-1}, and T^*c is now straight forward. Using the above representations, we have

$$T_y c^{-1} = \left(C(y)^T C(y)\right)^{-1} C(y)^T \quad \text{and} \quad T_y^* c = C(y)^T\,. \tag{15}$$

Note that the expression for $T_y c^{-1}$ is *not* a pseudo-inverse. The possible confusion arises since the six dimensional tangent space $T_y\mathcal{I}$ is locally embedded in $\mathbb{R}^7$. It should be noted that in many image-based visual servoing strategies employ the pseudo-inverse of the image Jacobian since the image feature points are treated as though moving freely in $\mathbb{R}^n$.

4 Controller

For the present work, we consider the case of so-called "eye-in-hand" VS, wherein the camera moves relative to the body which serves as an inertial reference frame. Let $(\boldsymbol{\Omega}, \boldsymbol{V})$ denote the angular and linear velocities, respectively, of the camera relative to the fixed, inertial body frame. Let $G = H^{-1}$ denote the transformation of the camera frame, $\mathcal{F}_c$, relative to the inertial body frame, $\mathcal{F}_b$. Note that

$$\begin{bmatrix} \widehat{\boldsymbol{\Omega}} & \boldsymbol{V} \\ 0^T & 0 \end{bmatrix} = G^{-1}\dot{G} = -\dot{H}H^{-1} = -\begin{bmatrix} \widehat{\boldsymbol{\omega}} & \boldsymbol{v} \\ 0^T & 0 \end{bmatrix}, \tag{16}$$

effectively mapping the identification of $T\mathrm{SE}(3)$ given by the right translation of $\dot{H}$ in (10) and (11) to the left translation of $\dot{G} = \frac{d}{dt}\big(H^{-1}\big)$. Note that this relationship clears up, once and for all, the kinematic distinction between "eye-in-hand" servoing and the so-called "fixed-camera" configuration, wherein the camera is fixed and the body is moving.

For simplicity, we posit a fully actuated purely kinematic plant model

$$\dot{G} = G\begin{bmatrix} \widehat{\boldsymbol{\Omega}} & \boldsymbol{V} \\ 0^T & 0 \end{bmatrix} \tag{17}$$

where we treat $(\boldsymbol{\Omega}, \boldsymbol{V}) \in \mathbb{R}^3 \circledS \mathbb{R}^3$ as control inputs. (We generalize this to a dynamical free rigid body in [4]). One possible control strategy involves planning a path $y_d(t) \in \mathcal{I}$ that moves from the initial configuration to the goal state and following the path via

$$\begin{bmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{bmatrix} = -T_y c^{-1}\begin{bmatrix} \boldsymbol{\xi}_d \\ \dot{\lambda}_d \\ \dot{\boldsymbol{s}}_{\boldsymbol{d}} \end{bmatrix} \tag{18}$$

where $\left[\boldsymbol{\xi}_d\,,\,\dot{\lambda}_d\,,\,\dot{\boldsymbol{s}}_{\boldsymbol{d}}\right]^T$ is the desired velocity $\dot{y}_d$, expressed using the tangent space identification in (12). The minus sign in the above expression arises due to the identification made above in (16).

4.1 Visual Servoing via Navigation Functions

The diffeomorphism c, the visible set $\mathcal{V}$, and its relatively simple image $\mathcal{I}$, provide tremendous leverage into the VS problem. Given a desired configuration $G^* = (H^*)^{-1}$, measured through its image $y^* = (Q^*, \lambda^*, \boldsymbol{s}^*) = c(H^*)$, there are many possible image-based control strategies we can employ to achieve our objective of driving $G \to G^*$.

An open-loop strategy, such as the one above in (18), may be undesirable. However, the generation of $\dot{y}_d$ can also be conceived as a feedback law, for example by using the method of Navigation Functions (NF's) [8,9,14]. A substantial benefit of using NF's is that they allow us to "lift" our kinematic controller to second order settings with little additional effort, while maintaining similar convergence guarantees (as we do for this problem in [4]). Moreover, these methods have already proven practicable for dynamic VS [5].

Let $\mathcal{D} \subset \mathcal{I}$ be compact "safe" domain. If we carefully design an artificial potential function $\varphi : \mathcal{D} \to [0, 1]$, then by letting

$$\dot{y}_d = -\nabla\varphi \tag{19}$$

the control law given by (18) drives G so that y converges to y^*, except for a set of measure zero. The following definition, adapted from [8], gives a set of conditions that guarantee essentially global convergence of the above controller (18), with $\dot{y}_d$ given in (19).

Definition 1. Let $\mathcal{D}$ be a smooth compact connected manifold with boundary, and $y^* \in \overset{\circ}{\mathcal{D}}$ be a point in its interior. A Morse function, $\varphi \in C^2[\mathcal{D}, [0, 1]]$ is called an *Navigation Function* if

(i) φ takes its unique minimum at $\varphi(y^*) = 0$;
(ii) φ achieves its maximum of unity uniformly on the boundary, i.e. $\partial\mathcal{D} = \varphi^{-1}(1)$.

For any function satisfying the above definition, the controller given by (18) will ensure convergence $y \xrightarrow{t\to\infty} y^*$ from all initial conditions in $\mathcal{D}$. For more information, see [8].

4.2 Computing a Safe Domain and Navigation Function

The next step is to compute a compact domain $\mathcal{D} \subset \mathcal{I}$ that is "safe" with respect to the FOV of our camera system in the sense that if $G^{-1} = H \in c^{-1}(\mathcal{D})$ then all the necessary features are visible. To illustrate, we treat the

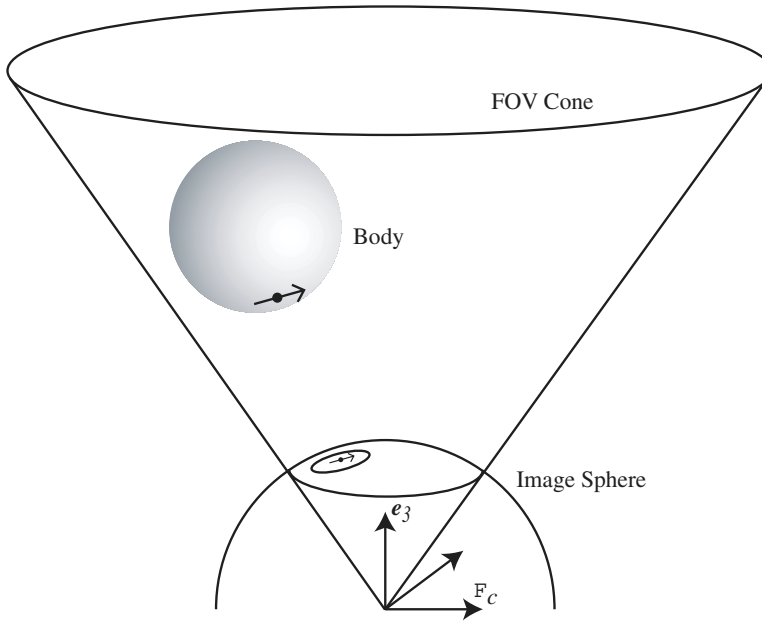


Fig. 2. A simple model of the FOV cone.

FOV as a cone originating at the camera origin, with center along $\boldsymbol{e}_3$, as shown in Figure 2. This cone reduces to a constraint on $\boldsymbol{s}$ and λ, namely

$$f(y) := \lambda \boldsymbol{s} \cdot \boldsymbol{e}_3 - \sqrt{(1-\lambda^2)(1-(\boldsymbol{s}\cdot\boldsymbol{e}_3)^2)} \geq \cos\theta$$

where θ is the angle from $\boldsymbol{e}_3$ to the edge of the FOV cone. Additionally, we constrain $\lambda \in [\lambda_{\min}, \lambda_{\max}] \subset (0,1)$ where the parameters $\lambda_{\min}$ and $\lambda_{\max}$ effectively keep the camera from moving too far from or too close to the camera body, respectively. Finally, we keep $\boldsymbol{q}_1$ from being too close to the edge of the projected circle, namely $\boldsymbol{q}_1 \cdot \boldsymbol{s} + \epsilon \geq \sqrt{1-\lambda^2}$. Putting these constraints together yields the compact manifold

$$\begin{aligned} \mathcal{D} = \big\{ y = (Q, \lambda, \boldsymbol{s}) \in \mathrm{SO}(3) \times [\lambda_{\min}, \lambda_{\max}] \times \mathrm{S}^2 : \\ f(y) \geq \cos\theta, \lambda_{\min} \leq \lambda \leq \lambda_{\max} \big\} \subset \mathcal{I}\,, \\ \text{where} \quad \theta \in (0, \pi/2)\,, \quad 0 < \lambda_{\min} < \lambda_{\max} < 1\,. \end{aligned} \tag{20}$$

Clearly $\mathcal{D} \subset \mathcal{I}$. Given this domain, one must construct an NF on $\mathcal{D}$. The construction of φ represents work in progress, however, we conjecture that given the relatively simple geometry of $\mathcal{D}$, that constructing a suitable NF should be straight forward. In fact, we believe (but have not yet formally shown) that $\mathcal{D} \simeq [0,1]^5 \times \mathrm{S}^1$ which is the same topology for which an NF has already been constructed for VS by the first author and colleagues [5].

5 Conclusion

In this paper, we presented a global diffeomorphism from a large subset of configurations in SE(3) — those that are "visible" — to an appropriately defined image space. Such constructions provide tremendous leverage because they shed light on the *geometry* of occlusion free servoing as well as provide a clear pathway to construct *global dynamical* visual servoing systems by using, for example, Navigation Functions.

A global, sensor-based representation of the configuration space leaves many open doors. For example, the control of underactuated and kinematically nonholonomic systems becomes possible in sensor space. Now that we now know it is possible to globally represent rigid motion using image coordinates, the next step is to construct a more general class of diffeomorphisms to the image plane that does not require designing special visual targets. We believe that with proper insight, the projection of a collection of rigidly connected feature points may be interpreted geometrically, again enabling a global representation of visible configurations. For example, perhaps depth can be described in terms of "moments", as suggested by Hamel and Mahoney [6], and orientation can be described in terms of the projection of two or three feature points.

Acknowledgments.
The first author was supported in part by DARPA/ONR under grants N00014-98-1-0747 and N66001-00-C8026, and NSF under grant ECS-9873474.

Proof That $c : \mathcal{V} \to \mathcal{I}$ is a Diffeomorphism

The proof proceeds in four parts. First, we show that c is smooth on $\mathcal{V}$. Next we show that $c(\mathcal{V}) \subset \mathcal{I}$. Third, we show that c is bijective by explicitly computing its inverse, c^{-1}, on $\mathcal{I}$. Finally, we show that c^{-1} is smooth on $\mathcal{I}$.

The function c is smooth. The function c is composed of smooth functions away from the set where the arguments of $\|\cdot\|^{-1}$ become zero. But those arguments are nonzero on $\mathcal{V}$. In particular:

(i) Equation (3) depends on $\|\boldsymbol{d}\|^{-1}$. However, $H \in \mathcal{V}$ implies $\|\boldsymbol{d}\| > \varrho$.
(ii) Equation (6) depends on $\|\boldsymbol{b}\|^{-1} = \|\boldsymbol{d} - \varrho \boldsymbol{r}_3\|^{-1}$. Visibility implies $\|\boldsymbol{d}\| > \varrho$, which in turn implies $\|\boldsymbol{d} - \varrho \boldsymbol{r}_3\| \geq \|\boldsymbol{d}\| - \|\varrho \boldsymbol{r}_3\| = \|\boldsymbol{d}\| - \varrho > 0$.
(iii) Equation (6) depends on $\|\Gamma_{\boldsymbol{q}_1} \boldsymbol{r}_2\|^{-1}$. This blows up iff $\boldsymbol{q}_1 = \pm \boldsymbol{r}_2$, i.e.

$$\boldsymbol{q}_1 = \frac{\boldsymbol{d} - \varrho \boldsymbol{r}_3}{\|\boldsymbol{d} - \varrho \boldsymbol{r}_3\|} = \pm \boldsymbol{r}_2$$

and hence, from (5)

$$\nu(H) = (\boldsymbol{d} - \varrho \boldsymbol{r}_3) \cdot \boldsymbol{r}_3 = \pm \|\boldsymbol{d} - \varrho \boldsymbol{r}_3\| \, \boldsymbol{r}_2 \cdot \boldsymbol{r}_3 = 0\,,$$
$$\implies H \notin \mathcal{V}\,.$$

This contradiction implies that $\|\Gamma_{\boldsymbol{q}_1} R \boldsymbol{a}^b\| > 0$ for $H \in \mathcal{V}$.

Hence, c is smooth on $\mathcal{V}$.

The image of $\mathcal{V}$ indeed is contained in $\mathcal{I}$. To see that $c(\mathcal{V}) \subset \mathcal{I}$, let $(Q, \lambda, \boldsymbol{s}) = c(H)$. By construction of c, we have that $(Q, \lambda, \boldsymbol{s}) \in \mathrm{SO}(3) \times (0,1) \times \mathrm{S}^2$. To show that $\boldsymbol{q}_1 \cdot \boldsymbol{s} > \sqrt{1-\lambda^2}$, note from (6) that

$$\boldsymbol{q}_1 \cdot \boldsymbol{s} = \frac{(\boldsymbol{d} - \varrho \boldsymbol{r}_3) \cdot \boldsymbol{s}}{\|\boldsymbol{d} - \varrho \boldsymbol{r}_3\|} = \frac{\|\boldsymbol{d}\| - \varrho \boldsymbol{s} \cdot \boldsymbol{r}_3}{\sqrt{\varrho^2 - 2\boldsymbol{d} \cdot \boldsymbol{r}_3 + \|\boldsymbol{d}\|^2}} = \frac{1 - \lambda\alpha}{\sqrt{1 - 2\lambda\alpha + \lambda^2}}$$

where $\alpha = \boldsymbol{s} \cdot \boldsymbol{r}_3$. From (5), $\alpha > \lambda$ to ensure $\nu(H) > 0$. Since the right hand side reaches its minimum of $\sqrt{1-\lambda^2}$ for $\alpha = \lambda$, we have that $\boldsymbol{q}_1 \cdot \boldsymbol{s} > \sqrt{1-\lambda^2}$.

The function c is bijective. Consider any $H \in \mathcal{V}$, with rotation $R = \begin{bmatrix}\boldsymbol{r}_1 & \boldsymbol{r}_1 & \boldsymbol{r}_1\end{bmatrix}$ and translation $\boldsymbol{d}$ as usual. Let $c(H) = (Q, \lambda, \boldsymbol{s}) \in \mathcal{I}$, where $Q = \begin{bmatrix}\boldsymbol{q}_1 & \boldsymbol{q}_2 & \boldsymbol{q}_3\end{bmatrix}$. Given $(\lambda, \boldsymbol{s})$, recovering the translation from (4) is trivial, namely $\boldsymbol{d} = c_1^{-1}(\lambda, \boldsymbol{s}) \in \mathcal{B}$. Recovering the rotation from $(Q, \lambda, \boldsymbol{s})$ requires a bit more care, but the result yields the unique inverse on $\mathcal{I}$ (see [4] for details):

$$\begin{aligned}
&c^{-1} : (Q, \lambda, \boldsymbol{s}) \mapsto H = \begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{r}_3 & \boldsymbol{d} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \text{where} \quad \boldsymbol{d} = \frac{\varrho}{\lambda}\boldsymbol{s}\,, \\
&\boldsymbol{r}_3 = \frac{1}{\lambda}\left(\boldsymbol{s} - \beta \boldsymbol{q}_1\right), \quad \beta = \|\boldsymbol{b}\| = \frac{\varrho}{\lambda}\left(\cos\phi - \sqrt{\lambda^2 - \sin^2\phi}\right) \\
&\boldsymbol{r}_2 = \left(\boldsymbol{q}_2 - \frac{\boldsymbol{r}_3 \cdot \boldsymbol{q}_2}{\boldsymbol{r}_3 \cdot \boldsymbol{q}_1}\boldsymbol{q}_1\right) \Bigg/ \left\|\boldsymbol{q}_2 - \frac{\boldsymbol{r}_3 \cdot \boldsymbol{q}_2}{\boldsymbol{r}_3 \cdot \boldsymbol{q}_1}\boldsymbol{q}_1\right\|, \quad \text{and} \quad \boldsymbol{r}_1 = \boldsymbol{r}_2 \times \boldsymbol{r}_3\,.
\end{aligned} \tag{21}$$

The function c^{-1} is smooth. Finally, we need only show that c^{-1} is smooth. But, c^{-1} is composed of smooth functions. There are two caveats:

(i) Equations involving $1/\lambda$. This is fine since $0 < \lambda < 1$.
(ii) Equation for $\boldsymbol{r}_2$. First, note that $\boldsymbol{r}_3 \cdot \boldsymbol{q}_1 = \boldsymbol{r}_3 \cdot \boldsymbol{b}/\|\boldsymbol{b}\| = \nu(H)/\|\boldsymbol{b}\| > 0$. Also, since $\boldsymbol{q}_2$ and $\boldsymbol{q}_1$ are linearly independent, the denominator can never be zero, so this equation is smooth on $\mathcal{I}$.

Hence c^{-1} is smooth, and c is a diffeomorphism $c : \mathcal{V} \approx \mathcal{I}$. □

References

1. Alessandro Chiuso, Roger Brocket, and Stefano Soatto. Optimal structure from motion: local ambiguities and global estimates. *International Journal of Computer Vision*, pages 195–228, September 2000.
2. Peter I. Corke and Seth A. Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Transactions on Robotics and Automation*, 17(4):507–515, 2001.
3. Noah Cowan. Binocular visual servoing with a limited field of view. In *Mathematical Theory of Networks and Systems*, Notre Dame, Indiana, August 2002.
4. Noah J. Cowan and Dong Eui Chang. Toward geometric visual servoing. Technical report, University of California, Berkeley, CA 94720, September 2002.
5. Noah J. Cowan, Joel D. Weingarten, and Daniel E. Koditschek. Visual servoing via navigation functions. *Transactions on Robotics and Automation*, 18(4), August 2002.
6. Tarek Hamel and Robert Mahony. Visual servong of an under-actuated dynamic rigid-body system: An image-based approach. *IEEE Transactions on Robotics and Automation*, 18(2):187–198, April 2002.
7. S. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, October 1996.
8. Daniel E. Koditschek. The application of total energy as a Lyapunov function for mechanical control systems. In *Dynamics and control of multibody systems (Brunswick, ME, 1988)*, pages 131–157. Amer. Math. Soc., Providence, RI, 1989.
9. Daniel E. Koditschek and Elon Rimon. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, 11:412–442, 1990.
10. H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.
11. Chien-Ping Lue, Gregory D. Hager, and Eric Mjolsness. Fast and globally convergent pose estimation from video images. *Transactions on Pattern Analysis and Machine Intelligence*, 22(6):610–622, June 2000.
12. Ezio Malis, Francois Chaumette, and Sylvie Boudet. Theoretical improvements in the stability analysis of a new class of model-free visual servoing methods. *IEEE Transactions on Robotics and Automation*, 18(2):176–186, April 2002.
13. Jerrold E. Marsden and Tudor S. Ratiu. *Introduction to Mechanics and Symmetry*, chapter 14.7. Springer, 2 edition, 1999.
14. Elon Rimon and Daniel E. Koditschek. Exact robot navigation using artificial potential fields. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, Oct 1992.
15. Camillo J. Taylor and James P. Ostrowski. Robust vision-based pose control. In *International Conference on Robotics and Automation*, volume 3, pages 2734–2740, San Francisco, CA, 2000. IEEE.
16. Hong Zhang and James P. Ostrowski. Visual servoing with dynamics: Control of an unmanned blimp. In *International Conf. on Robotics and Automation*, 1999.
17. Hong Zhang and Jim Ostrowski. Visual motion planning for mobile robots. *IEEE Transactions on Robotics and Automation*, 18(2):199–208, April 2002.

Vision-Based Online Trajectory Generation and Its Application to Catching

Akio Namiki and Masatoshi Ishikawa

University of Tokyo, 7-3-1 Bunkyo-ku, Hongo, Tokyo 113-8656, Japan

Abstract. In this paper, a method for vision-based online trajectory generation is proposed. The proposed method is based on a nonlinear mapping of visual information to the desired trajectory, and this nonlinear mapping is defined by learning based on constraints of dynamics and kinematics. This method is applied to a catching task, and a reactive and flexible motion is obtained owing to real-time high-speed visual information. Experimental results on catching a moving object using a high-speed vision chip system are presented.

1 Introduction

Human processing architecture for motor control is regarded as a hierarchical processing system composed of several layers such as reflex, control, trajectory generation, and so on [1]. Each layer has realtime afferent signal inputs from the vision system, and visual information affects the processing result strongly not only in low- level layers such as that of reflex but also in high level layers such as that corresponding to trajectory generation. This means that high-level processing is also processed dynamically by visual information.

In human reaching motion, for example, it is observed that visual information is used for feed-forward trajectory generation in addition to feedback control based on a position error between an end effector and target [3]. As a result, a human can catch a target well, even if it moves during the reaching motion. Such a realtime vision-based processing in a high-level layer produces a flexible action, reactive to changes of the environment.

On the other hand, in most conventional manipulation researches, real-time visual information is used mainly for servo control, and it is not directly used as much for high-level processing such as trajectory generation. Even if visual information affects trajectory generation, it is static or quasi-static, and the processing rate is not high. In most cases a trajectory is given as a time-based function which is not affected directly by visual information. However a vision-based online trajectory generator has more advantages to produce a reactive and flexible motion in the case that it is difficult to predict target motion because of disturbance and uncertainty.

In this paper, a method for vision-based online trajectory generation is proposed and applied to a catching task of a ball. This method is based on a nonlinear mapping of visual information to the desired trajectory, and this nonlinear mapping is defined by learning based on constraints of dynamics

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 249–264, 2003

and kinematics. In section 2 previous related works on manipulation using realtime visual information are presented. In section 3 we propose an algorithm for online vision-based trajectory generator, and in section 4 an experimental result using a high-speed vision system and a manipulator is presented.

2 Related Works

A number of research efforts have focused on the problem of using visual information in the execution of manipulation tasks, several of which addressed problems that are related to trajectory generation for robotic grasping.

In several researches, visual information is used for predicting a target trajectory [9,2,6]. Image processing requires a lot of processing power, which produces a processing delay. Human beings solve this problem using high level prediction because the throughput of human visual processing is low. But there are several problems requiring a long learning time to produce an optimal predictor, and they can not act in non-predictable condition.

On the other hand, in several researches a trajectory is directly generated using visual information. Xi et al. proposed a method based on the expression of the status equation [4]. In this method the time variable is replaced by a scalar that expresses sensor information. This approach is useful because conventional control approaches can be applied, but multilateral sensor information is difficult to express. Büehler et al. proposed a method in which the end-effector position is mapped to the sensory variables one by one, and with which they realized a juggling motion [5]. This approach has advantage that it is easy to express constraints of position and velocity, which is very important to execute dynamic manipulation. Several researches adopt similar approaches [16,15]. But they are purely based on geometric considerations and the dynamics of the manipulator is not considered. As a result they are not powerful enough to control a multi-axis manipulator which has heavy dynamics.

Such a direct online generator could not be applied easily to realtime manipulation tasks because the visual processing rate was too slow. But recently several types of high-speed vision systems have been developed. Our group has developed massively- parallel digital vision chip systems [7,8,10]. In our vision chip architecture the photo detectors and the parallel processing elements are integrated in a single system without the I/O bottleneck which results in a sampling rate higher than 1 KHz. This technique was applied to a sensory-motor fusion system [14] used to produce high- speed grasping and manipulation [13,12,11].

This shows that the speed of robotic visual recognition will soon become higher than that of a human. In such a situation the direct online generator becomes more important than the predictor, and a robotic system becomes able to act in non-predictable condition.

3 Vision-Based Online Trajectory Generator

In this section we propose a trajectory generator for catching a flying ball one-handed. This generator is based on a nonlinear mapping of visual information to the desired trajectory, and this nonlinear mapping is defined by learning based on constraints of geometry, dynamics and kinematics.

3.1 Problem Description

Suppose that the system can recognize the 3-dimensional position $\boldsymbol{r}_o \in \mathbb{R}^3$ of a ball by vision, and a desired trajectory of joints angle $\boldsymbol{q}_d \in \mathbb{R}^{m_q}$ (m_q : the number of joints) is generated from a nonlinear mapping of it as

$$\boldsymbol{q}_d = \boldsymbol{f}(\boldsymbol{r}_o). \tag{1}$$

And suppose that the joint angles $\boldsymbol{q} \in \mathbb{R}^{m_q}$ is controlled so as to track the desired position $\boldsymbol{q}_d$ using an appropriate controller. Fig.1 shows the block diagram.

The desired trajectory $\boldsymbol{q}_d$ should be planned so that the trajectory of the end effector $\boldsymbol{r} \in \mathbb{R}^3$ intersects the trajectory of the target $\boldsymbol{r}_o$ at a certain catching point. Fig.2 shows a catching task, in which the desired trajectory of an end effector $\boldsymbol{r}_d \in \mathbb{R}^3$ is generated so as to approach the object trajectory $\boldsymbol{r}_o$, and $\boldsymbol{r}$ is controlled to track $\boldsymbol{r}_d$. The trajectory and desired trajectory of the end effector are computed as

$$\boldsymbol{r}_d = \boldsymbol{l}(\boldsymbol{q}_d), \tag{2a}$$
$$\boldsymbol{r} = \boldsymbol{l}(\boldsymbol{q}), \tag{2b}$$

where the function $\boldsymbol{l}$ represents direct kinematics.

The problem is to define a formula of the function $\boldsymbol{f}$. Normally a manipulator has several constraints which are related to geometry, dynamics and kinematics. For this reason the function $\boldsymbol{f}$ should be optimized with respect to these constraints.

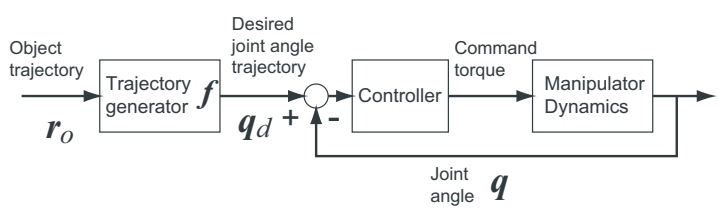


Fig. 1. The block diagram of a catching control. The desired trajectory of joint angles $\boldsymbol{q}_d$ is directly generated from the position of the target $\boldsymbol{r}_o$.

3.2 Terminal Geometric Constraints

Just as the target is caught, a match between the position and velocity of the target and the desired trajectory should be satisfied by the condition:

$$\boldsymbol{r}_e(t_c) = \dot{\boldsymbol{r}}_e(t_c) = \boldsymbol{0}, \tag{3}$$

where t_c represents the time just as the target is caught, and

$$\boldsymbol{r}_e \triangleq \boldsymbol{l} \circ \boldsymbol{f}(\boldsymbol{r}_o) - \boldsymbol{r}_o. \tag{4}$$

The vector $\boldsymbol{r}_e$ represents the error between the desired trajectory of the end effector and the target trajectory.

One solution of Eqn.(3) is the inverse kinematics function $\boldsymbol{f}(\boldsymbol{r}_o) = \boldsymbol{l}^{-1}(\boldsymbol{r}_o)$. But it is inconvenient for a trajectory generator since the domain of $\boldsymbol{l}^{-1}(\boldsymbol{r}_o)$ is limited within the area the end effector reaches,

In human catching motion, the 3-axis shoulder joints controls a rough position of the hand, and the elbow and wrist joints are mainly used for a match between the position and velocity of the target and the hand. It gives us a hint to set the formula of $\boldsymbol{f}$.

We select one appropriate elbow or wrist joint i, and suppose that the target is caught at the point $q_i = q_\alpha$, where $q_i \in \mathbb{R}$ is the i-th joint angle, and $q_\alpha \in \mathbb{R}$ is a constant value. We define a function $\boldsymbol{l}_c$ as

$$\boldsymbol{r}_o = \boldsymbol{l}_c(\boldsymbol{q}) \triangleq \boldsymbol{l}(\boldsymbol{q})|_{q_i=q_\alpha} + \left.\frac{\partial \boldsymbol{l}}{\partial q_i}\right|_{q_i=q_\alpha} (q_i - q_\alpha). \tag{5}$$

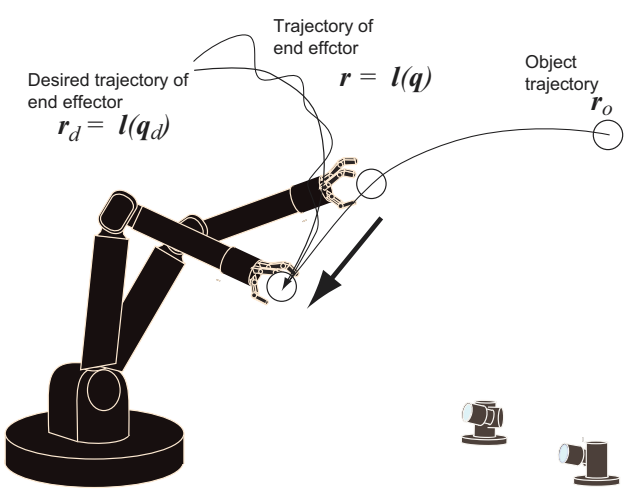


Fig. 2. Catching: The desired trajectory of an end effector $\boldsymbol{r}_d \in \mathbb{R}^3$ is generated so as to approach the object trajectory $\boldsymbol{r}_o$, and the position of the end effector $\boldsymbol{r}$ is controlled to track the desired trajectory $\boldsymbol{r}_d$.

This is the first order Taylor expansion of the direct kinematics $\boldsymbol{l}(\boldsymbol{q})$. Fig.2 describes the mapping $\boldsymbol{l}_c$. The first term $\boldsymbol{l}(\boldsymbol{q})|_{q_i=q_\alpha}$ represents the catching point, and $\left.\frac{\partial \boldsymbol{l}}{\partial q_i}\right|_{q_i=q_\alpha}$ represents a tangent vector of i-th joint motion at the catching point. In this function the distance between the target and the catching point is mapped to the i-th joint angle, and the target is caught at the point $q_i = q_\alpha$.

Using a function $\boldsymbol{l}_c$ we define a formula of a trajectory generator which satisfies the terminal geometry constraints as

$$\boldsymbol{q}_d = \boldsymbol{f}(\boldsymbol{r}_o) \triangleq \boldsymbol{g} \circ \boldsymbol{l}_c^{-1}(\boldsymbol{r}_o), \tag{6}$$

where

$$\boldsymbol{g}(\boldsymbol{q}) \triangleq \boldsymbol{q} + \boldsymbol{b}(\boldsymbol{q}) \tag{7}$$

and $\boldsymbol{b}(\boldsymbol{q})$ is an arbitrary function of joint angles $\boldsymbol{q}$, provided that $\dot{\boldsymbol{b}}|_{q_i=q_\alpha} = \boldsymbol{b}|_{q_i=q_\alpha} = \boldsymbol{0}$.

In Eqn.(6) the function $\boldsymbol{l}_c^{-1}$ is regarded as a nonlinear mapping from the target position coordinates to the joint angle coordinates. On the other hand, the function $\boldsymbol{g}$ is regarded as a nonlinear mapping from the joint angle coordinates to the joint angle coordinates, and it is expected that joint i is mainly used for a match between the target and the end effector.

It is clear that Eqn.(6) satisfies the terminal constraints described in Eqn.(3). But it is necessary to select the joint i and its catching angle q_α appropriately, so that the inverse mapping $\boldsymbol{l}_c^{-1}$ has a solution even if the target position $\boldsymbol{r}_o$ takes any value.

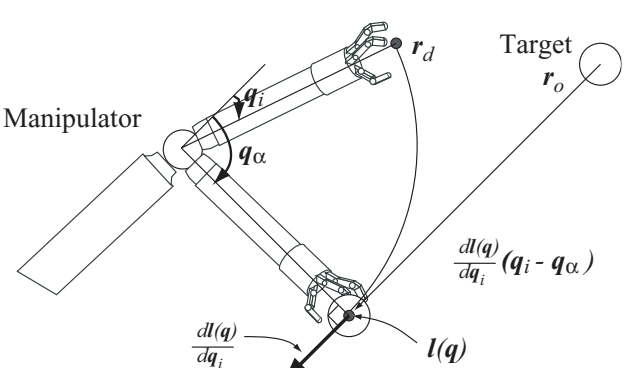


Fig. 3. A mapping from a target position $\boldsymbol{r}_o$ to a vector of joint angles $\boldsymbol{q}$. The distance between the target and the catching point is mapped to the i-th joint angle q_i.

3.3 Initial Geometric Constraints

At the beginning, a match between the position and velocity of the desired trajectory and of the actual effector should be satisfied by the condition:

$$\boldsymbol{q}_e(t_0) = \dot{\boldsymbol{q}}_e(t_0) = \boldsymbol{0}, \tag{8}$$

where $t_0 \in \mathbb{R}$ represents the initial time, and

$$\boldsymbol{q}_e \triangleq \boldsymbol{f}(\boldsymbol{r}_o) - \boldsymbol{q}, \tag{9}$$

where $\boldsymbol{q}_e \in \mathbb{R}^{m_q}$ represents the control error between the desired trajectory and the actual trajectory of the joint angles.

From Eqn.(7), $\boldsymbol{q}_e$ is rewritten as

$$\boldsymbol{q}_e = \boldsymbol{q}_o - \boldsymbol{q} + \boldsymbol{b}(\boldsymbol{q}_o), \tag{10}$$

where $\boldsymbol{q}_o = \boldsymbol{l}_c^{-1}(\boldsymbol{r}_o)$. To simplify the problem, we approximate the function $\boldsymbol{b}$ with analytical functions. In this paper we adopt a polynomial expression because initial geometric constraint are easily expressed in this way. Using Nth-order Taylor expansion at the point $q_i = q_\alpha$ and the terminal constraints, the function $\boldsymbol{b}$ is approximated as follows:

$$\boldsymbol{b}(\boldsymbol{q}) \simeq \sum_{\substack{0\le j_1,\, 0\le j_2,\cdots,0\le j_{m_q} \\ j_1+j_2+\cdots j_{m_q}\le N-2}}^{N} \boldsymbol{\alpha}_{j_1 j_2 \cdots j_{m_q}}\ {q_1}^{j_1} {q_2}^{j_2} \cdots (q_i - q_\alpha)^{j_i+2} \cdots {q_{m_q}}^{j_{m_q}}, \tag{11}$$

where $\boldsymbol{\alpha}_{j_1 j_2 \cdots j_{m_q}} \in \mathbb{R}^{m_q}$ is a constant parameter.

Based on Eqn.(11), the initial geometric constraints is written as,

$$C\,\boldsymbol{\alpha} = \begin{bmatrix} \boldsymbol{q}(t_0) - \boldsymbol{q}_o(t_0) \\ \dot{\boldsymbol{q}}(t_0) - \dot{\boldsymbol{q}}_o(t_0) \end{bmatrix}, \tag{12}$$

where $\boldsymbol{\alpha}$ is a vector which is defined by juxtaposing all $\boldsymbol{\alpha}_{j_1 j_2 \cdots j_{m_q}}$, and

$$C \triangleq \begin{bmatrix} \dfrac{\partial \boldsymbol{b}(t_0)}{\partial \boldsymbol{\alpha}} \\ \dfrac{\partial \dot{\boldsymbol{b}}(t_0)}{\partial \boldsymbol{\alpha}} \end{bmatrix} \tag{13}$$

is a constant matrix.

3.4 Dynamics constraints

Generally the dynamics of a manipulator in terms of the joint angle $\boldsymbol{q}$ is described by

$$M(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + H(\boldsymbol{q}, \dot{\boldsymbol{q}})\,\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}, \tag{14}$$

where $\boldsymbol{\tau} \in \mathbb{R}^{m_q}$ is the vector of joint torques, $M \in \mathbb{R}^{m_q \times m_q}$ is the inertia matrix, $H(\boldsymbol{q}, \dot{\boldsymbol{q}})\,\dot{\boldsymbol{q}} \in \mathbb{R}^{m_q}$ gives the coriolis and centrifugal force terms, and $\boldsymbol{g} \in \mathbb{R}^{m_q}$ is the gravity term.

On the other hand, the 1st order differentiation of Eqn.(1) becomes

$$\dot{\boldsymbol{q}}_d = F(\boldsymbol{r}_o)\,\dot{\boldsymbol{r}}_o \tag{15}$$

where $F \triangleq \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{r}_o}$.

Suppose $\boldsymbol{q} \simeq \boldsymbol{q}_d = \boldsymbol{f}(\boldsymbol{r}_o)$, Substituting Eqn.(15) into Eqn.(14),

$$\boldsymbol{\tau} = \tilde{M}(\boldsymbol{r}_o)F(\boldsymbol{r}_o)\ddot{\boldsymbol{r}}_o + \Big(\tilde{M}(\boldsymbol{r}_o)\dot{F}(\boldsymbol{r}_o) + \tilde{H}(\boldsymbol{r}_o, \dot{\boldsymbol{r}}_o)F(\boldsymbol{r}_o)\Big)\,\dot{\boldsymbol{r}}_o + \tilde{\boldsymbol{g}}(\boldsymbol{r}_o), \tag{16}$$

where

$$\tilde{M}(\boldsymbol{r}_o) \triangleq M(\boldsymbol{f}(\boldsymbol{r}_o)), \tag{17a}$$
$$\tilde{H}(\boldsymbol{r}_o, \dot{\boldsymbol{r}}_o) \triangleq H(\boldsymbol{f}(\boldsymbol{r}_o),\, F(\boldsymbol{r}_o)\,\dot{\boldsymbol{r}}_o), \tag{17b}$$
$$\tilde{\boldsymbol{g}}(\boldsymbol{r}_o) \triangleq \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{r}_o)). \tag{17c}$$

As a result, the following constraints should be satisfied:

$$\boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau}(\boldsymbol{r}_o, \dot{\boldsymbol{r}}_o, \ddot{\boldsymbol{r}}_o) \leq \boldsymbol{\tau}_{\max}, \tag{18}$$

where $\boldsymbol{\tau}_{\min} \in \mathbb{R}^{m_q}$ and $\boldsymbol{\tau}_{\max} \in \mathbb{R}^{m_q}$ are the minimum and maximum torque respectively.

3.5 Kinematics constraints

Supposing $\boldsymbol{q}_d = \boldsymbol{q}$, the joint velocity $\boldsymbol{\omega} \in \mathbb{R}^{m_q}$ is computed as

$$\boldsymbol{\omega} = \dot{\boldsymbol{q}} = F(\boldsymbol{r}_o)\,\dot{\boldsymbol{r}}_o. \tag{19}$$

As a result the following constraint should be satisfied:

$$\boldsymbol{\omega}_{\min} \leq \boldsymbol{\omega}(\boldsymbol{r}_o, \dot{\boldsymbol{r}}_o) \leq \boldsymbol{\omega}_{\max}, \tag{20}$$

where $\boldsymbol{\omega}_{\min} \in \mathbb{R}^{m_q}$ is the minimum angular velocity, and $\boldsymbol{\omega}_{\max} \in \mathbb{R}^{m_q}$ is the maximum angular velocity.

3.6 Trajectory generation algorithm

From Eqn.(5)$\sim$(7),(11), a direct mapping from the target position to the desired trajectory is defined as

$$\boldsymbol{q}_d = \boldsymbol{f}(\boldsymbol{r}_o) \triangleq \boldsymbol{g} \circ \boldsymbol{l}_c^{-1}(\boldsymbol{r}_o), \tag{21}$$

where

$$\boldsymbol{g}(\boldsymbol{q}) \triangleq \boldsymbol{q} + \sum_{\substack{0\le j_1, 0\le j_2, \cdots, 0\le j_{m_q} \\ j_1+j_2+\cdots j_{m_q}\le N-2}}^{N} \boldsymbol{\alpha}_{j_1 j_2 \cdots j_{m_q}} {q_1}^{j_1} {q_2}^{j_2} \cdots (q_i - q_\alpha)^{j_i+2} \cdots {q_{m_q}}^{j_{m_q}},$$

$$\boldsymbol{l}_c(\boldsymbol{q}) \triangleq \boldsymbol{l}(\boldsymbol{q})|_{q_i=q_\alpha} + \left.\frac{\partial \boldsymbol{l}}{\partial q_i}\right|_{q_i=q_\alpha} (q_i - q_\alpha).$$

From Eqn.(12),(18),(20) the optimal trajectory generator is found as the solution of

$$\min_{\boldsymbol{\alpha}} E(\boldsymbol{\alpha}) = \sum_{\boldsymbol{r}_o, \dot{\boldsymbol{r}}_o, \ddot{\boldsymbol{r}}_o} \left(\boldsymbol{\tau}^T K_\tau \boldsymbol{\tau} + \boldsymbol{\omega}^T K_\omega \boldsymbol{\omega} \right), \tag{23}$$

subject to

$$C\boldsymbol{\alpha} = \begin{bmatrix} \boldsymbol{q}(t_0) - \boldsymbol{q}_o(t_0) \\ \dot{\boldsymbol{q}}(t_0) - \dot{\boldsymbol{q}}_o(t_0) \end{bmatrix},$$
$$\boldsymbol{\tau}_{\min} \le \boldsymbol{\tau}(\boldsymbol{r}_o, \dot{\boldsymbol{r}}_o, \ddot{\boldsymbol{r}}_o;\, \boldsymbol{\alpha}) \le \boldsymbol{\tau}_{\max},$$
$$\boldsymbol{\omega}_{\min} \le \boldsymbol{\omega}(\boldsymbol{r}_o, \dot{\boldsymbol{r}}_o;\, \boldsymbol{\alpha}) \le \boldsymbol{\omega}_{\max},$$

where K_τ and K_ω are positive definite diagonal matrices.

To compute the minimum in Eqn.(23) various target trajectories are needed. But if the number of possible trajectories is large, it is difficult to consider every one of them. And if impossible trajectories are considered, the system performance goes down. To solve these problems it is necessary to optimize parameters using an online method.

The total control diagram is described in Fig.4. The parameters of the trajectory generator are changed depending on the dynamics and the kinematics of the manipulator and of the object. The adaptation can be executed in both online and offline modes. However, in dynamically changing environments it should be executed online.

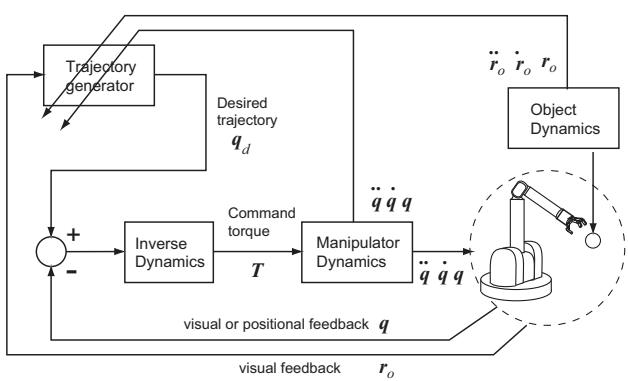


Fig. 4. Trajectory generation algorithm.

If the adaptation of the trajectory generator is not sufficient, the error is mainly compensated in the controller. As the trajectory generator approaches the optimal configuration, the load of the controller decreases.

4 Experiment

4.1 Experimental system

The experimental system is shown in Fig.6 (a), and the specification of the system is described in table 1.

The vision is a massively parallel vision system called CPV (column-parallel high speed vision system) [10]. The CPV has 128×128 photo detectors and pixel parallel processing array based on vision chip architecture and exclusive summation circuit for calculating moment values. Because visual processing is executed in parallel in the processing array, high-speed visual processing (moment detection, segmentation, etc) is realized within 1ms.

The arm is a 4-axis manipulator (Barrett Technology Inc.). The maximum speed of the end-effector is 6m/s, and its maximum acceleration is 30m/s^2 . Since the velocity of the human arm is about $6 - 10\text{m/s}$, the motion of the arm is as fast and reactive as a human motion. The hand is a 3-fingered and 3-axis hand. By reducing the weight of each finger and using high power actuators, fingers can close a 90-degree angle in 20ms.

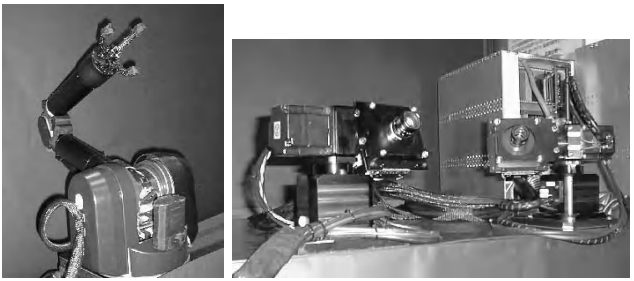
In Fig.6(b) kinematics of the arm is drawn. The arm has 4-axis joints, but, we used only the axis 1, the axis 2, and the axis 4 to simplify the problem. The axis 3 was fixed at the zero point. The initial position of the arm is $\boldsymbol{q} = [0, \frac{\pi}{4}, 0, 0]$, and $q_\alpha = \dfrac{\pi}{2}$.

The target was thrown in the field of two active vision system (AVS-II). Early image processing in CPV are achieved in the following order: segmentation of the image, extraction of the target area, and computation of the image moments. From these data, the position $\boldsymbol{r}^o$ of the target is computed. At last a Kalman filter is used on the observed target position to remove noise.

4.2 Experimental Result

We used a 4-th order polynomial expression in Eqn.(11). Parameters of this polynomial expression were optimized using all possible target trajectories in the neighborhood of the catching point. To simplify the problem we used an offline optimization method in this experiment. Fig.7 shows changes of the cost functions during optimization process. The cost of dynamics constraints $\sum \boldsymbol{\tau}^T K_\tau \boldsymbol{\tau}$ and the cost of kinematics constraints $\sum \boldsymbol{\omega}^T K_\omega \boldsymbol{\omega}$ were reduced at each optimization step.

Fig.8 shows the catching trajectory with optimization, and Fig.9 shows the time responses of the observed target and the end effector. These shows that smooth catching motion is achieved.



(a) Manipulator (b) Active Vision System (AVS-II)

Fig. 5. System components.

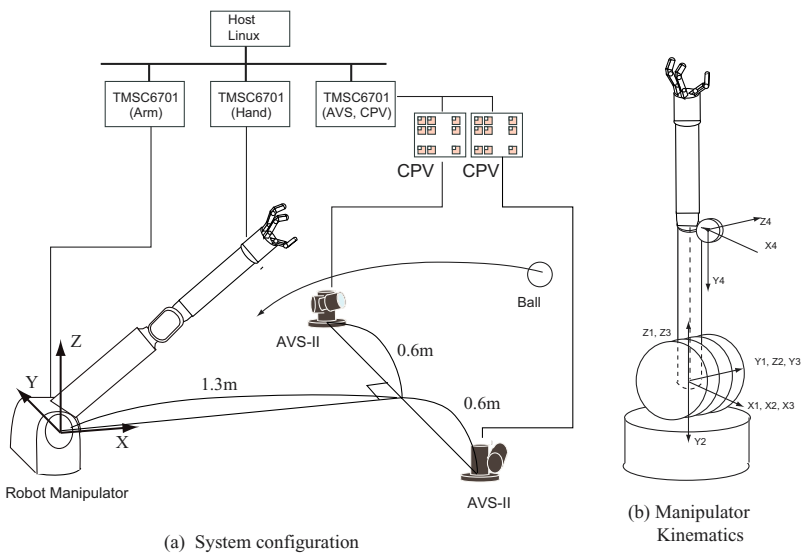


Fig. 6. Experimental system.

Fig.10 shows the time response of the joint angles, and Fig.11 shows the time response of the command torque during catching. In each fiugre the time response is shown both with optimization and without optimization ($\boldsymbol{\alpha} = \mathbf{0}$). In each case the desired trajectory of the end effector approached the target position finally. But, in the unoptimized method, the change of joint angle and command torque were bigger than that of the optimized method.This means that there were useless movement in the unoptimized method. As a result a catching task could not be achieved. On the other hand the optimized trajectory was smoother and there seemed to be not useless movement during catching. As a result stable motion could be achieved.

Table 1. System specification

High-speed active vision (AVS-II)	degrees of freedom	2DOF×2set
	maximum velocity (joint angle)	525[rad/s]
	resolution	128×128
	density	8[bit]
High-speed arm	degrees of freedom	4DOF
	maximum velocity (end-effector)	6 m/s
	maximum acceleration (end-effector)	30 m/s^2
	maximum force (end-effector)	26N
	total length	0.9m
High-speed hand	degrees of freedom	3fingers,3DOF
	closing time (90 degree)	20[ms]
	finger length	0.1[m]

In Fig.12 the catching task of a sphere is shown as a continuous sequence of pictures taken every frame (33ms). The target speed was fast (about 6m/s), and it is difficult to catch it only using simple feedback control. In our proposed algorithm an optimized mapping from visual information to motor command was computed. As a result there was very little delay in the arm motion, and the catching task was achieved.

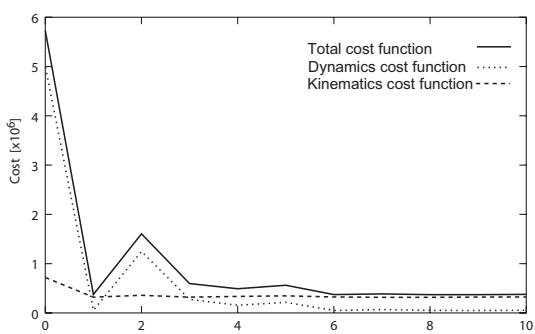


Fig. 7. Optimization of cost functions

5 Conclusion

In this paper we have proposed a realtime online trajectory generation algorithm using high-speed vision. This is based on a nonlinear mapping of visual information to the desired trajectory, and this nonlinear mapping is defined by learning based on constraints of dynamics and kinematics. As a result this method has more advantages to produce a reactive and flexible motion in the

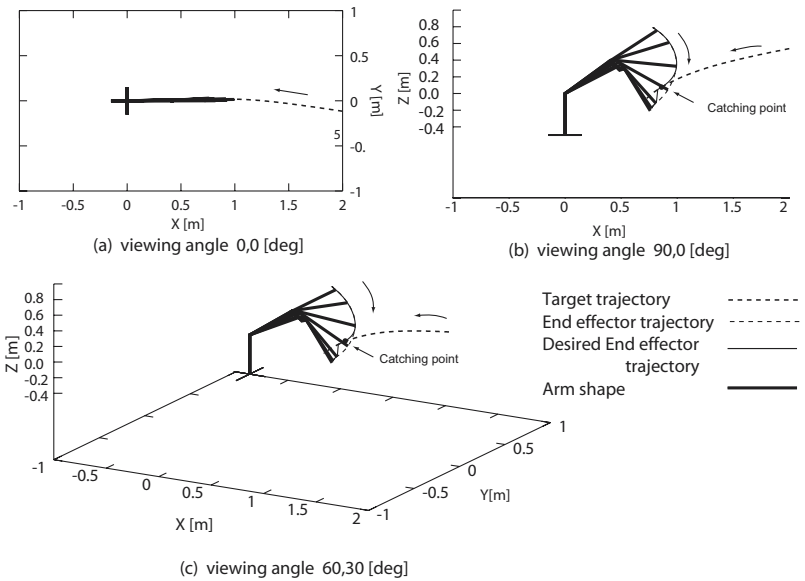


Fig. 8. Catching trajectory. Each trajectory is plotted every 1ms, while the arm shape is plotted every 50ms.

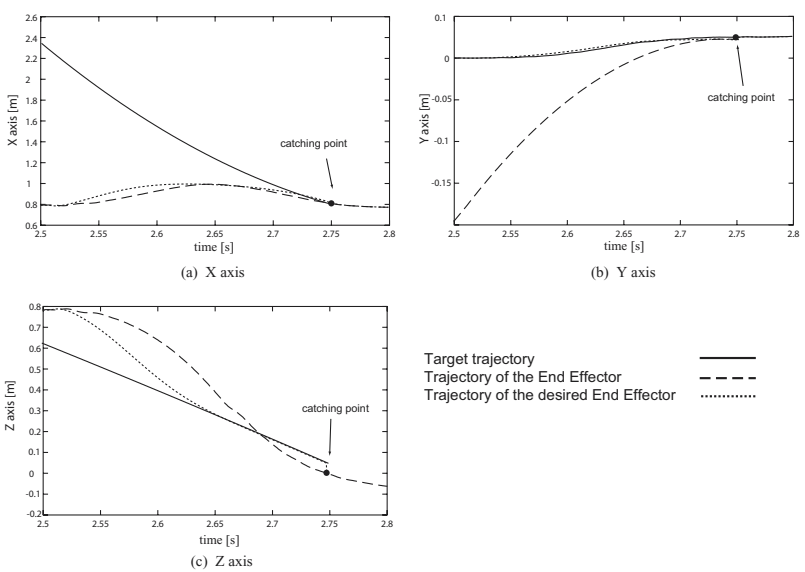


Fig. 9. Time response of the target and the end effector.

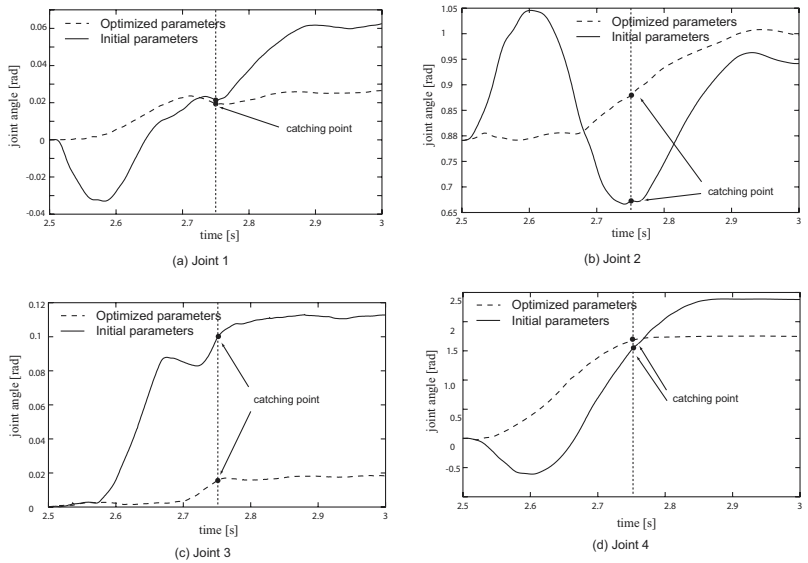


Fig. 10. Time response of joint angles. The dot lines represent the optimized trajectory. The solid lines represent the unoptimized trajectory.

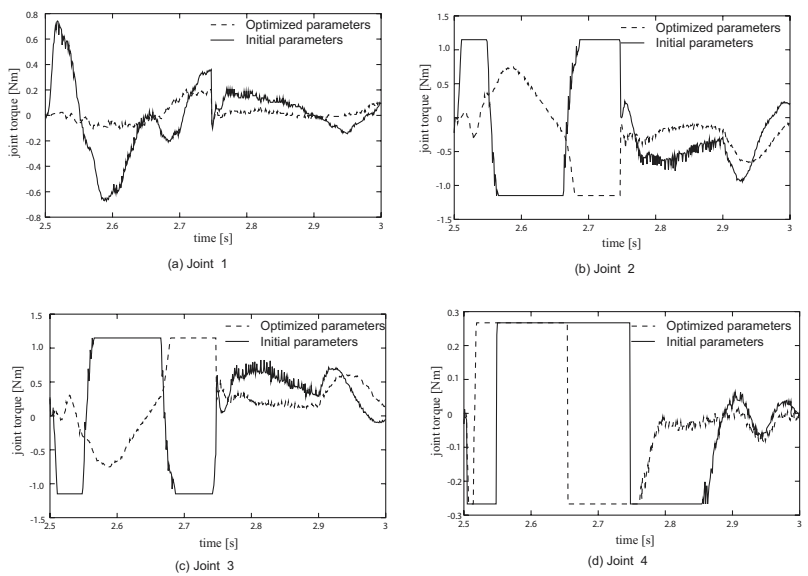


Fig. 11. Time response of joint command torques. The dot lines represent the optimized trajectory. The solid lines represent the unoptimized trajectory.

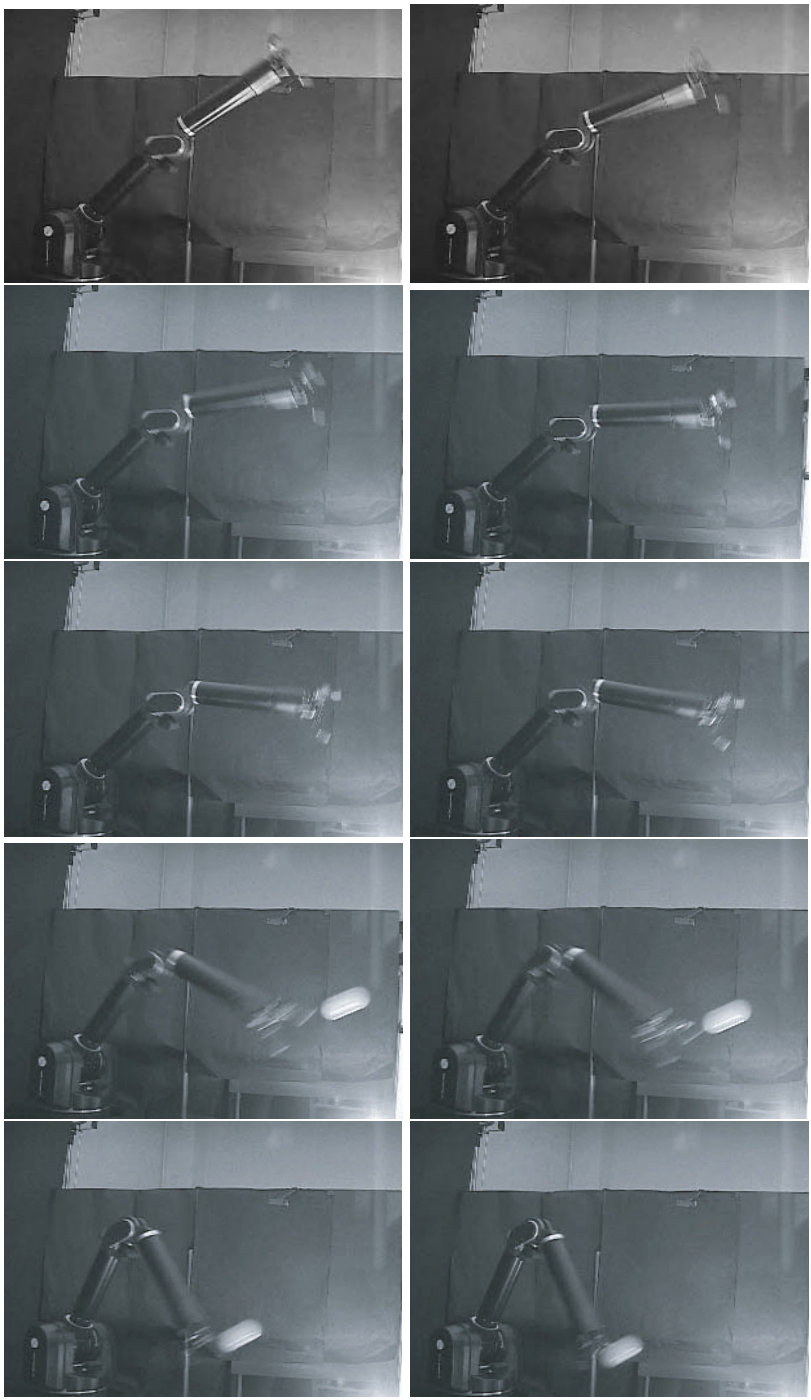


Fig. 12. Continuous sequence of pictures taken every frame (33ms).

case that it is difficult to predict target motion because of disturbance and uncertainty. The proposed method is applied to a catching task.

Future works are written as follows: quantitative estimation of effectiveness of mapping, a modular system using several types of mapping, application to other tasks except catching.

Soon the speed of robotic visual recognition will become higher than that of a human. In such a situation the direct online generator becomes more important than the predictor, and a robotic system becomes able to act in non-predictable condition.

References

1. Albus, J. (1991). Outline for a theory of intelligence. *IEEE Trans. on Syst., Man, and Cybern.*, 21(3):473–509.
2. Allen, P., Yoshimi, B., and Timucenko, A. (1991). Real-time visual servoing. *Proc. IEEE Int. Conf. Robot. and Automat.*, pages 2376–2384.
3. Bard, C., Turrell, Y., Fleury, M., Teasdale, N., Lamarre, Y., and Martin, O. (1999). Deafferentation and pointing with visual double-step perturbations. *Exp. Brain Res.*, 125:410–416.
4. B.Ghosh, N.Xi, and T.J.Tarn (1999). *Control in robotics and automation: sensor-based integration.* ACADEMIC Press.
5. Büehler, M., Koditshek, D., and Kindlmann, P. (1994). Planning and control of robotic juggling and catching tasks. *Int. J. of Robot. Res.*, 13(2):101–118.
6. Hong, W. and Slotine, J. (1995). Experiments in hand-eye coordination using active vision. *Proc. 4th Int. Symp. on Experimental Robot.*
7. Ishikawa, M., Morita, A., and Takayanagi, N. (1992). High speed vision system using massively parallel processing. *Proc. IEEE Int. Conf. Intelligent Robot. and Systems*, pages 373–377.
8. Komuro, T., Ishii, I., and Ishikawa, M. (1997). Vision chip architecture using general-purpose processing elements for 1ms vision system. *Proc. IEEE Int. Workshop on Computer Architecture for Machine Perception*, pages 276–279.
9. Kovio, A. and Houshangi, N. (1992). Real-time vision feedback for servoing robotic manipulator with self-tuning controller. *IEEE Trans. of Syst., Man, and Cybern.*, 21(1):134–142.
10. Nakabo, Y., Ishikawa, M., Toyoda, H., and Mizuno, S. (2000). 1ms column parallel vision system and its application of high speed target tracking. *Proc. IEEE Int. Conf. Robot. and Automat.*, pages 650–655.
11. Namiki, A. and Ishikawa, M. (2001). Sensory-motor fusion architecture based on high-speed sensory feedback and its application to grasping and manipulation. *Proc. Int. Symp. Robotics*, pages 784–789.
12. Namiki, A., Nakabo, Y., Ishii, I., and Ishikawa, M. (1999a). 1ms grasping system using visual and force feedback. *Video Proc. IEEE Int. Conf. Robot. Automat.*
13. Namiki, A., Nakabo, Y., Ishii, I., and Ishikawa, M. (1999b). High speed grasping using visual and force feedback. *Proc. IEEE Int. Conf. Robot. Automat.*, pages 3195–3200.
14. Namiki, A., Nakabo, Y., Ishii, I., and Ishikawa, M. (2000). 1ms sensory-motor fusion system. *IEEE/ASME Trans. Mechatron.*, 5(3):244–252.

15. Sakaguchi, T., Fujita, M., Watanabe, H., and Miyazaki, F. (1993). Motion planning and control for a robot performer. *Proc. IEEE Int. Conf. Robot. and Automat.*, 3:925–931.
16. Zhang, M. and Buehler, M. (1994). Sensor-based online trajectctory generation for smoothly grasping moving objects. *Proc. IEEE Int. Symp. Intelligent Control*, pages 141–146.

Stability Analysis of Invariant Visual Servoing and Robustness to Parametric Uncertainties

Ezio Malis

I.N.R.I.A. - ICARE Research Group, Sophia Antipolis, France.

Abstract. This paper concerns the stability analysis of a new visual servoing approach which is invariant on camera intrinsic parameters. Contrarily to standard methods, the invariant visual servoing approach can be used with a zooming camera or when the reference image is learned with a camera different from that used for servoing. Even if the error computed in an invariant space does not depend on the camera intrinsic parameters, they are needed to estimate the interaction matrix which links the camera velocity to the displacements of the features in the invariant space. Thus, calibration errors can affect the stability of the control law. For this reason, it is important to study the robustness of the proposed vision-based control with respect to uncertainties on the parameters of the system.

1 Introduction

Visual servoing is a very flexible method for the control of uncalibrated dynamic systems evolving in an unknown environment. Typical applications of visual servoing are the positioning of a robot and the tracking of objects using the information provided by an in-hand camera. The visual servoing approaches proposed in the literature [16,18] can be classified depending on the a priori knowledge available on the parameters of the system and on the observed object. If a 3D model of the object is available we can use a "*model-based*" approach [35,28], while if the 3D model of the object is unknown we must use a "*model-free*" approach [12,24]. Model-free methods, needs a preliminary learning step during which a reference image of the object is stored (teaching-by-showing). After the camera and/or the object have been moved, several vision-based control methods [2,12,25] have been proposed in order to drive the robot back to the reference position. When the current image observed by the camera is identical to the reference image the robot is back to the desired position. The model-free approach has the advantage of avoiding the knowledge of the model but it cannot be used with a zooming camera. If the camera *intrinsic* parameters (e.g. the focal length) change during the servoing, then the reference image must be learned again. Both model-based and model-free approaches are useful but, depending on the "a priori" knowledge we have of the scene, we must switch between them. In order to solve this problem, I propose in this paper a unified approach to vision-based control which can be used whether the model of the object is known or not [22]. The key idea of the unified approach, which

A. Bicchi, H.I. Christensen, D. Prattichizzo (Eds.): Control Problems in Robotics, STAR 4, pp. 265–280, 2003

is an extension of the work presented in [21], is to build a reference in a projective space which can be computed if the model is known or if an image of the object is available. Thus, only one low level visual servoing technique must be implemented at once. The new unified approach is called *invariant* visual servoing since we work in a projective space which is invariant to camera intrinsic parameters [21] and at the same time invariant to the knowledge of the 3D model of the object [22]. Contrarily to standard model-free approaches, this allows us to use the invariant visual servoing approach with a zooming camera or to learn the reference image with a camera different from that used for servoing [20]. There are various ways in which invariance to camera parameters can be obtained [14,34,21]. In [21] invariance to all the camera intrinsic parameters has been obtained by selecting three interest points to build a projective transformation. Consequently, the selection of the three points raised the problem of the best choice. The problem has been solved in [22] by building the projective transformation from all points available in the image. The control in the invariant space can be carried-out within the task-function framework [29] and its structure is very similar to standard image-based approaches [12,16,18]. Even if the task function of the invariant visual servoing does not depend on the camera intrinsic parameters, they are needed to estimate the interaction matrix which links the camera velocity to the displacements of the features in the invariant space [23]. Thus, calibration errors can affect the stability of the control law. In the recent past, research on the stability of image-based visual servoing has been concentrated on the solution of convergence problems [5]. Indeed, the image-based approach is a local method which, even in the absence of calibration errors, can fail if the initial camera displacement is too large [5]. In order to avoid these potential convergence problems several possible solutions have been proposed: hybrid, partitioned and interpolation approaches. In hybrid approaches, some global information is introduced by estimating the camera displacement between the current and reference views [25,27,8]. The rotation of the camera is thus controlled directly in the Cartesian space while some image-based information is used to control the translation. More recently, a partitioned approach [7] has been proposed in order to avoid the camera displacement reconstruction. Another solution to potential stability problem of the image-based approach is provided by interpolation approaches. These methods define a path in the image by interpolating initial and reference image features [17,26,23]. Thus, the error in the image is maintained small at each iteration of the control law. Interpolation approaches are an elegant solution to potential convergence problems of local approaches. In the case of the invariant visual servoing it is even possible to define a path in the projective space such that the robot follows a straight line in the Cartesian space [23]. Even using interpolation approaches, the problem of finding the local robustness domain of the vision-based control law has not been yet solved. Due to the complexity of the problem, only few theoretical results

have been obtained concerning the stability analysis of image-based visual servoing in the presence of calibration errors. The theoretical analysis has been carried out only in very simple cases [11], often considering a simplified model for the camera intrinsic parameters [6,19,9] but always supposing that the depth distribution was perfectly estimated [11,6,19,9,23]. The objective of this paper is to to study the robustness of visual servoing with respect to both intrinsic and extrinsic camera parameters. The application of robust control analysis tools [1,4,31] allows us to find an approximation of the robustness domain of the invariant visual servoing.

2 Modeling

2.1 Perspective projection

Let $\mathcal{F}_0$ be a frame attached to an object represented by the homogeneous coordinates of a discrete set of n 3D points $\boldsymbol{\mathcal{X}}_i = (X_i, Y_i, Z_i, 1)$ $(i = \{1, 2, ..., n\})$. Let $\mathcal{F}$ be the current camera frame and let the origin of the frame coincide with the center of projection. Let the plane of projection be parallel to the plane $(\overrightarrow{x}, \overrightarrow{y})$. Without loss of generality we can suppose that the distance between the two planes is 1. A 3D point $\boldsymbol{\mathcal{X}}_i \in \mathbb{P}^3$ is projected to the point $\mathbf{m}_i \in \mathbb{P}^2$ with normalized homogeneous coordinates:

$$\mathbf{m}_i = \frac{1}{Z_i} \left[\mathbf{R}_0 \ \mathbf{t}_0 \right] \boldsymbol{\mathcal{X}}_i = (x_i, y_i, 1) \tag{1}$$

where $\mathbf{R}_0$ and $\mathbf{t}_0$ are respectively the rotation and the translation between frame $\mathcal{F}_0$ and $\mathcal{F}$.

2.2 Camera model

Pinhole cameras perform a perspective projection of a 3D point. The information measured by the camera is an image point $\mathbf{p}_i$:

$$\mathbf{p}_i = \mathbf{K}\, \mathbf{m}_i = (u_i, v_i, 1) \tag{2}$$

The triangular matrix $\mathbf{K}(t)$ contains the camera intrinsic parameters:

$$\mathbf{K}(t) = \begin{bmatrix} f & sf & u_0 \\ 0 & rf & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

where f is the focal length (measured in pixels), u_0 and v_0 are the coordinates of the principal point (in pixels), s is the skew and r is the aspect ratio. In most of the papers dealing with the stability analysis of visual servoing several parameters are often supposed to be known. While s and r can be accurately estimated this is not true for the principal point. It has been shown not only

that its calibration is very sensitive to noise [13] but also that its inaccurate location has a significant effect on the calibration of others parameters or on the reconstruction accuracy [15]. The non-singular (3×3) matrix $\mathbf{K}(t)$ defines a projective transformation from the normalized coordinate system $\mathcal{M}$ to the image coordinate system $\mathcal{P}$. Thus, an approximation $\widehat{\mathbf{K}}$ of the matrix is needed in order to estimate a normalized point from a measured image point: $\widehat{\mathbf{m}}_i = \widehat{\mathbf{K}}^{-1}\,\mathbf{p}_i = (\widehat{x}_i, \widehat{y}_i, 1)$.

2.3 Invariance to camera intrinsic parameters

Suppose that n points are available. Using all the image points, with projective coordinates $\mathbf{p}_i = (u_i, v_i, 1)$, and all the the normalized points, with projective coordinates $\mathbf{m}_i = (x_i, y_i, 1)$, we can compute the following symmetric (3×3) matrices:

$$\mathbf{S}_p = \frac{1}{n}\sum_{i=1}^{n} \mathbf{p}_i\mathbf{p}_i^\top \qquad \text{and} \qquad \mathbf{S}_m = \frac{1}{n}\sum_{i=1}^{n} \mathbf{m}_i\mathbf{m}_i^\top \tag{4}$$

Since $\mathbf{p}_i = \mathbf{K}\mathbf{m}_i$, the matrix $\mathbf{S}_p$ can be written as a function of $\mathbf{S}_m$ and of the camera intrinsic parameters $\mathbf{K}$:

$$\mathbf{S}_p = \frac{1}{n}\sum_{i=1}^{n} \mathbf{p}_i\mathbf{p}_i^\top = \mathbf{K}\left(\frac{1}{n}\sum_{i=1}^{n} \mathbf{m}_i\mathbf{m}_i^\top\right)\mathbf{K}^\top = \mathbf{K}\,\mathbf{S}_m\,\mathbf{K}^\top \tag{5}$$

If the points are not collinear and $n > 3$ then $\mathbf{S}_p$ and $\mathbf{S}_m$ are positive definite matrices and they can be written, using a Cholesky decomposition, as:

$$\mathbf{S}_p = \mathbf{T}_p\,\mathbf{T}_p^\top \qquad \text{and} \qquad \mathbf{S}_m = \mathbf{T}_m\,\mathbf{T}_m^\top \tag{6}$$

where both $\mathbf{T}_p$ and $\mathbf{T}_m$ are (3×3) non-singular upper triangular matrices. Thus, from equations (5) and (6) we obtain:

$$\mathbf{T}_p = \mathbf{K}\,\mathbf{T}_m \tag{7}$$

The matrix $\mathbf{T}_p$ defines a projective transformation from the projective image space $\mathcal{P} \in \mathbb{P}^2$ to a new projective space $\mathcal{Q} \in \mathbb{P}^2$. Similarly, the matrix $\mathbf{T}_m$ defines a projective transformation from the projective space $\mathcal{M} \in \mathbb{P}^2$ to the same space $\mathcal{Q} \in \mathbb{P}^2$. We can compute the same vectors $\mathbf{q}_i \in \mathcal{Q}$ from image points:

$$\mathbf{q}_i = \mathbf{T}_p^{-1}\mathbf{p}_i \tag{8}$$

or from the knowledge of model of the object and the desired position:

$$\mathbf{q}_i = \mathbf{T}_m^{-1}\mathbf{m}_i \tag{9}$$

Equations (8) and (9) define the same point in $\mathcal{Q}$ since:

$$\mathbf{q}_i = \mathbf{T}_p^{-1}\mathbf{p}_i = \mathbf{T}_m^{-1}\mathbf{K}^{-1}\mathbf{p}_i = \mathbf{T}_m^{-1}\mathbf{K}^{-1}\mathbf{K}\mathbf{m}_i = \mathbf{T}_m^{-1}\mathbf{m}_i$$

As a consequence, the new projective space $\mathcal{Q}$ is independent on camera intrinsic parameters ($\mathbf{T}_m^{-1}$ and $\mathbf{m}_i$ do not depends on camera intrinsic parameters). Thus, the position of the camera can be controlled by driving the set of invariants $\mathbf{q}_i \ \forall i \in \{1, 2, ..., n\}$ to some reference values $\mathbf{q}_i^* \ \forall i \in \{1, 2, ..., n\}$ while the zoom of the camera can be controlled separately.

3 Vision-based control

The control of the pose of the camera rigidly attached to the robot end-effector is achieved by minimizing an error computed in the space $\mathcal{Q}$ which only depends on the camera pose. Let $\mathbf{t}$ and $\mathbf{R}$ be respectively the translation and the rotation between the reference camera frame $\mathcal{F}^*$ and the current camera frame $\mathcal{F}$. Let $\mathbf{r} = \theta\mathbf{u}$ be the (3×1) vector containing the axis of rotation $\mathbf{u}$ and the angle of rotation θ. Then, $\boldsymbol{\xi} = (\mathbf{t}, \mathbf{r})$ is a (6×1) vector containing global coordinates of an open subset $\mathcal{S} \subset \mathbb{R}^3 \times SO(3)$. The key idea of the proposed visual servoing approach is to always work in a projective space $\mathcal{Q} \in \mathbb{P}^2$ which can be computed from points belonging to the image space $\mathcal{P} \in \mathbb{P}^2$ (if the model is unknown) or points belonging to the projective space $\mathcal{M} \in \mathbb{P}^2$ (if the model is known). The approach does not need the explicit calibration of the camera and can be used even if the camera is zooming as shown in Figure 1.

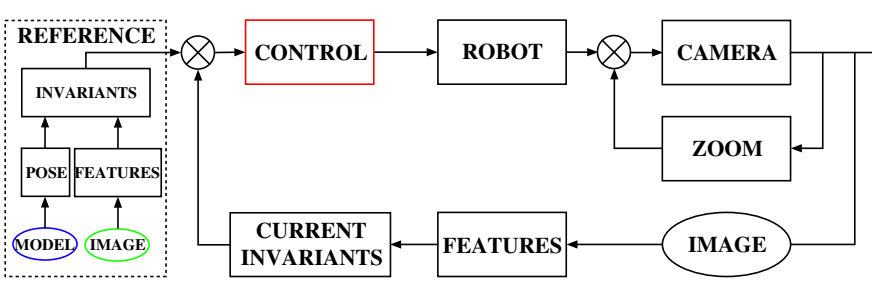


Fig. 1. Block diagram of the invariant control approach.

Similarly to the standard image-based approach, the control of the camera is achieved by stacking all the reference points of space $\mathcal{Q}$ in a (3n×1) vector $\mathbf{s}^*(\boldsymbol{\xi}^*) = (\mathbf{q}_1^*, \mathbf{q}_2^*, \cdots, \mathbf{q}_n^*)$. Similarly, the points measured in the current camera frame are stacked in the (3n×1) vector $\mathbf{s}(\boldsymbol{\xi}) = (\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_n)$. If $\mathbf{s}(\boldsymbol{\xi}) = \mathbf{s}^*(\boldsymbol{\xi}^*)$ then $\boldsymbol{\xi} = \boldsymbol{\xi}^*$ and the camera is back to the reference position whatever the camera intrinsic parameters. The derivative of vector $\mathbf{s}$ is:

$$\dot{\mathbf{s}} = \mathbf{L}\,\mathbf{v} \tag{10}$$

where the $(3n\times6)$ matrix $\mathbf{L}$ is called the interaction matrix and $\mathbf{v}$ is the velocity of the camera. The interaction matrix depends on current normalized points $\mathbf{m}_i(\boldsymbol{\xi}) \in \mathcal{M}$ ($\mathbf{m}_i$ can be computed from image points $\mathbf{m}_i = \mathbf{K}^{-1}\,\mathbf{p}_i$), on the invariant points $\mathbf{q}_i(\boldsymbol{\xi}) \in \mathcal{Q}$ and on the current depth distribution $\mathbf{z}(\boldsymbol{\xi}) = (Z_1, Z_2, ..., Z_n)$. Consider the following (6×1) task function:

$$\mathbf{e} = \widehat{\mathbf{L}}^{+}(\mathbf{s}-\mathbf{s}^*)$$

where $\widehat{\mathbf{L}}^{+}$ is the pseudo-inverse of an approximation of the true $(3n\times6)$ interaction matrix. In [12], the interaction matrix is supposed to be constant. In this paper, we consider the most general case when $\mathbf{C}$ depends on $\mathbf{s}$ as needed in [22]. In that case :

$$\dot{\mathbf{e}} = \frac{d\widehat{\mathbf{L}}^{+}}{dt}(\mathbf{s}-\mathbf{s}^*)+\widehat{\mathbf{L}}^{+}\dot{\mathbf{s}} = (\mathbf{O}(\mathbf{s}-\mathbf{s}^*)+\widehat{\mathbf{L}}^{+}\mathbf{L})\,\mathbf{v} \tag{11}$$

where $\mathbf{O}(\mathbf{s}-\mathbf{s}^*)$ is a 6×6 matrix such that $\mathbf{O}(\mathbf{s}-\mathbf{s}^*)|_{\mathbf{s}=\mathbf{s}^*} = 0$. Consider the following proportional control law:

$$\mathbf{v} = -\lambda\,\mathbf{e} \tag{12}$$

where λ is a positive scalar factor which tunes the speed of convergence. Using this control law the robot can be driven back to the reference position.

4 Stability Analysis

In this section, the local stability of the control law (12) is analyzed. The local stability is valid only in a neighborhood of the equilibrium point. However, when the initial error is large, it is possible to sample the trajectory and consider only small errors at each iteration of the control law [23]. Plugging equation (12) into (11), we obtain the following closed-loop equation:

$$\dot{\mathbf{e}} = -\lambda(\mathbf{O}(\mathbf{s}-\mathbf{s}^*)+\widehat{\mathbf{L}}^{+}\mathbf{L})\mathbf{e} \tag{13}$$

It is well know from control theory that the non-linear system (13) is locally asymptotically stable in a neighborhood of $\mathbf{s} = \mathbf{s}^*$ if and only if the linearized system is stable:

$$\dot{\mathbf{e}} = -\lambda\mathbf{Q}\mathbf{e} \tag{14}$$

where $\mathbf{Q} = \widehat{\mathbf{L}}^{+}\mathbf{L}|_{\mathbf{s}=\mathbf{s}^*}$. The linear system (14) is asymptotically stable *if and only if* $\mathbf{Q}$ has eigenvalues with positive real part:

$$\mathrm{real}(\mathrm{eig}(\mathbf{Q})) = \mathrm{real}(\mathrm{eig}(\widehat{\mathbf{L}}^{+}\mathbf{L})) > 0$$

The matrix depends on two set of unknown parameters: $\mathbf{Q} = \mathbf{Q}(\widehat{\mathbf{K}}, \widehat{\mathbf{z}})$. Obviously if $\mathbf{K} = \widehat{\mathbf{K}}$ and $\widehat{\mathbf{z}} = \mathbf{z}$ then $\mathbf{Q} = \mathbf{I}$ and the system is stable. If

$\mathbf{Q}$ is full rank then $\mathbf{e} = 0$ is the only equilibrium point of the system (i.e. if $\|\mathbf{e}\|$ decreases then it decreases towards $\mathbf{e} = 0$). It is well known from control theory that if $\mathbf{Q} > 0$ then the norm of the task function $\|\mathbf{e}\|$ decreases to zero. However, we need to prove that so does the error $\mathbf{s} - \mathbf{s}^*$ in the absence of local minima and singularities. The problem to know if, and in which case, a local minimum can be found is beyond the aim of this paper and it will be addressed in future work. In this paper, $\mathbf{Q}$ is supposed to be full rank. As already mentioned, the local asymptotic stability of the system can be proved considering the system linearized around $\mathbf{e} = 0$ (i.e. $\boldsymbol{\xi} = \boldsymbol{\xi}^*$):

$$\dot{\mathbf{e}} = -\lambda \mathbf{Q}(\boldsymbol{\xi})|_{\mathbf{e}=0}\mathbf{e} = -\lambda \mathbf{Q}(\boldsymbol{\xi}^*)\mathbf{e}$$

where $\mathbf{Q}(\boldsymbol{\xi}^*) = \widehat{\mathbf{L}}^+(\boldsymbol{\xi}^*)\mathbf{L}(\boldsymbol{\xi}^*)$. The system is locally stable if $\mathbf{Q}(\boldsymbol{\xi}^*) > 0$ since in that case $\mathbf{Q}(\boldsymbol{\xi}^*)$ has eigenvalues with positive real part. However, to prove the local asymptotic convergence of $\mathbf{e}$ to zero, we need also to show that $\mathbf{s} - \mathbf{s}^*$ never belongs to $\mathrm{Ker}(\widehat{\mathbf{J}}^+)$. This means that it exists a neighborhood $\mathcal{U}$ of $\boldsymbol{\xi}^*$ such that $\mathbf{e} = \widehat{\mathbf{L}}^+(\boldsymbol{\xi}^*)(\mathbf{s} - \mathbf{s}^*) \neq 0$, $\forall \boldsymbol{\xi} \in \mathcal{U}$ (i.e. $\mathbf{e} = 0$ only if $\mathbf{s}(\boldsymbol{\xi}) = \mathbf{s}^*$). Let us suppose that $\mathbf{s}(\boldsymbol{\xi}) \neq \mathbf{s}^*$ and therefore $\boldsymbol{\xi} \neq \boldsymbol{\xi}^* = 0$. The Taylor development of $\mathbf{s}(\boldsymbol{\xi})$ in a neighborhood of $\boldsymbol{\xi}^* = 0$ is:

$$\mathbf{s} - \mathbf{s}^* = \mathbf{L}(\boldsymbol{\xi}^*)\,\boldsymbol{\xi} + O^2(\boldsymbol{\xi}) \tag{15}$$

Multiplying by $\boldsymbol{\xi}^T\widehat{\mathbf{L}}^+(\boldsymbol{\xi}^*)$ (where $\boldsymbol{\xi}^T\widehat{\mathbf{L}}^+(\boldsymbol{\xi}^*) \neq 0$ since $\boldsymbol{\xi} \neq 0$ and $\widehat{\mathbf{L}}^+(\boldsymbol{\xi}^*)$ is full rank) both sides of equation (15) we obtain:

$$\boldsymbol{\xi}^T\widehat{\mathbf{L}}^+(\boldsymbol{\xi}^*)(\mathbf{s} - \mathbf{s}^*) = \boldsymbol{\xi}^T\widehat{\mathbf{L}}^+(\boldsymbol{\xi}^*)\mathbf{L}(\boldsymbol{\xi}^*)\boldsymbol{\xi} + O^3(\boldsymbol{\xi})$$

remember that if $\mathbf{Q} = \widehat{\mathbf{L}}^+(\boldsymbol{\xi}^*)\mathbf{L}(\boldsymbol{\xi}^*) > 0$ then $\boldsymbol{\xi}^T\mathbf{Q}\boldsymbol{\xi} \geq 2\sigma\|\boldsymbol{\xi}\|^2$, where $\sigma > 0$ is the minimum singular value of the positive definite matrix $\mathbf{Q} + \mathbf{Q}^T$. If $\widehat{\mathbf{L}}^+(\boldsymbol{\xi}^*)(\mathbf{s} - \mathbf{s}^*) = 0$ then:

$$0 \geq 2\sigma\|\boldsymbol{\xi}\|^2 + O^3(\boldsymbol{\xi})$$

that means:

$$\|\boldsymbol{\xi}\|^2 \leq |O^3(\boldsymbol{\xi})|$$

which is impossible since, by definition of $O^3(\boldsymbol{\xi})$, it exists a neighborhood of $\boldsymbol{\xi}^*$ in which:

$$\|\boldsymbol{\xi}\|^2 > |O^3(\boldsymbol{\xi})|$$

Therefore, $\mathbf{e} = \widehat{\mathbf{L}}^+(\boldsymbol{\xi}^*)(\mathbf{s} - \mathbf{s}^*) \neq 0$ if $\mathbf{s} \neq \mathbf{s}^*$ in a neighborhood of $\boldsymbol{\xi}^*$ and the system is locally asymptotically stable.

5 Robustness to Parametric Uncertainties

The parameters used to compute the control law are only roughly known and some of the can vary with time when the camera is zooming. Let $\mathbf{g} = (g_1, g_2, ..., g_p)$ a vector containing p parameters (in our case $p = n + 5$ where n is the total number of depth parameters and 5 is the total number of camera intrinsic parameters). The closed loop matrix $\mathbf{Q}$ can be expressed as a function of the true and estimated parameters $\mathbf{Q} = \mathbf{Q}(\mathbf{g}, \widehat{\mathbf{g}})$. Obviously, when the parameters are perfectly known $\mathbf{Q}(\mathbf{g}, \widehat{\mathbf{g}})|_{\widehat{\mathbf{g}}=\mathbf{g}} = \mathbf{I}$. Matrix $\mathbf{Q}$ can be viewed as a set of multivariate polynomials $\mathbf{Q}(\mathbf{g})$ whose variables are the parameters $\mathbf{g} = (g_1, g_2, ..., g_p)$. As already mentioned, the system is locally stable if and only if the eigenvalues of $\mathbf{Q}$ have positive real part. The eigenvalues of the (6×6) matrix $\mathbf{Q}$ are the roots of the characteristic polynomial:

$$p(\lambda, \mathbf{g}) = \sum_{k=0}^{6} c_k(\mathbf{g})\lambda^k$$

where the coefficients $c_k(\mathbf{g})$ are polynomial functions of the uncertain parameters $\mathbf{g}$. Given the measurement precision on the parameters $\mathbf{g}_i \in \left[\underline{g}_i, \overline{g}_i\right]$, it is possible to test the stability of the system. The necessary and sufficient conditions for the roots of the polynomial to be positive are obtained from the Routh-Hurwitz stability criterion without explicitly computing them. To check if the polynomial is stable we need to transform the bounds on the uncertainty into bounds on the coefficients of the polynomial. Indeed, if $\mathbf{g} \in \left[\underline{g}, \overline{g}\right]$ then $c_k(\mathbf{g}) \in [\underline{c}_k, \overline{c}_k]$. Thanks to the Kharitonov theorem [1], the stability of the uncertain polynomial can thus quickly checked using the Routh-Hurwitz stability criterion on the Kharitonov polynomials. However, the bounds on the coefficients of the polynomials are often conservative. Thus, if the Kharitonov polynomials are not stable we cannot conclude on the stability of the original system.

An approximation of the robustness domain can be obtained by bounding directly the eigenvalues of the closed-loop matrix. After setting $\tilde{g}_i = g_i - \widehat{g}_i$, matrix $\mathbf{Q}$ can be approximated as:

$$\mathbf{Q}(\mathbf{g}) \approx \mathbf{Q}_0 + \sum_{i=1}^{p} \tilde{g}_i \mathbf{Q}_i \tag{16}$$

where $\mathbf{Q}_0 = \mathbf{I}$, $\mathbf{Q}_i = \frac{\partial \mathbf{Q}(\mathbf{g})}{\partial g_i}|_{\mathbf{g}=\widehat{\mathbf{g}}}$ and $\tilde{\mathbf{g}}_i = (\tilde{g}_1, \tilde{g}_2, ..., \tilde{g}_p)$. Matrix $\mathbf{Q}$ can be regarded as a perturbation of the identity matrix $\mathbf{I}$. Let us define the SPECTRAL VARIATION of $\tilde{\mathbf{M}}$ with respect to $\mathbf{M}$ [31]:

$$\mathrm{sv}_{\mathbf{M}}(\tilde{\mathbf{M}}) = \max_i \min_j |\tilde{\lambda}_i - \lambda_j|$$

The Bauer-Fike theorem states that [31]:

$$\mathrm{sv}_{\mathbf{M}}(\tilde{\mathbf{M}}) \leq \|\tilde{\mathbf{M}} - \mathbf{M}\|$$

In our case, the spectral variation $\mathbf{S}$ with respect to $\mathbf{I}$ is:

$$\mathrm{sv}_{\mathbf{I}}(\mathbf{Q}) = \max_i |\tilde{\lambda}_i - 1| \leq \|\mathbf{E}\|$$

Thus, a simple sufficient condition for the stability of $\mathbf{Q}$ is $\|\mathbf{E}\| < 1$. Indeed, if $\|\mathbf{E}\| < 1$ then:

$$\max_i |\tilde{\lambda}_i - 1| < 1$$

which implies $\tilde{\lambda}_i > 0$. From the definition of spectral variation, all others eigenvalues $\lambda_k \ \forall k$ are such that $|\tilde{\lambda}_k - 1| \leq |\tilde{\lambda}_i - 1|$. Thus, $|\tilde{\lambda}_k - 1| < 1$ which means $\tilde{\lambda}_k > 0 \ \forall k$. Now, since $\mathbf{E} = \sum_{i=1}^{n} \tilde{g}_i \mathbf{Q}_i$:

$$\|\mathbf{E}\| \leq \sum_{i=1}^{m} |\tilde{g}_i| \|\mathbf{Q}_i\|$$

setting $\mu_i = \|\mathbf{Q}_i\| > 0$ the condition can be imposed by bounding the previous inequality:

$$\sum_{i=1, i \neq j}^{n} \mu_i |\tilde{g}_i| < 1$$

in this equation, the error $|\tilde{g}_i|$ is weighted by the scalars μ_i. The smaller is μ_i the less is the influence of the error $|\tilde{g}_i|$ on the stability of the system. Numerical examples show that the control is particularly robust to uncertainties on camera intrinsic parameters. As it will be shown in the experimental results, more than 50 % error of the focal length estimation can be tolerated. On the other hand, the system is less robust to uncertainties on the depth distribution. As an example, 10 % error on the depth distribution can be enough to make the system unstable.

6 Open problems

The standard procedure for visual servoing approaches is to design a control law for the nominal system [16,18]. Only few authors [3,32,33] have applied robust control techniques in order to directly take into account uncertainties at the design level. Despite these technique may be very conservatives their application to the invariant visual servoing scheme seems a possible solution to deal with the problem of large uncertainties on both camera intrinsic and extrinsic parameters. Indeed, if the environment is completely unknown and the system is uncalibrated the stability of the visual servoing in the presence of large errors on the parameters, can become a serious issue since the robustness domain is unknown. In addition to the problem of providing some information about the unknown depths of the object in the camera

frame (i.e. the camera *extrinsic* parameters), the proposed invariant approach introduces new control problems since several intrinsic parameters can vary with time. Since it is generally impossible to estimate accurately on-line all the time-varying parameters, it is necessary to take into account the influence of such uncertainties on the stability of the visual servoing directly at the design level. Another unresolved problem is the control of the zoom of the camera. Indeed, the zoom can be used to enlarge the field of view (zoom out) if the object is getting out of the image and to reduce the field of view (zoom in) to improve the extraction of the visual features. Unfortunately, these two objectives cannot be achieved at the same time and a compromise must be found.

7 Experimental Results

The vision-based control approach proposed in the paper has been validated on the 3 d.o.f. system Argés (see Figure 2) at INRIA Sophia-Antipolis. The Argés monocular system is an experimental platform used to develop active vision algorithms. The hardware is made of on-the-shelf components:

- a Computer controlled CCD Camera Acom1 PAL with a f=5.9 to 47.2 mm zoom-lens, automatic AGC 18dB and motor iris, a numerical auto-focus on 10bits, white balance, plus rs232C and video interface;
- a Pan-tilt turret, from RobotSoft, with a resolution of 3.086 minutes of arcs, a 4 lbs capacity and a speed up to 300 deg/sec, using constant current bipolar motor drives, via a rs232C interface;
- a linear degree of freedom, from CharlyRobot, with a resolution of 0.1 mm, using a slow screw driven control;

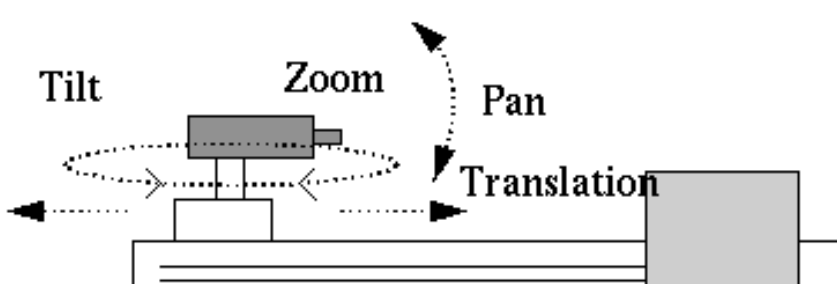
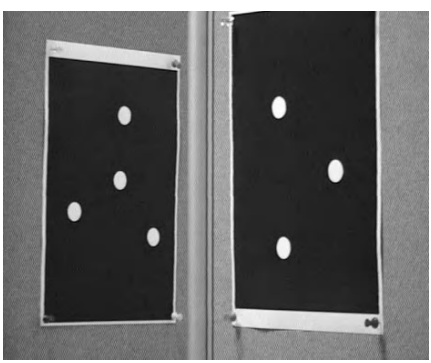


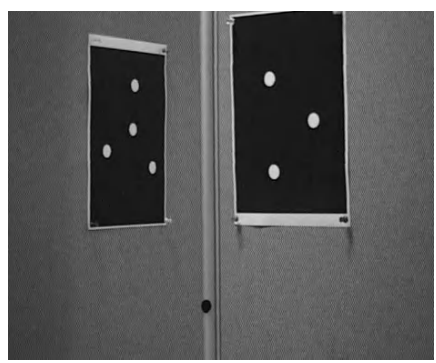
Fig. 2. The 3 d.o.f. robot Argés with a zooming camera.

The objective of the experiment is to position the camera with respect to an object represented by 7 points (see Figure 3(a)). Suppose that the model is unknown (i.e. model-based approaches cannot be used) and that the camera

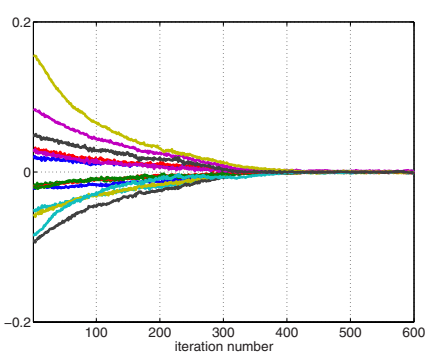
is zooming during the servoing (i.e. standard model-free approaches cannot be used). On the other hand, we can use the approach proposed in the paper. The camera, with a focal length $f^* = 2500$ pixels, is driven to the reference position and the corresponding image is stored (see Figure 3(a)). Then, the camera is displaced to its initial position. The initial displacement is $t_x = 350$ mm for the linear degree of freedom, $r_x = 4$ degrees and $r_y = 17$ degrees (i.e. tilt and pan respectively). At the initial position, the camera zooms out and the focal length is changed to $f_0 = 1670$ pixels. The corresponding initial image is given in Figure 3(b). The problem of matching/tracking features, common to all visual servoing techniques, is beyond the aim of this paper and it has been already investigated in the literature [10]. In the experiments, I focus on the general properties of the vision-based control approaches, therefore I consider that the matching/tracking problem has already been solved.



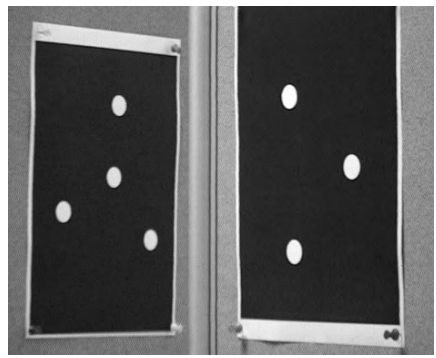
(a) Reference image



(b) Initial image



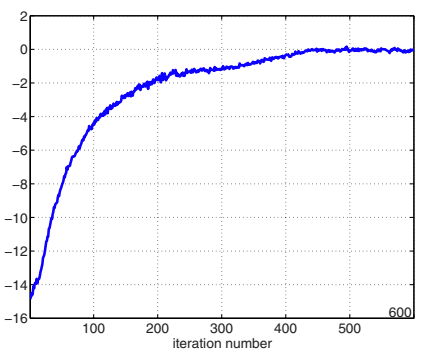
(c) Error in the invariant space



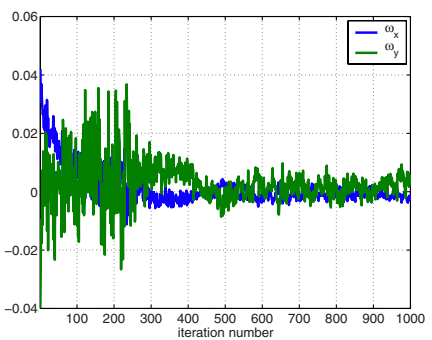
(d) Final image

Fig. 3. Camera positioning a with respect to 7 non coplanar points.

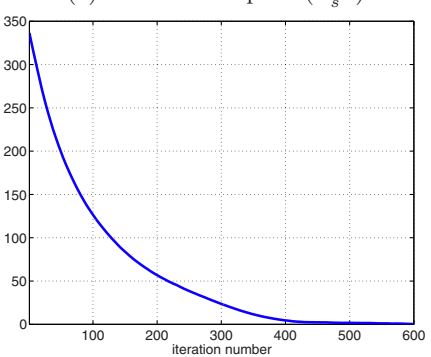
From the initial and reference images we can compute the invariants $\mathbf{q}_i$ and $\mathbf{q}_i^* \ \forall i \in \{1, 2, ..., n\}$ in the projective space $\mathcal{Q}$. Using the control law plotted in Figures 4(c) and (d), the error $\mathbf{q}_i - \mathbf{q}_i^*$ is zeroed (except for noise) (Figure 3(c)). Consequently, the camera is back to the reference position (i.e. the translational and rotational errors in Figures 4(e) and (f) converge to zero with an accuracy of 1 mm and 0.1 degrees). Since the camera parameters are unknown we use in the control law an extremely bad approximation of the focal length $\widehat{f} = 600$ pixels and we suppose that the principal point is in the center of the image. Despite the camera internal parameters $\widehat{\mathbf{K}}$ and the depth distribution $\widehat{\mathbf{z}}$ are only approximated, the control law is stable and converges. Obviously, if the calibration errors are big the performance of the visual servoing decreases (long time of convergence, unpredictable behavior). On the other hand, we can use the zoom of the camera to improve the performance of the servoing. During the servoing, the zoom is used to enlarge the field of view of the camera if the object is getting out of the image and to bound the size of the object to improve the robustness of features extraction. At the convergence, the camera focal length is $f \approx 2720$ (see Figure 4(b)). Consequently, the camera is back to the reference position, the images at the convergence (see Figure 4(d)) is not identical to the reference image because the camera has different focal lengths.



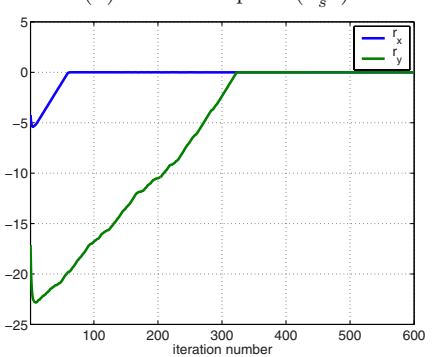
(a) Translation speed ($\frac{mm}{s}$)



(b) Rotation speed ($\frac{deg}{s}$)



(c) Translation error (mm)



(d) Rotation error (deg)

Fig. 4. Experimental results of the invariant visual servoing approach.

8 Conclusion

In this paper, it has been shown that the invariant visual servoing approach is robust to uncertainties on the parameters of the system. However, only a rough approximation of the robustness domain has been obtained. Since several intrinsic parameters can vary with time, and it is generally impossible to estimate accurately on-line all the time-varying parameters, it is necessary to take into account the influence of such uncertainties on the stability of the visual servoing. Thus, the approach could be considerably improved by directly applying robust control techniques at the design level.

Appendix A

The interaction matrix in the invariant space is obtained by stacking togheter all the interaction matrices $\mathbf{L}_{qi}$ relative to the points $\mathbf{q}_i$

$$\mathbf{L}_q = (\mathbf{L}_{q1}, \mathbf{L}_{q2}, ..., \mathbf{L}_{qn})$$

From equation (9), and knowing that $\dot{\mathbf{T}}_m^{-1} = -\mathbf{T}_m^{-1}\dot{\mathbf{T}}_m\mathbf{T}_m^{-1}$, we obtain the derivative of $\mathbf{q}_i$:

$$\dot{\mathbf{q}}_i = \dot{\mathbf{T}}_m^{-1}\mathbf{m}_i + \mathbf{T}_m^{-1}\dot{\mathbf{m}}_i = \mathbf{T}_m^{-1}\left(\dot{\mathbf{m}}_i - \mathbf{A}\mathbf{m}_i\right)$$

where $\mathbf{A}$ is the following triangular matrix:

$$\mathbf{A} = \dot{\mathbf{T}}_m\mathbf{T}_m^{-1} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & 0 \end{bmatrix}$$

The matrix $\mathbf{A}$ can be obtained by solving the following equation:

$$\dot{\mathbf{S}}_m = \dot{\mathbf{T}}_m\mathbf{T}_m^\top + \mathbf{T}_m\dot{\mathbf{T}}_m^\top = \dot{\mathbf{T}}_m\mathbf{T}_m^{-1}\mathbf{T}_m\mathbf{T}_m^\top + \mathbf{T}_m\mathbf{T}_m^\top\mathbf{T}_m^{-\top}\dot{\mathbf{T}}_m^\top = \mathbf{A}\mathbf{S}_m + \mathbf{S}_m\mathbf{A}^\top \tag{17}$$

The entries of the matrix $\mathbf{A}$ are linear functions of the camera velocity:

$$\mathbf{A}\mathbf{m}_i = \mathbf{C}_i\,\mathbf{v}$$

where $\mathbf{C}_i$ is a (3×6) matrix. Similarly, $\dot{\mathbf{m}}_i$ is a linear function of the camera velocity:

$$\dot{\mathbf{m}}_i = \mathbf{L}_{mi}\,\mathbf{v}$$

where:

$$\mathbf{L}_{mi} = \begin{bmatrix} -\frac{1}{Z_i} & 0 & \frac{x_i}{Z_i} & x_iy_i & -(1+x_i^2) & y_i \\ 0 & -\frac{1}{Z_i} & \frac{y_i}{Z_i} & 1+y_i^2 & -x_iy_i & -x_i \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Thus, the interaction matrix relative to the point $\mathbf{q}_i$ can be written as:

$$\mathbf{L}_{qi} = \mathbf{T}_m^{-1}\left(\mathbf{L}_{mi} - \mathbf{C}_i\right)$$

References

1. B. Barmish. *New Tools for Robustness of Linear Systems.* Macmilian, New York, 1994.
2. R. Basri, E. Rivlin, and I. Shimshoni. Visual homing: surfing on the epipole. *International Journal of Computer Vision*, 33(2):22–39, 1999.

3. D. Bellot and P. Danes. Towards an lmi approach to multicriteria visual servoing. In *European Control Conference*, Porto, Potugal, September 2001.
4. S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. SIAM, 1994.
5. F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In D. Kriegman, G. Hager, and A. Morse, editors, *The confluence of vision and control*, volume 237 of *LNCIS Series*, pages 66–78. Springer Verlag, 1998.
6. C. C. Cheah, S. Kawamura, and S. Arimoto. Feedback control for robotic manipulator with uncertain kinematics and dynamics. In *IEEE Int. Conf. on Robotics and Automation*, volume 4, pages 3607–3612, Leuven, Belgium, May 1998.
7. P. Corke and S. Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Transactions on Robotics and Automation*, 14(4):507–515, August 2001.
8. K. Deguchi. Optimal motion control for image-based visual servoing by decoupling translation and rotation. In *IEEE Int. Conf. on Intelligent Robots and Systems*, volume 2, pages 705–711, October 1998.
9. L. Deng, F. Janabi-Sharifi, and W. J. Wilson. Stability and robustness of visual servoing methods. In *IEEE Int. Conf. on Robotics and Automation*, volume 2, pages 1605–1609, Washington D.C., May 2002.
10. Y. Dufournaud, C. Schmid, and R. Horaud. Matching images with different resolutions. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 618–618, Hilton Head Island, South Carolina, USA, June 2000.
11. B. Espiau. Effect of camera calibration errors on visual servoing in robotics. In *3rd Int. Symposium on Experimental Robotics*, Kyoto, Japan, October 1993.
12. B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–326, June 1992.
13. O. Faugeras. *Three-dimensionnal computer vision: a geometric viewpoint*. MIT Press, Cambridge, MA, 1993.
14. G. D. Hager. Calibration-free visual control using projective invariance. In *IEEE Int. Conf. on Computer Vision*, pages 1009–1015, MIT, Cambridge (USA), June 1995.
15. R. Hartley and R. Kaucic. Sensitivity of calibration to principal point position. In *European Conference on Computer Vision*, volume 2, pages 433–446. Copenhagen, Denmark, May 2002.
16. K. Hashimoto. *Visual Servoing: Real Time Control of Robot manipulators based on visual sensory feedback*, volume 7 of *World Scientific Series in Robotics and Automated Systems*. World Scientific Press, Singapore, 1993.
17. K. Hashimoto and T. Noritsugu. Enlargement of stable region in visual servo. In *IEEE Conference on Decision and Control*, pages 3927–3932, Sydney, Australia, December 2000.
18. S. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Trans. on Robotics and Automation*, 12(5):651–670, October 1996.
19. R. Kelly, R. Carelli, O. Nasisi, B. Kuchen, and F. Reyes. Stable visual servoing of camera-in-hand robotic systems. *IEEE Trans. on Mechatronics*, 5(1):39–48, March 2000.
20. E. Malis. Vision-based control using different cameras for learning the reference image and for servoing. In *IEEE/RSJ International Conference on Intelligent Robots Systems*, volume 3, pages 1428–1433, Maui, Hawaii, November 2001.

21. E. Malis. Visual servoing invariant to changes in camera intrinsic parameters. In *International Conference on Computer Vision*, volume 1, pages 704–709, Vancouver, Canada, July 2001.
22. E. Malis. An unified approach to model-based and model-free visual servoing. In *European Conference on Computer Vision*, volume 4, pages 433–447, Copenhagen, Denmark, May 2002.
23. E. Malis. Vision-based control invariant to camera intrinsic parameters: stability analysis and path tracking. In *IEEE International Conference on Robotics and Automation*, volume 1, Washington, D.C., USA, May 2002.
24. E. Malis and F. Chaumette. Theoretical improvements in the stability analysis of a new class of model-free visual servoing methods. *IEEE Transaction on Robotics and Automation*, 18(2):176–186, April 2002.
25. E. Malis, F. Chaumette, and S. Boudet. 2 1/2 d visual servoing. *IEEE Trans. on Robotics and Automation*, 15(2):234–246, April 1999.
26. Y. Mezouar and F. Chaumette. Design and tracking of desirable trajectories in the image space by integrating mechanical and visibility constraints. In *IEEE Int. Conference on Robotics and Automation*, Seoul, South Korea, May 2001.
27. G. Morel, J. Szewczyk, S. Boudet, and J. Pot. Explicit incorporation of 2d constraints in vision based control of robot manipulators. In *Proc. ISER'99 : Experimental Robotics IV*, pages 99–108, Sidney, Australia, April 1999.
28. J. G. P. Martinet. Position based visual servoing using a nonlinear approach. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 531–536, Kyongju, Korea, October 1999.
29. C. Samson, M. Le Borgne, and B. Espiau. *Robot Control: the Task Function Approach*, volume 22 of *Oxford Engineering Science Series*. Clarendon Press, Oxford, UK, 1990.
30. C. Samson, M. Le Borgne, and B. Espiau. *Robot Control: the Task Function Approach*, volume 22 of *Oxford Engineering Science Series*. Clarendon Press, Oxford, UK, 1991.
31. G. W. Stewart and J.-g. Sun. *Matrix perturbation theory*. Computer Science and Science Computing. Harcourt Brace Jovanovich, 1990.
32. M. Sznaier, B. Murphy, and O. Camps. An lpv approach to synthesizing robust active vision systems. In *IEEE Conference on Decision and Control*, pages 2545–2550, Sydney, Australia, December 2000.
33. S. Tarbouriech and P. Soueres. Advanced control strategies for the visual servoing scheme. In *IFAC Symposium on Robot Control, SYROCO*, volume 2, pages 457–462, September 2000.
34. M. Werman, S. Banerjee, S. Dutta Roy, and M. Qiu. Robot localization using uncalibrated camera invariants. In *IEEE Int. Conference on Computer Vision and Pattern Recognition*, volume II, pages 353–359, Fort Collins, CO, June 1999.
35. W. J. Wilson, C. C. W. Hulls, and G. S. Bell. Relative end-effector control using cartesian position-based visual servoing. *IEEE Trans. on Robotics and Automation*, 12(5):684–696, October 1996.