LNCIS
LECTURE NOTES IN CONTROL AND INFORMATION SCIENCES
340

Moritz Diehl
Katja Mombaur (Eds.)

Fast Motions in Biomechanics and Robotics

Optimization and Feedback Control

Springer

Lecture Notes in Control and Information Sciences 340

Moritz Diehl, Katja Mombaur (Eds.)

Fast Motions in Biomechanics and Robotics

Optimization and Feedback Control

Springer

Editors

Moritz Diehl
Katja Mombaur

University of Heidelberg
Interdisciplinary Center for
Scientific Computing - INF 368
69120 Heidelberg
Germany
E-mail: M.Diehl@iwr.uni-heidelberg.de
katja.mombaur@iwr.uni-heidelberg.de

Library of Congress Control Number: 2006930404

ISSN print edition: 0170-8643
ISSN electronic edition: 1610-7411
ISBN-10 3-540-36118-9 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-36118-3 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media
springer.com

Typesetting: by the authors and techbooks using a Springer LaTeX macro package
Cover design: *design & production* GmbH, Heidelberg

Printed on acid-free paper SPIN: 11544777 89/techbooks 5 4 3 2 1 0

Preface

In the past decades, much progress has been made in the field of walking robots. The current state of technology makes it possible to create humanoid robots that nearly walk like a human being, climb stairs, or avoid small obstacles. However, the dream of a robot running as fast and as elegantly as a human is still far from becoming reality. Control of such fast motions is still a big technological issue in robotics, and the maximum running speed of contemporary robots is still much smaller than that of human track runners. The conventional control approach that most of these robots are based on does not seem to be suitable to increase the running speeds up to a biological level.

In order to address this challenge, we invited an interdisciplinary community of researchers from robotics, biomechanics, control engineering and applied mathematics to come together in Heidelberg at the first Ruperto-Carola-Symposium "Fast Motions in Biomechanics and Robotics – Optimization & Feedback Control" which was held at the International Science Forum (IWH) on September 7–9, 2005. The number of participants in this symposium was kept small in order to promote discussions and enable a fruitful exchange of ideas.

This volume contains a selection of papers from this symposium. Thus, one aim of the volume is to study the control and stabilization principles of biological motions and to determine which aspects can be exploited for the control of fast walking and running robots. In addition, the applicability of recent advances in control engineering, in particular in nonlinear model predictive control, to the field of walking robots is discussed. Another focus is on model based simulation and optimization methods that can be used for analysis and optimal design of motions and feedback control systems.

We would like to thank all authors for their interesting contributions, and all reviewers for their careful reading and their helpful comments which made it possible to assure a high quality of this book. In this context we would particularly like to thank Prof. Jim Bobrow for handling the editorial process of the papers in which we were involved.

Both the symposium and the work leading to this volume were financially supported by the Hengstberger-Prize donated by Dr. Klaus Georg and Sigrid Hengstberger, for which are very grateful. We also thank the members of the IWH team for their hospitality and friendly help with the organization of the symposium, in the beautiful surroundings of IWH.

Special thanks are due to Tanja Binder for doing the technical compilation of this book. We also thank Dr. Thomas Ditzinger and Heather King of Springer Verlag for their support with publishing this volume.

Heidelberg
June 2006

Moritz Diehl
Katja Mombaur

Contents

List of Contributors

Alamir, Mazen*
Laboratoire d'Automatique de Grenoble
CNRS-INPG-UJF
BP 46, Domaine Universitaire
38400 Saint Martin d'Hères – France
`mazen.alamir@inpg.fr`

Allgöwer, Frank*
Institute for Systems Theory and Automatic Control
University of Stuttgart
Pfaffenwaldring 9
70550 Stuttgart – Germany
`allgower@ist.uni-stuttgart.de`

Aoustin, Yannick*
Institut de Recherche en Communications et Cybernétique de Nantes
U.M.R. 6597 Ecole Centrale de Nantes
CNRS, Université de Nantes
1, rue de la Noë
BP 92 101
44321 Nantes Cedex 03 – France
`Yannick.Aoustin@irccyn.ec-nantes.fr`

Atkeson, Christopher
Carnegie Mellon University – USA

Berns, Karsten*
Arbeitsgruppe Robotersysteme
Fachbereich Informatik
Technische Universität Kaiserslautern
Postfach 3049
67653 Kaiserslautern – Germany
`berns@informatik.uni-kl.de`

Blickhan, Reinhard
Science of Motion
University of Jena
Seidelstr. 20
07749 Jena – Germany

Bobrow, James*
Department of Mechanical and Aerospace Engineering
Engineering Gateway 3220
University of California, Irvine
Irvine, CA 92697 – USA
`jebobrow@uci.edu`

Bock, Hans Georg*
Interdisciplinary Center for Scientific Computing
University of Heidelberg
INF 368
69120 Heidelberg – Germany
`scicom@iwr.uni-heidelberg.de`

* Participant of the symposium

Boyer, Frédéric
IRCCyN
Ecole Centrale de Nantes
Université de Nantes
UMR CNRS 6597
1, rue de la Noë
BP 92 101
44321 Nantes Cedex 03 – France
Frederic.Boyer@irccyn.ec-nantes.fr

Buche, Gabriel
LAG
Ecole Nationale d'Ingénieurs
Electriciens de Grenoble
BP 46
38402 St Martin d'Hères – France
Gabriel.Buche@inpg.fr

Chevallereau, Christine*
IRCCyN
Ecole Centrale de Nantes
Université de Nantes
UMR CNRS 6597
1, rue de la Noë
BP 92 101
44321 Nantes Cedex 03 – France
Christine.Chevallereau@irccyn.ec-nantes.fr

Clark, Jonathan*
University of Pennsylvania
Dept. of Electrical and Systems
Engineering
200 S. 33rd Street
Philadelphia, PA 19104-6314 – USA
jonclark@seas.upenn.edu

Cowan, Noah*
LIMBS Laboratory
Johns Hopkins University
230 Latrobe Hall
3400 N. Charles Street
Baltimore, MD 21218-2681 – USA
ncowan@jhu.edu

Diedam, Holger
Interdisciplinary Center for Scientific
Computing
University of Heidelberg
INF 368
69120 Heidelberg – Germany

Diehl, Moritz*
Interdisciplinary Center for Scientific
Computing
University of Heidelberg
INF 368
69120 Heidelberg – Germany
M.Diehl@iwr.uni-heidelberg.de

Djoudi, Dalila
IRCCyN
Ecole Centrale de Nantes
Université de Nantes
UMR CNRS 6597
1, rue de la Noë
BP 92 101
44321 Nantes Cedex 03 – France
Dalila.Djoudi@irccyn.ec-nantes.fr

Doi, Masahiro
Department of Micro-Nano Systems
Engineering
Nagoya University
Furo-cho 1
Chikusa-ku
Nagoya, 464-8603 – Japan
doi@mein.nagoya-u.ac.jp

Ebenbauer, Christian
Institute for Systems Theory and
Automatic Control
University of Stuttgart – Germany
ce@ist.uni-stuttgart.de

Findeisen, Rolf
Institute for Systems Theory and
Automatic Control
University of Stuttgart – Germany
findeise@ist.uni-stuttgart.de

Flörchinger, F.
Technische Universität
Kaiserslautern – Germany
floerchi@rhrk.uni-kl.de

Fukuda, Toshio*
Department of Micro-Nano Systems
Engineering
Nagoya University
Furo-cho 1
Chikusa-ku
Nagoya, 464-8603 – Japan
fukuda@mein.nagoya-u.ac.jp

Gerner, Hans Jürgen
Stiftung Orthopädische
Universitätsklinik Heidelberg

Geyer, Hartmut
Locomotion Laboratory
University of Jena
Dornburger Str. 23
07743 Jena – Germany

Giesl, Peter*
Zentrum Mathematik
Technische Universität München
Boltzmannstr. 3
85747 Garching – Germany
giesl@ma.tum.de

Grizzle, Jessy
Control Systems Laboratory
EECS Department
University of Michigan
Ann Arbor
Michigan 48109-2122 – USA
Grizzle@umich.edu

Hasegawa, Yasuhisa
Depertment of Intelligent Interaction
Technologies
University of Tsukuba
Tennodai 1-1-1
Tsukuba Ibaraki, 305-8573 – Japan
hase@esys.tsukuba.ac.jp

Iida, Funiya
Locomotion Laboratory
University of Jena
Dornburger Str. 23
07743 Jena – Germany

Kajima, Hideki
Department of Micro-Nano Systems
Engineering
Nagoya University
Furo-cho 1
Chikusa-ku
Nagoya, 464-8603 – Japan

Kloimwieder, Daniel Kaalim
Carnegie Mellon University – USA

Koditschek, Daniel
University of Pennsylvania
Dept. of Electrical and Systems
Engineering
200 S. 33rd Street
Philadelphia, PA 19104-6314 – USA
kod@ese.upenn.edu

Lamperski, Andrew
California Institute of Technology
Pasadena, CA – USA
andyl@cds.caltech.edu

Lee, Jusuk
LIMBS Laboratory
Johns Hopkins University
216A Latrobe Hall
3400 N. Charles Street
Baltimore, MD 21218-2681 – USA
jsl@jhu.edu

Lipfert, Susanne
Locomotion Laboratory
University of Jena
Dornburger Str. 23
07743 Jena – Germany

Longman, Richard*
Department of Mechanical Engineering
Columbia University
Rm. 232 S.W. Mudd Building
500 West 120th Street
New York, NY 10027 – USA
`rwl4@columbia.edu`

Luksch, Tobias*
Arbeitsgruppe Robotersysteme
Fachbereich Informatik
Technische Universität Kaiserslautern
Postfach 3049
67653 Kaiserslautern – Germany
`luksch@informatik.uni-kl.de`

Minekawa, Yohei
Locomotion Laboratory
University of Jena
Dornburger Str. 23
07743 Jena – Germany

Miossec, Sylvain
Joint Japanese-French Robotics Laboratory
Intelligent Systems Research Institute
National Institute of Advanced Science and Technology
AIST Tsukuba Central 2
Umezono 1-1-1
Tsukuba 305-8568 – Japan
`Sylvain.Miossec@aist.go.jp`

Mombaur, Katja*
Interdisciplinary Center for Scientific Computing
University of Heidelberg
INF 368
69120 Heidelberg – Germany
`Katja.Mombaur@iwr.uni-heidelberg.de`

Morris, B.
Control Systems Laboratory
EECS Department
University of Michigan
Ann Arbor
Michigan 48109-2122 – USA
`MorrisBJ@umich.edu`

Park, Frank
School of Mechanical and Aerospace Engineering
Seoul National University
Kwanak-ku, Shinlim-dong
San 56-1
Seoul 151-742 Korea
`fcp@snu.ac.kr`

Pratt, Jerry*
Florida Institute for Human and Machine Cognition
40 South Alcaniz Street
Pensacola, FL 32502 – USA
`jpratt@ihmc.us`

Raff, Tobias
Institute for Systems Theory and Automatic Control
University of Stuttgart – Germany
`raff@ist.uni-stuttgart.de`

Rummel, Jürgen
Locomotion Laboratory
University of Jena
Dornburger Str. 23
07743 Jena – Germany

Schablowski-Trautmann, Matthias*
Forschung Abteilung Orthopädie II
Schlierbacher Landstraße 200a
69118 Heidelberg – Germany
`Matthias.Schablowski@ok.uni-heidelberg.de`

Schmitt, John*
Department of Mechanical
Engineering
214 Rogers Hall
Oregon State University
Corvallis, OR 97331-6001 – USA
schmitjo@engr.orst.edu

Seyfarth, Andre*
Locomotion Laboratory
University of Jena
Dornburger Str. 23
07743 Jena – Germany
oas@uni-jena.de

Sideris, Athanasios
Department of Mechanical and
Aerospace Engineering
University of California, Irvine
Irvine, CA 92697 – USA
asideris@uci.edu

Tedrake, Russ
Massachusetts Institute
of Technology
Media Laboratory
32 Vassar Street
Cambridge, MA 02193 – USA
russt@mit.edu

Wagner, Heiko*
Biomechanics and Motor Control
WWU Münster
Horstmarer Landweg 62b
48149 Münster – Germany
Heiko.Wagner@uni-muenster.de

Westervelt, Eric*
Locomotion and Biomechanics
Laboratory
Department of Mechanical
Engineering
The Ohio State University
650 Ackerman Rd, Suite 255
Columbus, OH 43210 – USA
westervelt.4@osu.edu

Wieber, Pierre-Brice*
Projet BIPOP
INRIA Rhône-Alpes
38334 St-Ismier Cedex – France
Pierre-Brice.Wieber
@inrialpes.fr

Wisse, Martijn*
Department of Mechanical
Engineering
Delft University of Technology
Mekelweg 2
2628 CD Delft – The Netherlands
m.wisse@tudelft.nl

Re-injecting the Structure in NMPC Schemes

Application to the Constrained Stabilization of a Snakeboard

M. Alamir[1] and F. Boyer[2]

[1] Laboratoire d'Automatique de Grenoble, CNRS-INPG-UJF, BP 46, Domaine Universitaire, 38400 Saint Martin d'Hères, France
mazen.alamir@inpg.fr

[2] IRCCyN, 1, rue de la Noë BP 92 101, 44321 Nantes Cedex 3, France
Frederic.Boyer@emn.fr

Summary. In this paper, a constrained nonlinear predictive control scheme is proposed for a class of under-actuated nonholonomic systems. The scheme is based on fast generation of steering trajectories that inherently fulfill the contraints while showing a "*translatability*" property which is generally needed to derive stability results in receding-horizon schemes. The corresponding open-loop optimization problem can be solved very efficiently making possible a real-time implementation on fast systems (The resulting optimization problem is roughly scalar). The whole framework is shown to hold for the well known challenging problem of a snakeboard constrained stabilization. Illustrative simulations are proposed to assess the efficiency of the proposed solution under saturation constraints and model uncertainties.

1 Introduction

One of the most attractive features of Nonlinear Model Predictive Control (**NMPC**) schemes [1, 2] is their complete independence of the mathematical structure of the system's model. Indeed, from a conceptual point of view, given any system satisfying a rather intuitive set of assumptions, one may write down a concrete state feedback algorithm that theoretically asymptotically stabilizes a target equilibrium state.

Unfortunately, such generically defined formulations may lead to optimization problems that cannot be solved in the available computation time when rather fast dynamics are involved. This can be formulated in a kind of "*no free lunch*" statement:

Genericity reduces efficiency

Therefore, to overcome the consequences of the above unavoidable truth, specific features of each system under study have to be explicitly taken into account as far as possible.

In a series of papers [3, 4, 5, 6, 7, 8, 9], it has been shown that when the constrained stabilization is the main issue, that is, when the optimality is not rigorously required, efficient stabilizing **NMPC** schemes can be obtained as soon as open-loop "*steering trajectories*" can be generated by some systematic and efficient algorithm.

Now, the way these trajectories are generated is system dependent and the associated efficiency may be greatly increased if the specific structure of the system is explicitly exploited. This is what reduces the genericity to increase efficiency. By allowing low dimensional parametrization of these trajectories, corresponding low dimensional **NMPC** schemes can be defined in which, the decision variable is the parameter vector of the steering trajectory.

In this paper, it is shown that for a particular class of mechanical systems including the snakeboard, it is possible to use the particular structure of the system equations in order to derive efficient computation of parameterized steering trajectories. Moreover, these trajectories have the nice property of being structurally compatible with the saturation constraints on the actuators. Since in addition, they have the "*translatability property*", they can be used to implement a stable closed loop receding horizon feedback.

The paper is organized as follows: First, the particular class of mechanical systems under study is defined in Sect. 2 together with the associated assumptions. In Sect. 3, the proposed state feedback algorithm is explained and the associated convergence results are derived. The fact that the snakeboard falls into the particular class depicted in Sect. 2 is discussed in Sect. 4. Finally, illustrative simulations are proposed in Sect. 5 in order to assess the performance of the proposed solution.

2 The Class of Systems Considered

We consider nonlinear systems that may be described by the following set of ODE's

$$\dot{r} = f_1(\chi)g_1(\eta)\dot{\eta} \tag{1}$$

$$\dot{\eta} = f_2(\chi)g_2(\xi)\ ; \quad g_2(0) = 0 \tag{2}$$

$$\dot{\xi} = f_3(\xi, \chi, \dot{\chi}, u_1) \tag{3}$$

$$\ddot{\chi} = f_4(\xi, \chi, \dot{\chi}, u_2) \tag{4}$$

where equations (1)–(2) stands for a KINEMATIC stage while equations (3)–(4) represent the DYNAMIC stage. $r \in \mathbb{R}^{n_r}$ is a kind of generalized position; $\eta \in \mathbb{R}^{n_\eta}$ is an orientation variable; $\xi \in \mathbb{R}^{n_\xi}$ is a generalized velocity while $\chi \in \mathbb{R}$ stands for an internal configuration variable. $f_1 : \mathbb{R} \to \mathbb{R}$, $g_1 : \mathbb{R}^{n_\eta} \to \mathbb{R}^{n_r \times n_\eta}$. All the maps invoked in (1)–(4) are assumed to be continuously differentiable.

Note that equation (1) generally describes a **nonholonomic** constraint, namely, a constraint on the velocities that is not derived from a position related constraints. The control inputs u_1 and u_2 have to meet the following saturation constraints

$$\forall t \quad , \quad u_i(t) \in [-u_i^{max}, +u_i^{max}] \; ; \quad i \in \{1,2\} \tag{5}$$

for some given upper bounds u_1^{max} and u_2^{max}.

The aim of the present paper is to derive a state feedback algorithm that steers the sub-state (r, η, ξ) of the system (1)–(4) to the origin under the saturation constraint (5). Note that since $g_2(0) = 0$, the origin $(r, \eta, \xi) = 0$ is an equilibrium position for the dynamics (1)–(3) provided that ξ can be maintained at 0 by convenient use of the control input u_1 as it is suggested by Assumption 1 hereafter. Note also that χ is an internal configuration variable whose value is irrelevant for the control objective. This is for instance the angular position of the wheels in the snakeboard example (see Sect. 4).

Beside the structure depicted in equations (1)–(4), the class of systems of interest has to satisfy the following assumptions:

Assumption 1

1. *For all χ^f, there exists a feedback law*

$$u := (u_1, u_2) = K_1(\xi, \chi, \dot{\chi}, \chi^f)$$

under which, the closed loop behavior respects the constraints (5) and such that

$$(\xi = 0, \dot{\chi} = 0, \chi = \chi^f)$$

is a globally asymptotically stable equilibrium for the closed loop dynamics defined by (3)–(4) and K_1. Furthermore, the subset $\{\xi = 0\}$ is invariant under the closed loop behavior.

2. *For all $\chi^f \neq 0$ and all η^f, there exists a feedback law*

$$u = (u_1, u_2) = K_2(\eta, \xi, \chi, \dot{\chi}, \eta^f, \chi^f)$$

such that the closed loop behavior respects the constraints (5) and such that

$$(\eta = \eta^f, \xi = 0)$$

is a globally asymptotically stable equilibrium for the closed loop dynamics defined by (2)–(3) and K_2. Furthermore, the set $\{\chi = \chi^f, \dot{\chi} = 0\}$ is invariant under this dynamics.

3. *For all η^0, there exists a parameterized sequence $(\eta^k(p, \eta^0))_{k \geq 0}$ (defined for some parameter vector $p \in \mathbb{P}$ where $\mathbb{P}$ is a compact set) that is continuous in (p, η^0) such that $\eta^0(p, \eta^0) = \eta^0$, $\lim_{k \to \infty} \eta^k(p, \eta^0) = 0$* (**exponentially**) *and the following rank condition is satisfied for all $p \in \mathbb{P}$:*

$$Rank\Big[A(p, \eta^0) = \big(A_1(p, \eta^0) \ldots A_j(p, \eta^0) \ldots\big)\Big] = n_r \tag{6}$$

where

$$A_j(p,\eta^0) = \int_{\eta^{j-1}(p,\eta^0)}^{\eta^j(p,\eta^0)} g_1(\eta)d\eta \tag{7}$$

with $\sigma_{min}(A(p,\eta^0)) > s_{min} > 0$. *Furthermore, for all* $p \in \mathbb{P}$, *there exists* $p^+ \in \mathbb{P}$ *such that*

$$\eta^k(p^+,\eta^1(p,\eta^0)) = \eta^{k+1}(p,\eta^0) \quad \forall k \geq 0 \quad \text{(translatability)} \tag{8}$$

4. *The function* g_2 *in (2) is such that for any pair* (η^a,η^b), *there exists a sequence*

$$(\eta_j^{(a,b)})_{j=1}^{n_r} \ ; \quad \eta_1^{(a,b)} = \eta^a \ ; \quad \eta_{n_r}^{(a,b)} = \eta_b$$

such that the matrix

$$M(\eta^a,\eta^b) := \begin{pmatrix} \int_{\eta_1^{(a,b)}}^{\eta_2^{(a,b)}} g_1(\eta)d\eta \dots \int_{\eta_{n_r-1}^{(a,b)}}^{\eta_{n_r}^{(a,b)}} g_1(\eta)d\eta \end{pmatrix} \in \mathbb{R}^{n_r \times n_r} \tag{9}$$

has full rank.

5. *The function* f_1 *maps some domain* $\mathcal{D} \subset \mathbb{R}$ *onto* $\mathbb{R} - \{0\}$ *so that an inverse map* $f_1^{-1} : \mathbb{R} - \{0\} \rightarrow \mathcal{D}$ *may be defined by fixing some selection rule (when needed).*

In the following sequel, the state of the system is denoted by x, namely

$$x := \begin{pmatrix} r \ \eta \ \xi \ \chi \ \dot{\chi} \end{pmatrix}^T \in \mathbb{R}^n \ ; \quad n = n_r + n_\eta + n_\xi + 2$$

3 The Proposed Feedback Algorithm

The basic idea of the control algorithm is to decompose the behavior of the controller into basically 2 modes

1. In the first mode, $\xi \approx 0$ and the feedback K_1 is used to steer the state χ from some initial value χ_{j-1} to some final desired one χ_j. This is possible thanks to assumption 1.1. Note that under this mode the position r as well as the orientation variable η are maintained almost constant (since $\xi \approx 0$ and $g_2(0) = 0$ by assumption 1.5).
2. In the second mode, $\chi \approx \chi_j$ is maintained almost constant while the feedback K_2 is used to steer the variable η from some initial value η^{j-1} to some final value η^j. Note again that this is possible thanks to assumption 1.2. Note moreover that under this mode, ξ asymptotically goes to 0 enabling the first mode to be fired again.

Note that from equations (1) and (7), it comes that under constant $\chi \equiv \chi_j$, when the mode 2 is used with η changing from $\eta^{j-1}(p,\eta^0)$ to $\eta^j(p,\eta^0)$, the corresponding variation in r is given by:

$$\Delta_j r := \Big[A_j(p,\eta^0)\Big] f_1(\chi_j) \quad \text{under constant } \chi \equiv \chi_j \text{ (mode 2)} \tag{10}$$

Therefore, the condition

$$r^0 + A(p,\eta^0) \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_j \\ \vdots \end{pmatrix} = 0 \ ; \quad v_j \neq 0 \quad \text{for all } j \ ; \quad \chi_j = f_1^{-1}(v_j) \tag{11}$$

characterizes the family of sequences $(\chi_j)_{j\geq 0}$ such that when the two-modes procedure defined above is applied in an open-loop way, the vector (r,η,ξ) is steered to 0.

The state feedback proposed in the present paper amounts to use the open-loop strategy defined above in a receding horizon way. This is because the sequences $(\eta^j)_{j\geq 0}$ and $(\chi_j)_{j\geq 0}$ may become irrelevant because of unavoidable disturbances and because of the simple fact that during mode 1 [resp. mode 2], $\xi = 0$ [resp. $\chi = \chi_j$] cannot be rigorously satisfied making necessary to re-compute the steering strategy.

In order to properly define the proposed receding horizon formulation, the following definitions are needed:

Definition 1. *Given any (p,r^0,η^0), let $q \in \mathbb{N}$ be a sufficiently high integer for $A_j(p,\eta^0)$ to be negligible for $j > q$ and define the vector $\hat{v}(p,r^0,\eta^0) \in \mathbb{R}^q$ to be the solution of the following Linear Programming (LP) problem:*

$$\hat{v}(p,r^0,\eta^0) := \text{Arg} \min_{\tilde{v}\in\mathbb{R}^q} \Big[-\sum_{j=1}^{q} \tilde{v}_j\Big]$$
$$\text{under } r^0 + \big[A_{1\to q}(p,\eta^0)\big]\tilde{v} = 0 \text{ and } \|\tilde{v}\|_\infty \leq v_{max} \tag{12}$$

where $A_{1\to j}(p,\eta^0)$ is the matrix built with the j first matrix terms of $A(p,\eta^0)$ [see (6)] while v_{max} is a sufficiently high value making the above constrained problem feasible for any initial value r^0 of interest.

Note that for each candidate value of $p \in \mathbb{P}$, the LP problem (12) may be solved almost instantaneously using LP solvers. Note also that the cost function to be minimized suggests solutions that avoid vanishing components of $\hat{v}$. However, if in spite of this $\hat{v}_k(p,r^0,\eta^0) = 0$ for some k, intermediate values have to be introduced in the sequence $(\eta^j(p,\eta^0))_{j\geq 0}$ between $\eta^k(p,\eta^0)$ and $\eta^{k+1}(p,\eta^0)$ in order to remove this vanishing components without altering the remaining solution, namely $\hat{v}_i(p,r^0,\eta^0)$ for $i \geq k+1$. This is possible thanks to assumption 1.4. Indeed:

- Assume that for some k, one has $\hat{v}_k = 0$, this means that the positions r at the sampling instants k and $k+1$ are the same.
- Consider the following matrices defined according to (9):

$$M(\eta^k, \eta^{k+\frac{1}{2}})\ ; \quad M(\eta^{k+\frac{1}{2}}, \eta^{k+1}) \quad \text{where} \quad \eta^{k+\frac{1}{2}} := \frac{\eta^k + \eta^{k+1}}{2}$$

together with the corresponding sequences

$$\left(\eta_j^{k,k+\frac{1}{2}}\right)_{j=1}^{n_r}, \left(\eta_j^{k+\frac{1}{2},k+1}\right)_{j=1}^{n_r}$$

- Our aim is to prove that there exists a sequence of controls

$$\left(\hat{v}_j^k\right)_{j=1}^{2n_r}$$

with no zero elements and such that the net variation on r vanishes. But this amounts to find a vector $v \in \mathcal{C}_p := [-v_{max}, v_{max}]^{2n_r}$ with $v_i \neq 0$ for all i such that the vector

$$\underbrace{\left[M(\eta^{k+\frac{1}{2}}, \eta^{k+1})\right]^{-1}\left[M(\eta^k, \eta^{k+\frac{1}{2}})\right]}_{A} v \tag{13}$$

has no zero elements. But this is clearly always possible since the matrix A is regular according to assumption 1.4. (See lemma 1 in the appendix for a brief proof of this evident fact).

Repetitive application of this technique enables all vanishing (or too small) components of $\hat{v}$ to be removed.

Definition 2. *Given any (r^0, η^0), the optimal parameter vector $\hat{p}(r^0, \eta^0)$ is defined by*

$$\begin{aligned}\hat{p}(r^0, \eta^0) &:= \text{Arg}\min_{p \in \mathbb{P}} J(p, r^0, \eta^0) \\ &:= \sum_{j=0}^{q} \left[\|r^0 + A_{1\to j}(p, \eta^0)\hat{v}_{1\to j}(r^0, \eta^0)\|^2 + \alpha \cdot |\eta^j(p, \eta^0)|^2\right]\end{aligned} \tag{14}$$

namely, $\hat{p}(r^0, \eta^0)$ minimizes a quadratic cost on the excursion of the configuration vector (r, η).

Note that definition 2 assumes that the admissible parameter set $\mathbb{P}$ is such that (14) admits a solution. This is typically guaranteed because of the continuity of the cost function J w.r.t. p and the compactness of the admissible set of parameters $\mathbb{P}$.

Putting together definitions 1 and 2 enables us to define for any initial configuration (r^0, η^0) the optimal sequence given by

$$v^{opt}(r^0,\eta^0) := \hat{v}(\hat{p}(r^0,\eta^0),r^0,\eta^0) = \left(v_1^{opt}(r^0,\eta^0) \dots v_q^{opt}(r^0,\eta^0)\right) \in \mathbb{R}^q \quad (15)$$

and since according to the discussion that follows definition 1, one can assume without loss of generality that $v_j^{opt}(r^0,\eta^0) \neq 0$, the following may be defined for a given pair (r^0,η^0) thanks to assumption 1.5:

$$\hat{\eta}(r^0,\eta^0) := \eta^1(\hat{p}(r^0,\eta^0),\eta^0)\ ; \quad \hat{\chi}(r^0,\eta^0) := f_1^{-1}\Big(v_1^{opt}(r^0,\eta^0)\Big) \qquad (16)$$

which is nothing but the first part of the OPTIMAL open loop trajectory on (η,χ) mentioned above when used in a receding horizon way. Finally, in order to monitor the selection of the controller's mode, the following functions are needed:

$$V_1(x,\chi^f) = \max\Big\{|\chi-\chi^f|, |\dot{\chi}|, |\xi|\Big\}\ ;\ V_2(x,\eta^f) := \max\Big\{|\eta-\eta^f|, |\xi|\Big\} \quad (17)$$

More precisely, when $V_1(x,\mathcal{X}^f)$ approaches 0, this means that the controller task at mode 1 is likely to be achieved. Similarly, when $V_2(x,\eta^f)$ approaches 0, this means that the controller task in mode 2 is not far from being achieved. Now, it goes without saying that one cannot wait for V_1 or V_2 to be exactly equal to zero since this never happens rigorously. That is the reason why a finite threshold $\varepsilon > 0$ is used in the definition of the switching rules given hereafter.

Using the above notations, the proposed state feedback algorithm can be expressed as follows

FEEDBACK ALGORITHM

parameters $\varepsilon > 0$ a small threshold. $v_{max} > 0$ sufficiently large value to be used in (12)
1) Compute $\chi^f = \hat{\chi}(r(t),\eta(t))$ and $\eta^f = \hat{\eta}(r(t),\eta(t))$ according to (16)
2) <u>mode 1</u>
Use the feedback K_1 with χ^f as computed in step **1) until** $V_1(x,\chi^f) \leq \varepsilon$.
3) <u>mode 2</u>
Use the feedback K_2 with η^f and χ^f as computed in step **1) until** $V_2(x,\eta_f) \leq \varepsilon$
4) go to step 1)

As a matter of fact, the feedback algorithm presented above describes a Finite State Machine that is depicted in Fig. 1. Associated to the proposed feedback algorithm, the following convergence result can be proved

Proposition 1. *Let $(t_k)_{k\geq 0}$ be the infinite sequence of instants at which the algorithm visits the updating state of* **Step 1** *(see Fig. 1). We have the following asymptotic property*

$$\lim_{\varepsilon\to 0}\left[\lim_{k\to\infty}\left\|\begin{pmatrix} r(t_k)\\ \eta(t_k)\\ \xi(t_k)\end{pmatrix}\right\|\right] = 0 \qquad (18)$$

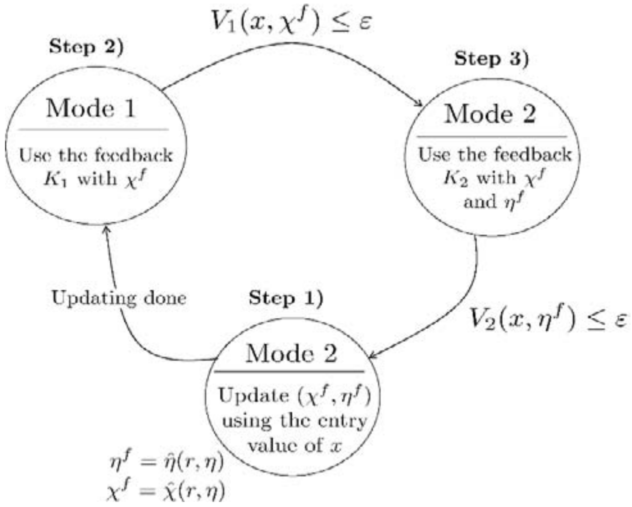

$V_1(x,\chi^f) \le \varepsilon$
Step 2)
Mode 1
Use the feedback K_1 with χ^f
Step 3)
Mode 2
Use the feedback K_2 with χ^f and η^f
$V_2(x,\eta^f) \le \varepsilon$
Updating done
Step 1)
Mode 2
Update (χ^f,η^f) using the entry value of x
$\eta^f = \hat\eta(r,\eta)$
$\chi^f = \hat\chi(r,\eta)$

Fig. 1. The Finite State Machine describing the proposed feedback algorithm

namely, by taking ε sufficiently small, the sequence $(r(t_k),\eta(t_k),\xi(t_k))$ may be steered as close as desired to 0*.*

SKETCH OF THE PROOF The regularity assumption on the functions appearing in (1)–(4) together with the rank condition and the uniform regularity of the matrix $A(p,\eta^0)$ used in (12) enables to prove the continuity of the optimal value function $\hat{J}(r^0,\eta^0) = J(\hat{p}(r^0,\eta^0),r^0,\eta^0)$ w.r.t its arguments r^0 and η^0. Using this property with the translatability assumption (8) on the parameterization being used enables to write a classical inequality in receding horizon analysis, namely

$$\hat{J}(r(t_{k+1}),\eta(t_{k+1})) - \hat{J}(r(t_k),\eta(t_k)) \le -\|r(t_k)\|^2 - \alpha\|\eta(t_k)\|^2 + O(\varepsilon) \quad (19)$$

where the final term regroups all the second order effects due to the use of finite stay time in each mode ($\varepsilon > 0$) and the use of finite horizon q in the equality constraint of (12). Finally, since by definition of the exit condition of **Step 3**, one necessarily has

$$|\xi(t_k)| \le \varepsilon \quad (20)$$

the result clearly follows from (19)–(20).

4 Application to the Constrained Stabilization of a Snakeboard

The snakeboard we are interested in is the mechanical system depicted in Figs. 3 and 2. It consists of two wheel-based platforms upon which a rider

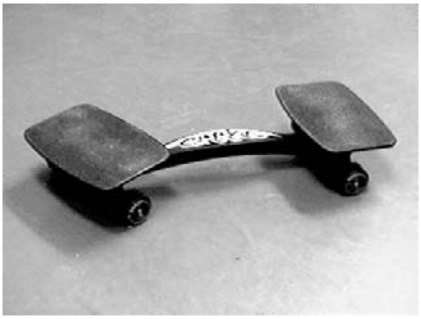


Fig. 2. The snakeboard "*without the rider*"

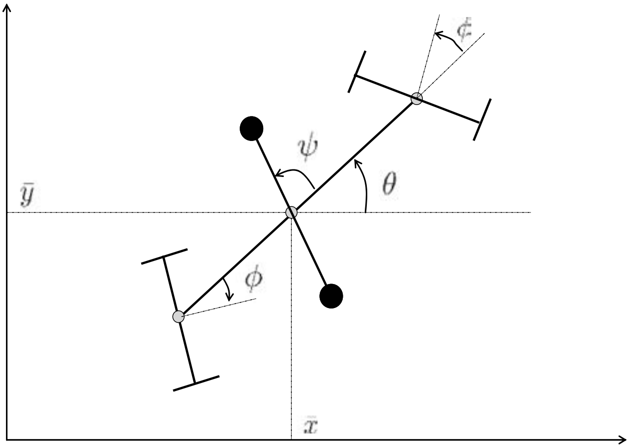


Fig. 3. Schematic view of the snakeboard's variables definition. Note that the front and the back steering angles are coupled to be the same

is to place each of his feet. These platforms are connected by a rigid coupler with hinges at each platform to allow rotation about the vertical axis. The snakeboard allows the rider to propel himself forward without having to make contact with the ground. This motion is obtained by using the conservation of angular momentum effect together with the nonholonomic constraints defined by the no-slipping condition at the wheels level.

This system was first outlined by [10] to be a particularly interesting nonholonomic system example that has no straightforward direct biological counterpart. In [10], a first dynamical model for the system was given and used to check some standard behaviors that correspond to some given oscillatory gaits. Furthermore, controllability analysis has been proposed showing that this system is locally controllable except at some singular configurations. The complete proof of controllability has been achieved in [11]. Since then, many works have been done to construct steering trajectories and control design. In

[12], a strategy leading to the design of steering trajectories has been proposed based on small amplitude, short duration and cyclic inputs. Based on such sinusoidal inputs, a controller based on average theory has also been proposed recently [13].

Much closer to our approach that does not invoke cyclic open loop parametrization is the work proposed in [14]. Indeed, in [14] the generation of path steering trajectories is based upon switching between two vector fields in order to obtain sub-curves starting from the initial configuration and ending at the desired one. The system is at rest at the switching instants. The constraints handling is obtained by means of time scaling. As long as the snakeboard is concerned, the difference between the approach proposed in [14] and the one proposed in the present paper lies in the following differences:

✓ The way the sub-curves defined above are derived in [14] deeply depends on the $2D$ nature of the snakeboard example (intersections of circles and/or straight lines, etc. are extensively used). In our approach, this is basically done, even in the general case by solving linear systems. In that sense, the approach proposed here seems to be more easily generalizable, as long as the assumed structure of the system holds.

✓ The choice of the steering trajectories in [14] is based on minimizing the corresponding number of switches whatever is the resulting transient spacial excursion. The reason behind this is that the system has to be at rest at switching instants. Therefore, having a high number of switches may lead to a slow motion. In our case, monitoring the number of switches can be directly and explicitly obtained by the number (q) of intermediate values in the parameterized sequence:

$$\left(\eta^k(p,\eta^0)\right)_{k=1}^{q}$$

while the additional d.o.f p are used to minimize the corresponding spacial excursion in the $x-y$ plane.

✓ The work in [14] concentrates basically on the open-loop steering strategy. It is not completely clear whether the resulting steering strategy can be used in a receding horizon manner in order to yield an effective feedback in presence of uncertainties or modelling errors. In other words, the "*translatability*" of the open-loop trajectories proposed in [14] has to be checked. If some other feedback strategy is to be used, this is still to be designed.

The model of the snakeboard used in the forthcoming developments is given by:

$$\begin{pmatrix}\dot{\bar{x}}\\ \dot{\bar{y}}\\ \dot{\theta}\end{pmatrix} = \begin{pmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}2l\cos^2\phi\\ 0\\ \sin(2\phi)\end{pmatrix}\xi \tag{21}$$

$$\dot{\xi} = \Big(-\frac{J_r}{2ml^2}\ddot{\psi} + \dot{\phi}\xi\Big)\cdot\tan\phi \tag{22}$$

$$u_2 = 2J_w\ddot{\phi} \tag{23}$$

$$u_1 = J_r\Big[(1 - \frac{J_r}{ml^2}\sin^2\phi)\ddot{\psi} + 2\dot{\phi}\xi\cos^2\phi\Big] \tag{24}$$

Note that the third line of equation (21) writes:

$$\xi = \frac{\dot{\theta}}{\sin(2\phi)} = \frac{\dot{\theta}}{2\sin(\phi)\cos(\phi)}$$

Using this in the equations enables us to show that te snakeboard equations (21)–(24) are of the standard form (1)–(4) provided that the following correspondances are used

$$r = \begin{pmatrix}\bar{x}\\ \bar{y}\end{pmatrix}\ ;\quad n_r = 2\ ;\quad \eta = \theta\ ;\quad n_\eta = 1\ ;\quad \chi = \phi\ ;\quad f_1(\chi) = \frac{2l}{\tan\phi}$$

$$g_1(\eta) = \begin{pmatrix}\cos\theta\\ \sin\theta\end{pmatrix}\ ;\quad f_2(\chi) = \sin(2\phi)\ ;\quad g_2(\xi) = \xi$$

with straightforward definitions of f_3 and f_4 that may be obtained by removing the auxiliary variable $\ddot{\psi}$. Note that in the classically used models [10, 14, 12], the configuration is given by $q = (\bar{x}, \bar{y}, \theta, \phi, \psi)$. Indeed, the variable ξ used in the above equations is an intermediate variable that can be removed since from (21), it comes that

$$\xi = \frac{\dot{\bar{x}}}{2l\cos\theta\cos^2\phi} \tag{25}$$

which is clearly a function of $(q, \dot{q})$. However, in the derivation of the control law, writing the equation in the form (21)–(24) is mandatory in order to fit the standard form (1)–(4). This is a classical feature: when using partially structural approach, the coordinate system plays a key role.

In order to use the feedback scheme defined in Sect. 3, the feedbacks K_1, K_2 and the parameterized sequence $\eta^k(p, \eta^0)$ are successively introduced.

4.1 Definition of the Feedback Laws $\mathbf{K_1}$ and $\mathbf{K_2}$

In both modes, the definition of u_2 is the same and is given by

$$u_2 := -2J_w\Big[\frac{2}{t_r}\dot{\phi} + \frac{1}{t_r^2}(\phi - \phi^f)\Big] \tag{26}$$

where t_r is a time scaling, namely, for $t_r = 1$, a pole placement (with identical double poles $= -1$) is assigned by imposing

$$\ddot{\phi} = -2\dot{\phi} - (\phi - \phi^f)$$

The parameter t_r is then used to meet the saturation requirement by taking:

$$t_r = \min_{\tau \in [0.01 \quad 10]} \tau \quad \text{such that} \quad 2J_w \cdot |\tfrac{2}{\tau}\dot{\phi} + \tfrac{1}{\tau^2}(\phi - \phi^f)| \leq u_2^{max} \tag{27}$$

As for u_1, it is mode dependent, namely

$$\begin{aligned} u_1|_{\text{mode } i} &:= J_r\Big[(1 - \frac{J_r}{2ml^2}\sin^2\phi)\psi^{pp}|_{\text{mode } i} + 2\dot{\phi}\xi\cos^2\phi\Big] \quad ; \\ &\times\ i \in \{1, 2\} \qquad (28) \\ \psi^{pp}|_{\text{mode } 1} &:= Sat_{\psi^{pp}_{min}(\phi,\dot{\phi},\xi)}^{\psi^{pp}_{max}(\phi,\dot{\phi},\xi)}\Big(10sign(\phi)\xi\Big) \qquad (29) \\ \psi^{pp}|_{\text{mode } 2} &:= \frac{2ml^2}{J_r}\left(\frac{\mu}{\tan\phi}\Big[-0.1|sin2\phi|\xi + (\xi - \xi_r(\mu))\Big] + \dot{\phi}\xi\right) \qquad (30) \end{aligned}$$

where

- $\xi_r(\mu)$ is a varying reference value for ξ that is given by

$$\xi_r := -0.1\mu \cdot sign(\sin 2\phi)(\theta - \theta^f) \tag{31}$$

- μ is an adaptive gain computed according to

$$\begin{aligned} &\mu = \max_{\nu \in [0 \quad 100]} \nu \quad \text{such that} \\ &\text{the r.h.s of (30) with } (\mu = \nu) \text{ is in} \quad [\psi^{pp}_{min}(\phi, \dot{\phi}, \xi), \psi^{pp}_{max}(\phi, \dot{\phi}, \xi)] \qquad (32) \end{aligned}$$

- $\psi^{pp}_{min}(\phi, \dot{\phi}, \xi)$ and $\psi^{pp}_{max}(\phi, \dot{\phi}, \xi)$ are the variable lower and upper bound on $\ddot{\psi}$ in (22) that are compatible with the saturation constraint on u_1 in (24), namely

$$\psi^{pp}_{min}(\phi, \dot{\phi}, \xi) := \min_{\tau \in \{-u_1^{max}, +u_1^{max}\}}\Big[\tau/J_r - 2\dot{\phi}\xi\cos^2\phi\Big]/(1 - \frac{J_r}{ml^2}\sin^2\phi) \tag{33}$$

$$\psi^{pp}_{max}(\phi, \dot{\phi}, \xi) := \max_{\tau \in \{-u_1^{max}, +u_1^{max}\}}\Big[\tau/J_r - 2\dot{\phi}\xi\cos^2\phi\Big]/(1 - \frac{J_r}{ml^2}\sin^2\phi) \tag{34}$$

In order to understand how (28) (with $i = 1$) asymptotically stabilizes ξ while meeting the saturation constraint, one may analyse what happens when ϕ converges to 0 under the action of u_2. Indeed, this asymptotically leads to an admissible region for $\ddot{\psi}$ that contains an open neighborhood of 0 [see equations (33)–(34)]. This with (22) in which $\dot{\phi} = 0$ is injected clearly shows that implementing (28) yields an asymptotically stable behavior for ξ.

As for the design of $u_1|_{\text{mode } 2}$, it is based on a sliding mode like approach in which the manifold $S = \xi - \xi_r$ is stabilized where ξ_r is the control that stabilizes θ around its desired value θ^f [see (31)]. Since ξ_r asymptotically tends to 0 with the error $\theta - \theta^f$, ξ does the same. Again, variable adaptive gain μ [see equation (32)] is used in order to meet the saturation constraints on u_1. To end this presentation of the feedback laws K_1 and K_2, it is worth noting that the constants 0.01, 10 and 0.1 that appear in equations (27), (29), (30) and (31) are used in order to avoid very high gains near the desired targets and to obtain compatible response times in the back-stepping design approach. They might have been left as design parameters. The choice fixed here aims to avoid having too many parameters to tune.

4.2 Definition of the Parameterized Sequences $(\eta^k(p, \eta^0))_{k \geq 0}$

Recall that for the snakeboard example, $\eta = \theta$. Consider the following parameterized trajectory

$$\Theta(t, p, \theta^0) := 2p_1\pi + (\theta^0 - 2p_1\pi)e^{-\lambda_\theta t} + p_2 e^{-\lambda_\theta t}\Big(1 - e^{-\lambda_\theta t}\Big) \quad ; \quad \lambda_\theta > 0 \quad (35)$$

where $p = (p_1, p_2)$ belongs to the compact subset $\mathbb{P} \subset \mathbb{R}^2$ defined by

$$\mathbb{P} := \{-1, 0, +1\} \times [-p_2^{max}, +p_2^{max}] \; ; \quad p_2^{max} > 0 \quad (36)$$

Note that for all $p \in \mathbb{P}$, $\Theta(0, p, \theta^0) = \theta^0$ while $\lim_{t\to\infty} \Theta(t, p, \theta^0) = 0 \ (mod\ 2\pi)$. Note also that the use of $p_2 \neq 0$ and $p_1 = 0$ enables non constant trajectories with identical boundary conditions $\theta^0 = \theta^f$ to be generated. This is crucial to obtain "*good*" solutions in some singular situations like for instance the one given by $\bar{x}^0 = 0$, $0 \neq \bar{y}^0 \approx 0$ and $\theta^0 = 0$. Indeed, without the parameter p_2, whatever small is $\bar{y}^0 \neq 0$, an entire rotation would be necessary to steer the snakeboard to the desired position.

Given an initial value θ^0 and some parameter vector $p \in \mathbb{P}$, the generation of the parameterized trajectory is done using the following three steps:

1. Choose some sampling period $\delta > 0$, take an integer $q \gg \dfrac{1}{\delta\lambda_\theta}$. Generate the sequence

$$(\theta^k(p, \theta^0))_{k=0}^q \quad \text{where} \quad \theta^k(p, \theta^0) := \Theta(j\delta, p, \theta^0)$$

2. Solve the corresponding LP problem (12) to get the optimal sequence

$$\hat{v}(p, \bar{x}^0, \bar{y}^0, \theta^0)$$

   Remove all vanishing (or too small) components, if any, by introducing intermediate terms in the sequence $\theta^k(p, \theta^0)$ as explained above (In fact, this has never been necessary in our experimentation).

3. Reduce the size of the resulting sequence $\theta^k(p,\theta^0)$ by keeping only the last value of all sequences corresponding to the same value of $\hat{v}(p,\bar{x}^0,\bar{y}^0,\theta^0)$. As an example, if the sequence $\hat{v}(p,\bar{x}^0,\bar{y}^0,\theta^0)$ takes the following form:

$$\hat{v}(p,\bar{x}^0,\bar{y}^0,\theta^0) = \begin{pmatrix} v_1 & v_1 & v_1 & v_2 & v_2 & v_3 & v_3 & v_3 \end{pmatrix}^T$$

then the reduced sequence $\theta^k(p,\theta^0)$ which is finally retained is the following one

$$\begin{pmatrix} \theta^3(p,\theta^0) & \theta^5(p,\theta^0) & \theta^8(p,\theta^0) \end{pmatrix}$$

this enables useless waste of time asking successive values of $\theta^k(p,\theta^0)$ to be successively approached with almost zero-velocity with the same value of ϕ.

4.3 Checking the Remaining Assumptions

The rank condition (6) is obviously satisfied provided that the sampling period $\delta > 0$ invoked in Sect. 4.2 is taken sufficiently small thanks to the properties of the trigonometric functions. The same can be said about the condition expressed in assumption 1.4. The translatability property naturally follows from the properties of the exponential functions used in the definition of the parameterized trajectory (35). Finally, the conditions of assumption 1.5 are clearly satisfied with $\mathcal{D} =]-\pi/2, +\pi/2[$.

5 Simulation Results

In this section, some simulations are proposed in order to show the efficiency of the proposed feedback algorithm. The numerical values used in the simulations as well as a recall on where each of them appears in the preceding sections is depicted in Table 1. The proposed simulations aim to underline the following features

1. The ability to explicitly handle the saturation constraints on the control input. This may be observed by comparing Figs. 4 and 5 where the same initial conditions are simulated for two different saturation levels, namely $u_1^{max} = u_2^{max} = 8$ (Fig. 4) and $u_1^{max} = u_2^{max} = 4$ (Fig. 5).

Table 1. Numerical values of the system's and the controller's parameters

Parameter	Appearing in	Value	Parameter	Appearing in	Value
l	(21)	0.5	J_r	(22)	0.72
ml^2	(28)	0.24	J_w	(23)	0.13
λ_θ	(35)	0.1	v_{max}	(12)	2
ε	Fig. 1	0.01	p_2^{max}	(36)	10

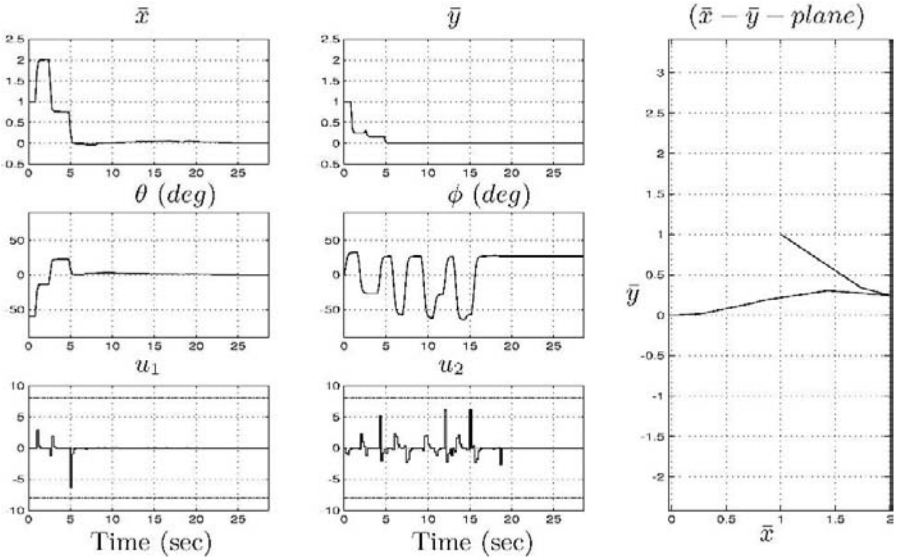


Fig. 4. Closed loop behavior. Initial condition $\bar{x}(0) = \bar{y}(0) = 1$, $\theta(0) = -\pi/4$ and $\xi(0) = 0$. Saturation levels $u_1^{max} = u_2^{max} = 8$. This figure is to be compared with Fig. 5 in order to appreciate the saturation constraints handling

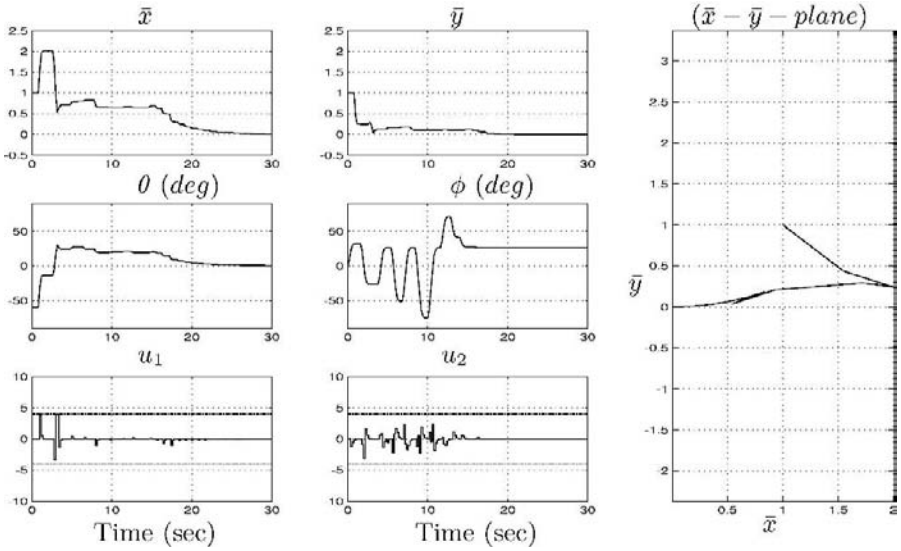


Fig. 5. Closed loop behavior. Initial condition $\bar{x}(0) = \bar{y}(0) = 1$, $\theta(0) = -\pi/4$ and $\xi(0) = 0$. Saturation levels $u_1^{max} = u_2^{max} = 4$. This figure is to be compared with Fig. 4 in order to appreciate the saturation constraints handling

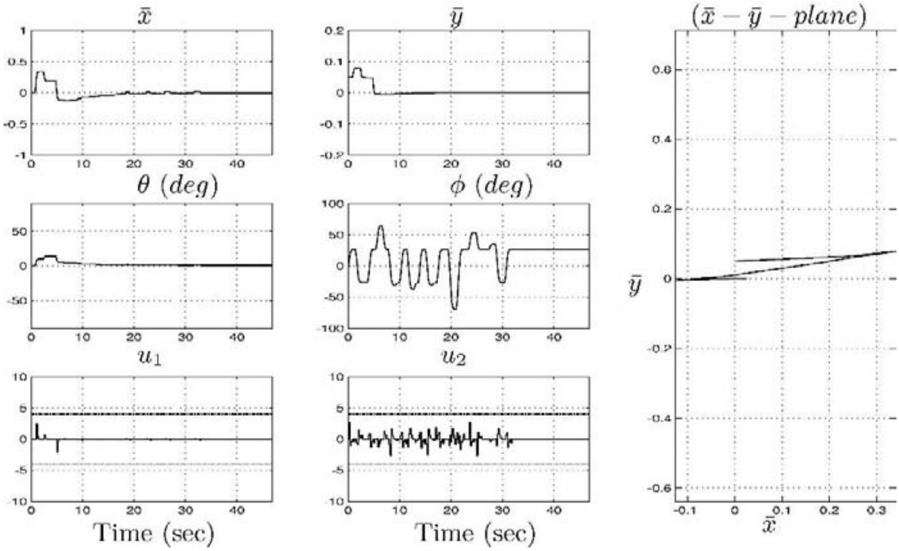


Fig. 6. Closed loop behavior. Initial condition $\bar{x}(0) = 0$, $\bar{y}(0) = 0.05$, $\theta(0) = 0$ and $\xi(0) = 0$. Saturation levels $u_1^{max} = u_2^{max} = 4$. Note how the parametrization of the trajectories avoids the need for a whole rotation even in this rather singular situation

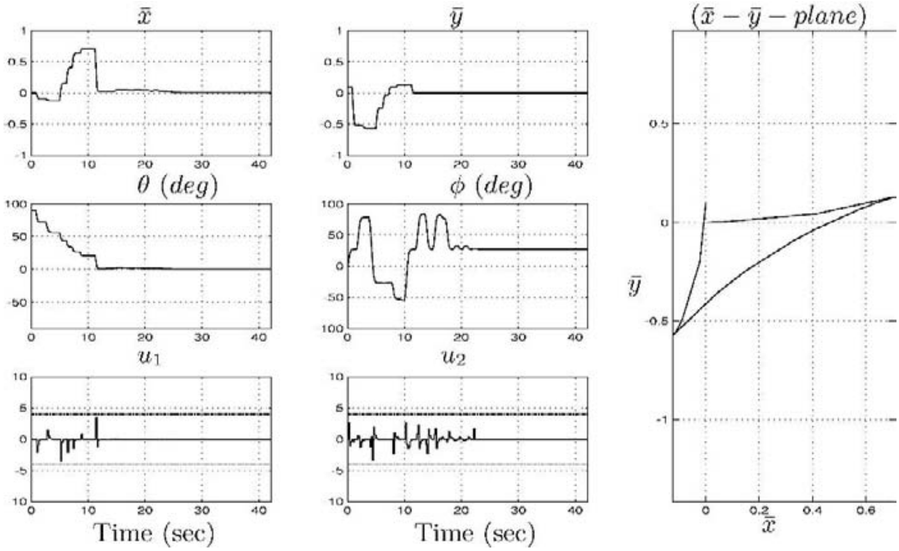


Fig. 7. Closed loop behavior. Initial condition $\bar{x}(0) = 0$, $\bar{y}(0) = 0.1$, $\theta(0) = \pi/2$ and $\xi(0) = 0$. Saturation levels $u_1^{max} = u_2^{max} = 4$

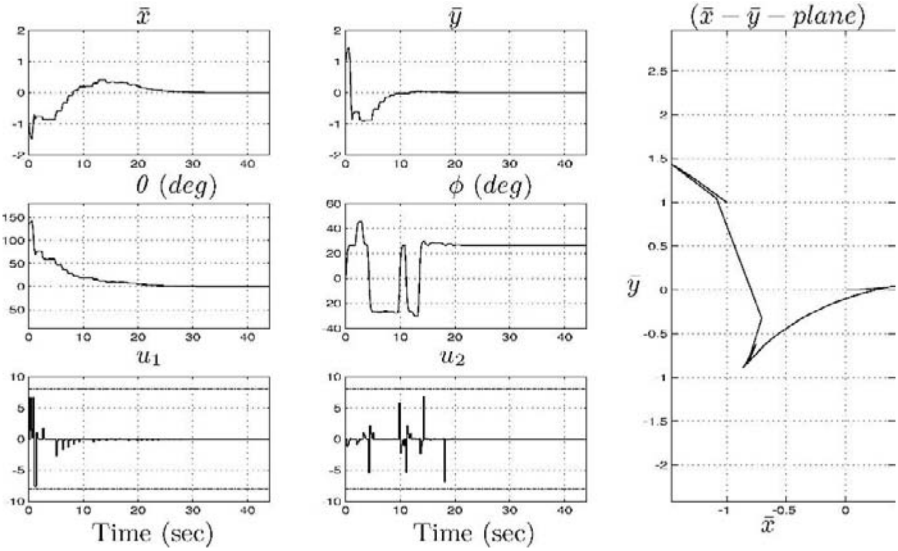


Fig. 8. Closed loop behavior. Initial condition $\bar{x}(0) = -1$, $\bar{y}(0) = 1$, $\theta(0) = 3\pi/4$ and $\xi(0) = 1$. Saturation levels $u_1^{max} = u_2^{max} = 8$. This simulation is done under nominal model without uncertainties. The result is to be compared to that of Fig. 9 where model uncertainties are introduced

2. The ability to realize rather "*economic*" trajectories when starting from some almost singular situations (like the ones shown on Figs. 6 and 7) avoiding whole rotations to be used.
3. Finally, Figs. 8 and 9 show the behavior of the closed loop under the nominal model (Fig. 8) and under model uncertainties (Fig. 9). The uncertainties are introduced in equations (22) and (23) as follows

$$\dot{\xi} = \Big(-\frac{J_r}{2ml^2}(1+\delta_1)\ddot{\psi} + \dot{\phi}\xi\Big) \cdot \tan\phi + \delta_2 \tag{37}$$

$$u_2 = 2J_w(1+\delta_3)\ddot{\phi}; \quad \delta_1 = -0.1 \quad ; \quad \delta_2 = 0.05; \quad \delta_3 = -0.1 \tag{38}$$

Namely, δ_1 and δ_3 stand for relative error on the values of the physical parameters while δ_2 stands for persistant external disturbance such as wind related drift term.

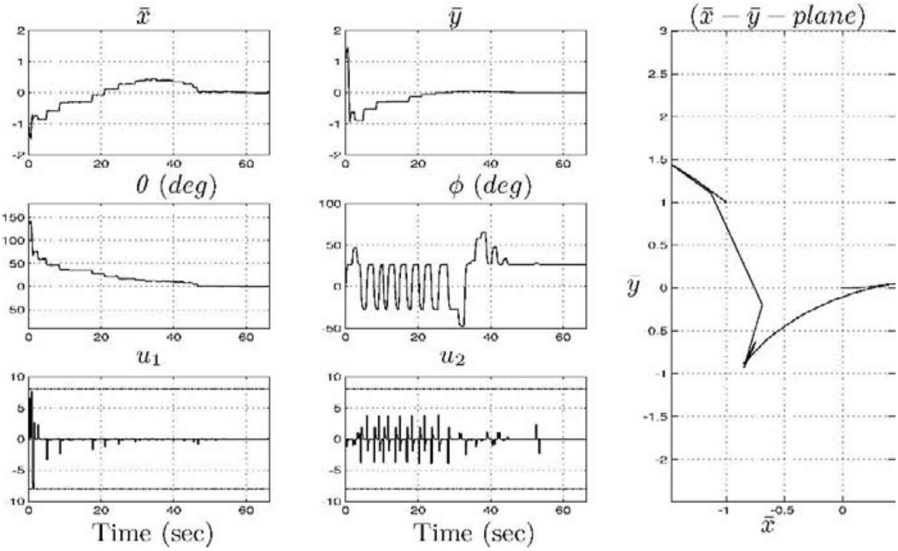


Fig. 9. Closed loop behavior. Initial condition $\bar{x}(0) = -1$, $\bar{y}(0) = 1$, $\theta(0) = 3\pi/4$ and $\xi(0) = 1$. Saturation levels $u_1^{max} = u_2^{max} = 8$. The uncertainties given by (37)–(38) are used to test the robustness of the proposed feedback algorithm

6 Conclusion & Discussion

It has been shown that by exploiting the particular structure of a class of nonholonomic system, it is possible to derive an efficient steering procedure that can be used in a receding-horizon scheme to yield a stabilizing state feedback. By doing so, a complex dynamic problem is transformed into a rather simple discrete problem that can be solved by linear programming tools. The solution is successfully applied to the snakeboard constrained stabilization problem.

The main drawback of the proposed approach is the constraint of almost stopping motion in-between each of the two control modes being used. This constraint can be practically avoided by using a state dependent switching parameter ε, namely $\varepsilon(x)$ that would be large when x is far from the desired state and small in its neighborhood.

A Appendix

Lemma 1. *Let $A \in \mathbb{R}^n$ be a regular matrix. Define a compact subset $\mathcal{C} \subset \mathbb{R}^n$ that contains a neighborhood of the origin. Let $\mathcal{C}_p \subset \mathcal{C}$ be the subset defined by*

$$\mathcal{C}_p := \Big\{ v \in \mathcal{C} \quad | \quad v_i \neq 0 \quad \forall i \in \{1, \dots, n\} \Big\} \tag{39}$$

then there always exist $v \in \mathcal{C}_p$ such that $Av \in \mathcal{C}_p$ ♠

Proof Let $\mathcal{O} \subset \mathcal{C}$ be a sufficiently small neighborhood of the origin such that $A^{-1}\mathcal{O}$ is also a neighborhood of the origin that is contained in $\mathcal{C}$. The set $A^{-1}(\mathcal{O} \cap \mathcal{C}_p)$ is clearly an open neighborhood of the origin that contains an element in $\mathcal{C}_p$. Let v be such an element, v clearly satisfies the requirements of the lemma. □

References

[1] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, pp. 789–814, 2000.

[2] F. Allgöwer, T. Badgwell, J. Qin, J. Rawlings, and S. Wright, "Nonlinear predictive control and moving horizon estimation. an introductory overview," in *Advances in control Highlights of ECC'99*, P. M. Frank, Ed., 1999, pp. 81–93.

[3] M. Alamir and G. Bornard, "Globally stabilising discontinuous feedback law for a class of nonlinear systems." *Eur. J. Control*, vol. 2, no. 1, pp. 44–50, 1996.

[4] M. Alamir and H. Khennouf, "Discontinuous receding horizon control-based stabilizing feedback for nonholonomic systems in power form," in *Proceedings of the IEEE Conference on Decision and Control*, New Orleans, 1995.

[5] M. Alamir and F. Boyer, "Fast Generation of Attractive Trajectories for an Under-Actuated Satellite : Application to Feedback Control Design," *Journal of Optimization in Engineering*, vol. 4, pp. 215–230, 2003.

[6] N. Marchand and M. Alamir, "Numerical Stabilisation of a Rigid Spacecraft with two Actuators," *Journal of Dynamic Systems, Measurement and Control*, vol. 125, pp. 489–491, 2003.

[7] M. Alamir and N. Marchand, "Numerical stabilization of nonlinear systems -exact theory and approximate numerical implementation," *European Journal of Control*, vol. 5, pp. 87–97, 1999.

[8] M. Alamir, " New path generation based receding horizon formulation for constrained stabilisation of nonlinear systems," *Automatica*, vol. 40, no. 4, pp. 647–652, 2004.

[9] M. Alamir and N. Marchand, "Constrained minimum-time oriented feedback control for the stabilization of nonhoonomic systems in chained form," *Journal of Optimization Theory and Applications*, vol. 118, no. 2, pp. 229–244, 2003.

[10] A. D. Lewis, J. P. Ostrowski, J. W. Burdick, and R. M. Murray, "Nonholonomic mechanics and locomotion: the snakeboard example," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, San Diega, USA, pp. 2391–2400.

[11] J. Ostrowski and J. Burdick, "Controllability tests for mechanical systems with constraints and symmerties," *Journal Applied Mathematics and Computer Science*, vol. 7, no. 2, pp. 101–127, 1997, proof of local controllability.

[12] J. Ostrowski, "Steering for a class of dynamic nonholonomic systems," *IEEE Transaction on Automatic Control*, vol. 45, no. 8, pp. 1492–1498, 2000.

[13] P. Vela and J. W. Burdick, "Control of biomimetic locomotion via averaging theory," in *Proceedings of IEEE Intl. Conf. on Robotics and Automation, ICRA2003.*

[14] S. Iannitti and K. M. Lynch, "Exact minimum control switch motion planning for a snakeboard," in *Proceedings of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IROS2003*, Las Vegas, USA.

Recent Advances on the Algorithmic Optimization of Robot Motion

J.E. Bobrow, F.C. Park, and A. Sideris

Department of Mechanical and Aerospace Engineering, University of California, Irvine, Irvine, CA 92697
{jebobrow, asideris}@uci.edu

Summary. An important technique for computing motions for robot systems is to conduct a numerical search for a trajectory that minimizes a physical criteria like energy, control effort, jerk, or time. In this paper, we provide example solutions of these types of optimal control problems, and develop a framework to solve these problems reliably. Our approach uses an efficient solver for both inverse and forward dynamics along with the sensitivity of these quantities used to compute gradients, and a reliable optimal control solver. We give an overview of our algorithms for these elements in this paper. The optimal control solver has been the primary focus of our recent work. This algorithm creates optimal motions in a numerically stable and efficient manner. Similar to sequential quadratic programming for solving finite-dimensional optimization problems, our approach solves the infinite-dimensional problem using a sequence of linear-quadratic optimal control subproblems. Each subproblem is solved efficiently and reliably using the Riccati differential equation.

1 Introduction

For many biological systems, it has long been observed that motion generation can likely be the result of a minimization process. The objective function used has been characterized by a physical criteria like energy, control effort, jerk, or time. Unfortunately, to date the algorithms that generate such optimal motions have been successfully used on only the simplest of robots. The need for such an algorithm is increasing dramatically since many new walking, crawling, hopping machines, rehabilitation devices, and free-flying air and space systems are currently under development. All of these devices will benefit from a numerically stable and efficient algorithm that produces optimal movements for them.

We are interested in obtaining solutions to optimal control problems for systems of the form

$$\dot{x} = f(x(t), u(t)) \ , \tag{1}$$

where $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n \in C^1$ (continuously differentiable) and $x(0) = x_o$. We assume that the optimal control cost functional has the form

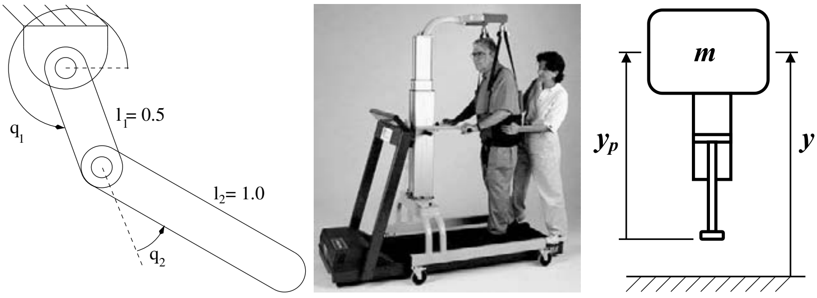


Fig. 1. The *left-hand system* was used for Case 1, a fully actuated robot. Case 2 is the same system, but with the base joint unactuated. The *center system* represents Case 3, which is an application to human step rehabilitation. The *right system* is a hopping machine for Case 4

$$\underset{u(t)}{\text{Minimize}} \; J(u(t)) = \phi(x(t_f)) + \int_0^{t_f} L(x(t), u(t), t)\; dt \;, \tag{2}$$

subject to (1) with $\phi : \mathbb{R}^n \rightarrow \mathbb{R} \in C^1$ and $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \rightarrow \mathbb{R} \in C^1$. Although the Maximum Principle [3] provides the optimality conditions for the solution to (2), it is not suitable for numerical computation. Because of the importance of solving these problems, many numerical algorithms and commercial software packages have been developed to solve them since the 1960's [1]. Most of the existing algorithms do not have adequate numerical stability properties and are too slow computationally to solve optimal control problems for current multibody systems. As a means to discuss the numerical features of algorithms, we provide example solutions in four case studies. These examples demonstrate the strength and limitations of current numerical algorithms.

Figure 1 shows model systems used for four case studies in this paper. Case 1 is a minimum effort control of a fully actuated robot. We have found that with care in the choice of basis functions, a parameterization of the joint motion can be used along with a static optimization method to adequately solve this problem. In addition, this approach easily allows one to handle the case when some of the initial states are free to vary, as was done in [10]. Case 2 is an underactuated robot. We have found that even with exact gradients of the dynamics, our parameterization of the motion had numerical problems from round-off errors during the simulations. Case 3 is an application to human leg step rehabilitation [21]. We experienced even more numerical problems for this problem due to the added ground constraint. Finally, Case 4 is a simplified gas actuated hopping machine. We found it difficult to achieve stable convergence with existing methods for this case. The cause was numerical integration errors introduced at the sudden change in the dynamics between the stance phase and the flight phase, and the fact that the times for the switch from stance to flight were not known apriori. The approach developed in this paper (see also [14]) efficiently solves Case 4 , and we feel that it has great potential for application to general optimal control problems for robot systems.

1.1 First Order Necessary Conditions for the Solution of the Optimal Control Problem

In order to discuss the solution to Cases 1–4, we first briefly summarize the first order necessary conditions for the optimal control of a general nonlinear system (see [3] for more details). First define the Hamiltonian as

$$H(x,u,\lambda,t) \equiv L(x,u,t) + \lambda^T f(x,u) \; , \tag{3}$$

where L and f were defined in (1) and (2). Then in (3), $\lambda(t)$ is chosen to satisfy the costate or adjoint equations

$$\dot{\lambda} = -H_x(x_o(t), u_o(t), \lambda(t), t) \; , \tag{4}$$

where H_x and H_u (used below) denote partial derivatives of H with respect to x and u respectively, and the boundary conditions are

$$\lambda(t_f) = \phi_x^T(x_o(t_f)).$$

Let $u_o(t)$ be a nominal control, $x_o(t)$ and $\lambda_o(t)$ be the corresponding solutions to (1) and (4), respectively. For general problems, the first order necessary conditions for a local minimum of J require that $H(x_o(t), u_o(t), \lambda(t), t)$ be minimized with respect to $u_o(t)$ subject to any constraints on it. For unconstrained controls u, the condition on H is

$$H_u(x_o(t), u_o(t), \lambda(t), t) = 0.$$

Note that both H_x and H_u require differentiation of the state equations (1) with respect to x and u and evaluation of these derivatives along the solution $(x_o(t), u_o(t))$. For multibody dynamic systems with more than a few degrees of freedom, the derivatives are generally not available due to the complexity of the equations of motion. However, in [15], the sensitivity algorithms based on matrix exponentials are developed specifically for this purpose. A brief introduction to that work is presented next.

1.2 Geometric Tools for Multibody Systems Analysis

To represent robot systems and their dynamics, we use a set of analytical tools for multibody systems analysis based on the mathematics of Lie groups and Lie algebras [11, 13]. In the traditional formulation, a rigid motion can be represented with the Denavit-Hartenberg parameters as a 4 × 4 homogeneous transformation $T(\theta, d) \in \mathrm{SE}(3)$, where θ is the rotation about the z-axis and

d is the translation along it. For a prismatic joint, d varies while θ is held constant. For a revolute joint, θ varies while d is held constant. With the geometric formulation, for either type of joint the transformation has the form

$$T(\theta, d) = e^{Ax} M \ ,$$

where $x = \theta$ for a revolute joint or $x = d$ for a prismatic joint, A contains the joint axis or direction, and M is a constant ($M = T(0, d)$ for a revolute joint, $M = T(\theta, 0)$ for a prismatic joint.) This exponential mapping and its inverse have explicit formulas: exp : $se(3) \rightarrow SE(3)$ and its inverse log : $SE(3) \rightarrow se(3)$ [13]; here se(3) denotes the Lie algebra of SE(3). Although SE(3) is not a vector space, se(3) is: the log formula provides a set of *canonical coordinates* for representing neighborhoods of SE(3) as open sets in a vector space.

The derivative of the exponential map with respect to the joint displacement x is just $\frac{dT}{dx} = Ae^{Ax}M$. In the coding of multibody dynamics algorithms, the exponential is the lowest level primitive required for all computations. One never needs to deal with sine and cosine terms or with making a distinction for each joint type.

The use of matrix exponentials to represent the link to link transformations for robot systems allows one to clarify the kinematic and dynamic equations. In the case of open chains containing prismatic or revolute joints, the forward kinematics can be written as a product of matrix exponentials [2]. Specifically, given a choice of inertial and tool reference frames, and a zero position for the mechanism, the forward kinematics can be written uniquely as

$$T_{0n}(q_1, \ldots, q_n) = e^{A_1 q_1} \cdots e^{A_n q_n}$$

where $q_1, \ldots, q_n$ are joint variables, and $A_1, \ldots, A_n \in$ se(3). The kinematics of closed chains can be obtained by further adding a set of algebraic constraints.

In order to determine optimal motions for the multibody systems of interest, a complete dynamic model is needed. In [13] a Lie group formulation of the dynamics has been developed, in which closed-form expressions for the inertia matrix and Coriolis terms are available. Using this representation, the forward and inverse dynamics can also be computed efficiently with $O(n)$ recursive algorithms. The inverse dynamics algorithm is shown in Fig. 2. In this algorithm, $V_i \in$ se(3) is the linear and angular velocity of link i, W is the applied force and moment, J is a 6×6 matrix of mass and inertia, S_i is the joint screw, and Ad and ad are standard operators from differential geometry [11]. A useful computational feature of this algorithm is that no distinction needs to be made for revolute or prismatic joints. In [15], this algorithm was extended to forward and inverse dynamics of partially actuated systems, and to produce the derivatives needed for many optimal control solvers, as discussed in the previous section.

Given the ability to compute the dynamics and derivatives of relatively complex systems, we now discuss some representative examples. The following

- **Initialization**

$$V_0 = \dot{V}_0 = \mathcal{W}_{n+1} = 0$$

- **Forward recursion: for $i = 1$ to n do**

$$\begin{aligned} T_{i-1,i} &= M_i e^{S_i q_i} \\ V_i &= \mathrm{Ad}_{T^{-1}_{i-1,i}}(V_{i-1}) + S_i \dot{q}_i \\ \dot{V}_i &= S_i \ddot{q}_i + \mathrm{Ad}_{T^{-1}_{i-1,i}}(\dot{V}_{i-1}) + \left[\mathrm{Ad}_{T^{-1}_{i-1,i}}(V_{i-1}), S_i \dot{q}_i\right] \end{aligned}$$

- **Backward recursion: for $i = n$ to 1 do**

$$\begin{aligned} \mathcal{W}_i &= \mathrm{Ad}^*_{T^{-1}_{i,i+1}}(\mathcal{W}_{i+1}) + J_i \dot{V}_i - \mathrm{ad}^*_{V_i}(J_i V_i) \\ \tau_i &= S_i^T \mathcal{W}_i \end{aligned}$$

Fig. 2. The POE recursive Newton-Euler inverse dynamics algorithm

case studies demonstrate both successes and difficulties that we have encountered in applying the optimality conditions of Sect. 1.1 to multibody systems problems.

2 Some Representative Case Studies

2.1 Case 1: Fully Actuated Robot

Consider the case of finding the minimum effort control which moves the two link planar robot shown in Fig. 1 from an outstretched horizontal position to a vertical position. Assume that both joints of the arm be actuated, and let the cost function be:

$$J = c_1 ||q(t_f) - q_d||^2 + c_2 ||\dot{q}(t_f)||^2 + \frac{1}{2} \int_0^{t_f} ||u||^2 dt \ , \tag{5}$$

where $q \in \mathbb{R}^2$ are the joint angles, $q_d = [\frac{\pi}{2}, 0]^T$, are the desired final joint positions in radians, $u \in \mathbb{R}^2$ are the corresponding joint torques, and $c_1 = c_2 = 100$ reflects the desire to reach the final vertical configuration with little error.

We used several approaches to solve this optimal control problem. The most straight forward approach, called the "shooting method," is to cleverly find the initial costate, $\lambda(0)$, such that when the state and costate equations are integrated forward from $t = 0$ to $t = t_f$ with $H_u = 0$, the proper final condition on the costate is satisfied ($\lambda(t_f) = \phi_x^T(x_o(t_f))$. Unfortunately, because the costate equations are not stable and highly sensitive to the initial conditions [3], the shooting method failed when applied to this problem. A second, more successful, approach used by several researchers [5, 4, 10, 12] in robotics is to approximate the motion of the joints using basis functions such as splines or a truncated Fourier Series. For instance, we used quintic B-spline polynomials with 12 uniformly spaced knot intervals to parameterize our solution as $q = q(t, P)$ with $P \in \mathbb{R}^2 \times \mathbb{R}^{12}$ being the amplitude of the spline basis functions. For any choice of P, we can compute the required control $u(t)$ by differentiating $q(t, P)$ with respect to time to obtain $\dot{q}$ and $\ddot{q}$ and, evaluating the equations of motion

$$M(q)\ddot{q} + h(\dot{q}, q) = u \; . \tag{6}$$

In order to use this "direct approach," we guessed an initial motion that kept the second link aligned with the first with $q_2(t) = 0$, and moved the first link smoothly from $q_1(0) = 0$ to $q_1(t_f) = \pi/2$, with $t_f = 2$ seconds. We then computed $J(u(P))$ in (5) and its gradient ∇J_P using adaptive Simpson quadrature. In this case, the integrand is known explicitly throughout the integral since all the terms in (6) are known explicitly in terms of P from the joint angles $q = q(t, P)$. Given $J(P)$ and its gradient, we could easily minimize it over P using Matlab's BFGS [9] algorithm in the function "fminunc."

Figure 3 shows the locally optimal solution found to this problem using the parameter optimization approach mentioned above. The frames are spaced at equal intervals in time, with $t_f = 2$ seconds. At first the robot allows gravity to take over and it swings down while folding up the second link. It then swings the first joint into the upward posture. A small pumping motion is applied to the second link in order to move it into the vertical posture. The initial value of the effort term in the cost function was 73.6 and the final value was 9.9. The computation time for this problem was about 2 minutes on a PIII-800 PC.

We have used this basic approach with our dynamics tools to solve a weight-lifting problem for a much more complex Puma 762 robot in [20], where we tripled the payload above the manufacturers specifications. Even though

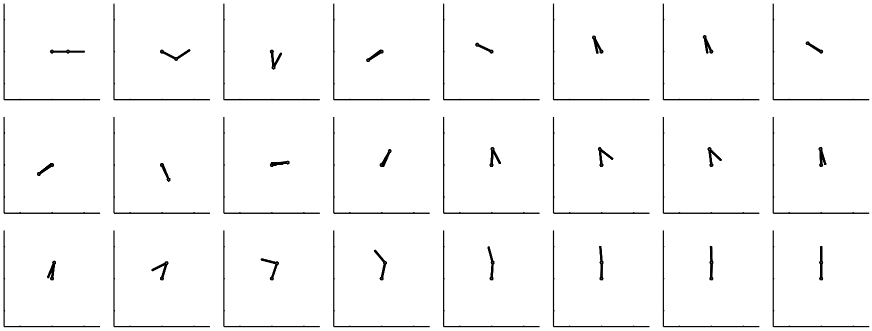


Fig. 3. Final path for fully-actuated planar 2R problem

the solution to the above problem was fairly stable numerically, we still needed to choose an appropriate set of basis functions in order to approximate the numerical solution. In this paper, we develop an approach that does not use any basis functions to approximate the solution.

2.2 Case 2: Underactuated Robot

The optimal control problem becomes less numerically stable and more difficult to solve when the system is underactuated. For instance, suppose the motor attached to the first joint of the above 2R robot is disconnected. The system then only has a motor at its elbow and is often called the Acrobot, which has been studied by Spong [17] and others. Consider the swing-up motion problem where the system starts from a hanging downward posture and the optimal control problem is to find an open-loop control torque for the elbow, if one exists, that drives the system to the upward posture of the previous example. This case is much more challenging than the previous one because we can no longer use (6) to compute $u(P)$ because the system is underactuated.

Choose the same objective function as (5), except the control u is now a scalar. One way to approach the problem is approximate the control with a set of basis functions and integrate the 4 state equations in order to evaluate (5). Any gradient-based numerical optimization will need both the value of $J(P)$ and its gradient $\nabla_p J$. Assuming that the state, costate, and boundary conditions are satisfied, the required derivative is

$$\frac{dJ}{dp_i} = \int_0^{t_f} H_u(x_o(t), u_o(t), \lambda(t), t) \frac{du}{dp_i} \, dt \,. \tag{7}$$

This derivative is valid for any $(x_o(t), u_o(t))$ even if they are not optimal [3]. Then, in order to evaluate the objective function and its gradient for use in any gradient-based nonlinear optimization algorithm, the following steps must be performed:

- Select a set of basis functions and parameters to define $u_o(t, P)$.
- Integrate the differential equations (1) of motion from 0 to t_f to obtain $x_o(t)$ and $J(P)$.
- Evaluate the costate (4) boundary conditions $\lambda(t_f) = \phi_x^T(x_o(t_f))$ and integrate the costate equations backwards in time from t_f to 0 to obtain $\lambda(t)$.
- Evaluate the gradient of J using (7).

We used this method to solve the Acrobot swing-up problem with one modification. That is, instead of viewing the control u_2 as the torque at joint 2, we defined the motion of joint q_2 with our spline functions and considered that to be the control in (1). In doing so, the actual joint torque $u_2(t)$ can be computed algebraically from (6) assuming that $(q, \dot{q})$ are known. The advantage of doing this is that the state reduces to 2 dimensions in this case

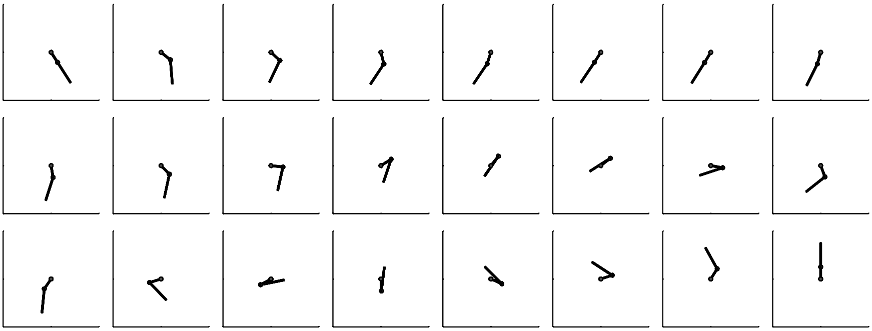


Fig. 4. Optimal swing up motion for an Acrobot with $q_1(0) = -1.0$

$x = [q_1, \dot{q}_1]^T$ since the motion of the second joint is known from the parameterization.

Note that it is not apparent what, if any, elbow motion will drive the system to the desired final configuration. Our initial guess for the elbow motion was very poor. We did not move the elbow joint at all during the motion, and let the system move like a rigid pendulum would with an initial condition $q_1(0) = -1.0, \dot{q}(0) = q_2(0) = \dot{q}_2(0) = 0.0$. Figure 4 shows the final motion obtained using Matlab's nonlinear parameter optimization with gradients computed as described above. The motion produced is similar to those proposed by Spong [17], in which the lower link pumps energy into the system and this energy causes the first link to move into the vertical position. In addition to this example, we have used this basic approach to solve for much more complex optimal high-dive motions for a human-like diver in [16].

When we computed the above solution to the underactuated Acrobot, we did not expect numerical difficulties, since we had the exact gradient of the objective function and the optimization algorithm has well-established convergence properties for this case [9]. However, we did encounter some numerical problems and had to adjust some of the tolerances in the optimizer in order to achieve convergence, and the computation time, even in the best of cases (about 5 minutes on a PIII-800 PC), was much longer than in the previous example. The problem was that the round-off errors encountered during the numerical solution of (1), (4), and (7) lead to large relative errors in the gradient when the algorithm is near convergence. The algorithm developed in this paper alleviates these difficulties.

2.3 Case 3: Underactuated Systems with Contact Constraints-Human Step Training Example

One important application of our proposed algorithm is the generation of optimal inputs for the robotic rehabilitation of paralyzed individuals [22]. In

[21] we examined a method to control the stepping motion of a paralyzed person suspended over a treadmill (see Fig. 1) using a robot attached to the pelvis. Leg swing motion was created by moving the pelvis without contact with the legs. The problem is formulated as an optimal control problem for an underactuated articulated chain. Similar to the underactuated Acrobot, the optimal control problem is converted into a discrete parameter optimization and a gradient-based algorithm is used to solve it.

To simulate a paralyzed person, a dynamic model for a branched kinematic chain was used to approximate the kinematics and dynamics of a human subject. For the dynamic analysis, four rigid bodies were used, with the head arms and torso lumped together as a single body, and the upper leg, lower leg, and foot modeled as single rigid bodies. The torso had five degrees of freedom with three translations and two rotations. Rotation about an axis through the upper leg joints was ignored. The upper leg was connected to the torso with a three degree of freedom ball joint. The lower leg was connected to the upper leg by a one dof rotation. Similarly, the foot was connected to the lower leg by a one dof rotation. Thus the entire system had ten degrees of freedom. For the swing hip, knee and ankle joints, a torque was applied to simulate the stiffness of passive tissue, but no torque from the muscles since the person is assumed to be paralyzed. A total of 32 B-spline parameters were used in the optimization to specify the motion of the swing hip. This problem differed from the Acrobot because we had to constrain the motion of the foot to avoid contact with the ground, and the motion of the legs to avoid contact with each other. We used penalty functions to enforce these collision avoidance constraints.

In the optimization results shown in Fig. 5, we found the motion of the swing hip that produced a step for the swing leg that was as close as possible

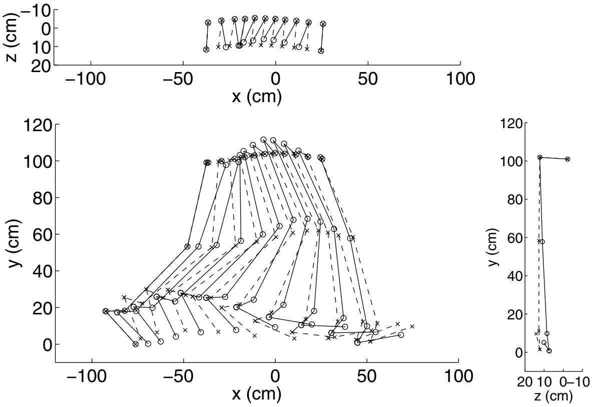


Fig. 5. Motion of the pelvis can be used to create motion for a paralyzed swing leg. The *solid lines* show gait which results from optimal motion of the pelvis, and the *dashed lines* are the gait recorded from the motion capture system

to a normal human gait. The optimal control found from our algorithm lifted the swing hip to avoid collision between the swing leg and the ground. At the same time, it twisted the pelvis to pump energy into the paralyzed leg and moved the leg close to the desired final configuration, while avoiding collision between the legs. Thus we found a strategy that could achieve repetitive stepping by shifting the pelvis alone. The optimized, pelvic motion strategies are comparable to "hip-hiking" gait strategies used by people with lower limb prostheses or hemiparesis.

Even though there were relatively few parameters (32) in the optimization, it was not numerically stable and took approximately 4 hours to converge, with approximately five minutes used for every major iteration of the optimization. The problems were again due to round-off errors introduced by the computation of the gradient in (7). The penalty functions for obstacle avoidance exacerbated the problem since they effectively created a "stiff" system of differential equations. The above results only considered the swing phase of the gait cycle. In our initial attempts to combine the stance phase with the swing phase in the optimal control solution were numerically unstable and did not converge to a solution.

Based on our initial results from the simple hopping machine considered in Case 4, we believe that with our new algorithm it will be possible to combine the stance and swing phases and reliably compute an optimal motion for Case 3 in just a few minutes of computation time. This would make it possible to compute an optimal motion for each patient in a clinical setting.

2.4 Case 4: Minimum Fuel Hopping

In order to explore the difficulties associated with the change in dynamics between the stance phase and swing phase of motion mentioned in Case 3, we considered a simple one-dimensional hopping system shown in Fig. 1. This system is driven by a pneumatic actuator, with the location of the piston relative to the mass under no external loading defined as y_p. After contact occurs with the ground with $y \leq y_p$, the upward force on the mass from actuator can be approximated by a linear spring with $F = k(y_p - y)$, where k is the spring constant. The position y_p can be viewed as the unstretched spring length and it can be easily changed by pumping air into or out of either side of the cylinder. The equations of motion for the mass are $m\ddot{y} = F(y, y_p) - mg$, where mg is the force due to gravity, and $F(y, y_p) = \left\{ \begin{smallmatrix} 0 & y > y_p \\ k(y_p - y) & \text{otherwise} \end{smallmatrix} \right.$. Note that in this case $F(y, y_p)$ is not differentiable at $y = y_p$, and gradient-based methods will have difficulties with this. However, the discontinuity in the derivative can easily be smoothed. For instance, let the spring compression be $e = y_p - y$ and choose an $\alpha > 0$, then

$$F(e) = \begin{cases} 0 & 0 > e \\ \frac{k}{2\alpha} e^2 & 0 \leq e < \alpha \\ ke - \frac{k\alpha}{2} & \text{otherwise} \end{cases}$$

is C^1. The final equation of motion for this system relates the air flow into the cylinder, which is the control $u(t)$, to the equilibrium position y_p of the piston. Assume for the following that the equation $\dot{y}_p = u$ approximates this relationship.

When the hopping machine begins its operation, we are interested in starting from rest, and reaching a desired hop height y_N^o at time t_f. If we minimize

$$J(u) = \frac{1}{2} q_{fin} (y(N) - y_N^o)^2 + \dot{y}(N)^2 + \frac{t_f}{2N} \sum_{n=0}^{N-1} \left[q \; y_p(n)^2 + r \; u(n)^2 \right] \;, \quad (8)$$

the terms outside the summation reflect the desire to reach the height at time t_f with zero velocity, and the terms inside the summation reflect the desire to minimize the gas used to achieve this. The weighting on y_p is used to keep the piston motion within its bounds. We first attempted to solve this problem by parameterizing the control $u(t)$ with B-splines and using the basic steps used in Cases 2 and 3. Even after considerable tweaking of tolerances, the gradient-based algorithm would not converge. This drove us to develop the algorithm described in the next section.

3 Problem Formulation and Background Results

We assume that the dynamic system defined by (1) and the performance measure (2) have been descretized by a suitable numerical integration scheme. To simplify the notation, we use the same function and variable names for the discrete-time versions of the continuous-time variables. A more detailed discussion of this material can be found in [14].

$$\underset{u(n),\; x(n)}{\text{Minimize}} \; J = \phi(x(N)) + \sum_{n=0}^{N-1} L(x(n), u(n), n) \quad (9)$$

$$\text{subject to} \qquad x(n+1) = f(x(n), u(n)); \quad x(0) = x_0 \quad (10)$$

We further assume a quadratic performance index, namely:

$$\begin{aligned} L(x(n), u(n), n) = {} & \frac{1}{2} [x(n) - x^o(n)]^T Q(n) [x(n) - x^o(n)] \\ & + [u(n) - u^o(n)]^T R(n) [u(n) - u^o(n)] \end{aligned} \quad (11)$$

and

$$\phi(x) = \frac{1}{2} [x - x^o(N)]^T Q(N) [x - x^o(N)] \quad (12)$$

In (11) and (12), $u^o(n)$, $x^o(n)$, $n = 1, \dots\ N$ are given control input and state offset sequences. In standard optimal regulator control problem formulations, $u^o(n)$, $x^o(n)$ are usually taken to be zero with the exception perhaps of $x^o(N)$, the desired final value for the state. The formulation considered here addresses the more general optimal tracking control problem and is required for the linear quadratic step in the proposed algorithm presented in Sect. 3.2.

3.1 First Order Optimality Conditions

We next briefly review the first order optimality conditions for the optimal control problem of (9) and (10), in a manner that brings out certain important interpretations of the adjoint dynamical equations encountered in a control theoretic approach and Lagrange Multipliers found in a pure optimization theory approach such as that mentioned in Sect. 1.1.

Let us consider the *cost-to-go*:

$$J(n) \equiv \sum_{n=k}^{N-1} L(x(k), u(k), k) + \phi(x(N)) \tag{13}$$

with L and ϕ as defined in (11) and (12) respectively. We remark that $J(n)$ is a function of $x(n)$, and $u(k)$, $k = n, \dots, N-1$ and introduce the sensitivity of the cost to go with respect to the current state:

$$\lambda^T(n) = \frac{\partial J(n)}{\partial x(n)} \tag{14}$$

Since

$$J(n) = L(x(n), u(n), n) + J(n+1)\ , \tag{15}$$

we have the recursion:

$$\begin{aligned} \lambda^T(n) &= L_x(x(n), u(n), n) + \lambda^T(n+1) f_x(x(n), u(n)) \\ &= [x(n) - x^o(n)]^T Q(n) + \lambda^T(n+1) f_x(x(n), u(n)) \end{aligned} \tag{16}$$

by using (11) and where L_x and f_x denote the partials of L and f respectively with respect to the state variables. The previous recursion can be solved backward in time ($n = N-1, \dots, 0$) given the control and state trajectories and it can be started with the final value:

$$\lambda^T(N) = \frac{\partial L(N)}{\partial x(N)} = [x(N) - x^o(N)]^T Q(N) \tag{17}$$

derived from (12). We now compute the sensitivity of $J(n)$ with respect to the current control $u(n)$. Clearly from (15),

$$\begin{aligned} \frac{\partial J(n)}{\partial u(n)} &= L_u(x(n), u(n), n) + \lambda^T(n+1) f_u(x(n), u(n)) \\ &= [u(n) - u^o(n)]^T R(n) \\ &\quad + \lambda^T(n+1) f_u(x(n), u(n)) \end{aligned} \tag{18}$$

since $\frac{\partial J(n+1)}{\partial u(n)} = \frac{\partial J(n+1)}{\partial x(n+1)} \cdot \frac{\partial x(n+1)}{\partial u(n)} = \lambda^T(n+1) f_u(x(n), u(n))$. In (19) L_u and f_u denote the partials of L and f respectively with respect to the control variables and (11) is used.

Next note that $\frac{\partial J}{\partial u(n)} = \frac{\partial J(n)}{\partial u(n)}$ since the first n terms in J do not depend on $u(n)$. We have then obtained the gradient of the cost with respect to the control variables, namely:

$$\nabla_u J = \left[\frac{\partial J(0)}{\partial u(0)} \; \frac{\partial J(1)}{\partial u(1)} \cdots \frac{\partial J(N-1)}{\partial u(N-1)} \right] . \tag{19}$$

Assuming u is unconstrained, the first order optimality conditions require that

$$\nabla_u J = 0. \tag{20}$$

We remark that by considering the Hamiltonian

$$H(x, u, \lambda, n) \equiv L(x, u, n) + \lambda^T f(x, u) , \tag{21}$$

we have that $H_u(x(n), u(n), \lambda(n+1), n) \equiv \frac{\partial J}{\partial u(n)}$, i.e. we uncover the generally known but frequently overlooked fact that the partial of the Hamiltonian with respect to the control variables u is the gradient of the cost function with respect to u. We emphasize here that in our approach for solving the optimal control problem, we take the viewpoint of the control variables $u(n)$ being the independent variables of the problem since the dynamical equations express (recursively) the state variables in terms of the controls and thus can be eliminated from the cost function. Thus in taking the partials of J with respect to u, J is considered as a function $u(n)$, $n = 0, \ldots, N-1$ alone, assuming that $x(0)$ is given. With this perspective, the problem becomes one of unconstrained minimization, and having computed $\nabla_u J$, Steepest Descent, Quasi-Newton, and other first derivative methods can be brought to bear to solve it.

Note that we are limiting the problem to the case of unconstrained controls, but for many problems in robotics, control constraints are not the limiting factor. In addition, soft constraints on actuation bounds can often be enforced with penalty functions with good fidelity since these constraints are usually not known precisely. With the control unconstrained, and with the large-scale character of the problem, methods that take advantage of the special structure of the problem become viable. The Linear Quadratic Regulator algorithm is such an approach in case of linear dynamics. We review it next and we remark that it corresponds to taking a Newton step in view of the previous discussion.

3.2 Linear Quadratic Tracking Problem

We next consider the case of linear dynamics in the optimal control problem of (9) and (10). In the following, we distinguish all variables corresponding to the linear optimal control problem that may have different values in the nonlinear optimal control problem by using an over-bar. When the cost is quadratic as in (11) we have the well-known Linear Quadratic Tracking problem. The control theoretic approach to this problem is based on solving the first order necessary optimality conditions (also sufficient in this case) in an efficient manner by introducing the Riccati equation. We briefly elaborate on this derivation next, for completeness and also since most references assume that the offset sequences $x^o(n)$ and $u^o(n)$ are zero. First, we summarize the first order necessary optimality conditions for this problem.

$$\begin{aligned} \bar{x}(n+1) &= A(n)\bar{x}(n) + B(n)\bar{u}(n) && (22)\\ \bar{\lambda}^T(n) &= [\bar{x}(n) - \bar{x}^o(n)]^T Q(n) && (23)\\ &\quad + \bar{\lambda}^T(n+1)A(n) \\ \partial\bar{J}(n)/\partial\bar{u}(n) &= [\bar{u}(n) - \bar{u}^o(n)]^T R(n) && (24)\\ &\quad + \bar{\lambda}^T(n+1)B(n) \ = \ 0 \end{aligned}$$

Note that the system dynamical equations (22) run forward in time $n = 0, \ldots, N-1$ with initial conditions $\bar{x}(0) = \bar{x}_0$ given, while the adjoint dynamical equations (24) run backward in time, $n = N-1, \ldots, 0$ with final conditions $\bar{\lambda}^T(N) = [\bar{x}(N) - \bar{x}^o(N)]^T Q(N)$. From (25), we obtain

$$\bar{u}(n) = \bar{u}^o(n) - R(n)^{-1} B(n)^T \bar{\lambda}(n+1) \tag{25}$$

and by substituting in (22) and (24), we obtain the classical two-point boundary system but with additional forcing terms due to the $\bar{x}^o(n)$ and $\bar{u}^o(n)$ sequences.

$$\begin{aligned} \bar{x}(n+1) &= A(n)\bar{x}(n) - B(n)R(n)^{-1}B(n)^T\bar{\lambda}(n+1) && (26)\\ &\quad + B(n)\bar{u}^o(n) \\ \bar{\lambda}^T(n) &= Q(n)\bar{x}(n) + A^T(n)\bar{\lambda}(n+1) - Q(n)\bar{x}^o(n) && (27) \end{aligned}$$

The system of (26) and (27) can be solved by the *sweep method* [3], based on the postulated relation

$$\bar{\lambda}(n) = P(n)\bar{x}(n) + s(n) \tag{28}$$

where $P(n)$ and $s(n)$ are appropriate matrices that can be found as follows. For $n = N$, (28) holds with

$$P(N) = Q(N), \qquad s(N) = -Q(N)\bar{x}^o(N). \tag{29}$$

We now substitute (28) in (26) and after some algebra we obtain

$$\bar{x}(n+1) = M(n)A(n)\bar{x}(n) + v(n) \tag{30}$$

where we defined

$$M(n) = \left[I + B(n)R(n)^{-1}B(n)^T P(n+1)\right]^{-1} \tag{31}$$

$$v(n) = M(n)B(n)[\bar{u}^o(n) - R(n)^{-1}B(n)^T s(n+1)] \tag{32}$$

By replacing $\bar{\lambda}(n)$ and $\bar{\lambda}(n+1)$ in (27) in terms of $\bar{x}(n)$ and $\bar{x}(n+1)$ from (28), we obtain

$$\begin{aligned} P(n)\bar{x}(n) + s(n) = {} & Q(n)\bar{x}(n) + A^T(n)\,[P(n+1)\bar{x}(n+1) \\ & + s(n+1)] - Q(n)\bar{x}^o(n), \end{aligned}$$

and by expressing $\bar{x}(n+1)$ from (30) and (32) above, we get

$$\begin{aligned} & P(n)\bar{x}(n) + s(n) = \\ & \qquad Q(n)\bar{x}(n) + A^T(n)P(n+1)M(n)A(n)\bar{x}(n) \\ & \qquad -A^T(n)P(n+1)M(n)B(n)R(n)^{-1}B(n)^T s(n+1) \\ & \qquad +A^T(n)P(n+1)M(n)B(n)\bar{u}^o(n) \\ & \qquad +A^T(n)s(n+1) - Q(n)\bar{x}^o(n) \end{aligned}$$

The above equation is satisfied by taking

$$P(n) = Q(n) + A^T(n)P(n+1)M(n)A(n) \tag{33}$$

$$\begin{aligned} s(n) = {} & A^T(n)[I - P(n+1)M(n)B(n)R(n)^{-1}B(n)^T]s(n+1) \\ & +A^T(n)P(n+1)M(n)B(n)\bar{u}^o(n) - Q(n)\bar{x}^o(n) \end{aligned} \tag{34}$$

Equation (33) is the well-known Riccati difference equation and together with the auxiliary equation (34), which is unnecessary if $\bar{x}^o(n)$ and $\bar{u}^o(n)$ are zero, are solved backward in time ($n = N-1, \ldots, 1$), with final values given by (29) and together with (31) and (32). The resulting values $P(n)$ and $s(n)$ are stored and used to solve forward in time ($n = 0, \ldots, N-1$), (30) and (25) for the optimal control and state trajectories. These equations are summarized in Table 1.

3.3 Formulation of the SLQ Algorithm

In the proposed SLQ algorithm, the control at stage $k+1$ is found by performing a one-dimensional search from the control at stage k and along a search direction that is found by solving an Linear Quadratic (LQ) optimal control problem. Specifically, let $U_k = [u(0)\ u(1) \ldots u(N-1)]$ be the optimal solution candidate at step k, and $X_k = [x(1)\ x(2) \ldots x(N)]$ the corresponding state trajectory obtained by solving the dynamical equations (10) using U_k and with the initial conditions $x(0)$. We next linearize the state equations (10) about the nominal trajectory of U_k and X_k. The linearized equations are

$$\bar{x}(n+1) = f_x(x(n),\ u(n))\bar{x}(n) + f_u(x(n),\ u(n))\bar{u}(n) \tag{35}$$

with initial conditions $\bar{x}(0) = 0$. We then minimize the cost index (9) with respect to $\bar{u}(n)$. The solution of this LQ problem gives $\bar{U}_k = [\bar{u}(0)\ \bar{u}(1) \ldots \bar{u}(N-1)]$, the proposed search direction. Thus, the control variables at stage $k+1$ of the algorithm are obtained from

Table 1. Algorithm to solve the Discrete-Time Finite-Horizon Linear Quadratic Tracking optimal control problem

1. **Solve backward ($n = N-1, \ldots, 0$) with $P_N \equiv Q_N$ and $s_N \equiv -Q_N \bar{x}_N^o$:**

$$\begin{aligned} M(n) &= \left[I + B(n)R(n)^{-1}B(n)^T P(n+1)\right]^{-1} \\ P(n) &= Q(n) + A(n)^T P(n+1) M(n) A(n) \\ s(n) &= A(n)^T \left[I - P(n+1)M(n)B(n)R(n)^{-1}B(n)^T\right] s(n+1) \\ &\quad + A(n)^T P(n+1) M(n) B(n) \bar{u}^o(n) - Q(n)\bar{x}^o(n) \end{aligned}$$

2. **Solve forward ($n = 0, \ldots, N-1$) with $\bar{x}(0) = \bar{x}_0$:**

$$\begin{aligned} v(n) &= M(n)B(n)[\bar{u}^o(n) - R(n)^{-1}B(n)^T s(n+1)] \\ \bar{x}(n+1) &= M(n)A(n)\bar{x}(n) + v(n) \\ \bar{\lambda}(n+1) &= P(n+1)\bar{x}(n+1) + s(n+1) \\ \bar{u}(n) &= \bar{u}^o(n) - R(n)^{-1}B(n)^T \bar{\lambda}(n+1) \end{aligned}$$

$$U_{k+1} = U_k + \alpha_k \cdot \bar{U}_k \tag{36}$$

where $\alpha_k \in \mathbb{R}^+$ is appropriate stepsize the selection of which is discussed later in the paper. Note again our perspective of considering the optimal control problem as an unconstrained finite-dimensional optimization problem in U.

We emphasize that $\bar{U}_k$ as computed above is *not* the steepest descent direction. It is the solution to a linear quadratic tracking problem for a nonlinear system that has been linearized about U_k. Note that the objective function is not linearized for this solution. Our algorithm is different than standard *Quasilinearization* [3] and *Neighboring Extremal* [18] methods where the adjoint equations are also linearized and two-point boundary problems are solved.

3.4 Properties of the SQL Algorithm

In this section, we prove two important properties of the proposed algorithm. First, we show that search direction $\bar{U}$ is a descent direction.

Theorem 1 *Consider the discrete-time nonlinear optimal control problem of (9) and (10), and assume a quadratic cost function as in (11) and (12) with $Q(n) = Q^T(n) \geq 0$, $Q(N) = Q^T(N) \geq 0$ and $R(n) = R^T(n) > 0$, $n = 0, 1, \ldots, N-1$. Also consider a control sequence $U \equiv [u(0)^T \ldots u^T(N-1)]^T$ and the corresponding state trajectory $X \equiv [x(1)^T \ldots x^T(N)]^T$. Next, linearize system (10) about U and X and solve the following linear quadratic problem:*

$$\underset{\bar{u}(n),\bar{x}(n)}{\mathit{Minimize}}\ \bar{J} = \frac{1}{2}[\bar{x}^T(N) - \bar{x}^o(N)]Q(N)[\bar{x}(N) + \bar{x}^o(N)] + \frac{1}{2}\sum_{n=0}^{N-1}\left\{[\bar{x}(n) - \bar{x}^o(n)]^T Q(n)[\bar{x}(n) - \bar{x}^o(n)] + [\bar{u}(n) - \bar{u}^o(n)]^T R(n)[\bar{u}(n) - \bar{u}^o(n)]\right\} \tag{37}$$

subj. to

$$\bar{x}(n+1) = f_x(x(n),u(n))\bar{x}(n) + f_u(x(n),u(n))\bar{u}(n); \qquad \bar{x}(0) = 0\ , \tag{38}$$

where $\bar{x}^o(n) \equiv x^o(n) - x(n)$*,* $\bar{u}^o(n) \equiv u^o(n) - u(n)$*. Then if* $\bar{U} \equiv [\bar{u}(0)^T \ldots \bar{u}^T(N-1)]^T$ *is not zero, it is a descent direction for the cost function (9), i.e.* $J(U + \alpha \cdot \bar{U}) < J(U)$ *for some* $\alpha > 0$*.*

Proof: We establish that $\bar{U}$ is a descent direction by showing that:

$$\nabla_u J \cdot \bar{U} = \sum_{n=0}^{N-1} \frac{\partial J(n)}{\partial u(n)} \bar{u}(n) < 0\ , \tag{39}$$

since $\nabla_u J$ in (19) is the gradient of the cost function with respect to the control variables. Now, the components of $\nabla_u J$ are expressed in (19) in terms of the adjoint variables $\lambda(n)$ that satisfy recursion (16) with final values given by (17). On the other hand, $\bar{x}(n)$ and $\bar{u}(n)$ together with adjoint variables $\bar{\lambda}(n)$ satisfy the first order optimality conditions for the linear quadratic problem given in (22), (24) and (25), where $A(n) = f_x(x(n),u(n))$ and $B(n) = f_u(x(n),u(n))$. Let us define

$$\tilde{\lambda}(n) = \bar{\lambda}(n) - \lambda(n) \tag{40}$$

and note from (16) and (24) that

$$\tilde{\lambda}(n)^T = \bar{x}(n)^T Q(n) + \tilde{\lambda}(n+1)^T A(n); \quad \tilde{\lambda}(N) = Q(N)\bar{x}(N)\ . \tag{41}$$

Next through the indicated algebra, we can establish the following relation:

$$\begin{aligned}
&\frac{\partial J(n)}{\partial u(n)} \cdot \bar{u}(n) = \\
&\quad = \Big([u(n) - u^o(n)]^T R(n) + \lambda(n+1)^T B(n)\Big)\bar{u}(n) \\
&\qquad \text{(using (19))} \\
&\quad = -\tilde{\lambda}(n+1)^T B(n)\bar{u}(n) - \bar{u}(n)^T R(n)\bar{u}(n) \\
&\qquad \text{(using (25))} \\
&\quad = -\tilde{\lambda}(n+1)^T \bar{x}(n+1) + \tilde{\lambda}(n+1)^T A(n)\bar{x}(n) - \bar{u}(n)^T R(n)\bar{u}(n) \\
&\qquad \text{(using (39))} \\
&\quad = -\tilde{\lambda}(n+1)^T \bar{x}(n+1) + \tilde{\lambda}(n)^T \bar{x}(n) - \bar{x}(n)^T Q(n)\bar{x}(n) \\
&\qquad - \bar{u}(n)^T R(n)\bar{u}(n). \qquad \text{(using (41))}
\end{aligned}$$

Finally, summing up the above equation from $n = 0$ to $n = N - 1$ and noting that $\bar{x}(0) = 0$ and from (29) that $\bar{\lambda}(N) = Q(N)\bar{x}(N)$, gives:

$$\begin{aligned} \nabla_u J \cdot \bar{U} &= \sum_{n=0}^{N-1} \frac{\partial J(n)}{\partial u(n)} \cdot \bar{u}(n) \\ &= -\sum_{n=0}^{N-1} [\bar{x}(n)^T Q(n)\bar{x}(n) + \bar{u}(n)^T R(n)\bar{u}(n)] \\ &\quad - \bar{x}^T(N)Q(N)\bar{x}(N) \ < \ 0 \end{aligned} \tag{42}$$

and the proof of the theorem is complete. ■

We remark that the search direction $\bar{U}$ can be found by the LQR algorithm of Table 1 with $A(n) = f_x(x(n), u(n))$ and $B(n) = f_u(x(n), u(n))$.

The next result shows that the proposed SLQ algorithm does in fact converge to a control locally minimizing the cost function (9).

Theorem 2 *Starting with an arbitrary control sequence U_0, compute recursively new controls:*

$$U_{k+1} = U_k + \alpha_k \cdot \bar{U}_k \tag{43}$$

where the direction U_k is obtained as in Theorem 1 by solving the LQR problem of (37) and the linearized system (39) about the solution U_k and corresponding state trajectory X_k; also α_k is obtained by minimizing $J[U_k+\alpha\bar{U}_k]$ over $\alpha > 0$. Then U_k converges (in the Euclidean norm sense) to a control that locally minimizes the cost function (9) subject to the system equations (10).

Proof: See [14], or note that given the result of the previous theorem, standard convergence proofs (see [7]) apply with either an exact or an inexact linesearch such as the Armijo, Goldstein, or Wolfe search rules [9].

4 Numerical Example

We conducted numerical experiments the hopping system discussed in Case 4, above. We minimized (8) with the following parameters: $k/m = 100$, $g = 386.4$, $\alpha = 0.1$. We assumed that all states were initially zero, and that the initial control sequence was zero. The cost function parameters were selected as: $t_f = 1$, $q = 1000$, $r = 1.0$. As in the last example, a simple Euler approximation was used to discretize the equations, with $N = 50$.

As shown in Fig. 6, the algorithm produced an alternating sequence of stance phases and flight phases for the hopping system and it naturally identified the times to switch between these phases. If one were to use collocation methods to solve this problem with explicit consideration of the different dynamics in the different phases, one would have to guess the number of switches between phases and would need to treat the times at which the switch occurs

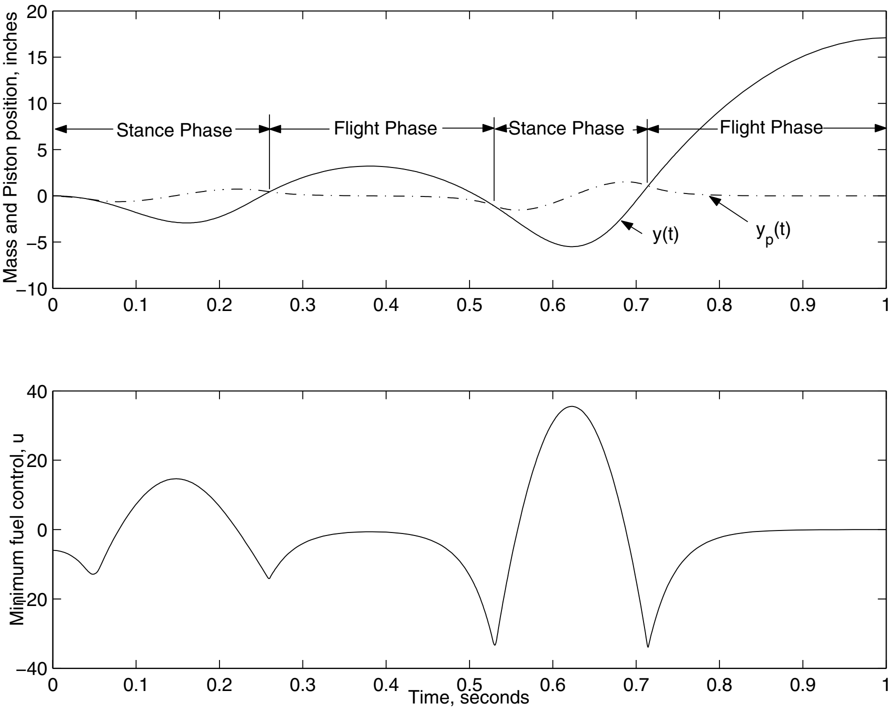


Fig. 6. Maximum height hopping motion and minimum fuel control

as variables in the optimization. We note that our algorithm converged much faster when the weighting on the control r is increased; also the number of iterations required for convergence in this problem increases for larger y_N^o, ranging from 3 for $y_N^o = 1$, to 166 for $y_N^o = 50$. In addition, the algorithm failed to converge for $\alpha < 1 \times 10^{-5}$, which demonstrates the need for the dynamics to be continuously differentiable.

5 Conclusion

We discussed the formulation and solution of several important optimal control problems for robotic systems. The most challenging case by far is an underactuated system with contact constraints. We developed an algorithm for solving such nonlinear optimal control problems for systems with quadratic performance measures and unconstrained controls. Contact constraints were accounted for with penalty functions. Each subproblem in the course of the algorithm is a linear quadratic optimal control problem that can be efficiently solved by Riccati difference equations. We show that each search direction generated in the linear quadratic subproblem is a descent direction, and that the algorithm is convergent. Computational experience has demonstrated that the algorithm converges quickly to the optimal solution.

References

[1] J.T. Betts, “Survey of Numerical Methods for Trajectory Optimization,” *Journal of Guidance, Control and Dynamics,* vol. 21: (2) 193–207, 1999.

[2] R.W. Brockett, “Robotic manipulators and the product of exponentials formula,” *Proc. Int. Symp. Math. Theory of Networks and Systems*, Beer Sheba, Israel, 1983.

[3] A.E. Bryson and Y.C. Ho, *Applied Optimal Control,* Wiley, New York, 1995.

[4] Y.-C. Chen,“Solving robot trajectory Planning problems with uniform cubic B-splines,” *Optimal Control Applications and Methods*, vol. 12, No. 4, pp. 247–262, 1991.

[5] A. De Luca, L. Lanari and G. Oriolo, “A Sensitivity Approach to Optimal Spline Robot Trajectories,” *Automatica*, vol. 27, No. 3, pp. 535–539, 1991.

[6] Enright P.J, Conway B.A. Optimal finite-thrust spacecraft trajectories using collocation and nonlinear programming. *Journal of Guidance, Control, and Dynamics,* 1991; 14:981–985.

[7] R. Fletcher, *Practical Methods of Optimization*, Wiley, 2nd edition, 1987.

[8] Hargraves C.R., Paris S.W. Direct trajectory optimization using nonlinear programming and collocation. *Journal of Guidance, Control, and Dynamics,* 1987; 10:338–342.

[9] D.G. Luenberger, *Linear and Nonlinear Programming,* Addison Wesley, 1989.

[10] B. Martin and J.E. Bobrow, “Minimum Effort Motions for Open Chained Manipulators with Task-Dependent End-Effector Constraints,” *International Journal of Robotics Research,* vol. 18, No. 2, February, 1999, pp. 213–224.

[11] R.W. Murray, Z. Li, and S.S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, Boca Raton, Florida, 1993.

[12] J.P. Ostrowski, J.P. Desai, V. Kumar, “Optimal gait selection for nonholonomic locomotion systems,” *International Journal of Robotics Research*, vol. 19, (no. 3), March 2000. pp. 225–237.

[13] F.C. Park, J.E. Bobrow, and S.R. Ploen, “A Lie Group Formulation of Robot Dynamics,” *International Journal of Robotics Research,* vol. 14, No. 6, December 1995, pp. 609–618.

[14] A. Sideris and J. E. Bobrow, “An Efficient Sequential Linear Quadratic Algorithm for Solving Nonlinear Optimal Control Problems,” *Proceeding of the 2005 IEEE Conference on Decision and Control,* Portland, Oregon.

[15] G.A. Sohl, and J.E. Bobrow, “A Recursive Dynamics and Sensitivity Algorithm for Branched Chains,” *ASME Journal of Dynamic Systems, Measurement and Control,* vol. 123, no. 3, Sept. 2001.

[16] G.A. Sohl, and J.E. Bobrow, “Optimal Motions for Underactuated Manipulators,” *1999 ASME Design Technical Conferences,* Las Vegas, September.

[17] M. Spong, “The swing up control problem for the Acrobot,” *IEEE CONTR SYST MAG,* vol. 15: (1) 49–55 Feb. 1995.

[18] Veeraklaew T., and Agrawal S.K., “Neighboring optimal feedback law for higher-order dynamic systems,” *ASME J. Dynamic Systems, Measurement, and Control,* 124 (3): 492–497 Sep. 2002.

[19] von Stryk O. Numerical solution of optimal control problems by direct collocation. *Optimal Control*, Bulirsch R, Miele A, Stoer J, Well KH (eds), *International Series of Numerical Mathematics,* vol. 111. Birkshauser Verlag; Basel, 1993; 129–143.

[20] Chia-Yu E. Wang, Wojciech K. Timoszyk and James E. Bobrow, "Payload Maximization for Open Chained Manipulators: Finding Weightlifting Motions for a Puma 762 Robot," *IEEE Transactions on Robotics and Automation,* April 2001, vol. 17, No. 2 pp. 218–224.

[21] C.-Y.E. Wang, J.E. Bobrow, and D.J. Reinkensmeyer, "Swinging From The Hip: Use of Dynamic Motion Optimization in the Design of Robotic Gait Rehabilitation," *Proceedings of the IEEE Conference on Robotics and Automation,* Seoul, Korea, May 2001.

[22] I. Wickelgren, "Teaching the spinal cord to walk," *Science*, vol. 279, pp. 319–321, 1998.

A Spring Assisted One Degree of Freedom Climbing Model

J.E. Clark[1] and D.E. Koditschek[2]

[1] University of Pennsylvania, Philadelphia, PA, USA
jonclark@grasp.upenn.edu
[2] University of Pennsylvania, Philadelphia, PA, USA
kod@ese.upenn.edu

Summary. A dynamic model of running–the spring-loaded inverted pendulum (SLIP)–has proven effective in describing the force patterns found in a wide variety of animals and in designing and constructing a number of terrestrial running robots. Climbing or vertical locomotion has, on the other hand, lacked such a simple and powerful model. Climbing robots to date have all been quasi-static in their operation. This paper introduces a one degree of freedom model of a climbing robot used to investigate the power constraints involved with climbing in a dynamic manner. Particular attention is paid to understanding how springs and body dynamics can be exploited to help relieve a limited power/weight ratio and achieve dynamic running and climbing.

1 Introduction

We seek a fast and agile robot that can traverse both vertical and horizontal real world terrain. Dynamic locomotion over unstructured and natural terrain has proven to be a difficult task. A large number of walking robots have been built, but only recently have running robots been developed that can move at speeds of bodylengths/second over rough terrain [6, 21, 15]. The *Rhex* [6] and *Sprawl* [15] families of dynamic machines are based on a Spring-loaded Inverted Pendulum (SLIP) model of running developed from biomechanical research [13, 20]. They have simple morphologies with only one actuator per leg, are polypedal, run mostly open-loop, and rely on springs in the legs to passively self-stabilize.

On the other hand, there have only been a few legged robots that can climb vertical surfaces, and they have generally been limited to quasi-static climbing regimes. Their reliance on vacuum or magnetics to achieve the adhesive forces necessary for vertical climbing has limited them to select man-made surfaces such as glass and metal [19, 10, 23]. Recently foot-hold based [11, 12] and vectored thrust based climbers [1] have been developed, but they only slightly extend the range of traversable surfaces and do not address the role of dynamics in climbing.

The motivation for this work is the ongoing development of the *RiSE* robot–a bio-inspired hexapedal platform that is intended to both run and climb vertical surfaces [8]. Currently under development are micro-spine [7, 16] and dry-adhesive feet [22] to allow attachment to a wide range of natural vertical environments. The current instantiation of the robot, however, is otherwise like the remainder of climbing robots in that it moves slowly and its control is based on quasi-static assumptions. The purpose of this work is to explore how to achieve dynamic climbing and how that can be used to improve the performance of the RiSE climbing robot.

To this end we discuss some of the fundamental differences between dynamic running and climbing and introduce a simple one-dimensional dynamic climbing model to investigate approaches to mitigate some of the difficulties in achieving dynamic climbing.

1.1 Dynamic Climbing

We reserve the term *dynamic* for robots that manage their kinetic as well as their potential energy. For example, dynamic level ground running can be distinguished from quasi-static locomotion by the phasing of kinetic and gravitational potential energy during a stride. Generally, dynamic runners are distinguished in physical structure by their essential use of springs in the legs. These leg springs act as reservoirs that can store and return energy at whatever required rate. In a typical dynamic gait, the spring energy is collected during the initial phase of a stride ("compression") and returned during the second phase ("decompression") as work done against gravity needed to raise again the center of mass back close to its height lost in the initial phase. Dynamic runners can (but need not) adopt an aerial phase gait to buy time for leg recirculation, thereby affording speeds that surpass the inevitable frequency limits of their leg swing actuators. In such situations, springs can recover and return the kinetic energy otherwise lost in body deceleration.

Properly arranging these exchanges of kinetic and spring and gravitational energy requires control schemes designed to do more than simply track the joint reference trajectories typically used by walkers. The resulting dynamic stability confers a degree of agility and maneuverability impossible to achieve in quasi-static walking gaits. The question arises whether spring assistance can be introduced in climbing that yields analogous benefits.

The major difference between climbing and running is in the alignment of the gravity vector with respect to the direction of travel. We suggest that this has three primary impacts on legged climbers. The first is that travel aligned with the gravity vector implies that any forward progression increases the gravitational potential of the robot, resulting in a net drain on the rest of the system's energy. As a consequence the SLIP model, which relies on the interchange of kinetic and gravitational potential energy from hopping or bouncing to regulate the energy during a stride, no longer applies.

In addition to necessarily changing the way in which kinetic and spring potential energy are exchanged with gravitational potential energy, a vertical heading also implies that ground impacts are not induced by gravity. Especially for the front feet ground contact must be actively generated. This changes, and to some degree reduces, the role of springs in mitigating ground contact forces. Successful running robots have required springs to regulate the impact at ground contact, and this is not necessarily the case for climbers.

A third major difference for climbing robots is the necessity of bi-lateral or "adhesive" foot constraints. The development of feet that create an in-pulling force to the wall is one of the major design requirements in a climbing robot. Having to create feet that grasp the wall to deal with the inherent pitch instability serendipitously reduces the chance of tipping in the roll direction–which is a major source of instability in level ground runners. Once attached, tipping becomes less of a problem, but motions such as repositioning the feet on the ground via sliding, as is often done in turning, become more difficult.

In some sense these differences make climbing easier than running since severe foot impacts and lateral tipping are less likely to occur. On the other hand getting good foot attachment and regulating the system's energy become much more difficult. The problem of attachment has and continues to receive a fair amount of attention. The second problem, more effectively using the system's power resources, motivates our investigation with a simple dynamic climbing model.

A dynamic robot may lend scansorial machines advantages relative to today's quasi-static climbers analogous to the superiority of level ground runners over their quasi-static walking counterparts: simplified control; improved efficiency; access to and mobility through otherwise impassible terrain obstacles; and, of course, faster speeds.

1.2 Power and Speed Constraints

We propose a simple one-dimensional climbing model to investigate the power requirements and constraints associated with dynamic behavior. As a target for dynamic motion we set a stride frequency of 3.5 Hz for our 3 kg robot. Specifically, this figure is associated with the natural frequency of the linear mass spring model associated with purely vertical SLIP hopping.

At lower frequencies, back-of-the-envelope calculations developed in Appendix A1 suggest that spring-extension requirements for SLIP-like running (i.e., resonant bouncing in the sagittal plane over level ground) would incur impractically long leg compression.

Another method that has been used to characterize the onset of running is the Froude number, (v^2/gl), where v is the average fore-aft running speed, l is the leg length and g is the constant of gravitational acceleration. The Froude number is a dimensionless constant that has been used in biomechanics to predict the dynamic similarity in legged animals that is manifest over a wide range of sizes. It has been shown that many animals prefer to switch from a walk to a run at speeds where their Froude number is about 0.5 [4, 18].

RiSE climbing at the target frequency of 3.5 Hz would travel at 0.55 m/s which corresponds to a Froude number of 0.2. While it is not clear that the Froude number is as applicable to climbing as it is to terrestrial locomotion, it does give some indication of when velocity begins to significantly affect the energetics of motion. The relatively low value of our target frequency's Froude number with respect to observed animal gait transitions suggests that the target frequency we have chosen for dynamic climbing is probably not too high.

With the current trajectory-tracking, quasi-static control scheme the robot can climb with a stride frequency of 0.5 Hz. Is it theoretically possible to achieve the required 7x increase in speed without changing the motors or decreasing the robot's mass?

The current robot, weighing 3 kg and equipped with two 4.5 W rated servo motors for each of its six legs, has an input electrical power-to-mass ratio of 18:1. In order to climb vertically at our dynamic threshold (0.53 m/s with a stride length of 0.15 m) requires a mechanical output power of 16 W just to deliver the energy expended to increase the system's gravitational potential.

Experiments on Geckos running up vertical walls has shown that the mechanical power that they expend to run at speeds up to 10 bodylengths/second is about 5 W per kilogram of animal [9]–about the same ratio as for RiSE were it to run at 3.5 Hz. What is remarkable is that for the gecko the mechanical power expended when climbing is only about 10% greater than the amount of energy lost to gravitational potential.

The 16 W power requirement for RiSE running at this speed represents 30% of the maximum continuous electrical input power that the robot motor's can consume without thermal damage. In reality only a small percentage of the motors' 54 W rating will be converted into useful mechanical work. The two major reasons for this are (1) the motors are run at maximum power for only a small segment of the stride and (2) motor inefficiencies (e.g. thermal losses in the windings and mechanical losses in the bearings) and system "drag" (e.g., transmission losses, generation of internal forces and negative work associated with securing footholds) significantly diminish the mechanically useful component of the power the motors consume.

In this paper we address the first of these two problems. Specifically we consider how to use dynamic gaits and body/leg springs to better utilize the available motor power. We show that in the ideal case these approaches significantly reduce the peak power demanded from the motors permitting a smaller gear reduction, which in turn allows a higher stride frequency.

With ideal motors the changing demands for torque and speed during the leg cycle could be met by implementing a torque control law and allowing the motor to operate at different points along the speed-torque curve. As shown in Fig. 2 for the current motors, only 20% of the torque range is available for continuous use. This dramatically reduces the flexibility of any such motor control law. Instead we investigate how to use passive springs and the body's inertia to maximize the application of the available electrical power.

The remainder of this paper is organized as follows. Section 2 describes the one-dimensional model of climbing that we use to evaluate the efficacy of the proposed schemes. The simplifying assumptions and equations of motion are given. Section 3 details the numerical studies undertaken and compares the various cases considered. Section 4 reviews the results and gives some areas of future work.

2 Model Description

2.1 Assumptions

The RiSE robot (see Fig. 1a) is a six limbed climber with two controlled degrees of freedom per leg. Each leg can rotate about an axis parallel to the direction of motion, lifting the foot away from the ground (wing DOF). The second actuated degree of freedom controls the rotation of the crank segment of a four bar mechanism connecting the foot to the body. The foot is attached to the follower link of the mechanism and traces a roughly elliptical path (see Fig. 3) in a plane passing through the line of action of the wing DOF.

With the assumption that the wing DOF is primarily used to specify the point of touchdown and lift-off in the cycle, the motion of the legs and body can be modeled as planar. This abstraction neglects, among other things, body pitch away from the wall–which is known to be a significant issue.

The further assumptions that the robot uses an alternating tripod gait and that lateral motions of the robot are of secondary importance allow the

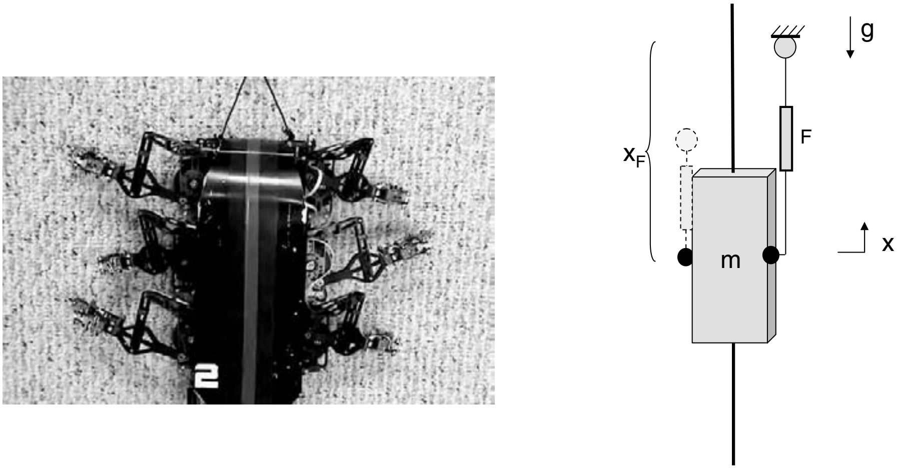


Fig. 1. (**A**) Picture of RiSE climbing and (**B**) schematic of the simple climbing model

construction of a single degree of freedom model of climbing, shown in Fig. 1b. The model consists of a point mass body with two virtual, massless legs. The extension of the foot (X_F) during stance is fixed by the leg kinematics. In this very simple model, we are ignoring friction from the legs, foot slipping, etc. Although this oversimplification of the system ignores many real and important effects, it is hopefully sufficient to examine some basic power and stability issues and provide a basis for future examinations.

2.2 Stance Dynamics

The sum of the forces in the vertical direction is given by:

$$m\ddot{x} = F - mg \tag{1}$$

where m is the mass of the body, F is the force generated by the motor, and g is the gravitational constant opposing the motion.

The force generated by the leg actuator is based on a very simple motor model. Due to thermal concerns arising from the nearly continuous use of the motors when climbing, we assume that the motors operate within their recommended operating range, shown in Fig. 2. Although the stall torque of the motor is 28.8 mNm, the continuous operational limit (τ_{Max}) is only 4.98 mNm. The represents about 20% of the speed/torque curve given by:

$$\tau = \frac{\omega - \omega_{nl}}{-k_m}$$

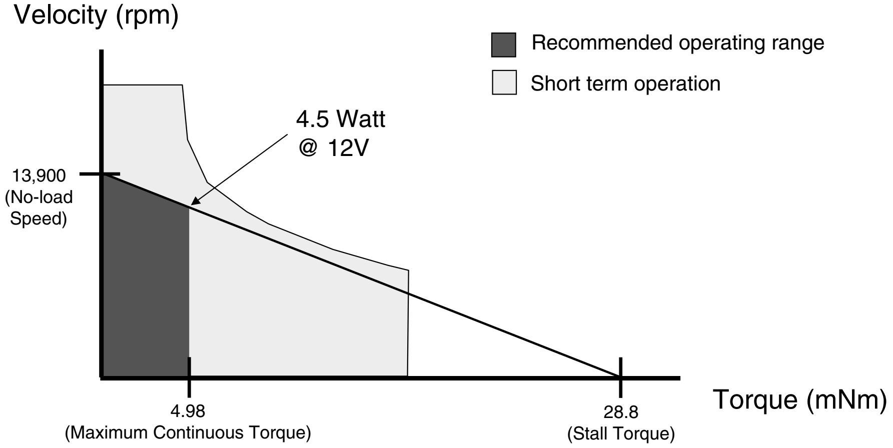


Fig. 2. Model and specifications for the motors used on the *RiSE* robot. Data from Maxon Precision Motors Inc. RE-16 motor (16 mm diameter, graphite brushes, 4.5 Watt, part number 118730) [2]

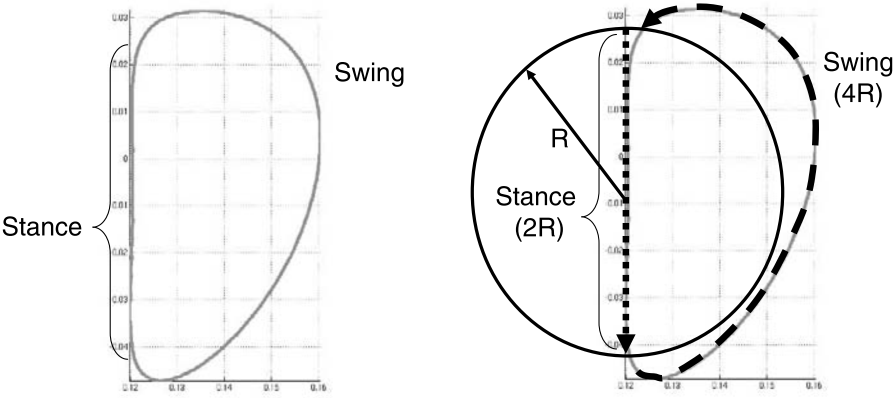


Fig. 3. Nominal RiSE foot trajectory and simplification used in the model. The RiSE leg kinematics result in a swing phase that is almost twice as long as the stance phase. (Leg trajectory from RISE [8])

where ω is the angular velocity of the motor, ω_{nl} is the no-load velocity limit, and k_m is the slope of the speed/torque curve. Thus the maximum continuously available torque is given by:

$$\tau = min\left(\frac{\omega - \omega_{nl}}{-k_m}, \tau_{Max}\right)$$

In order to adapt to a one DOF linear model the trajectory of the four-bar traveler is approximated with a circle of radius R, as shown in Fig. 3.

Linear Coordinates

With the following conversions:

$$F_{Max} = \frac{\tau_{Max}}{R}$$

$$\dot{x}_{nl} = R\,\omega_{nl}$$

$$k_e = \frac{\dot{x}_{nl}}{F_{Max}} = \frac{R^2\omega_{nl}}{\tau_{Max}} = R^2 k_m$$

the motor torque law becomes:

$$F = min\left(\frac{\dot{x} - \dot{x}_{nl}}{-k_e}, F_{Max}\right)$$

where (k_m) is the slope of the force/velocity curve. With the addition of a gear reduction (G), the force from the motor (F) becomes:

$$F = min\left(\frac{\dot{x} - \frac{\dot{x}_{nl}}{G}}{-G^2 k_e}, GF_{Max}\right) = min\left(\frac{G^2\dot{x} - G\dot{x}_{nl}}{-k_e}, GF_{Max}\right) \tag{2}$$

Combining (1) and (2) yields:

$$\ddot{x} = \frac{1}{m}\left(min\left(\frac{G^2\dot{x} - G\dot{x}_{nl}}{-k_e}, GF_{Max}\right) - mg\right) \tag{3}$$

Gait transitions occur when $(X_F - x) = 0$, i.e. when the leg has reached the end of the vertical section of travel. Since the four-bar mechanism fixes the gait trajectory, X_F is fixed at $2R$. Whether the leg is capable of resetting within the duration of the last stance phase is a function of the swing dynamics, described below.

2.3 Massless Swing Dynamics

Of course in the physical system the mass of the legs is non-zero, and the leg's trajectory during the swing phase is a function of its dynamics. Initially, however, these dynamics are ignored and the swing phases is considered as a binary state: either the leg can return to the touchdown position in time, or it cannot.

The time that it will take the leg to retract, t_{Swing}, is bounded by no-load, $\dot{x}_{nl}$, and max continuous velocity, $\dot{x}_{\tau_{Max}}$, of the foot, as given below:

$$\frac{G\,2R}{\dot{x}_{\tau_{Max}}} \geq t_{Swing} \geq \frac{G\,2R}{\dot{x}_{nl}} \tag{4}$$

The left side of 4 represents an upper bound on the duration of the swing phase.

3 Numerical Simulation

3.1 Trajectory vs. Force Based Gaits

In the current control philosophy a gait is generated by specifying a desired trajectory for the path of the feet. Typically four phases are specified: swing, attachment, stance, and detachment. Using the motor encoder readings and PD control the legs attempt to track this trajectory throughout the stride. Forces are generated when errors in the tracking occur. These generally correspond to contact with the ground during attachment, lifting the robot in the face of gravity during stance, and the inertial resistance to the rapid acceleration during swing. Figure 4 shows an idealized trajectory and the corresponding torques generated by the motors. The figure on the left is a projection of foot trajectory in the wing plane. A trace of the wing angle with respect to time is shown on the bottom right. In the plot on the right the solid horizontal line represents the maximum continuous torque as specified by the motor's manufacturer. The dotted line is the motor's mean torque over a stride. The motor torque curve itself is an abstraction of reality where the large spike in the torque graph corresponds to the body acceleration of the foot during stance, and the smaller spike to the acceleration of the foot during swing.

It should be noted that current gaits are designed for effective attachment and detachment rather than optimizing speed or utilization of available motor power. Significant improvements in terms of speed can (and are being) made by refining the shape of the target trajectory–especially during stance and swing–such that the torque demands more closely match the abstraction shown in Fig. 4.

Due to the large gear reduction employed, the ability to shape the torque trajectory is limited. If the peak load on the motors was decreased or distributed more efficiently throughout the stride then the gearing could be reduced and the top speed dramatically increased.

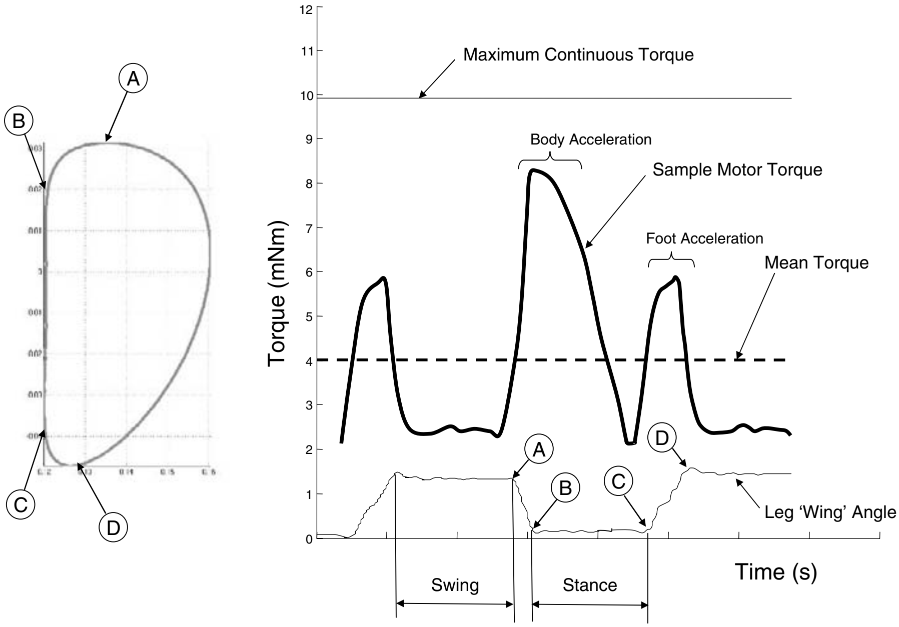


Fig. 4. Sample leg trajectory projected onto the "wing" plane (*left*) and in terms of the wing angle (*right, bottom*). These are used to contextualize the idealized force pattern (*right, top*). The leg trajectory and mean torque plots are based on RiSE robot data [3]

Simulation Results

Figure 5 shows how the theoretical minimum stride frequency for the simple model described in Sect. 2.2 varies as a function of gear ratio, G. Since each leg has a kinematically fixed stride length, the stride period is inversely proportional to velocity. The dashed curved line represents the stance phase speed limit for each gear ratio, G. The sloping starred line represents the swing phase reset threshold. Point (A) shows the theoretical maximum speed with the RiSE v1.0 gear ratio. At this gear ratio any higher speeds would require the leg to complete the swing phase faster than the motor can handle. Below a certain gear ratio (point (C) in Fig. 5) the robot no longer has enough force to overcome gravity and cannot climb. Increasing the gear ratio reduces the overall available speed and above a threshold, point (B), the duration of the swing phase becomes the limiting factor.

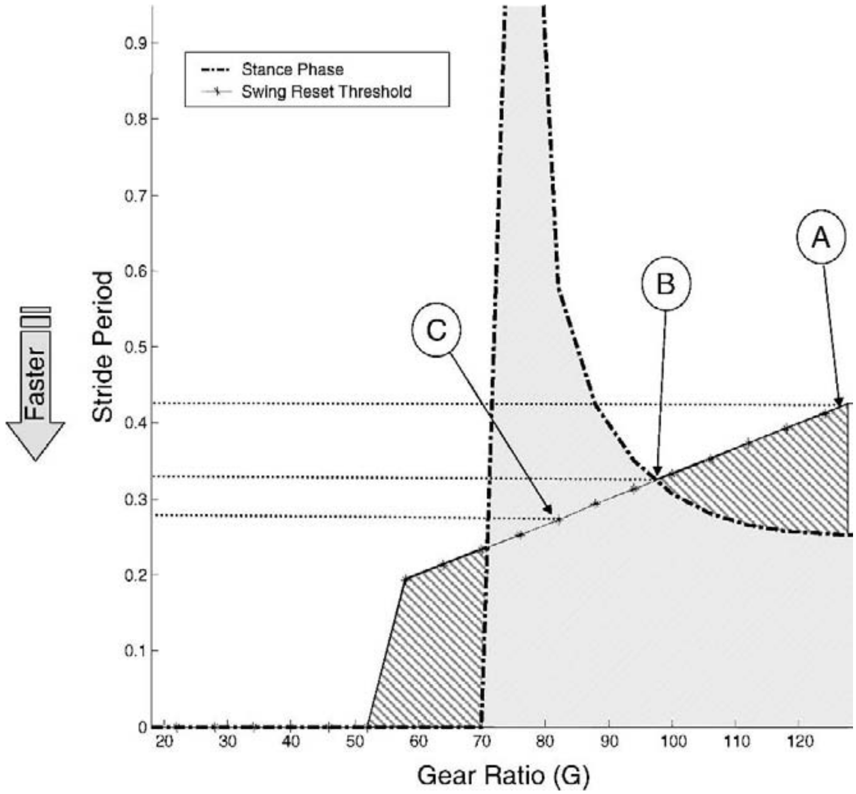


Fig. 5. Simulation results for the base model, showing the minimum possible stride period (maximum speed) for various gear ratios (G). The *shaded areas* under the curves represent regions that are inadmissible as they invalidate either swing or stance phase constraints

Point (B) is therefore the theoretical upper bounds on the velocity of the robot using the current trajectory-based gait formulation. This stride frequency (and speed) will, of course, never actually be reached due to the non-instantaneous, and non-trivial, attachment and detachment requirements. Nevertheless it is clear that without some change to the energetics and actuation scheme of the robot it will never be able to reach our target dynamic threshold.

Momentum

If, however, non-zero attachment speeds and the momentum of the body of the robot is explicitly considered in the control scheme the robot could accelerate from one stride to another and higher "steady-state" velocities are possible. This requires either allowing the foot trajectories to change as a function of body velocity or the adoption of some sort of force-based control scheme as is done in our simple model. In either case the new maximum speed will be limited by the swing phase reset time. In this case the fastest configuration corresponds to the lowest gear ratio that will actually lift the robot, as shown by point (C) in Fig. 5.

A potential difficulty with this approach is ensuring that the foot attachment trajectory remains viable as the body velocity increases. This problem is being considered in ongoing work on foot attachment dynamics.

3.2 Spring-Assisted Climbing

An alternative method for increasing speeds with a limited power budget is by the intelligent use of springs to redistribute the cyclic loading and level out the demands on the motors. By lowering the peak force requirements the drive-train gear ratio can be reduced to speed up the overall motion of the robot. This also brings the mean loading on the motors closer to the maximum continuous operation level. Since these motors get the most power at 1/2 of stall, and they are limited to 20% stall by the thermal constraints, maximum achievable power coincides with the maximum continuous operation point.

In this section we consider two approaches to using springs to assist the motion of the body in climbing. The first approach stores energy from the end of the stance phase when the body is decelerating in preparation for attachment and then releases it at the beginning of the next stance phase to help re-accelerate the body. The second approach uses the motor to store energy in a spring during the swing phase, which is then released to assist lifting the body during stance.

3.3 Stance-Stance Transfer

The inspiration and physical motivation for this approach came from observing fast climbing animals such as the gecko which swing their tails and flex their backs as they run. One advantage of these motions may be that they shift the phasing of the power distribution to the beginning of a stride when it is needed most to accelerate a body after slowing for foot contact. A proper model for this behavior would include multiple bodies and degrees of freedom. Here we hypothesize that at least part of the effect of these motions can be captured by the linear body spring as shown in Fig. 6. The spring is loaded during the end of the stance phase as the robot slows for attachment and then is released at the beginning of the next stride to assist with the re-acceleration of the body.

The net effect of this body/tail spring is to lower the peak torque spike during stance. This in turn allows us to further change the gear ratio, reducing the maximum continuous torque limit and increasing the stride frequency. With the addition of this spring the equation of motion for the body during stance becomes:

$$\ddot{x} = \frac{1}{m}\left(min\left(\frac{G^2\dot{x} - G\dot{x}_{nl}}{-k_e}, GF_{Max}\right) + k(X_F - x) - mg\right) \tag{5}$$

where X_F, the rest length of the spring, is located at the midpoint of stance.

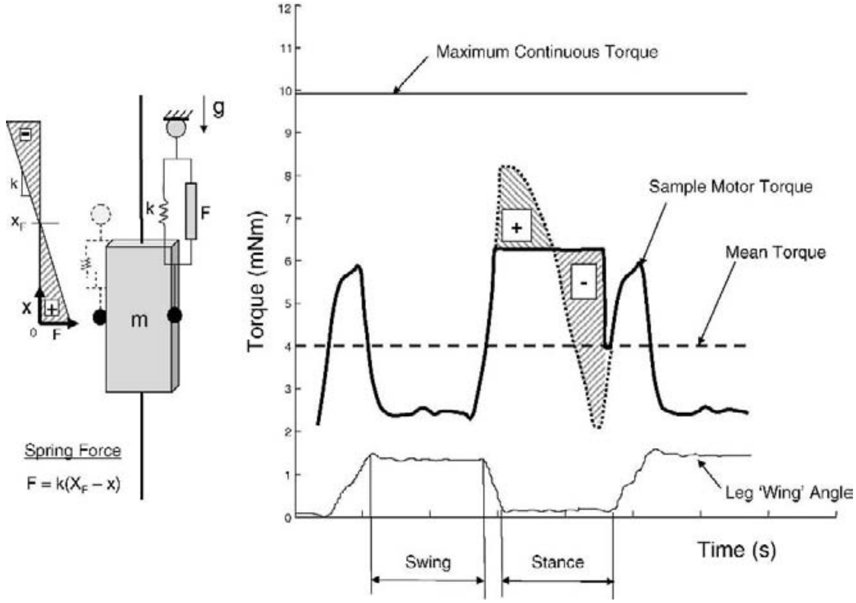


Fig. 6. Schematic of model with body spring and the effect of the spring on the nominal torque profile. The "+" region represents when the spring is assisting the motor, and the "−" region represents when the motor is stretching the spring. (As before, the leg trajectory and mean torque plots are based on RiSE robot data [3])

The maximum stiffness of the virtual leg is limited by the force available from the motors (F_{Max}), as given by (6) where (p) is the number of motors.

$$k \leq \frac{G\, F_{Max}\, p}{2R} \tag{6}$$

Magnitudes ranging from 0-80% of the maximum force were evaluated. The simulation results with this spring are shown in Fig. 7.

As in Fig. 5 the areas under the curved lines represent speeds for which the stance phase displacement requirement is not satisfied, and the starred diagonal line represents the swing phase requirement. Points (A) and (B) are the same as in Fig. 5. Point (C) represents the maximum speed with the body spring which yields a 16% improvement over trajectory refinement alone, case (B).

It appears that the use of such a body spring increases the maximum possible speed for a given G, but does not lower the gear ratio which is necessary to overcome gravity and lift the robot. Thus the use of a body spring to some

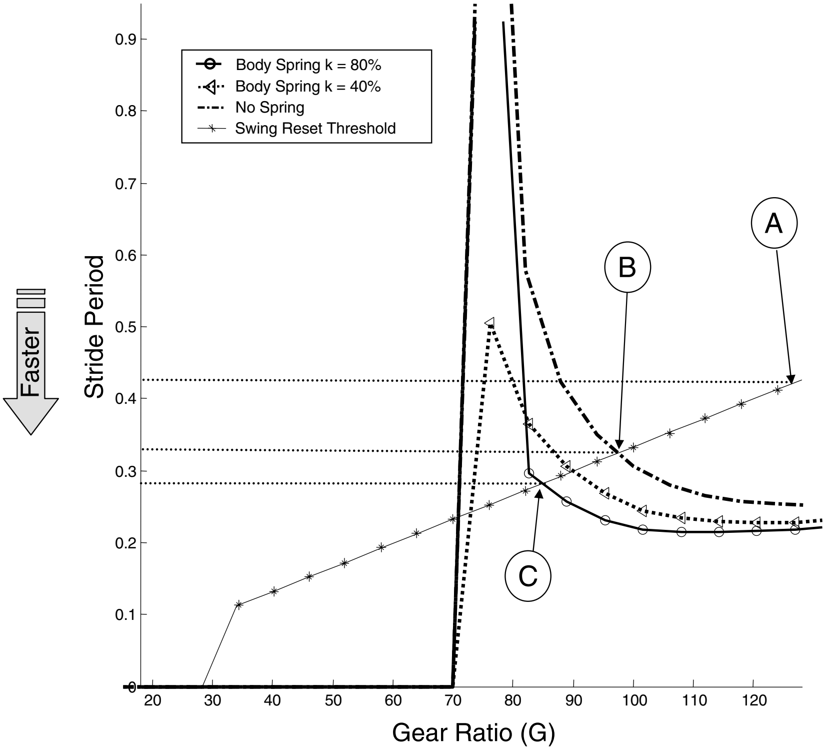


Fig. 7. Simulation results for the model with the body spring. Minimum possible stride periods for various gear ratios (G) and spring constants (k)

degree duplicates the benefit from implementing a stride-to-stride velocity adaptation scheme.

3.4 Swing-Stance Transfer

A second approach to using springs to more effectively and evenly apply the torque from the motors is to use a spring connecting the foot to the body. As shown on the left in Fig. 8, this is modeled as a spring acting in parallel to the actuator in each virtual leg. This spring is loaded during swing phase as the leg resets to a position relative to the body ready for touch down. The spring is then released at the beginning of stance to assist with the acceleration of the body. As shown on the right in Fig. 8, this adds a load to the motors during swing (when their torque output capabilities are currently underutilized) and mitigates the force requirements at the beginning of stance.

The addition of this spring results in the same body equation of motion as in Sect. 3.3, but the spring is now fully loaded at the beginning of stance, and is fully unloaded at the end. The spring constant, k, is chosen in the same manner as in the previous section.

Figure 9 shows the effect of changing the gear ratio G on the stride period for a range of spring constants from 0–80% of the maximum spring constant for each G as given by (6). For each value of G and k, the resulting minimum

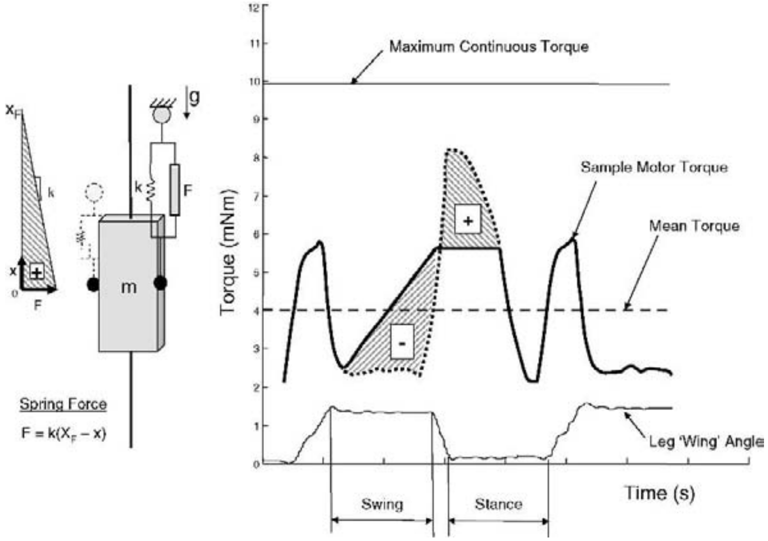


Fig. 8. Schematic of model with leg spring and the effect of the spring on the nominal torque profile. The "+" region represents when the spring is assisting the motor, and the "–" region represents when the motor is stretching the spring. (As before, the leg trajectory and mean torque plots are based on RiSE robot data [3])

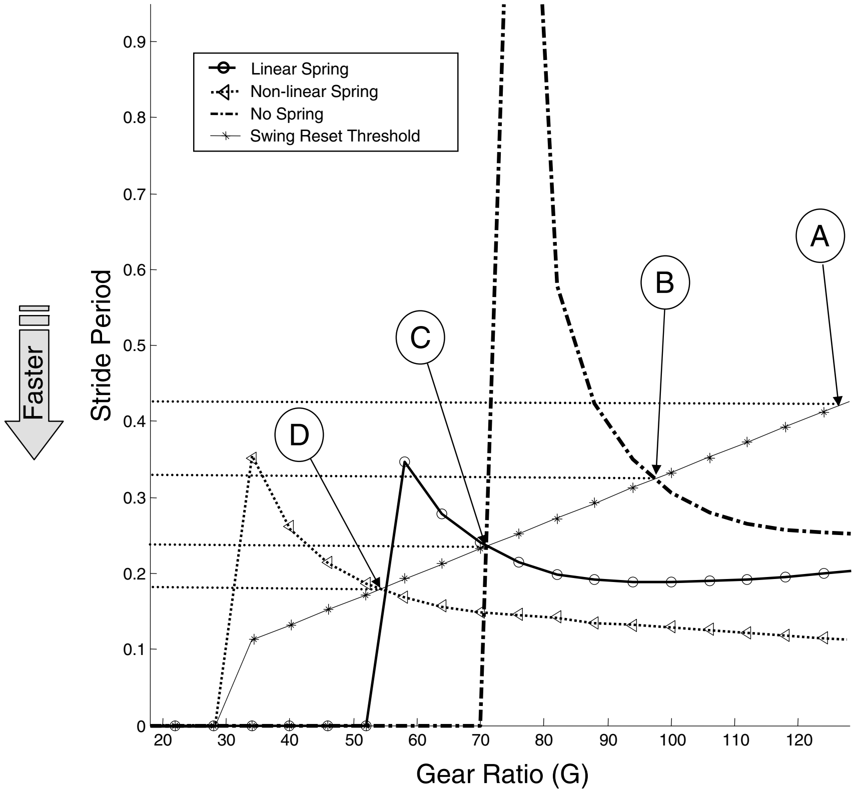


Fig. 9. Simulation results for the model with the leg spring. Minimum possible stride periods for various gear ratios (G) and spring constants (k)

stride period is shown. As before, the line corresponding to the stride period limit for retraction of the leg during swing is indicated with a starred line.

Points (A) and (B) are the same as in Fig. 5 and represent the maximum possible speed without springs. Point (C) indicates the maximum speed configuration for a model with a linear spring with $k = 80\%$ of maximum. The use of a linear spring increases the maximum speed by 36% to 0.62 m/s.

Softening Spring

While advantageous for their simplicity, linear springs are not optimal in terms of energy storage for a limited stretching force. If the linear spring were replaced by a softening spring the spring potential energy at maximum deflection would increase. In the limit, with a constant force spring, the energy storage would double. A constant force spring, however, would add a large load to the beginning of the swing phase when the motors may already be saturated attempting to accelerate the leg. The leg spring analysis in the previous section was repeated with the linear spring replaced by one with a spring equation of: $F = kx^{\frac{1}{2}}$. This significantly increases the energy storage for the spring with minimal interference with the acceleration of the leg in swing. Point (D) in Fig. 9 represents the fastest configuration that can be achieved with this softening spring, resulting in a speed increase to 0.84 m/s.

Retraction Dynamics

In this swing-to-stance spring approach the load during the swing phase is significantly increased, therefore it becomes important to determine what happens when the leg dynamics are explicitly considered. In other words, at what point will the inertia of the legs and the frictional losses in the springs erode any benefit from running dynamically or adding leg springs.

To this end the dynamics of the swing legs were modeled with (7).

$$\ddot{x_f} = \frac{1}{m_2}\left(min\left(\frac{G^2\dot{x_f} - G\dot{x_f}_{nl}}{-k_e}, GF_{Max}\right) + k(X_{body} - x_f) - b\dot{x_f}m_2g\right) \quad (7)$$

Where m_2 is the effective inertia of the robot's leg and b is the damping term, as given by (8).

$$b = 2\zeta\sqrt{mk} \quad (8)$$

While both adding inertia to the legs and increasing the losses in the spring detract from the gains suggested by the simulation, the model can still climb with a spring constant, $k = 80\%$ of maximum, at speeds equivalent to what the simple swing phase model predicts with leg inertias of 30% of the bodymass and a damping coefficient $\zeta = 0.3$.

3.5 Results of Numerical Study

Table 1 summarizes the cases considered thus far and gives the maximum frequency, speed, and percent improvement for each case. With a force-optimization series of trajectory refinements the robot's speed can (theoretically) be significantly improved. Obviously real-world issues associated with mechanical losses and foot attachment/detachment will prevent the actual achievement of the 3.05 Hz theoretical speed predicted from implementing trajectory refinements.

The last row in the table (Combination) shows the effect of combining the body and softening foot springs and allowing the body velocity to increase from stride to stride (momentum), which results in a 2.9x improvement over the trajectory refinement case alone. Even if drag and attachment losses only permitted achieving 40% of the theoretical speed limit, with these changes we get near the 3.5 Hz dynamic threshold that we established previously. Of the various elements, the non-linear foot spring contributes the most.

Another option to improve the speed of the robot climbing is to alter its power/weight ratio. As a point of comparison the motor specs for larger motors

Table 1. Frequency, velocity, and percent improvement for cases (B) to (G)

Case	Frequency	Velocity (m/s)	% Impr.
(A) Trajectory Refinement	3.05	0.46	0
(B) Body Momentum	3.66	0.55	20
(C) Body Spring	3.55	0.53	16
(D) Foot Spring-Linear	4.15	0.62	36
(E) Foot Spring-Softening	5.59	0.84	83
(F) Body + Foot Springs	5.78	0.87	89
(G) Combination	8.85	1.33	190

from the same vendor and product line were used in the model to see how much the power needed to be increased to match the effect of adding springs and dynamic gaits. In this case the 4.5 W motors for RiSE were replaced with 20 W versions. The total mass of the robot was left unchanged and the simulations repeated. This 4.4x increase in power resulted in a 2.3x increase in speed. A net increase slightly less than with springs/dynamics.

In reality the use of larger motors brings with it a significant increase in mass and complexity. The addition of these larger motors would add additional 1.5 kg to the robot's mass, not including the necessary changes to the body, battery, and electronics design. Fundamentally, increasing the size of the motors does not substantially increase their power to weight ratio. Other motors do have higher power/weight ratios than the ones chosen for our robot, but these suffer from other draw backs such as controllability.

Of course reducing the weight of the robot, were we able to find a way to do it without loosing performance, would help as well. The simulations with the simple model described here suggest that gains in speed comparable to the use of springs or the addition of (magically) more powerful motors can be achieved by reducing the robot's weight by about 50%.

In the absence of further improvements to the power/weight ratio of our robots, the simulations suggest that with our current design it is not possible to reach our target speed. Simply refining the gait and reducing the inefficiencies in the system will, by themselves, be insufficient. With an appropriate use of springs and body dynamics, however, our target speed of 3.5 Hz becomes theoretically possible. The actual realization of these speeds, however, depends on how well these concepts can be incorporated with the ongoing work in improving trajectory refinement and foot attachment.

4 Conclusion

One of the significant problems in achieving fast climbing is the power demand associated with delivering the required work against gravity at higher speeds. Having sufficient on-board power for fast locomotion on the level ground has proven challenging, and motion against gravity, obviously, only exacerbates the problem. One approach to increasing the use of a robot's onboard power is switch from a position-based control scheme to forced-based approach. By explicitly regulating the motors' output rather than relying on position tracking errors, the actuators can be much more effectively utilized during a stride. This adoption of this control framework also enables the robot to build up speed over a number of strides, further increasing the performance gains.

Even if the switching the fundamental control scheme from a position-based approach proves infeasible (e.g., perhaps because our limited degrees of freedom require intricately planned approaches and departures from stance to guaranteed adequate wall adhesion and limited perturbation during detachment), much of the advantage of a force-based system could be duplicated by very careful trajectory tuning and adaptation.

In either case the refinement of the force trajectory can bring substantial performance benefits to the robot. The simple model employed here suggests that as a theoretical upper limit they could allow the robot to climb at 3Hz or at 0.46 m/s, which is near our dynamic threshold. Of course attachment, detachment and other physical constraints will necessarily reduce the actual gains.

However, in this respect the remainder of the numerical results from the simplified model are encouraging. They suggest that the appropriate addition of body and leg springs could double the robot's speed over this value. The further incorporation of a variable stride period could almost triple the speed over trajectory refinement alone. This is more than the effect of increasing the motor's power 4.4x! (that is of course without acknowledging and accounting for the weight of the larger motors). While the advantages of these approaches are not entirely "free", they do represent a significant gain. When the various refinements are combined, the model results suggest that locomotion at our dynamic threshold of 3.5 Hz (or 2 bodylengths/second) should be achievable.

4.1 Future Work

In order to implement body dynamic dependent gait trajectories some sort of control system to measure body velocity and alter the leg trajectory may be necessary to ensure good attachment of the feet. More detailed foot/substrate interaction tests may provide the empirical data necessary to develop such a controller.

Although we have assumed that foot contact once made will only be broken at the desired detachment point, this clearly does not reflect the reality. Reducing the demands on the foot attachment mechanism could be one of the major advantages of dynamic climbing. It seems possible that an appropriate use of springs and the robot's body's dynamics could significantly reduce the required foot reaction forces. If force threshold limits were added to the feet the effectiveness of the various schemes proposed in this paper could be evaluated in this regard.

Although not addressed here, the stability of dynamic climbers is a topic of interest. Besides the fundamental issues of insuring that the front feet stay attached to the wall, there are a number of other possible ways to consider stability. Many of these arise with the shift from a position-based control to a force-based system. The numeric simulation results suggest that when the velocity is allowed to vary from stride to stride that the simple climber tends to quickly converge to a steady period-1 gait. Have we been fortuitous in our parameter selection, or are these limit cycles almost inevitable? Are they local in nature hence hard to achieve in practice or do they have large basins (e.g., are they globally asymptotic stable)? The model we have used may be simple enough to permit a careful mathematical analysis of the system dynamics.

A second interesting question that arises from decoupling multiple limbs from a trajectory-tracking control scheme is the question of synchronization. A related climbing study with an open-loop climbing model [17] indicates that legs tend to synchronize rather than staying 180 degrees out of phase. Is this also true for this model, and if so what sort of controller needs to be established to maintain a regular alternating gait?

Looking further ahead, we wonder if with the addition of a lateral degree of freedom to the model we can begin to duplicate the motions and ground reaction forces seen in dynamic climbing animals such as geckos and cockroaches.

We believe that enabling a robot with the ability to both dynamicly run and climb is an compelling goal. The achievement of both with an (inherently) constrained power/weight ratio is a difficult task. The creative use of springs and system dynamics to modify the climbing motion of the robot may enable the construction of such robots. While we are not there yet, we at least have some simple models that suggest that it may be possible.

Acknowledgments

Thanks to RiSE team for many stimulating conversations, specifically: Ron Fearing, Martin Buehler for their thoughts on motors and power in climbing, Jonathan Karpick for ideas about how to use dynamics in climbing, Dan Goldman for discussions about how biological systems climb, and Haldun Komsuoglu for insights about modeling simple climbers. We also appreciate Clark Haynes and Al Rizzi's help with the robot trajectory and power consumption data. The development of the RiSE platform was supported by DARPA under grant number N66001-05-C-8025. Jonathan Clark is supported by the IC Postdoctoral Fellow Program under grant number HM158204-1-2030.

Appendix A1: Rise Robot Constants

Table 2. RiSE specific model parameter values

Variable	Value	Description
T_{Stall}	0.0288 Nm	Stall torque
T_{Max}	0.00498 Nm	Maximum continuous torque
ω_{nl}	13,900 rpm	No-load speed
R	0.0762 m	Radius of foot trajectory
G	126	Base Gear ration
p	6	Number of motors per tripod
m	3 kg	Mass of the robot

Appendix A2: Minimum Leg Frequency

In order to achieve resonant hopping as described by the spring-loaded inverted pendulum (SLIP) model, the motor activation frequency–whether used primarily for recirculating vertical leg springs as in RHex [6] or for powering vertical leg strokes in phase with passive recirculating springs as in Sprawl [14]–should match the natural frequency of the body's oscillation. For a SLIP-type hopper the natural frequency, ω_n, during stance is a function of the body mass, M, and stiffness of the legs, k, that varies in a rather subtle manner with the particular steady state gait for even the simplest two degree of freedom models [5]. Empirically, we find this function is effectively approximated by that characterizing a one degree of freedom spring-mass system:

$$\omega_n = \sqrt{\frac{k}{M}} \tag{9}$$

The lower limit on the spring stiffness is constrained by its maximum displacement, Δx, which in turn is fixed by the leg kinematics. Although the force-extension profile of a spring can vary significantly depending upon whether it is "hardening" or "softening", it will suffice for our present order-of-magnitude analysis to consider the simplest relationship of constant stiffness arising from a Hooke's law spring. For this model, a lower bound on the excursion of the leg spring corresponds to when the force on the spring is equal to gravity, giving:

$$k = \frac{Mg}{\Delta x} \tag{10}$$

Combining equations (9) and (10) gives:

$$\omega_n = \sqrt{\frac{k}{M}} = \sqrt{\frac{Mg}{\Delta x\ M}} = \sqrt{\frac{g}{\Delta x}}$$

For RiSE with a kinematically achievable $\Delta x = 0.02$ m:

$$\omega_n = 22\ rad/s = 3.5\ cycles/s$$

If large airborne phases are allowed the body oscillation frequency would become slower than the body spring-mass frequency, ω_n. Any gains from this, however, would be set off by the increased required deflection of the spring, Δx, during stance.

References

[1] http://www.vortexhc.com/vmrp.html.

[2] http://www.maxonmotorusa.com.

[3] Personal Communication from Al Rizzi and Clark Haynes.

[4] R. McN. Alexander. Mechanics and scaling of terrestrial locomotion. In T. J. Pedley, editor, *Scale Effects in Animal Locomotion*, pp. 93–110. Academic Press, 1977.

[5] R. Altendorfer, D. E. Koditschek, and P. Holmes. Stability analysis of legged locomotion models by symmetry-factored return mpas. *International Jouranl of Robotics Research*, 23(11):979–1000, 2004.

[6] R. Altendorfer, N. Moore, H. Komsuoglu, M. Buehler, Jr. Brown H. B, D. McMordie, U. Saranli, R. Full, and D. E. Koditschek. Rhex: A biologically inspired hexapod runner. *Autonomous Robots*, 11(3):207–213, 2001.

[7] A. T. Asbeck, S. Kim, W. R. Provancher, M. R. Cutkosky, and M. Lanzetta. Scaling hard vertical surfaces with compliant microspine arrays. In *Proceedings, Robotics Science and Systems Conf., MIT, Cambridge, MA, June 8–10*, 2005.

[8] K. Autumn, M. Buehler, M. Cutkosky, R. Fearing, R. J. Full, D. Goldman, R. Groff, W. Provancher, A. A. Rizzi, U. Saranli, A. Saunders, and D. E. Koditschek. Robotics in scansorial environments. In Grant R. Gerhart, Charles M. Shoemaker, and Douglars W. Gage, editors, *Unmanned Ground vehicle Technology VII*, vol. 5804, pp. 291–302. SPIE, 2005.

[9] K. Autumn, S. T. Hsieh, D. M. Dudek, J. Chen, Chitaphan. C., and R. J. Full. Dynamics of geckos running vertically. 2005. in press.

[10] C. Balaguer, A. Gimenez, J. M. Pastor, V. M. Padron, and C. Abderrahim. A climbing autonomous robot for inspection applications in 3d complex environments. *Robotica*, 18:287–297, 2000.

[11] D. Bevly, S. Dubowsky, and C. Mavroidis. A simplified cartesian-computed torque controller for highly geared systems and its application to an experimental climbing robot. *Transactions of the ASME. Journal of Dynamic Systems, Measurement and Control*, 122(1):27–32, 2000.

[12] T. Bretl, S. Rock, and J. C. Latombe. Motion planning for a three-limbed climbing robot in vertical natural terrain. In *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 205, pp. 2946–2953. Piscataway, NJ, USA: IEEE, 2003.

[13] G. A. Cavagna, N. C. Heglund, and Taylor C. R. Mechanical work in terrestrial locomotion: Two basic mechanisms for minimizing energy expenditure. *American Journal of Physiology*, 233, 1977.

[14] J. G. Cham, S. A. Bailey, J. E. Clark, R. J. Full, and M. R. Cutkosky. Fast and robust: Hexapedal robots via shape deposition manufacturing. *International Journal of Robotics Research*, 21(10), 2002.

[15] J. G. Cham, J. Karpick, J. E. Clark, and M. R. Cutkosky. Stride period adaptation for a biomimetic running hexapod. In *International Symposium of Robotics Research*, Lorne Victoria, Australia, 2001.

[16] S. Kim, A Asbeck, M. R. Cutkosky, and W. R. Provancher. Spinybotii: Climbing hard walls with compliant microspines. In *Proceedings, IEEE – ICAR, Seattle, WA, July 17–20*, 2005.

[17] Haldun Komsuoglu. *Toward a formal framework for open-loop stabilization of rhythmic tasks*. PhD thesis, University of Michigan, 2004.

[18] R. Kram, A. Domingo, and D. P. Ferris. Effect of reduced gravity on the preferred walk-run transition speed. *The Journal of Experimental Biology*, 200:821–826, 1991.

[19] G. La Rosa, M. Messina, G. Muscato, and R. Sinatra. A low-cost lightweight climbing robot for the inspection of vertical surfaces. *Mechatronics*, 12(1):71–96, 2002.

[20] T. A MCMahon and G. C. Cheng. Mechanics of running. how does stiffness couple with speed? *Journal of Biomechanics*, 23(1):65–78, 1990.

[21] R. D. Quinn, G. M. Nelson, R. J. Bachmann, D. A. Kingsley, J. Offi, and R. E. Ritzmann. Insect designs for improved robot mobility. In Dillmann Berns, editor, *Proceedings of CLAWAR 2001, Karlsruhe, Germany*, pp. 69–76, 2001.

[22] M. Sitti and R. S. Fearing. Synthetic gecko foot-hair micro/nano-structures for future wall-climbing robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 1, pp. 1164–1170. Piscataway, NJ, USA: IEEE, 2003.

[23] Z. L. Xu and P. S. Ma. A wall-climbing robot for labelling scale of oil tank's volume. *Robotica*, 20:209–212, 2002.

Fast Direct Multiple Shooting Algorithms for Optimal Robot Control

M. Diehl[1], H.G. Bock[1], H. Diedam[1], and P.-B. Wieber[2]

[1] Interdisciplinary Center for Scientific Computing, University of Heidelberg, Im Neuenheimer Feld 368, D-69120 Heidelberg, Germany
m.diehl@iwr.uni-heidelberg.de

[2] INRIA Rhône-Alpes, Projet BIPOP, 38334 St Ismier Cedex, France

Summary. In this overview paper, we first survey numerical approaches to solve nonlinear optimal control problems, and second, we present our most recent algorithmic developments for real-time optimization in nonlinear model predictive control.

In the survey part, we discuss three direct optimal control approaches in detail: (i) single shooting, (ii) collocation, and (iii) multiple shooting, and we specify why we believe the direct multiple shooting method to be the method of choice for nonlinear optimal control problems in robotics. We couple it with an efficient robot model generator and show the performance of the algorithm at the example of a five link robot arm. In the real-time optimization part, we outline the idea of nonlinear model predictive control and the real-time challenge it poses to numerical optimization. As one solution approach, we discuss the *real-time iteration scheme.*

1 Introduction

In this paper, we treat the numerical solution of optimal control problems. We consider the following simplified optimal control problem in ordinary differential equations (ODE).

$$\underset{x(\cdot),u(\cdot),T}{\text{minimize}} \quad \int_0^T L(x(t),u(t))\,dt \;+\; E\left(x(T)\right) \tag{1}$$

subject to

$$\begin{aligned}
x(0)-x_0 &= 0, && && \text{(fixed initial value)}\\
\dot{x}(t)-f(x(t),u(t)) &= 0, && t\in[0,T], && \text{(ODE model)}\\
h(x(t),u(t)) &\geq 0, && t\in[0,T], && \text{(path constraints)}\\
r\left(x(T)\right) &= 0 && && \text{(terminal constraints).}
\end{aligned}$$

The problem is visualized in Fig. 1. We may or may not leave the horizon length T free for optimization. As an example we may think of a robot that

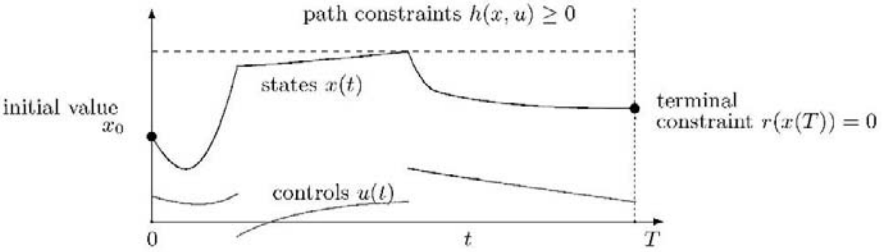

path constraints $h(x, u) \geq 0$
states $x(t)$
initial value x_0
terminal constraint $r(x(T)) = 0$
controls $u(t)$
0
t
T

Fig. 1. Simplified Optimal Control Problem

shall move in minimal time from its current state to some desired terminal position, and must respect limits on torques and joint angles. We point out that the above formulation is by far not the most general, but that we try to avoid unneccessary notational overhead by omitting e.g. differential algebraic equations (DAE), multi-phase motions, or coupled multipoint constraints, which are, however, all treatable by the direct optimal control methods to be presented in this paper.

1.1 Approaches to Optimal Control

Generally speaking, there are three basic approaches to address optimal control problems, (a) dynamic programming, (b) indirect, and (c) direct approaches, cf. the top row of Fig. 2.

(a) Dynamic Programming [5, 6] uses the principle of optimality of subarcs to compute recursively a feedback control for all times t and all x_0. In the continuous time case, as here, this leads to the Hamilton-Jacobi-Bellman (HJB) equation, a partial differential equation (PDE) in state space. Methods to numerically compute solution approximations exist, e.g. [34] but the approach severely suffers from Bellman's "curse of dimensionality" and is restricted to small state dimensions.

(b) Indirect Methods use the necessary conditions of optimality of the infinite problem to derive a boundary value problem (BVP) in ordinary differential equations (ODE), as e.g. described in [13]. This BVP must numerically be solved, and the approach is often sketched as "first optimize, then discretize". The class of indirect methods encompasses also the well known calculus of variations and the Euler-Lagrange differential equations, and the Pontryagin Maximum Principle [40]. The numerical solution of the BVP is mostly performed by shooting techniques or by collocation. The two major drawbacks are that the underlying differential equations are often difficult to solve due to strong nonlinearity and instability, and that changes in the control structure, i.e. the sequence of arcs where different constraints are active, are difficult to handle: they usually require a completely new problem setup. Moreover, on so called *singular arcs*, higher index DAE arise which necessitate specialized solution techniques.

(c) Direct methods transform the original infinite optimal control problem into a finite dimensional *nonlinear programming problem (NLP)*. This NLP is then solved by variants of state-of-the-art numerical optimization methods, and the approach is therefore often sketched as "first discretize, then optimize". One of the most important advantages of direct compared to indirect methods is that they can easily treat inequality constraints, like the inequality path constraints in the formulation above. This is because structural changes in the active constraints during the optimization procedure are treated by well developed NLP methods that can deal with inequality constraints and active set changes. All direct methods are based on a finite dimensional parameterization of the control trajectory, but differ in the way the state trajectory is handled, cf. the bottom row of Fig. 2.

For solution of constrained optimal control problems in real world applications, direct methods are nowadays by far the most widespread and successfully used techniques, and we will focus on them in the first part of this paper.

1.2 Nonlinear Model Predictive Control

The optimization based feedback control technique "Nonlinear Model Predictive Control (NMPC)" has attracted much attention in recent years [1, 36],

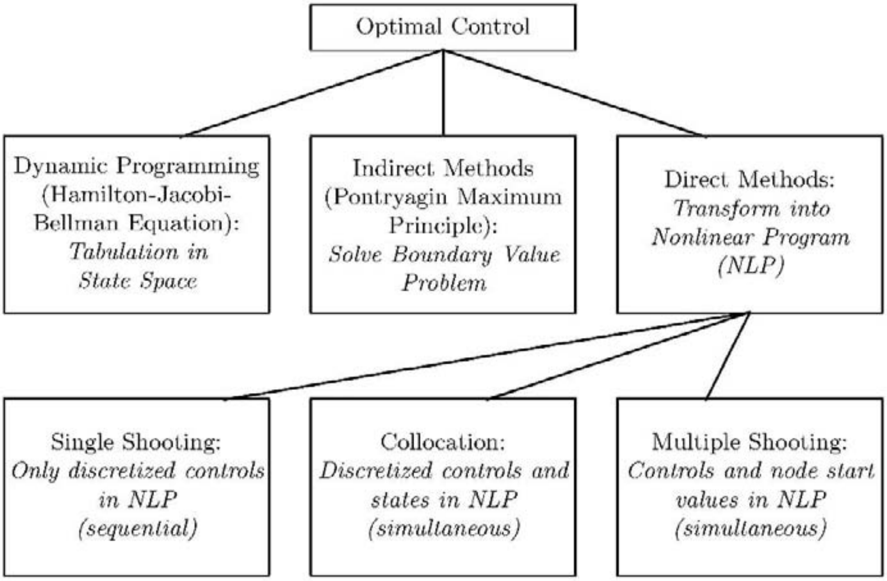

Optimal Control
Dynamic Programming (Hamilton-Jacobi-Bellman Equation): *Tabulation in State Space*
Indirect Methods (Pontryagin Maximum Principle): *Solve Boundary Value Problem*
Direct Methods: *Transform into Nonlinear Program (NLP)*
Single Shooting: *Only discretized controls in NLP (sequential)*
Collocation: *Discretized controls and states in NLP (simultaneous)*
Multiple Shooting: *Controls and node start values in NLP (simultaneous)*

Fig. 2. Overview of numerical methods for optimal control

in particular in the proceess industries. Its idea is, simply speaking, to use an open-loop optimal control formulation to generate a feedback control for a closed-loop system. The current system state is continuously observed, and NMPC solves repeatedly an optimal control problem of the form (1), each time with the most current state observation as initial value x_0. Assuming that the optimal control trajectory can be computed in negligible time, we can apply the first bit of our optimal plan to the real world system, for some short duration δ. Then, the state is observed again, a new optimization problem is solved, the control again applied to the real system, and so on. In this way, feedback is generated that can reject unforeseen disturbances and errors due to model-plant-mismatch.

Among the advantages of NMPC when compared to other feedback control techniques are the flexibility provided in formulating the control objective, the capability to directly handle equality and inequality constraints, and the possibility to treat unforeseen disturbances fast. Most important, NMPC allows to make use of reliable nonlinear process models $\dot{x} = f(x, u)$ so that the control performance can profit from this important knowledge, which is particularly important for transient, or periodic processes. It is this last point that makes it particularly appealing for use in robotics.

One essential problem, however, is the high on-line computational load that is often associated with NMPC, since at each sampling instant a nonlinear optimal control problem of the form (1) must be solved. The algorithm must predict and optimize again and again, in a high frequency, while the real process advances in time. Therefore, the question of fast *real-time optimization* has been intensively investigated [4, 28, 51, 44, 9]. We refer to Binder et al. [10] for an overview of existing methods. One reason why most applications of NMPC have so far been in the process industries [42] is that there, time scales are typically in the range of minutes so that the real-time requirements are less severe than in mechanics. However, we believe that it is only a matter of time until NMPC becomes an important feedback technique in robotics, too. The second scope of this paper is therefore to present some of our latest ideas regarding the fast real-time optimization for NMPC, which are based on direct optimal control methods.

1.3 Paper Outline

The paper is organized as follows. In the next section we will describe three popular direct optimal control methods, single shooting, collocation, and multiple shooting. We will argue why we believe the last method to be the method of choice for nonlinear optimal control problems in robotics, and in Sect. 3 we will present its coupling to an efficient robot model generator and show its application to the time optimal point to point maneuver of a five link robot arm. In Sect. 4 we will discuss nonlinear model predictive control (NMPC) and show how the challenge of fast online optimization can be addressed by the so called *real-time iteration* scheme, in order to make NMPC of fast robot motions possible. Finally, in Sect. 5, we conclude the paper with a short summary and an outlook.

2 Direct Optimal Control Methods

Direct methods reformulate the infinite optimal control problem (1) into a finite dimensional *nonlinear programming problem (NLP)* of the form

$$\min_{w} a(w) \quad \text{subject to} \quad b(w) = 0, \quad c(w) \geq 0\ , \tag{2}$$

with a finite dimensional vector w representing the optimization degrees of freedom, and with differentiable functions a (scalar), b, and c (both vector valued). As said above, all direct methods start by a parameterization of the control trajectory, but they differ in the way how the state trajectory is handled. Generally, they can be divided into *sequential* and *simultaneous* approaches.

In *sequential* approaches, the state trajectory $x(t)$ is regarded as an implicit function of the controls $u(t)$ (and of the initial value x_0), e.g. by a forward simulation with the help of an ODE solver in *direct single shooting* [31, 45]. Thus, simulation and optimization iterations proceed sequentially, one after the other, and the NLP has only the discretized control as optimization degrees of freedom.

In contrast to this, *simultaneous* approaches keep a parameterization of the state trajectory as optimization variables within the NLP, and add suitable equality constraints representing the ODE model. Thus, simulation and optimization proceed simultaneously, and only at the solution of the NLP do the states actually represent a valid ODE solution corresponding to the control trajectory. The two most popular variants of the simultaneous approach are *direct collocation* [8] and *direct multiple shooting* [12].

We will present in detail the mentioned three direct approaches. As all direct methods make use of advanced NLP solvers, we also very briefly sketch one of the most widespread NLP solution methods, Sequential Quadratic Programming (SQP), which is also at the core of the real-time iteration scheme to be presented in the second part.

A Tutorial Example

For illustration of the different behaviour of sequential and simultaenous approaches, we will use the following tutorial example with only one state and one control dimension. The ODE $\dot{x} = f(x, u)$ is slightly unstable and nonlinear.

$$\underset{x(\,\cdot\,),\,u(\,\cdot\,)}{\text{minimize}} \qquad \int\limits_{0}^{3} x(t)^2 + u(t)^2 \; dt$$

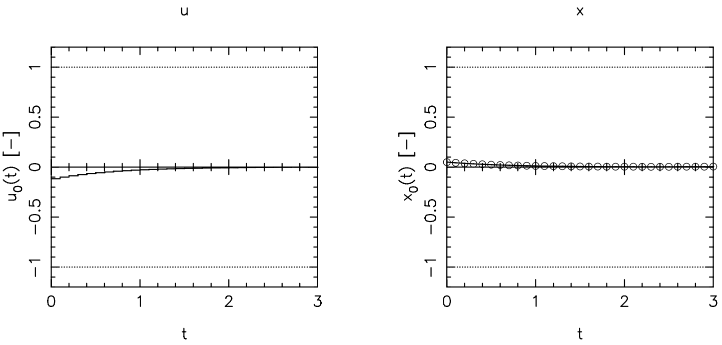


Fig. 3. Solution of the tutorial example

subject to

$$\begin{aligned}
x(0) &= x_0, && \text{(initial value)}\\
\dot{x} &= (1+x)x+u, \quad t\in[0,3], && \text{(ODE model)}\\
\begin{bmatrix}1-x(t)\\1+x(t)\\1-u(t)\\1+u(t)\end{bmatrix} &\geq \begin{bmatrix}0\\0\\0\\0\end{bmatrix}, \quad t\in[0,3], && \text{(bounds)}\\
x(3) &= 0. && \text{(zero terminal constraint).}
\end{aligned}$$

We remark that due to the bounds $|u|\leq 1$, we have uncontrollable growth for any $x\geq 0.618$ because then $(1+x)x\geq 1$. We set the inital value to $x_0 = 0.05$. For the control discretization we will choose $N = 30$ control intervals of equal length. The solution of this problem is shown in Fig. 3.

2.1 Sequential Quadratic Programming (SQP)

To solve any NLP of the form (2), we will work within an iterative *Sequential Quadratic Programming (SQP)*, or *Newton-type* framework. We omit all details here, and refer to excellent numerical optimization textbooks instead, e.g. [39]. We need to introduce, however, the *Lagrangian function*

$$\mathcal{L}(w,\lambda,\mu) = a(w) - \lambda^T b(w) - \mu^T c(w) \; ,$$

with so called *Lagrange multipliers* λ and μ, that plays a preeminent role in optimization. The necessary conditions for a point w^* to be a local optimum of the NLP (2) are that there exist multipliers λ^* and μ^*, such that

$$\nabla_w \mathcal{L}(w^*,\lambda^*,\mu^*) = 0 \; , \tag{3}$$

$$b(w^*) = 0 \; , \tag{4}$$

$$c(w^*) \geq 0, \quad \mu^* \geq 0, \quad c(w^*)^T\mu^* = 0 \; . \tag{5}$$

In order to approximately find such a triple (w^*, λ^*, μ^*) we proceed iteratively. Starting with an initial guess (w_0, λ_0, μ_0), a standard full step SQP iteration for the NLP is

$$w_{k+1} = w_k + \Delta w_k \ , \tag{6}$$

$$\lambda_{k+1} = \lambda_k^{\mathrm{QP}}, \quad \mu_{k+1} = \mu_k^{\mathrm{QP}} \ , \tag{7}$$

where $(\Delta w_k, \lambda_k^{\mathrm{QP}}, \mu_k^{\mathrm{QP}})$ is the solution of a quadratic program (QP). In the classical Newton-type or SQP approaches, this QP has the form

$$\begin{array}{ll} \underset{\Delta w \,\in\, \mathbb{R}^{n_w}}{\min} & \frac{1}{2} \ \Delta w^T \, A_k \, \Delta w \quad + \quad \nabla_w a(w_k)^T \Delta w \\ \text{subject to} & \begin{cases} b(w_k) + \nabla_w b(w_k)^T \Delta w = 0 \\ c(w_k) + \nabla_w c(w_k)^T \Delta w \geq 0 \end{cases} \end{array} \tag{8}$$

where A_k is an approximation of the Hessian of the Lagrangian,

$$A_k \approx \nabla_w^2 \mathcal{L}(w_k, \lambda_k, \mu_k) \ ,$$

and $\nabla_w b(w_k)^T$ and $\nabla_w c(w_k)^T$ are the constraint Jacobians. Depending on the quality of the Hessian approximation we may expect linear, super-linear or even quadratic convergence. Practical SQP methods differ e.g. in the type of globalisation strategy, in the type of QP solver used, or in the way the Hessian is approximated – e.g. by BFGS updates or by a Gauss-Newton Hessian. This last approach is favourable for least squares problems, as e.g in tracking or estimation problems. When the objective is given as $a(w) = \|r(w)\|_2^2$, the Gauss-Newton Hessian is given by $A_k = 2\nabla_w r(w_k) \nabla_w r(w_k)^T$. It is a good approximation of the exact Hessian $\nabla_w^2 \mathcal{L}(w_k, \lambda_k, \mu_k)$ if the residual $\|r(w)\|_2^2$ is small or if the problem is only mildly nonlinear.

2.2 Direct Single Shooting

The single shooting approach starts by discretizing the controls. We might for example choose grid points on the unit interval, $0 = \tau_0 < \tau_1 < \ldots < \tau_N = 1$, and then rescale these gridpoints to the possibly variable time horizon of the optimal control problem, $[0, T]$, by defining $t_i = T\tau_i$ for $i = 0, 1, \ldots, N$. On this grid we discretize the controls $u(t)$, for example piecewise constant, $u(t) = q_i$ for $t \in [t_i, t_{i+1}]$, so that $u(t)$ only depends on the the finitely many control parameters $q = (q_0, q_1, \ldots, q_{N-1}, T)$ and can be denoted by $u(t; q)$. If the problem has a fixed horizon length T, the last component of q disappears as it is no optimization variable. Using a numerical simulation routine for solving the initial value problem

$$x(0) = x_0, \quad \dot{x}(t) = f(x(t), u(t; q)), \quad t \in [0, T] \ ,$$

we can now regard the states $x(t)$ on $[0, T]$ as dependent variables, cf. Fig. 4. We denote them by $x(t; q)$. The question which simulation routine should be

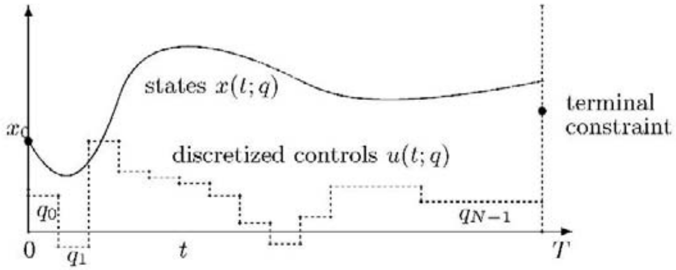


Fig. 4. Illustration of single shooting

chosen is crucial to the success of any shooting method and depends on the type of ODE model. It is essential to use an ODE solver that also delivers sensitivities, as they are needed within the optimization. We also discretize the path constraints to avoid a semi-infinite problem, for example by requiring $h(x(t), u(t)) \geq 0$ only at the grid points t_i, but we point out that also a finer grid could be chosen without any problem. Thus, we obtain the following finite dimensional nonlinear programming problem (NLP):

$$\underset{q}{\text{minimize}} \quad \int_0^T L(x(t;q), u(t;q))\, dt \; + \; E\,(x(T;q)) \tag{9}$$

subject to

$$\begin{aligned} h(x(t_i;q), u(t_i;q)) &\geq 0, \qquad i = 0, \ldots, N, \text{ (discretized path constraints)} \\ r\,(x(T;q)) &= 0. \qquad\qquad\qquad \text{(terminal constraints)} \end{aligned}$$

This problem is solved by a finite dimensional optimization solver, e.g. Sequential Quadratic Programming (SQP), as described above.

The behaviour of single shooting (with full step SQP and Gauss-Newton Hessian) applied to the tutorial example is illustrated in Fig. 5. The initialization – at the zero control trajectory, $u(t) = 0$ – and the first iteration are shown. Note that the state path and terminal constraints are not yet satisfied in the first iteration, due to their strong nonlinearity. The solution (up to an accuracy of 10^{-5}) is obtained after seven iterations. The strong points of single shooting are (i) that it can use fully adaptive, error controlled state-of-the-art ODE or DAE solvers, (ii) that it has only few optimization degrees of freedom even for large ODE or DAE systems, and (iii) that only initial guesses for the control degrees of freedom are needed. The weak points are (i) that we cannot use knowledge of the state trajectory x in the initialization (e.g. in tracking problems), (ii) that the ODE solution $x(t;q)$ can depend very nonlinearly on q, as in the example, and (iii) that unstable systems are difficult to treat.

However, due to its simplicity, the single shooting approach is very often used in engineering applications e.g. in the commercial package gOPT [41].

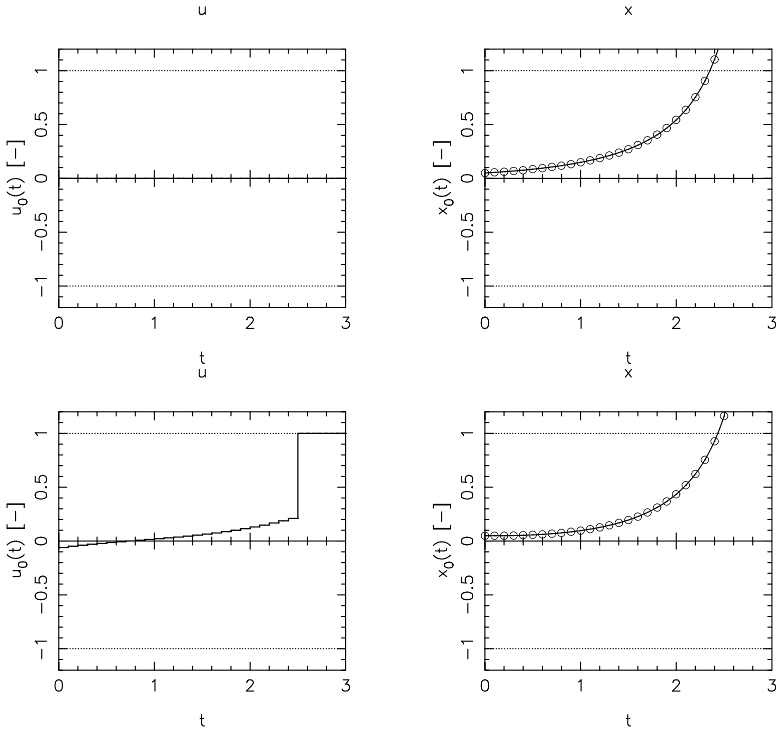


Fig. 5. Single shooting applied to the tutorial example: Initialization and first iteration

2.3 Collocation

We only very briefly sketch here the idea of the second direct approach, collocation. We start by discretizing both, the controls *and* the states on a fine grid. Typically, the controls are chosen to be piecewise constant, with values q_i on each interval $[t_i, t_{i+1}]$. The value of the states at the grid points will be denoted by $s_i \approx x(t_i)$. In order to avoid notational overhead, we will in the remainder of this section assume that the length of the time horizon, T, is constant, but point out that the generalization to variable horizon problems by the above mentioned time transformation is straightforward. In collocation, the infinite ODE

$$\dot{x}(t) - f(x(t), u(t)) = 0, \quad t \in [0, T]$$

is replaced by finitely many equality constraints

$$c_i(q_i, s_i, s_i', s_{i+1}) = 0, \quad i = 0, \ldots, N-1 \ ,$$

where the additional variables s_i' might represent the state trajectory on intermediate "collocation points" within the interval $[t_i, t_{i+1}]$. By a suitable choice of the location of these points a high approximation order can be achieved, and typically they are chosen to be the zeros of orthogonal polynomials. But we sketch here only a simplified tutorial case, where no intermediate variables s'_i are present, to give a flavour of the idea of collocation. Here, the additional equalities are given by

$$c_i(q_i, s_i, s_{i+1}) := \frac{s_{i+1} - s_i}{t_{i+1} - t_i} - f\left(\frac{s_i + s_{i+1}}{2}, q_i\right).$$

Then, we will also approximate the integrals on the collocation intervals, e.g. by

$$l_i(q_i, s_i, s_{i+1}) := L\left(\frac{s_i + s_{i+1}}{2}, q_i\right)(t_{i+1} - t_i) \approx \int\limits_{t_i}^{t_{i+1}} L(x(t), u(t))dt$$

After discretization we obtain a large scale, but sparse NLP:

$$\begin{array}{ll} \underset{s,\,q}{\text{minimize}} & \displaystyle\sum_{i=0}^{N-1} l_i(q_i, s_i, s_{i+1}) \;+\; E\left(s_N\right) \\ \text{subject to} & \end{array}$$

$$\begin{array}{rll} s_0 - x_0 = 0, & & \text{(fixed initial value)} \\ c_i(q_i, s_i, s_{i+1}) = 0, & i = 0, \ldots, N-1, & \text{(discretized ODE model)} \\ h(s_i, q_i) \geq 0, & i = 0, \ldots, N, & \text{(discretized path constraints)} \\ r\left(s_N\right) = 0. & & \text{(terminal constraints)} \end{array}$$

This problem is then solved e.g. by a reduced SQP method for sparse problems [8, 48], or by an interior-point method [7]. Efficient NLP methods typically do not keep the iterates feasible, so the discretized ODE model equations are only satisfied at the NLP solution, i.e., simulation and optimization proceed simultaneously. The advantages of collocation methods are (i) that a very sparse NLP is obtained (ii) that we can use knowledge of the state trajectory x in the initialization (iii) that it shows fast local convergence (iv) that it can treat unstable systems well, and (v) that it can easily cope with state and terminal constraints. Its major disadvantage is that adaptive discretization error control needs regridding and thus changes the NLP dimensions. Therefore, applications of collocation do often not address the question of proper discretization error control. Nevertheless, it is successfully used for many practical optimal control problems [3, 50, 47, 14, 54].

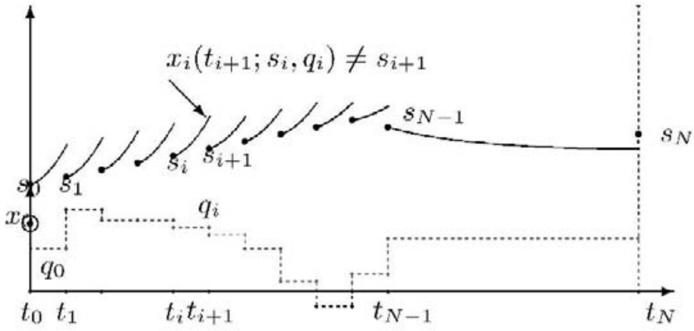


Fig. 6. Illustration of multiple shooting

2.4 Direct Multiple Shooting

The direct multiple shooting method (that is due to Bock and Plitt [12]) tries to combine the advantages of a simultaneous method like collocation with the major advantage of single shooting, namely the possibility to use adaptive, error controlled ODE solvers. In direct multiple shooting, we proceed as follows. First, we again discretize the controls piecewise on a coarse grid

$$u(t) = q_i \quad \text{for} \quad t \in [t_i, t_{i+1}] \,,$$

where the intervals can be as large as in single shooting. But second, we solve the ODE on each interval $[t_i, t_{i+1}]$ independently, starting with an artificial initial value s_i:

$$\begin{aligned} \dot{x}_i(t) &= f(x_i(t), q_i), \quad t \in [t_i, t_{i+1}] \,, \\ x_i(t_i) &= s_i. \end{aligned}$$

By numerical solution of these initial value problems, we obtain trajectory pieces $x_i(t; s_i, q_i)$, where the extra arguments after the semicolon are introduced to denote the dependence on the interval's initial values and controls. Simultaneously with the decoupled ODE solution, we also numerically compute the integrals

$$l_i(s_i, q_i) := \int_{t_i}^{t_{i+1}} L(x_i(t_i; s_i, q_i), q_i) dt \,.$$

In order to constrain the artificial degrees of freedom s_i to physically meaningful values, we impose continuity conditions $s_{i+1} = x_i(t_{i+1}; s_i, q_i)$. Thus, we arrive at the following NLP formulation that is completely equivalent to the single shooting NLP, but contains the extra variables s_i, and has a block sparse structure.

$$\underset{s,\,q}{\text{minimize}} \quad \sum_{i=0}^{N-1} l_i(s_i, q_i) \;+\; E\,(s_N) \tag{10}$$

subject to

$$\begin{aligned}
s_0 - x_0 &= 0, & & \text{(initial value)}\\
s_{i+1} - x_i(t_{i+1}; s_i, q_i) &= 0, \quad & i = 0, \ldots, N-1, \; & \text{(continuity)}\\
h(s_i, q_i) &\geq 0, & i = 0, \ldots, N, \; & \text{(discretized path constraints)}\\
r\,(s_N) &= 0. & & \text{(terminal constraints)}
\end{aligned}$$

If we summarize all variables as $w := (s_0, q_0, s_1, q_1, \ldots, s_N)$ we obtain an NLP in the form (2). The block sparse Jacobian $\nabla b(w^k)^T$ contains the linearized dynamic model equations, and the Hessian $\nabla_w^2 \mathcal{L}(w_k, \lambda_k, \mu_k)$ is block diagonal, which can both be exploited in the tailored SQP solution procedure [12]. Because direct multiple shooting only delivers a valid (numerical) ODE solution when also the optimization iterations terminate, it is usually considered a simultaneous method, as collocation. But sometimes it is also called a *hybrid method*, as it combines features from both, a pure sequential, and a pure simultaneous method. Its advantages are mostly the same as for collocation, namely that knowledge of the state trajectory can be used in the initialization, and that it robustly handles unstable systems and path state and terminal constraints.

The performance of direct multiple shooting – and of any other simultaneous method – is for the tutorial example illustrated in Fig. 7. The figure shows first the initialization by a forward simulation, using zero controls. This is one particularly intuitive, but by far not the best possibility for initialization of a simultaneous method: it is important to note that the state trajectory is by no means constrained to match the controls, but can be chosen point for point if desired. In this example, the forward simulation is at least reset to the nearest bound whenever the state bounds are violated at the end of an interval, in order to avoid simulating the system in areas where we know it will never be at the optimal solution. This leads to the discontinuous state trajectory shown in the top row of Fig. 7. The result of the first iteration is shown in the bottom row, and it can be seen that it is already much closer to the solution than single shooting, cf. Fig. 5. The solution, cf. Fig. 3, is obtained after two more iterations. It is interesting to note that the terminal constraint is already satisfied in the first iteration, due to its linearity. The nonlinear effects of the continuity conditions are distributed over the whole horizon, which is seen in the discontinuities. This is in contrast to single shooting, where the nonlinearity of the system is accumulated until the end of the horizon, and the terminal constraint becomes much more nonlinear than necessary. Any simultaneous method, e.g. collocation, would show the same favourable performance as direct multiple shooting here.

As said above, in contrast to collocation, direct multiple shooting can combine adaptivity with fixed NLP dimensions, by the use of adaptive ODE/DAE

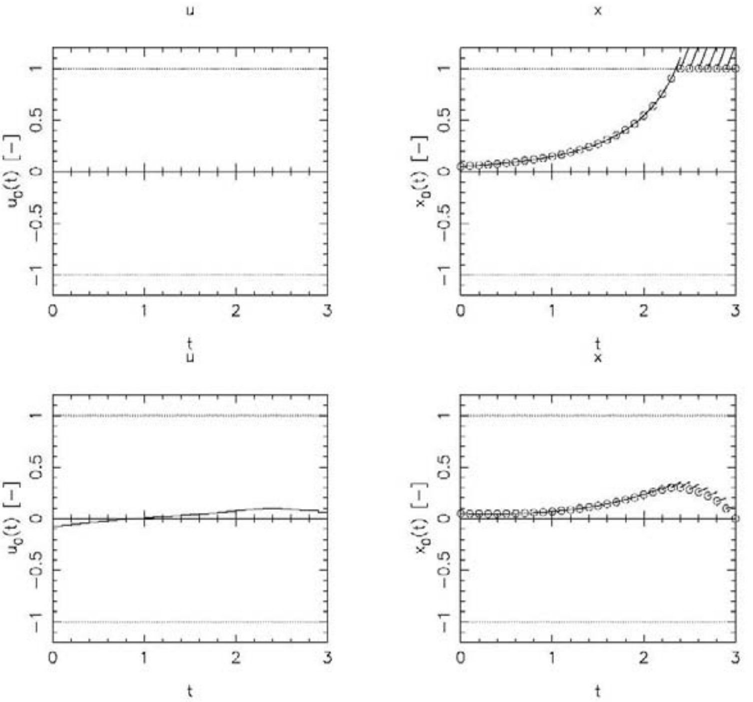


Fig. 7. Multiple shooting applied to the tutorial example: Initialization and first iteration

solvers. Within each SQP iteration, the ODE solution is often the most costly part, that is easy to parallelize. Compared to collocation the NLP is of smaller dimension but less sparse. This loss of sparsity, together with the cost of the underlying ODE solution leads to theoretically higher costs per SQP iteration than in collocation. On the other hand, the possibility to use efficient state-of-the-art ODE/DAE solvers and their inbuilt adaptivity makes direct multiple shooting a strong competitor to direct collocation in terms of CPU time per iteration. From a practical point of view it offers the advantage that the user does not have to decide on the grid of the ODE discretization, but only on the control grid. Direct multiple shooting was used to solve practical offline optimal control problems e.g. in [24, 33], and it is also used for the calculations in this paper. It is also widely used in online optimization and NMPC applications e.g. in [44, 43, 52, 53, 18, 25].

3 Time Optimal Control of a Five Link Robot Arm

We consider the time optimal point to point motion of a robot arm with five degrees of freedom. Figure 8 shows the robot and its possible movements. To

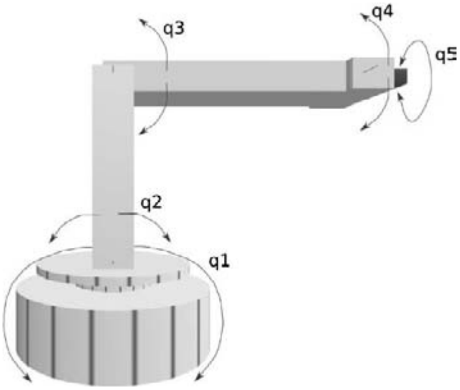


Fig. 8. Robot appearance with simplified last link and manipulator

provide a better visualization the last link and the manipulator in the images are shorter and simplified compared to the assumed model parameters.

The robot is modelled as a kinematic chain of rigid bodies, i.e., the robot is assumed to just consist of joints and links between them. The robot has a rotational base joint with two degrees of freedom, followed by two links with rotary joints, and finally one rotational joint at the "hand" of the arm. Each of the five joints contains a motor to apply a torque $u_i(t)$. The geometric description of the robot uses the notation of Denavit and Hartenberg [16]. To provide the data for the dynamic calculation each link is associated with an inertia tensor, the mass and the position of the center of mass. This approach leads to a set of five generalized coordinates $(q_1(t), \dots, q_5(t))$ each representing a rotation in the corresponding joint. We have chosen parameters that correspond to a small toy robot arm, and which are listed in Table 1 using the conventional Denavit-Hartenberg notation. The corresponding equations of motion can then be generated automatically by a script from the HuMAnS Toolbox [29].

3.1 Fast Computations of the Dynamics of Robots

The dynamics of a robot is most usually presented in its Lagrangian form

Table 1. Dynamic data of the example robot, and Denavit-Hartenberg parameters

Joint i	Mass m_i	c.o.m. r_i	Inertia tensor I_i	α_i	a_i	θ_i	d_i
1	0.1	$(0, 0, 0)^\mathrm{T}$	$\mathrm{diag}(23, 23, 20) \cdot 10^{-6}$	0	0	$q_1(t)$	0
2	0.02	$(0.06, 0, 0)^\mathrm{T}$	$\mathrm{diag}(7, 118, 113) \cdot 10^{-6}$	$-\frac{\pi}{2}$	0	$-\frac{\pi}{2}+q_2(t)$	0
3	0.1	$(0.06, 0, 0)^\mathrm{T}$	$\mathrm{diag}(20, 616, 602) \cdot 10^{-6}$	0	0.12	$\frac{\pi}{2}+q_3(t)$	0
4	0.03	$(0, -0.04, 0)^\mathrm{T}$	$\mathrm{diag}(-51, -7, -46) \cdot 10^{-6}$	0	0.12	$\frac{\pi}{2}+q_4(t)$	0
5	0.06	$(0, 0, 0.1)^\mathrm{T}$	$\mathrm{diag}(650, 640, 26) \cdot 10^{-6}$	$\frac{\pi}{2}$	0	$q_5(t)$	0

$$M(q(t))\,\ddot{q}(t) + N(q(t), \dot{q}(t)) = u(t)\ ,$$

which gives a compact description of all the nonlinear phenomena and can be manipulated easily in various ways. Since the mass matrix $M(q(t))$ is Symmetric Definite Positive, it is invertible and the acceleration of the system can be related with the controls $u(t)$ either in the way

$$u(t) = M(q(t))\,\ddot{q}(t) + N(q(t), \dot{q}(t)) \tag{11}$$

or in the way

$$\ddot{q}(t) = M(q(t))^{-1}\big(u(t) - N(q(t), \dot{q}(t))\big)\ , \tag{12}$$

corresponding respectively to the inverse and direct dynamics of the system. Very helpful from the point of view of analytical manipulations [56], this way of describing the dynamics of a robot is far from being efficient from the point of view of numerical computations, neither in the form (11) nor (12). Especially the presence of a matrix-vector multiplication of $O(N^2)$ complexity in both (11) and(12), and of a matrix inversion of $O(N^3)$ complexity in (12) can be avoided: recursive algorithms for computing both (11) and (12) with only an $O(N)$ complexity are well known today.

The first algorithm that has been investigated historically for the fast computation of the dynamics of robots is the Recursive Newton-Euler Algorithm that allows computing directly the controls related to given accelerations exactly as in (11). Extensions have been devised also for cases when not all of the acceleration vector $\ddot{q}$ is known, in the case of underactuated systems such as robots executing aerial maneuvers [49]. This recursive algorithm is the fastest way to compute the complete dynamics of a robotic system and should be preferred therefore as long as one is not strictly bound to using the direct dynamics (12). This is the case for collocation methods but unfortunately not for shooting methods.

The Recursive Newton-Euler Algorithm has been adapted then in the form of the Composite Rigid Body Algorithm in order to compute quickly the mass matrix that needs to be inverted in the direct dynamics (12), but we still have to face then a matrix inversion which can be highly inefficient for "large" systems. The computation of this mass matrix and its inversion can be necessary though for systems with unilateral contacts, when some internal forces are defined through implicit laws [55].

The Articulated Body Algorithm has been designed then to propose a recursive method of $O(N)$ complexity for computing directly the accelerations related to given torques as in (12) but without resorting to a matrix inversion. Even though generating a slightly higher overhead, this algorithm has been proved to be more efficient than the Composite Rigid Body Algorithm for robots with as few as 6 degrees of freedom [26]. Moreover, avoiding the matrix inversion allows producing a less noisy numerical result, what can greatly enhance the efficiency of any adaptive ODE solver to which it is connected [2]. For these reasons, this recursive algorithm should be preferred as soon as one needs to compute the direct dynamics (12), what is the case for shooting methods.

Now, one important detail when designing fast methods to compute numerically the dynamics of a robot is to generate offline the computer code corresponding to the previous algorithms. Doing so, not only is it possible to get rid of constants such as 0 and 1 with all their consequences on subsequent computations, but it is also possible to get rid of whole series of computations which may appear to be completely unnecessary, depending on the specific structure of the robot. Such an offline optimization leads to computations which can be as much as twice faster than the strict execution of the same original algorithms.

The HuMAnS toolbox [29], used to compute the dynamics of the robot for the numerical experiment in the next section, proposes only the Composite Rigid Body Algorithm, so far, so even faster computations should be expected when using an Articulated Body Algorithm. Still, this toolbox produces faster computations than other generally available robotics toolboxes thanks to its offline optimization of the generated computer code (a feature also present in the SYMORO software [30]).

3.2 Optimization Problem Formulation

In order to solve the problem to minimize a point to point motion of the robot arm, we consider the following example maneuver: the robot shall pick up an object at the ground and put it as fast as possible into a shelf, requiring a base rotation of ninety degrees. We formulate an optimal control problem of the form (1), with the following definitions:

$$\begin{aligned}
x(t) &= (q_1(t), \dots, q_5(t), \dot{q_1}(t), \dots, \dot{q_5}(t))^{\mathrm{T}} \\
u(t) &= (u_1(t), \dots, u_5(t))^{\mathrm{T}} \\
L(x(t), u(t)) &= 1 \\
E(x(T)) &= 0 \\
f(x(t), u(t)) &= \begin{pmatrix} (\dot{q_1}(t), \dots, \dot{q_5}(t))^{\mathrm{T}} \\ M(x(t))^{-1} \cdot (u(t) - N(x(t))) \end{pmatrix} \\
x_0 &= (-0.78, 0.78, 0, 0.78, 0, 0, 0, 0, 0, 0)^{\mathrm{T}} \\
r(x(T)) &= x(T) - (0.78, 0, -0.78, 0.78, 0, 0, 0, 0, 0, 0)^{\mathrm{T}} \\
h(x(t), u(t)) &= \begin{pmatrix} x_{\max} - x \\ x - x_{\min} \\ u_{\max} - u \\ u - u_{\min} \\ (1, 0, 0, 1) \cdot T_5^0(x(t)) \cdot (0, 0, l, 1)^{\mathrm{T}} - 0.05 \\ (0, 0, 1, 1) \cdot T_5^0(x(t)) \cdot (0, 0, l, 1)^{\mathrm{T}} + 0.15 \end{pmatrix}
\end{aligned}$$

The controls $u(t)$ are the torques acting in the joints. The cost functional $\int_0^T L(x,u)dt + E(x(T))$ is the overall maneuver time, T. Within the dynamic model $\dot{x} = f(x,u)$, the matrix $M(x(t))$ is the mass matrix which is calculated in each evaluation of $f(x(t),u(t))$ and inverted using a cholesky algorithm. The vector $N(x(t))$ describes the combined centrifugal, Coriolis and gravitational force. The initial and terminal constraints $x(0) = x_0$ and $r(x(T)) = 0$ describe the desired point to point maneuver. As the states and controls have lower and upper bounds, and as the robot hand shall avoid hitting the the ground as well as its own base, we add the path constraints $h(x,u) \geq 0$. Here, the matrix $T_5^0(x(t)$ describes the transformation that leads from the local end effector position $(0,0,l,1)^{\mathrm{T}}$ in the last frame to the absolute coordinates in the base frame.

3.3 Numerical Solution by Direct Multiple Shooting

We have coupled the automatic robot model generator HuMAnS [29] with an efficient implementation of the direct multiple shooting method, the optimal control package MUSCOD-II [32, 33]. This coupling allows us to use the highly optimized C-code delivered by HuMAnS within the model equations $\dot{x} = f(x,u)$ required by MUSCOD-II in an automated fashion. In the following computations, we choose an error controlled Runge-Kutta-Fehlberg integrator of order four/five. We use 30 multiple shooting nodes with piecewise constant controls. Within the SQP method, a BFGS Hessian update and watchdog line search globalisation is used.

For initialization, the differential states on the multiple shooting nodes are interpolated linearly between desired initial and terminal state, as shown in Fig. 9. The maneuver time for initialization was set to 0.3 seconds, and the controls to zero. Starting with this infeasible initialization, the overall optimization with MUSCOD-II took about 130 SQP iterations, altogether requiring about 20 CPU seconds on a standard LINUX machine with a 3 GHz Pentium IV processor. The solution is shown in Fig. 10. The calculated time optimal robot movement of 0.15 seconds duration is illustrated in Fig. 11 with screenshots from an OpenGL visualization.

4 Nonlinear Model Predictive Control

As mentioned in the introduction, Nonlinear Model Predictive Control (NMPC) is a feedback control technique based on the online solution of open-loop optimal control problems of the form (1). The optimization is repeated again and again, at intervals of length δ, each sampling time $t_k = k\delta$ for the most currently observed system state $\bar{x}(t_k)$, which serves as initial value $x_0 := \bar{x}(t_k)$ in (1). We have introduced the bar to distinguish the observed system states $\bar{x}(t)$ from the predicted states $x(t)$ within the optimal control problem. Note that the time t_k from now on is the physical time, and no

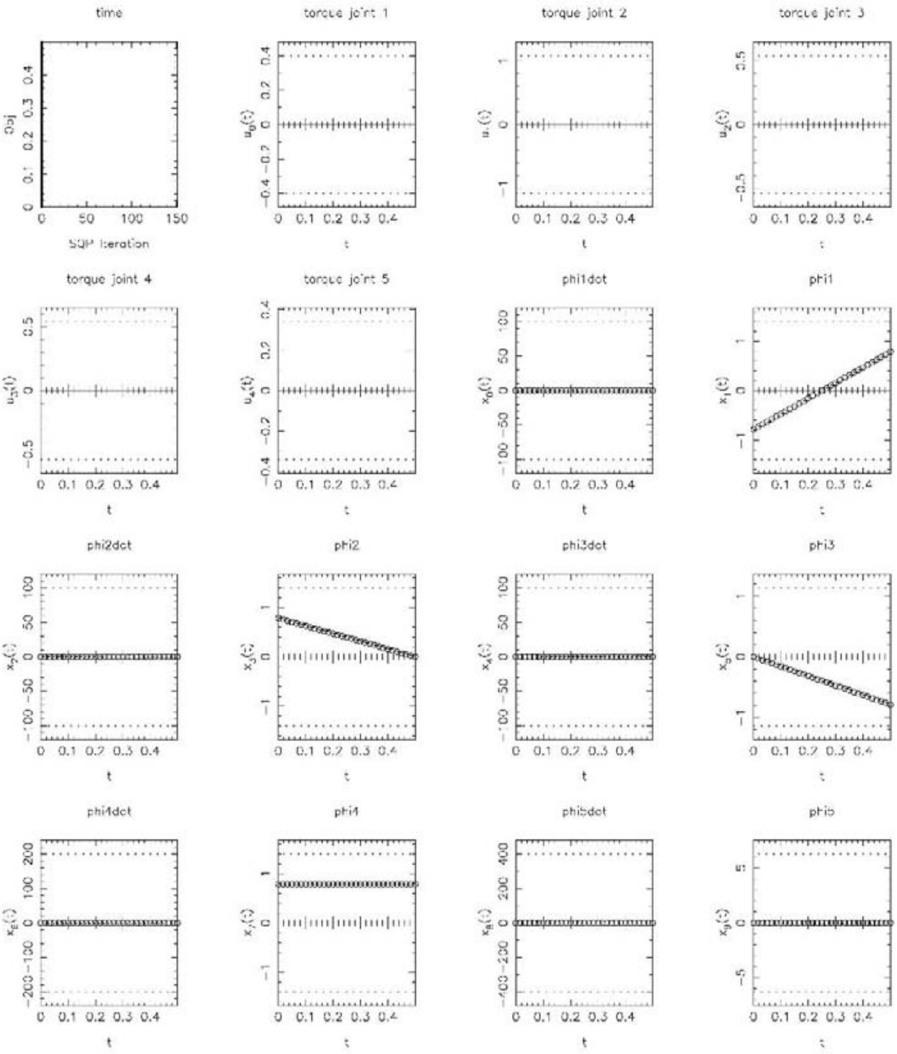


Fig. 9. Initalization of the optimization problem by linear interpolation

longer the time at a discretization point within the optimal control problem, as in Sect. 2. We stress that for autonomous systems, as treated in this paper, the NMPC optimization problems differ by the varying initial values only, and that the time coordinate used within the optimal control problem (1) can be assumed to always start with $t = 0$ even though this does not reflect the physical time. From now on, we will denote the time coordinate within the optimal control problem with τ in this section to avoid confusion.

To be specific, we denote the optimal solution of the optimal control problem (1) by $u^*(\tau; \bar{x}(t_k))$, $\tau \in [0, T^*(\bar{x}(t_k))]$, to express its parametric depen-

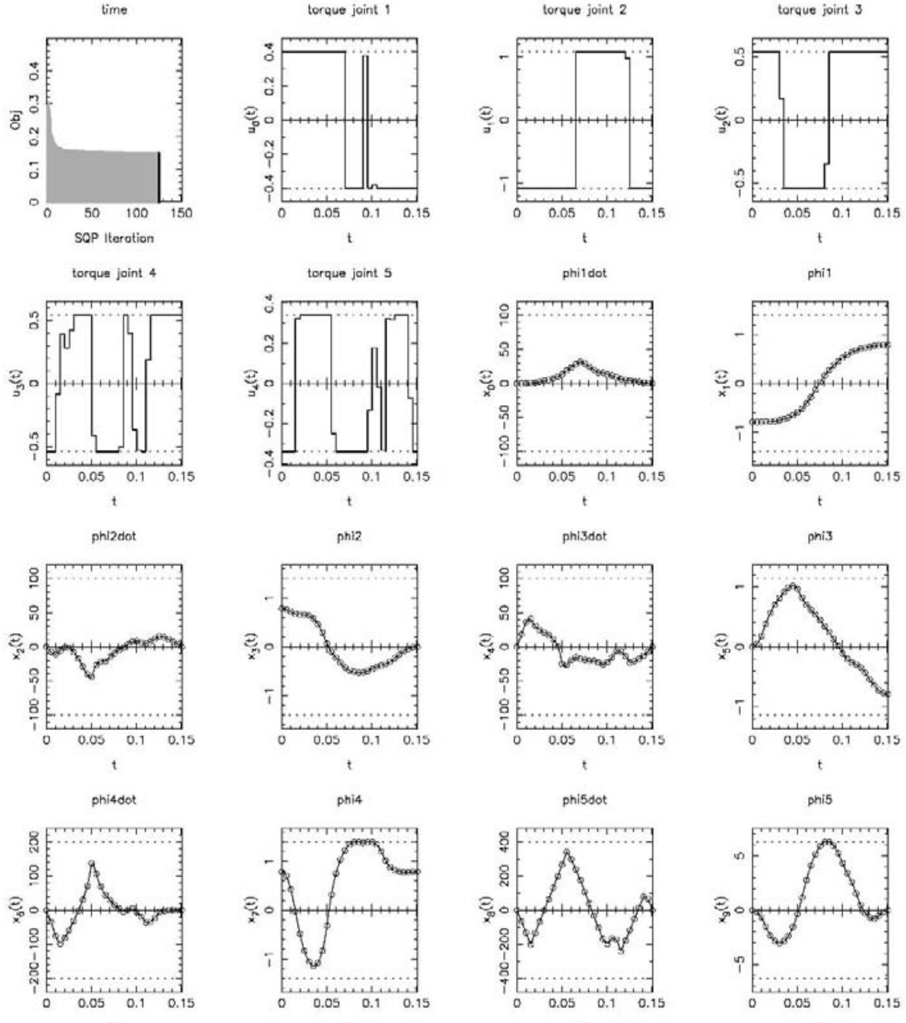


Fig. 10. Solution of the optimization problem, obtained after 130 SQP iterations and 20 CPU seconds

dence on the initial value $\bar{x}(t_k)$. The feedback control implemented during the following sampling interval, i.e. for $t \in [t_k, t_{k+1}]$, is simply given by $u_0^*(\bar{x}(t_k)) := u^*(0; \bar{x}(t_k))$.[1] Thus, NMPC is a sampled data feedback con-

[1] Sometimes, instead of the optimal initial control value $u^*(0; \bar{x}(t_k))$, the whole first control interval of length δ, i.e., $u^*(\tau; \bar{x}(t_k))$, $\quad \tau \in [0, \delta]$, is applied to the real process. This is more appealing in theory, and stability proofs are based on such an NMPC formulation. When a control discretization with interval lengths not smaller than the sampling time is used, however, both formulations coincide.

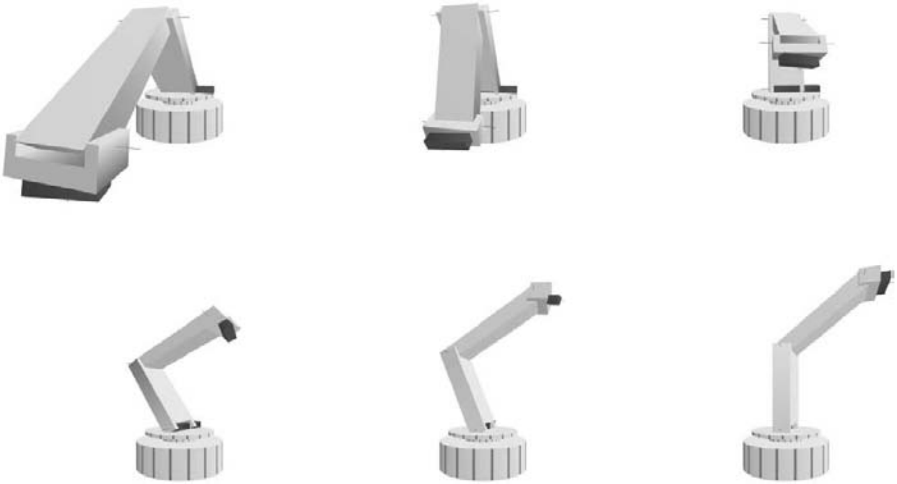


Fig. 11. Visualization of the time optimal point to point motion from Fig. 10

trol technique. It is closely related to *optimal feedback control* which would apply the continuous, non-sampled feedback law $u_0^*(\bar{x}(t))$ for all t, which can be called the limit of NMPC for infinitely small sampling times δ. Note that the optimal predicted maneuver time $T^*(x_k)$ would typically be shrinking for an optimal point to motion. In this case we speak of *shrinking horizon NMPC* [10]. If a large disturbance occurs, the horizon might also be enlarged as the future plan is changed. In the other case, when the horizon length is fixed to $T = T_p$, where the constant T_p is the *prediction horizon* length, we speak of *moving*, or *receding horizon control (RHC)* [37]. The moving horizon approach is applicable to continuous processes and so widely employed that the term NMPC is often used as synonymous to RHC. When a given trajectory shall be tracked, this is often expressed by the choice of the cost function in form of an integrated least squares deviation on a fixed prediction horizon. In fast robot motions, however, we believe that a variable time horizon for point to point maneuvers will be a crucial ingredient to successful NMPC implementations. A shrinking horizon NMPC approach for robot point to point motions that avoids that $T^*(x)$ shrinks below a certain positive threshold was presented by Zhao et al. [57]. For setpoint tracking problems, extensive literature exists on the stability of the closed loop system. Given suitable choices of the objective functional defined via L and E and a terminal constraint of the form $r(x(T)) = 0$ or $r(x(T)) \geq 0$, stability of the nominal NMPC dynamics can be proven even for strongly nonlinear systems [37, 15, 38, 35].

One important precondition for successful NMPC applications, however, is the availability of reliable and efficient numerical optimal control algorithms. Given an efficient offline optimization algorithm – e.g. one of the three SQP based direct methods described in Sect. 2 – we might be tempted to restart it again and again for each new problem and to solve each problem until a prespecified convergence criterion is satisfied. If we are lucky, the computation time is negligible; if we are not, we have to enter the field of real-time optimization.

The Online Dilemma

Assuming that the computational time for one SQP iteration is more or less constant, we have to address the following dilemma: If we want to obtain a sufficiently exact solution for a given initial value $\bar{x}(t_k)$, we have to perform several SQP iterations until a prespecified convergence criterion is satisfied. We can suppose that for achieving this we have to perform n iterations, and that each iteration takes a time ϵ. This means that we obtain the optimal feedback control $u_0^*(\bar{x}(t_k))$ only at a time $t_k+n\epsilon$, i.e., with a considerable delay. However at time $t_k + n\epsilon$ the system state has already moved to some system state $\bar{x}(t_k + n\epsilon) \neq \bar{x}(t_k)$, and $u_0^*(x(t_k))$ is *not* the exact NMPC feedback, $u_0^*(\bar{x}(t_k + n\delta))$. In the best case the system state has not changed much in the meantime and it is a good approximation of the exact NMPC feedback. Also, one might think of predicting the most probable system state $\bar{x}(t_k + n\epsilon)$ and starting to work on this problem already at time t_k. The question of which controls have to be applied in the meantime is still unsolved: a possible choice would be to use previously optimized controls in an open-loop manner. Note that with this approach we can realize an NMPC recalculation rate with intervals of length $\delta = n\epsilon$, under the assumption that *each* problem needs at most n iterations and that each SQP iteration requires at most a CPU time of ϵ. Note also that feedback to a disturbance comes with a delay δ_d of one full sampling time. Summarizing, we would have $\delta_d = \delta = n\epsilon$.

4.1 Real-Time Iteration Scheme

We will now present a specific answer to the online dilemma, the real-time iteration scheme [17, 20]. The approach is based on two observations.

- Due to the online dilemma, we will never be able to compute the *exact* NMPC feedback control $u_0^*(\bar{x}(t_k))$ without delay. Therefore, it might be better to compute only an *approximation* $\tilde{u}_0(\bar{x}(t_k))$ of $u_0^*(\bar{x}(t_k))$, if this approximation can be computed much faster.
- Second, we can divide the computation time of each cycle into a a short feedback phase (FP) and a possibly much longer preparation phase (PP). While the feedback phase is only used to evaluate the approximation $\tilde{u}_0(\bar{x}(t_k))$, the following preparation phase is used to *prepare* the next feedback, i.e., to compute $\tilde{u}_0(\bar{x}(t_{k+1}))$ as much as possible without knowledge of $\bar{x}(t_{k+1})$.

This division of the computation time within each sampling interval allows to achieve delays δ_d that are smaller than the sampling interval δ, see Fig. 12.

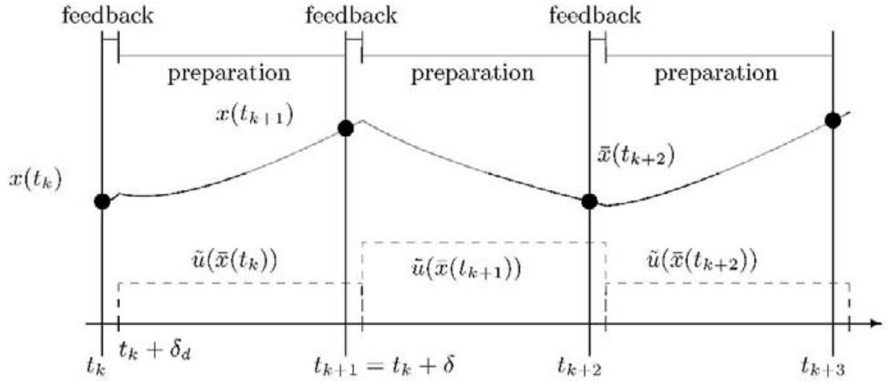


Fig. 12. Division of the computation time in the real-time iteration scheme; real system state and control trajectory, for sampling time δ and feedback delay $\delta_d \ll \delta$

The crucial question is, of course, which approximation $\tilde{u}_0(\bar{x}(t_k))$ should be used, and how it can be made similar to the exact NMPC feedback $u_0^*(\bar{x}(t_k))$.

In its current realization, the real-time iteration scheme is based on the direct multiple shooting method. The online optimization task is to solve a sequence of nonlinear programming problems of the form (10), but with varying initial value constraint $s_0 - \bar{x}(t_k) = 0$. Similar to the NLP notation (2), in the online context we have to solve, as fast as possible, an NLP

$$P(\bar{x}(t_k)) : \qquad \min_{w} a(w) \quad \text{subject to} \quad b_{\bar{x}(t_k)}(w) = 0, \quad c(w) \geq 0 \; , \tag{13}$$

where the index takes account of the fact that the first equality constraint $s_0 - \bar{x}(t_k) = 0$ from $b_{\bar{x}(t_k)}(w) = 0$ depends on the initial value $\bar{x}(t_k)$, and where $w = (s_0, q_0, s_1, q_1, \ldots, s_N)$. Ideally, we would like to have the solution $w^*(\bar{x}(t_k))$ of each problem $P(\bar{x}(t_k))$ as quick as possible, and to take the NMPC feedback law to be the first control within $w^*(\bar{x}(t_k))$, i.e., to set $u_0^*(\bar{x}(t_k)) := q_0^*(\bar{x}(t_k))$. The exact solution manifold $w^*(\,\cdot\,)$ in dependence of the initial value $\bar{x}(t_k)$ is sketched as the solid line in Fig. 13 – nondifferentiable points on this manifold are due to active set changes in the NLP. The exact solution, however, is not computable in finite time.

Initial Value Embedding

The major idea underlying the real-time iteration scheme is to initialize each new problem $P(\bar{x}(t_k))$ with the most current solution guess from the last problem, i.e. with the solution of $P(\bar{x}(t_{k-1}))$. In a simultaneous method like direct multiple shooting, it is no problem that the initial value constraint $s_0 - \bar{x}(t_k) = 0$ is violated. On the contrary, because this constraint is linear, it can be shown that the first SQP iteration after this "initial value embedding" is a first order predictor for the correct new solution, even in the presence of

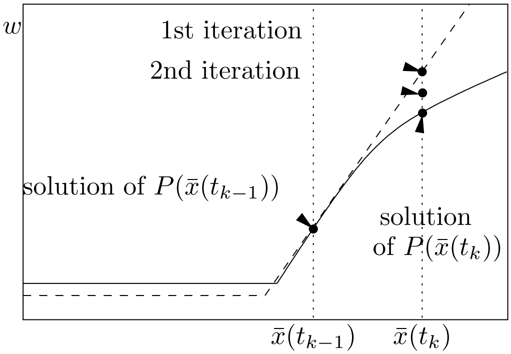

w
1st iteration
2nd iteration
solution of $P(\bar{x}(t_{k-1}))$
solution of $P(\bar{x}(t_k))$
$\bar{x}(t_{k-1})$
$\bar{x}(t_k)$

Fig. 13. Exact solution manifold (*solid line*) and tangential predictor after initial value embedding (*dashed line*), when initialized with the solution of $P(\bar{x}(t_{k-1}))$. The first iteration delivers already a good predictor for the exact solution of $P(\bar{x}(t_k))$

active set changes [17]. This observation is visualized in Fig. 13, where the predictor delivered by the first SQP iteration is depicted as dashed line.

In the real-time iteration scheme, we use the result of the first SQP iteration directly for the approximation $\tilde{u}_0(\bar{x}(t_k))$. This would already reduce the feedback delay δ_d to the time of one SQP iteration, ϵ. Afterwards, we would need to solve the old problem to convergence in order to prepare the next feedback. In Fig. 13 also the second iterate and solution for problem $P(\bar{x}(t_k))$ are sketched. But two more considerations make the algorithm even faster.

- First, the computations for the first iteration can be largely performed *before* the initial value $\bar{x}(t_k)$ is known. Therefore, we can reduce the delay time further, if we perform all these computations before time t_k, and at time t_k we can quickly compute the feedback response $\tilde{u}_0(\bar{x}(t_k))$ to the current state. Thus, the feedback delay δ_d becomes even smaller than the cost of one SQP iteration, $\delta_d \ll \epsilon$.
- Second, taking into account that we already use an approximate solution of the optimal control problem we can ask if it is really necessary to iterate the SQP until convergence requiring a time $n\epsilon$ for n SQP iterations. Instead, we will considerably reduce the preparation time by performing just one iteration per sampling interval. This allows shorter sampling intervals that only have the duration of one single SQP iteration, i.e., $\delta = \epsilon$. A positive side-effect is that this shorter recalculation time most probably leads to smaller differences in subsequent initial states $\bar{x}(t_k)$ and $\bar{x}(t_{k+1})$, so that the initial value embedding delivers better predictors.

These two ideas are the basis of the real-time iteration scheme. It allows to realize feedback delays δ_d that are much shorter than a sampling time, and sampling times δ that are only as long as a single SQP iteration, i.e. we have $\delta_d \ll \delta = \epsilon$. Compared with the conventional approach with $\delta_d = \delta = n\epsilon$, the

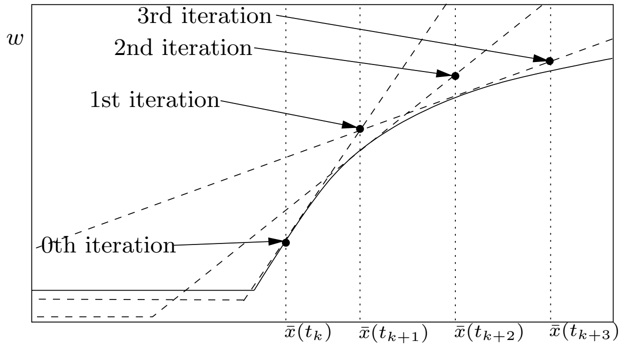


Fig. 14. Sketch of the real-time iterations that stay close to the exact solution manifold (*solid line*)

focus is now shifted from a sequence of optimization problems to a sequence of SQP iterates: we may regard the SQP procedure iterating uninterrupted, with the only particularity that the initial value $\bar{x}(t_k)$ is modified *during* the iterations. The generation of the feedback controls can then be regarded as a by-product of the SQP iterations. Due to the initial value embedding property, it can be expected that the iterates remain close to the exact solution manifold for each new problem. In Fig. 14 four consecutive real-time iterates are sketched, where the dashed lines show the respective tangential predictors.

Applications

The real-time iteration scheme has successfully been used in both simulated and experimental NMPC applications, among them the experimental NMPC of a high purity distillation column [23] described by a 200 state DAE model with sampling times δ of 20 seconds, or simulated NMPC of a combustion engine described by 5 ODE, with sampling times of 10 milliseconds [27]. Depending on the application, the feedback delay δ_d was between 0.5 and 5 percent of the sampling time. Within the studies, the approximation errors of the real-time iteration scheme compared to exact NMPC are often negligible. The scheme's theoretical contraction properties have been investigated in [21] for the variant described in this paper, and in [22, 19] for other variants. Recently, several novel variants of the real-time iteration scheme have been proposed that can either work with inexact jacobians within the SQP procedure [11], or that only evaluate the jacobians on a subspace [46]. These variants offer advantages for large scale systems where they promise to allow sampling times that are one or two orders of magnitude smaller than in the standard implementation. A numerical application of the standard real-time iteration scheme to the time optimal point to point motion of a robot arm described by 8 ODE was presented in [57], with CPU times still in the order of 100 milliseconds per sampling time. We expect that the development of real-time iteration variants that are specially tailored to robotics applications will make NMPC of fast robot motions possible within the next five years.

5 Summary and Outlook

In this tutorial paper, we have tried to give a (personal) overview over the most widely used methods for numerical optimal control, and to assess the possibility of real-time optimization of fast robot motions. We discussed in detail direct optimal control methods that are based on a problem discretization and on the subsequent use of a nonlinear programming algorithm like sequential quadratic programming (SQP). We compared three direct methods, (i) direct single shooting as a *sequential* approach, together with (ii) direct collocation and (iii) direct multiple shooting as *simultaneous* approaches. At hand of a tutorial example we have illustrated the better nonlinear convergence properties of the simultaneous over the sequential approaches that can be observed in many other applications, too. The direct multiple shooting method allows to use state-of-the-art ODE/DAE integrators with inbuilt adaptivity and error control which often shows to be an advantage in practice. At the example of the time optimal motion of a robot arm we have demonstrated the ability of direct multiple shooting to cope even with strongly nonlinear two point boundary value optimization problems. Using the coupling of an efficient tool for generation of optimized robot model equations, HuMAnS, and a state-of-the-art implementation of the direct multiple shooting method, MUSCOD-II, computation times for a five link robot are in the order of 200 milliseconds per SQP iteration. Finally, we discussed the possibility to generate optimization based feedback by the technique of nonlinear model predictive control (NMPC), and pointed out the necessity of fast online optimization. We have presented the *real-time iteration* scheme – that is based on direct multiple shooting and SQP – as a particularly promising approach to achieve this aim. The scheme uses an *initial value embedding* for the transition from one optimization problem to the next, and performs exactly one SQP-type iteration per optimization problem to allow short sampling times. Furthermore, each iteration is divided into a preparation and a much shorter feedback phase, to allow an even shorter feedback delay. Based on the ongoing development of the presented approaches, we expect NMPC – that performs an online optimization of nonlinear first principle robot models within a few milliseconds – to become a viable technique for control of fast robot motions within the following five years.

References

[1] F. Allgöwer, T.A. Badgwell, J.S. Qin, J.B. Rawlings, and S.J. Wright. Nonlinear predictive control and moving horizon estimation – An introductory overview.

In P. M. Frank, editor, *Advances in Control, Highlights of ECC'99*, pp. 391–449. Springer, 1999.

[2] U.M. Ascher, D.K. Pai, and B.P. Cloutier. Forward dynamics, elimination methods, and formulation stiffness in robot simulation. *International Journal of Robotics Research*, 16(6):749–758, 1997.

[3] V. Bär. Ein Kollokationsverfahren zur numerischen Lösung allgemeiner Mehrpunktrandwertaufgaben mit Schalt– und Sprungbedingungen mit Anwendungen in der optimalen Steuerung und der Parameteridentifizierung. Master's thesis, Universität Bonn, 1984.

[4] R.A. Bartlett, A. Wächter, and L.T. Biegler. Active set vs. interior point strategies for model predictive control. In *Proc. Amer. Contr. Conf.*, pp. 4229–4233, Chicago, Il, 2000.

[5] R.E. Bellman. *Dynamic Programming.* University Press, Princeton, 1957.

[6] D.P. Bertsekas. *Dynamic Programming and Optimal Control*, vol. 1. Athena Scientific, Belmont, MA, 1995.

[7] L.T. Biegler, A.M. Cervantes, and A. Waechter. Advances in simultaneous strategies for dynamic process optimization. *Chemical Engineering Science*, 4(57):575–593, 2002.

[8] L.T. Biegler. Solution of dynamic optimization problems by successive quadratic programming and orthogonal collocation. *Computers and Chemical Engineering*, 8:243–248, 1984.

[9] L.T. Biegler. Efficient solution of dynamic optimization and NMPC problems. In F. Allgöwer and A. Zheng, editors, *Nonlinear Predictive Control*, vol. 26 of *Progress in Systems Theory*, pp. 219–244, Basel Boston Berlin, 2000. Birkhäuser.

[10] T. Binder, L. Blank, H.G. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J.P. Schlöder, and O.v. Stryk. Introduction to model based optimization of chemical processes on moving horizons. In M. Grötschel, S.O. Krumke, and J. Rambau, editors, *Online Optimization of Large Scale Systems: State of the Art*, pp. 295–340. Springer, 2001.

[11] H.G. Bock, M. Diehl, E.A. Kostina, and J.P. Schlöder. Constrained optimal feedback control of systems governed by large differential algebraic equations. In L. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, and B. van Bloemen Waanders, editors, *Real-Time and Online PDE-Constrained Optimization.* SIAM, 2006. (in print).

[12] H.G. Bock and K.J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings 9th IFAC World Congress Budapest*, pp. 243–247. Pergamon Press, 1984.

[13] A.E. Bryson and Y.-C. Ho. *Applied Optimal Control.* Wiley, New York, 1975.

[14] A. Cervantes and L.T. Biegler. Large-scale DAE optimization using a simultaneous NLP formulation. *AIChE Journal*, 44(5):1038–1050, 1998.

[15] H. Chen and F. Allgöwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205–1218, 1998.

[16] J. Denavit and R.S. Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. *ASME Journal of Applied Mechanics*, 22:215–221, 1955.

[17] M. Diehl. *Real-Time Optimization for Large Scale Nonlinear Processes.* PhD thesis, Universität Heidelberg, 2001. http://www.ub.uni-heidelberg.de/archiv/1659/.

[18] M. Diehl. *Real-Time Optimization for Large Scale Nonlinear Processes*, vol. 920 of *Fortschr.-Ber. VDI Reihe 8, Meß-, Steuerungs- und Regelungstechnik*. VDI Verlag, Düsseldorf, 2002. Download also at: http://www.ub.uni-heidelberg.de/archiv/1659/.

[19] M. Diehl, H.G. Bock, and J.P. Schlöder. A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on Control and Optimization*, 43(5):1714–1736, 2005.

[20] M. Diehl, H.G. Bock, J.P. Schlöder, R. Findeisen, Z. Nagy, and F. Allgöwer. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *J. Proc. Contr.*, 12(4):577–585, 2002.

[21] M. Diehl, R. Findeisen, and F. Allgöwer. A stabilizing real-time implementation of nonlinear model predictive control. In L. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, and B. van Bloemen Waanders, editors, *Real-Time and Online PDE-Constrained Optimization*. SIAM, 2004. (submitted).

[22] M. Diehl, R. Findeisen, F. Allgöwer, H.G. Bock, and J.P. Schlöder. Nominal stability of the real-time iteration scheme for nonlinear model predictive control. *IEE Proc.-Control Theory Appl.*, 152(3):296–308, 2005.

[23] M. Diehl, R. Findeisen, S. Schwarzkopf, I. Uslu, F. Allgöwer, H.G. Bock, E.D. Gilles, and J.P. Schlöder. An efficient algorithm for nonlinear model predictive control of large-scale systems. Part II: Application to a distillation column. *Automatisierungstechnik*, 51(1):22–29, 2003.

[24] M. Diehl, D.B. Leineweber, A.A.S. Schäfer, H.G. Bock, and J.P. Schlöder. Optimization of multiple-fraction batch distillation with recycled waste cuts. *AIChE Journal*, 48(12):2869–2874, 2002.

[25] M. Diehl, L. Magni, and G. De Nicolao. Online NMPC of unstable periodic systems using approximate infinite horizon closed loop costing. *Annual Reviews in Control*, 28:37–45, 2004.

[26] R. Featherstone and D. Orin. Robot dynamics: Equations and algorithms. In *Proceedings of the IEEE International Conference on Robotics & Automation*, 2000.

[27] H.J. Ferreau, G. Lorini, and M. Diehl. Fast nonlinear model predictive control of gasoline engines. In *CCA Conference Munich*, 2006. (submitted).

[28] R. Findeisen, M. Diehl, I. Uslu, S. Schwarzkopf, F. Allgöwer, H.G. Bock, J.P. Schlöder, and E.D. Gilles. Computation and performance assesment of nonlinear model predictive control. In *Proc. 42th IEEE Conf. Decision Contr.*, pp. 4613–4618, Las Vegas, USA, 2002.

[29] INRIA. Humans toolbox. http://www.inrialpes.fr/bipop/software/humans/, Feb 2005.

[30] W. Khalil and D. Creusot. Symoro+: A system for the symbolic modelling of robots. *Robotica*, 15:153–161, 1997.

[31] D. Kraft. On converting optimal control problems into nonlinear programming problems. In K. Schittkowski, editor, *Computational Mathematical Programming*, vol. F15 of *NATO ASI*, pp. 261–280. Springer, 1985.

[32] D.B. Leineweber. *Efficient reduced SQP methods for the optimization of chemical processes described by large sparse DAE models*, vol. 613 of *Fortschritt-Berichte VDI Reihe 3, Verfahrenstechnik*. VDI Verlag, Düsseldorf, 1999.

[33] D.B. Leineweber, A.A.S. Schäfer, H.G. Bock, and J.P. Schlöder. An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process

optimization. Part II: Software aspects and applications. *Computers and Chemical Engineering*, 27:167–174, 2003.

[34] P.L. Lions. *Generalized Solutions of Hamilton-Jacobi Equations.* Pittman, 1982.

[35] L. Magni, G. De Nicolao, L. Magnani, and R. Scattolini. A stabilizing model-based predictive control for nonlinear systems. *Automatica*, 37(9):1351–1362, 2001.

[36] D.Q. Mayne. Nonlinear model predictive control: Challenges and opportunities. In F. Allgöwer and A. Zheng, editors, *Nonlinear Predictive Control*, vol. 26 of *Progress in Systems Theory*, pp. 23–44, Basel Boston Berlin, 2000. Birkhäuser.

[37] D.Q. Mayne and H. Michalska. Receding horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 35(7):814–824, 1990.

[38] G. de Nicolao, L. Magni, and R. Scattolini. Stability and robustness of nonlinear receding horizon control. In F. Allgöwer and A. Zheng, editors, *Nonlinear Predictive Control*, vol. 26 of *Progress in Systems Theory*, pp. 3–23, Basel Boston Berlin, 2000. Birkhäuser.

[39] J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer, Heidelberg, 1999.

[40] L.S. Pontryagin, V.G. Boltyanski, R.V. Gamkrelidze, and E.F. Miscenko. *The Mathematical Theory of Optimal Processes.* Wiley, Chichester, 1962.

[41] PSE. *gPROMS Advanced User's Manual.* London, 2000.

[42] S.J. Qin and T.A. Badgwell. Review of nonlinear model predictive control applications. In B. Kouvaritakis and M. Cannon, editors, *Nonlinear model predictive control: theory and application*, pp. 3–32, London, 2001. The Institute of Electrical Engineers.

[43] L.O. Santos. *Multivariable Predictive Control of Nonlinear Chemical Processes.* PhD thesis, Universidade de Coimbra, 2000.

[44] L.O. Santos, P. Afonso, J. Castro, N.M.C. de Oliveira, and L.T. Biegler. On-line implementation of nonlinear MPC: An experimental case study. In *ADCHEM2000 – International Symposium on Advanced Control of Chemical Processes*, vol. 2, pp. 731–736, Pisa, 2000.

[45] R.W.H. Sargent and G.R. Sullivan. The development of an efficient optimal control package. In J. Stoer, editor, *Proceedings of the 8th IFIP Conference on Optimization Techniques (1977), Part 2*, Heidelberg, 1978. Springer.

[46] A. Schäfer, P. Kühl, M. Diehl, J.P. Schlöder, and H.G. Bock. Fast reduced multiple shooting methods for nonlinear model predictive control. *Chemical Engineering and Processing*, 2006. (submitted).

[47] V.H. Schulz. *Reduced SQP methods for large-scale optimal control problems in DAE with application to path planning problems for satellite mounted robots.* PhD thesis, Universität Heidelberg, 1996.

[48] V.H. Schulz. Solving discretized optimization problems by partially reduced SQP methods. *Computing and Visualization in Science*, 1:83–96, 1998.

[49] G.A. Sohl. *Optimal Dynamic Motion Planning for Underactuated Robots.* PhD thesis, University of California, 2000.

[50] O. von Stryk. Numerical solution of optimal control problems by direct collocation. In *Optimal Control: Calculus of Variations, Optimal Control Theory and Numerical Methods*, vol. 129. Bulirsch et al., 1993.

[51] M.J. Tenny, S.J. Wright, and J.B. Rawlings. Nonlinear model predictive control via feasibility-perturbed sequential quadratic programming. *Computational Optimization and Applications*, 28(1):87–121, April 2004.

[52] S. Terwen, M. Back, and V. Krebs. Predictive powertrain control for heavy duty trucks. In *Proceedings of IFAC Symposium in Advances in Automotive Control*, pp. 451–457, Salerno, Italy, 2004.

[53] A. Toumi. *Optimaler Betrieb und Regelung von Simulated Moving Bed Prozessen.* PhD thesis, Fachbereich Bio und Chemieingenieurwesen, Universität Dortmund, 2004.

[54] Oskar von Stryck. Optimal control of multibody systems in minimal coordinates. *Zeitschrift fur Angewandte Mathematik und Mechanik 78, Suppl 3*, 1998.

[55] P.-B. Wieber. *Modélisation et Commande d'un Robot Marcheur Anthropomorphe.* PhD thesis, Ecole des Mines de Paris, 2000.

[56] P.-B. Wieber. Some comments on the structure of the dynamics of articulated motion. In *Proceedings of the Ruperto Carola Symposium on Fast Motion in Biomechanics and Robotics*, 2005.

[57] J. Zhao, M. Diehl, R. Longman, H.G. Bock, and J.P. Schlöder. Nonlinear model predictive control of robots using real-time optimization. In *Proceedings of the AIAA/AAS Astrodynamics Conference*, Providence, RI, August 2004.

Stability Analysis of Bipedal Walking with Control or Monitoring of the Center of Pressure

D. Djoudi and C. Chevallereau

IRCCyN, Ecole Centrale de Nantes, Université de Nantes, UMR CNRS 6597
BP 92101, 1 rue de la Noë, 44321 Nantes cedex 3, France
{Dalila.Djoudi,Christine.Chevallereau}@irccyn.ec-nantes.fr

Summary. The objective of this study is to analyze the stability of two control strategies for a planar biped robot. The unexpected rotation of the supporting foot is avoided via the control of the center of pressure or CoP. For the simultaneous control of the joints and of the CoP, the system is under-actuated in the sense that the number of inputs is less than the number of outputs. Thus a control strategy developed for planar robot without actuated ankles can be used in this context. The control law is defined in such a way that only the geometric evolution of the biped configuration is controlled, but not the temporal evolution. The temporal evolution during the geometric tracking is completely defined and can be analyzed through the study of a model with one degree of freedom. Simple conditions, which guarantee the existence of a cyclic motion and the convergence toward this motion, are deduced. These results are illustrated with some simulation results. In the first control strategy, the position of the CoP is tracked precisely, in the second one, only the limits on the CoP position are used to speed-up the convergence to the cyclic motion.

1 Introduction

The control of many walking robots is based on the notion of center of pressure CoP [11, 12] also called ZMP by Vukobratovic and his co-workers [14, 13]. As long as the CoP remains inside the convex hull of the foot-support, the supporting foot does not rotate and the contact with the ground is guaranteed. Control strategies are often decomposed into a low level and a high level. The low level ensures the tracking of the reference motion, and the high level modifies the reference motion in order to ensure that the CoP remains inside the convex hull of the foot-support.

Since the respect of the expected condition of contact with the ground is more important than a tracking error, this kind of control strategy is interesting. In many experimental works, how to modify the reference motion is not detailed [11], and it seems that this point has not been studied theoretically. The modification of the reference motion has obviously an important effect on the stability of the walking (in the sense of the convergence toward a cyclic motion) and its robustness (in the sense of the reaction of the robot in the presence of perturbation).

Our control strategy is based on simultaneous control of the joint and on the evolution of the CoP position. The unexpected rotation of the supporting foot is avoided via the control of the position of the center of pressure. Since the joints and the position of the CoP are controlled simultaneously, the system becomes under-actuated in the sense that the number of inputs is less than the number of outputs. Thus a control strategy developed for planar robots without actuated ankles can be used in this context [9, 2, 4]. An extension of the work of Westervelt et al. [15], for the completely actuated robot based on a prescribed evolution of the ankle torque was proposed [5]. In the proposed study, the position of the CoP is prescribed, not the ankle torque.

The control law is defined in such a way that only the geometric evolution of the biped configuration is controlled, but not the temporal evolution. This strategy can be seen as an on-line modification of the joint reference motion with respect to time in order to ensure that the position of the center of pressure will be satisfying. The modification of the reference motion corresponds to determine the acceleration along a given path[1] in the joint space. This modification is interesting in the presence of impact, because for all the possible reference motions, the configuration of the robot at impact is the same, and the set of all the reference motions is invariant with respect to impact. As a consequence the impact phase, and the possible variation of the instant of impact have no disturbing effect [3].

Assuming a perfect robot model, and without perturbation, the temporal evolution during the geometric tracking is completely defined and can be analyzed through the study of a model with one degree of freedom. The Poincaré return map can be used to study the stability of the proposed control law.

The practical constraints on the position of the CoP do not imply that this point follows exactly a desired path, but that the position of the CoP evolves between some limits. Thus a second control law is proposed to speed up the convergence to the cyclic motion. The position of the CoP is no longer controlled but only monitored to avoid the rotation of the supporting foot [3]. In this case the control strategy is based on a heuristics proposed by Wieber [16]. In this paper a stability study of this control law is proposed.

Section 2 presents the dynamic model of the biped. A planar biped model with massless feet is considered. Section 3 is devoted to the formulation of the first control strategy and to the existence of a cyclic motion. In Sect. 4 we present the second control strategy. Some simulation results are presented in Sect. 5 in the case of a precise modeling of the robot and in Sect. 6 in the

[1] The time evolution is not specified for a path.

case of an imprecise modeling. Some properties of the two control strategies are given in Sect. 7. Section 8 concludes the paper.

2 Bipedal Model

2.1 The Biped

The biped under study walks in a vertical sagittal xz plane. It is composed of a torso and two identical legs. Each leg consists of two segments and a massless foot. The ankles, the knees and the hips are one-degree-of-freedom rotational frictionless joints. The walk is considered as single support phases separated by impacts (instantaneous double support) with a full contact between the sole of the feet and the ground. The angle of the supporting knee is denoted q_1. The angle of the supporting hip is denoted q_2. The angle of the swing hip is denoted q_3. The angle of the swing knee is denoted q_4. During swing phase the foot is aligned horizontally thus the angle of the swing ankle can be calculated. The supporting ankle angle allows to choose the orientation of the supporting shank with respect to the vertical q_5. Vector $q = [q_1, q_2, q_3, q_4, q_5]^T$ of variables (Fig. 1) describes the shape of the biped during single support. Since the free foot is massless no torque is required at the swing ankle. The torque at the supporting ankle will be treated in a special way thus it is denoted $\Gamma_a = \Gamma_5$. The torques are grouped into a torque vector $\Gamma = [\Gamma_1, \Gamma_2, \Gamma_3, \Gamma_4, \Gamma_5]^T$.

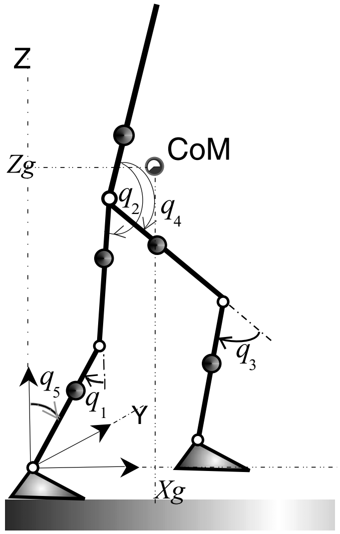


Fig. 1. The studied biped: generalized coordinates

In the simulation, we use the following biped parameters [2]. The lengths of the thighs and of the shanks are 0.4 m. However, their masses are different: 6.8 kg for each thigh and 3.2 kg for each shank. The length of the torso is 0.625 m and its mass is 17.05 kg. The center of mass are placed on the line representing the link in Fig. 1. The distance between the joint actuator and the center of mass is 0.1434 m for the torso, 0.163 m for the shanks, and 0.123 m for the thigh. The moments of inertia of the segments are also taken into account, there values are defined around the joint axis, and there value are 1.8694 kgm^2 for the torso, 0.10 kgm^2 for the shanks, and 0.25 kgm^2 for the thigh. The inertia of the motor of the hip and of the knee are 0.83 kgm^2. The feet is massless and have no inertia. The size of the feet are $h_p = 0.08\,\text{m}$, $l_{min} = 0.06\,\text{m}$, $l_{max} = 0.2\,\text{m}$ (Fig. 2).

2.2 Dynamic Modeling

The walking gait is composed of stance phases. A passive impact separates the stance phases. The legs swap their roles from one step to the next one. Thus the study of a step allows us to deduce the complete behavior of the robot. Only a single support phase and an impact model are derived.

The Single Support Phase Model

Using Lagrange's formalism, the i^{th} line of the dynamic model for $i = 1, \ldots, 5$ (q_i is the i^{th} element of vector q) is:

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} + \frac{\partial P}{\partial q_i} = Q_i \tag{1}$$

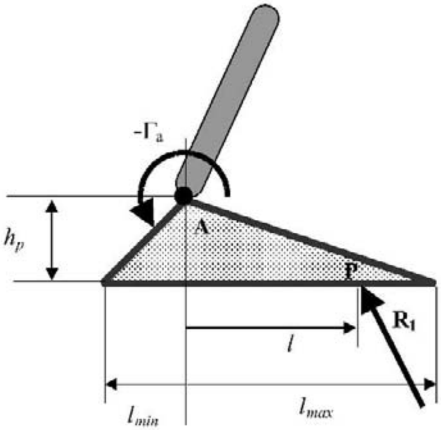


Fig. 2. The equilibrium of the supporting foot

where K is the kinetic energy and P is the potential energy. The virtual work δW of the external torques and forces, given by expression $\delta W = \sum Q_i \delta q_i = Q^T \delta q$, defines the vector Q of the generalized forces.

The kinetic energy K is independent of the coordinate frame chosen. Since coordinate q_5 defines the orientation of the biped as a rigid body, the inertia matrix is independent of this variable, it depends only of "internal" variables represented by vector $q_c = [q_1, q_2, q_3, q_4]^T$.

The dynamic model can be written:

$$M(q_c)\ddot{q} + h(q, \dot{q}) = \Gamma \tag{2}$$

where $M(q_c)$ is a (5×5) inertia matrix and vector $h(q, \dot{q})$ contains the centrifugal, Coriolis and gravity forces.

The fifth equation of system (1) is:

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_5}\right) + \frac{\partial P}{\partial q_5} = \Gamma_a \tag{3}$$

For our planar biped and our choice of the coordinates in the single support, the term $\frac{\partial K}{\partial \dot{q}_5}$ is the angular momentum of the biped about the stance ankle A (Fig. 2). We denote this term by σ_A. Thus we have:

$$\frac{\partial K}{\partial \dot{q}_5} = \sigma_A = N(q_c)\dot{q} \tag{4}$$

where $N(q_c)$ is the fifth line of the inertia matrix $M(q_c)$.

The expression $\frac{\partial P}{\partial q_5}$ is equal to $-mgx_g$ if the abscissa of the stance ankle is 0, m is the mass of the biped, g is the gravity acceleration. Thus the fifth equation of the dynamic model of the biped in the single support can be written in the following simple form:

$$\dot{\sigma}_A - mgx_g = \Gamma_a \tag{5}$$

The Reaction Force During the Single Support Phase

The position of the mass center of the biped can be expressed as function of the angular coordinates vector q:

$$\begin{bmatrix} x_g \\ z_g \end{bmatrix} = \begin{bmatrix} f_{xi}(q) \\ f_{zi}(q) \end{bmatrix} \tag{6}$$

The vector-function $f_i(q) = [f_{xi}(q) \quad f_{zi}(q)]^T$ depends on vector q and on the biped parameters (lengths of the links, masses, positions of the centers of mass). The index i denotes the stance leg, for support on leg 1, $f_1(q)$ is used.

When leg 1 is on the ground, a ground reaction force, R_1, exists. The global equilibrium of the robot makes it possible to calculate this force:

$$m \begin{bmatrix} \ddot{x}_g \\ \ddot{z}_g \end{bmatrix} + mg \begin{bmatrix} 0 \\ 1 \end{bmatrix} = R_1 \tag{7}$$

Equation (7) can also be written:

$$\begin{aligned} m\frac{\partial f_{x1}(q)}{\partial q}\ddot{q} + m\dot{q}^T \frac{\partial^2 f_{x1}(q)}{\partial q^2}\dot{q} &= R_{x1} \\ m\frac{\partial f_{z1}(q)}{\partial q}\ddot{q} + m\dot{q}^T \frac{\partial^2 f_{z1}(q)}{\partial q^2}\dot{q} + mg &= R_{z1} \end{aligned} \tag{8}$$

where $\frac{\partial^2 f_{x1}(q)}{\partial q^2}$ and $\frac{\partial^2 f_{z1}(q)}{\partial q^2}$ are (5×5) matrices.

Equilibrium of the Supporting Foot

The supporting foot is exposed to the ground reaction force and the ankle torque $-\mathit{\Gamma}_a$. The equilibrium law gives:

$$-\mathit{\Gamma}_a - lR_{z1} - h_p R_{x1} = 0 \tag{9}$$

Thus if the horizontal CoP position is l then the torque at the supporting ankle is, using (7):

$$\mathit{\Gamma}_a = -l(m\ddot{z}_g + mg) - h_p(m\ddot{x}_g) \tag{10}$$

The horizontal CoP position l is directly defined by the robot dynamics as it can be seen in the following equation obtained by combining equations (4), (5), (6), (8) and (10):

$$(N_0(q) + lN_l(q))\ddot{q} + h_0(q, \dot{q}) + lh_l(q, \dot{q}) = 0 \tag{11}$$

with

$$\begin{aligned} N_0 &= N(q_c) + mh_p\frac{\partial f_{x1}(q)}{\partial q} \\ N_l &= m\frac{\partial f_{z1}(q)}{\partial q} \\ h_0 &= \dot{q}^T \frac{\partial N(q_c)}{\partial q}\dot{q} - mgf_{x1}(q) + mh_p\dot{q}^T \frac{\partial^2 f_{x1}(q)}{\partial q^2}\dot{q} \\ h_l &= m\dot{q}^T \frac{\partial^2 f_{z1}(q)}{\partial q^2}\dot{q} + mg \end{aligned}$$

The Impact Model

When the swing leg 2 touches the ground with a flat foot at the end of the single support of leg 1, an inelastic impact takes place. We assume that the ground reaction force at the instant of impact is described by a Dirac delta-function with intensity I_{R_2}. The velocity of foot 2 becomes zero just after the impact. Two kinds of impact can occur depending on whether the stance leg takes off or not. Here, for simplicity, we study walking with instantaneous double support. Thus at impact the stance leg 1 takes off and there is no impulsive ground reaction force on leg 1. The robot configuration q is assumed to be constant at the instant of impact, and there are jumps in the velocities. The velocity vectors just before and just after impact, are denoted $\dot{q}^-$ and $\dot{q}^+$ respectively. The torques are limited, thus they do not influence the instantaneous double support. It can be shown that the impact model can be written as [4]:

$$\dot{q}^+ = E(\triangle(q)\dot{q}^-) \tag{12}$$

where $\triangle(q)$ is a 5×5 matrix, and E is a permutation function describing the legs exchange. For the following single support phase the joints are relabelled in order to study only one dynamic model for single support (SS) and to take into account the change on the supporting ankle.

Intensity I_{R_2} of the impulsive reaction force is:

$$I_{R_2} = m\left(\frac{\partial f_2(q)}{\partial q}\triangle(q) - \frac{\partial f_1(q)}{\partial q}\right)\dot{q}^- \tag{13}$$

3 The First Control Law

In this study, walking is considered as single support phases with a full foot contact. While this is not a necessary condition for walking, and animals and humans do not enforce this constraint during walking, many control algorithms for bipedal robots enforce this constraint during walking in order to prevent difficulties associated with the loss of actuation authority when the foot rotates. To avoid foot rotation, the CoP must be inside the supporting area [13]. In order to ensure this behavior, the CoP position is controlled to follow a desired path l^d [11], but as shown in the previous section, the position of the CoP is directly connected to the dynamics of the motion. It is not possible to prescribe independently a desired evolution of the joints $q^d(t)$ and of the position of the CoP $l^d(t)$. With respect to such a task, the biped can be seen as an under-actuated system, and the control strategy developed for such a system can be used. Thus, the objective of the control law presented in this section is only to track a reference path for q and l rather than a reference motion [4]. A motion differs from a path by the fact that a motion is a temporal evolution along a path. A joint path is the projection of a joint motion in the joint space. The difference between motion and path are illustrated on Fig. 3 for a two joint-robot.

Only a geometrical tracking is desired and a time scaling control [6] is used. A reference joint path is assumed to be known. Thus the desired configuration of q and l for the biped are not expressed as a function of time. But they are

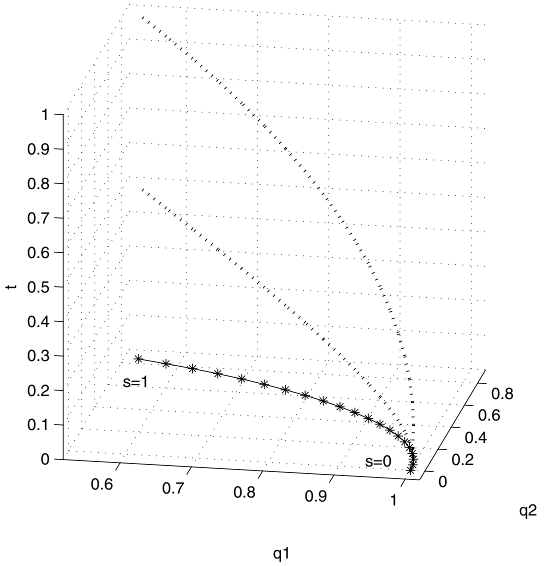


Fig. 3. The dotted lines are two motions $(q_1(t), q_2(t))$ corresponding to the same path represented by the solid line. A path is a line in the joint space, this line can be graduated as a function of a new variable denoted s, and then can be expressed by $(q_1(s), q_2(s))$. This function s is defined such that the initial configuration correspond to $s = 0$, the final configuration corresponds to $s = 1$. Any monotonic function $s(t)$ defines a motion corresponding to the path $q(s)$. For example $s = t/T$ defines a motion of duration T. If a joint variable, for example q_2, has a monotonic evolution along the path, the path can also be parametrized by q_2, in this case it can be expressed as $q_1(q_2)$

function of the scalar path parameter s, a normalized virtual time: $q^d(s), l^d(s)$. The desired walking of the robot corresponds to an increasing function $s(t)$. This function. $s(t)$ is not known a priori, the set of all the motions that correspond to the desired path is considered.

The proposed strategy can be extended without difficulty to walking including a rotation about the toe of the supporting foot, since this phase corresponds to a motion such that the position of the center of pressure is imposed. The main difficulty is that a sub-phase must be added [5].

3.1 Choice of a Reference Path

The reference path $q^d(s), l^d(s)$ is designed in order to obtain cyclic motion of the biped. The walk is composed of single supports separated by instantaneous passive impacts. The legs swap their roles from one step to the next one, so the reference path needs only to be defined for a single step. The evolution of the path parameter s along the step k is denoted $s_k(t)$, the scalar path parameter s_k must increase strictly monotonically with respect to time from 0 to 1.

For $0 < s_k(t) < 1$, the robot configuration $q^d(s_k)$ is such that the swing leg is above the ground. The swing leg touches the ground at $s_k = 0, 1$ exactly. In consequence for any increasing function $s_k(t)$ from 0 to 1, the configuration of the biped at impact is the expected one. The control inputs are the torques. The torque acts on the second derivative of q and directly on l via the dynamic model. Thus the reference trajectory $q^d(s_k)$ must be twice differentiable, but no continuity condition exists for $l^d(s_k)$. Vectors $q^d(0)$ and $q^d(1)$ describe the initial and final biped configurations of the biped during a single support. As the legs swap their roles from one step to the following one the desired configurations are such that $q^d(1) = E(q^d(0))$ where E is a permutation function describing the leg exchange.

The reference path is defined such that if the reference path is exactly tracked before the impact then the reference path is exactly tracked after the impact. If the reference path is perfectly tracked, before the impact $k+1$, the vector of joint velocities is $\dot{q}^- = \frac{dq^d(1)}{ds}\dot{s}_k(1)$ and after the impact, $\dot{q}^+ = \frac{dq^d(0)}{ds}\dot{s}_{k+1}(0)$. The velocity at the end and at the beginning of the step are connected by the impact model and the legs exchange (12). Thus we have:

$$\frac{dq^d(0)}{ds}\dot{s}_{k+1}(0) = E(\triangle(q^d(1))\frac{dq^d(1)}{ds}\dot{s}_k(1)) \tag{14}$$

We choose:

$$\frac{dq^d(0)}{ds} = E(\triangle(q^d(1))\frac{dq^d(1)}{ds}\alpha) \tag{15}$$

With this choice we have the following equality: $\dot{s}_{k+1}(0) = \frac{\dot{s}_k(1)}{\alpha}$.

For configuration $q^d(1)$, and vector $\frac{dq^d(1)}{ds}$ the amplitude of the vector $\frac{dq^d(0)}{ds}$ can be modified by the choice the values of α (but not its direction). This point will be commented in Sect. 5.1.

Some hypotheses (no sliding, no rotation of the feet, take-off of the previous supporting leg) are made on the behavior of the robot at the impact, the corresponding constraints on the joint trajectory can be deduced [4, 7].

3.2 Definition of the Control Law

The control law must ensure that the joint coordinates follow the joint reference path $q^d(s)$ and that the position of the CoP is $l^d(s)$. It follows from the definition of the joints reference path that the desired velocity and acceleration of the joint variables are:

$$\begin{aligned} \dot{q}^d(t) &= \frac{dq^d(s(t))}{ds}\dot{s} \\ \ddot{q}^d(t) &= \frac{dq^d(s(t))}{ds}\ddot{s} + \frac{d^2q^d(s(t))}{ds^2}\dot{s}^2 \end{aligned} \tag{16}$$

The increasing function $s(t)$ defines the desired motion, but since the control objective is only to track a reference path, the evolution $s(t)$ is free and the second derivative $\ddot{s}$ will be treated as a "supplementary control input". Thus, the control law will be designed for a system with equal number of inputs and outputs. The control inputs are the five torques $\Gamma_j, j = 1, \ldots, 5$, plus $\ddot{s}$. The chosen outputs are the five angular variables of vector $q(t) - q^d(s(t))$ and $l(t) - l^d(s(t))$.

The control law is a non-linear control law classically used in robotics. But in order to obtain a finite-time stabilization of one of the desired trajectories, the feedback function proposed by Bhat and Berstein is used [1, 9]. The joint tracking errors are defined with respect to the trajectories satisfying (16):

$$\begin{aligned} e_q(t) &= q^d(s(t)) - q(t) \\ \dot{e}_q(t) &= \frac{dq^d(s(t))}{ds}\dot{s} - \dot{q}(t) \end{aligned} \tag{17}$$

The desired behavior in closed loop is:

$$\ddot{q} = \ddot{q}^d + \frac{1}{\epsilon^2}\psi \tag{18}$$

where ψ is a vector of five components $\psi_l, l = 1, \ldots, 5$ with:

$$\psi_l = -sign(\epsilon \dot{e}_{q_l})|\epsilon \dot{e}_{q_l}|^\nu - sign(\phi_l)|\phi_l|^\nu \tag{19}$$

and $0 < \nu < 1$, $\epsilon > 0$, $\phi_l = e_{q_l} + \frac{1}{2-\nu} sign(\epsilon \dot{e}_{q_l})|\epsilon \dot{e}_{q_l}|^{2-\nu}$, ν and ϵ are parameters to adjust the settling time of the controller. Taking into account the expression of the reference motion, (18) can be rewritten as:

$$\ddot{q} = \frac{dq^d(s)}{ds}\ddot{s} + v(s, \dot{s}, q, \dot{q}) \tag{20}$$

with

$$v(s, \dot{s}, q, \dot{q}) = \frac{d^2 q^d(s)}{ds^2}\dot{s}^2 + \frac{1}{\epsilon^2}\psi$$

For the position of the CoP, the desired closed loop behavior is:

$$l(t) = l^d(s(t))$$

The dynamic model of the robot is described by eq. (2). The position of the CoP is defined via (11). Thus the control law must be such that:

$$\begin{aligned} M(q)(\frac{dq^d(s)}{ds}\ddot{s} + v) + h(q, \dot{q}) &= \Gamma \\ (N_0(q) + l^d(s)N_l(q))(\frac{dq^d(s)}{ds}\ddot{s} + v) + h_0(q, \dot{q}) + l^d(s)h_l(q, \dot{q}) &= 0 \end{aligned} \tag{21}$$

We can deduce that, in order to obtain the desired closed loop behavior, it is necessary and sufficient to choose:

$$\ddot{s} = \frac{-(N_0(q) + l^d(s)N_l(q))v - h_0(q,\dot{q}) - l^d(s)h_l(q,\dot{q})}{(N_0(q) + l^d(s)N_l(q))\frac{dq^d(s)}{ds}} \tag{22}$$

$$\Gamma = M(q)\left(\frac{dq^d(s)}{ds}\ddot{s} + v\right) + h(q,\dot{q}) \tag{23}$$

If $(N_0(q) + l^d(s)N_l(q))\frac{dq^d(s)}{ds} \neq 0$, the control law (22)–(23) ensures that $q(t)$ converges to $q^d(s(t))$ in a finite time, which can be chosen as less than the duration of one step [1, 9], and that $l(t) = l^d(s(t))$. Without initial errors, a perfect tracking of $q^d(s(t))$ and $l^d(s)$ is obtained.

3.3 Stability Study

Our main goal is to design a control strategy that ensures a stable periodic motion of the biped. The control law (22)–(23) ensures that the motion of the biped converges in a finite time toward a reference path. The settling time can be chosen to be less than the duration of the first step. Since the impact is a geometric condition and due to the characteristics of the joints reference path (Sect. 3.1), any step k begins with $s_k = 0$ and finishes with $s_k = 1$. Since the control law is designed to converge before the end of the first step and since the reference path is such that if the tracking is perfect before the impact, it will be perfect afterward, after the first step a perfect tracking is obtained. The biped with control law (22)–(23) follows perfectly the reference path, starting from the second step. Thus:

$$\begin{aligned} q(t) &= q^d(s(t)) \\ \dot{q}(t) &= \frac{dq^d(s)}{ds}\dot{s}(t) \\ \ddot{q}(t) &= \frac{dq^d(s)}{ds}\ddot{s}(t) + \frac{d^2q^d(s)}{ds^2}\dot{s}(t)^2 \\ l(t) &= l^d(s(t)) \end{aligned} \tag{24}$$

These equations define the zero dynamics corresponding to the proposed control law. To know whether a cyclic motion will be obtained, the behavior of the evolution of $\dot{s}_k(t)$ is studied for $k = 2 \ldots \infty$. The dynamics of s is deduced from the dynamic model (11) with the condition (24). The acceleration $\ddot{s}$ is:

$$(N_{s0}(s) + l^d(s)N_{sl}(s))\ddot{s} + h_{s0}(s,\dot{s}) + l^d(s)h_{sl}(s,\dot{s}) = 0 \tag{25}$$

with

$$
\begin{aligned}
N_{s0} &= (N(q^d(s)) + m h_p \frac{\partial f_{x1}(q)}{\partial q}) \frac{dq^d(s)}{ds} \\
N_{sl} &= m \frac{\partial f_{z1}(q)}{\partial q} \frac{dq^d(s)}{ds} \\
h_{s0} &= \left[\frac{dq^d(s)}{ds}^T \left(\frac{\partial N(q)}{\partial q} + m h_p \frac{\partial^2 f_{x1}(q)}{\partial q^2} \right) \frac{dq^d(s)}{ds} \right] \dot{s}^2 \\
&\quad + \left[\left(N(q^d(s)) + m h_p \frac{\partial f_{x1}(q)}{\partial q} \right) \frac{d^2 q^d(s)}{ds^2} \right] \dot{s}^2 \qquad (26) \\
&\quad - m g f_{x1}(q^d(s)) \\
h_{sl} &= m \left[\frac{dq^d(s)}{ds}^T \frac{\partial^2 f_{z1}(q)}{\partial q^2} \frac{dq}{ds} + \frac{\partial f_{z1}(q)}{\partial q} \frac{d^2 q^d(s)}{ds^2} \right] \dot{s}^2 + m g
\end{aligned}
$$

This equation along with the constraints (24) describe completely the behavior of the system.

One single support phase begins with $s = 0$ and finishes with $s = 1$. The evolution of $\dot{s}_{k+1}$ during the step $k + 1$ is uniquely defined by initial value $\dot{s}_{k+1}(0)$. The integration of (25) along one step, starting with $\dot{s}_{k+1}(0)$, defines the final value $\dot{s}_{k+1}(1)$.

The single support phases are separated by impact phases; the evolution of the zero dynamics is such that s restarts with $s = 0$ and $\dot{s}_{k+1}(0) = \frac{\dot{s}_k(1)}{\alpha}$ (due to the definition of the reference joint path (15)). Thus the final value of $\dot{s}_{k+1}(1)$ can be easily defined numerically as a function of $\dot{s}_k(1)$, we define function φ by: $\dot{s}_{k+1}(1) = \varphi(\dot{s}_k(1))$. The existence of a cyclic motion and the convergence to it can be studied via function φ as it is classically done using the method introduced by H. Poincaré [9, 10]. The fixed point of this function defines the cyclic velocity $\dot{s}_c(1)$, it corresponds to the intersection between the function φ and the identity function. If the norm Δ of the slope of the function φ at $\dot{s}_c(1)$ is less than 1, then for an initial state close to the cyclic motion, the biped motion will converge toward the cyclic motion.

If the desired evolution of the position of the CoP is piecewise constant, the stability analysis can be conducted mostly analytically [7]. If the desired evolution of the position of the CoP is arbitrary, the stability analysis is conducted numerically in this chapter.

4 The Second Control Law

The physical constraint on the position of the CoP is that the position of the CoP is between l_{min} and l_{max} but it is not necessary that $l(s)$ follows exactly $l^d(s)$. If a cyclic motion corresponding to $q^d(s)$, $l^d(s)$ exists, it can be interesting to converge quickly toward this cyclic motion defined by $\dot{s}(t) = \dot{s}_c(t)$. The corresponding cyclic motion can be defined by the stability study of the first control law. Now we assume that the corresponding cyclic motion is given as a function $\dot{s}_c(s)$ for $0 \leq s \leq 1$. To achieve this objective, the constraint $l(s) = l^d(s)$ can be relaxed to: $l_{min} < l(s) < l_{max}$.

To converge toward the cyclic orbit in the phase plane $s, \dot{s}$, we define an error between the current state and the orbit:

$$e_v = \dot{s}(s) - \dot{s}_c(s) \tag{27}$$

and to nullify this error the desired acceleration $\ddot{s}^d$ is chosen such that: $\dot{e}_v + K_{vs} e_v = 0$ where K_{vs} defines the convergence rate to the cyclic motion. Thus the desired acceleration is:

$$\ddot{s}(s)^d = \frac{d(\dot{s}_c(s))}{ds}\dot{s} + K_{vs}(\dot{s}_c(s) - \dot{s}(s)) \tag{28}$$

But the position l of the CoP, and the acceleration $\ddot{s}$ are linked by the dynamic model. And even if the constraint on l is relaxed, the condition of non-rotation of the feet holds, and l is monitored to be within the domain $\Im =]l_{min}, l_{max}[$ in all the control process. If the same closed loop behavior is desired for the joint variables (22), gives:

$$\ddot{s} = \frac{-N_0(q)v - h_0(q,\dot{q}) + l(N_l(q)v - h_l(q,\dot{q}))}{N_0(q)\frac{dq^d(s)}{ds} + lN_l(q)\frac{dq^d(s)}{ds}} \tag{29}$$

where l must be chosen such that $l \in \Im$. Differentiating (29) with respect to l shows that $\ddot{s}$ is monotonic with respect to l. Thus the limits $l_{min} < l < l_{max}$ can be easily transformed with limits on $\ddot{s}$. For this purpose, the extreme values for $\ddot{s}$ are defined as follows:

$$\begin{aligned} u_1 &= \frac{-N_0(q)v - h_0(q,\dot{q}) + l_{min}(N_l(q)v - h_l(q,\dot{q}))}{N_0(q)\frac{dq^d(s)}{ds} + l_{min}N_l(q)\frac{dq^d(s)}{ds}} \\ u_2 &= \frac{-N_0(q)v - h_0(q,\dot{q}) + l_{max}(N_l(q)v - h_l(q,\dot{q}))}{N_0(q)\frac{dq^d(s)}{ds} + l_{max}N_l(q)\frac{dq^d(s)}{ds}} \end{aligned} \tag{30}$$

For given values of s, $\dot{s}$, two cases occur depending on whether the denominator can be zero or not for $l \in \Im$. The denominator is zero for $l(s) = -\frac{N_0(q)\frac{dq^d(s)}{ds}}{N_l(q)\frac{dq^d(s)}{ds}}$. If for any l such that $l \in \Im$, the denominator of eq. (29) is not zero, then $\ddot{s}$ is bounded for any acceptable value l and $\min(u_1, u_2) < \ddot{s} < \max(u_1, u_2)$. If for one value l such that $l \in \Im$, the denominator of eq. (29) is zero, then $\ddot{s}$ is unbounded and $\ddot{s}$ cannot be in the interval $]\min(u_1, u_2), \max(u_1, u_2)[$ with acceptable values of l.

Thus the proposed control law is the following: like the previous control law, the reference path $q^d(s)$ is tracked using eq. (23) but eq. (22), which corresponds to the constraint $l(s) = l^d(s)$, is replaced by the following:

$$\begin{aligned}
&\text{if } -\frac{N_0(q)\frac{dq^d(s)}{ds}}{N_l(q)\frac{dq^d(s)}{ds}} < l_{min} \text{ or } -\frac{N_0(q)\frac{dq^d(s)}{ds}}{N_l(q)\frac{dq^d(s)}{ds}} > l_{max} \text{ then}\\
&\quad \ddot{s} = \begin{cases} \min(u_1,u_2), & \text{if } \ddot{s}^d < \min(u_1,u_2)\\ \max(u_1,u_2), & \text{if } \ddot{s}^d > \max(u_1,u_2)\\ \ddot{s}^d, & \text{otherwise} \end{cases}\\
&\text{if } l_{min} < -\frac{N_0(q)\frac{dq^d(s)}{ds}}{N_l(q)\frac{dq^d(s)}{ds}} < l_{max} \text{ then}\\
&\quad \ddot{s} = \begin{cases} \min(u_1,u_2), & \text{if } \min(u_1,u_2) < \ddot{s}^d \leq u_{12}\\ \max(u_1,u_2), & \text{if } u_{12} < \ddot{s}^d < \max(u_1,u_2)\\ \ddot{s}^d, & \text{otherwise} \end{cases}
\end{aligned} \tag{31}$$

where $u_{12} = \frac{\min(u_1,u_2)+\max(u_1,u_2)}{2}$. This control law ensures a convergence to the cyclic motion with a convergence rate defined by K_{vs} under the constraint $l \in \Im$.

The control law (31), (23) ensures that $q(t)$ converges to $q^d(s(t))$ in a finite time, which can be chosen less than the duration of one step [1, 9], and ensures that $l \in \Im$. The biped with control law (31), (23) follows perfectly the reference path after this first step. To know if a cyclic motion will be obtained, the behavior of the evolution of $\dot{s}_k(t)$ is studied for $k = 2\ldots\infty$ and for an initial velocity $\dot{s}_2(0)$. The stability analysis is done numerically like for the first control law.

The convergence rate to the cyclic motion depends on the choice of the value K_{vs}. Higher values of K_{vs} speed up convergence toward the cyclic motion if there is no saturation due to the limits on l.

5 Walking Simulation using Correct Model Parameters

5.1 A Reference Path

The proposed control law was tested on the reference path corresponding to the stick-diagram presented in Fig. 4, for the biped presented in Fig. 1. The joint path $q^d(s)$ is defined by a fourth order polynomial evolution with respect to s.

This reference path has been defined to produce an optimal cyclic motion for the robot Rabbit [2], this robot has the same physical property that the robot described in Sect. 2 but Rabbit has no feet ($h_p = 0$, $l = 0$). As the studied robot has feet, and a linear evolution of the position of the CoP is considered, the existence of a cyclic motion is not ensured and if it exists it is of course not optimal.

For the robot without feet, the optimization process is described in [8]. The reference path is described by an instantaneous double support configuration

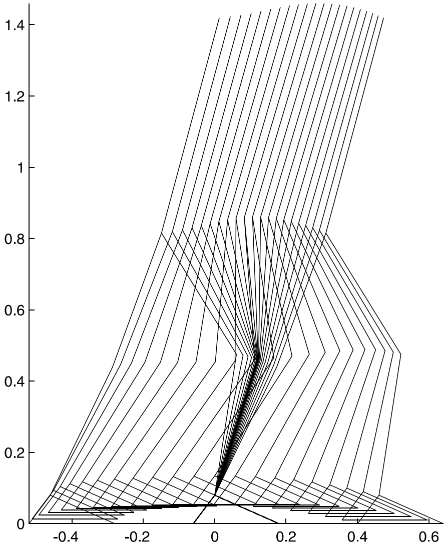


Fig. 4. The stick diagram of the desired trajectory. The configuration of the robot are drawn for $s = 0, 0.05, 0.1, 0.15 \ldots, 0.95, 1$. Thus a sequence of pictures of the robot are given. The desired motions of the robot are such that the configuration of the robot coincides at some instant to each picture, but it is not imposed that these instants are equally distributed in the period of one step

$q^d(1)$, the final direction of the joint velocity $\frac{q^d(1)}{ds}$, an intermediate single support configuration $q^d(0.5)$, and α. The initial double support configuration is defined by permutation: $q^d(0) = Eq^d(1)$. The direction of the initial velocity is defined by equation (15). Then the desired path is determined by a polynomial 4th order function of s connecting these configurations and velocities. The integral of the norm of the torque for the cyclic motion is minimized for a given advance velocity. The free leg tip must be above a sinusoidal function with a maximum of 5 cm. The limit of the actuator are taken into account (maximal torque less than 150 Nm). The reference path corresponding to the Fig. (4) is obtained for given advance velocity $vel = 1.41\,\mathrm{ms}^{-1}$. The optimal solution is such that: $q^d(1) = [5.73° \ 185.08° \ 40.43° \ 133.33° \ 25.81°]^T$, $\frac{q^d(1)}{ds} = [3.57° s^{-1} \ 32.60° s^{-1} \ -61.60° s^{-1} \ 0.09° s^{-1} \ 29.50° s^{-1}]^T$, $q^d(0.5) = [19.97° \ 161.22° \ 42.51° \ 154.93° \ 17.72°]^T$ and $\alpha = 1.98$

5.2 The First Control Law

For this joints path, a linear evolution of the CoP position is chosen. When s varies from 0 to 1, l^d varies from $-0.06\,\mathrm{m}$ to $0.18\,\mathrm{m}$.

The control law imposes that $q(s) = q^d(s)$, $l(s) = l^d(s)$ after the first step. The stability of the complete system is determined by the evolution of $s(t)$. It

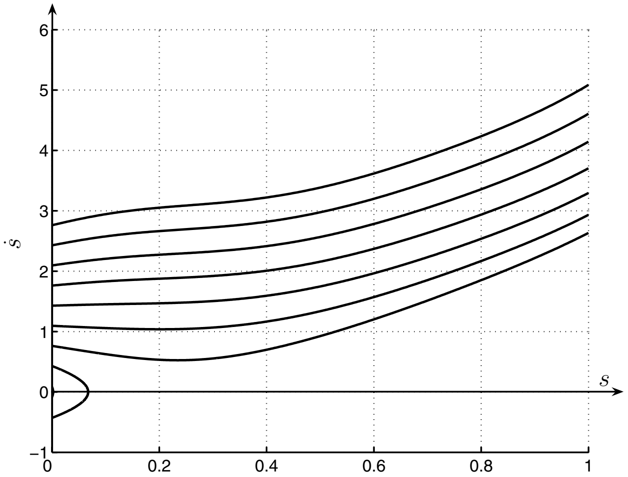


Fig. 5. The phase plane for the zero dynamics (25) on single support: $\dot{s}$ against s

can be described in a phase plane $s, \dot{s}$ for $0 \leq s \leq 1$ initialized with different values for $\dot{s}$. For example, the phase plane is shown in Fig. 5.

For a sufficiently high initial velocity $\dot{s}(0)$, successful stepping pattern can be achieved. At low initial velocity $\dot{s}(0)$, the robot falls back. Taking the impact phase into account (here $\alpha = 1.98$), the Poincaré return map $\dot{s}_{k+1}(1) = \varphi(\dot{s}_k(1))$ is drawn in Fig. 6. For the example the cyclic motion is such that $\dot{s}_c(1) = 3.9\,\mathrm{s}^{-1}$. The corresponding average motion velocity is $vel = 1.5$ m/s. The slope of function φ is estimated numerically: $\Delta = 0.68$; it is less than 1, thus the proposed control law is stable. The minimal value and the maximal value of the velocity $\dot{s}_k(1)$ such that the step can be achieved are defined numerically. For smaller initial velocities the biped falls back, for higher velocities the biped takes off since the normal ground reaction vanishes.

Assuming no modeling error and initializing the state of the biped out of the periodic orbit (with an initial velocity 60% higher than the cyclic value), the results of one simulation for 20 walking steps are illustrated in Fig. 7. The convergence toward a cyclic motion can be shown for the five joints via their evolution in their phase plane. For example the evolution of the trunk is shown in Fig. 7-a. This convergence is also illustrated via the evolution of the position of the CoP with respect to time in Fig. 7-b. For each step, this evolution is linear from $-0.06\,\mathrm{m}$ to $0.18\,\mathrm{m}$, but the duration of the step varies. At the beginning;, the steps are faster and then a cyclic behavior is obtained. Figure 7-c presents the time-history of $\dot{s}$, it clearly converges toward a cyclic motion, the final value of $\dot{s}$ before each impact is the cyclic value obtained on the Poincaré map.

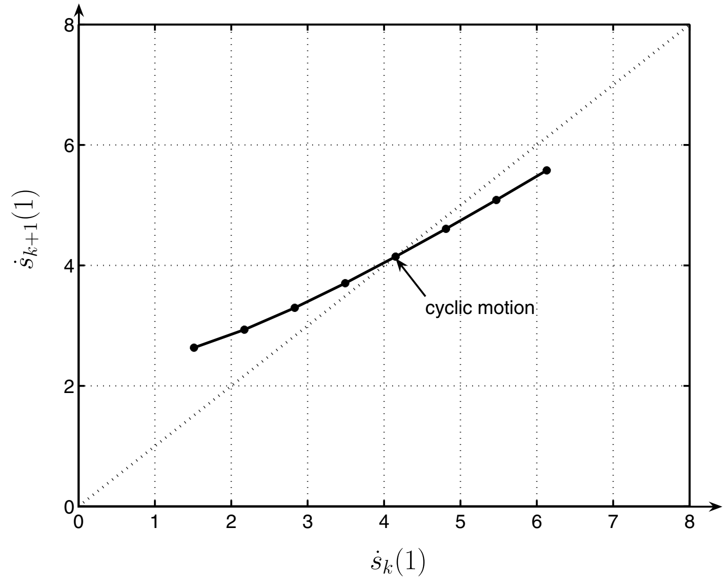


Fig. 6. The Poincaré map: $\dot{s}_{k+1}(1) = \varphi(\dot{s}_k(1))$

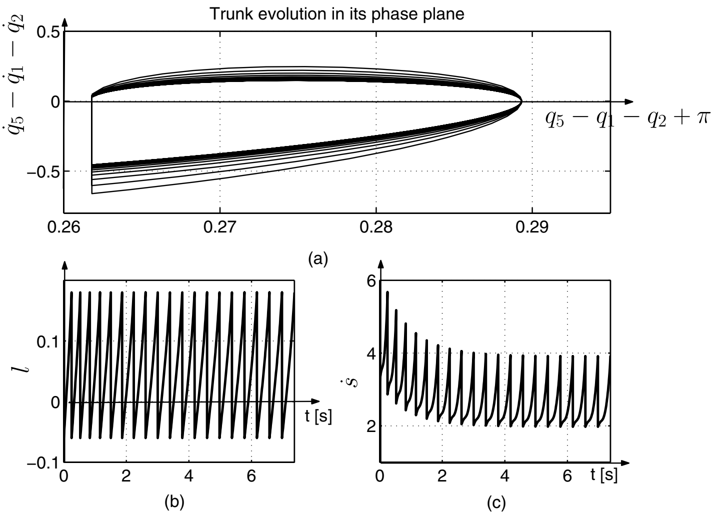


Fig. 7. The convergence toward a cyclic motion is observed in simulation with the proposed control law, without modeling error. (**a**) The trunk evolution is drawn in its phase phase (the absolute trunk velocity with respect to the absolute trunk orientation), at impact the velocity changes but not the orientation. It tends toward a limit cycle. (**b**) During each step, the *horizontal* position of the CoP with respect to time $l(t)$ evolves from -0.06 m to 0.18 m. The duration of the step tends toward a constant value. (**c**) $\dot{s}(t)$ tends toward a cyclic motion

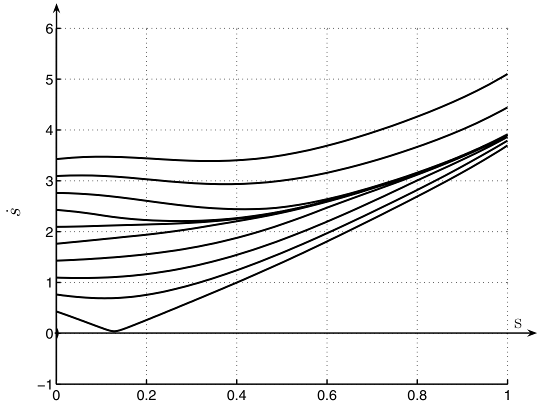


Fig. 8. The phase plane for the zero dynamics on single support, (31), for the second control law with $K_{vs} = 20$

5.3 The Second Control Law

The second control law was tested on the same reference trajectory $q^d(s)$. The desired evolution of $\dot{s}(s)$ is the cyclic motion corresponding to the previous control law.

The control law imposes some constraints $q(s) = q^d(s)$ that are assumed to be perfectly respected. The free dynamics that results from imposing these constraints on the system configuration are described by $s, \dot{s}$ and eq. (31) can be represented in the phase plane. The phase plane is shown in Fig. 8 for $K_{vs} = 20$.

The convergence toward the cyclic motion is clear when Figs. 5 and 8 are compared. When $K_{vs} = 20$, for initial velocities varying from 1.4 to 2.8, the cyclic motion is reached in one step. This feature gives a horizontal behavior of the Poincaré map about the fixed point. The motion can be initiated with a lower velocity $\dot{s}(0)$ than for the first control law because when the current motion converges toward the cyclic motion, it helps prevent the biped from falling back.

The control strategy is properly illustrated by the evolution of $l(s)$ corresponding to the evolution of the biped for various initial velocities $\dot{s}(0)$ in Fig. 9. When the real motion of the biped is slower than the cyclic one, the position of the CoP is moved backwards to increase the motion velocity until the limit l_{min} is reached. When the real motion of the biped is faster than the cyclic one, the position of the CoP is moved forwards to decrease the motion velocity until the limit l_{max} is reached. With a high gain, the position of the CoP is on the limit almost all the time

The single support phases are separated by impact phases. The Poincaré return maps can be deduced and are presented in the Fig. 10, for $K_{vs} = 2$ and $K_{vs} = 20$.

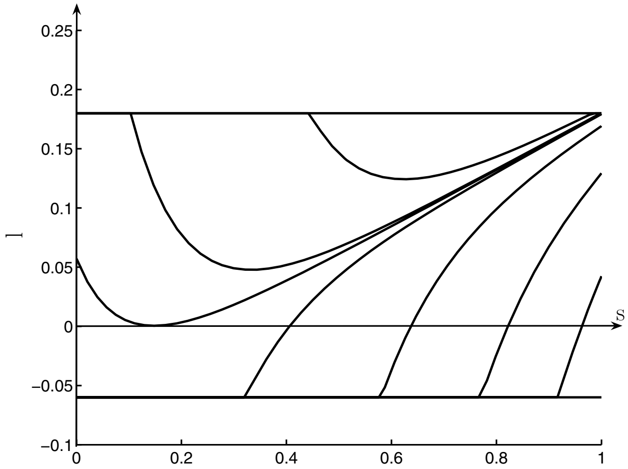


Fig. 9. The evolution of the position of the CoP $l(s)$, for various initial velocities $\dot{s}(0)$, for the second control law with $K_{vs} = 20$

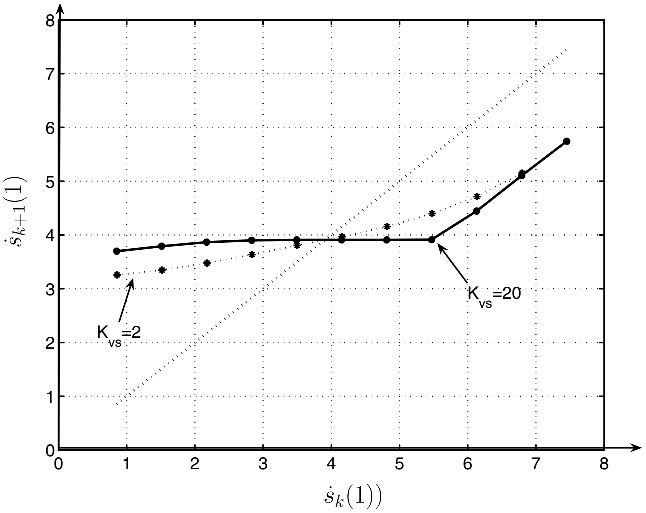


Fig. 10. The Poincaré return map for the second control law with $K_{vs} = 2$ (*solid line*) and $K_{vs} = 20$ (*dotted line*), $\dot{s}_{k+1}(1)$ is shown against $\dot{s}_k(1))$

Since this control law is defined to obtain convergence toward the cyclic motion corresponding to the first control law, the fixed point of the Poincaré maps is the same (see Figs. 6, 10). The minimal and maximal values of the velocity $\dot{s}_k(1)$ such that the step can be achieved are defined numerically. It can be noted that the minimal initial velocity is lower for the second control strategy than for the first one. With $K_{vs} = 2$, at the fixed point the slope is

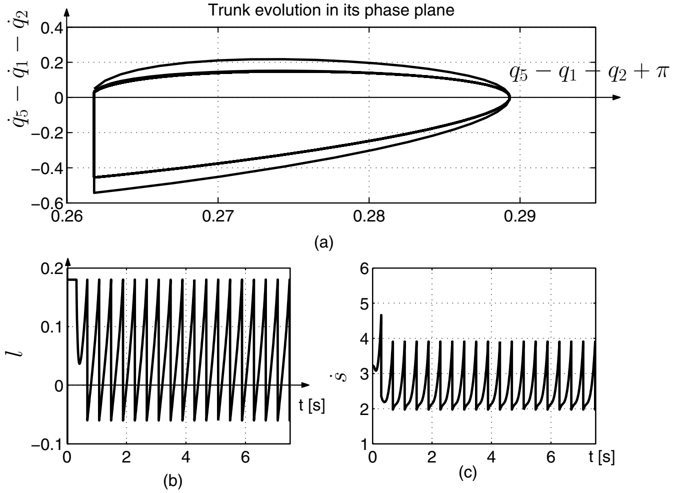


Fig. 11. The convergence toward a cyclic motion is observed in simulation with the second control law, with $K_{vs} = 20$, without modeling error. (**a**) The trunk evolution is drawn in its phase phase (the absolute trunk velocity with respect to the absolute trunk orientation), it tends toward a limit cycle. (**b**) The *horizontal* position of the CoP with respect to time $l(t)$ is bounded. It tends toward the same cyclic evolution as in Fig. 7(b). (**c**) $\dot{s}(s)$ tends toward to the same cyclic motion as in Fig. 7(c)

about $\Delta = 0.23$; it is less than the value obtained for the first control law thus the convergence is faster. For $K_{vs} = 20$ the convergence is so fast that the slope is close to horizontal at the fixed point, in one step the cyclic motion is almost joined. When the initial velocity is far beyond the cyclic one, the constraint on l produces a saturation on $\ddot{s}$ almost all the time, thus almost the same behavior is obtained with $K_{vs} = 2$ or $K_{vs} = 20$.

Assuming no modeling error and initializing the state of the robot out of the periodic orbit (with an initial velocity 60% higher than the cyclic value), the results of one simulation for 20 walking steps of the robot are illustrated in Fig. 11. The convergence toward a cyclic motion can be shown for the trunk via its evolution in its phase plane (Fig. 11-a). In one step the cyclic motion is reached. This convergence is also illustrated via the evolution of the position of the CoP with respect to time (Fig. 11-b). To slow down the motion, for the first step, the position of the CoP stays on the front limit (l_{max}). After the evolution of the CoP corresponds to the desired cyclic one, it is linear from -0.06 m to 0.18 m. Figure 11-c presents the evolution of $\dot{s}$ with respect to time, it clearly converges toward the desired cyclic motion.

6 Control of Walking with Imprecise Model Data

In practice the robot parameters are not perfectly known. We assume that we have some errors on the masses and consequently on the inertia moments of the robot links. We simulate the following case of errors:

- the mass errors are: +10% for the thighs, +30% for the shanks and +50% for the trunk. The error on the inertia moment of the trunk is +30%;
- since the reference path is designed with a false model, the velocity after the impact is not the expected one;
- as the position l of the CoP is calculated via the dynamic model, $l(s)$ will not be exactly $l^d(s)$.

This choice of errors is arbitrary. We choose that the real robot is heavier than the model used in the control law, this point is commented.

6.1 The First Control Law

Initializing the state of the robot in the same conditions as in 5.2; the behavior obtained for 20 walking steps is presented in Fig. 12. Some tracking errors exist particularly at the beginning of each step due to the effect of impact, thus the path followed is not exactly the expected one (but the tracking errors

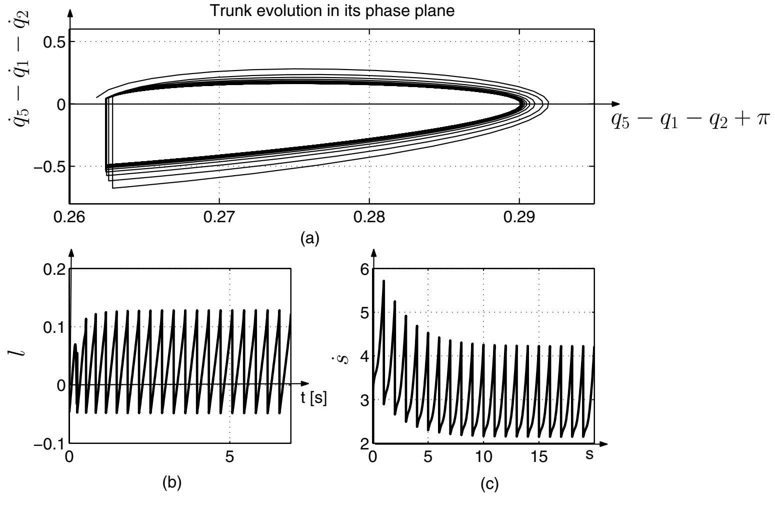


Fig. 12. The convergence toward a cyclic motion is observed in simulation with the proposed control law, with modeling error. (**a**) The trunk evolution is drawn in its phase phase (the absolute trunk velocity with respect to the absolute trunk orientation), it tends toward a limit cycle. (**b**) The *horizontal* position of the CoP with respect to time $l(t)$ tends toward a cyclic evolution different from Fig. 11(b). (**c**) $\dot{s}(s)$ tends toward a cyclic motion different from the motion in Fig. 7(c)

are cyclic). The convergence toward a cyclic motion is shown for the trunk evolution via its phase plane in Fig. 12-a. This convergence is also illustrated via the evolution of $\dot{s}$ with respect to s in Fig. 12-c, it clearly converges toward a stable cyclic motion. The cyclic motion is close to the expected one but not exactly the same, because it is the result of the motion of the CoP and of the dynamic model. Since the real robot is heavier than the robot's model used, we have greater ground reaction forces; consequently the real evolution l of the CoP in Fig. 12-b varies between extreme values smaller in absolute value than the desired values. The difference between $l(s)$ and $l^d(s)$ is higher for large value of $\dot{s}$. In this case there is no problem because constraints of equilibrium of the supporting foot are always satisfied. Otherwise if the real robot was lighter than the modeled one, the CoP could be outside the sole and the constraints of equilibrium of the supporting foot could be violated. So a security margin is necessary when the minimum and the maximum values for the CoP evolution are defined. The best way is to define l_{min} and l_{max} with some margins with respect to real size of the foot (see Fig. 2).

6.2 The Second Control Law

In order to illustrate some robustness properties of the second control law proposed in (Sect. 4), we test the same modeling error as in Sect. 6.

- Since the reference path is designed with a false model, the velocity after the impact is not the expected one;
- In the case of perfect modeling the control law (31) assumes that the limits on $\ddot{s}$ corresponds to $l_{min} < l < l_{max}$. But this relation is based on the dynamic model, since the dynamic model is not perfectly known, this transformation will induce some errors.

A simulation of 20 walking steps is presented in Fig. 13. The biped state is initialized out of the periodic orbit (with an initial velocity 60% higher than the cyclic value). The convergence toward a cyclic motion can be shown via the trunk evolution in its phase plane in Fig. 13-a; some errors can be observed at the impact times. The convergence toward the cyclic motion can be also shown in Fig. 13-b via the evolution of the CoP with respect to time. The evolution of the CoP is not the expected one even if the evolution of $\dot{s}$ converges clearly toward the expected cyclic motion with the end of the second step (Fig. 13-c and 12-c).

In the presence of modeling errors, the two control laws will not give the same cyclic behavior. Due to the second control law, $\dot{s}$ will converge toward $\dot{s}_c$, and the average velocity of the robot does not change, which is not the case for the first control law.

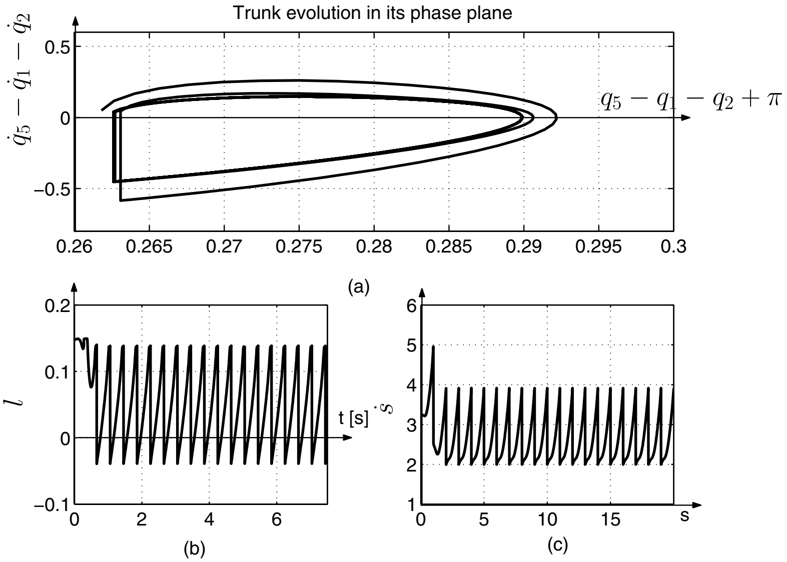


Fig. 13. The convergence toward a cyclic motion is observed in simulation with the second control law, with K_{vs}=20, with modeling errors. (**a**) The trunk evolution is drawn in its phase phase (the absolute trunk velocity with respect to the absolute trunk orientation), it tends toward a limit cycle. (**b**) The horizontal position of the CoP with respect to time $l(t)$ is bounded. It tends toward a cyclic evolution different from Fig. 11(b). (**c**) $\dot{s}(s)$ tends toward to the same cyclic motion as in Fig. 7(c)

7 Discussion Section

Even if the stability studies for the two proposed control laws are conducted numerically for the examples, based on the analytical study of the robot without feet [4] and on the case of a desired piecewise evolution of the position of the center of pressure [7], and also based on numerous simulations, some general conclusions can be given:

- For the first control law, the choice of $l^d(s)$ has a large effect on the existence of a cyclic motion and on the average velocity of the robot. If the position of the CoP is moved forward, the average velocity of the cyclic motion is slowed down. This property is limited: if the position of the CoP is too much forward, no cyclic motion exists. In order that the robot walks faster, a simple solution is to move the desired evolution of the CoP backward.
- The stability property of the first control strategy is essentially due to the fact that the vertical velocity of the center of mass is directed downward just before the impact [4], [7]. Thus this property depends essentially on the choice of $q^d(s)$.

- A larger variation of $l^d(s)$ during one step has two effects, the basin of attraction of the control law is slightly increased, the convergence rate is slightly decreased.
- For the first control law, the control speed $\dot{s}$ is not directly controlled, as shown in Fig. 5, but only stabilizes step by step. The impact phase has a key role in this stabilization. For the example, if the single support phase can be achieved, $\dot{s}$ increases non linearly during the single support and decreases linearly during the impact phase, thus a stable speed is reached as in a passive walking.
- When the desired joint references and the desired position of the center of pressure are defined, since they are not function of time, we do not have to worry about the dynamic consistency. The joints reference need only to be twice differentiable and to satisfy the start and stop conditions corresponding to the impact model. The second derivative of s is calculated to satisfy the dynamic consistency.
- In the development of the control, a finite time controller is defined in eq. (18), to insure a fast convergence to the zero dynamic manifold. Such a controller is not required for the simulation and experiments. The dynamic model is used to calculate the position of the CoP and the admissible limits, for the experiments because it implies that the dynamic model must be "correctly" known. The robustness tests (Sect. 6) have demonstrated that an acceptable behavior can be obtained in the presence of an imprecise model.
- For the second control law, an arbitrary function $\dot{s}_c(s)$ can be chosen even if $\frac{\dot{s}_c(1)}{\dot{s}_c(0)} \neq \alpha$. If this function $\dot{s}_c(s)$ is not consistent with the constraint on the dynamic model ($l_{min} < l(s) < l_{max}$), the closed loop system will converge to an evolution "close" to $\dot{s}_c(s)$ but consistent with the constraint on the dynamic model. This can be used to choose faster or slower motion. For the proposed example, if we choose $\dot{s}_c = 1$, we obtain a cyclic motion with an average velocity of $0.51\,\mathrm{ms}^{-1}$, the CoP position is in the forward part of the feet and often on the toe limit. If we choose $\dot{s}_c = 4$ we obtain a cyclic motion with an average velocity of $1.5\,\mathrm{ms}^{-1}$, the CoP position is often in the limit of the foot.
- The proposed control laws can be extended to walking including rotation of the foot about the toe [5].
- We hope that the second control strategy can be directly used for robot walking in 3D, even if the position of the CoP is limited in the sagittal and frontal plane.

8 Conclusion

For a planar biped, the proposed control strategy consists in the tracking of a reference path instead of a reference motion for the joints and for the position of the CoP. The biped adapts its temporal evolution according to the dynamic constraint that relies the position of the CoP and the joint acceleration. In this context a complete study has been presented.

The conditions of stability are inequalities. Thus a certain robustness is naturally contained in the proposed control strategy. In spite of tracking errors and/or modeling errors, the behavior of the biped converges to a cyclic motion. In the presence of modeling errors, the obtained cycle is slightly modified with respect to the predicted cycle, but stable walking is obtained as it has been observed in simulation.

Two control strategies have been proposed. In the first case, the CoP is constrained to be a function of the robot configuration and the geometric evolution of the joints are controlled, but the temporal evolution is free; the natural convergence toward a cyclic motion is used. In the second case, the convergence to the cyclic motion is forced by using the CoP as a control input to correct for errors in the configuration speed, $\dot{s}$, and the limits on the CoP position are used $l_{min} < l < l_{max}$.

References

[1] S. P. Bhat and D. S. Bernstein. Continuous finite-time stabilization of the translational and rotational double integrators. *IEEE Transaction on Automatic Control*, 43(5):678–682, 1998.

[2] C. Chevallereau, G. Abba, Y. Aoustin, F. Plestan, E. R. Westervelt, C. Canudas-de Witt, and J. W. Grizzle. Rabbit: A testbed for advanced control theory. *IEEE Control System Magazine*, 23(5):57–78, 2003.

[3] C. Chevallereau and A. Lounis. On line reference trajectory adaptation for the control of a planar biped. In *5th international Conference on Climbing and Walking Robots CLAWAR*, pp. 427–435, Paris, France, 2002.

[4] C. Chevallereau, Formal'sky A. M., and D. Djoudi. Tracking of a joint path for the walking of an under actuated biped. *Robotica*, 22(1):15–28, 2004.

[5] Jun Ho Choi and J.W. Grizzle. Planar bipedal walking with foot rotation. In *paper submitted to ACC*, 2005.

[6] O. Dahl and L. Nielsen. Torque-limited path following by online trajectory time scaling. *IEEE Trans. on Automat. Contr*, 6(5):554–561, 1990.

[7] D. Djoudi and C. Chevallereau. Stability analysis of a walk of a biped with control of the zmp. In *IROS'05*, 2005.

[8] D. Djoudi, C. Chevallereau, and Y. Aoustin. Optimal reference motions for walking of a biped robot. In *ICRA 05*, 2005.

[9] J. W. Grizzle, G. Abba, and F. Plestan. Asymptotically stable walking for biped robots: analysis via systems with impulse effects. *IEEE Trans. on Automat. Contr.*, 46:51–64, 2001.

[10] J. Guckenheimer and P. Holmes. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields.* Springer-Verlag, 1985.

[11] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka. The development of honda humanoï¿$\frac{1}{2}$ robot. In *Proc. of the IEEE International Conference on Robotics and Automation*, pp. 1321–1326, Leuven, Belgium, 1998.

[12] F. Pfeiffer, K. Loffler, and M. Gienger. The concept of jogging johnie. In *IEEE Int. Conf. on Robotics and Automation*, 2002.

[13] M. Vukobratovic and B. Borovac. Zero moment point -thirty five years of its live. *Int. Journal of Humanoid Robotics*, 1(1):157–173, 2004.

[14] M. Vukobratovic, B. Borovac, D. Surla, and D. Stokic. *Biped Locomotion.* Springer-Verlag, 349p, 1990.

[15] E.R. Westervelt, J.W. Grizzle, and D.E. Koditschek. Hybrid zero dynamics of planar biped walkers. *IEEE Transaction on Automatic Control*, 48(1):42–56, 2003.

[16] P.-B. Wieber and C. Chevallereau. Online adaptation of reference trajectories for the control of walking systems. *Robotics and Autonomous Systems*, 54:559–566, 2006.

Multi-Locomotion Control of Biped Locomotion and Brachiation Robot

T. Fukuda[1], M. Doi[1], Y. Hasegawa[2], and H. Kajima[1]

[1] Department of Micro-Nano Systems Engineering, Nagoya University
Furo-cho 1, Chikusa-ku, Nagoya, 464-8603, JAPAN
{fukuda, doi}@mein.nagoya-u.ac.jp

[2] Department of Intelligent Interaction Technologies, University of Tsukuba
Tennodai 1-1-1, Tsukuba, Ibaraki, 305-8573, JAPAN
hase@esys.tsukuba.ac.jp

Summary. This paper first introduces a multi-locomotion robot with high mobility and then proposes Passive Dynamic Autonomous Control (PDAC) for the comprehensive control method of multiple types of locomotion. PDAC is the method to take advantage of the robot inherent dynamics and to realize natural dynamic motion. We apply PDAC to a biped walk control. On the assumption that the sagittal and lateral motion can be separated and controlled individually, each motion is designed based on the given desired step-length and period. In order to stabilize walking, the landing position control according to the status is designed. In addition, a coupling method between these motions, which makes the period of each motion identical, is proposed. Finally, we show that the multi-locomotion robot realizes the 3-dimensional dynamic walking using the PDAC control.

1 Introduction

In recent years there have been many successful researches that focus on dynamic and skillful motions inspired by animal dexterity [13, 15, 16, 21, 27]. However, in general, they were mainly focused on a single type of motion, such as biped or quadruped locomotion. On the other hand, many animals, such as primates, use a primary form of locomotion but switch to other types depending on their surroundings, situation and purpose. For instance, a gorilla has high mobility in a forest by selecting a bipedal walking in a narrow space, a quadrupedal walking on rough terrain and a brachiation in the forest canopy. Inspired by high mobility of an animal, we have developed a anthropoid-like "Multi-locomotion robot" that can perform several types of locomotion and choose the proper one on an as-need basis (Fig. 1) [12]. A development of a bio-inspired robot which has multiple types of locomotion for high mobility is challenging, because other problems arise in addition to research topics on humanoid robot study. One is a comprehensive control architecture that is

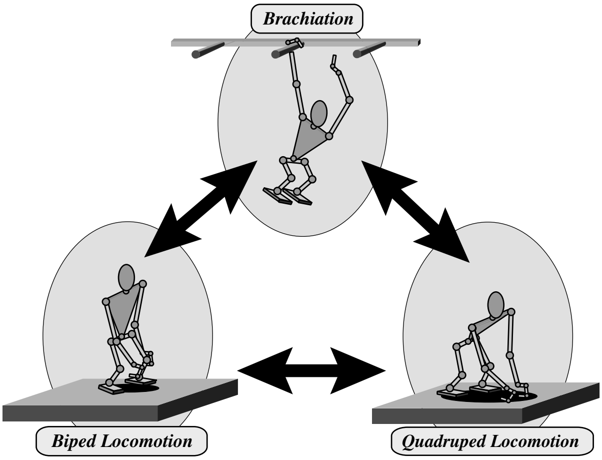


Fig. 1. Concept of the multi-locomotion robot

capable to achieve multiple types of locomotion. A common control architecture should be designed when the robot achieves a seamless transient motion connecting one locomotion to another such as a transient from trot to gallop, because a transient motion between typical types of locomotion can not be realized by fusing control signals from the corresponding controllers. Based on this notion, we have proposed a novel method named Passive Dynamic Autonomous Control (PDAC) [2, 3] that achieves not only a bipedal walk but also a quadrupedal walk. This paper focuses on the PDAC algorithm and control method for the bipedal walk.

A lot of research of ZMP-based control [31] has been presented [19, 28]. However, ZMP-based control could not realize an efficient locomotion since it does not take advantage of the robot inherent dynamics. To solve this problem, it is necessary to develop a dynamics-based method. Some researchers proposed a method to use the robot dynamics directly by making the point-contact between a robot and the ground [7, 14, 21, 33]. Miura and Shimoyama [17] presented a stilt-like biped and control method to stabilize the gaits by changing the robot posture at foot-contact. Kajita et al. [13] proposed a control method based on the conserved quantity introduced due to a horizontal COG (Center Of Gravity) trajectory. Goswami et al. [6, 29] reported a method to realize quasi-passive walking on the horizontal ground. Grizzle and Westervelt et al. [8, 20, 32] proposed a control method of an underactuated planar robot with a trunk and proved its stability. Although some of these point-contact methods actually realized smooth dynamic walking, their walking was 2-dimensional or that of a robot without trunk. Thus, the main goal of a biped walk is to propose the new control method based on point-contact and realize 3-dimensional dynamic walking of a multiple link robot with a trunk.

In this paper, we introduce the novel control method named Passive Dynamic Autonomous Control (PDAC). PDAC assumes the two following premises: 1) point-contact 2) interlocking. The second premise means that the angles of the robot joints are connected with the angle around contact point. Although this concept was proposed first by Grizzle et al. [8], we propose another new method to control the robot dynamics by means of PDAC. The approach of PDAC is to describe the robot dynamics as a 1-DOF autonomous system, which is the dynamics around the contact point. This approach makes it possible to calculate the period from foot-contact to next foot-contact (we term this foot-contact period hereinafter), hence the foot-contact period of the lateral motion and that of the sagittal one can be made identical. Each motion is designed by means of PDAC based on the assumption that the sagittal and lateral motions can be separated. After that, by keeping the conservative quantity of the autonomous system, the walking motion is stabilized. In addition, we propose a coupling method of each motion to make each foot-contact period identical. Finally, by means of the proposed method, 3-dimensional natural dynamic walking based on the robot inherent dynamics is achieved.

In the following section, the multi-locomotion robot is introduced and then PDAC is explained in detail in Sect. 3. The 3-dimensional walking is designed by means of PDAC in Sect. 4. Section 5 describes the experimental results. Finally, Sect. 6 is conclusion.

2 Multi–Locomotion Robot

The dimensions of the multi-locomotion robot we developed is designed based on those of a gorilla, and therefore the robot is called "Gorilla Robot III". Figure 2 shows the overview of Gorilla Robot III and its link structure. This robot is about 1.0 [m] tall, weighs about 22.0 [kg], and consists of 25 links and 26 AC motors including two grippers. The real-time operating system VxWorks (Wind River Systems Inc) runs on a Pentium III PC for processing sensory data and generating its behaviors. The rate gyroscope and force sensor attached at each wrist measures the angular velocity around the grasping bar to calculate the pendulum angle during the brachiation, and reaction forces from grasping bars in order to judge whether the robot successfully grasps the bar or not, respectively. Some photo sensors are attached on the sole in order to perceive foot-contact.

This robot has been designed to perform biped locomotion, quadruped locomotion and brachiation. We also consider the intermediate motion between a bipedal and quadrupedal walk in order to realize seamless transfer from a bipedal walk to a quadrupedal walk and from a quadrupedal walk to a bipedal walk without pause. As the first step, we designed the controller for both locomotion using the same algorithm "PDAC". The snapshot of the quadrupedal walk is shown in Fig. 3, and a bipedal walk is shown in Sect. 5. Brachiation is an interesting form of locomotion performed by long-armed apes by using

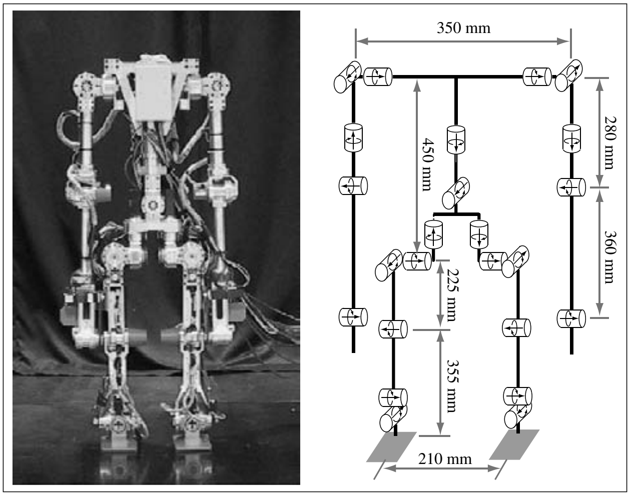


Fig. 2. Gorilla Robot III

their arms to swing from branch to branch. This motion is a dynamic and dexterous action routinely performed by some kinds of apes [26, 30]. Fukuda et al. developed a six-link brachiation robot [4] as a pioneering research analyzing dynamics of brachiation, “Brachiator II” [5, 9, 18, 22, 23] that is 2-link underactuated system like “Acrobot” [1, 27], and “Brachiator III” [10, 24] that achieves three-dimensional brachiation with redundant mechanisms.

Based on these studies, we designed over-hand and side-hand motions of “Gorilla Robot II” [12], using a motion learning algorithm, and “Gorilla Robot III” achieves a continuous brachiation shown in Fig. 4 by implicitly using the PDAC method in locomotion action control.

3 Passive Dynamic Autonomous Control

3.1 Target Dynamics

The concept of PDAC is the same as what Grizzle et al. [8] has proposed. We begin with the two following premises. First, the contact state between a robot and the ground is point-contact. Second, robot joints are interlocked with the angle around the contact point. The first premise means that the first joint of a robot, i.e. the ankle joint of the stance leg, is passive. The second means that the angles of active joints are described as a function of the angle around the contact point. Assuming that PDAC is applied to the serial n-link rigid robot shown in Fig. 5, these two premises are expressed as follows:

$$\tau_1 = 0 \tag{1}$$

$$\Theta = [\theta_1, \theta_2, \cdots, \theta_n]^T = [f_1(\theta), f_2(\theta), \cdots, f_n(\theta)]^T = f(\theta) \ , \tag{2}$$

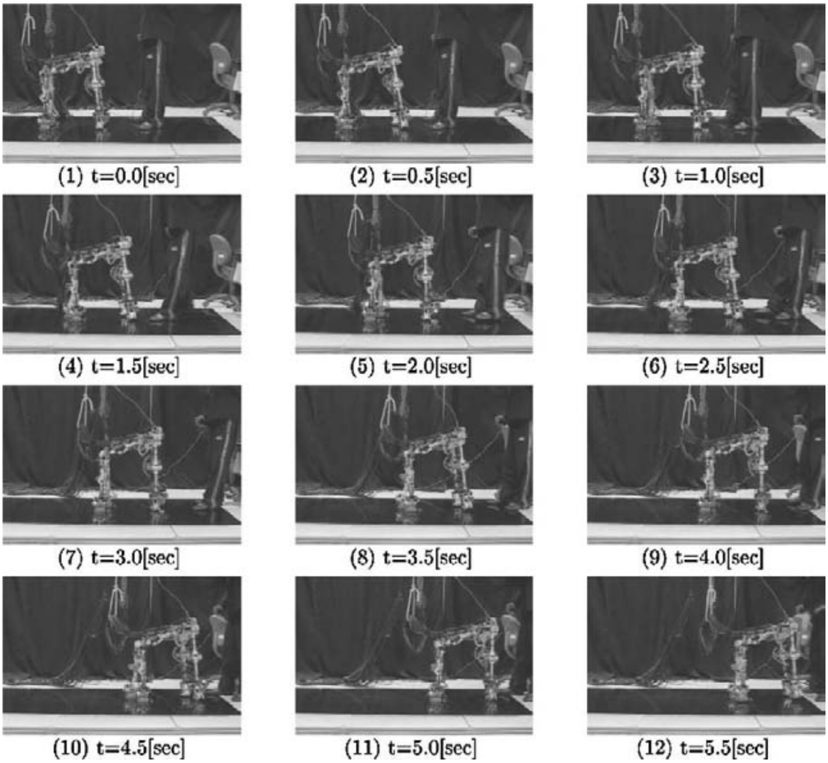

(1) t=0.0[sec]
(2) t=0.5[sec]
(3) t=1.0[sec]
(4) t=1.5[sec]
(5) t=2.0[sec]
(6) t=2.5[sec]
(7) t=3.0[sec]
(8) t=3.5[sec]
(9) t=4.0[sec]
(10) t=4.5[sec]
(11) t=5.0[sec]
(12) t=5.5[sec]

Fig. 3. Snapshots of the quadrupedal walking of PDAC. The step length is about 0.09 [m] and velocity is about 0.176 [m/s]

where θ is the angle around the contact point in the absolute coodinate system. Since it has no effect on the robot dynamics due to point-contact, level ground is assumed, therefore $\theta_1 = f_1(\theta) = \theta$.

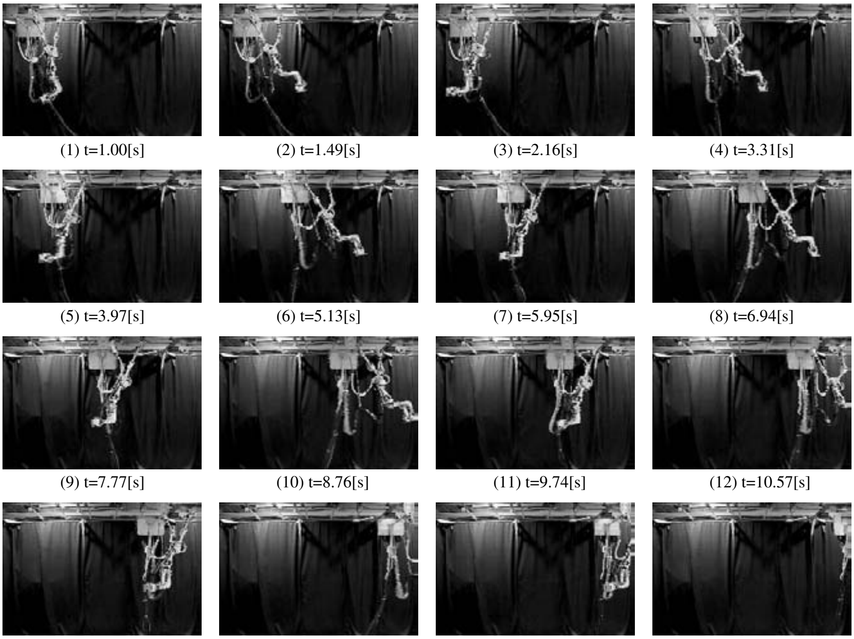
The dynamic equations of this model are given by

$$\boldsymbol{\frac{d}{dt}\left(M(\Theta)\dot{\Theta}\right) - \frac{1}{2}\frac{\partial}{\partial \Theta}\left(\dot{\Theta}^T M(\Theta)\dot{\Theta}\right) - G(\Theta) = \tau} \ , \tag{3}$$

where $M(\Theta) = [M_1(\Theta), M_2(\Theta), \cdots, M_n(\Theta)]^T$, $\Theta = [\theta_1, \theta_2, \cdots, \theta_n]^T$, $G(\Theta) = [G_1(\Theta), G_2(\Theta), \cdots, G_n(\Theta)]^T$, $\tau = [\tau_1, \tau_2, \cdots, \tau_n]^T$, $\frac{\partial}{\partial \Theta} = [\frac{\partial}{\partial \theta_1}, \frac{\partial}{\partial \theta_2}, \cdots, \frac{\partial}{\partial \theta_n}]^T$. Since in this model the dynamic equation around the contact point has no term of the Coriolis force, it is given as

$$\boldsymbol{\frac{d}{dt}\left(M_1(\Theta)\dot{\Theta}\right) - G_1(\Theta) = \tau_1} \ . \tag{4}$$

By differentiating Eq. (2) with respect to time, the following equation is acquired,



(1) t=1.00[s] (2) t=1.49[s] (3) t=2.16[s] (4) t=3.31[s]

(5) t=3.97[s] (6) t=5.13[s] (7) t=5.95[s] (8) t=6.94[s]

(9) t=7.77[s] (10) t=8.76[s] (11) t=9.74[s] (12) t=10.57[s]

(13) t=11.40[s] (14) t=12.39[s] (15) t=13.21[s] (16) t=14.70[s]

Fig. 4. Snapshots of continuous brachiation. All bars are set at regular intervals of 0.4 m and at the same height of 2.7 m

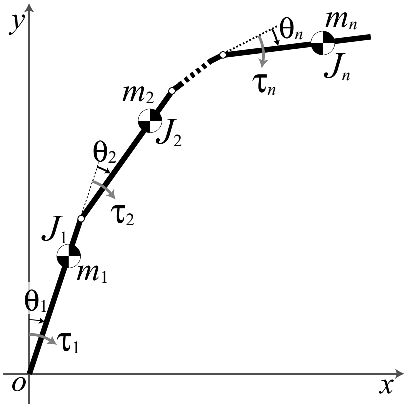


Fig. 5. Mechanical model of the serial n-link rigid robot. θ_i and τ_i are the angle and the torque of the ith joint respectively. m_i and J_i are the mass and the moment of inertia of the ith link respectively

$$\dot{\Theta} = \frac{\partial f(\theta)}{\partial \theta}\dot{\theta} = \left[\frac{\partial f_1(\theta)}{\partial \theta}, \frac{\partial f_2(\theta)}{\partial \theta}, \cdots, \frac{\partial f_n(\theta)}{\partial \theta}\right]^T \dot{\theta} \, . \tag{5}$$

Substituting Eqs. (1), (2) and (5) into Eq. (3) yields the following dynamic equation,

$$\frac{d}{dt}\Big(M(\theta)\dot{\theta}\Big) = G(\theta) \, , \tag{6}$$

where

$$M(\theta) := M_1\Big(f(\theta)\Big)\frac{df(\theta)}{d\theta} \tag{7}$$

$$G(\theta) := G_1\Big(f(\theta)\Big) \, . \tag{8}$$

By multiplying both sides of Eq. (6) by $M(\theta)\dot{\theta}$ and integrating with respect to time, the dynamics around the contact point is obtained as follows:

$$\int \Big(M(\theta)\dot{\theta}\Big)\frac{d}{dt}\Big(M(\theta)\dot{\theta}\Big)\, dt = \int M(\theta)G(\theta)\dot{\theta}\; dt \tag{9}$$

$$\Longleftrightarrow \dot{\theta} = \frac{1}{M(\theta)}\sqrt{\int 2G(\theta)M(\theta)\; d\theta} \, . \tag{10}$$

Assuming that the integration in right side of Eq. (10) is calculated as $\int G(\theta)M(\theta)\; d\theta = D(\theta) + C$, Eq. (10) is described as the following 1-DOF autonomous system,

$$\begin{aligned} \dot{\theta} &= \frac{1}{M(\theta)}\sqrt{2\big(D(\theta)+C\big)} && (11) \\ &:= F(\theta) \, . && (12) \end{aligned}$$

In this paper, we term Eqs. (11) and (12) the target dynamics.

3.2 Dynamics Interlocking

As mentioned previously, PDAC is based on the two premises: passivity and interlocking. These premises make it possible to describe the whole robot dynamics as a 1-DOF autonomous system, owing to which the simple and valid controller based on the robot dynamics can be composed. However, interlocking of joint angles has the possibility to create a problem that the robot vibrates and the controller loses its stability during locomotion, especially at foot-contact, since if the passive joint vibrates, all of other active joints also do. In order to solve this problem, all of the robot joints are controlled according to the desired dynamics of each joint derived from the interlocking function Eq. (2) and the target dynamics Eq. (12) as follows:

$$\dot{\theta}_i = \frac{\partial f_i}{\partial \theta}F\big(f_i^{-1}(\theta_i)\big) \qquad (i = 1, 2, 3, \cdots) \, . \tag{13}$$

These desired dynamics are independent from each other, thus it is necessary to connect the desired dynamics of the active joints with the target dynamics in order to prevent the whole walking motion being broken in case of error between the target dynamics and the actual dynamics of θ. Hence, we define the connection between the target dynamics and the active joints. The controller decides the desired angular velocities of each joint as described below,

$$\dot{\theta}_1^d = F\big(f_1^{-1}(\theta_1)\big) \tag{14}$$

$$\dot{\theta}_i^d = \frac{\partial f_i}{\partial \theta} F\big(f_i^{-1}(\theta_i)\big) + k_i\big(f_i(\theta) - \theta_i\big) \qquad (i = 2, 3, \cdots) \tag{15}$$

$$\Longleftrightarrow \dot{\Theta}^d := F(\Theta) \; , \tag{16}$$

where k_i is the strength of connection determined experimentally since its value has little effect on the robot dynamics. As for humanoid robots, the ground slope at the contact point is deduced from the angle of the ankle joint of the swing leg at foot-contact, and θ is calculated from θ_1 and the ground slope. The remarkable point is that if there is no error such as model error or disturbance, the second term of Eq. (15) is constantly zero and the actual dynamics of θ is identical with the target dynamics.

Figure 6 shows the block diagram of PDAC of bipedal locomotion. The control loop including a robot (enclosed by the dotted line in Fig. 6) has no input, thus it can be considered that the control system is autonomous. This autonomy makes it possible to realize natural dynamic motion based on the inherent dynamics of a robot. The loop described by the broken line is executed only at the moment of foot-contact. In this loop, the target dynamics of the next step is determined according to both the desired parameters such as walking velocity and the robot status, then F is updated. Since this updating compensates the error between the previous target dynamics and the actual ones around the contact point, it is possible to realize stable walking.

3.3 PDAC Constant

Since as mentioned previously, the target dynamics is autonomous, in addition, independent of time, it is considered as a kind of conservative system. Therefore, it is conceivable that the target dynamics has a conserved quantity. As for PDAC, it is the constant of integration in right side of Eq. (10). That is, C in Eq. (11) is the conserved quantity of the target dynamics, which is named PDAC Constant. It is clear that PDAC Constant is decided in accordance with initial condition and that the robot motion is generated as it is kept constant. In order to stabilize walking, the controller updates the target dynamics according to PDAC Constant. This method to update is presented later.

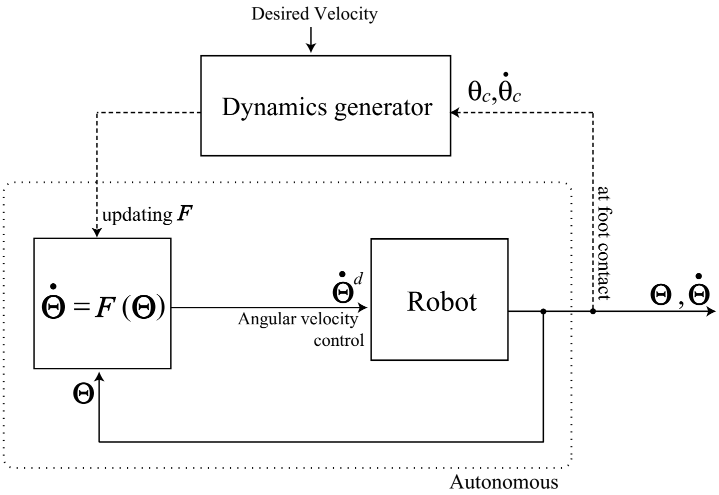


Fig. 6. Block diagram of PDAC of bipedal locomotion. θ_c and $\dot{\theta}_c$ are the angle and the angular velocity of θ_1 at foot-contact respectively

4 Bipedal Walking Control

In this paper, it is assumed that lateral motion and sagittal one can be separated and controlled independently since lateral side-to-side rocking motion is quite small and step-length in the sagittal plane is relatively short. Although both motions are composed independently, the period from foot-contact to next foot-contact (foot-contact period) in both planes are necessarily identical. We design each motion by means of PDAC by giving both the desired step-length, λ^d, and desired foot-contact period, T^d, and propose a coupling method of both motions. In addition, the landing position control is designed based on PDAC. At first the sagittal motion control is presented that is followed by the lateral motion control satisfying the condition of the foot-contact period is explained.

4.1 Sagittal Motion Control

3-Link Model

For the sake of simplicity, the 3-link model as shown in Fig. 7 is used, i.e. the upper body of the robot is not moved. The dynamic equation of this model is described as Eq. (3) and that of the ankle joint of the stance leg is Eq. (4) where $n = 3$. The left side of Eq. (4) is described as follows:

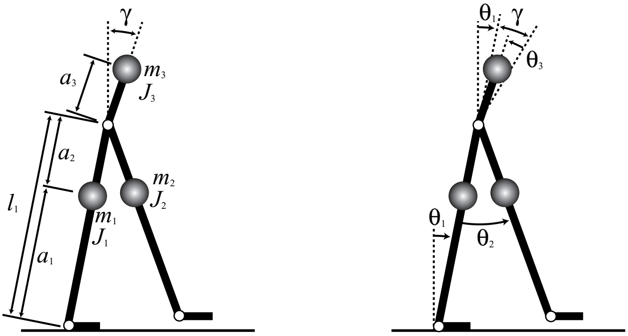


Fig. 7. 3-link model in the sagittal plane. m_i, J_i, l_i and a_i are the mass, the moment of inertia, the length of link and the distance from the joint to the link COG of link i respectively. γ is the angle of forward tilting. In the *right figure*, θ_1, θ_2 and θ_3 are the ankle angle of the stance leg, the angle from the stance leg to the the swing leg, the angle to swing the trunk up respectively

$$\begin{aligned} M_{11}(\Theta) = {} & J_1 + J_2 + J_3 + m_1 a_1^2 + m_2 l_1^2 + m_2 a_2^2 - 2 m_2 a_2 l_1 \cos\theta_2 \\ & + m_3 l_1^2 + m_3 a_3^2 + 2 m_3 a_3 l_1 \cos(\gamma - \theta_3) \end{aligned} \tag{17}$$

$$M_{12}(\Theta) = -J_2 - m_2 a_2^2 + m_2 a_2 l_1 \cos\theta_2 \tag{18}$$

$$M_{13}(\Theta) = -J_3 - m_3 a_3^2 - m_3 a_3 l_1 \cos(\gamma - \theta_3) \tag{19}$$

$$\begin{aligned} G_1(\Theta) = {} & (m_1 a_1 + m_2 l_1 + m_3 l_1) g \sin\theta_1 + m_2 g a_2 \sin(\theta_2 - \theta_1) \\ & + m_3 g a_3 \sin(\theta_1 + \gamma - \theta_3) \ , \end{aligned} \tag{20}$$

where $M_1(\Theta) = [M_{11}(\Theta), M_{12}(\Theta), M_{13}(\Theta)]$.

Interlocking of Sagittal Joints

Grizzle et al. [8] used the following interlocking in their previous paper to maintain the angle of the torso at some constant value and to command the swing leg to behave as the mirror image of the stance leg. In this paper, we use the same interlocking, that is,

$$\theta_1 = f_1(\theta) = \theta - \beta \tag{21}$$

$$\theta_2 = f_2(\theta) = 2\theta \tag{22}$$

$$\theta_3 = f_3(\theta) = \theta \ , \tag{23}$$

where β is the ground slope at the contact point (ascent is positive). From Eqs. (21)–(23) and (1), Eq. (6) is

$$M_s(\theta) = (J_1 - J_2 + m_1 a_1^2 + m_2 l_1^2 - m_2 a_2^2 + m_3 l_1^2) + m_3 a_3 l_1 \cos(\gamma - \theta) \tag{24}$$

$$:= E_1 + E_2 \cos(\gamma - \theta) \tag{25}$$

$$G_s(\theta) = (m_1 a_1 + m_2 l_1 + m_2 a_2 + m_3 l_1) g \sin\theta + m_3 g a_3 \sin\gamma \tag{26}$$

$$:= E_3 + E_4 \sin\theta \ . \tag{27}$$

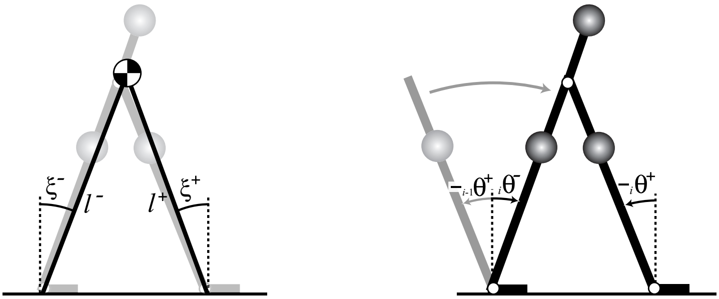


Fig. 8. Parameters at foot-contact. l^- and ξ^- are the length and inclination of the inverted pendulum which connects the supporting foot and the COG of the whole robot before impact, while l^+ and ξ^+ are those after impact. ${}_i\theta^-$ and ${}_i\theta^+$ are the angles around the contact point before and after impact of the ith step

Thus,

$$\int M_s(\theta)G_s(\theta)d\theta = \int \big(E_1 + E_2\cos(\gamma - \theta)\big)\big(E_3 + E_4\sin\theta\big)d\theta \tag{28}$$

$$= E_2E_4\left(\frac{\sin(\gamma\theta)}{2} - \frac{\cos(2\theta - \gamma)}{4}\right)$$
$$+E_2E_3\sin(\theta - \gamma) - E_1E_4\cos\theta + E_1E_3\theta + C_s \tag{29}$$

$$:= D_s(\theta) + C_s \tag{30}$$

where C_s is the integral constant, which is PDAC Constant of the sagittal motion. From Eq. (11), the target dynamics in the sagittal plane is

$$\dot{\theta} = \frac{1}{M_s(\theta)}\sqrt{2\big(D_s(\theta) + C_s\big)} \tag{31}$$

$$:= F_s(\theta)\ . \tag{32}$$

From Eqs. (21)–(23), $f_1^{-1}(\theta_1) = \theta_1 + \beta$, $f_2^{-1}(\theta_2) = \frac{1}{2}\theta_2$, $f_3^{-1}(\theta_3) = \theta_3$ are obtained, thus the desired angular velocities of sagittal joints are described as follows:

$$\dot{\theta}_1^d = F_s(\theta_1 + \beta) \tag{33}$$

$$\dot{\theta}_2^d = 2F_s\left(\frac{\theta_2}{2}\right) + k_2(2\theta - \theta_2) \tag{34}$$

$$\dot{\theta}_3^d = F_s(\theta_3) + k_3(\theta - \theta_3)\ . \tag{35}$$

Foot Contact Model

In this paper, it is assumed that foot-contact occurs instantaneously and the angular momentum around the contact point is varied instantly. The angular momentum is described as

$$P = M_s(\theta)\dot{\theta} \ . \tag{36}$$

Figure 8 depicts some parameters at foot-contact. Assuming that the translational velocity along the pendulum at foot-contact is zero since it is quite small, the angular velocity around the contact point is acquired as follows:

$$l^+ P^+ = l^- P^- \cos(\xi^- + \xi^+) \tag{37}$$

$$\Longleftrightarrow {}_i\dot{\theta}^+ = \frac{l^- M_s({}_i\theta^-)^-}{l^+ M_s({}_i\theta^+)^+} \cos(\xi^- + \xi^+) \ {}_i\dot{\theta}^- \tag{38}$$

$$\Longleftrightarrow {}_i\dot{\theta}^+ := H_s \, {}_i\dot{\theta}^- \tag{39}$$

From this value, and the PDAC Constant at the ith step, ${}_iC_s$, is obtained as

$${}_iC_s = \frac{1}{2}\Big(M_s\left({}_i\theta^+\right) {}_i\dot{\theta}^+\Big)^2 - D_s\left({}_i\theta^+\right) \ . \tag{40}$$

Desired PDAC Constant

Since the target dynamics is the 1-DOF autonomous system, it is possible to calculate the foot-contact period by integrating Eq. (12) with respect to time. The foot-contact period satisfying the desired step-length is calculated as below:

$$\dot{\theta} = F_s(\theta) \tag{41}$$

$$\Longleftrightarrow \frac{1}{F_s(\theta)} d\theta = dt \tag{42}$$

$$\Longleftrightarrow \int_{{}_i\hat{\theta}^+}^{{}_{i+1}\hat{\theta}^-} \frac{1}{F_s(\theta)} \ d\theta = \hat{T}_s \ , \tag{43}$$

where ${}_i\hat{\theta}^+ = {}_{i+1}\hat{\theta}^- = \sin^{-1} \frac{\lambda^d}{2l_1}$ are the desired value of ${}_i\theta^+$ and ${}_{i+1}\theta^-$ that can be calculated from the desired step-length. This period is necessarily identical with the desired foot-contact period, thus

$$\hat{T}_s = T^d \ . \tag{44}$$

In order to generate the stable cyclic walking, the angular velocity around the contact point after impact must be kept constant, that is,

$${}_i\dot{\theta}^+ = {}_{i+1}\dot{\theta}^+ \ . \tag{45}$$

By solving two conditions, Eqs. (44) and (45), by means of two dimensional approximation of ${}_i\hat{\theta}^+$ and ${}_{i+1}\hat{\theta}^-$, the desired PDAC Constant is determined,

$$C_s^d = C_s^d(T^d, \lambda^d) \ . \tag{46}$$

Stabilization of Sagittal Motion

Since the target dynamics is a conservative system, if the PDAC Constant is kept constant, the stability of the motion is guaranteed. In order to stabilize walking, step-length is varied according to the PDAC Constant. This method takes advantage of the loss of angular momentum at foot-contact, that is, if the step-length is long, the loss is high, while if short, it is low. The control strategy is to adjust the step-length of the next step after every foot-contact in accordance with the desired PDAC Constant and the actual one as follows:

$$\frac{C_s^d + D_s\left({}_{i+1}\theta^+\right)}{M_s\left({}_{i+1}\theta^+\right)^2} = \frac{D_s\left({}_{i+1}\theta^-\right) + {}_iC_s}{M_s\left({}_{i+1}\theta^-\right)^2} H_s^2 \tag{47}$$

$$\Longleftrightarrow {}_{i+1}\theta^- = {}_{i+1}\theta^-\left(C_s^d,\ {}_iC_s\right)\ . \tag{48}$$

Note, however, that since in this paper level ground is assumed, ${}_{i+1}\theta^- = {}_{i+1}\theta^+$.

This stabilizing control makes it possible to keep PDAC Constant in the vicinity of the desired value. Therefore, sagittal motion is kept stable.

Here, the point to notice is that the foot-contact period differs from the desired foot-contact period due to stabilization. Hence, it is necessary to control the lateral motion so that the period of lateral motion is identical with the following period of sagittal motion,

$$T_s = \int_{{}_i\theta^+}^{{}_{i+1}\theta^-} \frac{1}{F_s(\theta)}\, d\theta\ . \tag{49}$$

4.2 Lateral Motion Control

Lateral Motion

Many reserchers investigated and proposed lateral motion control [11, 14, 25]. In this paper, we design the lateral motion by means of PDAC as depicted in Fig. 9. In order to continue the side-to-side rocking motion, a robot lifts its pelvis in phases (A) and (B). The inverted pendulum whose length is variable is used as the model of the lateral plane since the motion to lift the pelvis is quite small, in addition, the robot posture is varied little thus the motion to lift the pelvis can be considered as lengtherning the pendulum. The lateral motion can be continued in spite of the loss of angular momentum at foot-contact by changing the pendulum length at impact.

Collision Inverted Pendulum Model

The following model shown in Fig. 10 is used as the model of the lateral motion: two inverted pendulums which are opposite each other continue to

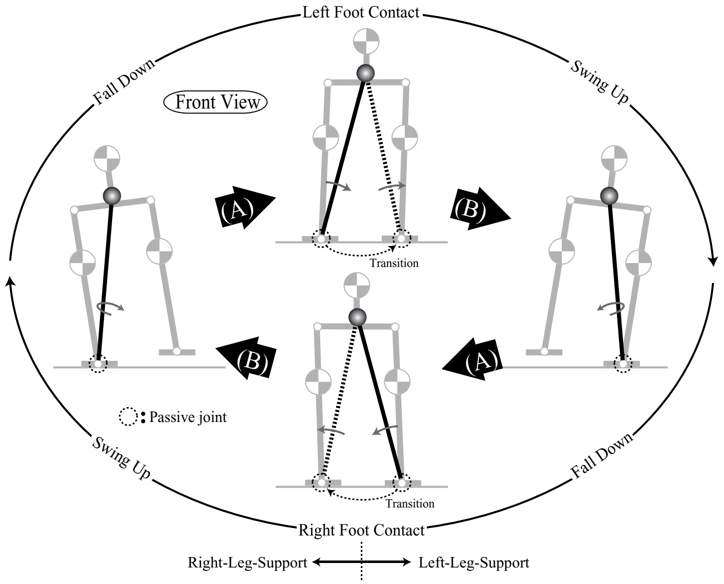


Fig. 9. The lateral motion of lateral-based walk (*front view*). The inverted pendulum falls off in phase(A) and swings up in phase(B)

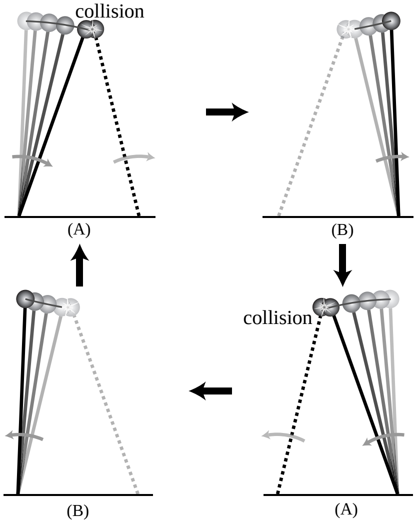


Fig. 10. Motion of CIPM. The collision between the foot and the ground is regarded as that between two pendulums. (**A**) and (**B**) correspond to those in Fig. 9

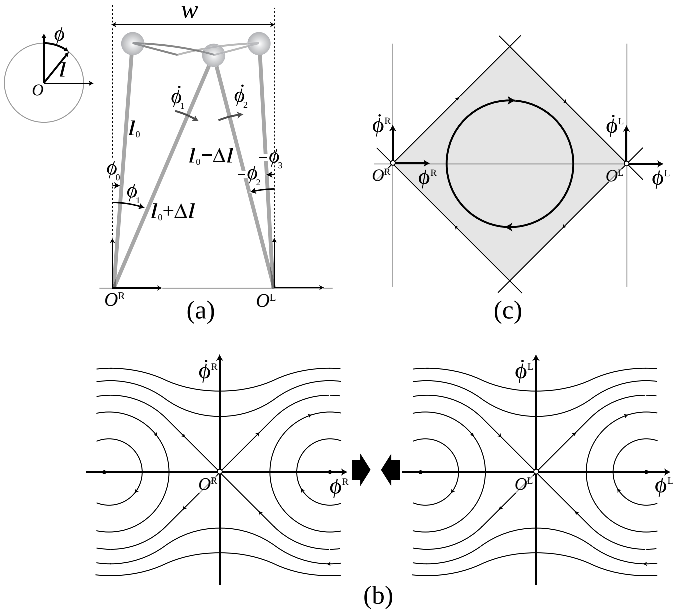


Fig. 11. (**a**) Trajectory of COG and polar coordinate systems Σ^R, Σ^L. l and ϕ denote the length and the angle of an inverted pendulum. (l_0, ϕ_0) and $(l_0 + \Delta l, \phi_1)$ are the coordinates in Σ^R at the beginning and ending of phase (A), $(l_0 - \Delta l, \phi_2)$ and (l_0, ϕ_3) is that of Σ^L of phase (B) respectively. $\dot{\phi}_1$, $\dot{\phi}_2$ denotes the angular velocity at the end of phase (A) and at the beginning of phase (B). (**b**) Phase portraits of ϕ^R and ϕ^L (**c**) Phase portrait of CIPM. The *gray tetragon* surrounded by the pair of separatrixes is named CIP-Area

rock, iterating the collision between them, which is named Collision Inverted Pendulum Model (CIPM). This CIPM is intuitively like the Newton's Pendulum inverted. Figure 11 shows the trajectory of COG and two coordinate systems Σ^R and Σ^L that correspond to the right- and left-leg-support period respectively, and Fig. 11(b) depicts the phase portraits of ϕ^R and ϕ^L. These two phase portrait's coalescing yields the phase portrait of CIPM (see Fig. 11(c)). In the phase portrait of CIPM, there is the area in which one has the circular nature between the coordinate systems Σ^R and Σ^L. In this area, the periodic motion can be realized due to the circular nature.

Interlocking of Lateral Joints

The interlocking in the lateral plane is defined as below,

$$Phase(A) : l = f_A(\phi) = a_1\phi + b_1 \tag{50}$$

$$Phase(B) : l = f_B(\phi) = a_2\phi^2 + b_2\phi + c_2 \ , \tag{51}$$

where

$$a_1 = \frac{1}{\phi_1 - \phi_0} \Delta l \tag{52}$$

$$b_1 = l_0 - \frac{\phi_0}{\phi_1 - \phi_0} \Delta l \tag{53}$$

$$a_2 = \frac{1}{(\phi_3 - \phi_2)^2} \Delta l \tag{54}$$

$$b_2 = -\frac{2\phi_2}{(\phi_3 - \phi_2)^2} \Delta l \tag{55}$$

$$c_2 = l_0 - \frac{\phi_3^2 - 2\phi_2\phi_3}{(\phi_3 - \phi_2)^2} \Delta l \ . \tag{56}$$

l is the monotoneally increasing function of ϕ meeting the following conditions in phase (A): $f_A(\phi_0) = l_0$ and $f_A(\phi_1) = l_0 + \Delta l$, while in phase (B): $f_B(\phi_2) = l_0 - \Delta l$, $f'(\phi_2) = 0$ and $f(\phi_3) = l_0$.

The dynamic equation of the angle of an inverted pendulum is described as follows:

$$\frac{d}{dt}\left(\left(ml^2 + J \right) \dot{\phi} \right) = mgl \sin\phi \ . \tag{57}$$

From the interlocking, Eq. (6) is described as follows:

$$M_{l_N}(\phi) = m f_N(\phi)^2 + J \tag{58}$$

$$G_{l_N}(\phi) = mg f_N(\phi) \sin\phi \ , \tag{59}$$

where the suffix N means phase (N) (N=A, B). From Eq. (11), the target dynamics in the lateral plane is

$$\dot{\phi} = \frac{1}{M_{l_N}(\phi)} \sqrt{\int 2M_{l_N}(\phi) G_{l_N}(\phi) \ d\phi} \tag{60}$$

$$:= \frac{1}{M_{l_N}(\phi)} \sqrt{2\left(D_{l_N}(\phi) + C_{l_N} \right)} \tag{61}$$

$$:= F_{l_N}(\phi) \ , \tag{62}$$

where C_{l_N} is the integral constant, which is PDAC Constant of the lateral dynamics.

Assuming that the collision between the swing leg and the ground is perfectly non-elastic, the angular velocity of the inverted pendulum after impact is

$$\dot{\phi}_2 = \frac{v_1}{l_0 - \Delta l} \cos(\phi_1 - \phi_2 - \zeta) \tag{63}$$

$$= \frac{\sqrt{a_1^2 + (l + \Delta l)^2}}{l_0 - \Delta l} \cos(\phi_1 - \phi_2 - \zeta) \, \dot{\phi}_1 \tag{64}$$

$$:= H_l \, \dot{\phi}_1 \ , \tag{65}$$

where

$$v_1 = \left. \sqrt{(l\dot{\phi})^2 + \dot{l}^2} \right|_{\phi=\phi_1} = \sqrt{a_1^2 + (l + \Delta l)^2}\dot{\phi} \tag{66}$$

$$\zeta = \tan^{-1}\left.\left(\frac{\dot{l}}{l\dot{\phi}}\right)\right|_{\phi=\phi_1} = \tan^{-1}\left(\frac{a_1}{l + \Delta l}\right) . \tag{67}$$

Δl is the control value of the lateral motion. It is calculated from the condition of the beginning of phase(A) and the end of phase(B): $F_{l_A}(\phi_0) = 0$ and $F_{l_B}(\phi_3) = 0$. That is,

$$\frac{D_{l_A}(\phi_1) - D_{l_A}(\phi_0)}{M_{l_A}(\phi_1)^2} H_l^2 = \frac{D_{l_B}(\phi_3) - D_{l_B}(\phi_2)}{M_{l_B}(\phi_2)^2} . \tag{68}$$

Δl is so small that it can be obtained from this equation by means of the linear approximation of Δl,

$$\Delta l = \Delta l(\phi_0, \phi_3) . \tag{69}$$

Finally, it is necessary to determine the desired amplitude of the rocking motion, ϕ_0^d, so that the foot-contact period in the lateral plane matches with the desired foot-contact period. This condition is described as below,

$$\int_{\phi_0^d}^{\phi_1} \frac{1}{F_{l_A}(\phi)} d\phi + \int_{\phi_2}^{\phi_0^d} \frac{1}{F_{l_B}(\phi)} d\phi = T^d . \tag{70}$$

By means of two dimensional approximation of ϕ, it is possible to calculate ϕ_0^d from Eq. (70),

$$\phi_0^d = \phi_0^d(T^d) . \tag{71}$$

By setting ϕ_3 at $-\phi_0^d$ at the beginning of phase (A) of every step, the lateral motion can be stabilized.

Coupling with the Sagittal Motion

As mentioned previously, it is necessary that the foot-contact period of the sagittal motion and that of the lateral motion are made identical. In case of the adjustment of step-length, the sagittal foot-contact period differs from the desired foot-contact period, thus the lateral motion needs to be varied according to the period of Eq. (49).

In order to control the lateral foot-contact period, the foot width is adjusted as shown in Fig. 12. $l - \Delta l + \delta l$ and $\phi_2 + \delta\phi_2$ are acquired from ϵ geometrically. It is assumed that this adjustment is so small that its effect on

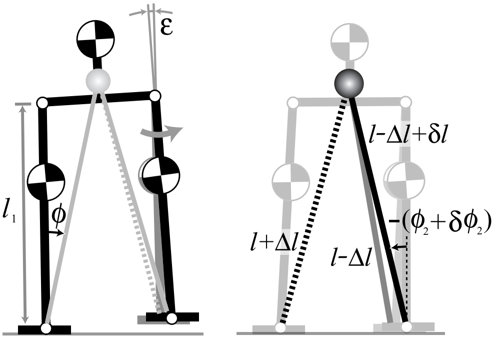


Fig. 12. Adjustment of foot width. ϵ is the angle to open the swing leg to adjust the foot width. $l - \Delta l + \delta l$ and $\phi_2 + \delta\phi_2$ are the pendulum length and angle at the beginning of phase (B) after adjustment

the target dynamics in phase (A) can be neglected. By the adjustment, the parameters of the target dynamics in phase (B) are varied as follows:

$$a_2 = \frac{1}{\big(\phi_3 - (\phi_2 + \delta\phi_2)\big)^2}(\Delta l - \delta l) \tag{72}$$

$$b_2 = -\frac{2\phi_2}{\big(\phi_3 - (\phi_2 + \delta\phi_2)\big)^2}(\Delta l - \delta l) \tag{73}$$

$$c_2 = l_0 - \frac{\phi_3^2 - 2(\phi_2 + \delta\phi_2)\phi_3}{\big(\phi_3 - (\phi_2 + \delta\phi_2)\big)^2}(\Delta l - \delta l)\ . \tag{74}$$

The condition that the pendulum pauses at the end of phase (B) is $F_{l_B}(\phi_3) = 0$, hence

$$D_{l_B}(\phi_3) - D_{l_B}(\phi_2 + \delta\phi_2) + \Big(M(\phi_2 + \delta\phi_2)\dot{\phi}_2\Big)^2 = 0\ . \tag{75}$$

In addition, the condition that the foot-contact period must satisfy is

$$\int_{\phi_2+\delta\phi_2}^{\phi_3} \frac{1}{F_{l_B}(\phi)}d\phi + \int_{\phi_3}^{\phi_1} \frac{1}{F_{l_A}(\phi)}d\phi = T_s\ . \tag{76}$$

The first term of the left side in Eq. (76) is the period of phase (B) and the second term is that of the subsequent phase (A). The two conditions of Eq. (75) and (76) have two unknowns, i.e. the adjustment value, ϵ, and the pendulum angle at the end of phase (B), ϕ_3. By solving these two conditions by means of linear approximation of ϵ and two dimensional approximation of ϕ_3, the adjustment value, ϵ, can be calculated

$$\epsilon = \epsilon(\phi, \dot{\phi}, T_s)\ . \tag{77}$$

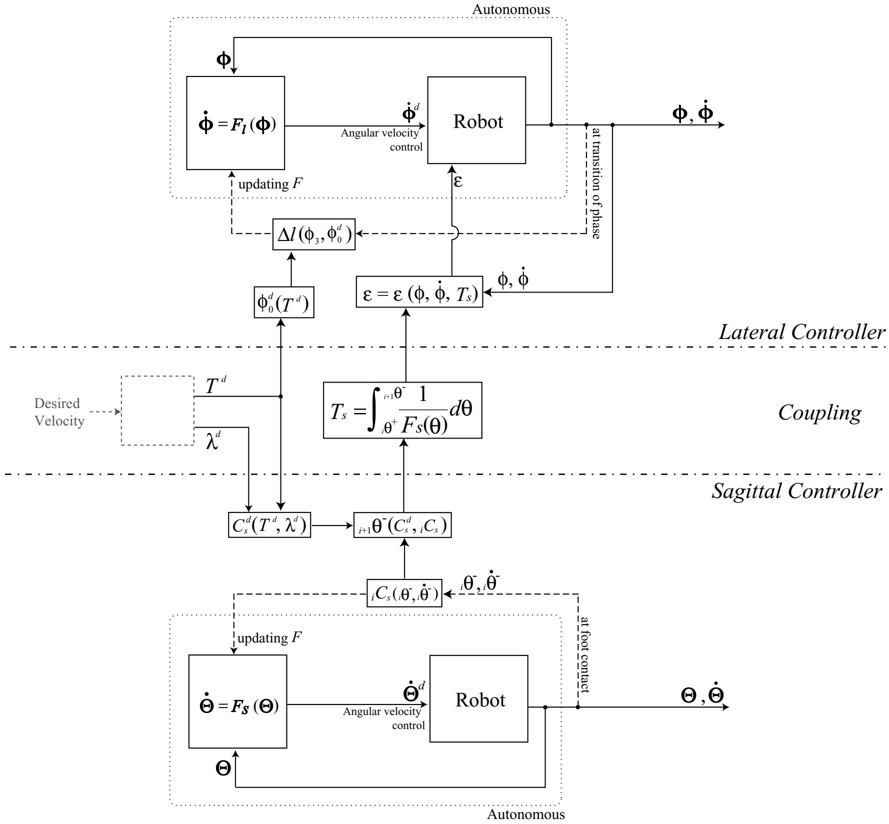


Fig. 13. Block diagram of the coupling between the sagittal and lateral motions

Figure 13 depicts the block diagram of the algorithm described in the previous sections. At foot-contact, the sagittal controller decides the step-length, i.e. the value of θ at the next foot-contact in order to stabilize the sagittal motion. Next, the foot-contact period of the sagittal motion is calculated by integration. Finally, the lateral controller determines the adjustment value of foot width according to both the sagittal foot-contact period and the present status in the lateral plane. This series of controls can be considered as the landing position control of three dimensional walking since the step-length is adjusted in the sagittal plane and the foot width is adjusted in the lateral plane.

The box enclosed by a gray dashed line is the algorithm to decide the desired foot-contact period and step-length so that the energy consumption is minimized. However, this has not been solved and is future work, hence we

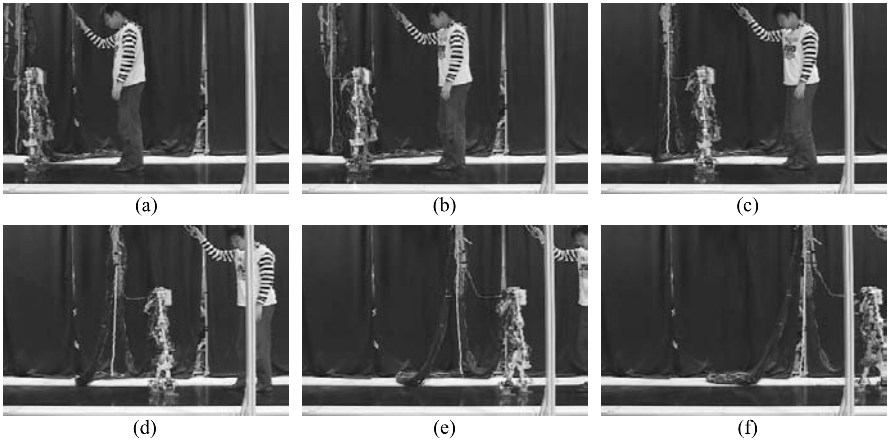


Fig. 14. Snapshots of the walking of PDAC. Each figure shows the snapshots at (**a**)1st (**b**)7th (**c**)12th (**d**)16th (**e**)19th (**f**)22nd step

give the desired foot-contact period and step-length to the controller directly in this paper.

5 Experiment

The experiment of the walking proposed in the previous section on flat and level ground was conducted. Since, in order to start the walking, the robot needs potential energy, we lifted up the lateral pendulum to the position at the beginning of phase (A) and released. In experiment, the robot bends its knee joint of the swing leg so as to prevent the foot being in friction with the ground immediately after foot-contact on the assumption that the effect of knee bending on the robot dynamics can be neglected. The foot of the swing leg is actuated so as to be kept parallel to the ground.

The desired step-length is given to be gradually increased within the initial 5 steps up to 0.15[m] and the desired foot-contact period is given at 0.7[s]. In consequence, dynamic and natural walking is realized over 25 steps. The step-length is about 0.15[m] and the walking velocity is about 0.23[m/s]. Figure 14 shows the snapshots of the PDAC walking at the 1st, 7th, 12th, 16th, 19th, 22nd step respectively. The angle and angular velocity of the lower body joints are depicted in Fig. 15 and Fig. 16. As shown in these figures, smooth dynamics motion is realized periodically.

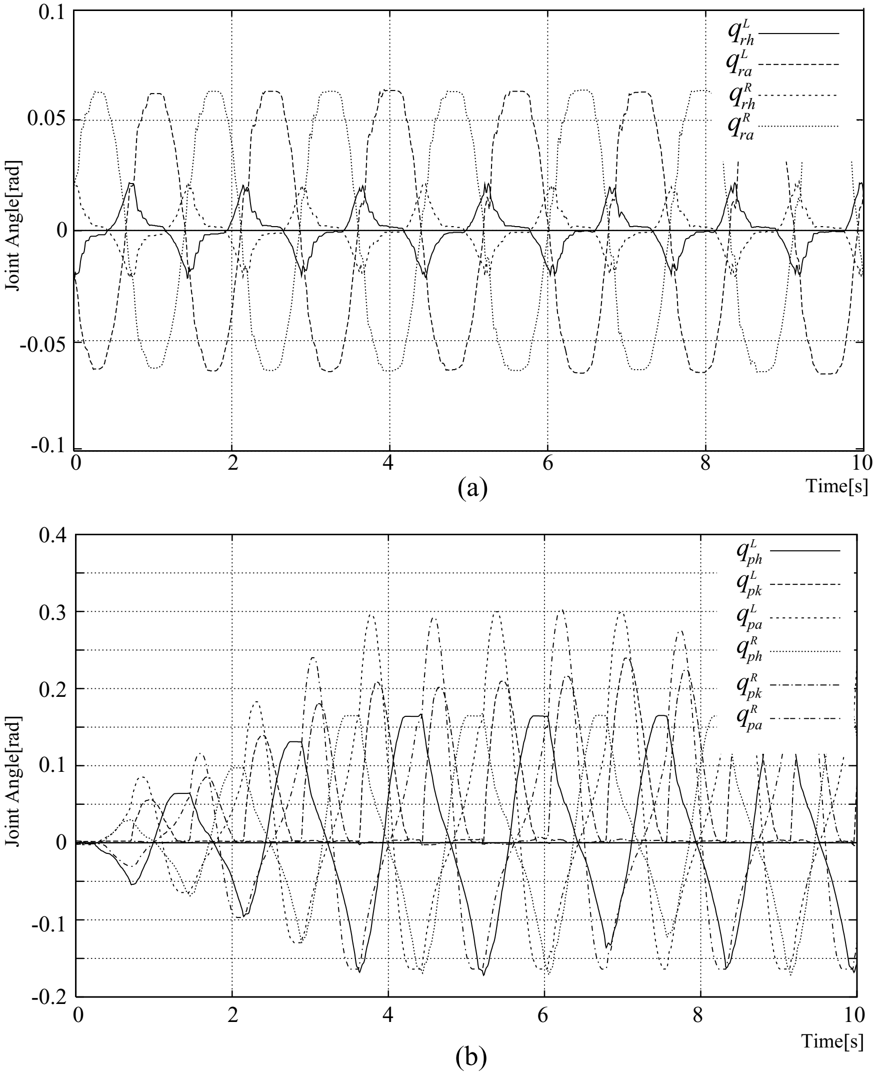


Fig. 15. Angle of the joints (**a**) in the lateral plane (**b**) in the sagittal plane

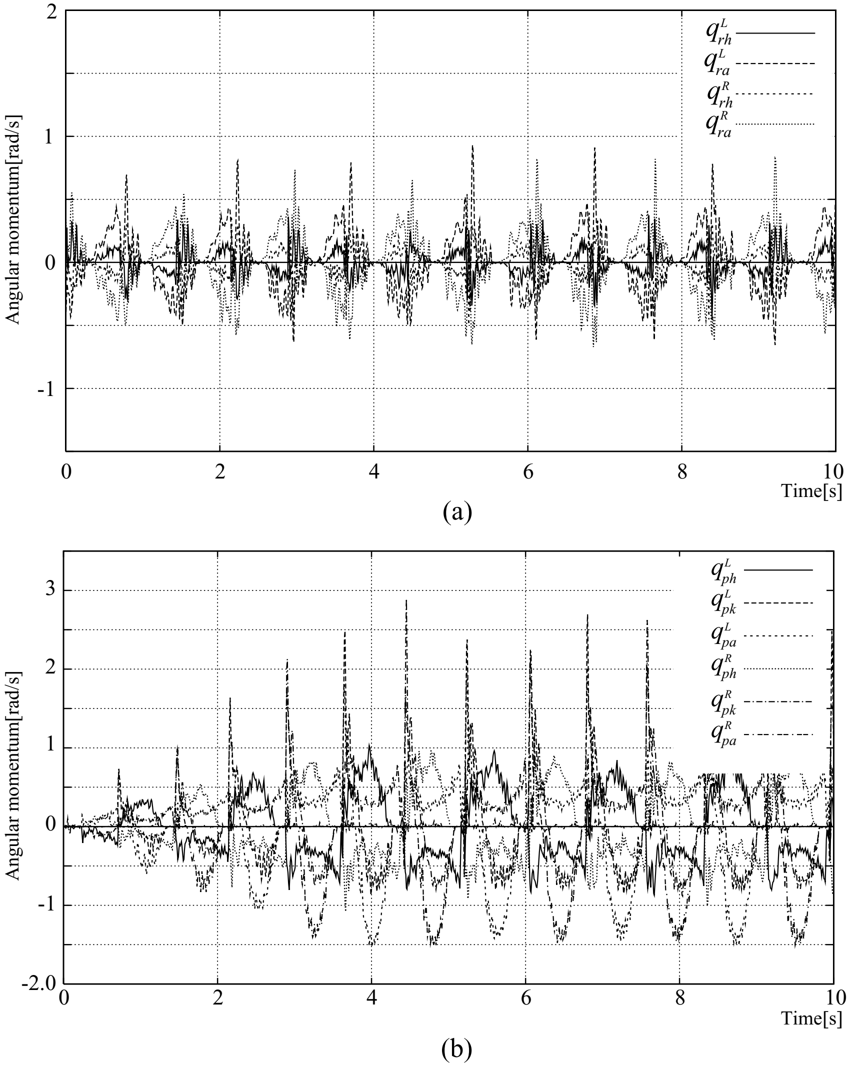


Fig. 16. Angular Velocity of the joints (**a**) in the lateral plane (**b**) in the sagittal plnae

6 Conclusion

This paper first introduced a multi-locomotion robot with high mobility and then proposed Passive Dynamic Autonomous Control (PDAC) for the comprehensive control method of multiple types of locomotion. PDAC is the method to take advantage of the robot inherent dynamics and realize natural dynamic motion. We applied PDAC to the biped walk control. On the assumption that the sagittal and lateral motion can be separated and controlled individually, each motion was designed based on the given desired step-length and period. In order to stabilize walking, the landing position control according to the status was designed. In addition, a coupling method between these motions, which makes the period of each motion identical, was proposed. Finally, the 3-dimensional dynamic walking whose step-length is about 0.15[m] and velocity is about 0.23[m/s] was realized.

References

[1] Matthew D. Berkemeier and Ronald S. Fearing (1999), Tracking Fast Inverted Trajectories of the Underactuated Acrobot, IEEE Trans. on Robotics and Automation, 15(4):740–750.

[2] M. Doi, Y. Hasegawa, and T. Fukuda (2004), Passive Trajectory Control of the Lateral Motion in Bipedal Walking, Proc. of IEEE International Conference on Robotics and Automation: 3049–3054.

[3] M. Doi, Y. Hasegawa, and T. Fukuda (2004), Passive Dynamic Control of Bipedal Walking, Proc. of IEEE-RAS/RSJ International Conference on Humanoid Robots.

[4] T. Fukuda, H. Hosokai and Y. Kondo (1991), Brachiation type of mobile robot, Proc. of IEEE Int. Conf. Advanced Robotics: 915–920.

[5] T. Fukuda, F. Saito and F. Arai (1991), A study on the brachiation type of mobile robot(heuristic creation of driving input and control using CMAC), Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems: 478–483.

[6] A. Goswami, B. Espiau and A. Keramane (1997), Limit cycles in a passive compass gait biped and passivity-mimicking control laws, Autonomous Robots 4:274–286, Kluwer Academic Publishers.

[7] A.A. Grishin, A.M. Formal'sky, A.V. Lensky and S.V. Zhitomirsky (1994), Dynamic Walking of a Vehicle With Two Telescopic Legs Controlled by Two Drives, The International Journal of Robotics Research, 13(2):137–147.

[8] J.W. Grizzle, G. Abba and F. Plestan (2001), Asymptotically Stable Walking for Biped Robots: Analysis via Systems with Impulse Effects, IEEE Trans. on Automatic Control, 46(1):56–64.

[9] Y. Hasegawa, T. Fukuda and K. Shimojima (1999), Self-scaling reinforcement learning for fuzzy logic controller – applications to motion control of two-link brachiation robot, IEEE Trans. Industrial Electronics, 46(6):1123–1131.

[10] Y. Hasegawa, Y. Ito and T. Fukuda (2000), Behavior coordination and its modification on brachiation-type mobile robot, Proc. of IEEE Int. Conf. on Robotics and Automation: 3984–3989.

[11] H. Hemami and B.F. Wyman (1979), Modeling and Control of Constrained Dynamic Systems with Application to Biped Locomotion in the Flontal Plane, IEEE Trans. on Automatic Control, AC-24(4): 526–535.

[12] H. Kajima, Y. Hasegawa and T. Fukuda (2002), Study on brachiation controller -adjustment method of strength and timing parameters-, Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems: 2497–2502.

[13] S. Kajita, T. Yamaura and A. Kobayashi (1992), Dynamic Walking Control of a Biped Robot Along a Potential Energy Conserving Orbit, IEEE Trans. on Robotics and Automation, 8(4):431–438.

[14] A.D. Kuo (1999), Stabilization of Lateral Motion in Passive Dynamic Walking, The International Journal of Robotics Research, 18(9):917–930.

[15] Qinghua Li, Atsuo Takanishi and Ichiro Kato (1992), Learning control of compensative trunk motion for biped walking robot based on ZMP, Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems: 597–603.

[16] T. McGeer (1990), Passive dynamic walking, The Int. Journal of Robotics Research, 9(2):62–82.

[17] H. Miura and I. Shimoyama (1984), Dynamic Walking of a biped, The International Journal of Robotics Research, 3(2):60–74.

[18] J. Nakanishi, T. Fukuda and D.E. Koditschek (2000), A brachiating robot controller, IEEE Trans. on Robotics and Automation, 16(2):109–123.

[19] K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba and H. Inoue (2002), Online Generation of Humanoid Walking Motion based on a Fast Generation Method of Motion Pattern that Follows Desired ZMP, Proc. of the Int. Conf. Intelligent Robots and Systems: 2684–2689.

[20] F. Plestan, J.W. Grizzle, E.R. Westervelt and G. Abba (2003), Stable Walking of a 7-DOF Biped Robot, IEEE Trans. on Robotics and Automation, 19(4):653–668.

[21] M.H. Raibert (1986), Legged Robots that Balance. MIT Press, Cambridge, MA.

[22] F. Saito, T. Fukuda and F. Arai (1993), Swing and locomotion control for two-link brachiation robot, Proc. of IEEE Int. Conf. on Robotics and Automation: 719–724.

[23] F. Saito, T. Fukuda and F. Arai (1994), Swing and locomotion control for a two-link brachiation robot, IEEE Control Syst. Mag. 14(1):5–12.

[24] F. Saito and T. Fukuda (1997), A First Result of The Brachiator III – A New Brachiation Robot Modeled on a Siamang, Artificial Life V: 354–361, MIT Press.

[25] A. Sano and J. Furusho (1991), Realization of Dynamic Quadruped Locomotion in Pace Gait by Controlling Walking Cycle, International Symposium on Experimental Robotics: 491–502.

[26] E.L. Simons (1972), Primate Evolution: An Introduction to Man's Place in Nature, Macmillan.

[27] M. Spong, (1995) The swing up control problem for the acrobot, IEEE Control Syst. Mag. 15(1):49–55.

[28] A. Takanishi, M. Ishida, Y. Yamazaki and I. Kato (1985), The Realization of Dynamic Walking Robot WL-10RD, Proc. of Int. Conf. Advanced Robotics: 459–466.

[29] B. Thuilot, A. Goswami and B. Espiau (1997), Bifurcation and Chaos in a Simple Passive Bipedal Gait, Proc. of Int. Conf. Robotics and Automation: 792–798.

[30] James R. Usherwood and John E.A. Bertram (2003), Understanding brachiation: insight from a collisional perspective, The Journal of Experimental Biology, 206:1631–1642.

[31] M. Vukobratovic, B. Brovac, D. Surla and D. Stokic (1990), Biped Locomotion: dynamics, stability, control, and application, Springer-Verlag.

[32] E.R. Westervelt, G. Buche and J.W. Grizzle (2004), Inducing Dynamically Stable Walking in an Underactuated Prototype Planar Biped, Proc. of Int. Conf. Robotics and Automation: 4234–4239.

[33] M. Yamada, J. Furusho and A. Sano (1985), Dynamic control of walking robot with kick-action, Proc. of the ICAR: 405–412.

On the Determination of the Basin of Attraction for Stationary and Periodic Movements

P. Giesl[1] and H. Wagner[2]

[1] Zentrum Mathematik, TU München, Boltzmannstr. 3, D-85747 Garching bei München, Germany
giesl@ma.tum.de

[2] Motion Science, University of Münster, Horstmarer Landweg 62b, D-48149 Münster, Germany
heiko.wagner@uni-muenster.de

Movements of humans are achieved by muscle contractions. Humans are able to perform coordinated movements even in the presence of perturbations from the environment or of the muscles themselves. But which properties of the muscles and the geometry of the joints are responsible for the stability? Does the stability depend on the joint angle? How large are the perturbations, the muscle-skeletal system can cope with before reflexes or controls by the brain are necessary? To answer these questions, we will derive a mathematical model of the muscle-skeletal system without reflexes. We present different mathematical methods to analyze these systems with respect to the stability of movements and thus provide the mathematical tools to answer the above questions. This paper is a companion paper to [13] where the biological applications of the mathematical methods presented in this paper are discussed in more detail.

Stationary and periodic movements are modelled by autonomous and time-periodic differential equations. If small perturbations to these movements are corrected by the system, the movement is called stable and the set of these perturbations is called the basin of attraction of the movement. The basin of attraction is the appropriate quantity to describe how stable a movement is, since it measures, how large perturbations to the movement may be, which still are led back to the desired movement. The basin of attraction thus describes the self-stabilizing properties of the muscle-skeletal system without control mechanisms. If a human runs, it is important how uneven the ground may be before he either falls down or has to adjust his muscle activations to the new situation.

Let us describe how the paper is organized: In Sect. 1 we present a mathematical model of a single human joint including antagonistic muscles. More precisely, we introduce models of a human elbow joint and a human knee including muscular and outer forces as well as the geometry of the joint. In Sects. 2 and 3 we present different mathematical methods to analyze the stability and the basin of attraction of a stationary or periodic movement, i.e. for autonomous and periodic systems, respectively. Some examples of these methods in biomechanical applications are given in this paper, and many more are presented in [13] within this volume.

1 Biomechanical Models

In Sect. 1.1 we present a mathematical model of the human elbow joint, for more details cf. [7] and [9]. In Sect. 1.2 we describe the model of the human knee joint. Note that the models of other joints and other positions of the arms and legs can be derived in a very similar way.

1.1 A Model of the Elbow Joint

Consider the following situation: the upper arm of a person is attached to the body, the elbow pointing downwards. The lower arm (ulna) is free to move in the sagittal plane so that the system is totally described by the angle β between upper and lower arm at the elbow joint. The person holds a load in the hand. Denoting the angular velocity $\omega = \dot{\beta}$ we obtain the following equation of motion:

$$\begin{cases} \dot{\beta} = \omega \\ \dot{\omega} = \frac{1}{J} T(t, \beta, \omega) =: f(t, \beta, \omega) \end{cases} \tag{1}$$

where $J = \left(\frac{1}{3}m_u + m_l\right) l^2$ denotes the moment of inertia, m_u and m_l denote the mass of the ulna and of the load, respectively, and l is the length of the ulna. The torque $T(t, \beta, \omega) = T_o(t, \beta, \omega) + T_m(t, \beta, \omega)$ consists of two parts corresponding to the outer forces and the muscle forces.

The outer forces include the gravitational forces acting on the arm and on the load in the hand. In the periodic case we assume that the person is walking which gives an additional term corresponding to the periodic vertical acceleration $a(t)$ applied to the system. Altogether, $T_o(t, \beta, \omega) = \left(\frac{1}{2}m_u + m_l\right) (g + a(t))\, l \sin\beta$, cf. [2].

In our model we consider only the three most important muscles at the elbow joint, namely the extensor muscle triceps brachii and the two flexor muscles biceps and brachioradialis. For each muscle the torque is given by $T(t, \beta, \omega) = E(t) \cdot f_l(\beta) \cdot H(\beta, \omega) \cdot h(\beta)$, where $E(t) \in [0, 1]$ denotes the activation level of the muscle which is assumed to be constant or periodic with respect to t, f_l denotes the dependency of the muscle on its length, H denotes the Hill-function modelling the dependency of the muscle on its velocity and h denotes the effective moment arm, reflecting the geometry of the joint. We will discuss the force-length function f_l and the Hill-function H in more detail.

Force-Length Function

We model the force-length functions $f_{l_{bic}}$ of the biceps by (2), cf. [9],

$$\begin{aligned} f_{l_{bic}}(\beta) &= \tilde{f}_l(z_{bic}(l_{bic}(\beta))) \ , & (2) \\ \tilde{f}_l(z) &= \tilde{a} \arctan[\tilde{b}(z-\tilde{c})] + \tilde{d} \ , & (3) \\ z_{bic}(x) &= \frac{z_0}{l_{bic}(8\pi/9) - l_{bic}(\pi/3)} \left[x - l_{bic}\,(8\pi/9)\right] + 1 \ , & (4) \end{aligned}$$

where $\tilde{a} = \frac{1.09}{\pi}$, $\tilde{b} = 15$, $\tilde{c} = 0.6$, $\tilde{d} = 0.49$. $\tilde{f}_l$ of (3) models the function described in [11], β denotes the joint angle, $l_{bic}(\beta) = \sqrt{k_H^2 + k_U^2 - 2k_H k_U \cos\beta}$ the length of the muscle depending on personal data such as the distances between the elbow joint and the point where the tendon is attached to the upper and lower arm k_H, k_U, respectively. z_{bic} denotes a normalized muscle length, cf. [7]. z_0 is a muscle-dependent parameter, often $z_0 = 0.55$. The formulas for M. brachioradialis are similar.

The extensor muscle of the elbow joint is approximately working around the optimum muscle length. Therefore, the influence of the force-length relation can be neglected, i.e. we set $f_{l_{ext}}(\beta) = 1$.

Hill-Function

Hill [10] introduced the following force-velocity relation

$$H(v) = \frac{c}{v+b} - a \text{ for } v \geq 0 \tag{5}$$

where $v \geq 0$ denotes the velocity of the contraction of the muscle and a, b, c are person-dependent constants. The excentric part, i.e. $v < 0$ is modelled such that H is a C^2-function, cf. [7]

$$H(v) = A + \frac{B}{v-D} + \frac{C}{(v-D)^2} \text{ for } v < 0 \tag{6}$$

where A, B, C, D are chosen such that $H \in C^2(\mathbb{R}^2, \mathbb{R})$ and $\lim_{v\to-\infty} H(v) = 1.5 \cdot H(0)$, i.e. the muscle can generate 1.5 times the isometric force $H(0)$ for very large negative velocities, cf. [7].

The Formula for T

Note that the velocity of the muscle v is connected to the angular velocity ω by $v = h(\beta) \cdot \omega$, where h denotes the effective moment arm, cf. [7]. Altogether, T_m is given by

$$\begin{aligned} T_m(t,\beta,\omega) = \; & E_{ext}(t)\, H_{ext}[h_{ext}(\beta)\cdot\omega]\, h_{ext}(\beta) \\ & + E_{flex}(t)\, f_{l_{bic}}(\beta)\, H_{bic}[h_{bic}(\beta)\cdot\omega]\, h_{bic}(\beta) \\ & + E_{flex}(t)\, f_{l_{brach}}(\beta)\, H_{brach}[h_{brach}(\beta)\cdot\omega]\, h_{brach}(\beta) \qquad (7) \end{aligned}$$

Recall that E denotes the (periodic or constant) muscle activation, H the Hill-function, h the effective moment arm and f_l the force-length relation. The index denotes the flexor muscles biceps and brachioradialis as well as the extensor muscle. Note that the activation levels of both flexor muscles are assumed to be equal and are denoted by $E_{flex}(t)$.

Plugging the formulas for T_o and T_m, cf. (7), in (1) we obtain the following system.

$$\begin{cases} \dot{\beta} &= \omega \\ \dot{\omega} &= \frac{1}{J}\left[\left(\frac{1}{2}m_u + m_l\right)(g + a(t))\, l\, \sin\beta + E_{ext}(t)\, H_{ext}[h_{ext}(\beta)\cdot\omega]\, h_{ext}(\beta) \right. \\ &\qquad +E_{flex}(t)\, f_{l_{bic}}(\beta)\, H_{bic}[h_{bic}(\beta)\cdot\omega]\, h_{bic}(\beta) \\ &\qquad \left. +E_{flex}(t)\, f_{l_{brach}}(\beta)\, H_{brach}[h_{brach}(\beta)\cdot\omega]\, h_{brach}(\beta)\right] \\ &=: f(t,\beta,\omega) \end{cases} \tag{8}$$

Special Features

For the mathematical analysis we hence study a system of differential equation of the following form

$$\begin{cases} \dot{\beta} = \omega \\ \dot{\omega} = f(t,\beta,\omega) \end{cases} \tag{9}$$

where $f(t,\beta,\omega)$ is either (i) independent of t (autonomous system) or (ii) periodic with respect to t, i.e. there is a minimal period $T > 0$ such that $f(t+T,\beta,\omega) = f(t,\beta,\omega)$ for all (t,β,ω). Note that the partial derivative of the Hill-function with respect to ω is strictly negative, i.e. $H_\omega(\beta,\omega) < 0$ holds for all $(\beta,\omega) \in \mathbb{R}^2$. Hence, this also holds for f, i.e.

$$f_\omega(t,\beta,\omega) < 0 \tag{10}$$

for all (t,β,ω).

Examples

In this paper we study the following two examples:

1. **Standing.** We assume that the person is not moving (T_o is independent of t), and holds the weight at constant activation levels E_{ext} and E_{flex} (T_m is independent of t) at angle β_0. Thus, $f = f(\beta,\omega)$ is independent of t and E_{ext} and E_{flex} are chosen such that $f(\beta_0, 0) = 0$ holds, i.e. $\beta(t) = \beta_0$ and $\omega(t) = 0$ is a solution.
2. **Walking.** We assume that the person is walking in a periodic way (T_o is periodic with respect to t), and holds the weight at a constant level for a person watching from outside. An example for such a situation is a waiter carrying a tray while walking. The tray is supposed not to move, so that the elbow angle must compensate the movement the walking imposes on the shoulder and the arm. This is achieved by suitable periodic activation levels $E_{ext}(t)$ and $E_{flex}(t)$. In this case, $f = f(t,\beta,\omega)$ is a periodic function of t, cf. Sect. 3.

1.2 A Model of the Knee Joint

We assume that the foot is fixed to the ground, the upper and lower legs are connected by the knee joint, the body mass is concentrated in the hip and the hip can only move on a line perpendicular to the ground. The height of the hip is denoted by x, its velocity by $v = \dot{x}$. In this case, our model includes one extensor and one flexor muscle, and we set $f_{l_{ext}}(x) = f_{l_{flex}}(x) = 1$. We neglect the mass of the leg, and we set the Hill-function of the flexor $H_{flex}(v) = f_{iso_{flex}}$. We consider the following equations of motion, cf. [12]

$$\begin{cases} \dot{x} &= v \\ \dot{v} &= E_{ext}(t) H_{ext}[h_{ext}(x) \cdot v]\, h_{ext}(x) + E_{flex}(t) f_{iso_{flex}} h_{flex}(x) - mg \\ &=: f(t, x, v), \end{cases} \tag{11}$$

where the effective moment arms and the Hill-function of the extensor are modelled by

$$\begin{aligned} h_{ext}(x) &= a_{ext} + \frac{c_{ext}}{b_{ext} - x} \\ h_{flex}(x) &= -a_{flex} - \frac{c_{flex}}{b_{flex} - x} \\ H_{ext}(u) &= \tanh[-(5.6 + 4.5 \tanh(-u - 0.4))u] \cdot \\ &\qquad f_{isoext}(0.25 \tanh(10u) + 0.75) + f_{iso_{ext}}, \end{aligned}$$

where $a_{ext} = 0.08$, $b_{ext} = 0.88$, $c_{ext} = 0.035$, $a_{flex} = 0.01$, $b_{flex} = 0.87$, $c_{flex} = 0.04$, $f_{iso_{ext}} = f_{iso_{flex}} = 21012$. Note that this model approximates the effective moment arms of [12], Fig. 6. These moment arms include a model of Menschik with moving center of rotation in the knee. The length are 44 cm of the thigh and 43 cm for the lower leg. We use this model in Sect. 3.6.

2 Autonomous Systems

In this section we study equilibria and their stability of the autonomous system (12), which is (9) being independent of t.

$$\begin{cases} \dot{\beta} = \omega \\ \dot{\omega} = f(\beta, \omega) \end{cases} \tag{12}$$

Whereas the stability of equilibria can often be checked by the eigenvalues, for the basin of attraction we use Lyapunov functions (Sect. 2.1). The following methods for their construction are discussed: via linearization (Sect. 2.2), using special properties of the equation (Sect. 2.3) and with radial basis functions (Sect. 2.4). Note that for the sake of simplicity the Theorems and Definitions in this paper are only stated for the two-dimensional phase space $x = (\beta, \omega) \in \mathbb{R}^2$, but they hold in most cases also for arbitrary dimensions, i.e. $x \in \mathbb{R}^n$.

2.1 Basin of Attraction, Stability and Lyapunov Functions

The equilibria of (12), i.e. solutions which are constant in time, are the zeros (β, ω) of the right-hand side vector field of (12), i.e. $\omega = 0$ and $f(\beta, \omega) = 0$. Hence, equilibria are points $(\beta_0, 0)$ which satisfy $f(\beta_0, 0) = 0$. For the muscle model this is achieved by suitable activations E_{ext} and E_{flex}.

We also use the notation

$$\dot{x} = F(x) , \tag{13}$$

where $x = (\beta, \omega)$ and $F(x) = \binom{x_2}{f(x_1, x_2)}$ for the equation (12).

The stability of such an equilibrium $(\beta_0, 0)$ in the hyperbolic case can by studied by the eigenvalues of the Jacobian matrix of first derivatives

$$DF(\beta_0, 0) = \begin{pmatrix} 0 & 1 \\ f_\beta(\beta_0, 0) & f_\omega(\beta_0, 0) \end{pmatrix} ,$$

which are

$$\lambda_{1,2} = \frac{1}{2} \left(f_\omega(\beta_0, 0) \pm \sqrt{f_\omega(\beta_0, 0)^2 + 4 f_\beta(\beta_0, 0)} \right)$$

if $f_\omega(\beta_0, 0)^2 + 4 f_\beta(\beta_0, 0) \geq 0$ (real eigenvalues) and

$$\lambda_{1,2} = \frac{1}{2} \left(f_\omega(\beta_0, 0) \pm i \sqrt{-f_\omega(\beta_0, 0)^2 - 4 f_\beta(\beta_0, 0)} \right)$$

otherwise (complex eigenvalues). In both cases, the real parts of both eigenvalues are strictly negative, if and only if $f_\beta(\beta_0, 0) < 0$, since $f_\omega(\beta_0, 0) < 0$ holds by (10). Hence, if $f_\beta(\beta_0, 0) < 0$, then the equilibrium is asymptotically stable.

From now on we will assume $f(\beta_0, 0) = 0$ and $f_\beta(\beta_0, 0) < 0$. We seek to determine the basin of attraction of the asymptotically stable equilibrium $(\beta_0, 0) =: x_0$.

Definition 1. *The basin of attraction $A(x_0)$ of an asymptotically stable equilibrium x_0 for the ordinary differential equation $\dot{x} = F(x)$ is defined by*

$$A(x_0) = \{\xi \in \mathbb{R}^2 \mid x(t) \overset{t \to \infty}{\longrightarrow} 0\}$$

where $x(t)$ denotes the solution of $\dot{x} = F(x)$ with initial value $x(0) = \xi$.

A powerful method to determine the basin of attraction is the method of a Lyapunov function $v \colon \mathbb{R}^2 \to \mathbb{R}$. The main property of a Lyapunov function is that it is decreasing along solutions. v is decreasing along solutions, if and only if the orbital derivative v', i.e. the derivative of v along solutions of the differential equation, is negative. The formula for the orbital derivative follows from the chain rule

$$\frac{d}{dt}v(x(t))\Big|_{t=0} = \langle \nabla v(x(t)), \dot{x}(t)\rangle\Big|_{t=0} \overset{(13)}{=} \langle \nabla v(\xi), F(\xi)\rangle = v'(\xi)\ .$$

Theorem 1 (Lyapunov function). *Let x_0 be an equilibrium of $\dot{x} = F(x)$. Let $v \in C^1(\mathbb{R}^2, \mathbb{R})$ be a function and $K \subset \mathbb{R}^2$ be a compact set with neighborhood B such that*

1. $x_0 \in \overset{\circ}{K}$,
2. $v'(x) = \langle \nabla v(x), F(x)\rangle < 0$ holds for all $x \in K \setminus \{x_0\}$,
3. $K = \{x \in B \mid v(x) \leq R\}$ with a constant $R \in \mathbb{R}$.

Then $K \subset A(x_0)$.

We sketch the proof of Theorem 1: If we start at $x(0)$ with $v(x(0)) \leq R$, then the solution satisfies $v(x(t)) \leq R$ for all $t \geq 0$ since v is decreasing along solutions, i.e. solutions stay in the set K of Theorem 1. Since the function v is bounded from below, $v(x(t))$ will tend to a constant value and the orbital derivative $v'(x(t))$ will tend to 0. Thus, the solution tends to the only point where $v'(x) = 0$ holds, namely to the equilibrium x_0, and $x(0)$ belongs to the basin of attraction $A(x_0)$.

Thus, level sets of v provide a tool to determine a subset of the basin of attraction. Although there are many existence theorems for Lyapunov functions, their explicit construction without knowledge of the solutions of $\dot{x} = F(x)$ is a difficult problem. Several methods for the calculation of Lyapunov functions are presented in the next sections.

2.2 Linearization

We consider the linearized system at the equilibrium point x_0, namely $\dot{x} = DF(x_0)(x - x_0)$. This is a linear system and, thus, one can easily calculate a Lyapunov function of the form $v(x) = (x - x_0)^T C(x - x_0)$, where the positive definite matrix C is the unique solution of the matrix equation $DF(x_0)^T C + CDF(x_0) = -I$. Note that the orbital derivative is given by $v'(x) = (x - x_0)^T [DF(x_0)^T C + CDF(x_0)](x - x_0) = -\|x - x_0\|^2$. The function v is not only a Lyapunov function for the linearized system, but also for the nonlinear system in a neighborhood of x_0, cf. Lemma 1. We call such a function a *local Lyapunov function*, since this neighborhood can be very small and thus the set K of Lemma 1 may be a very small subset of the basin of attraction $A(x_0)$. Thus, other methods will be discussed in the following sections.

Lemma 1. *Under the above assumptions on the Lyapunov function $v(x) = (x - x_0)^T C(x - x_0)$ of the linearized system, there is a compact set K with neighborhood B such that $x_0 \in \overset{\circ}{K}$, $v'(x) = \langle \nabla v(x), F(x)\rangle < 0$ holds for all $x \in K \setminus \{x_0\}$ and $K = \{x \in B \mid v(x) \leq R\}$ with $R > 0$.*

2.3 Special Lyapunov Function

In many applications, a Lyapunov function is calculated using special properties or physical insight into the system of differential equation considered. Also in our case one can use the special structure (12) and the information on the negative sign of f_ω, cf. (10), to obtain a Lyapunov function, cf. [9].

Theorem 2. *Let $f \in C^1(\mathbb{R}^2, \mathbb{R})$ satisfy $f_\omega(\beta, \omega) < 0$ for all (β, ω). Moreover, assume that $(\beta_0, 0)$ is an equilibrium point of (12) with $f_\beta(\beta_0, 0) < 0$. Then*

$$V(\beta, \omega) = -\int_0^\beta f(\tilde{\beta}, 0)\, d\tilde{\beta} + \frac{1}{2}\omega^2 \tag{14}$$

is a Lyapunov function such that

1. *$V'(\beta, \omega) < 0$ for all (β, ω) with $\omega \neq 0$,*
2. *V attains a local minimum at $(\beta_0, 0)$.*

The integral in (14) can be calculated explicitly in the case of our model (8), cf. [9]. Using the Lyapunov function V we obtain again a subset S of the basin of attraction by Theorem 1 through level sets of V.

Corollary 1. *Let the assumptions of Theorem 2 hold. Let $(\beta_0, 0)$ be an equilibrium with $\beta_0 \in (0, \pi)$. Moreover, let $\beta_0 < \beta_1 < \pi$ be such that $(\beta_1, 0)$ is an equilibrium. Set*

$$S := \{(\beta, \omega) \mid V(\beta, \omega) < V(\beta_1, 0)\} \cap (0, \pi) \times \mathbb{R}$$

If $\overline{S}$ is connected and compact, and $f(\beta, 0) \neq 0$ holds for all $(\beta, 0) \in S \setminus \{(\beta_0, 0)\}$, then $S \subset A(\beta_0, 0)$.

The set S is in general larger than the set K of Lemma 1. In particular, it covers the whole β-axis up to the next unstable equilibrium $(\beta_1, 0)$. A similar Corollary holds also for an equilibrium $(\beta_2, 0)$ with $\beta_2 < \beta_0$.

2.4 Approximation via Radial Basis Functions

A Lyapunov function is characterized by its negative orbital derivative. In this general approach to construct Lyapunov functions via radial basis functions we consider Lyapunov functions V_1 and V_2 with certain negative orbital derivatives, namely $V_1'(x) = -\|x - x_0\|^2$ defined for $x \in A(x_0)$ and $V_2'(x) = -c < 0$ defined for $x \in A(x_0) \setminus \{x_0\}$. These equations for the orbital derivatives are linear first-order partial differential equations.

The method seeks to find an approximate solution v_i, $i = 1, 2$ of the partial differential equation which is close enough to the solution V_i such that the orbital derivative v_i' is negative. Then v_i is itself a Lyapunov function and can be used to determine a subset of the basin of attraction by Theorem 1.

The approximation is achieved using radial basis functions [8], which is a meshfree approximation method using a certain ansatz for the approximating function v. More precisely, for $x, y \in \mathbb{R}^2$ we set

$$v(x) = \sum_{j=1}^{N} \alpha_j \langle \nabla_y \varPsi(x-y), F(x_j) \rangle \Big|_{y=x_j} \tag{15}$$

where $\varPsi(x) = \psi(\|x\|)$ denotes a fixed radial basis function and $X_N = \{x_1, \ldots, x_N\} \subset \mathbb{R}^2$ a grid. The coefficients α_j are chosen such that v satisfies the equation on the grid, i.e. $v_1'(x_j) = V_1'(x_j) = -\|x_j - x_0\|^2$ or $v_2'(x_j) = V_2'(x_j) = -c$ for all $j = 1, \ldots, N$. The explicit calculation of α_j is easily achieved solving a system of linear equations.

The density of the grid points used for the approximation determines the error of the approximation. The following error estimates are obtained, using Wendland's functions as radial basis functions, cf. [14] and [8]. Theorem 3 shows that if the grid is dense enough, then $v_i'(x) < 0$ holds. Finally, by level sets of the function v_i and Theorem 1, we can find a subset of the basin of attraction.

Note that we can modify the method by not only prescribing the values of v_i' on the grid X_N but, additionally, the values of v_i on another grid $\varXi_M$, e.g. on a level set of the local Lyapunov function. Thus, one can find subsets of the basin of attraction which cover each compact set in the basin of attraction, cf. [8].

Theorem 3 (Error estimate). *Consider $\dot{x} = F(x)$, and let $f \in C^\sigma(\mathbb{R}^2, \mathbb{R})$ where $\sigma \geq \sigma^* := \frac{3}{2} + k$ and $k \in \mathbb{N}$ denotes the parameter of the Wendland function. Let K be a compact set with $K \subset A(x_0) \setminus \{x_0\}$.*

There is a constant c^* such that for all grids $X_N := \{x_1, \ldots, x_N\} \subset K$ with fill distance h in K (i.e. $h := \sup_{x \in K} \inf_{x_j \in X_N} \|x - x_j\|$)

$$|v'(x) - V'(x)| \leq c^* h^\kappa \text{ holds for all } x \in K \,, \tag{16}$$

where $\kappa = \frac{1}{2}$ for $k = 1$ and $\kappa = 1$ for $k \geq 2$. $v \in C^{2k-1}(\mathbb{R}^2, \mathbb{R})$ is the approximation of Vsatisfying $V'(x) = -\|x - x_0\|^2$ or $V'(x) = -c$ with respect to the grid X_N using Wendland's function as radial basis function, i.e. $\varPsi(x) = \psi_{l,k}(\mu\|x\|)$, $\mu > 0$ and $l := k + 2$.

The estimate (16) in Theorem 3 implies $v'(x) \leq V'(x) + c^* h^\kappa \leq -\|x - x_0\|^2 + c^* h^\kappa$ for $V'(x) = -\|x - x_0\|^2$. If the grid is dense enough (i.e. the fill distance h is small enough), then $v'(x) < 0$ holds except for a small neighborhood of x_0. The same is true for the approximation of the function $V'(x) = -c$ since this function is not defined at $x = x_0$. This means, that we have to solve the problem differently near x_0: here we consider some local Lyapunov function, e.g. the Lyapunov function of Sect. 2.2, in a neighborhood of x_0. Details of the method can be found in [8] and [4]. An application of the method to the elbow joint is given in [3].

Note that the error estimate of this method requires a certain smoothness of the function F. In particular, the minimal requirement is $F \in C^3(\mathbb{R}^2, \mathbb{R})$ with the parameter of Wendland's function $k = 1$. However, the Hill-function is in general not C^3 at zero, the turning point from concentric to excentric behaviour. The Hill-function is assumed to be of the form $H(v) = \frac{c}{v+b} - a$ for $v \geq 0$ (concentric part, cf. (5)). It can be modelled in a smooth way for $v < 0$: for $H \in C^2$ cf. (6), and for $H \in C^3$ cf. [3]. Even if one uses a Hill-function model which is only C^1 or C^2 at zero, the method works in examples, but the error estimates cannot be proved, cf. [3] for examples and further discussion.

3 Periodic Systems

In this section we study periodic movements which lead to time-periodic differential equations. We define the basin of attraction for periodic orbits (Sect. 3.1). In order to check their stability one can use Floquet theory (Sect. 3.2). To find a subset of their basin of attraction one seeks to find a Lyapunov function, e.g. one of the above Lyapunov functions of an adjacent autonomous system (Sect. 3.3), which can be extended by radial basis functions (Sect. 3.4). An alternative method to the use of Lyapunov functions is Borg's criterion, where the exact position of the periodic movement is not required (Sect. 3.5), cf. [6]. Finally, Borg's method and Floquet theory are applied to periodic movements of the human knee (Sect. 3.6).

3.1 Basin of Attraction and Lyapunov Functions

We consider the time-periodic system

$$\dot{x} = F(t, x) , \tag{17}$$

where $F(t + T, x) = F(t, x)$ for all $t \geq 0$ and $T > 0$ is minimal with this property. The simplest solution of (17) is a periodic orbit $\Omega = \{(t, \tilde{x}(t)) \in S^1_T \times \mathbb{R}^2\}$, where $\tilde{x}(t)$ is a solution of (17) with $\tilde{x}(t + T) = \tilde{x}(t)$ for all $t \geq 0$ and S^1_T denotes the circle of radius T. The phase space is now the cylinder $S^1_T \times \mathbb{R}^2$.

Definition 2. *The basin of attraction $A(\Omega)$ of the periodic orbit $\Omega = \{\tilde{x}(t) \in S^1_T \times \mathbb{R}^2\}$, where $\tilde{x}(t)$ is a solution of the time-periodic ordinary differential equation $\dot{x} = F(t, x)$ is defined by*

$$A(\Omega) = \{(\tau, \xi) \in S^1_T \times \mathbb{R}^2 \mid \|x(t) - \tilde{x}(t)\| \overset{t\to\infty}{\longrightarrow} 0\} ,$$

where $x(t)$ denotes the solution of $\dot{x} = F(t, x)$ with initial value $x(\tau) = \xi$.

The basin of attraction can again be calculated using a Lyapunov function $v \in C^1(S^1_T \times \mathbb{R}^2, \mathbb{R})$. A Lyapunov function is still a function which is decreasing along solutions, i.e. a function with negative orbital derivative. Note, however, that the orbital derivative $v'(\tau, \xi)$ has now the following expression:

$$\begin{aligned} v'(\tau,\xi) &= \frac{d}{dt} v(t,x(t))\bigg|_{t=\tau} \\ &= \langle \nabla_x v(t,x(t)), \dot{x}(t) \rangle \bigg|_{t=\tau} + v_t(\tau,\xi) \\ &\overset{(17)}{=} \langle \nabla_x v(\tau,\xi), F(\tau,\xi) \rangle + v_t(\tau,\xi)\ . \end{aligned}$$

If $v'(\tau, \xi) < 0$, then level sets of v provide a tool to determine a subset of the basin of attraction. Note that the following theorem, similarly to the corresponding Theorem 1 in the autonomous case, does not make any assumptions about the stability of $\tilde{x}(t)$.

Theorem 4 (Lyapunov function, periodic case). *Let $\Omega = \{(t, \tilde{x}(t)) \in S^1_T \times \mathbb{R}^2\}$ be a periodic orbit of $\dot{x} = F(t, x)$. Let $v \in C^1(S^1_T \times \mathbb{R}^2, \mathbb{R})$ be a function and $K \subset S^1_T \times \mathbb{R}^2$ be a compact set with neighborhood B such that*

1. *$\Omega \subset \overset{\circ}{K}$,*
2. *$v'(t,x) = \langle \nabla_x v(t,x), F(t,x) \rangle + v_t(t,x) < 0$ holds for all $x \in K \setminus \Omega$,*
3. *$K = \{(t,x) \in B \mid v(t,x) \leq R\}$ with a constant $R \in \mathbb{R}$.*

Then $K \subset A(\Omega)$.

3.2 Stability and Floquet Theory

We perform a transformation of the system (17) which transforms the periodic solution $\tilde{x}(t)$ to the zero solution, i.e. $(t, y) := (t, x - \tilde{x}(t))$, and consider the transformed system

$$\dot{y} = G(t, y)\ , \tag{18}$$

where $G(t, y) = F(t, y + \tilde{x}(t)) - \dot{\tilde{x}}(t)$. $y(t) = 0$ is the (periodic) solution of (18) which corresponds to the periodic solution $\tilde{x}(t)$ of (17), and its stability is the same as the stability of $\tilde{x}(t)$ with respect to (17).

In order to study the stability, we consider again the linearization of (18) near the periodic solution. However, the stability cannot be checked as easily as in the autonomous case; one has to use Floquet theory.

Let us first consider a linear system of the form

$$\dot{y} = G(t)y\ , \tag{19}$$

where $G(t + T) = G(t)$ is a (2×2)-matrix for each t.

Theorem 5 (Floquet). *Each fundamental matrix $X(t)$ of (19) has a representation of the form*

$$X(t) = P(t)e^{Bt}$$

where $P(t+T) = P(t)$ is a t-periodic, (2×2)-matrix-valued function and B is a (2×2)-matrix. The eigenvalues of B are called Floquet exponents.

If the real part of all Floquet exponents is negative, then the zero solution of (19) is asymptotically stable.

Hence, the Floquet exponents play a similar role for the determination of the asymptotic stability as the eigenvalues of the Jacobian matrix evaluated at the equilibrium in the autonomous case. Now we return to the nonlinear problem $\dot{y} = G(t, y)$.

Theorem 6 (Stability, periodic case). *Consider (18) and assume that $G(t, y) = G(t)y + H(t, y)$, where $\lim_{y \to 0} \frac{\|H(t,y)\|}{\|y\|} = 0$ for all $t \in S_T^1$. If the real part of all Floquet exponents of the linearized equation $\dot{y} = G(t)y$ is negative, then the zero solution of (18) is asymptotically stable.*

Hence, in order to determine the stability one has to find the linearization of the system, i.e. $G(t)$, and in the next step one has to calculate the matrix B and its eigenvalues for the linearized system $\dot{y} = G(t)y$. This is done numerically for the example of two periodic knee movements, cf. Sect. 3.6 and Table 1.

3.3 Linearization of an Adjacent Autonomous System

We seek to construct a Lyapunov function for the time-periodic system $\dot{y} = G(t, y)$. In most examples the determination of the linearized system $\dot{y} = G(t)y$ and thus also the local Lyapunov function can only be obtained numerically. In this section, however, we consider the special situation that the time-periodic system is "near" to an autonomous system and we use the local Lyapunov function of the autonomous function as local Lyapunov function of the periodic system. More precisely, we assume in this section that there is a parameter $\lambda \in \mathbb{R}$ such that

$$\dot{y} = G(t, y) = \tilde{G}(t, y, \lambda), \text{ where } \tilde{G}(t, y, 0) \text{ is autonomous (independent of } t)$$

and $|\lambda|$ is small; the precise conditions are summarized in Proposition 1.

In Sect. 2.2 we have used a Lyapunov function of the linearized system which turned out to be a local Lyapunov function also for the nonlinear system. The idea for time-periodic systems in this section is to consider the adjacent autonomous system and then to use the Lyapunov function V for the linearized autonomous system. The following proposition, cf. e.g. [2], shows that this local Lyapunov function V is a Lyapunov function for the original time-periodic system in a neighborhood of the zero solution, provided that $|\lambda|$ is small enough.

Proposition 1. *Consider the system*

$$\dot{y} = a(y) + b(t, y, \lambda) \tag{20}$$

where $a \in C^1(\mathbb{R}^2, \mathbb{R}^2)$, $b \in C^1(S^1_T \times \mathbb{R}^2 \times \mathbb{R}, \mathbb{R}^2)$ *and the following properties hold:*

1. $a(0) = 0$,
2. $b(t, 0, \lambda) = 0$ *for all* $t \in S^1_T$ *and* $\lambda \in \mathbb{R}$,
3. $b(t, y, 0) = 0$ *for all* $t \in S^1_T$ *and* $y \in \mathbb{R}^2$.

Then $y = 0$ *is a solution of (20) for all* $\lambda \in \mathbb{R}$.

Let $V(t, y) = V(y) = y^T C y$ *be the local Lyapunov function of the autonomous linearized system* $\dot{y} = D_y a(0) y$ *according to Lemma 1. If* $|\lambda|$ *is small enough, then* $V'(t, y) < 0$ *holds for all* $(t, y) \in S^1_T \times \mathbb{R}^2$ *with* $0 < \|y\| \leq \delta$, *where* $\delta = \delta(\lambda)$.

Conditions 1 and 2 ensure that 0 is a solution for all $\lambda \in \mathbb{R}$ and Condition 3 shows that for $\lambda = 0$ the system is autonomous. Note that this Lyapunov function is independent of t and thus level sets of V are cylinders in the space $S^1_T \times \mathbb{R}^2$. Since this function is only a local Lyapunov function, i.e. the orbital derivative is negative only in a small neighborhood $\|y\| \leq \delta$ of the zero solution, we seek to find a Lyapunov function for a larger set using radial basis functions.

3.4 Approximation via Radial Basis Functions

As in Sect. 2.4 we seek to construct a Lyapunov function for the zero solution $\Omega := \{(t, 0) \mid t \in S^1_T\}$ by approximation via radial basis functions, cf. [2]. In order to use a similar approach as in Sect. 2.4 we add a differential equation, namely $\dot{t} = 1$, and study the following (autonomous) three-dimensional problem

$$\begin{cases} \dot{t} &= 1 \\ \dot{y} &= G(t, y). \end{cases}$$

Note that the periodicity is reflected by $(t, y) \in S^1_T \times \mathbb{R}^2$.

Similarly to Sect. 2.4 we consider Lyapunov functions with certain negative orbital derivatives, namely $V'(t, y) = -\|y\|^2$ defined for $(t, y) \in A(\Omega)$, which is a linear partial differential equation.

The method seeks to find an approximate solution v of this partial differential equation which is close enough to the solution V such that the orbital derivative v' is negative. Then v is itself a Lyapunov function and can be used to determine a subset of the basin of attraction.

Fix a radial basis function $\Psi(t, y)$ where $(t, y) \in \mathbb{R}^3$ with compact support, e.g. Wendland's function cf. [14], and a grid $X_N = \{\eta_1, \ldots, \eta_N\}$ with $\eta_i = (t_i, y_i) \in [0, T) \times \mathbb{R}^2$. The ansatz for the approximating function $v(t, y)$ is then

$$\sum_{m\in\mathbb{Z}}\sum_{j=1}^{N}\alpha_j\left\langle \nabla_{(\tau,z)}\Psi(t-\tau,y-z),\begin{pmatrix}1\\ G(t_j+mT,y_j)\end{pmatrix}\right\rangle\Bigg|_{(\tau,z)=(t_j+mT,y_j)}$$

Note that the summation over m is zero except for finitely many m due to the compact support of Ψ and reflects the fact that $t\in S^1_T$. In particular, by this ansatz $v(t+T,y)=v(t,y)$ holds for all $t\geq 0$.

The coefficients α_j are chosen such that v satisfies the equation on the grid X_N, i.e. $v'(t_j,y_j)=V'(t_j,y_j)=-\|y_j\|^2$ holds for all $j=1,\ldots,N$. The explicit calculation of α_j is achieved solving a system of linear equations.

Again, we have to solve the problem differently near $y=0$: here we consider some local Lyapunov function, e.g. the Lyapunov function of Sect. 3.3, in a neighborhood of 0. Details of the method and an application to the elbow joint are given in [2].

Note that also in the time-periodic case a modification of the method such that also the function values of v are prescribed on a level set of the local Lyapunov function or on $(t,y)=(t,0)$ improve the results. The ansatz is then altered to a mixed ansatz, cf. [2].

An application to biomechanics is also found in [2]. We consider a person who walks and assume that the walking induces a displacement of the shoulder and the upper arm of the form $\lambda\sin\nu t$. Hence, the acceleration acting on the system corresponding to an outer force is given by $a(t)=-\lambda\nu^2\sin\nu t$. Hence the equation of motion reads, cf. (8)

$$\begin{cases}\dot\beta=\omega\\ \dot\omega=\frac{1}{J}\left[\left(\frac{1}{2}m_u+m_l\right)\cdot(g-\lambda\nu^2\sin\nu t)\cdot l\cdot\sin\beta\right.\\ \qquad\qquad +E_{ext}(t)H_{ext}[h_{ext}(\beta)\cdot\omega]\,h_{ext}(\beta)\\ \qquad\qquad +E_{flex}(t)f_{l_{bic}}(\beta)H_{flex}[h_{bic}(\beta)\cdot\omega]\,h_{bic}(\beta)\\ \qquad\qquad \left.+E_{flex}(t)f_{l_{brach}}(\beta)H_{flex}[h_{brach}(\beta)\cdot\omega]\,h_{brach}(\beta)\right]\end{cases}$$

We are interested in a solution where the position of the hand is constant relatively to the ground. This periodic solution is given by $\beta^*(t)=\arccos(-\frac{\lambda}{l}\sin\nu t)$. The example studied in [2] assumes a constant flexor activation $E_{ext}=0.7$. The activation $E_{flex}(t)$ is chosen such that $\beta^*(t)$ is a solution of the system. In this example only a small amplitude $\lambda=0.5$ cm is considered, so that the construction of the local Lyapunov function as in Sect. 3.3 succeeds. The other parameters are $\nu=2\pi$ and $m_l=5$ kg.

3.5 Borg's method

In this section we apply a generalization of Borg's method, cf. [1], to the problem of the determination of the basin of attraction of a periodic orbit of a general time-periodic system (17), cf. [6]. In contrast to the method of Lyapunov functions, the knowledge of the position of the periodic orbit $\tilde{x}(t)$ is not required. However, we will see, that it can be helpful also for this method.

Borg's condition means that two adjacent solutions approach each other with respect to a Riemannian metric. This can easily be checked by the sign of a certain function $L_M(t,x)$, cf. (21) of Theorem 7.

Definition 3. *The matrix-valued function $M \in C^1(S^1_T \times \mathbb{R}^2, \mathbb{R}^{2\times 2})$ will be called a* Riemannian metric*, if $M(t,x)$ is a symmetric and positive definite matrix for each $(t,x) \in S^1_T \times \mathbb{R}^2$.*

The Riemannian metric $M(t,x)$ defines a point-dependent scalar product through $\langle v,w\rangle_{(t,x)} = v^T M(t,x)w$. The usual Euclidean metric is obtained by $M(t,x) = I$.

A set is called positively invariant, if solutions starting in this set, remain in the set for all positive times.

Definition 4. *A set $K \subset S^1_T \times \mathbb{R}^2$ is called positively invariant if for all $(\tau,\xi) \in K$, we have $(t,x(t)) \in K$ for all $t \geq 0$, where $x(t)$ denotes the solution of the initial value problem $\dot{x} = F(t,x)$, $x(\tau) = \xi$.*

Theorem 7. *(cf. [6]) Consider the equation $\dot{x} = F(t,x)$, where $F \in C^1(S^1_T \times \mathbb{R}^2, \mathbb{R}^2)$. Let $\emptyset \neq K \subset S^1_T \times \mathbb{R}^2$ be a connected, compact and positively invariant set. Let M be a Riemannian metric in the sense of Definition 3. Moreover, assume $L_M(t,x) < 0$ for all $(t,x) \in K$, where*

$$L_M(t,x) := \max_{w\in\mathbb{R}^2, w^T M(t,x)w=1} L_M(t,x;w) \tag{21}$$

$$L_M(t,x;w) := w^T \left[M(t,x)D_xF(t,x) + \frac{1}{2}M'(t,x)\right] w, \tag{22}$$

and $M'(t,x)$ denotes the matrix with entries

$$m_{ij} = \frac{\partial M_{ij}(t,x)}{\partial t} + \sum_{k=1}^{2} \frac{\partial M_{ij}(t,x)}{\partial x_k} F_k(t,x)$$

which is also the orbital derivative of $M(t,x)$.

Then there exists one and only one periodic orbit $\Omega \subset K$, which is exponentially asymptotically stable. Moreover, for its basin of attraction $K \subset A(\Omega)$ holds, and the largest real part $-\nu_0$ of all Floquet exponents of Ω satisfies

$$-\nu_0 \leq -\nu := \max_{(t,x)\in K} L_M(t,x)\ .$$

For the proof one considers two solutions $x(t)$ and $y(t)$ with adjacent initial values and defines their time-dependent distance with respect to the Riemannian metric $A(t) := \left[(y(t)-x(t))^T M(t,x(t))(y(t)-x(t))\right]^{1/2}$. Since $L_M(t,x) < 0$, the solutions approach each other and the distance $A(t)$ decreases exponentially. Thus, all solutions in K have the same ω-limit set characterizing the long-time behaviour, since K is connected. The study of a Poincaré-map shows that this ω-limit set is a periodic orbit.

The application of the method poses similar problems as in the case of Lyapunov functions: it is known that for each exponentially asymptotically stable periodic orbit there exists a Riemannian metric such that (21) holds, cf. [6], but there is no general construction method available. Moreover, we have to find a positively invariant set K.

Assume now again that the periodic orbit is known, which is the case in many biomechanical applications. There are two cases, where [6] gives suggestions for a Riemannian metric: (i) if $F(t,x)$ is "near" to an autonomous system – then one can use the metric of the autonomous system; this is similar to the approach of Sect. 3.3. (ii) if the eigenvalues of $D_xF(t,\tilde{x}(t))$ are real, distinct and negative for all $t \in S^1_T$. In this case, denote by $S(t)$ a smooth matrix-valued function on S^1_T, such that its columns are eigenvectors of $D_xF(t,\tilde{x}(t))$ for each t. Note that multiplication with $S(t)$ corresponds to a transformation such that the eigenvectors are the new coordinate axes. Then $M(t) = (S^{-1}(t))^T S^{-1}(t)$ is Riemannian metric. In this case (21) and (22) turn out to have the following form, cf. [6]

$$L_M(t,x) = \max_{w \in \mathbb{R}^2, \|w\|=1} w^T \left[S^{-1}(t) D_xF(t,x) S(t) + (S^{-1})'(t) S(t)\right] w \ , \quad (23)$$

where $(S^{-1})'(t)$ denotes the orbital derivative of $S(t)$. Formulas for the evaluation of the maximum in (23) are available, cf. [6]. Moreover, since the Riemannian metric only depends on t, we can easily determine a positively invariant set.

Proposition 2. *Let $\Omega = \{(t,\tilde{x}(t)) \in S^1_T \times \mathbb{R}^2\}$ be a periodic orbit and let $M(t)$ be a Riemannian metric which only depends on t. Moreover assume $L_M(t,x) < 0$ for all $(t,x) \in K^r$ where $r > 0$ and*

$$K^r := \{(t,x) \in S^1_T \times \mathbb{R}^2 \mid [x - \tilde{x}(t)]^T M(t) [x - \tilde{x}(t)] \le r^2\} \ .$$

Then Ω is exponentially asymptotically stable, and $K^r \subset A(\Omega)$ holds.

If L_M is negative for all points of the periodic orbit, then Theorem 7 implies that the periodic orbit is exponentially asymptotically stable and we obtain an upper bound for the largest real part of all Floquet exponents. Since then L_M is negative also in a neighborhood of the periodic orbit, we can determine a subset of its basin of attraction using Proposition 2.

3.6 Example: Periodic Movements of the Knee

For the example of a human knee-joint, cf. Sect. 1.2, we consider the periodic solution

$$\tilde{x}(t) := x_0 + \Delta_x \sin(t) \ ,$$

where $x_0 := 0.77$ and $\Delta_x := 0.07$. With $\tilde{v}(t) := \dot{\tilde{x}}(t)$ and $\tilde{a}(t) := \ddot{\tilde{x}}(t)$ we have the following formula for the extensor activation $E_{ext}(t)$ as a function of the (given) flexor activation $E_{flex}(t)$

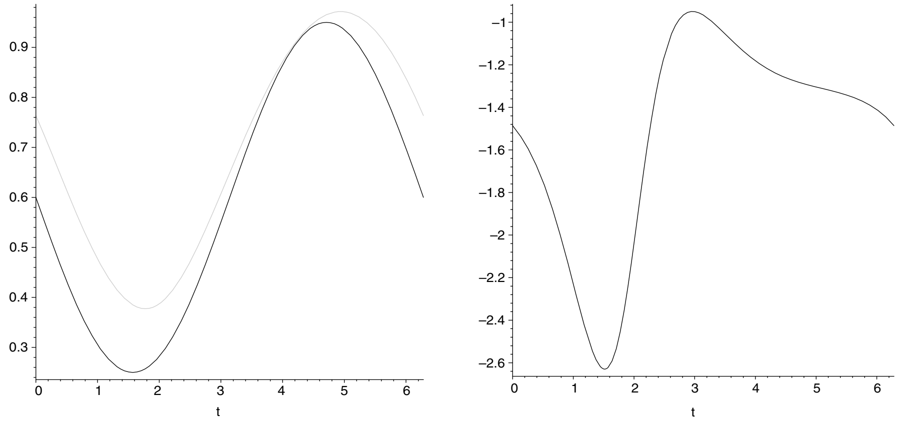


Fig. 1. (i) Uniform activation. *Left*: the activation functions $E_{flex}(t)$ (prescribed sine-function, black) and $E_{ext}(t)$ (calculated using (24), grey). *Right*: the function $L_M(t, \tilde{x}(t))$. The maximum value is -0.95

$$E_{ext}(t) = \frac{mg + \tilde{a}(t) - E_{flex}(t) f_{iso_{flex}} h_{flex}(\tilde{x}(t))}{H[h_{ext}(\tilde{x}(t))\tilde{v}(t)] h_{ext}(\tilde{x}(t))} \tag{24}$$

to ensure that $\tilde{x}(t)$ is a solution of (11).

We consider a flexor activation of the following form:

$$E_{flex}(t) := -d_1 \sin(t + d_2 \cos(t)) + d_3, \text{ where}$$

- Example (i): $d_1 = 0.35$, $d_2 = 0$ and $d_3 = 0.6$ (cf. Fig. 1 – uniform activation)
- Example (ii): $d_1 = 0.3$, $d_2 = 1.2$ and $d_3 = 0.6$ (cf. Fig. 2 – non-uniform activation).

The Jacobian of the right-hand side at time t is given by

$$\begin{aligned} D_{(x,v)} F(t, \tilde{x}(t), \tilde{v}(t)) &= \begin{pmatrix} 0 & 1 \\ f_x(t, \tilde{x}(t), \tilde{v}(t)) & f_v(t, \tilde{x}(t), \tilde{v}(t)) \end{pmatrix} \\ &=: \begin{pmatrix} 0 & 1 \\ f_x(t) & f_v(t) \end{pmatrix} \end{aligned}$$

and its eigenvalues at time t are

$$\lambda_{1,2}(t) = \frac{1}{2} \left(f_v(t) \pm \sqrt{f_v(t)^2 + 4 f_x(t)} \right).$$

The eigenvalues are real, distinct and negative, if and only if

$$\left\{ \begin{array}{r} f_v(t) < 0 \\ f_x(t) < 0 \\ f_v(t)^2 + 4 f_x(t) > 0 \end{array} \right.$$

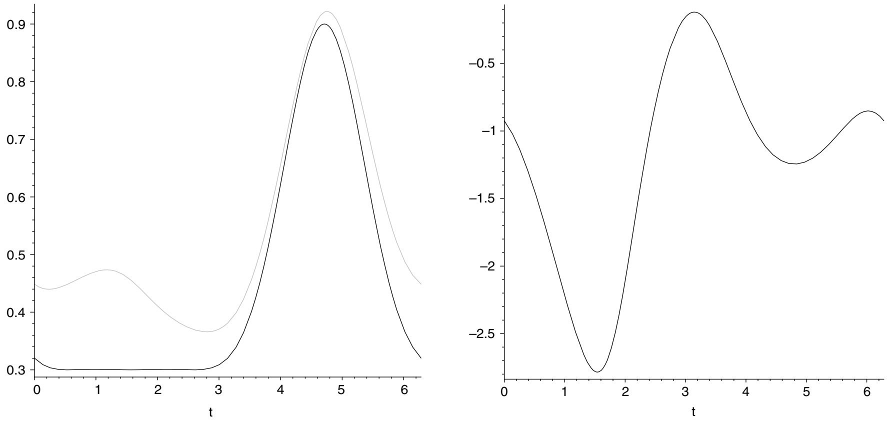


Fig. 2. (ii) Non-uniform activation. *Left*: the activation functions $E_{flex}(t)$ (prescribed, black) and $E_{ext}(t)$ (calculated using (24), grey). *Right*: the function $L_M(t, \tilde{x}(t))$. The maximum value is -0.12

holds for all t. Note that $f_v(t) < 0$ holds for all t because of the property $H_u(u) < 0$ of the Hill-function.

For both examples (i) and (ii) these conditions are satisfied for all t. Figures 1 and 2 show the activation functions and the functions $L_M(t, \tilde{x}(t))$, which turn out to be negative for all times. Hence, the periodic orbit is asymptotically stable for both examples by Theorem 7. The uniform activation, example (i), gives an upper bound of -0.95 for the largest real part of the Floquet exponents, whereas the example (ii) gives an upper bound of -0.12.

We also calculate the Floquet exponents numerically, cf. Sect. 3.2, which takes a considerably longer time, but gives a sharper result on the largest real part of the Floquet exponents: again, the uniform activation (i) leads to a more negative Floquet exponent than the activation in example (ii), cf. Table 1. Thus, a uniform activation seems to be favourable concerning the stability.

Table 1. The largest real part of the Floquet exponents for the two examples. Borg's method A. provides an upper bound for the largest real part of the Floquet exponents. The numerical calculation B. gives an approximation of the exact value of the largest real part of all Floquet exponents. However, it requires a large number of time-steps corresponding to a division of the interval $[0, 2\pi]$ to calculate the matrix $P(t)$ and B, cf. Sect. 3.2

Example	A. Borg's method	B. numerical calculation	steps
(i) Uniform	-0.95	-3.43	37670
(ii) Non-uniform	-0.12	-1.60	55972

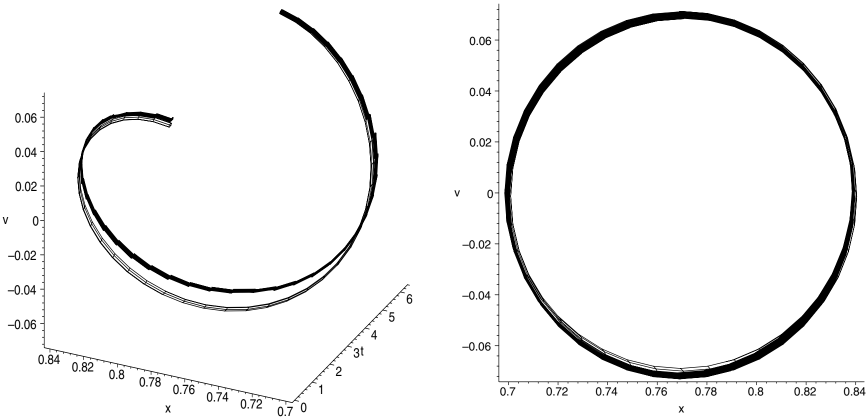


Fig. 3. A subset of the basin of attraction for example (i), uniform activation. *Left*: (t, x, v)-space. *Right*: projection on the (x, v)-plane

Using Proposition 2 we calculate subsets of the basin of attraction for the uniform activation, example (i), shown in Fig. 3 in the (t, x, v)-space and projected to the (x, v)-space. The subset obtained with this method, however, is very small. A numerical calculation of solutions with different initial conditions shows that the real basin of attraction is much larger and contains the whole part of the phase space shown in Fig. 3. The advantage of Borg's method, however, is that the conditions can be checked easily and fast, whereas the calculation of many solutions with different initial conditions requires much more time.

4 Conclusion

In this paper we presented mathematical models of the human elbow and knee joint. We discussed different methods to analyze the stability and the basin of attraction of stationary and periodic movements. These mathematical methods serve as tools to answer biological questions as posed in the introduction. The properties of the muscles and the geometry of the joints which are responsible for the stability are the positive slope of the force-length function (elbow joint, cf. [7]) and the moving center of rotation (knee joint, cf. [12]). A high co-activation also stabilizes the system, cf. [7] and [5]. The stability of stationary movements depends on the position of the joint angle: for the elbow small angles are stable, whereas large angles are unstable. The dependency on the angle and the co-activation is also reflected in the size of the basin of attraction, cf. [13]. For periodic movements, general answers to these questions are more difficult to obtain, but we hope that the methods presented in this paper serve to analyze periodic movements in more detail in the future. For an analysis of the self-stabilizing properties of biological systems with emphasis on the biological reasons and implications, cf. also [13].

References

[1] Borg G. (1960). A condition for the existence of orbitally stable solutions of dynamical systems. Kungl. Tekn. Högsk. Handl. 153:1–12

[2] Chen J. (2004). Stability of the movements of the elbow joint. MSc Thesis, Technical University of Munich

[3] Ernst M. (2005). Analyse der Stabilität und der Einzugsbereiche nichtlinearer dynamischer Systeme am Beispiel eines Armmodells. Diploma Thesis, University of Jena

[4] Giesl P. (2004). Approximation of domains of attraction and Lyapunov functions using radial basis functions. In: Proceedings of the NOLCOS 2004 conference Stuttgart, Germany vol. II, 865–870

[5] Giesl P. (2003). The optimal elbow angle for acrobatics – stability analysis of the elbow with a load. In: PAMM Proc. Appl. Math. Mech. 3:68–71

[6] Giesl P. (2004). Z. Anal. Anwendungen 23(3):547–576

[7] Giesl P., Meisel D., Scheurle J., Wagner H. (2004). Journal of Theoretical Biology 228(1):115–125

[8] Giesl P. (2004). Construction of Lyapunov functions using radial basis functions. Habilitation Thesis, Technical University of Munich

[9] Giesl P., Wagner H. (submitted) Lyapunov functions and the basin of attraction for a single-joint muscle-skeletal model

[10] Hill A. (1938). Proc Royal Soc. (B) London 126:136–195

[11] Murray W., Buchanan T., Scott D. (2000). J. Biomech. 33:943–952

[12] Wagner H. (1999). Zur Stabilität der menschlichen Bewegung. PhD Thesis, J.W. Goethe-Universität, Frankfurt a.M.

[13] Wagner H., Giesl P. (2006). Self-stability in Biological Systems, this volume

[14] Wendland H. (1998). J. Approx. Theory 93:258–272

Task-Level Control of the Lateral Leg Spring Model of Cockroach Locomotion

J. Lee[1], A. Lamperski[2], J. Schmitt[3], and N. Cowan[1]

[1] Johns Hopkins University, Baltimore, MD
{jsl,ncowan}@jhu.edu
[2] California Institute of Technology, Pasadena, CA
andyl@cds.caltech.edu
[3] Oregon State University, Corvallis, OR
schmitjo@engr.orst.edu

Summary. The Lateral Leg Spring model (LLS) was developed by Schmitt and Holmes to model the horizontal-plane dynamics of a running cockroach. The model captures several salient features of real insect locomotion, and demonstrates that horizontal plane locomotion can be passively stabilized by a well-tuned mechanical system, thus requiring minimal neural reflexes. We propose two enhancements to the LLS model. First, we derive the dynamical equations for a more flexible placement of the center of pressure (COP), which enables the model to capture the phase relationship between the body orientation and center-of-mass (COM) heading in a simpler manner than previously possible. Second, we propose a reduced LLS "plant model" and biologically inspired control law that enables the model to follow along a virtual wall, much like antenna-based wall following in cockroaches.

1 Introduction

For decades, researchers have posited low-dimensional spring-mass models to describe the COM dynamics and ground reaction forces in a broad variety of running animals [2, 4, 9, 11, 12, 19]. In order to understand the complex body mechanics of running animals, they have simplified the problem by decoupling the mechanics into the sagittal and horizontal planes. For animals whose locomotion occurs primarily in the sagittal plane, the locomotion dynamics have been modeled as a spring-loaded inverted pendulum (SLIP) [2, 16, 26, 27]. Insects, whose motion occurs primarily in the horizontal plane, have dynamics that have been approximated by a lateral leg spring (LLS) model [23, 24]. Results of the LLS suggest that the mechanical structure of an insect may be used to produce stable periodic gaits when running at high speeds, without relying solely on proprioceptive reflexes and detailed neural feedback for stability.

The LLS models insect locomotion, specifically that of the cockroach *Blaberus discoidalis*. Cockroaches run using an alternating tripod gait [4]. Experiments have shown that the forces produced by this tripod of legs can be well represented by a single effective leg [10, 15]. Since the total mass of the legs of the insect is less than 6% of the total mass, the LLS model approximates each alternating tripod as a single massless, spring-loaded *virtual leg* that attaches to the midline of the body at a point called the center of pressure (COP). As illustrated in Fig. 1, the COP is offset from the center of mass (COM) by a displacement, d, where d may lie in front of the COM ($d > 0$) or behind the COM ($d < 0$). The model assumes that the foot pivots freely without slipping about its attachment to the ground, r_{foot}, and that the leg can rotate freely about the COP. This implies that no moments about the foot or COP can be generated, and forces will be applied to the body along the length of the leg. A full stride for the model consists of a left and right step phase. A step phase begins with the relaxed spring extended at an angle $\pm\beta_0$ with respect to the body centerline. The body moves forward, compressing and extending the elastic spring, until the spring returns to its original length, at which point the leg is lifted, the next leg is placed down, and the cycle repeats.

Changes in the foot placements between left and right step phases result in a hybrid dynamical system. Systems with piecewise-holonomic constraints such as these can display asymptotic stability [21]. For gaits encountered in the LLS model, periodic motions exhibit neutral eigendirections due to energy conservation and SE(2) invariance. Therefore, stability is partially asymptotic in the sense that perturbations to periodic orbits in the direction of the eigenvectors of conserved quantities and symmetries do not grow or decay, but simply result in movement to a different, stable gait. Gaits in the LLS model display partial asymptotic stability in the heading direction and angular velocity as a result of angular momentum trading between left and right step phases. The mechanical structure of the model therefore self-stabilizes the locomotion system [23]. If $d < 0$ then the gaits are asymptotically stable in heading and angular velocity, i.e. the body approaches straight trajectories if the trajectory begins in the basin of attraction for the stable periodic orbit. If $d = 0$, the periodic orbits exhibit neutral stability in angular velocity and asymptotic stability in heading. If $d > 0$, periodic orbits are unstable. To show stability, one takes Poincaré sections at the beginning of a full stride, and numerically approximates the fixed points and eigenvalues of the linearized return map.

While the energetically conservative fixed and moving center of pressure models of [23, 24] reproduce many salient features of the kinematics and forces exhibited experimentally by *Blaberus discoidalis*, detailed comparisons illuminate limitations of the LLS. In particular, the fixed COP models previously investigated consider only COPs on the fore-aft body axis, and consequently only produce sinusoidal variations in θ; in contrast, the animal produces cosinusoidal variations [28]. This is due to the fact that a fixed COP located behind the center of mass along the fore-aft body axis is only capable of producing a positive or negative moment, rather than a moment that changes sign during each step phase. Additionally, fore-aft and lateral force profile magnitudes are typically reversed in comparison to those observed experimentally. Allowing the leg attachment point to vary from fore to aft in the moving COP model serves to address the qualitative discrepancies in the moment and yawing profiles. However, while qualitatively correct yaw and moment profiles are produced by the model, quantitative comparisons reveal that the variations in each remain an order of magnitude smaller than those observed experimentally. An activated muscle model introduced by Schmitt and Holmes [25] attempts to correct the moment and yawing oscillations by introducing hip torques and muscle activation. While these authors obtained correct moment profiles in this manner, they are obtained at the expense of increased model complexity and inverted fore-aft force profiles.

The goal of this paper is to modify the LLS model to better match the actual cockroach, with as few parameters as possible, and to extend it to serve as a plant model for control. To compare our model to the previous LLS, we consider features salient to cockroach locomotion, such as stability, body motion kinematics, forces and moments, stride frequency, etc. For control, we use a biologically inspired antenna-like measurement [6, 7, 8], and show numerically that the closed-loop system dynamics asymptotically track a virtual wall in the environment. In addition, the controller maintains the LLS model's energy conserving nature.

2 Dynamics and Simulation of an Enhanced LLS Model

The goal of this research is to control the LLS model from step-to-step to achieve a locomotion objective such as following along a wall and avoiding obstacles in a planar environment. Using a controlled form of the LLS as a "plant model" may provide insights into our longer term objective of controlling a legged robot such as RHex [1], Sprawl [5], or Whegs [20]. It is known, for example, that RHex exhibits a dynamically similar gait in both the sagittal and horizontal planes to a cockroach. Toward that end, this section explores the effects of COP placement and movement on the steady-state dynamics of the LLS model. The goal is to uncover the simplest possible mechanism to match biological data, while still providing the possibility for control.

2.1 LLS Dynamics with 2-D COP Placement for a Left Step

We propose an alternative (or a simpler) solution to the moving COP; laterally offset the fixed COP (i.e. position the COP in the positive x-direction of the body frame $\{B\}$). This has a similar effect as the moving COP scheme; the leg generates a clockwise torque during the first half of a step, and an opposite

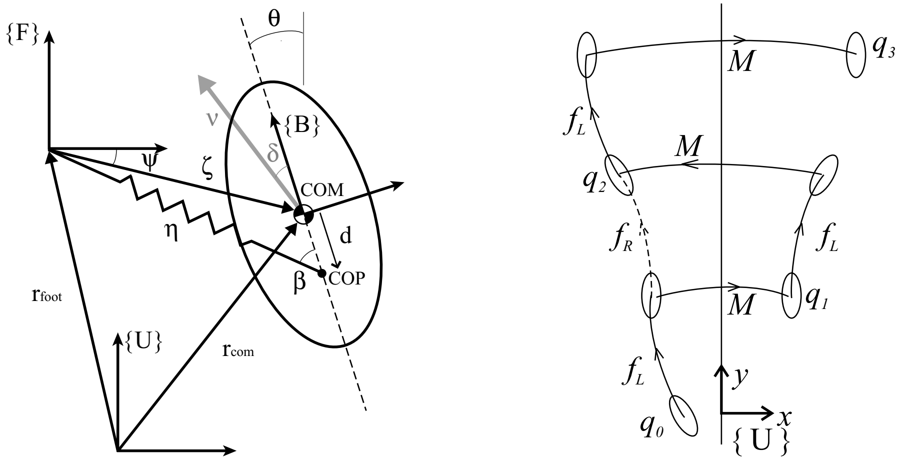


Fig. 1. *Left:* A schematic model of the LLS model, showing the coordinates used for expressing Hamilton's equations. *Right:* Illustration of multi-step dynamics. The dynamics of each left step phase are given by (4). A right step can be achieved by first flipping the states about the y-axis, integrating the left step map, and then flipping back. Breaking this chain in the correct place leads to a single "integrate and flip" return map, $f(q) := Mf_L(q)$, that will simplify controller design

torque during the last half, assuming the body angle, θ, is greater than zero at the start of a left-leg step.

In order to validate our alternative solution, we represent the position of the COP during the left-leg step as:

$$\begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} b_1 + c_1(\psi - \theta) \\ b_2 + c_2(\psi - \theta) \end{bmatrix} , \tag{1}$$

where d_1 and d_2 are along the x and y-axis of the LLS body frame $\{B\}$, and ψ and θ are shown in Fig. 1. In this representation, we allow the COP to be either fixed ($c_1 = c_2 = 0$) or moving ($c_1 \neq 0$ or $c_2 \neq 0$) from any offset (b_1 and b_2) in the body frame $\{B\}$. This freedom allows us to test different COP placement protocols, including the case where the COP moves backwards while offset to the side [28]. This representation implicitly assumes that during the next right-leg step, the COP position will be mirrored about the y-axis about the body frame $\{B\}$. If $d_1 = 0$, then we have the equation introduced in [23].

Consider the generalized coordinates $r = (\zeta, \psi, \theta)$, as depicted in Fig. 1. For the left step phase, the Hamiltonian of the LLS system implemented with a linear spring is

$$H = \frac{p_\zeta^2}{2m} + \frac{p_\psi^2}{2m\zeta^2} + \frac{p_\theta^2}{2I} + \frac{k(\eta - l_0)^2}{2} \tag{2}$$

where ζ, ψ, k, l_0, I, and m denote the distance from the foot placement to the COM, the angle from the foot placement to the COM, the linear spring stiffness, the relaxed leg length, the moment of inertia, and the mass, respectively. The length of the leg is given in terms of the COP location by

$$\eta = \Big[b_1^2 + b_2^2 + \zeta^2 + \phi\big(2b_1c_1 + 2b_2c_2 + (c_1^2 + c_2^2)\phi\big) \\ + 2\zeta\big((b_1 + c_1\phi)\cos\phi + (b_2 + c_2\phi)\sin\phi\big)\Big]^{1/2} . \quad (3)$$

Hamilton's equations of motion with our new COP and the linear leg spring model are given by

$$\begin{aligned} \dot\zeta &= \frac{p_\zeta}{m}, & \dot p_\zeta &= \frac{p_\psi^2}{m\zeta^3} - \frac{k(\eta - l_0)}{\eta}\big(\zeta + (b_1 + c_1\phi)\cos\phi + (b_2 + c_2\phi)\sin\phi\big) , \\ \dot\psi &= \frac{p_\psi}{m\zeta^2}, & \dot p_\psi &= -\frac{k(\eta - l_0)}{\eta}\big(b_1c_1 + b_2c_2 + (c_1^2 + c_2^2)\phi \\ &&&\quad + \zeta(b_2 + c_1 + c_2\phi)\cos\phi - \zeta(b_1 - c_2 + c_1\phi)\sin\phi\big), \\ \dot\theta &= \frac{p_\theta}{I}, & \dot p_\theta &= -\dot p_\psi , \end{aligned} \quad (4)$$

where $\phi \triangleq \psi - \theta$. We assume when a step commences, the spring is uncompressed, $\eta = l_0$. Because the spring starts at and returns to rest length at step transitions, no step-to-step impacts dissipate energy, and thus energy is conserved in the LLS model.

2.2 Hybrid Step-to-Step Dynamics

The generalized coordinates $r = (\zeta, \psi, \theta)$ and their conjugate momenta, p_r, provide a convenient set of local coordinates for expressing the *within-step* Hamiltonian dynamics (4) of the LLS. However, they provide an inconvenient representation when considering the *step-to-step* dynamics because they depend on the frame $\{F\}$ that moves every step. As a remedy, we follow [23], and use $q = (s, g) \in \mathcal{S} \times \mathrm{SE}(2)$, where $s = (v, \delta, \dot\theta) \in \mathcal{S} \subset \mathbb{R}^3$ are the "internal" states, and $g \in \mathrm{SE}(2)$ is the pose. The speed, v, is the magnitude of the COM velocity, and the relative heading, δ, is the angular difference between the orientation, θ, and the angle of the COM velocity vector (see Fig. 1). The local coordinates (θ, x, y) parameterize SE(2) without singularities through the usual relationship,

$$g = \begin{bmatrix} \cos\theta & -\sin\theta & x \\ \sin\theta & \cos\theta & y \\ 0 & 0 & 1 \end{bmatrix} \quad (5)$$

(written as a homogeneous transformation matrix) so we conflate the two when convenient and often write $g = (\theta, x, y)$ in an abuse of notation. The dynamical equations can be recast using the state variables q, which we omit for simplicity of presentation. Instead, we consider the state q_i, $i = 0, 1, 2 \ldots$

as the discrete state, where $q_k = (s_k, g_k)$ corresponds to the state at the beginning of the k^{th} step. If $h \in \text{SE}(2)$ and $q = (s, g) \in \mathcal{S} \times \text{SE}(2)$ then we define the left action of SE(2) on $\mathcal{S} \times \text{SE}(2)$ by $hq = (s, hg)$, where hg is the group product on SE(2).

At the beginning of the k^{th} step (for k even), the leg is at rest length, $\eta = l_0$, and the leg touch-down angle starts at β_0 relative to the y-axis of the body frame $\{B\}$, $\beta = \beta_0$. This information, together with the state $q_k = (s_k, g_k)$, uniquely determines the initial conditions for integration of Hamilton's equations. When η again reaches the spring rest length l_0, the hybrid system transitions to the right step, as described below. The final values of (r, p_r) at the end of the k^{th} step uniquely determine the states $q_{k+1} = (s_{k+1}, g_{k+1})$, used to start the subsequent step. Thus, the left step dynamics map the state $f_L : q_k \mapsto q_{k+1}$ according to a simple change of variables into coordinates (r, p_r), followed by integration of Hamilton's equations. By inspection of Hamilton's equations (4), note that the left-step mapping is left invariant under rigid transformations of the initial condition, since the equations are not functions of (x, y), and θ never shows up without $-\psi$, both of which are with respect to the world frame. Hence, $f_L(s, hg) = hf_L(s, g)$. Note that this implies that $q_{k+1} = (s_{k+1}, g_{k+1}) = f_L(s, g_k) = g_k f_L(s, e)$, where $e \in \text{SE}(2)$ is the identity.

Let $\{A_k\}$ denote the location of the body frame at the beginning of the k^{th} step. In other words, g_k is the transformation from $\{A_k\}$ to the world frame $\{U\}$. For k odd, the right leg is down, and Hamilton's equations (4) are identical, so long as we express them in terms of a *left-handed* frame. We do this by taking a mirror image around the y-axis of frame $\{A_k\}$ at the beginning and end of the k^{th} step (k odd), to write down the right step map in terms of the left one. This can be expressed in terms of local coordinates $q = (v, \delta, \dot{\theta}, \theta, x, y)^T$ as first "flipping" $(\delta, \dot{\theta}, \theta, x)$, integrating the left step map, and then flipping back, namely

$$f_R(q) = M f_L\big(Mq), \text{ where } M = \text{diag}\{1, -1, -1, -1, -1, 1\} \ . \tag{6}$$

Note that $MM = I$. We chose to flip about the y-axis for notational simplicity, but in principle any left-handed frame would work. This mapping leaves the right step map left-invariant under SE(2).

For finding symmetric steady-state gaits, it will be convenient to define a special step-to-step return map that amounts to an "integrate and flip" (see Fig. 1, *Right*). For a complete stride that includes a left step and then a right step, the stride-to-stride mapping is given by $f_{L-R} = f_R \circ f_L$, namely

$$f_{L-R} : q \mapsto M f_L(M f_L(q)) = (f \circ f)(q), \quad \text{where} \quad f(q) := M f_L(q) \ . \tag{7}$$

This approach eliminates the need to distinguish between left and right steps for control purposes. Note, however, that f is *not* left-invariant, even though both f_L and f_R are left-invariant. The resulting state evolution is given simply by

$$q_{k+1} = f(q_k) \ , \tag{8}$$

keeping in mind that for odd steps, the value for q_k in this step-to-step formulation has already been "flipped".

2.3 Simulation Methods

We simulated the LLS model using Matlab and the convention discussed in Sect. 2.2; for every right-step, convert it to a left-leg step, simulate the within step dynamics, and then convert it back to a right-leg step. This enabled us to specify the COP position using (1) and integrate the equation of motion (4) without the explicit representations of a left or right step in the equations. We used Matlab's `ode45` with time varying step size to integrate the equations of motion. The integration terminated as soon as the compressed leg returned back to its relaxed length l_0. To specify a moving COP, we selected b_i and $d_i(kT)$, $i = 1, 2$ where $d_i(kT)$ denotes the COP position at the start of k-th step. To meet this restriction, c_i is allowed to vary at each step, although it shouldn't vary at an equilibrium point.[4]

We found the equilibrium point $q_0 = (v, \delta, \theta, \dot{\theta}, x)^T$ using the Levenberg-Marquardt method in Matlab's `fsolve` function. While fixing the state v to a desired value, the function minimized the error difference of a step, $f(q) - q$. We also found the stride-to-stride Jacobian, A_{stride}, and step-to-step Jacobian, A_{step}, about the equilibrium point using a central difference approximation. The i^{th} columns are given by $[A_{\text{stride}}]_i = (f_{L-R}(q + e_i\epsilon) - f_{L-R}(q - e_i\epsilon))/2\epsilon$ and $[A_{\text{step}}]_i = (f(q + e_i\epsilon) - f(q - e_i\epsilon))/2\epsilon$, where $\epsilon = 1 \times 10^{-6}$ and e_i is the i-th column of 5×5 identity matrix. In Sect. 3.2, we discuss the LLS stability from the eigenvalues of A_{stride}, while in Sect. 4, we use A_{step} to control the LLS model.

Unless otherwise noted, we used the following parameters and measurements of death-head cockroaches, *Blaberus discoidalis,* used in [19, 22, 24]: $m = 0.0025\,\text{kg}$, $I = 2.04 \times 10^{-7}\,\text{kg}\,\text{m}^2$. The choices for l_0, k, v, and β_0 were chosen to satisfy constraints on the stride length ($L_s = 0.02 - 0.025\,\text{m}$) and stride frequency ($f_s = 10\,\text{Hz}$), and generally fell in the ranges $k = 2.25 - 3.5\,\text{N}\,\text{m}^{-1}$, $l_0 = 0.008 - 0.015\,\text{m}$, $\beta_0 = 0.8 - 1.2\,\text{rad}$, $d_i = 0.002\,\text{m}$, and $v = 0.2 - 0.25\,\text{m/s}$.

3 Analysis of COP Placements

3.1 Effects of Various COP Placements

In order to match the LLS system with an actual cockroach data (Fig. 2), we need to understand the effects of b_i and c_i (or $d_i(kT)$) on the overall system.

[4] Instead, the values for b_i and c_i can be specified directly [24]. This causes $d_i(kT)$ to change depending on the quantity $(\psi(kT) - \theta(kT))$.

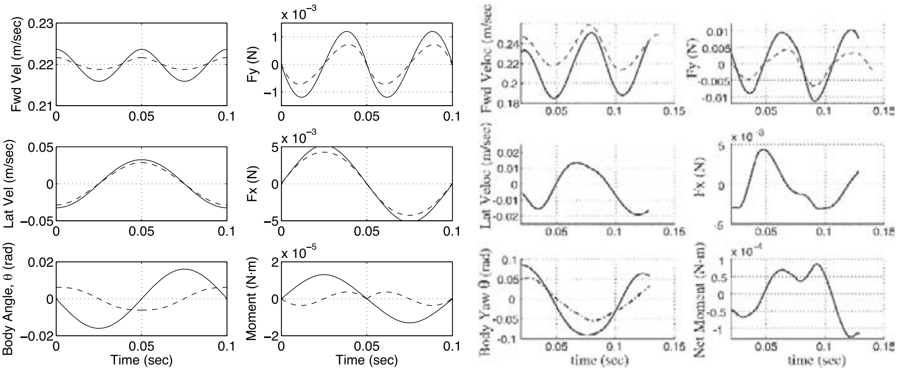


Fig. 2. *Left:* A stride of the original LLS model ($d_1 \equiv 0$) with a fixed COP (*solid*) and a moving COP (*dashed*). The used parameters for the fixed COP are: $v = 0.226$ cm/s, $k = 2.4$ N/m, $\beta_0 = 1$ rad, $l_0 = 0.0102$ m, $d_2 = -0.0025$ m; for the moving COP are: $v = 0.2235$ m/s, $k = 3.52$ N/m, $\beta_0 = 1.125$ rad, $l_0 = 0.0082$ m, $d_2 = 0.0025$ m $\rightarrow$ -0.0025 m. *Right:* Experimental measurements of *Blaberus discoidalis* from several sources, [14, 15, 18]; figure from [22]. (Notice, since the right figure doesn't start from $t = 0$, the stride period is roughly the same between the two figures)

To do so, our initial attempt is to consider various protocols for the COP placements:

(a) Increment d_1 while $d_2 = 0$;
(b) Increment d_2 while $d_1 = 0$;
(c) Increment the amplitude of a moving d_2 while $d_1 = 0$;
(d) Increment the offset of a moving d_2 while $d_1 = 0$;
(e) Increment d_1 while d_2 is moving;

Figure 3 illustrates these protocols schematically for a left step; for a right step, the COP path is mirrored about the body y-axis. For each protocol and their parameter increments, we found the corresponding equilibrium points and simulated a full stride (starting with a left step) from the equilibrium points. The results from the simulations are shown in Figs. 4, 5, and 6. For each incrementing parameter, we plotted the result using different shades of

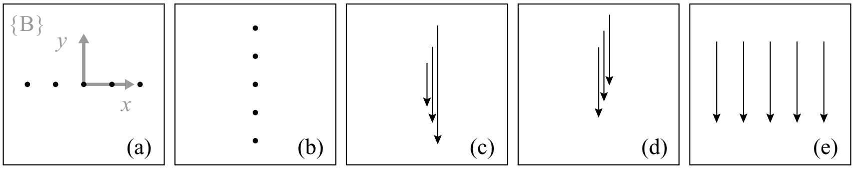
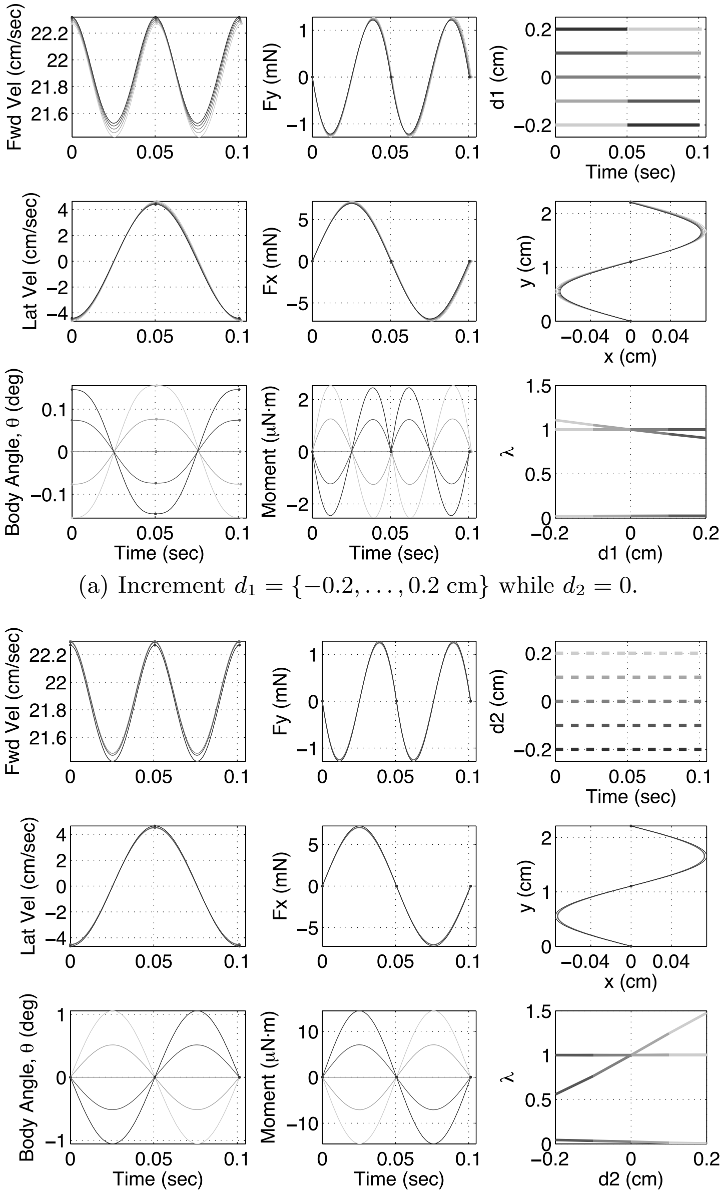


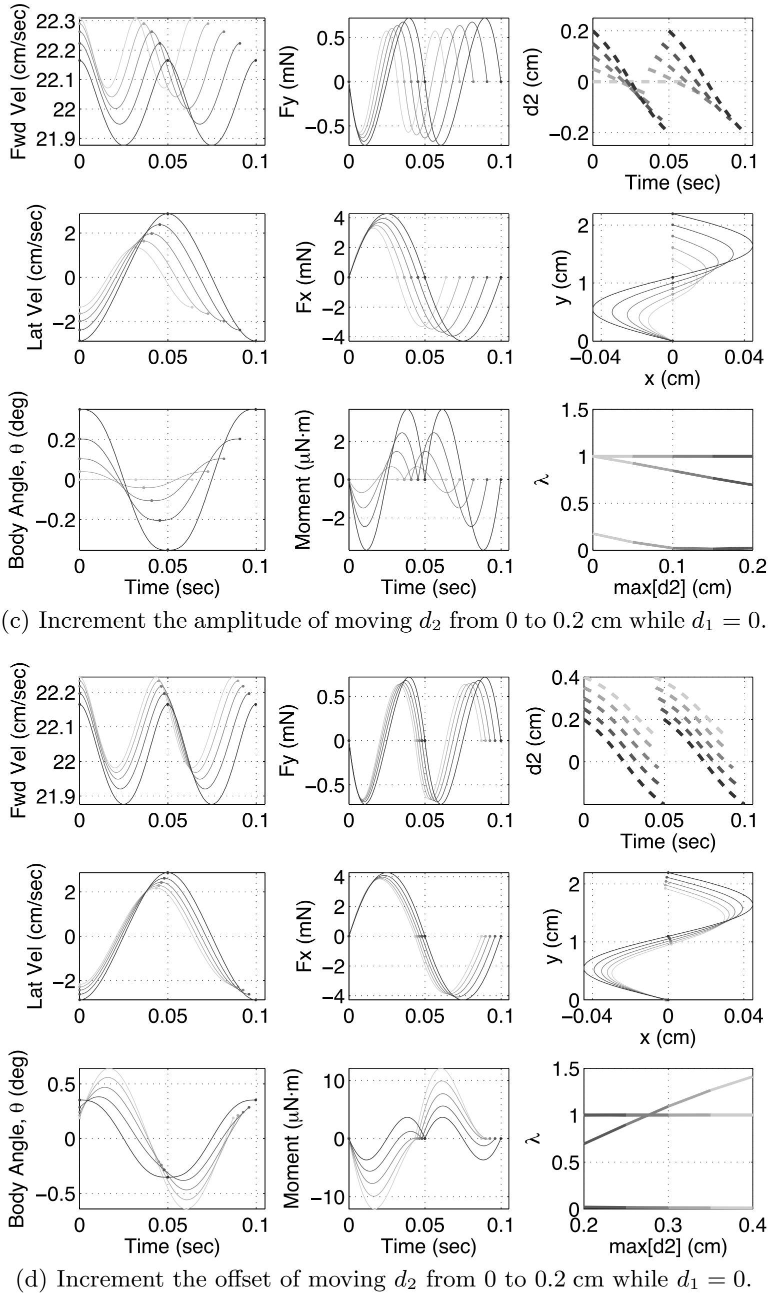
Fig. 3. COP placement protocols for a left step with respect to the body frame $\{B\}$. The *solid dots* indicate fixed COP positions; the *arrows* indicate the direction, magnitude, and offset of moving COPs



(a) Increment $d_1 = \{-0.2, \ldots, 0.2 \text{ cm}\}$ while $d_2 = 0$.

(b) Increment $d_2 = \{-0.2, \ldots, 0.2 \text{ cm}\}$ while $d_1 = 0$.

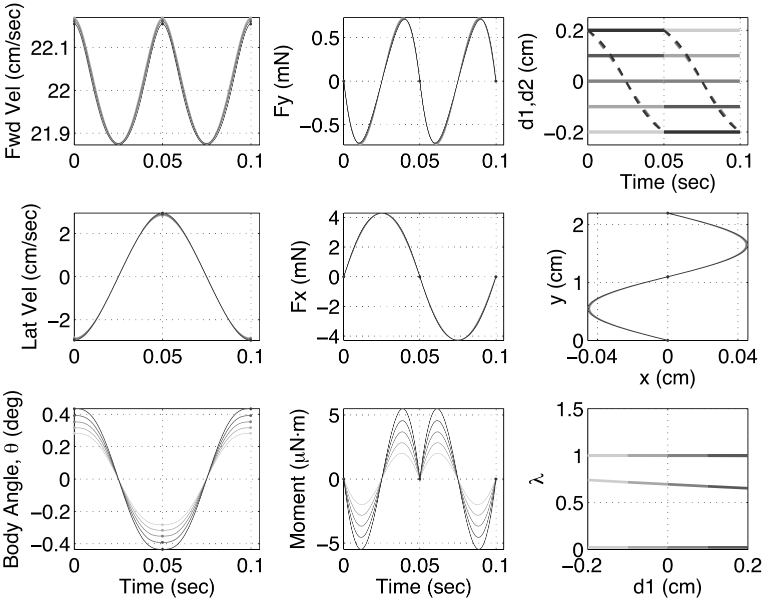
Fig. 4. See text for description of each COP protocol. The parameters used in this figure are: $m = 0.0025\,\text{kg}$, $I = 2.04 \times 10^{-7}\,\text{kg}\,\text{m}^2$, $k = 3.52\,\text{N}\,\text{m}^{-1}$, $v = 0.2275\,\text{m/s}$, $l_0 = 0.0127\,\text{m}$, $\beta_0 = 1.12\,\text{rad}$ (or $64.2\,^\circ$). Note, unlike Fig. 2, these figures have scaled units (e.g. cm and mN) for clarity



(c) Increment the amplitude of moving d_2 from 0 to 0.2 cm while $d_1 = 0$.

(d) Increment the offset of moving d_2 from 0 to 0.2 cm while $d_1 = 0$.

Fig. 5. The parameters used here are: $m = 0.0025\,\mathrm{kg},\ I = 2.04 \times 10^{-7}\,\mathrm{kg\,m^2},\ k = 3.52\,\mathrm{N\,m^{-1}},\ v = 0.2235\,\mathrm{m/s},\ l_0 = 0.0082\,\mathrm{m},\ \beta_0 = 1.125\,\mathrm{rad}$



(e) Increment $d_1 = \{-0.2, \ldots, 0.2 \text{ cm}\}$ (solid) while d_2 (dashed) is moving from 0.2 to −0.2 cm.

Fig. 6. The parameters used here are: $m = 0.0025\,\mathrm{kg},\ I = 2.04 \times 10^{-7}\,\mathrm{kg\,m^2},\ k = 3.52\,\mathrm{N\,m^{-1}},\ v = 0.2235\,\mathrm{m/s},\ l_0 = 0.0082\,\mathrm{m},\ \beta_0 = 1.125\,\mathrm{rad}$

gray. The first two columns of a subfigure shows the COM velocity and leg-spring forces in lateral and forward directions (i.e. x and y directions in the inertial frame $\{U\}$), the body angle, and the moment. The last column shows d_1 (solid line) and/or d_2 (dashed line) as a function of time, COM path, and the eigenvalues as a function of the incrementing parameter. The rest of the parameters (i.e. k, v, β_0, and l_0) were chosen to closely match the stride length and frequency of cockroach data [22].

Protocol (a): Fixed COP on lateral axis. Figure 4(a) shows results of a simulated LLS model in which we fixed the COP at various positions along the x-axis of the body frame $\{B\}$. As desired, when $d_1 > 0$, the profiles of the body angle, θ, and the moment waveforms resemble actual cockroach data (Fig. 2), as well as that of the moving COP proposed by Schmitt and Holmes [24] (reproduced in Fig. 2, *Left*). Note that the positive cosinusoidal waveforms of the body angle (which agrees with the biological data) for a fixed COP only occur when d is on the positive x-axis of $\{B\}$. Figure 4(a) indicates that the increase in $|d_1|$ amplifies the body angle and the moment waveforms while the other measurements, including the stride length and frequency, remain relatively constant. This isolated effect of d_1 will be useful later on when we fit the data to another waveform. In addition, the eigenvalue plot shows that the system becomes unstable when $d_1 < 0$ and stable when $d_1 > 0$.

Protocol (b): Fixed COP on fore-aft axis. As a comparison to the previous result, Fig. 4(b) shows the effects of different locations d_2 for a fixed COP. Although the body angle is sinusoidal (not cosinusoidal, like the cockroach), the location of d_2 does have a larger impact on the magnitude of body angle and the stability of the system (steeper slope for the moving eigenvalue) than d_1 in the previous protocol. We speculate that one cause of this differences in impact level is due to the large value of β_0; since $\beta_0 = 1.12 > \pi/4$, the leg force is oriented primarily in the lateral direction rather than the fore-aft direction. Thus, changes in d_2 cause greater moment arm changes than the equal changes in d_1. We will utilize this effect in Sect. 4 by using d_2 as our control input. We also note that the body velocity (and position) and foot forces of this figure matches the previous figure. Although not shown, as the fixed COP position traverses in this neighborhood, without the restriction of $d_1 = 0$ or $d_2 = 0$, the body velocity and foot force waveforms remain relatively constant. On the other hand, the waveforms for the body angle and the moment go through phase and amplitude changes.

Protocol (c): Incrementing magnitude of a moving COP on the fore-aft axis. For fore-aft COP motion along the body frame y-axis, Fig. 5(c) shows the effects of changing the magnitude of COP motion. Unlike the previous protocols, varying the magnitude of a moving COP causes large changes to all the kinematics, step length, and step frequency. Although it is not shown here, further increase in magnitude (also observed in [22]) or reversing the direction (i.e. aft to fore) of the moving COP drives the system unstable.

Protocol (d): Forward Shifting of a Moving COP. Figure 5(d) shows the effects of shifting a moving COP in y direction in $\{B\}$. It shows that, as the offset b_2 increases (or decreases, although not shown), the body loses its cosinusoidal waveform and eventually becomes unstable. We emphasize that the system does not go unstable as soon as the offset $b_2 > 0$ nor $b_2 < 0$. Also the instability does not necessarily occur even though the moving COP remains in front of the COM most of the time. Along with Protocol (c), we introduce one possible explanation of these results in Sect. 3.3.

Protocol (e): Lateral Shifting of a Moving COP. Lastly, we look at the result of incrementing the lateral offset to a moving COP, as shown in Fig. 6(e). The result resembles that of Protocol (a) in Fig. 4(a); the changes in d_1 mostly affect the magnitude of body angle and moments, but the waveforms all remain qualitatively the same shape. Also, the increase in d_1 has amplified the waveforms, and the moving COP has stabilized the system even with $d_1 < 0$, in contrast to the results of Fig. 4(a).

From the results from these protocols, we conclude that we can achieve the desired cosinusoidal waveforms by laterally offsetting a fixed COP or moving COP. However, both cases produce body angle and moment variations that remain an order of magnitude smaller than those of a cockroach. This can be remedied with a very large – possibly non-physical – COP offset of $d_1 =$

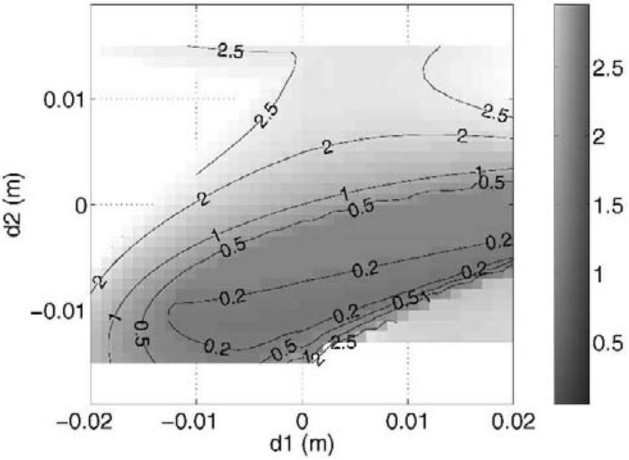


Fig. 7. Maximum eigenvalue (neglecting two invariant unity eigenvalues for energy and direction) of the linearized return map as a function of our new, two dimensional COP locations. The *dark gray* indicates the parameter regime of maximum stability and the neutral stability occurs when the contour reads 1. Eigenvalues greater that 3 are empty. The parameter values used are: $v = 0.25\,\text{m/s}$, $k = 2.4\,\text{Nm}^{-1}$, $l_0 = 0.01\,\text{m}$, $\beta_0 = 1\,\text{rad}$, and -0.02 m $< d_1, d_2 <$ 0.02 m

0.025 m $\approx 2l_0$ and $d_2 = 0$, which means that the virtual foot touchdown position will be far off to the positive x-axis in $\{B\}$ along with the COP. The resulting magnitude of the body angle was about 2 ° (or 0.035 rad) with the moment of 0.3×10^{-4} N m. This is within an order of magnitude of the cockroach variations 5.7° (or 0.1 rad) and 1×10^{-4} N m in Fig. 2.

3.2 Stability as a Function of Fixed COP Position

Figures 4(a) and 4(b) showed the stability plot of the LLS with a fixed COP along the x and y-axis of $\{B\}$, respectively. Figure 7 shows a contour plot of the maximum non-unity eigenvalues as a function of more general 2D fixed COP positions. Note that the neutrally stable (i.e. $\max \lambda = 1$) gait corresponding to $(d_1, d_2) = (0, 0)$ found by Schmitt and Holmes [23] lies along a neutral stability contour through the origin of the d-plane. There is a large stable region ($\max \lambda < 1$) "inside" the neutral stability contour and an abrupt area of instability in the lower-right corner of the plot. Notice that the stable region ($\max \lambda < 1$) extends to a part of $d_2 > 0$ region for $d_1 > 0$. This indicates that we can achieve stability for the fixed COP that is in front of COM, as long as it is sufficiently offset to the right ($d_1 > 0$). We also notice that around the origin, the gradient of the eigenvalues is greater in the direction of y-axis than x-axis of $\{B\}$. This hints that a small displacement of the fixed COP in d_2 should give us a greater control than that of d_1. We utilize this notion in Sect. 4.

Our long-term goal is to match the LLS to biological or robotic locomotion performance, possibly using the LLS as a plant model for control. Therefore, we examined the equilibrium state values, δ^*, θ^*, and $\dot{\theta}^*$, in Fig. 8, as a recipe for future comparisons to biological and robotic systems. As expected, the two contours $\theta^* = 0$ and $\dot{\theta}^* = 0$ indicate purely oddly symmetric (sinusoidal) and evenly symmetric (cosinusoidal) yaw motions, respectively, and these symmetries only occur on those contours.

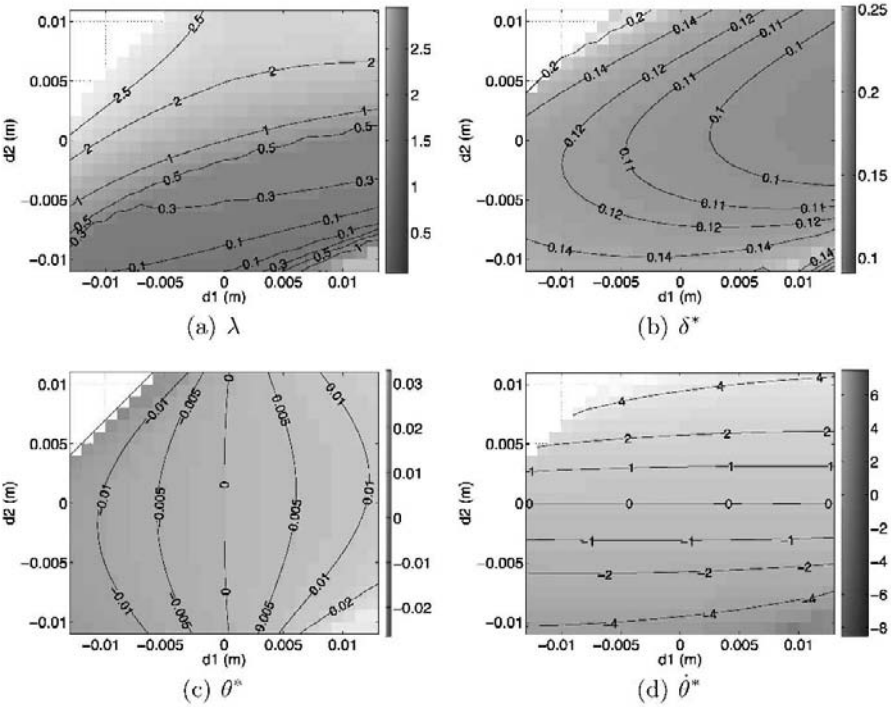
3.3 Comparing Fixed vs. Moving COP Models

From the observations above, we consider the relationship between a fore-aft moving COP and a fixed, laterally offset COP. These two scenarios generate similar waveforms; in fact, using very similar parameters, we can nearly match the body motions and forces using these two strategies, as shown in Fig. 9. To find a good match, we relied on trial and error, using Figs. 4(a) and 5(c) as a guide to adjust d_1 and d_2 and we referred to [22] to adjust l_0, β_0, and k. As shown, the body angle (yaw) motions match nearly exactly, while for the other measurements, the fixed COP exceeded the moving COP somewhat in magnitude, although the results are qualitatively similar.

We compare the moving COP model to a model with a fixed COP on the positive x-axis of $\{B\}$, as follows. As the LLS moves through a left step, the leg intersects the body centerline at a point that moves fore-aft, as depicted in Fig. 10(a). Suppose there is another LLS system with a moving COP that traces out the same path, and has the same foot touchdown position as the fixed COP case. With appropriate parameters (and possibly a nonlinear leg spring), the fixed COP LLS model might approximate the moving COP model. By approximating the moving COP with the fixed COP in this way, we can predict which moving COP protocols might be stable on the basis of the stability contour map (Fig. 7, Sect. 3.2). Using this approach, we address below (without formal proofs) unanswered questions from Sect. 3.1.

In Protocol (c), we considered increasing the magnitude of a moving COP. We approximate this case using the effective fixed COP and effective β_0 shown in Fig. 10(b) and (c); a larger magnitude can be created by moving the effective fixed COP in the x direction and/or decreasing the value of leg touchdown angle β_0. From Fig. 4(a), we saw that the increase in d_1 for a fixed COP improved stability and amplified the body angle and moment, which agrees with increasing the magnitude of fore-aft motion in the moving COP, as shown in Fig. 5(c). Similarly, a moving COP that is shifted forward, as in Protocol (d), can be approximated by shifting the effective fixed COP forward, as shown in Fig. 10(d). Figure 8(a) shows that the effective fixed COP will first be stable, but eventually it will be unstable as the offset increases further.

Earlier, we indicated that the system became unstable when the moving COP moved from back to front (i.e. aft to fore) along the body centerline. As Fig. 10(e) shows, the effective fixed COP would then be placed on the left side of the body centerline which, according to Fig. 8(a), would probably



(a) λ

(b) δ^*

(c) θ^*

(d) $\dot{\theta}^*$

Fig. 8. Contour map of the maximum non-unity eigenvalue and the equilibrium points δ^*, θ^*, and $\dot{\theta}^*$. The parameter values are the same as Fig. 7

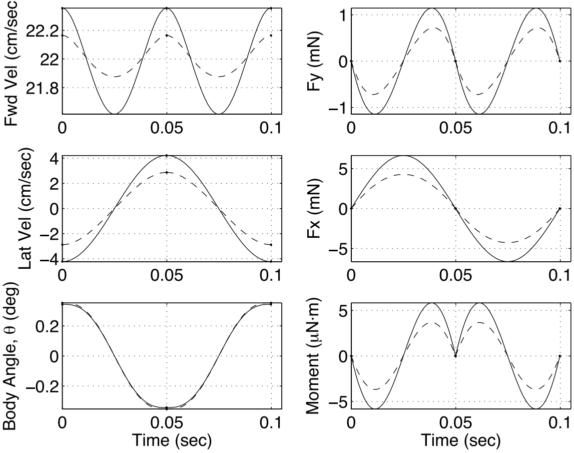


Fig. 9. Comparison between moving and fixed COP. $m = 0.0025\,\mathrm{kg}$, $I = 2.04 \times 10^{-7}\,\mathrm{kg\,m^2}$, $k = 3.52\,\mathrm{N\,m^{-1}}$, $\beta_0 = 1.125\,\mathrm{rad}$. Moving COP (*dashed*) $v = 0.2235\,\mathrm{m/s}$, $l_0 = 0.0082\,\mathrm{m}$, $d_1 = 0\,\mathrm{m}$, $d_2 = 0.002 \rightarrow -0.002\,\mathrm{m}$. Fixed COP (*solid*) $v = 0.2275\,\mathrm{m/s}$, $l_0 = 0.0128\,\mathrm{m}$, $d_1 = 0.005\,\mathrm{m}$, $d_2 = 0\,\mathrm{m}$

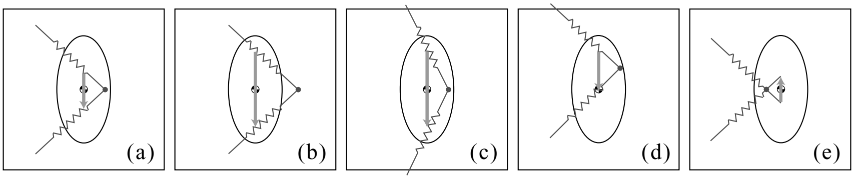


Fig. 10. Comparison between a moving COP and an effective fixed COP during a left step under different protocols. Fixed COP is denoted by a filled dot, and the moving COP is denoted by a gray arrow

make the system unstable. This also suggests we can achieve stability for a forward moving COP if we choose our offsets carefully.

In Protocol (e), we increased the lateral offset of a moving COP. We can represent this simply by laterally shifting the effective fixed COP which is similar to Protocol (a), Fig. 4(a). Indeed that is what we observed in Fig. 6(e). This explains why the system remained stable when $d_1 < 0$; the effective fixed COP position was to the right of the COM ($d_1 > 0$). This implies that for cockroaches, if their mechanics limit the magnitude of d_2, i.e. they cannot have a large c_2, then an increase d_1 will achieve the desired stability, or vice versa; this would explain the shift in the moving COP observed in cockroaches [28].

In summary, the moving COP model is more complex than the fixed COP, but they have similar performance in matching biological data. Thus, in the next section, we assume the COP is fixed to the right of the COM within each step, but let the controller adjust the location of the COP between steps.

4 LLS Control: Wall Following

In addition to their remarkable stability, cockroaches also exhibit extraordinary maneuverability. For example the American cockroach, *Periplaneta americana*, follows walls using tactile feedback from their antenna, making up to 25 turns per second in response to jagged walls in the environment [3, 6].

Despite its simplicity, the LLS model captures many salient features of the dynamics and stability of steady-state cockroach locomotion. Building on these strengths, we explored using the LLS as a "plant model" for control. Schmitt and Holmes [24] tested the idea of moving the COP to steer locomotion. They noted that briefly moving the COP in front of the COM generates large turns of 20–70°. Other possible control parameters, such as the spring stiffness, leg length, and step-length can also be used for steering, but Full et al. [13] contend that moving the COP is the most effective, and least fragile. Moreover, moving the COP for steering seems to be consistent with animal turning behavior [17].

4.1 LLS Plant Model

In Sect. 3, we compared the effects of moving the COP within a step, versus keeping the COP fixed. Both models can, with appropriate parameters, demonstrate asymptotic stability in the relative heading, δ, and angular velocity, $\dot{\theta}$, but neutral stability in running speed, $v \in \mathbb{R}^+$, orientation θ, and x (if we're running in the y direction of $\{U\}$). As discussed above, the moving COP adds complexity but provides very little advantage over the laterally offset but fixed COP model when it comes to matching steady-state cockroach data. Therefore, we explored using step-to-step adjustments of the COP as an input to control the overall body location in $g \in \mathrm{SE}(2)$. Because there are no energy losses between steps due to impacts, the controlled LLS remains piecewise Hamiltonian and energy conserving.

Initially, we explored control laws that varied d_1, d_2, β_0, and combinations thereof. We found that a highly effective control scheme was to fix β_0 and place the nominal COP to the right of the COM (for left steps), varying only the fore-aft COP location (d_2) from step-to-step. This is consistent with biological observations that rapid maneuvering in cockroaches occurs with large changes in the fore-aft COP [17]. Specifically, we used the step-to-step control input

$$d_k = \alpha e_1 + e_2 u_k, \quad \text{where} \quad e_1 = [\,1,\ 0\,]^T,\ e_2 = [\,0,\ 1\,]^T\ , \tag{9}$$

k is the stride number, $u_k \in \mathbb{R}$ is the control input, and α is a scalar. As shown in Sect. 2, selecting $\alpha > 0$ ensures that for $u_k \equiv 0$, the system is asymptotically stable in δ and $\dot{\theta}$, and neutrally stable in v and g. The result is a step-to-step return map,

$$q_{k+1} = f(q_k, u_k)\ , \tag{10}$$

that is no different from the step-to-step uncontrolled LLS in (8), except that between steps the COP location can be adjusted according to (9).

4.2 Antenna-Based Output Function

We assume that the LLS controller will have at its disposal proprioceptive measurements at each step, such as its angular velocity, $\dot{\theta}_k$, and relative heading, δ_k, but not necessarily its position and orientation relative to our arbitrarily assigned world reference frame, $\{U\}$. Therefore, in order for the LLS to achieve some task level goal in SE(2), it needs exteroceptive feedback. For this, we derive inspiration from nature, and assume the LLS has a simplified antenna that measures its distance from a surface in the environment as depicted in Fig. 11.

Our antenna model follows [6, 7, 8] and assumes the antenna estimates the distance, γ, from the body centerline to a "wall" – in this case the y-axis – ahead of the COM a known, fixed *preview distance*, ℓ. Under these assumptions

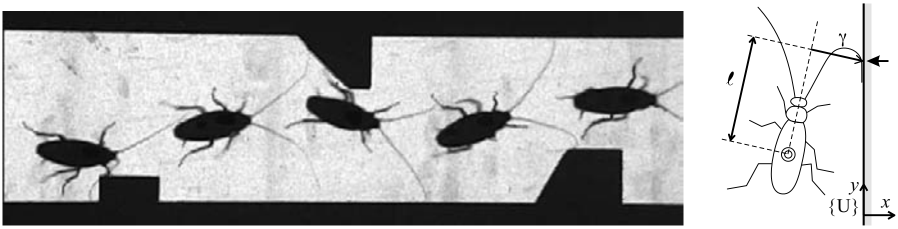


Fig. 11. *Left:* Multiple exposures of the cockroach *P. americana* negotiating a set of obstacles at high speed, using feedback from its antennae [6]. *Right:* A simplified model of an antenna as a distance-to-wall sensor

$$\gamma = \ell \tan\theta - x \sec\theta \; . \tag{11}$$

The above equation (11) relates the LLS pose to the antenna measurement, γ. We make no attempt to avoid collisions of the LLS with the virtual wall and for simplicity, our controller will drive the LLS to align itself directly on top of the y-axis, though this can easily be extended to drive the LLS to follow at an offset distance from the wall. Together, the proprioceptive and exteroceptive measurements yields the measurement function:

$$h(q) := \left[\delta, \dot{\theta}, \gamma\,\right]^T \; . \tag{12}$$

4.3 Reduced Return Map

To simplify controller analysis and design, we reduced the model, by using translational symmetry and energy conservation, as follows. Recall that the left- and right-step mappings, f_L and f_R, are invariant to SE(3), but the step-to-step return map, $f = Mf_L$, is not. However, that mapping is invariant to pure y motions (had we chosen a different left-handed frame, translational invariance would have been in the direction of the axis of symmetry of the reflection to that frame). This was by design: our goal for control is wall following, and for simplicity, we have chosen to follow the y-axis. In addition, the output mapping, h, is y-translation invariant. Thus we removed the y equation by setting $y = 0$ at the beginning of each step. Naturally, we ignored the final value of y when finding an equilibrium as we did in Sect. 2. To remove v note that

$$H = \frac{1}{2}mv^2 + \frac{1}{2}I\dot{\theta}^2 + \frac{1}{2}k(\eta - l_0)^2 = H_0 \equiv \text{constant} \; . \tag{13}$$

So, at each step

$$v = \left[\frac{2}{m}\left(H_0 - \frac{1}{2}I\dot{\theta}^2 - \frac{1}{2}k(\eta - l_0)^2\right)\right]^{1/2} \; . \tag{14}$$

Thus we defined a transformation

$$T_H : (\delta, \dot{\theta}, \theta, x) \mapsto (v, \delta, \dot{\theta}, \theta, x, 0) \tag{15}$$

that assigns $y = 0$ and computes v from (14). Note that T_H is invertible and T_H^{-1} is the transformation that simply removes the v and y coordinates. Then, we define the reduced variables and mapping

$$q^r = (\delta, \dot{\theta}, \theta, x) \quad \text{and} \quad f^r(q_k^r, u_k) = T_H^{-1}\big(f(T_H(q_k^r), u_k)\big) \ . \tag{16}$$

4.4 Linearized Return Map, Controllability and Observability

As a preliminary control task, we chose to have the LLS follow a virtual "wall" coincident with the y-axis. To find an equilibrium, we used similar techniques to those described in Sect. 2 to find equilibrium trajectories, ensuring that $x = 0$ at the beginning and end of each step, corresponding to exact wall following. The result was an equilibrium q^*, such that $q^* = f^r(q^*, 0)$. To address controllability, we numerically linearized the return map around a nominal equilibrium trajectory, to obtain the local return map

$$\begin{aligned} e_{k+1} &= Ae_k + Bu_k \\ z_k &= Ce_k \end{aligned} \tag{17}$$

where

$$A = \left.\frac{\partial f^r}{\partial q^r}\right|_{q^r=q^*, u=0} \qquad B = \left.\frac{\partial f^r}{\partial u}\right|_{q^r=q^*, u=0} , \tag{18}$$

and $e_k = q_k^r - q^*$. The linearized output matrix can be derived analytically,

$$C = \left[\frac{\partial h}{\partial q^r}\right]_{q^r=q^*} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \ell & -1 \end{bmatrix} \tag{19}$$

but to date we only have numerical approximations to A and B. The reduced system (A, B, C) is stabilizable and observable for the parameters $m = 0.0025$ kg, $I = 2.04 \times 10^{-7}$ kg m^2, $l_0 = 0.01$ m, $k = 2.25$ N/m, $\beta_0 = 0.8$ rad, and a nominal COP offset of $\alpha = 0.0025$ m.

4.5 Antenna-Based Control Strategy

Because the system is completely observable, state feedback is possible; however, we found that the following simple output feedback to be quite effective:

$$u_k = Kz_k \ , \tag{20}$$

where z_k is the system output from (17). The closed loop system dynamics are governed by the $e_{k+1} = (A + BKC)e_k$, so to find a good choice for the gain, K, we evaluated the eigenvalues of the system matrix $(A + BKC)$. Amidst a variety of possible feedback gains, we selected $K = [\ 0,\ 0.001,\ 0.5\ ]$, which lead to complex conjugate pairs of closed loop eigenvalues at $-0.4643 \pm j0.2607$ and $0.3478 \pm j0.4827$. A demonstration of this controller, executed on the full nonlinear LLS dynamics, is shown in Fig. 12.

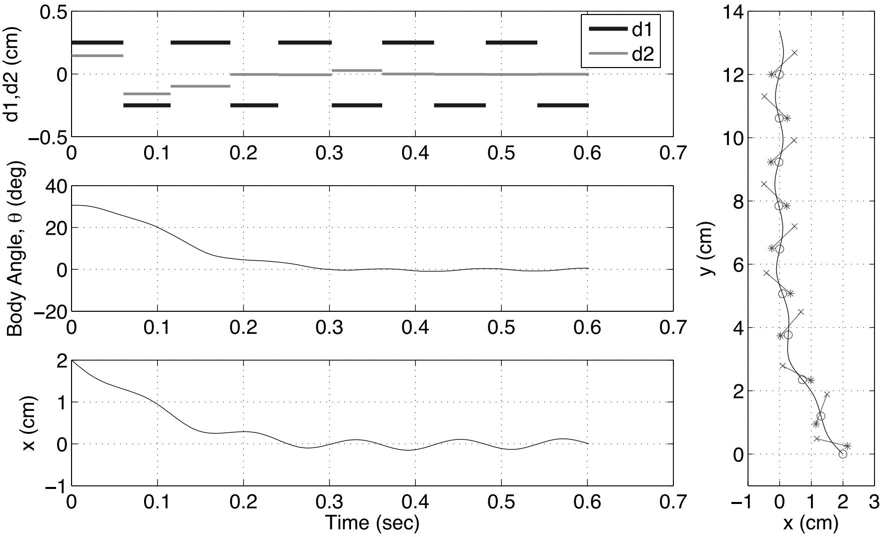


Fig. 12. Simulation of the controlled LLS following the y-axis using the feedback control law (20). In this control law, COP lies nominally along the body x, namely $d = [\ 0.25\text{cm},\ \ 0\ ]^T$ (for the *left step*). The output feedback controller (20) varies the COP in the d_2 direction. In the simulation, the LLS starts out rotated 30° counterclockwise from the y-axis, and 2 cm to the right. The figure on the *right* shows COM (○), COP (∗), and foot (×) positions at the start of each step

5 Conclusion

In this paper, we revised the LLS model to achieve the same phase relationship between the θ and δ as a real cockroach, using a fixed and laterally offset COP. Also we investigated how the COP location governed the overall system stability, and related the fixed COP model to the moving COP model presented by Schmitt and Holmes [23].

For control purposes, we reduced the system state to four dimensions, using translation symmetry and energy conservation. We then applied a very simple output-based feedback strategy to update the COP location between strides based on an antenna-like measurement. Using this controller, the reduced system dynamics were linearly stable. In the future, we will explore using the LLS model as a control template [12] for horizontal-plane control of running robots.

References

[1] R. Altendorfer, N. Moore, H. Komsuoglu, M. Buehler, H. B. Brown Jr., D. McMordie, U. Saranli, R. J. Full, and D. E. Koditschek. RHex: A biologically-inspired hexapod runner. *Autonomous Robots*, 11(3):207–213, 2001.

[2] R. Blickhan and R. J. Full. Similarity in multilegged locomotion: Bouncing like a monopode. *J Comp Physiol A*, 173:509–517, 1993.

[3] J. M. Camhi and E. N. Johnson. High-frequency steering maneuvers mediated by tactile cues: antenna wall-following in the cockroach. *J Exp Bio*, 202:631–643, 1999.

[4] G. A. Cavagna, N. C. Heglund, and C. R. Taylor. Walking, running, and galloping: Mechanical similarities between different animals. In T. J. Pedley, editor, *Scale Effects in Animal Locomotion, Proceedings of an International Symposium*, pp. 111–125. Academic Press, New York, USA, 1975.

[5] J. G. Cham, S. A. Bailey, J. E. Clark, R. J. Full, and M. R. Cutkosky. Fast and robust: Hexapedal robots via shape deposition manufacturing. *The International Journal of Robotics Research*, 21(10), 2002.

[6] N. J. Cowan, J. Lee, and R. J. Full. Dynamical control of antennal wall following in the American cockroach. In *The Society for Integrative and Comparative Biology*, p. 126, San Diego, CA, January 2005.

[7] N. J. Cowan, J. Lee, and R. J. Full. Task-level control of rapid wall following in the American cockroach. *J Exp Biol*, 209:1617–1629, 2006.

[8] N. J. Cowan, E. J. Ma, M. Cutkosky, and R. J. Full. A biologically inspired passive antenna for steering control of a running robot. In *International Symposium on Robotics Research.* Siena, Italy, 2003.

[9] M. H. Dickinson, C. T. Farley, R. J. Full, M. R. Koehl, R. Kram, and S. L. Lehman. How animals move: An integrative view. *Science*, 288(5463):100–106, 2000.

[10] R. J. Full, R. Blickhan, and L. H. Ting. Leg design in hexapedal runners. *J Exp Biol*, 158:369–390, 1991.

[11] R. J. Full and C. T. Farley. Musculoskeletal dynamics in rhythmic systems – A comparative approach to legged locomotion. In J. M. Winters and P. E. Crago, editors, *Biomechanics and Neural Control of Movement*, pp. 192–205. Springer-Verlag, New York, 2000.

[12] R. J. Full and D. E. Koditschek. Templates and anchors: neuromechanical hypotheses of legged locomotion on land. *J Exp Bio*, 202(23):3325–3332, 1999.

[13] R. J. Full, T. M. Kubow, J. Schmitt, P. Holmes, and D. E. Koditschek. Quantifying dynamic stability and maneuverability in legged locomotion. *Integrative and Comparative Biology*, 42(1):149–157, 2002.

[14] R. J. Full and M. S. Tu. Mechanics of six-legged runners. *J Exp Bio*, 148:129–146, 1990.

[15] R. J. Full and M. S. Tu. Mechanics of a rapid running insect: two-, four-, and six-legged locomotion. *J Exp Bio*, 156:215–231, 1991.

[16] F. Iida and R. Pfeifer. “Cheap” rapid locomotion of a quadruped robot: Self-stabilization of bounding gait. In F. Groen, N. Amato, A. Bonarini, E. Yoshida, and B. Krose, editors, *Intelligent Autonomous Systems 8*, pp. 642–649. IOS Press, 2004.

[17] D. L. Jindrich and R. J. Full. Many-legged maneuverability: Dynamics of turning in hexapods. *J Exp Bio*, 202(12):1603–1623, 1999.

[18] R. Kram, B. Wong, and R. J. Full. Three-dimensional kinematics and limb kinetic energy of running cockroaches. *J Exp Bio*, 200:1919–1929, 1997.

[19] T. M. Kubow and R. J. Full. The role of the mechanical system in control: a hypothesis of self-stabilization in hexapedal runners. *Philosophical Transactions of*

the Royal Society of London Serieseoleol B-Biological Sciences, 354(1385):849–861, 1999.

[20] R. D. Quinn, G. M. Nelson, R. J. Bachmann, D. A. Kingsley, J. Offi, and R. E. Ritzmann. Insect designs for improved robot mobility. In Berns and Dillmann, editors, *Conference on Climbing and Walking Robots*, pp. 69–76, Karlsruhe, Germany, 2001. Professional Engineering Publications.

[21] A. Ruina. Nonholonomic stability aspects of piecewise-holonomic systems. *Rep. Math. Phys.*, 42(1-2):91–100, 1998.

[22] J. Schmitt, M. Garcia, R. C. Razo, P. Holmes, and R. J. Full. Dynamics and stability of legged locomotion in the horizontal plane: a test case using insects. *Biol Cybern*, 86:343–353, 2002.

[23] J. Schmitt and P. Holmes. Mechanical models for insect locomotion: dynamics and stability in the horizontal plane I. Theory. *Biol Cybern*, 83:501–515, 2000.

[24] J. Schmitt and P. Holmes. Mechanical models for insect locomotion: dynamics and stability in the horizontal plane-II. Application. *Biol Cybern*, 83:517–527, 2000.

[25] J. Schmitt and P. Holmes. Mechanical models for insect locomotion: active muscles and energy losses. *Biol Cybern*, 89:43–55, 2003.

[26] W. J. Schwind and D. E. Koditschek. Approximating the stance map of a 2-dof monoped runner. *Journal of Nonlinear Science*, 10(5):533–568, 2000.

[27] A. Seyfarth, H. Geyer, M. Gunther, and R. Blickhan. A movement criterion for running. *J Biomech*, 35(5):649–655, May 2002.

[28] L. H. Ting, R. Blickhan, and R. J. Full. Dynamic and static stability in hexapedal runners. *J Exp Biol*, 197:251–269, 1994.

Investigating the Use of Iterative Learning Control and Repetitive Control to Implement Periodic Gaits

R.W. Longman[1] and K.D. Mombaur[2]

[1] Dept. of Mechanical Engineering, Columbia University, New York, 10027, USA
rwl4@columbia.edu

[2] IWR, Universität Heidelberg, Im Neuenheimer Feld 368, 69120 Heidelberg, Germany
katja.mombaur@iwr.uni-heidelberg.de

Summary. In the next few years considerable effort will be expended to make humanoid robots that can do true dynamic walking, or even running. One may numerically compute a desired gait, e.g. one that has been optimized to be asymptotically stable without feedback. One would normally give the gait as commands to the controllers for the robot joints. However, control system outputs generally differ from the command given, and the faster the command changes with time, the more deviation there is. Iterative learning control (ILC) and repetitive control (RC) aim to fix this problem in situations where a command is repeating or periodic. Since gaits are periodic motions, it is natural to ask whether ILC/RC can be of use in implementing gaits in hardware. These control concepts are no substitutes for feedback control but work in combination with them by adjusting the commands to the feedback controllers from a higher level perspective. It is shown that the gait problem does not precisely fit either the ILC or the RC problem statements. Gait problems are necessarily divided into phases defined by foot strike times, and furthermore the durations of the phases are not the same from cycle to cycle during the learning process. Several methods are suggested to address these issues, and four repetitive control laws are studied numerically. The laws that include both position and velocity error in the updates are seen to be the most effective. It appears that with appropriate refinement, such generalized RC laws could be very helpful in getting hardware to execute desired gaits.

1 Introduction

In the last few years, many humanoid and biped walking robots have been built executing periodic or quasi-periodic gaits [1, 2]. So far such robots are rather slow moving compared to their biological counterparts, and the traditional control approach keeps them as close as possible to a quasi-static equilibrium during the motion, e.g. [5, 25]. As research progresses into making robots that do true dynamic walking or running, in addition to dealing with the dynamic behavior of the nonlinear multibody robot system, it will become necessary to address the imperfect dynamics of any feedback control system that is used to attempt to execute the chosen periodic gait.

In recent years, the fields of iterative learning control (ILC) and repetitive control (RC) have appeared [3, 19], the goal of which is to improve the performance of feedback control systems by adjusting the commands given to them. In principle, these techniques could be combined with any chosen feedback control concept. ILC and RC have now developed to the point that commercial use of the methods has started to appear. Recently, robots have been delivered to Daimler-Chrysler in Detroit using ILC to improve performance. And RC is now being used at the factory to learn a command for each track of computer disk drives. The improved performance allows the tracks to be closer together, and hence the disks can store more data. Similar methods are being used to speed up chip manufacturing, allowing the manufacturing hardware to operate faster while maintaining the needed precision, and hence increase productivity.

ILC suddenly began to develop quickly in 1984 motivated by robots performing repetitive operations in a manufacturing environment. Each time the task is performed, the robot restarts from the same initial conditions. When a feedback control system is given a tracking command, the response is not the same as the command, even under ideal circumstances with perfect measurements and no plant noise disturbances. Generally, the faster the requested motion in the tracking command, the larger the discrepancy between what is asked for and what is produced. The purpose of ILC is to use the error observed in the previous run (or repetition) to adjust the command in the current run, aiming to converge to that command that actually produces the desired trajectory. ILC asks the control system to execute commands that are not what you want the controllers to do, so that they actually do what you want them to do.

RC is a closely related type of control. Instead of making repeated runs of a desired finite time trajectory, each time starting from the same initial condition, RC aims to perfectly execute a periodic command, or to perfectly execute a constant command in the presence of a periodic disturbance (or to execute a periodic command with a periodic disturbance, each having the same period). The RC law learns from the measured error in the previous period (or cycle) instead of the previous run, adjusting the command in the current period, aiming to get to zero tracking error. Transients can propagate from one period to the next in RC, but cannot go from one run to the next in ILC, and this results in the two problems having different conditions for stability, i.e. for convergence to zero tracking error.

As gait research progresses from relatively slow robot walking motions to full dynamic walking, and then to running, the issues of imperfect execution of high speed commands by feedback control systems will become a serious issue. The desired periodic gaits are commands to the feedback controllers for each robot joint. If the discrepancy between the commanded trajectory and the trajectory executed by the feedback controllers is large enough, the robot will fall. Since ILC and RC are new fields of control theory that address how to make feedback control systems actually perform desired repeating or periodic motions, it is natural to ask whether ILC or RC can be used to implement high speed gaits. It is the purpose of this paper to make an initial evaluation of what ILC/RC can do for this problem, and to put forward some concepts that might address the issues that are raised.

2 Feedback Control System Errors that ILC/RC can Fix

Consider the performance of typical feedback control systems executing a time varying command. Suppose one wishes to control the output $y(t)$ of a first order system (the plant) $dy/dt + ay = w + v$ where $w(t)$ is a variable we can manipulate to change y, e.g. we can apply a torque to a robot link to make the output angle change. Typically, whatever variable we can manipulate, nature can also influence with various disturbances, e.g. in the case of a robot link, gravity can supply a torque history as the link follows the desired path, and v denotes such disturbances. Now consider applying a proportional controller to make the output follow a command. Then $w(t) = Ke(t) = K(y_C(t) - y(t))$ and the manipulated variable w is proportional to the measured error $e(t)$, the command $y_C(t)$ minus the measured output $y(t)$. The performance is then predicted by the closed loop differential equation

$$\frac{dy(t)}{dt} + (K+a)y(t) = Ky_C(t) + v(t) \tag{1}$$

whose solution is

$$y(t) = e^{-(K+a)t}y(0) + \int_0^t e^{-(K+a)(t-\tau)}Ky_C(\tau)d\tau + \int_0^t e^{-(K+a)(t-\tau)}v(\tau)d\tau \tag{2}$$

The middle term on the right is the part of the solution that is responding the command we give the system. But it is not equal to the command. Instead it is a convolution integral of the command, creating a form of weighted average of all the commands we have given the system in the past. The weighting factor decays going backward in time, so that more recent commands are more important in the weighted average. Therefore, for any command that is changing with time, the feedback control system does this convolution integral of what we asked it to do, not what we asked it to do. And the faster the command changes with time, the more effect the averaging has on the result. The first term on the right represents transients and is the response to initial conditions. The last term on the right gives the effect of disturbances on the performance of the control system. If the system has the same disturbance every time we give the command, then this disturbance is also a source of repeating error.

It is the purpose of ILC and RC to converge to a command $y_C(t)$ that is no longer equal to the desired output $y_D(t)$, but one which has the property that the system output given by the right hand side of (2) is in fact equal to $y_D(t)$. In the case of ILC the command has the property that over a finite time interval starting at time zero, the right side of (2) converges to the desired trajectory as the repetitions progress. In the case of RC, the command has the property that as time increases, the right hand side converges to the desired trajectory. Both fix deterministic errors in following a command, and also cancel any disturbances that repeat. ILC also learns to handle the first term on the right, since it is present in every run, while RC learns to get zero error as time goes to infinity, and for an asymptotically stable system the first term disappears with time.

The iterations solve an inverse problem of finding the input that produces the desired output. In the simple example above, one can directly solve this inverse problem, e.g. using the desired output $y_D(t)$ for the output in (1) the command needed is:

$$y_C(t) = \frac{1}{K}\left(-\frac{dy_D(t)}{dt} - (K+a)y_D(t) + v(t)\right) \tag{3}$$

There are usually difficulties with this in practice. First, when done in discrete time, the inverse problem is usually ill-conditioned [11]. Second, if the external disturbance $v(t)$ is an important aspect of the problem, one needs to know this function which may be difficult. And third, the solution is only as good as the model is. ILC and RC find an inverse solution iterating with the real world behavior, instead of the model, without needing to know $v(t)$, and without totally relying on a model.

3 ILC/RC Background

The most effective ILC/RC design methods are based on linear systems theory, and the discussion presented here is limited to this approach. Results have been generated for doing nonlinear ILC on equations having the form of multibody dynamic systems. These results are likely to be not as practical as the linear design methods. First, they rely on all of the dynamics in the physical system having the multibody dynamics form, which is often not the case when actuators and sensors and effective feedback controllers are included. Second, they create very complex control laws that are more complicated to implement. And third, linear methods as in [4] can converge to tracking errors that approach the minimum possible error, the repeatability level of the system, and do so relatively quickly without requiring the complicated modeling of the nonlinear system. Figure 1 shows the robot used in [4], and

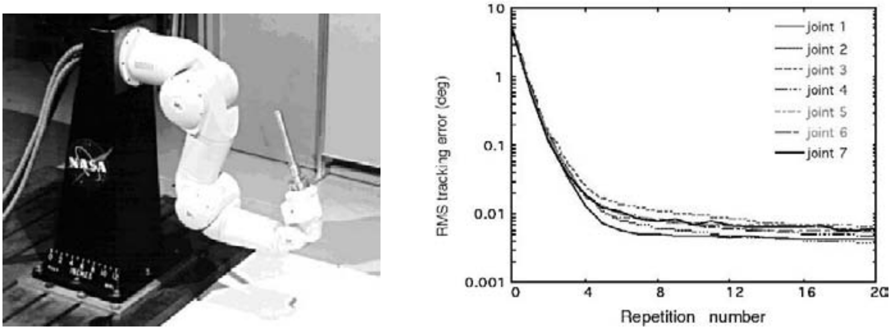


Fig. 1. Robotics Research Corporation robot, and RMS error vs. repetitions using an ILC law

also the tracking error for each ILC repetition for each robot link following a high speed trajectory versus repetitions. The error is decreased by a factor of roughly 1000 in about 12 runs. Note that the final error level is actually below the repeatability level of the robot when measured on a day to day basis, so the ILC is fixing errors of the size of how different the robot behaves from one day to the next. No amount of modeling could predict such errors. The fact that ILC does not rely heavily on a model allows it to fix such errors. Nevertheless, in gait problems one might want to revisit the question of the usefulness of using fully nonlinear methods, or at least using some form of feedback linearization.

There is a related issue for robot applications. One may consider creating an ILC or RC law that has multiple inputs and outputs, one for each of the joints variables. On the other hand, it is much simpler to use a decentralized ILC or RC approach, that applies a separate independent law to each of the separate feedback control systems for each robot joint as if there were no coupling between joints in the nonlinear dynamics. Again, the results in [4] are obtained using decentralized ILC, suggesting that the simple decentralized approach can be very effective in robot applications.

Both ILC and RC must necessarily be implemented by digital control methods, because the control updates are based on data from a previous repetition or a previous period, and therefore must be measured and stored in a computer or microprocessor. One will normally use a zero order hold on the input signal that the ILC or RC adjusts. Consider ILC. The objective is to perform a finite time trajectory, and get zero error at the sample times, i.e. we want the output $y(kT)$ to converge to the desired output $y_D(kT)$ for $k = 1, 2, 3, \ldots, N$. Here T is the sample time interval of the digital control system, and the desired trajectory is N steps long. The error is $e(kT) = y_D(kT) - y(kT)$. The simplest form of ILC is based on pure integral control concepts being applied in repetitions to each of the time steps of the problem. Stated in words for a robot link, if the robot link were 2 degrees too low at a certain time step in the last run or repetition, then add 2 degrees, or 2 degrees times a learning gain ψ, to the command this repetition. Mathematically, this is written as

$$u_j(kT) = u_{j-1}(kT) + \psi e_{j-1}((k+1)T) \tag{4}$$

where j is the current repetition number, $j-1$ is the previous repetition. Based on the ILC description above, u represents the command to the feedback control system, but we note that some of the ILC literature accomplishes the learning by modifying the error signal going into the controller or the manipulated variable coming out of the controller instead, and then the u in (4) either represents the error signal from the comparator or the w output of the controller as discussed above (1) [20]. The +1 in the argument of the error is introduced to account for the one time step delay going through the feedback control system (or the plant equations), i.e. there is usually a one time step delay from the time step in which one changes the command (or the manipulated variable) to the first time step in the output where a resulting change is observed. The computations of the command (or manipulated variable) history to use in the next run can be made between runs, computed in a batch mode.

The RC equivalent of this learning law is used when one wants to execute a periodic command, and this time the period is N time steps long. The mathematical expression of the corresponding RC law becomes

$$u(kT) = u((k-N)T) + \psi e((k-N+1)T) \tag{5}$$

Instead of looking back to a previous repetition, one looks back one period. Note that unlike ILC which makes a batch update of the command history for the next repetition, RC normally runs with updates made every time step, in real time.

ILC law (4) is almost always stable for sufficiently small gains ψ, but the learning transients are very likely to be impractical [8]. However, there is an important exception to this that occurs when the sample time of the ILC updates is sufficiently long that the system comes close to a steady state response before the next update arrives. RC law (5) is usually unstable. In both cases the error may decrease very substantially in the first few iterations or periods, and then the error starts to grow [9, 10]. It can be that one is satisfied with this level of improvement and simply stops the process when the error is a minimum. To improve performance and obtain stability robustness one normally generalizes the above laws to include a dynamic compensator in place of the gain ψ, and introduces a zero-phase low-pass filter cutoff of the learning [4, 24, 22, 21, 8]. Equations (4) and (5) can take the form

$$\underline{u}_j = F\left[\underline{u}_{j-1} + L\underline{e}_{j-1}\right] \tag{6}$$

$$U(z) = F(z)z^{-N}\left[U(z) + L(z)E(z)\right] \tag{7}$$

In (6) the underbar indicates a column matrix of the history of the associated variable in a repetition, and F and L are matrices representing the low pass filter and the compensator, respectively. Equation (7) for RC converts to the z-transform domain with $F(z)$ and $L(z)$ being the transfer functions of the cutoff filter and the compensator.

4 Dynamic Models for Walking Motions

The purpose of this section is to present the mathematical models of walking to which the concepts of ILC/RC will be applied. We start by giving the general form of dynamic walking models, and then present the specific stiff-legged biped walker used for the numerical tests later in this paper.

Mathematical models of gaits involve distinct model phases with possibly different degrees of freedom, each described, in the general form, by a different set of differential equations. These can be ordinary differential equations (ODEs)

$$\dot{q}(t) = v(t) \tag{8}$$

$$\dot{v}(t) = a(t) = M^{-1}(q(t), p) \cdot f(q(t), v(t), w(t), p) \tag{9}$$

In these equations, the vector q contains the position variables of the system, and v the corresponding velocities; together they form the vector of state variables $x^T = (q^T, v^T)$. The vector y used in the context of ILC and RC is typically equal to q. The scalar t is the physical time, a the vector of accelerations, $w(t)$ are the input torques or forces of the robot, and p is the vector of model parameters (like geometric or inertial data). M denotes the mass matrix, and f the vector of forces.

Alternatively, depending on the choice of coordinates, one may obtain a system of differential-algebraic equations (DAE) of index 3 for some or all phases

$$M(q(t), p) \cdot a = f(q(t), v(t), u(t), p) - G^T(q(t), p)\lambda \tag{10}$$

$$g_{pos}(q(t), p) = 0 \tag{11}$$

with the Lagrange multipliers λ, the constraint equations g_{pos}, and their partial derivatives $G = \frac{\partial g_{pos}}{\partial q}$. We formulate the DAEs in the equivalent index 1 form with invariants

$$\dot{q}(t) = v(t) \tag{12}$$

$$\dot{v}(t) = a(t) \tag{13}$$

$$\begin{pmatrix} M(q(t), p) & G^T(q(t), p) \\ G(q(t), p) & 0 \end{pmatrix} \begin{pmatrix} a \\ \lambda \end{pmatrix} = \begin{pmatrix} f(q(t), v(t), w(t), p) \\ \gamma(q(t), v(t), p) \end{pmatrix} \tag{14}$$

$$g_{pos} = g(q(t), p) = 0 \tag{15}$$

$$g_{vel} = G(q(t), p) \cdot \dot{q}(t) = 0\ . \tag{16}$$

with the abbreviation

$$\gamma(q(t), v(t), p) = -v^T \, \frac{d\, G(q(t), p)}{d\, q} \, v \, . \tag{17}$$

Phase boundaries are implicitly defined by the roots of switching functions

$$s_i(t, q(t), v(t), p) = 0 \, . \tag{18}$$

At these switching points, there may be discontinuities in the right hand side of the linear system, i.e. $\Delta f(q, v, w, p), \Delta\gamma(q, v, p)$ (which translates into discontinuities in the accelerations Δa), or even in the velocities, $\Delta v(t, q, v, w, p)$, i.e. in the state variables themselves. Walking problems also involve a number of complex linear and nonlinear, coupled and decoupled equality and inequality constraints; e.g. the periodicity constraints on the state variables (or a subset thereof) $\tilde{x}(T_{cycle}) = \tilde{x}(0)$. The cycle time T_{cycle} is generally a priori unknown. In this paper, we investigate the simple example of a planar stiff-legged biped walker with two degrees of freedom. The state variables of this robot are the stance leg angle ϕ_1 and the swing leg angle ϕ_2, and the corresponding velocities $x^T = (\phi_1, \phi_2, \dot{\phi}_1, \dot{\phi}_2)$. The robot has two torque actuators - one corresponding to each degree of freedom - the first one $w_1(t)$ at the hip, and the second one $w_2(t)$ at the ankle to replace the action of a foot with an actuated ankle joint. For repetitive control problems it is convenient to introduce a second set of state variables corresponding to the torques with $\bar{x}^T = (\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2)$, where

$$\theta_1 = \phi_1 - \phi_2 \tag{19}$$

$$\theta_2 = \phi_1 \tag{20}$$

This model can be considered as an extension of the classical passive-dynamic stiff-legged bipeds of McGeer [12]. The robot is shown in Fig. 2. It is characterized by three free parameters $p = (m, l, c)^T$ with

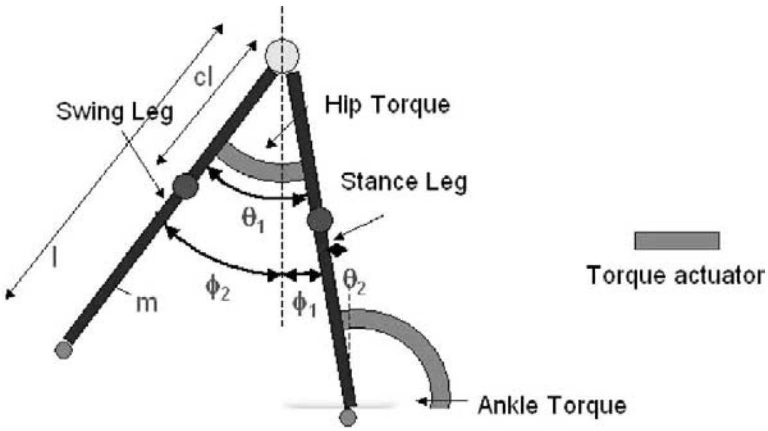


Fig. 2. Sketch of stiff-legged biped investigated in this paper

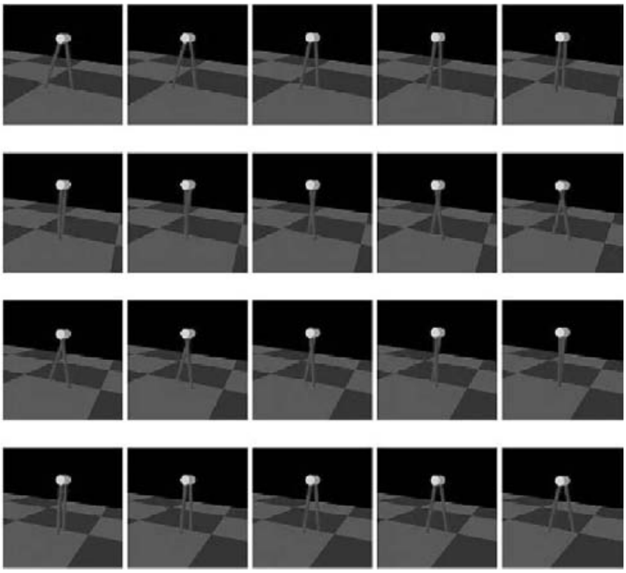


Fig. 3. Two steps of periodic reference solution of stiff-legged biped

- the mass of each leg, denoted by m
- the leg length l
- the relative location of the leg's center of mass measured from the hip, c.

Using these three parameters, the moment of inertia Θ of a leg is defined as

$$\Theta = \frac{1}{6} m l^2 (1 + 2c^2 - 2c)\ . \tag{21}$$

One cycle of the gait model includes one step of the robot followed by a leg switch, and not a full physical gait cycle consisting of two steps (as presented in Fig. 3). Applying periodicity constraints to this model assures the generation of equal right and left steps, which would not necessarily be the case otherwise. One cycle of this model consists of one continuous phase describing the forward swing and a discrete phase including the sudden change of velocities at foot contact and the leg switch.

The dynamic equations of this robot model are

$$M \cdot \begin{pmatrix} \ddot{\phi}_1 \\ \ddot{\phi}_2 \end{pmatrix} = F \tag{22}$$

with mass matrix

$$M = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix} \tag{23}$$

$$\text{with } m_{11} = 2ml^2 + \varTheta - 2ml^2c + ml^2c^2 \tag{24}$$

$$m_{12} = ml^2c\,\sin\phi_2\,\sin\phi_1 - ml^2c\,\cos\phi_2\,\cos\phi_1 \tag{25}$$

$$m_{21} = ml^2c\,\sin\phi_2\,\sin\phi_1 - ml^2c\,\cos\phi_2\,\cos\phi_1 \tag{26}$$

$$m_{22} = ml^2c^2 + \varTheta \tag{27}$$

and force term

$$F = \begin{pmatrix} -m\dot\phi_2^2 l^2 c\,\sin\phi_2\,\cos\phi_1 + m\dot\phi_2^2 l^2 c\,\cos\phi_2\,\sin\phi_1 \\ +2mgl\,\sin\phi_1 - mgl\,\sin\phi_1 c + w_1 + w_2 \\ \\ -m\dot\phi_1^2 l^2\,\sin\phi_1 c\,\cos\phi_2 + m\dot\phi_1^2 l^2\,\cos\phi_1 \\ c\,\sin\phi_2 - mglc\,\sin\phi_2 - w_1 \end{pmatrix} \tag{28}$$

The end of the step is determined by the equation

$$s(t, x, p) = \phi_1 + \phi_2 = 0 \tag{29}$$

The torques at hip and ankle are produced by feedback control systems using proportional control and rate feedback following a commanded trajectory $\theta_{i,C}$:

$$w_i(t) = K_1(\theta_{i,C} - \theta_i) - K_2\dot\theta_i \tag{30}$$

More details about this robot model as well as a description of possible extensions using springs and dampers in parallel with the torque actuators are given in [18].

5 Open Loop Stable Gaits

Previous research by the authors established that it is possible to have running gaits that are open-loop stable, meaning that they will return to the stable gait after small enough disturbances to position and velocity. This is accomplished without any feedback adjustment of the torque histories being applied to each joint. In the motions of ballet dancers and athletes, one suspects that there is often some inherent stability of the motions used, and we see that in running, hopping and somersaults this is also possible for robots [14, 17, 16]. In is generally preferable to create systems that are open loop stable and use feedback to augment the stability, than to rely totally on feedback to stabilize the motion. Pilots prefer to fly airplanes that do not immediately go unstable if there is a failure in the attitude control system.

With this in mind, we consider implementing an open loop stable gait to test the principles of ILC and RC. Numerically solving for such gaits for actuated robots is not trivial and requires an appropriate selection of model parameters p and driving torques $w(t)$. We have developed special numerical optimization techniques [17, 18] that can produce self-stabilizing solutions for walking robots. They have been applied to stabilize a series of different monopod and biped robots [15], one new example is given in another paper in this proceedings volume [13]. As a stability measure, we use the spectral radius of the Jacobian of the Poincaré map which is associated with the periodic solution. If the spectral radius is smaller than one, the solution is asymptotically stable, and if it is larger than one, the solution is unstable. For the stiff-legged biped described above, we have determined an open loop stable solution that is characterized by a spectral radius of 0.7, well below one, but which is also very efficient and requires only small torque inputs. The parameters p of this solution are $m = 1$ kg, $l = 0.1\,\mathrm{m}$, and $c = 0.25$; the cycle time is 0.4586 s. The initial values are $x_0^T = (0.25, -0.25, -1.943, -2.688)^T$. The corresponding torque inputs as well as the trajectories of the angles and rates are shown in Fig. 4. More information about the solution, as well as the objective functions used to create it, are given in [18].

6 Learning to Execute Open Loop vs. Closed Loop Stable Gaits

6.1 Problem Statement

One can pose a couple of different kinds of gait problems that might benefit from use of ILC or RC:

- **Problem 1.** As discussed above, one expects that there are benefits to using gaits that are open loop stable, so that there is already some inherent stability to the motion. An open loop stable solution obtained numerically gives a torque input history for each joint, the resulting output history or gait, and its derivative. The next step is to design a feedback control system for each link, since ILC and RC normally adjust the command to feedback control systems. The objective of the ILC/RC is to succeed in making the control system in hardware execute the chosen gait, i.e. the chosen time histories for each joint angle.
- **Problem 2.** If one were not concerned with open loop stable solutions, one could include the feedback control system equations with the robot dynamic equations, and design the gait using as inputs the position command histories given to the feedback control system instead of the torques applied to the robot dynamics. One can include the controller gains as parameters to be optimized (as well as the original model parameters p above) while finding the gait based on a chosen optimality criterion. Of course, the solution is most likely not open loop stable. Because the feedback controller equations have been used in the design of the gait, the hardware would

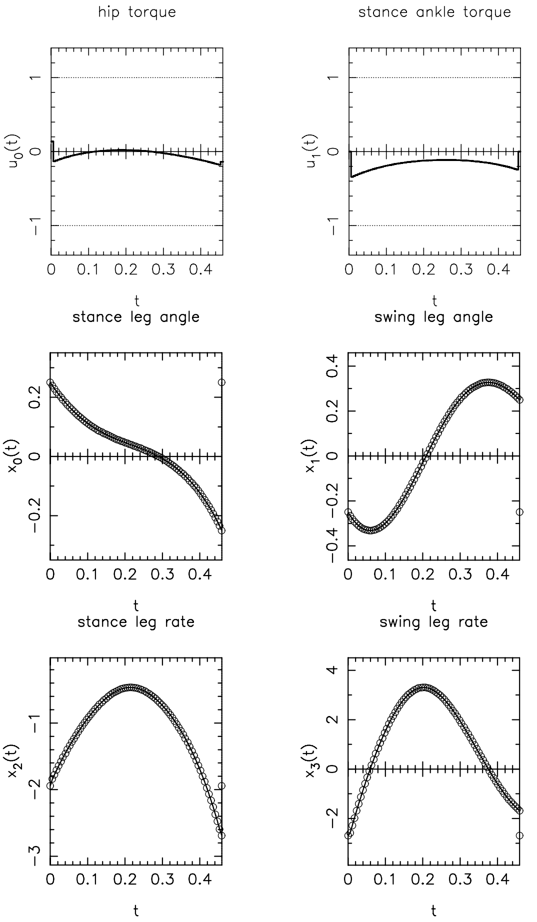


Fig. 4. Torques and states for open-loop stable solution

actually perform the gait desired if the model used for both the robot dynamics and the control system dynamics perfectly represent the behavior of the hardware. The ILC/RC might be used to perfect the execution of the gait by fixing errors caused by inaccurate or incomplete modeling of the robot dynamics, actuators, and control system components.

6.2 Implementation Issues

For both problems the ILC/RC result serve as the basis for implementing the desired motion on level ground. With sufficient stability perhaps this is all that one needs for reasonably level ground. For more uneven terrain, one may next try to design an outer loop that adjusts the commands to the feedback control systems to handle such things as uneven ground. The outer loop is analogous to the upper level trajectory generator in industrial robots and might use additional sensor information such as vision feedback to modify the gait mode to adapt to the terrain.

This paper will consider issues in addressing Problem 1. ILC/RC are methods of solving inverse problems: given a desired output of some dynamic system or component, find the input that would produce that output. It is normally done with the physical robot hardware, but of course one can also use a mathematical model. This gives rise to three possible levels of implementing ILC/RC:

(i) Do the iterations on the hardware.
(ii) Do the iterations on a computer using a model. If the model is good enough, this can work when the solution is applied to the hardware.
(iii) Do (ii) to get an initial command to give in hardware, and then continue the ILC/RC iterations in hardware to correct for any deficiencies in the model.

The numerical studies reported below, directly illustrate (i) where one presumes the computer model is functioning as the real world model. They also automatically illustrate the process one goes through in performing (ii). And then by introducing some changed parameter, i.e. inaccurately modeled parameter, and continuing the learning process one can illustrate (iii).

There are two short cuts for accomplishing (ii), one of which simply eliminates the need for (ii) altogether. These are: use of torque feedforward in the control system design, and do an inverse problem on the controller alone, instead of the complete system.

6.3 Torque Feedforward

Perhaps the most logical implementation of the open loop stable gait solution is to use torque feedforward as in Fig. 5. The solution is a torque history $w(t)$, a desired output history or gait, $y_D(t)$, and its derivative $\dot{y}_D(t)$. Since we consider a decentralized implementation, there is a separate controller for each joint angle with its own desired output history $y_D(t)$. If the actuator actually applies this torque to the robot links, and we give $y_D(t)$ as the command $y_C(t)$, and if the robot model was perfect, then the error signal $e(t)$ would be zero in the block diagram. Then the feedback only starts to function if there is some deviation from the desired trajectory. Several comments apply:

(1) The actuator may not apply the torque we want. If it is a DC motor with voltage input being adjusted by the controller (and the feedforward signal) it will not apply the intended torque, but if one can use current as the input it will accomplish the goal if one knows the motor constants correctly. In order to try to do this, one often uses a current feedback loop around the motor. In addition, the back electro motive force (emf) introduces a rate feedback inherent in the motor. Hence, the actuator has dynamics of its own and will not exactly apply the intended torque.

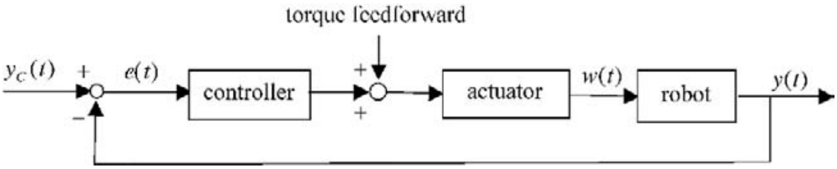


Fig. 5. Feedback control system using torque feedforward

(2) In positioning systems it is often desirable to use rate feedback, meaning the feedback loop takes a measurement of the output, makes a separate measurement of the output rate, which is multiplied by a constant and then the two are added together to produce the signal subtracted from the command to get the "error", $e(t)$. If this is being done, then one must compute what this signal would be when the output is the desired output, and use it as the command given the control system.

(3) The approach totally avoids solving any inverse problem. But it does not fix any errors related to use of an imperfect model or an imperfect actuator, although the feedback loop may make partial adjustments. Then one can apply ILC/RC in hardware to fix remaining errors.

6.4 Inverting Controller Equations

Consider the feedback control block diagram of Fig. 6 including the rate feedback that is typically used in robotics. The usual ILC/RC application solves the inverse problem, given the desired output $y(t) = y_D(t)$, find input $y_C(t)$ to produce it. In the process of having solved for the desired periodic gait, we know more than just the desired output, we also know its derivative, and the torque $w(t)$. Therefore, we can instead solve the inverse problem for the blocks introduced for control: given output $w(t)$, desired position history $y_D(t)$ and its velocity (which together determine the feedback signal) find $y_C(t)$. In the examples below, we use an idealized proportional control system with rate feedback, and in this simple case doing the suggested inverse problem is simple and immediate. Suppose that the actuator can be represented by a simple gain, and this gain can be combined with the proportional control gain in the controller block, and the product called K_1. The feedback signal is $y_D(t) + K_2\dot{y}_D(t)$, where K_2 is the rate feedback gain. Then

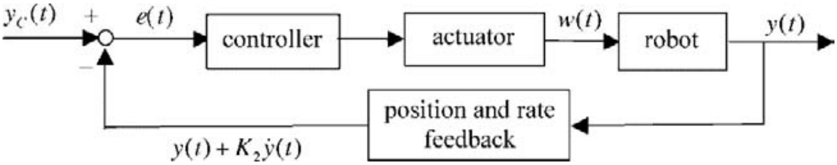


Fig. 6. Feedback control system using rate feedback

$$w(t) = K_1 \left(y_C(t) - [y_D(t) + K_2 \dot{y}_D] \right) \tag{31}$$

Substituting the computed open loop stable time histories, one can solve for the needed command $y_C(t)$ to produce the gait. In simple situations one can do this. In the real world the control system design is likely to be more complicated, requiring inversion of dynamic equations, which is the domain of ILC/RC. Classical control system designers are likely to introduce a low pass filter, possibly as a noise filter, possibly as a way to control the bandwidth of the system to avoid exciting vibrations. They are likely to introduce compensators to modulate the frequency response behavior, which introduces extra dynamics with both poles and zeros. And, as discussed above, the actuator can have some dynamics. Just introducing the back emf of a motor puts in a rate feedback loop feeding the motor, which is missing in the block diagram. If one has a full computer model of all of this, one can aim to solve this inverse problem, and ILC/RC might again be an appropriate method to use.

7 Some Non-Standard ILC/RC Problems

The gait problem does not immediately fit the ILC or RC descriptions. The following two non-standard ILC/RC problems address some of the issues related to gait problems.

Timing Belt Drive Problem. Figure 7 shows a double reduction timing belt drive system that might be used in a copy machine when one needs to have a very uniform velocity of the output shaft. Using a well designed velocity feedback control system, the frequency content of the velocity error is given in Fig. 8 (left) [6, 7]. All of the peaks are related to inaccuracies in the gearing, and include errors that have the periods of one rotation for each shaft, and each belt, including fundamental and harmonics. In addition, the large peaks at 80 Hz and 240 Hz are at the frequencies for tooth meshing of each timing belt. Because gearing is involved, all of these frequencies have a common period which can be used by a repetitive control system to eliminate the peaks. The best experimental result is shown in Fig. 8 (right) where all of the peaks have been eliminated. However, this problem does not completely

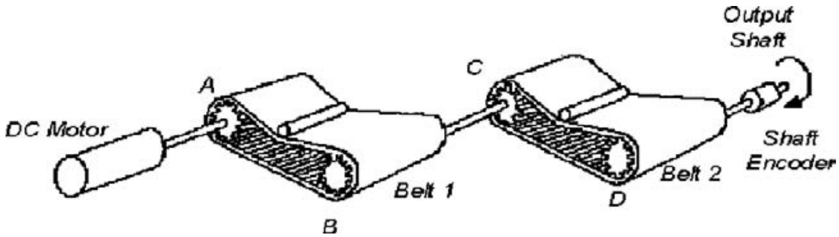


Fig. 7. Double reduction timing belt drive system

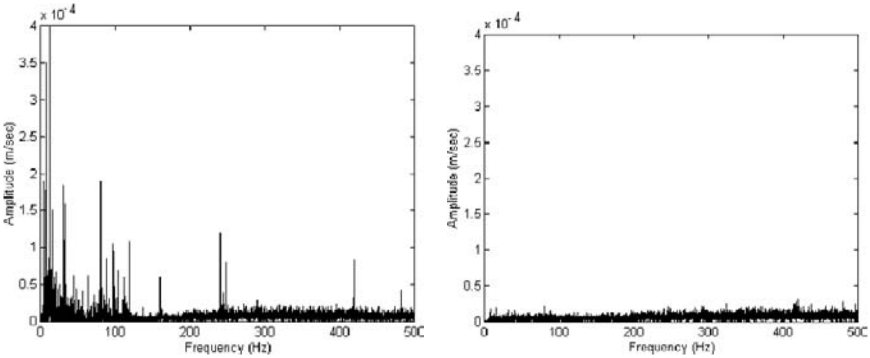


Fig. 8. Frequency spectrum of velocity error of timing belt system using feedback only (*left*) and at repetition 50 using batch phase cancellation RC (*right*)

fit the standard repetitive control assumptions, because as the velocity varies, so does the period. The error to be corrected is actually periodic with the output shaft rotation angle, not with time. To address this issue, these experiments used an index pulse on the output shaft to know when the next period started. The data from the previous period was taken at evenly spaced time steps, and sometimes had more steps than the correct number, and sometimes had less. Some ad hock rule was used to decide what to do when there was missing data for the update. If the period had varied more, one might have made some adjustments to match the time scales for each period. If one has measurements of the angle at each time step, one could do interpolation in the data to explicitly make updates related to each angle instead of each time. There are in fact many repetitive control applications that have this same modeling difficulty.

Cam Problem. Figure 9 (top) shows a cam follower system driven by a DC motor, nominally at a constant speed. The cam is normally designed with its lift curve, the following dwell, the return curve, and the subsequent dwell, all assumed to be made with a prescribed constant cam rotation rate. Of course, as the cam rotates the lever arm to the contact point lengthens and shortens making the resistance to rotation vary with cam angle. The change in resistance to motion is a disturbance to any speed control system. This means that the resulting lift displacement history, dwell durations, and return history as a function of time are not the intended histories or durations. In addition, there are imperfections in the machining of the cam. Suppose we would like to fix these errors by using the velocity control system for the shaft rotation rate to speed up and slow down the cam in such a way as to have the lift and return curves be the intended functions of time, and have the dwell parts be the desired time durations.

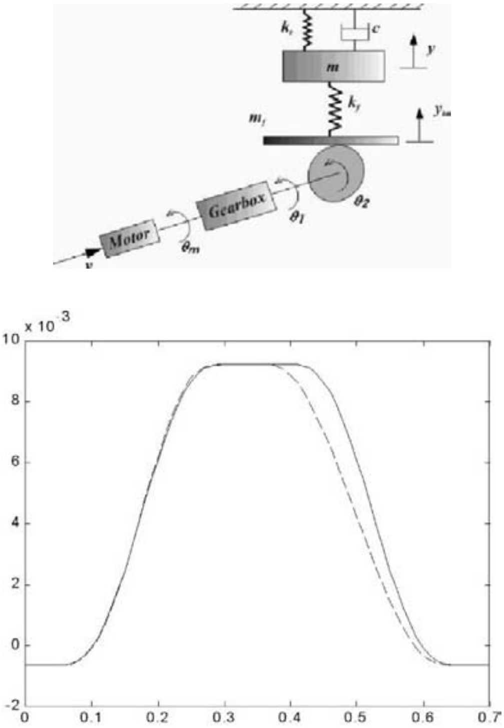


Fig. 9. Cam follower system, and comparison of desired cam displacement (*solid line*) and displacement after 50 repetitions

This problem has several unusual aspects. First, there are four phases, the rise, the dwell, the return, and the following dwell. Second, during the dwell phases presumably the cam has a constant radius so that there is no change in lift, but even if this is not true one is not able to fix the problem by changing the speed. So the only objective to be accomplished in the dwell phases is to make sure they have the right time durations, so that the next phase starts at the right time. Reference [23] reports both simulations and experiments in learning to get the desired output curves. What was done was to learn the lift phase first. Once it converged to a satisfactory error level, the iterations to learn the next phase started. The dwell phases computed the average velocity of the dwell from the end minus start angles, divided by the end minus start times. The error in this average velocity was multiplied by a gain such as ψ and added to the command given in the previous learning cycle to produce the command during the dwell this cycle. Figure 9 (bottom) shows an example of learning iterations after the lift part of the curve has been learned, and iterations are being made to get the top dwell right. Errors in the return are being ignored until the learning of the top dwell is complete. It is fortunate in this problem that the initial conditions for the phase being learning are not heavily dependent on whether the previous phase has been learned already, because the dwell phases allow time for decay of transients. Hence, this problem was treated as a sequence of ILC problems for each phase learning one at a time. The simple learning law of equation (4) was used, but with a chosen time step lead in the error term, creating a linear phase lead compensation [8]. No attempt was made to use a zero-phase low-pass filter to stabilize the process. This was done partly for simplicity, and partly because of nonlinearities in the system. A relatively long sample time for the ILC updates was used to slow the growth of the instability, and the learning process was stopped before the instability started to become evident in the behavior.

8 Approaches to Handling the Non-Standard Nature of Gait Problems

Applying ILC or RC to the gait problem has some of the same issues as the above problems, but introduces additional difficulties as well. The equations are highly nonlinear, and include jump discontinuities. It could be a challenging problem to deal with the nonlinear nature in some direct way, and one would prefer to try to approach the problem as in the nonlinear cam problem, using a simple learning law that might improve the error substantially before an instability sets in, and then stop the learning when the error is a minimum. Note however, that such methods are more successful in ILC than RC, and the gait problem seems to have more relationship to RC problems. We comment that learning high frequency components can be slow and can create stability problems, and discontinuities even in the derivative of variables being controlled introduce some high frequency components. The problem has distinct phases with ground impacts denoting the start of a new phase. As in the timing belt problem, the period or duration of a phase varies with each cycle, the index pulse or impact indicating when the next cycle or phase starts. As in the cam problem the duration of each phase is important. The phases in the cam problem used different learning laws and started with reasonably repeating initial conditions when learned in sequence, so treatment as a sequence of ILC problems worked. But like RC the initial conditions for each period or phase in the gait problem do not repeat until convergence, indicating use of an RC formulation. The RC control laws from phase to phase need not be particularly different, but the fact that the phases have different duration may introduce jumps in the learning process across phase boundaries, the same points where there can be jump discontinuities in velocities. It is not clear how these jumps will affect the learning process. In the next section several different learning laws will be studied. One immediate issue to consider is the question of how to look back at the current phase in the previous cycle to pick which data point is most relevant to the current update. The standard RC looks back at the corresponding time step (modified by the usual one step time delay). But since the duration is different from one run to the next, perhaps it would be better if one looked at data corresponding to the same percentage of time in the cycle.

9 Numerical Investigation of Possible Learning Schemes

In this section we present four different approaches to the repetitive control of walking motion and discuss their effects on the example of the stiff-legged biped described in Sect. 4 with the simple feedback controller (30). Each law describes an algorithm to compute the inputs $y_{C,j}(kT)$ to the feedback controller at sample times k $(k = 1, ...N_j)$ of cycle j, depending on the inputs of the previous cycle $y_{C,j-1}$, errors of the previous cycle etc. All these laws have in common that they rely on a synchronization of the phase with a significant event occurring once per cycle – in this case the touchdown of the swing foot, which is also very easy to detect in practice. This event is starting a new phase with relative time in the phase equal to zero. Therefore – even though the problem is closer to RC - we prefer to use a notation which is more of ILC type with the sampling time index k reset to 0 in every cycle. Note that this is purely by mathematical convenience and does not influence results.

All RC laws presented depend on one or more gains that can be tuned and typically have a large impact on the performance. We have investigated some sets of constants for each law (without doing a real optimization); and we present for each law the best constants we have found so far. No proof of convergence is given for these learning laws. Given the nonlinear nature of the problem with multiple phases and jump discontinuities it might be very difficult to establish such proofs. However, as noted in [8], good learning transients can be more important than stability (i.e convergence of the learning scheme). Furthermore [9] demonstrates that unstable learning laws can be very useful in applications.

In order to allow a comparison of the different laws, we display the following result plots:

- for each law, the error histories of angles θ_1 and θ_2 over time (shifted by the duration of the first cycle), see Figs. 10, 12, 14, and 16.
- for each law, a comparison of the outputs for angle θ_1, the corresponding reference trajectories, and the commanded trajectory at the beginning ($t = 0\,\mathrm{s}...2\,\mathrm{s}$) and at the end ($t = 18\,\mathrm{s}...20\,\mathrm{s}$) of the investigated learning process, see Figs. 11, 13, 15, and 17
- the RMS errors for each cycle of all four laws, in overview Fig. 18
- the development of cycle times α_j over a number of cycles for all four laws, in Fig. 19.

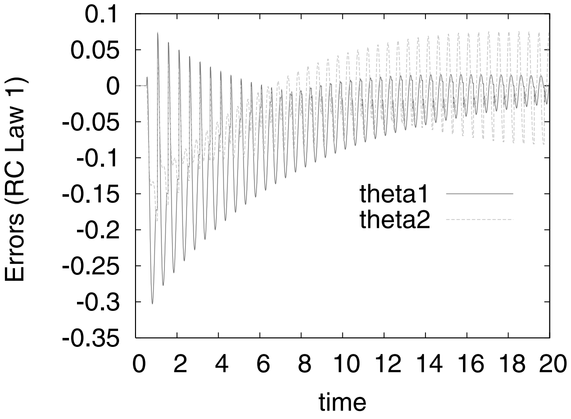


Fig. 10. Error histories of angles θ_1 and θ_2 using RC law 1

9.1 RC Law 1

$$y_{C,j}(kT) = y_{C,j-1}(kT) + \psi_1 e_{j-1}((k+1)T) \tag{32}$$

$$\text{with} \quad e_{j-1}(kT) = y_D(kT) - y_{j-1}(kT) \tag{33}$$

This first law is the most simple and straightforward one: The error term e_{j-1} that is used to correct the input of the system only compares the actual trajectory of the previous cycle and the reference trajectory at identical time points. The knowledge about the duration of the previous cycle is not used. The constant ψ_1 is chosen as 0.1.

As shown in Fig. 10, the law works quite well – despite its simplicity – to reduce the position variable errors (especially the RMS error of the relative hip angle θ_1 is reduced, while the error of the absolute stance leg angle θ_2 goes up a little again). However, this law does a poor job correcting the wrong initial cycle times (Fig. 19). Figure 11 gives some more details about the learning process, since it shows a comparison of the actual output angle θ_1, the desired angle $\theta_{1,D}$, and the commanded angle $\theta_{1,C}$ at 2 different intervals of the learning process. The upper picture shows the first few cycles: as in all other cases we start by commanding the desired trajectory, with the result that the output trajectory is far off. The lower picture shows cycles between 18 and 20 s (after roughly 36–40 cycles modified by law 1), with the desired and actual trajectory being quite close, and the commanded trajectory being quite different. These pictures also show the adjustment of phase times (the reference trajectory does a step change to zero after termination of the step, while the actual output step is still not finished) with a large error (18%) in the beginning, and a smaller, but still a significant difference (8%) at the end.

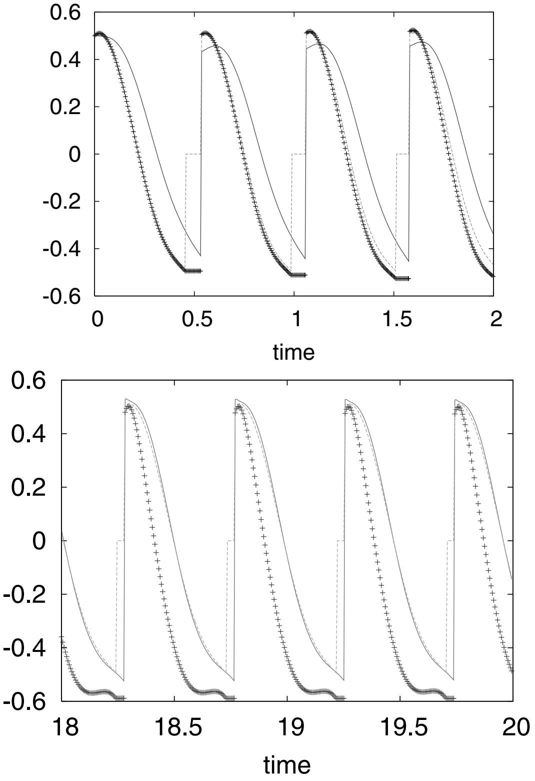


Fig. 11. Comparison of output trajectory (*solid line*), reference trajectory (*dashed lines*) and commanded trajectory (*crosses*) for angle θ_1 using RC law 1, at beginning (*top figure*) and end (*bottom figure*) of learning process

9.2 RC Law 2

$$y_{C,j}(kT) = c_1(\alpha_{j-1}) \cdot (y_{C,j-1}(\alpha_{j-1}kT) + \psi_1 e_{j-1}((k+1)T) \tag{34}$$

$$\text{with} \quad e_{j-1}(kT) = y_D(kT) - y_{j-1}(\alpha_{j-1}kT) \tag{35}$$

$$\alpha_{j-1} = \frac{T_{cycle,j-1}}{T_{cycle,ref}} \; . \tag{36}$$

In this law, the constant ψ_1 is again 0.1. This second law uses the factor α_{j-1} to introduce information about the previous cycle time and by computing the error between corresponding (and not identical) points of the current and reference cycle. For the evaluation of the right hand side of eqn. (35), linear interpolation is used between sample points. But it is important to note that while this error computation may be the more logical one, it does not punish

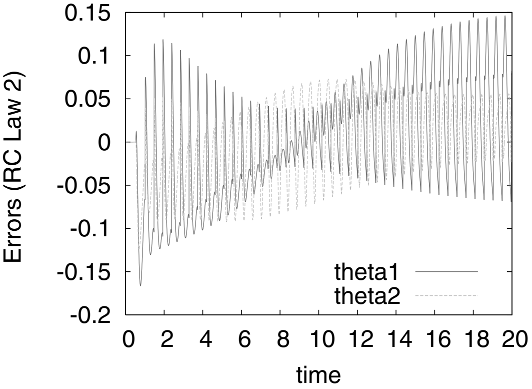


Fig. 12. Error histories of angles θ_1 and θ_2 using RC law 2

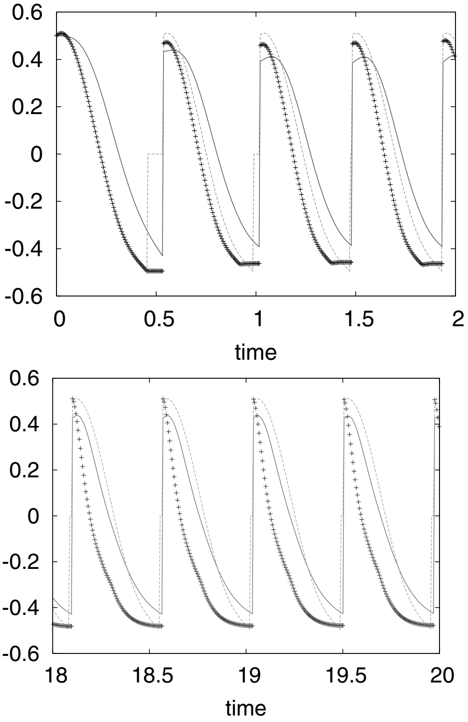


Fig. 13. Comparison of output trajectory (*solid line*), reference trajectory (*dashed lines*) and commanded trajectory (*crosses*) for angle θ_1 using RC law 2, at beginning (*top*) and end (*bottom*) of learning process

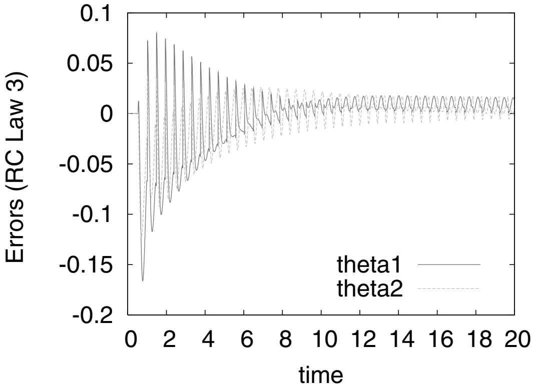


Fig. 14. Error histories of angles θ_1 and θ_2 using RC law 3

any errors of the cycle time; in fact a linearly scaled slower or faster cycle would actually lead to an error of zero. So it is important to introduce some correcting factor for wrong α_{j-1} which we do in the form of c_1, which is a function of α_{j-1} that has to satisfy $c_1(1) = 1$. Again, there are obviously many possible choices, and in this case we have set it to

$$c_1(\alpha_{j-1}) = \frac{1}{\sqrt{\alpha_{j-1}}} \,. \tag{37}$$

As shown in Fig. 19, this law does a better job than the first one in correcting the cycle duration. The reduction of position errors is roughly the same as for the first law with the inverse effect on the two degrees of freedom: this time errors of the stance leg angle are corrected better than errors of the relative hip angle (see Fig. 18). However, as in the case of law 1, there is no continuous reduction in one of the position errors, and the development beyond the investigated time frame is unclear. But we expect that it would be possible to improve the performance of this law with a tuned factor $c_1(\alpha)$.

There is however another possibility to improve the adjustment of cycle times (instead of using factor c_1) which is the inclusion of error terms on velocity level. The performance of this approach is investigated in the following two RC laws, numbers 3 and 4.

9.3 RC Law 3

$$\begin{aligned} y_{C,j}(kT) &= y_{C,j-1}(\alpha_{j-1}kT) + \psi_1 e_{pos,j-1}((k+1)T) \\ &\quad + \psi_2 e_{vel,j-1}((k+1)T) && (38) \\ \text{with} \quad e_{pos,j-1}(kT) &= e_{j-1}(kT) = y_D(kT) - y_{j-1}(\alpha_{j-1}kT) && (39) \\ e_{vel,j-1}(kT) &= \dot{y}_D(kT) - \dot{y}_{j-1}(\alpha_{j-1}kT)) && (40) \\ \text{and} \quad & \alpha_{j-1} \text{ as above.} && (41) \end{aligned}$$

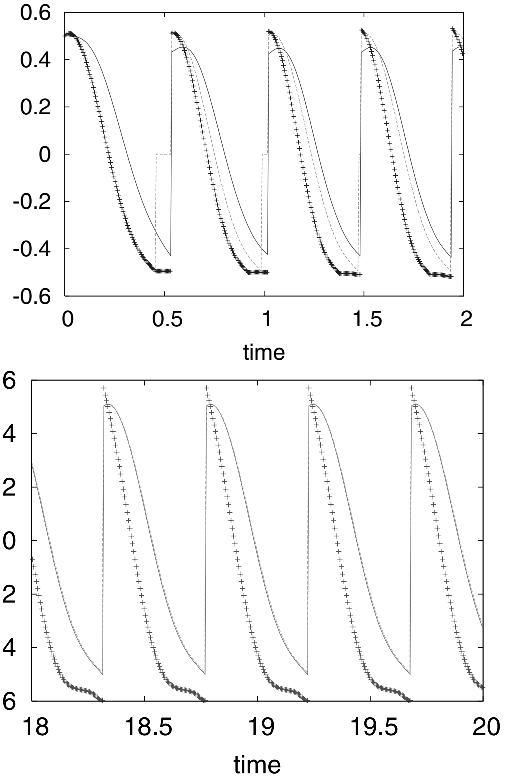


Fig. 15. Comparison of output trajectory (*solid line*), reference trajectory (*dashed lines*) and commanded trajectory (*crosses*) for angle θ_1 using RC law 3, at beginning (*top*) and end (*bottom*) of learning process

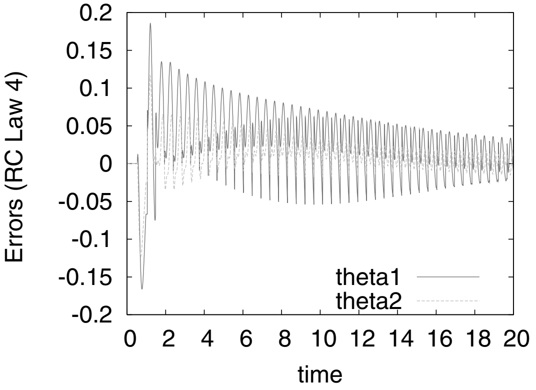


Fig. 16. Error histories of angles θ_1 and θ_2 using RC law 4

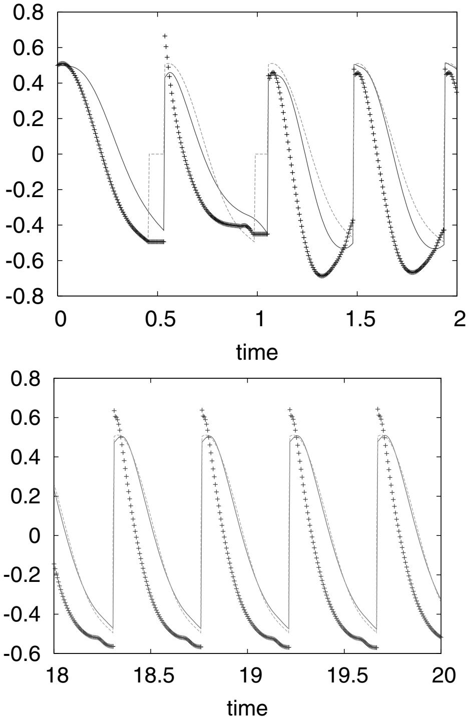


Fig. 17. Comparison of output trajectory (*solid line*), reference trajectory (*dashed lines*) and commanded trajectory (*crosses*) for angle θ_1 using RC law 4, at beginning (*top*) and end (*bottom*) of learning process

This law stems from the above RC law 2 skipping the leading factor c_1, but adding another correcting term which is proportional to the velocity errors. The constants are chosen as $\psi_1 = 0.1$, and $\psi_2 = 0.01$. As the Figs. 14, 15, 18 and 19 show, this law works extremely well both in correcting state errors and cycle duration. The cycle duration α_j is correct to 3 digits after only 3 cycles. The difference between the desired and actual output angle θ_1 is barely visible after 18 s (in the lower part of Fig. 15).

9.4 RC Law 4

$$\begin{aligned} y_{C,j}(kT) &= y_{C,j-1}(\alpha_{j-1}kT) + \psi_1 e_{pos,j-1}((k+1)T) \\ &\quad + \psi_2 (1-\alpha_{j-1})^2 e_{vel,j-1}((k+1)T) \qquad (42) \\ \text{with} \quad & \alpha_{j-1}, e_{pos,j-1}, e_{pos,j-1} \text{ as above.} \qquad (43) \end{aligned}$$

RC law 4 is a modified version of law 3 with an additional factor in front of the velocity error term. The motivation behind this was to avoid asking too

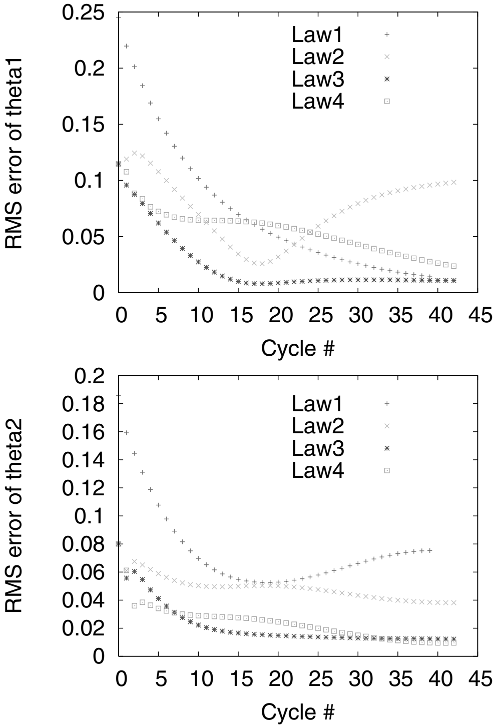


Fig. 18. RMS errors of θ_1 and θ_2 for RC laws 1–4

much from the RC controller, namely to correct errors in $2n$ state variables (the positions and velocities) while only modifying n input variables (the commanded position histories). The constants chosen in this case are $\psi_1 = 0.1$ and $\psi_2 = 8.0$. As Figs. 16, 17, 18 and 19 show, the performance of this law is also very good, comparable to that of law 3. In the particular case investigated here, law 4 seems to do slightly better on the absolute stance angle, and law 3 does better on the realtive hip angle corrections. While the start of the learning process according to law 4 (Fig. 17) clearly shows a different behavior than in the case of law 3, there is a clear resemblance of the commanded trajectories after 20s of the learning processes following laws 3 and 4.

9.5 Discussion of Simulation Results

Four different methods of RC for gaits have been studied here. One can either learn from the error in the previous cycle for the corresponding time step (as implemeted in law 1), or for the corresponding percent of the time used for that phase in that cycle (laws 2–4). The second approach is expected to significantly improve the size of deviations tolerated by the algorithm before the process goes unstable. However, since (without any other correcting terms)

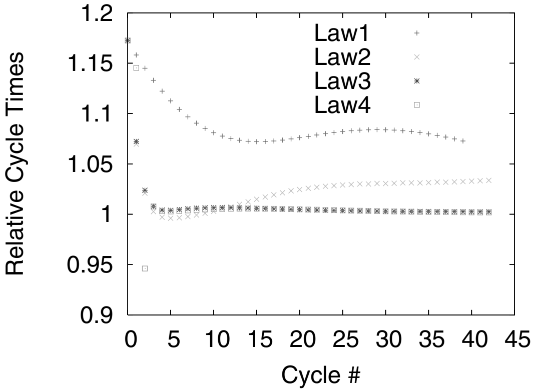


Fig. 19. Development of cycle times for RC laws 1–4

it does not penalize the error in the duration of the phase, one would like to introduce some extra aspect to the learning process to do so, e.g. introduce a factor depending on the relative cycle change (as in law 2).

The two most promising of the four RC approaches for gait problems that we investigated seemed to be the two laws that were based on both position and velocity errors in the previous cycle (laws 3 and 4). For the computation of errors and new commands the duration of the previous cycle (relative to the desired reference cycle time) was explicitely taken into account in both cases. In digital control one would not normally include both terms (position and velocity errors) because the number of input variables, i.e. the commands given for each time step, are not enough to independently control both the position and the velocity of the output at each time step. Hence in law 4 we included (in contrast to law 3) a cancellation of the velocity error term in the case of correct cycle time adjustment. After these first results for a specific walker and a specific feedback control law, it is hard to judge which of the two approaches might perform better in general. We think that both laws deserve further investigation on more test examples, also including more extensive studies of the most suitable choices of gains in the laws.

10 Conclusions

The concepts of ILC and RC aim at improving the performance of feedback control systems by adjusting the commands given to them for the execution of a repetitive or cyclic maneuver. The purpose of this paper is twofold: first, to discuss the general issues of transferring the ideas of ILC/RC to gait problems; and second, to present particular implementations in the form of four RC laws applied to a simple robot model with simple feedback control laws.

It has been shown that the problem of fixing errors in hardware execution of periodic gaits does not perfectly fit the problem formulations for either ILC or RC, but is closest to that of RC. The gait problem differs in that it must be separated into phases that start at foot strike, and that the durations of the phases can vary each cycle until reaching convergence to the desired gait.

Four different methods of addressing these extra issues have been presented. The results are summarized in Sect. 9.5 above. The two most promising RC approaches investigated included both an update on the command based on the error of the previous cycle, and a second update term based on the velocity error, where we also studied including a cancellation of the velocity term in the case of correct cycle time adjustment. Both laws deliver excellent results of adjusting cycle time and eliminating tracking errors. We intend to perform further investigations along these lines, involving other robot models and combining the concept of RC with other underlying feedback control systems used in contemporary walking robots.

We note that ILC and RC are notorious for exhibiting substantial decay in the error followed eventually by growth of the error, and much of the literature seeks ways to address this problem. No attempt has been made here to determine whether the RC laws result in asymptotical convergence to the desired solution when applied to the nonlinear robot dynamic equations. However, even if the laws are unstable, they may be very useful in practice. One simply uses the RC law to decrease the error, and turns it off when the error starts to grow – an approach that is used in the computer disk drive industry to good effect.

Our results suggest that with appropriate modifications it will be possible to use repetitive control concepts to significantly improve the execution of chosen periodic gaits by real walking robots.

Acknowledgments

The first author would like to express his thanks to the Alexander von Humboldt Foundation for a research award that partially supported the research presented here. The second author gratefully acknowledges financial support by the State of Baden-Württemberg through the Margarete von Wrangell program.

References

[1] *Proceedings of the IEEE-RAS International Conference on Humanoid Robots (Humanoids, 2005).* IEEE, Tsukuba, Japan, 2005.

[2] K. Berns. Walking machines catalogue. *http://www.fzi.de/ipt/WMC/walking-machines-katalog/walking-machines-katalog.html*, 2000.

[3] Z. Bien and J.-X. Xu, editors. *Iterative Learning Control: Analysis, Design, Integration, and Applications.* Kluwer Academic Publishers, Boston, 1998.

[4] H. Elci, R. W. Longman, M. Phan, J.-N. Juang, and R. Ugoletti. Discrete frequency based learning control for precision motion control. In *Proceedings of the 1994 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 2767–2773, San Antonio, TX, Oct. 1994.

[5] Honda. Asimo. *http://asimo.honda.com*, 2006.

[6] Y. P. Hsin, R. W. Longman, E. J. Solcz, and J. de Jong. Experimental comparisons of four repetitive control algorithms. In *Proceedings of the 31st Annual Conference on Information Sciences and Systems*, pp. 854–860, Johns Hopkins University, Department of Electrical and Computer Engineering, Baltimore, Maryland, 1997.

[7] Y. P. Hsin, R. W. Longman, E. J. Solcz, and J. de Jong. Experiments bridging learning and repetitive control. *Advances in the Astronautical Sciences*, 95:671–690, 1997.

[8] R. W. Longman. Iterative learning control and repetitive control for engineering practice. *International Journal of Control*, 73(10):930–954, 2000.

[9] R. W. Longman and Y.-C. Huang. Use of unstable repetitive control for improved tracking accuracy. *Adaptive Structures and Composite Materials: Analysis and Applications, ASME*, AD-45, MD-54:315–324, Nov. 1994.

[10] R. W. Longman and Y.-C. Huang. The phenomenon of apparent convergence followed by divergence in learning and repetitive control. *Intelligent Automation and Soft Computing, Special Issue on Learning and Repetitive Control*, 8(2):107–128, 2002.

[11] R. W. Longman, Y.-T. Peng, T. Kwon, H. Lus, R. Betti, and J.-N. Juang. Adaptive inverse iterative learning control. *Advances in the Astronautical Sciences*, 114:115–134, 2003.

[12] T. McGeer. Passive dynamic walking. *International Journal of Robotics Research*, 9:62–82, 1990.

[13] K. D. Mombaur. Performing open-loop stable flip-flops – an example for stability optimization and robustness analysis of fast periodic motions. In *Fast Motions in Robotics and Biomechanics – Optimization and Feedback Control*, Lecture Notes in Control and Information Science. Springer, 2006.

[14] K. D. Mombaur, H. G. Bock, J. P. Schlöder, and R. W. Longman. Human-like actuated walking that is asymptotically stable without feedback. In *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 4128–4133, Seoul, Korea, May 2001.

[15] K. D. Mombaur, H. G. Bock, J. P. Schlöder, and R. W. Longman. Open-loop stability – a new paradigm for periodic optimal control and analysis of walking mechanisms. In *Proceedings of IEEE CIS-RAM 04, Singapore*, 2004.

[16] K. D. Mombaur, H. G. Bock, J. P. Schlöder, and R. W. Longman. Self-stabilizing somersaults. *IEEE Transactions on Robotics*, 21(6), Dec. 2005.

[17] K. D. Mombaur, R. W. Longman, H. G. Bock, and J. P. Schlöder. Open-loop stable running. *Robotica*, 23(01):21–33, January 2005.

[18] K. D. Mombaur, R. W. Longman, J. P. Schlöder, and H. G. Bock. Optimizing spring-damper design in human-like walking that is asymptotically stable without feedback. 2006, submitted.

[19] K. Moore and J.-X. Xu. Special issue on iterative learning control. *International Journal of Control*, 73(10), July 2000.

[20] S. J. Oh, R. W. Longman, and Y.-P. Hsin. The possible block diagram configurations for repetitive control to cancel periodic plant or measurement dis-

turbances. In *Proceedings of the 2004 AIAA/AAS Astrodynamics Specialist Conference*, Providence, RI, Aug. 2004.

[21] B. Panomruttanarug and R. W. Longman. Frequency based optimal design of FIR zero-phase filters and compensators for robust repetitive control. *Advances in the Astronautical Sciences*, 123, to appear.

[22] B. Panomruttanarug and R. W. Longman. Repetitive controller design using optimization in the frequency domain. In *Proceedings of the 2004 AIAA/AAS Astrodynamics Specialist Conference*, Providence, RI, Aug. 2004.

[23] N. Phetkong, M. S. Chew, and R. W. Longman. Morphing mechanisms part 1: Using iterative learning control to morph cam follower motion. *American Journal of Applied Sciences*, 2(5):897–903, 2005.

[24] A. M. Plotnik and R. W. Longman. Subtleties in the use of zero-phase low-pass filtering and cliff filtering in learning control. *Advances in the Astronautical Sciences*, 103:673–692, 1999.

[25] Sony. Qrio – Sony Dream Robot. *http://www.sony.net/SonyInfo/QRIO/*, 2006.

Actuation System and Control Concept for a Running Biped

T. Luksch, K. Berns, and F. Flörchinger

Technische Universität Kaiserslautern, Germany
{luksch,berns}@informatik.uni-kl.de, floerch@web.de

Summary. Dynamic walking with two-legged robots is still an unsolved problem of todays robotics research. Beside finding mathematical models for the walking process, suitable mechanical designs and control methods must be found. This paper presents concepts for the latter two points. As biological walking makes use of the elastic properties of e.g. tendons and muscles, a joint design using a pneumatic rotational spring with adjustable stiffness is proposed. Equations to model the spring's dynamics as well as the supporting sensor systems and electronics are presented. For controlling the robot a behaviour-based approach is suggested.

1 Introduction

Looking at todays two-legged robots and their way of walking, one might get the impression that there is still a long way to go before human-like running will be achieved. Sereval reasons for this can be found: most bipeds walk in a static manner shifting their weight carefully to always maintain a stable position; no dynamic effects are taken advantage of. Energy consumption is much higher compared to natural movements as no energy from the dynamics of the system is reused. More often than not classical robot mechanics known from industrial applications are used for construction whereas more exotic concepts like elastic elements are rarely involved. The control algorithms applied are often based on a complete physical model of the robot and its environment and do seldom allow the freedom of unknown ground conditions or other external disturbances. Fast running motion is impossible for most of these machines not only because the occurring impacts could destroy the mechanics but also because the problem of controlling fast dynamical biped locomotion in unstructured environment is still unsolved.

This paper presents the first steps of the development of a two-legged walking robot addressing some of the above topics, namely the actuation system, the system architecture and a behaviour-based control concept. As some of the ideas in the design originate from natural human walking, some aspects of what biology has come up with should be mentioned.

It has been shown that mammals use the elastic components in their legs (tendons, ligaments and muscles) to run economically, while maintaining consistent support mechanics across various surfaces. Representations of the running leg as a simple spring have described the mechanics of a running leg remarkably well. McMahon and Cheng describe the spring leg having two stiffnesses: k_{leg} as the actual leg stiffness and k_{vert} as the effective vertical stiffness of the runner [12]. Kerdok et al. examine the effect of different ground stiffnesses on the energetics and mechanics of human running [8]. Elastic elements seem to play a crucial role in natural fast locomotion concerning energy consumption, robustness and simpleness of control.

This paper is structured as follows: Section 2 discusses some of the more biologically motivated research efforts found in the literature. Mechanical aspects as well as control architectures are looked at. The following section will introduce the concepts of a leg design for fast locomotion. Several aspects like actuation, elastic elements or electronics will be mentioned with the focus on the knee construction as one of the most important joints. Section 4 will address the control problem. A behaviour-based architecture is proposed and first results of periodic movement control in simulation are presented. The paper concludes with a summary and outlook.

2 State-of-the-Art

Several projects can be found in the literature trying to apply more ideas from nature than most other walking robot efforts. This section will introduce some of these research projects, highlighting first the mechanical aspects, followed by control designs for fast or dynamically walking robots.

2.1 Mechanics

Most contributions to constructing walking machines that are able to move energy efficiently, run or even jump include elastic element to store energy, absorb impacts or support actuators in producing high torques. This follows the concepts found in nature as mentioned in the previous section.

The Robotics and Multibody Mechanics Group of D. Lefeber in Brussels is building the biped robot *Lucy* (Fig. 1(c)) featuring artificial pneumatic muscles [17], [5]. Recent efforts in the field of actuated passive walking have produced an actuator called MACCEPA (*The Mechanically Adjustable Compliance and Controllable Equilibrium Position Actuator*, Fig. 1(a)). This low cost design is able to set a desired angle with an adjustable compliance or stiffness. The low weight, easy control and low stiffness make it an appropriate choice for actuated passive walking, but will not allow precise or highly dynamical movement. The passive dynamic walker *Veronica* (Fig. 1(b)) featuring these actuators is currently under development.

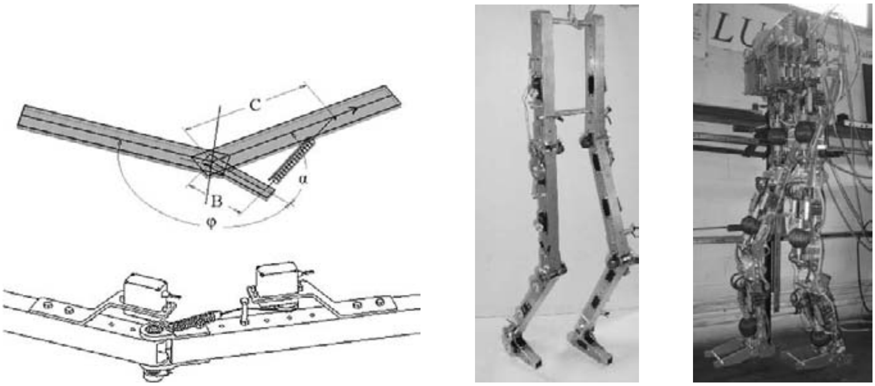


Fig. 1. (**a**) The MACCEPA actuator – (**b**) Veronica – (**c**) Lucy

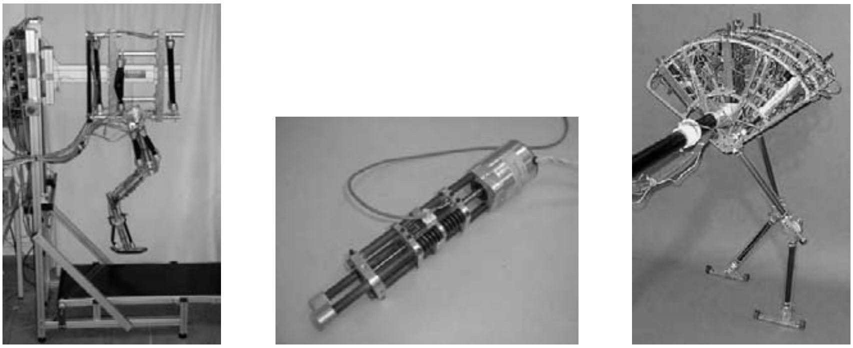


Fig. 2. (**a**) PANTER leg – (**b**) Series Elastic Actuator – (**c**) Spring Flamingo

Another project using fluidic pneumatic muscles is running at FZI in Karlsruhe. A prototype leg called *Panter* (Fig. 2(a)) for fast running locomotion has been constructed [1]. Further research is done on the dynamics of the artificial muscles and their behaviour using a quick release mechanism [9]. Among the advantages of these actuators are implicit compliance, behaviour close to the biological muscle and high forces. On the other side they need compressed air supply and are not easy to control.

Already in the 1980s researchers at MIT developed the *Series Elastic Actuator* [18] (Fig. 2(b)). This linear electric actuator is coupled with a spring-damper system. Walking robot prototypes constructed with these actuators include the *Spring Flamingo* (Fig. 2(c)) and the biped walker *M2* [14]. The Series Elastic Actuator is now distributed by Yobotics, Inc[1].

A few four-legged walking machines with elastic elements are developed by F. Iida at R. Pfeifer's institute in Zurich. The main research interest lies in the

[1] www.yobotics.com

self-stabilisation process. Experiments are carried out using robots equipped with servomotors in combination with springs controlled by synchronized oscillators [7].

2.2 Control

In Japan many research projects on biped walking exist which mostly follow the traditional concepts of robot construction and control. Honda is developing the robot Asimo and recently managed a running-like locomotion mode. But these motions and trajectories are completely pre-calculated and easily disturbed by small unknown obstacles. Similar projects outside Japan have been done by e.g. F. Pfeiffer (*Johnnie*, [13]) or the Kaist company (*KHR-1*, [10]).

Another group working on the dynamical pre-calculation of joint trajectories is D. Lefeber's group in Brussels. Walking and running trajectories for biped robots have been calculated [16]. The running cycle has been divided into three phases including a flight phase without foot contact. The resulting force trajectories for each joint are to be combined with the output of stability reflexes.

The implicit control of passive dynamic walkers is emerging from their sophisticated mechanics. Limb lengths and weights are chosen in a way that the robot can walk down a slop without any actuators. First in-depth experiments and theoretical examinations on this topic have been done by T. McGeer [11]. Recent work includes the research projects at TU Delft by M. Wisse et al. [19]. Some machines of this group are equipped with additional actuators like fluidic muscles to substitute the potential energy gained from the slope, e.g. the robot Denise as shown in Fig. 3(a). Another example is the machine by A. Ruina et al. at Cornell University (Fig. 3(b)) using springs and DC motors [4]. The control algorithms involved are straight forward as they only have to

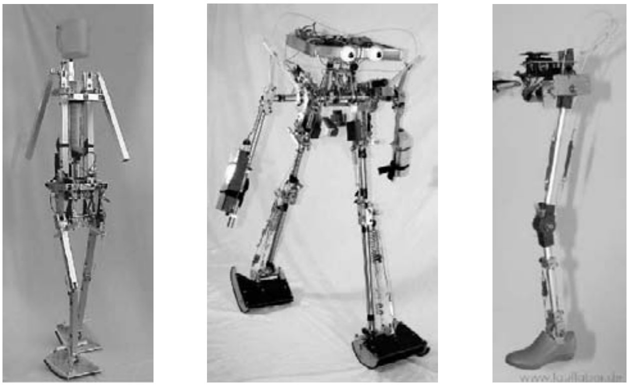


Fig. 3. (**a**) Denise from TU Delft – (**b**) Cornell Biped – (**c**) BioLeg2 from Jena

support the implicit control of the mechanics and have even been shown to be managed by a computational unit as simple as the LEGO Mindstorms RCX computer.

The walking laboratory of A. Seyfarth at Jena is examining the physics of human walking, running and jumping [15] as well as the control of legged robots. It has been shown that a rather simple control strategy can suffice to create walking or running behaviour if the mechanics implicitly supports it. Several small single leg prototypes have been build (e.g. Fig. 3(c)) including only one actuator driving the hip movement and a spring generating the knee motions [6].

The research cited above as well as contributions by biologists, e.g. [20], seem to suggest that the control of walking robots can drastically be simplified by clever mechanics (Embodiment). Especially including elastic elements could result in a more natural and faster walking behaviour. The following sections introduce considerations on building and controlling such a two-legged walking machine.

3 Leg Prototype for Fast Locomotion

This section introduces a possible mechatronical construction for a two-legged walking machine. After discussing preconditions and the resulting design decisions, special attention is given to the knee layout focusing on a pneumatic spring unit. Some considerations on the necessary electronics and sensor systems follow.

3.1 Design of an Elastic Joint Actuation System

The long term and admittingly ambitious goal of this research project is to build a walking robot that can stand, walk, run and jump. A first prototype will consist only of a single leg with an actuated joint in the hip and the knee. As the knee is the most demanding joint in a biped locomotion system, the design of an elastic knee joint is the initial task to solve. The limb lengths are assumed to be 50 cm, the body mass to be 20 kg. As jumping will put the most stress on the mechanics this task will serve as calculation basis.

Knee Design

The knee design has to meet several requirements:

- The construction should allow the lower leg to freely swing back without using any additional energy. To achieve this either a direct DC motor without gear (Fig. 4(b)) or a motor with gear, a clutch and additional loop control (Fig. 4(c)) could be used. As a design decision is to put as much implicit control in the mechanics as possible the first variant would

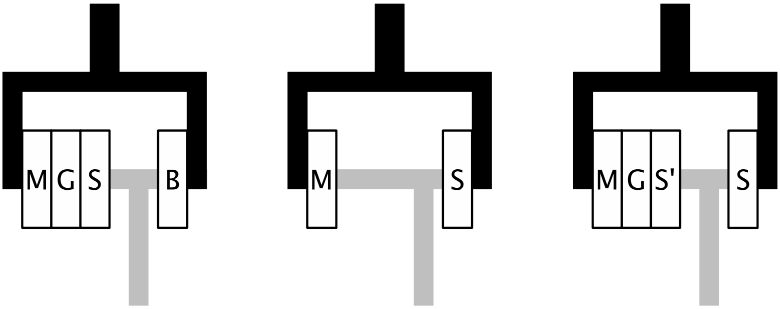


Fig. 4. (**a**) serial spring – (**b**) par. spring, direct drive – (**c**) par. spring, gearbox. (with M: DC motor, G: gearbox, S: adjustable spring, S': stiff spring, B: bearing)

be preferable. But as there are not DC motors on the market producing the high torques needed, a compromise will probably be implemented in the first testbed.

- The system should be able to deal with hard impacts. Again a gear box would be the second choice as it would have to be decoupled with e.g. a spring of high stiffness and a clutch (S' in Fig. 4(c)).
- To support the motor in producing enough torque an elastic element is to be included to store energy. Such a spring could be mounted serial (Fig. 4(a)) or parallel (Fig. 4(b), (c)) to the motor. In the first case the motor would have to counter-act to the spring energy. The second variant has the disadvantage of a fixed equilibrium position. But as the system is to be energy efficient, the second variant is chosen.

The spring mounted parallel to the drive should possess a variable stiffness to adapt to the current situation. In the case of normal walking the spring should be soft to allow the leg to swing forward and harder before ground impact and support phase. In the case of jumping the spring should be stiff for the whole cycle. These precondition and the goal to achieve a compact actuator unit mainly located in the joint itself led to the development of a pneumatic rotatory spring with variable volume.

Pneumatic Spring

The schematic design of the pneumatic spring can be seen in Fig. 5(b). The piston is attached to the shaft in the middle of the spring and compresses the volume V_2 with inceasing rotation ϕ. Variable separations can be added to decrease the initial volume $V_1(\phi = 0)$. The separations are inserted at angles k where a smaller k results in higher stiffness of the spring. Each variable separation can be switched open by an outlet valve. This way the working volume can be changed and the spring's stiffness can be adapted.

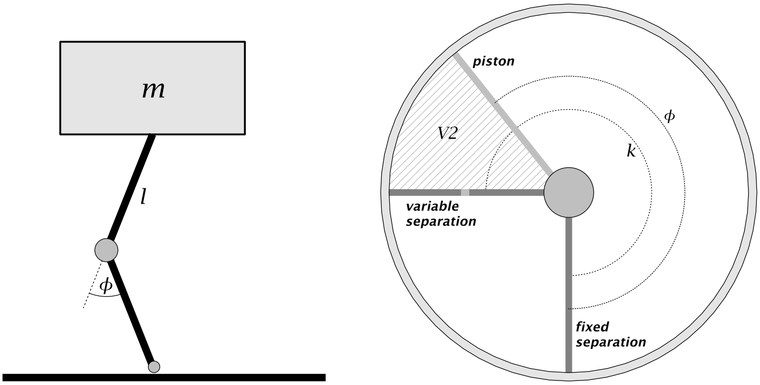


Fig. 5. (**a**) first prototype – (**b**) schematic drawing of pneumatic spring

The following calculation are based on several assumptions: the testbed robot (Fig. 5(a)) will consist of only one leg; the body mass m of the robot is 20 kg; the limb length l is 500 mm; the robot should be able to jump 20 cm high; the spring should be able to store the impact energy of a 70 cm fall.

The torque $M_m(\phi)$ on the knee joint created by the body mass given a knee angle ϕ can be given as

$$M_m = mg\sin(\frac{1}{2}\phi)l \tag{1}$$

resulting in a maximum torque of 85 Nm at a defined maximum angle of 120 degree. The aim of the pneumatic spring is to compensate this static torque.

Starting from the ideal gas equation

$$pV = MRT$$

we assume the adiabatic case, i.e. there is no energy transferred as heat to the environment. This leads to the adiabatic state equation following the Poisson Law:

$$pV^\kappa = const \Rightarrow p_1 V_1^\kappa = p_2 V_2^\kappa \Rightarrow p_2 = p_1 \left(\frac{V_1}{V_2}\right)^\kappa$$

with κ being the adiabatic exponent ($\kappa = 1.67$ for one atom gases like argon, $\kappa = 1.4$ for two atom gases and $\kappa = 1$ for non-adiabatic processes). To calculate the pressure p_2 we have to know the start volume $V_1 = \frac{Vk}{2\pi}$ and the start pressure p_1:

$$p_2 = p_1 \frac{\left(\frac{k}{2\pi}V\right)^\kappa}{\left(\frac{k-\phi}{2\pi}V\right)^\kappa}$$

with V being the volume of the complete spring. The resulting torque only depends on this pressure p_2 and the radius r and the applying lever h:

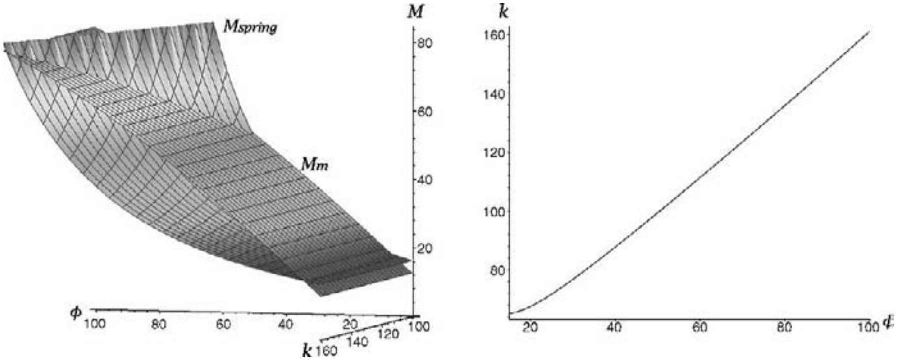


Fig. 6. (**a**) static torque M_m and spring torque M_{spring} – (**b**) optimal values of k

$$M_{spring} = 0.110^{-8} p_2 \pi r^2 h$$

From this torque and the static torque M_m an optimal placement k of the variable separation can be calculated at which the torque M_m is compensated. Figure 6(a) visualizes the two torques. The resulting intersection curve representing the optimal choices for k can be calculated as

$$k_{opt}(\phi) = \frac{\phi}{1 - \left(\frac{0.110^{10} mg \sin(\frac{\phi+\phi_0}{\pi}) l}{p_1 \pi r^2 h}\right)^{-\frac{1}{\kappa}}}$$

and is plotted in Fig. 6(b). This nearly linear correlation can be approximated with multiple switchable variable separations. Further calculations have been done for the dimensioning of the shaft and sealing. The resulting design is shown in Fig. 7.

Present development focuses on the integration of the motor and the pneumatic spring following Fig. 4(c). A brushless DC motor by Macon serves as actuator combined with a low gear transmission ratio. This unit is coupled with the driving axis by a clutch and an additional spring of high stiffness. Position encoders are placed at the motor axis and the driving joint axis. The pneumatic spring is coupled parallel to the motor.

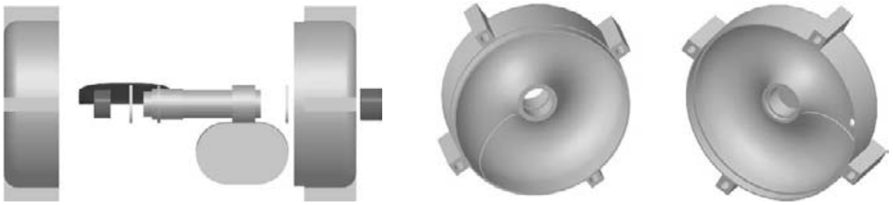


Fig. 7. Prototypical design of the pneumatic spring

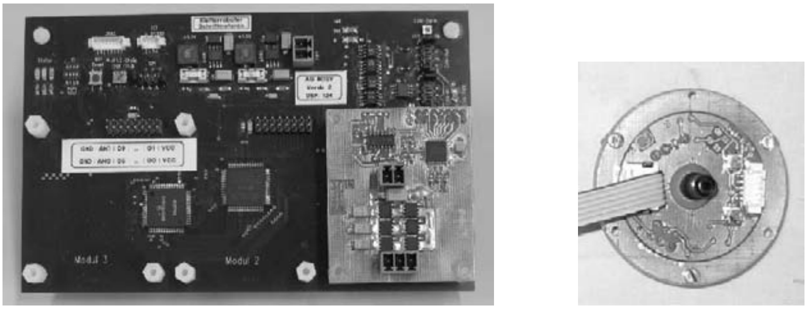


Fig. 8. (**a**) DSP board with power electronics – (**b**) pressure measurement sensor

3.2 Sensor Systems and Electronics

Fast locomotion does not only challenge the mechanics but also the electronic and computational subsystems. To successfully move at high speed several sensor systems have to be processed by controlling electronics and software.

Sensor Systems

To measure the pressure in the pneumatic spring as described in Sect. 3.1 a pressure sensor system has been developed capable of handling the occurring pressures (Fig. 8(b)). A programmable microchip for temperature dependent calibration as well as an A/D converter are directly integrated on the sensor board. High definition optical position encoders are used to measure the joint and shaft angles.

Power Electronics

To drive the DC motors power electronics have been developed and modified for high currents. The board shown as plugable module in Fig. 8(a) is capable of supplying 10 Ampere at 24 Volt and can drive brushed or brushless motors. For short periods it can even deliver higher currents. The circuit is further able to measure motor currents, this way providing a statement about the produced motor torque.

DSP/CPLD Board for Fast Closed-Loop Control

All sensor and actuator subsystems are connected to a DSP/CPLD board that was already designed for previous projects. Multiple boards can be connected via CAN-Bus to a embedded main computer. The DSP is able to manage the low-level control algorithms at high speed as the Motorola DSP 564803 is well suited for complex control and filtering tasks. Higher level control algorithms are computed on the embedded PC. The next section will provide information on these control algorithms.

4 Behaviour Control for a Running Biped

This section will introduce a concept for the control of a fast moving biped. A behaviour-based architecture is proposed for the levels above the basic torque or position control and will be briefly described in the following section. To support the inherent periodic movement of the mechanics, basic periodic motion units are introduced. Some first results from the periodic movements are presented and further steps for the implementation are discussed.

4.1 Basic Behaviour Units

The proposed architecture as introduced in [2] has been successfully applied on a four-legged walking machine as well as to wheel-driven mobile indoor and outdoor platforms. The basic computational unit is a behaviour module as shown in Fig. 9. Each module processes sensor inputs $\mathbf{e}$ to calculate a resulting action $\mathbf{u}$. This output can be influenced by two values, the activation or motivation ι coming from higher level behaviours, and the inhibition i used by lower level behaviours. In addition each behaviour features two special meta information signals, the activity $a \in [0, 1]$ stating the current amount of action produced by the behaviour; and the target rating $r \in [0, 1]$ corresponding to the current evaluation of the situation relative to the behaviour's goal. The actual functionality of the behaviour is defined by the transfer function $F(\mathbf{e}, \iota, i) = \mathbf{u}$.

All behaviours are arranged in a hierarchical network. The coordination of competing behaviours is solved by fusion nodes. These nodes calculate a resulting action based on the behaviours meta signals. Either arbitration or fusion of the values is possible. Further details on the behaviour-based architecture can be found in e.g. [2] or [3]. This behaviour architecture is to be used as reactive control layer fusioned with the basic periodic leg movement. Possible reflexes include posture stabilisation or reactions to disturbances.

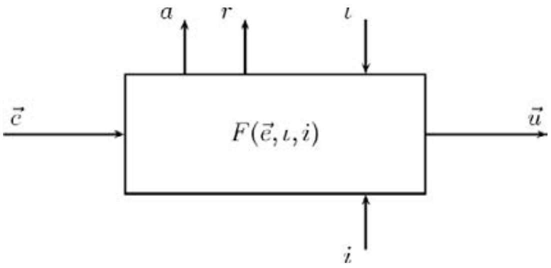


Fig. 9. The basic behaviour module

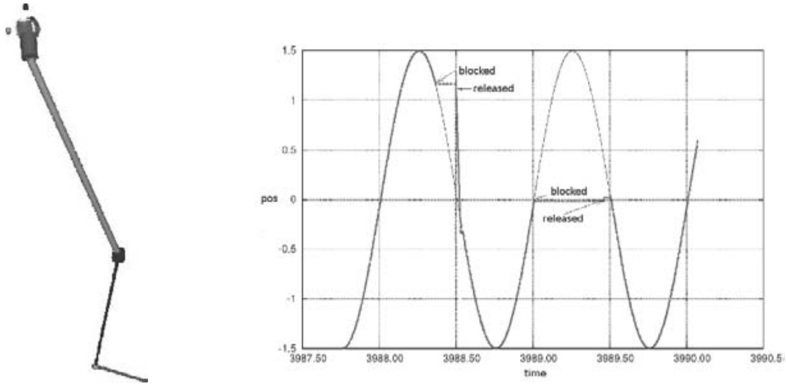


Fig. 10. (**a**) sample kinematic of human-like leg – (**b**) disturbed oscillation

4.2 Periodic Motion Units

In order to support the periodic movement of the legs while walking, running or jumping, an oscillator-like unit can be used. Such a periodic behaviour or action unit should supply the actuator with additional torque on top of the natural movement to maintain the locomotion. A tool has been developed to test different oscillator strategies on different leg kinematics. The description of the legs can be given in an extended Denavit-Hartenberg format. The direct kinematic problem is automatically solved to be able to investigate foot point trajectories.

Coupled periodic action units can be attached to the individual joints to generate walking movements in a simulation environment. These movements can be disturbed to observe how the control units can snap back to their oscillating behaviour. Figure 10 shows the generated 3D model of a sample kinematic with 6 degrees of freedom and the oscillator behaviour during a disturbed joint movement. It can be observed how the current joint angle movement (thick line) is accelerated (first disturbance) or slowed down (second disturbance) depending on the current deviation to fall back to the reference oscillation (thin line).

5 Summary and Outlook

This paper presented the mechanical design and first control concepts for a running biped. For fast running or jumping locomotion that is still energy efficient it is crucial to exploit the natural leg movement as done by passive walkers and to store energy during motion. For this task a rotatory pneumatic spring has been developed to be integrated into the actuation unit. It has an adjustable stiffness while being able to generate high torques and to withstand the high stress emerging during jumping. The actuation unit should be able to meet the demands of a knee joint and with only slight modifications be used as a hip joint.

The next research steps will be to build the proposed prototype with actuated hip and knee and to compare its performance with the results from dynamic simulation. Control algorithms following the described oscillation principles combined with a reactive control based on the behaviour-based architecture of Sect. 4.1 will be tested on the prototype leg. Future work will include the design of an elastic foot with two degrees of freedom and following that a two legged 8 DOF prototype.

References

[1] J. Albiez, T. Kerscher, F. Grimminger, U. Hochholdinger, R. Dillmann, and K. Berns. Panter – prototype for a fast-running quadruped robot with pneumatic muscles. In *6th International Conference on Climbing and Walking Robots (CLAWAR)*, Catania, Italy, September 17–19 2003.

[2] J. Albiez, T. Luksch, K. Berns, and R. Dillmann. An activation-based behavior control architecture for walking machines. *The International Journal on Robotics Research*, vol. 22:203–211, 2003.

[3] J. Albiez, T. Luksch, K. Berns, and R. Dillmann. A behaviour network concept for controlling walking machines. In *2nd International Symposium on Adaptive Motion of Animals and Machines (AMAM)*, Kyoto, Japan, March 4–8 2003.

[4] S. Collins, A. Ruina, R. Tedrake, and M. Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307:1082–1085, 2005.

[5] F. Daerden and D. Lefeber. Pneumatic artificial muscles: actuators for robotics and automation. *European Journal of Mechanical and Environmental Engineering*, 47(1):10–21, 2002.

[6] H. Geyer, A. Seyfarth, and R. Blickhan. Spring-mass running: simple approximate solution and application to gait stability. *Journal of Theoretical Biology*, 232(3):315–328, 2004.

[7] F. Iida and R. Pfeiffer. Self-stabilization and behavioral diversity of embodied adaptive locomotion. In *Embodied Artificial Intelligence*, pp. 119–129. Springer, 2003.

[8] A. Kerdok, A. Biewener, T. McMahon, P. Weyand, and H. Herr. Energetics and mechanics of human running on surfaces of different stiffness. *J. Appl. Physiol.*, 92:469–478, 2002.

[9] T. Kerscher and et al. Flumat – dynamic modelling of fluid muscles using quick-release. In *3rd International Symposium on Adaptive Motion in Animals and Machines*, 2005.

[10] J.-H. Kim and J.-H. Oh. Walking control of the humanoid platform khr-1 based on torque feedback control. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2004.

[11] T. McGeer. Passive dynamic walking. *International Journal of Robotics Research*, 9:62–82, 1990.

[12] T. McMahon and G. Cheng. The mechanics of running: How does stiffness couple with speed? *J. Biomech.*, 23(1):65–78, 1990.

[13] F. Pfeiffer, K. Löffler, and M. Gienger. The concept of jogging johnnie. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2002.

[14] J. Pratt. Exploiting inherent robustness and natural dynamics in the control of bipedal walking robots. In *PhD thesis*. MIT Press, Cambridge, 2000.

[15] A. Seyfarth, H. Geyer, M. Günther, and R. Blickhan. A movement criterion for running. *J. Biomech.*, 35(5), 2002.

[16] J. Vermeulen. *Trajectory Generation for Planar Hopping and Walking Robots: An Objective Parameter and Angular Momentum Approach*. PhD thesis, Vrije Universiteit Brussel, May 2004.

[17] B. Verrelst, R. Van Ham, B. Vanderborght, F. Daerden, and D. Lefeber. The pneumatic biped "lucy" actuated with pleated pneumatic artificial muscles. *Autonomous Robots*, 18:201–213, 2005.

[18] M. M. Williamson. Series elastic actuators. Technical report, Massachusetts Insitute of Technology, Artificial Intelligence Lab, 1995.

[19] M. Wisse. *Essentials of dynamic walking, Analysis and design of two-legged robots*. PhD thesis, TU Delft, 2004.

[20] H. Witte, H. Hoffmann, R. Hackert, C. Schilling, M. Fischer, and H. Preuschoft. Biomimetic robotics should be based on functional morphology. *J. Anat.*, 204:331–342, 2004.

Dynamical Synthesis of a Walking Cyclic Gait for a Biped with Point Feet

S. Miossec[1] and Y. Aoustin[2]

[1] Joint Japenese-French Robotics Laboratory, Intelligent Systems Research Institute, National Institute of Advanced Science and Technology, AIST Tsukuba Central 2, Umezono 1-1-1, Tsukuba, 305-8568 Japan
Sylvain.Miossec@aist.go.jp

[2] Institut de Recherche en Communications et Cybernétique de Nantes, U.M.R. 6597 Ecole Centrale de Nantes, CNRS, Université de Nantes, 1 rue de la Noë, BP 92101, 44321 Nantes Cedex 3, FRANCE
Yannick.Aoustin@irccyn.ec-nantes.fr

Summary. This paper deals with a methodology to design optimal reference trajectories for walking gaits. This methodology consists of two steps: (i) design a parameterized family of motions, and (ii) determine the optimal parameters giving the motion that minimizes a criterion and satisfies some constraints within this family. This approach is applied to a five link biped, the prototype *Rabbit*. It has point feet and four actuators which are located in each knee and haunch. *Rabbit* is under-actuated in single support since it has no actuated feet and is overactuated in double support. To take into account this under-actuation, a characteristic of the family of motions considered is that the four actuated joints are prescribed as polynomials in function of the absolute orientation of the stance ankle. There is no impact. The chosen criterion is the integral of the square of torques. Different technological and physical constraints are taken into account to obtain a walking motion. Optimal process is solved considering an order of treatment of constraints, according to their importance on the feasibility of the walking gait. Numerical simulations of walking gaits are presented to illustrate this methodology.

1 Introduction

For more than thirty years walking robots and particularly the bipeds have been the objects of research. For example Vukobratovic and his co-author [1] have proposed in 1968 their famous Zero-Moment Point (*ZMP*), for the analysis of a biped gait with feet. In 1977, optimal trajectories [2] were designed for a bipedal locomotion using a parametric optimization. Formal'sky completely characterized the locomotion of anthropomorphic mechanisms in [3] in 1982. Sutherland and Raibert proposed their principle about virtual legs for walking robots in the paper [4] in 1983. Currently Humanoïds such as *Honda* biped in

[5] and *HRP2* biped in [6] (Humanoïd Robotics Project 2), which are probably on the technological point-of-view the most advanced biped robots, lead to many popular demonstrations of locomotion and interaction with their environment. In parallel, some research is done on legged robots with less degrees of freedom. Here it is worked with the control, the model and the reference trajectories to design walking bipedal gaits more fluid. See for example [7] where a biped with telescopic legs is studied, [8] where the famous dog *Aibo* from *Sony* is used to design biped gaits, [9] where an intuitive approach is developed for a biped locomotion or [10] where an accurate analysis of the gravity effects is made to give necessary and sufficient conditions to ensure a cyclic walking gait for a biped without feet.

In this paper, the efforts are focused on the design by a parametric optimization of a walking gait. This approach necessitates two steps: (i) design a parameterized family of motions, and (ii) determine the optimal parameters giving the motion that minimizes a criterion and satisfies some constraints within this family. The motion obtained is later used as a reference motion. This approach is applied to a planar five-link biped without feet and with four actuators only. The family of motions considered is composed of a single-support phase and a double-support phase, with no impact. The minimization criterion is the integral over the motion of the square of torques. Therefore it is a criterion of torque minimization. The originality of the present work is double:

- To overcome the underactuated characteristic of the biped, the four variables defined as polynomials in single support are function of another generalized coordinate, the absolute orientation at the stance leg ankle. This allows to define the configurations of the biped during the single support phase, while the dynamics of the not controlled degree of freedom are still unknown. In double support, two actuated joints are also prescribed as functions of the absolute orientation at the stance leg ankle, which is a polynomial function in time.
- There is a classification and a treatment of constraints according to their importance on the feasibility of the walking gait.

This paper does not address the stability of the motion obtained. The reader may refer to [11] which gives conditions of stability of the non controlled degree of freedom during the single support phase, and additionally a measure of this stability. It has been proved that the presence of the double support phase practically guarantees the stability.

This article is organized as follows: the dynamical model of the biped under interest is presented in Sect. 2 for the single and the double-support phase. Section 3 is devoted to the definition of the family of reference trajectories, their constraints and their parameters. The calculation of the criterion in torque during the single support and the double support, and the optimization process to determine the optimal parameters are presented in Sect. 4. Some simulation results are shown Sect. 5. Section 6 contains our conclusion and perspectives.

2 Dynamic Model

2.1 Presentation of the Biped and Notations

A planar five-link biped is considered and is composed by a torso and two identical legs with knee and point feet (see Fig. 1 for a diagram of the studied biped). There are four identical motors, which drive the haunches and the knees. We note $\Gamma = [\Gamma_1, \Gamma_2, \Gamma_3, \Gamma_4]^T$ the torque vector, $q = [\alpha, \delta^T]^T$=$[\alpha, \delta_1, \delta_2, \delta_3, \delta_4]^T$ the vector composed of the orientation of the stance leg and the actuated joint variables, and $X = [q^T, x_t, y_t]^T$ the vector of generalized coordinates. The components (x_t, y_t) define the position of the center of gravity of the trunk.

2.2 A Reduced Model

The optimization process to determined reference trajectories, which will be presented in the next sections leads to many CPU operations. Therefore the strategy was to use a reduced model that needs less computations. To obtain this reduced model, we consider that the contact between the leg tip 1 and

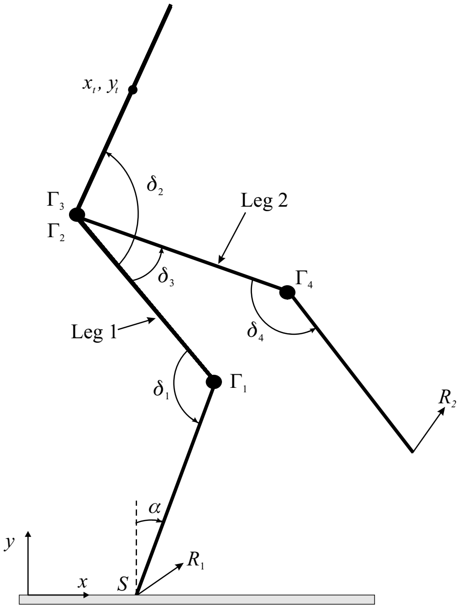


Fig. 1. Biped in the sagittal plane

the ground is acting as a pivot: there is no take off and no slipping. Then the biped configuration is described with vector q only. This model is reduced by comparison to a more general model that would be written with vector X. We obtain the reduced model by using Lagrange's equations:

$$A(\delta)\ddot{q} + H(q,\dot{q}) + Q(q) = D_{\Gamma}\Gamma + D_2(q)R_2 \tag{1}$$

where $A(\delta)(5\times5)$ is the symmetric positive inertia matrix of the biped. As the kinetic energy of the biped is invariant under a rotation of the world frame [12], and viewed that α defines the orientation of the biped, the 5×5-symmetric positive inertia matrix is independent of this variable, i.e. $A = A(\delta)$. Vector $H(q,\dot{q})(5 \times 1)$ represents the centrifugal, Coriolis effects, and $Q(q)(5 \times 1)$ is the gravity effects vector. $D_{\Gamma}(5 \times 4)$ is a constant matrix composed of ones and zeros. $D_2(q)$ is the 5×2-Jacobian matrix converting the ground reaction in the leg tip 2 into the corresponding joint torques.

Taking into account Coulomb dry and viscous frictions, Γ has the following form

$$\Gamma = \Gamma_u - \Gamma_s sign(D_{e\Gamma}^T\dot{q}) - F_v D_{e\Gamma}^T\dot{q} \tag{2}$$

where $\Gamma_s(4\times4)$ and $F_v(4\times4)$ are diagonal matrices representing respectively the dry friction and the viscous friction. Γ_u is the motors torque vector when considering the joint friction.

In the case of double support, the point foot 2 is in contact with the ground. Then the position variables q, the velocity variables $\dot{q}$, and the acceleration variables $\ddot{q}$ are constrained. In order to write these relations, we define the position, velocity and acceleration of the point foot 2 in an absolute frame. The position of the point foot 2 is noted $d_2(X)$. By differentiation of $d_2(X)$ we obtain the relation between the velocity $V_2 = (V_{2x}\ V_{2y})^T$ of the point foot 2 and $\dot{q}$,

$$V_2 = D_{e2}(q)^T\dot{q}\ . \tag{3}$$

By another differentiation we obtain the relation between the acceleration $\dot{V}_2 = (\dot{V}_{2x}\ \dot{V}_{2y})^T$ of the point foot 2 and $\ddot{q}$,

$$\dot{V}_2 = D_{e2}(q)^T\ddot{q} + \dot{D}_{e2}(q)^T\dot{q} = D_{e2}(q)^T\ddot{q} + C_{e2}(q,\dot{q})\ . \tag{4}$$

Then the contact constraints for the point foot 2 with the ground are given by the three vector-matrix equations:

$$\begin{cases} d_2(X) = const\ , \\ \\ V_2 = 0\ , \\ \\ \dot{V}_2 = 0\ . \end{cases} \tag{5}$$

These vector-matrix equations (5) mean that the position of the point foot 2 remains constant, and then the velocity and acceleration of the point foot 2 are zero.

During the double-support phase, both legs are in contact with the ground. Then the dynamic model is formed of both vector-matrix equations (1) and (5). During the single-support phase on leg 1, the dynamic model is simply written as (1) with the ground reaction for foot 2 in the air is $R_2 = (0\,0)^T$.

Model (1) allows us to compute the torques and the dynamic model of α easier (10). However, it is not possible to take into account a single-support on the leg 2 with (1). Furthermore we cannot calculate the ground reaction with model (1) only. We add the two following equations, obtained from the Newton's second law at the center of mass G of the biped

$$\begin{cases} M\ddot{x}_G = R_{1x} + R_{2x} \\ M\ddot{y}_G = R_{1y} + R_{2y} - Mg \end{cases} \tag{6}$$

where M is the mass of the biped and (x_G, y_G) are the coordinates of G.

3 Definition of the Walk and Its Constraints

Our objective is to design a cyclic bipedal gait. There are two aspects for this problem. The definition of a parameterized family of reference trajectories and the method to determine a particular solution in this restricted space. This section is devoted to the definition of the parameterized family of reference trajectories. The optimal process to choose the best solution of parameters from the point of view of a given criterion will be described in the next section. The parameterized family of reference motions is such that one degree of freedom, which changes monotonically during a step composed of a single-support phases and a double-support phases, will be used as a variable to define the other degrees of freedom. These special solutions lead to a particular simple dynamical model of the biped in single support which can be calculated from (1). An impactless bipedal gait is considered because in [13] numerical results proved that the insertion of an impact with this walking gait for the studied biped is a very difficult challenge. The condition found to obtain no impact was simply that the velocity of free foot must reach the ground with zero velocity. After the choice of parameters, the constraints will be determined. In the following, indices "ss" and "ds" respectively indicate the single-support phase and the double-support phase.

3.1 Restrictions of Motion Considered in Single Support

During the single support, the biped has five degrees of freedom. With the four actuators for the biped, only four output variables can be prescribed. Then the biped is underactuated in single support. In previous experiments, see for example, [7, 14, 15], researchers observed that for most of walking gaits of biped robots the ankle angle α of the stance leg changes absolutely monotonically during the single-support phase. Therefore, it is possible to use the angle variable α instead of time t as an independent variable during the single-support phase of the bipedal gait. As a consequence α like time will have to be monotonic. But this choice will not eliminate potentially optimal motions in the space in which we seek for solutions, since so far all motions observed were satisfying this property. Thus the four joint variables δ_j are prescribed as polynomial functions of the ankle angle α, $\delta_{j,ss}(\alpha)$ $(j = 1, \ldots, 4)$. The behavior of α is governed by the dynamic model (1). To deal with the underactuation the advantage of this approach is that the complete set of configurations is defined during the motion of the biped and it is not necessary to anticipate a duration for the single-support phase, which is the result of the integration of (1). The order of these polynomial functions (7) is chosen at four to specify initial, final and intermediate configurations, plus initial and final joint velocity variables,

$$\delta_{j,ss}(\alpha) = a_{j0} + a_{j1}\alpha + a_{j2}\alpha^2 + a_{j3}\alpha^3 + a_{j4}\alpha^4 \ . \tag{7}$$

Let us note that it would be possible to prescribe other variables as Cartesian variables. But to avoid the problems of singularity of the inverse geometric model in the single-support phase, we prefer to work with angular variables only. However some authors, for example [2, 16], use Cartesian coordinates of the hip for the definition of the bipedal gait. The joint variables are then prescribed. However since the biped is underactuated the evolution of the angle α must be such that the biped motion satisfies the dynamic model. Considering the relations (7) we introduce for the variables of the reference motion $q = q(\alpha)$ the following temporal derivatives

$$\begin{aligned} \dot{q}(\alpha, \dot{\alpha}) &= q^* \dot{\alpha} \\ \ddot{q}(\alpha, \dot{\alpha}, \ddot{\alpha}) &= q^* \ddot{\alpha} + q^{**} \dot{\alpha}^2 \end{aligned} \tag{8}$$

where the notation $()^*$ means partial derivative with respect to α, and the $\dot{()}$ represent derivation with respect to time. Then we have $q^* = [1\, \delta_1^*\, \delta_2^*\, \delta_3^*\, \delta_4^*]^T$ and $q^{**} = [0\, \delta_1^{**}\, \delta_2^{**}\, \delta_3^{**}\, \delta_4^{**}]^T$. By calculating the angular momentum of the biped at the fixed point S (see Fig. 1), we obtain the general form

$$\sigma = \sum_{i=1}^{4} f_i(\delta_1, \delta_2, \delta_3, \delta_4)\dot{\delta}_i + f_5(\delta_1, \delta_2, \delta_3, \delta_4)\dot{\alpha} \ . \tag{9}$$

We can obtain two first order differential equations on σ and α (see [15])

$$\begin{cases} \dot{\sigma} = -Mg\,(x_G(\alpha) - x_S) \\ \dot{\alpha} = \dfrac{\sigma}{f(\alpha)}. \end{cases} \tag{10}$$

M is the biped mass, g the acceleration of gravity, $x_G(\alpha)$ and x_S are respectively the horizontal component of the positions of the biped's mass center and of the foot of the stance leg. The first equation of (10) comes from the dynamic momentum equation at S when eliminating q from (7). The second equation of (10) follows from (9) when eliminating q and $\dot q$ using (7) and (8). This differential system (10) is equivalent to the first line of (1). By identification, it is possible to determine $f(\alpha)$ and $x_G(\alpha)$ from (1). The simple model (10) completely defines the dynamic behavior of the biped in single support for the reference motion. From (10) we can deduce that (see [17])

$$\dot\sigma = \frac{d\sigma}{d\alpha}\dot\alpha = \frac{d\sigma}{d\alpha}\frac{\sigma}{f(\alpha)} = \frac{1}{2}\frac{d\sigma^2}{d\alpha}\frac{1}{f(\alpha)} = -Mg\left(x_G(\alpha) - x_S\right).$$

Finally this calculation leads to the relation due to [17]

$$\frac{d\sigma^2}{d\alpha} = -2Mg\left(x_G(\alpha) - x_S\right) f(\alpha)\,. \tag{11}$$

If α is strictly monotone, the integration of (11) gives

$$\sigma^2 - \sigma_{iss}^2 = -2Mg \int_{\alpha_{iss}}^{\alpha} \left(x_G(s) - x_S\right) f(s) ds \tag{12}$$

where σ_{iss} is the angular momentum at the beginning of single support characterized by the initial value α_{iss}. Then the dynamic of the biped is completely defined from (10) as function of $\mathit{\Phi}(\alpha) = \sigma^2 - \sigma_{iss}^2 = \dot\alpha^2 f^2(\alpha) - \dot\alpha_{iSS}^2 f^2(\alpha_{iSS})$ such as

$$\dot\alpha = -\frac{\sqrt{\mathit{\Phi}(\alpha) + f(\alpha_{iSS}) 2\dot\alpha_{iSS}^2}}{f(\alpha)}\,. \tag{13}$$

$\ddot\alpha$ is obtained from the second equation of (10)

$$\ddot\alpha = \frac{\dot\sigma f(\alpha) - \sigma \dot f(\alpha)}{f^2(\alpha)} = -\frac{Mg\left(x_G(\alpha) - x_S\right) + \frac{df(\alpha)}{d\alpha}\dot\alpha^2}{f(\alpha)}\,. \tag{14}$$

From the solution of the differential equation in α (11) and using relations (13) and (14) the numerical simulation to find the optimal motion and the calculation of constraints will be easier.

The authors of [17] showed that system (10) behaves like an inverted pendulum. Therefore the only possible non-monotone behavior would be that the biped fall back if the initial velocity of single support is not sufficient. The condition to ensure the monotony of α has been added as a constraint in the optimization process, see (18).

3.2 Restrictions of Motion Considered in Double Support

In double support, the biped has three degrees of freedom. With its four actuators, the biped is over actuated. Hence the motion of the biped is completely defined with three prescribed degrees of freedom. For a question of convenience for the use of the inverse geometric model, the ankle angle α and both joint variables, δ_j $(j = 1, 2)$ are prescribed. A polynomial function in time of third-order (15) is chosen to define α. In a concern to be homogeneous with the single support phase we define both joint angular variables δ_j, as polynomial functions of third-order in α. Then initial and final configurations, and initial and final velocities can be defined for these three prescribed variables. The duration of the double-support phase is a parameter. Hence we get

$$\begin{cases} \alpha(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \ , \\ \\ \delta_j(\alpha) = a_{j0} + a_{j1}\alpha + a_{j2}\alpha^2 + a_{j3}\alpha^3 . \end{cases} \tag{15}$$

It should be noted that there is no differential equations needed for the definition of the motion, since the biped is over-actuated in double support.

3.3 Optimization Parameters

A boundary value problem has to be solved to design this cyclic bipedal gait with successive single and double-support phases. This boundary value problem depends on parameters to prescribe the initial and final conditions for each phase. Taking into account the conditions of continuity between the phases and the conditions of cyclic motion we will enumerate now in detail the minimal number of parameters which are necessary to solve this boundary value problem on a half step k (a half step is considered as a single support and a double support).

1. Seven parameters are needed to define the initial and final configurations in double support. The parameters α_{ids}, $\delta_{1,ids}$, θ_{ids}, α_{fds}, $\delta_{1,fds}$, θ_{fds} and d, the distance between both tips of stance legs in double support are chosen. The use of the absolute orientation of the trunk, θ (see Fig. 2) instead of $\delta_{2,fds}$ is easier and does not change the problem.
2. Time T_{ds} of the double support is given as a parameter.
3. The initial velocity of the biped in single support is prescribed by only three parameters $\dot{\alpha}_{iss}$, $\delta^*_{1,iss}$, $\delta^*_{2,iss}$. The velocities $\delta^*_{3,iss}$ and $\delta^*_{4,iss}$ are deduced taking into account the null velocity of the leg tip which takes off.
4. The final velocity of the biped in single support is prescribed by only three parameters $\dot{\alpha}_{fss}$, $\delta^*_{1,fss}$, $\delta^*_{2,fss}$. The velocities $\delta^*_{3,fss}$ and $\delta^*_{4,fss}$ are deduced taking into account the absence of impact of the swing leg tip on the ground, which is equivalent to a null velocity of this tip.
5. With the chosen order for the polynomial functions (7) (fourth order) it is necessary to specify five conditions for each function $\delta_{j,ss}$, $j = 1, \ldots, 4$. Then the fifth coefficient is calculated by defining an intermediate configuration. Let intermediate configuration in single support be determined with the five following parameters α_{int}, $\delta_{1,int}$, θ_{int} and the coordinates

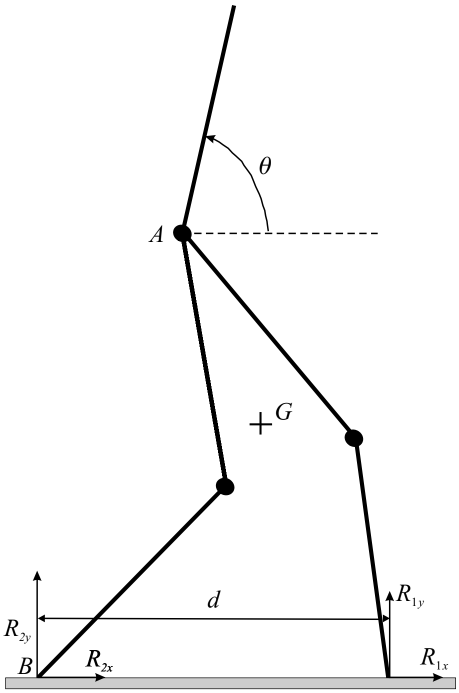


Fig. 2. Biped in the sagittal plane (the point G is the center of mass of the biped)

($x_{p,int}$ and $y_{p,int}$) of the swing leg tip. The angle α_{int} is fixed equal to $\dfrac{\alpha_{iss}+\alpha_{fss}}{2}$.

Then finally the vector of parameters has eighteen coordinates

$$p = [T_{ds}, \alpha_{ids}, \delta_{1,ids}, \theta_{ids}, \alpha_{fds}, \delta_{1,fds}, \theta_{fds}, d, \dot{\alpha}_{iss}, \delta^*_{1,iss}, \ldots$$

$$\delta^*_{2,iss}, \alpha_{fss}, \delta^*_{1,fss}, \delta^*_{2,fss}, \delta_{1,int}, \theta_{int}, x_{p,int}, y_{p,int}] \ .$$

3.4 Constraints

Constraints have to be considered to design nominal gait. We will present them according to their importance on the feasibility of the walking gait.

- First, no motion is possible if the distance $d(A,B)$ between the tip of leg 2 and the hip joint, for initial and final configurations of the double support and the intermediate configuration of the single support, is such that

$$d(A,B) > 2 \times l \tag{16}$$

where l is the common length of the femur and the tibia. In other words, there is no solution with the geometrical model to compute δ_3 and δ_4.

- Constraint (16) is also taken into account during the motion of the biped in double support. The maximum value of $d(A, B)$ in function of α is considered.
- The mechanical stops of joints for initial, intermediate and final configurations of each phase and during the motion are

$$\begin{cases} -260^\circ < (\delta_2)_{min}\,, & (\delta_2)_{max} < -110^\circ \\ -260^\circ < (\delta_2 - \delta_3)_{min}\,, & (\delta_2 - \delta_3)_{max} < -110^\circ \\ -230^\circ < (\delta_1)_{min}\,, & (\delta_1)_{max} < -127^\circ \\ -230^\circ < (\delta_4)_{min}\,, & (\delta_4)_{max} < -127^\circ \end{cases}$$

The notation $()_{max}$ and $()_{min}$ stands respectively for the maximum and minimum value over one step.

- In double support the monotony condition for variable α is imposed

$$\max_{t \in [0, T_{ds}]} \dot{\alpha}(t) < 0\,. \tag{17}$$

- In single support, the monotony condition for variable α is imposed by the inequality

$$\Phi_{min} + f(\alpha_{iss}) 2 \dot{\alpha}_{iss}^2 > 0 \tag{18}$$

where $\Phi_{min} = \min_{\alpha \in [\alpha_{iss}, \alpha_{fss}]} \Phi(\alpha)$.

- In single support it is fundamental to avoid the singularity $f(\alpha) = 0$ to simulate one step. Then we define the following constraint

$$\min_{\alpha \in [\alpha_{iss}, \alpha_{fss}]} f(\alpha) > 0\,. \tag{19}$$

Now the following constraints can be violated during the optimization process to simulate a half step. However they are important for experimental objectives. The optimization process will ensure their verification.

- Each actuator has physical limits such that

$$\begin{cases} \Big(|\Gamma_1^*(\alpha)| - \Gamma_{max}(|\dot{\delta}_1|)\Big)_{max} < 0 \\ \Big(|\Gamma_2^*(\alpha)| - \Gamma_{max}(|\dot{\delta}_2|)\Big)_{max} < 0 \\ \Big(|\Gamma_3^*(\alpha)| - \Gamma_{max}(|\dot{\delta}_2 - \dot{\delta}_3|)\Big)_{max} < 0 \\ \Big(|\Gamma_4^*(\alpha)| - \Gamma_{max}(|\dot{\delta}_4|)\Big)_{max} < 0 \end{cases} \tag{20}$$

Function $\Gamma_{max}(*)$ can be deduced from a template, torque actuator/velocity, given by the actuator manufacturer.

- We must take into account constraints on the ground reaction $R_j = (R_{jx}\ R_{jy})^T$ in the tip of the stance leg j, $j = 1$ in single support and $j = 1, 2$ in double support. The ground reaction must be inside a friction cone defined by the friction coefficient μ. This is equivalent to write both inequalities

$$\begin{aligned} R_{jx} - \mu R_{jy} &< 0 \\ -R_{jx} - \mu R_{jy} &< 0. \end{aligned}$$

By summing these two inequalities, the condition of no take off is deduced

$$\Rightarrow R_{jy} > 0\ . \tag{21}$$

- There is also a constraint on the swing leg tip to avoid an impact with the ground during its transfer. This constraint is defined by a parabola function

$$\min_{\alpha \in [\alpha_{iss}, \alpha_{fss}]} \left[y(\alpha) - \left(\frac{x^2(\alpha)}{d^2} - 1 \right) y_{max} \right]$$

where (x, y) are the coordinates of the swing leg tip and y_{max} is the maximum height of the parabola.

- Optimal motions are defined for different velocities with the constraint

$$d = v(T_{ss} + T_{ds}) \tag{22}$$

where d is the distance between the tips of stance legs (see Fig. 2), v is the desired average velocity of the biped, and T_{ss} is the time of the single-support phase. The calculation of time T_{ss} of the single-support phase is given by $T_{ss} = \int_{\alpha_{iss}}^{\alpha_{fss}} \frac{1}{\dot{\alpha}} d\alpha$

4 Optimal Walk

Many values of parameters presented in Sect. 3 can give a periodic bipedal gait satisfying constraints (16)–(22).

Then a parametric optimization process, minimizing a criterion under nonlinear constraints, is possible to find a particular nominal motion. Let us define this optimization process

$$\begin{aligned} &\min_p C(p) \\ &g_i(p) \le 0 \quad i = 1, 2, \ldots, n \end{aligned} \tag{23}$$

where p is the vector of parameters, $C(p)$ is the criterion to minimize with n constraints $g_i(p) \le 0$ to satisfy. We give now some details about the way to calculate the criterion during the single-support phase and the double-support phase, and about the optimization process.

4.1 Criterion

To find the nominal motion criterion C_Γ, which is a torque minimizing criterion, is considered

$$C_\Gamma = \frac{1}{d}\int\limits_{0}^{T_{ss}+T_{ds}} \Gamma^T\Gamma dt = \frac{1}{d}\left(\int\limits_{\alpha_{iss}}^{\alpha_{fss}} \frac{\Gamma^T\Gamma}{\dot{\alpha}} d\mu + \int\limits_{0}^{T_{ds}} \Gamma^T\Gamma dt\right) \tag{24}$$

where T_{ss} and T_{ds} are the times of single support and double support. For electrical motors such as DC motors the torque is usually proportional to the induced current. Then the criterion C_Γ represents the losses by Joule effects to cover distance d, see [18, 19]. To consider an energy minimizing criterion, it would only be necessary to add the losses by friction in the joints.

4.2 Single-Support Phase

From calculation of the integral term (12) using the polynomial functions (7), we obtain $\Phi(\alpha) = \sigma^2 - \sigma_{iss}^2$. Velocity $\dot{\alpha}$ and acceleration $\ddot{\alpha}$ can be obtained with relations (13) and (14). We then have determined the dynamics of the under actuated biped in single support for a reference trajectory. The torques are determined from the four last equations of (1)

$$A_{25}(\delta)\ddot{q} + H_{25}(q,\dot{q}) + Q_{25}(q) = D_{\Gamma 25}\Gamma \tag{25}$$

where $A_{25}(4\times 5)$, $H_{25}(4\times 5)$ and $D_{\Gamma 25}(4\times 4)$ are the submatrices of A, H and D_Γ, $Q_{25}(4\times 1)$ is the subvector of Q. The invertible matrix $D_{\Gamma 25}$ allows to determine the torque vector Γ. The ground reaction $R_i = (R_{ix}, R_{iy})$ at the tip of the stance leg i are calculated using the equations (6).

4.3 Double-Support Phase

From relations (15) $\alpha(t)$, $\dot{\alpha}(t)$ and $\ddot{\alpha}(t)$ are calculated as polynomial functions of time first at each time step, then $\delta_j(\alpha)$, $\dot{\delta}_j(\alpha)$ and $\ddot{\delta}_j(\alpha)$ $(j = 1, 2)$ are determined. There is an infinit set of solutions for the torques to realize the double support, because the biped is overactuated. Only three generalized coordinates, for example $\alpha(t)$, δ_1 and δ_2, are necessary to describe the motion completely. Then, we can parameterize the solution of torques as function of one variable. To find this variable we consider equation (6) and the equation of the angular momentum theorem applied at the leg tip 1. The equation of the angular momentum theorem in double support is equivalent to equation (10) but with the effect of ground reaction force of foot 2. It is also equivalent to the first line of model (1). This additional equation reads

$$A_1(\delta)\ddot{q} + H_1(q,\dot{q}) + Q_1(q) = -dR_{2y} \tag{26}$$

where $A_1(1 \times 5)$ and $H_1(1 \times 5)$ are the first line of A and H, $Q_1(1 \times 1)$ is the first element of Q. Term d is the distance between the two leg tips on the ground. Component R_{2x} does not appear in equation (26) because the ground is assumed to be horizontal and plane. From the second line of (6) and (26), for a given acceleration of the biped there is only one solution for R_{1y} and R_{2y}, independent of the torques. The torques only influence R_{1x} and R_{2x}. For this reason, a solution for the torques can be found as function of R_{1x} or R_{2x} as parameter. Let us choose R_{2x} and define the minimization problem with the associated constraint on component R_{2x}

$$\min_{R_{2x}} \Gamma^{*T}\Gamma^{*}$$

$$\begin{cases} -\mu R_{1y} - R_{1x} \leq 0 \\ -\mu R_{1y} + R_{1x} \leq 0 \\ -\mu R_{2y} - R_{2x} \leq 0 \\ -\mu R_{2y} + R_{2x} \leq 0\ . \end{cases} \tag{27}$$

The choice of the particular solution of this optimization problem is because it is also the solution that minimizes the criteria (24). With the four last lines of the vector-matrix equations (1) and (2) a relation between torques Γ^* and R_{2x} can be written

$$\Gamma^* = J - KR_{2x} \tag{28}$$

with $K = D_{\Gamma 25}^{-1} D_{2x\ 25}$ and

$$J = D_{\Gamma 25}^{-1}\left(A_{25}\ddot{q} + H_{25}(q,\dot{q}) + Q_{25}(q) - D_{2y\ 25}R_{2y}\right) + \Gamma_s \text{sign}(D_\Gamma^T \dot{q}) + F_v D_\Gamma^T \dot{q}.$$

The solution $R_{2x\ opt\Gamma}$ which minimizes the square of the torques without constraints is given when $\Gamma^{*T}\frac{\partial \Gamma^*}{\partial R_{2x}} = 0$. Considering equation (28) $R_{2x\ opt\Gamma}$ is given by

$$R_{2x\ opt\Gamma} = \frac{K^T J}{K^T K}\ . \tag{29}$$

Defining a minimum value R_{2xinf} and a maximum value R_{2xsup}, the constraints on R_{2x} can be written under the simple form,

$$R_{2xinf} \leq R_{2x} \leq R_{2xsup} \tag{30}$$

Then a solution for the minimization problem (27) is given by three cases

- if $R_{2x\ inf} \leq R_{2x\ opt\Gamma} \leq R_{2x\ sup}$ then $R_{2x} = R_{2x\ opt\Gamma}$,
- if $R_{2x\ opt\Gamma} \leq R_{2x\ inf}$ then $R_{2x} = R_{2x\ inf}$,
- if $R_{2x\ sup} \leq R_{2x\ opt\Gamma}$ then $R_{2x} = R_{2x\ sup}$.

In the case where there is no solution, i.e $R_{2xinf} \geq R_{2xsup}$, we choose R_{2x} to minimize the violation of constraints such as

$$R_{2x} = \frac{R_{2x\ inf} + R_{2x\ sup}}{2} \ .$$

In this last situation, the constraints are not satisfied. However, the optimization process will tend to satisfy the constraints of the motion, and the final solution will always satisfy $R_{2xinf} \leq R_{2xsup}$. This violation will only occur during the optimization process.

4.4 Optimization Algorithm

The algorithm *NPSOL*, see [20] is used to solve this optimization problem with its nonlinear constraints. The sequence of treatment of constraints according to their importance is described Fig. 3. From level 0 to level 4, the constraints must be satisfied to simulate one step. Other constraints as the maximum velocity of the biped, the torques limits are considered in level 5.

Sometimes, while solving the problem (23), the optimization process can ask a value of the criterion or the constraints in a point p_0 where they are not defined. Therefore an intermediate optimization process is started to find another point p_M, the closest from p_0. For example if constraints $g_i(p_0) \leq 0$, $i = 1, 2, \ldots, m_0$ are not satisfied, p_M is determined as the solution of the problem

$$\begin{gathered} \min_p \|p_0 - p\| \\ g_i(p) \leq 0 \quad i = 1, 2, \ldots, m_0 \ . \end{gathered} \tag{31}$$

Then the constraints not defined at the point p_0 will be computed at the point p_M. And using gradient information at p_M, an interpolated value will be determined at p_0. This interpolation ensures that constraints and criteria are continuous and differentiable functions, even at the boundary of their space of definition. This is a necessary condition for the optimization program to solve this modified problem.

During the optimization process the constraints can be violated. But it tends to satisfy the constraints at the end of the optimization. Since we add in the problem the constraints specifying the sub-space where all constraints and criterion are defined, at the end of the optimization the walking motion will be defined and satisfy all the constraints. The only situation where the algorithm will not find a solution that satisfies constraints is if there is no such solution (if we ask for a walk too fast and the actuators are not sufficient to do it, for example) or if the problem is not convex. Indeed the algorithm used is a local optimization algorithm. For a non convex problem, it will probably find only a local non feasible solution, whereas other feasible solution exists. However, we have tried many random initial conditions for the optimization process and always found the same optimal solution that satisfied constraints. We can therefore assume that our problem is convex.

To solve the intermediate optimization problem (31) and the general optimization problem (23), the gradient in function of the vector of parameters p of the criterion and constraints is necessary. To obtain an efficient algorithm, these gradients were calculated analytically.

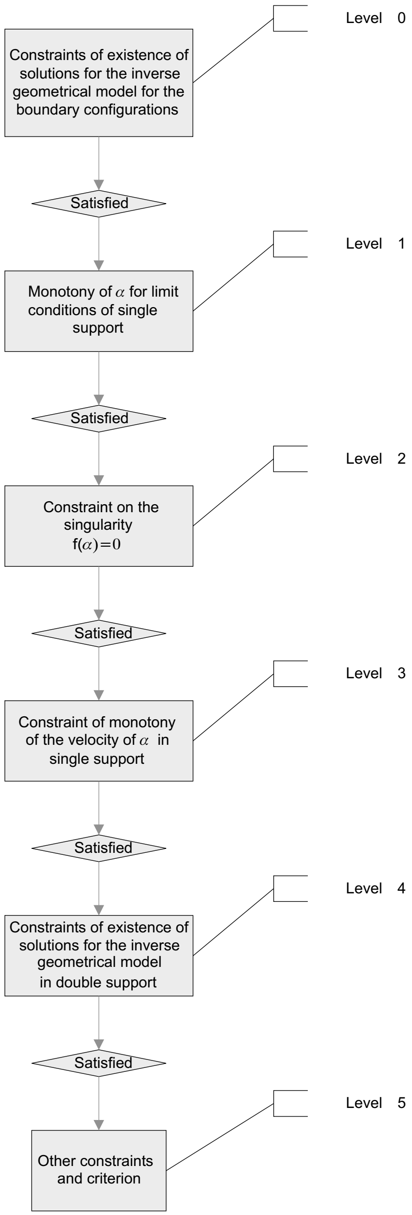

Level 0
Constraints of existence of solutions for the inverse geometrical model for the boundary configurations
Satisfied
Level 1
Monotony of α for limit conditions of single support
Satisfied
Level 2
Constraint on the singularity f(α)=0
Satisfied
Level 3
Constraint of monotony of the velocity of α in single support
Satisfied
Level 4
Constraints of existence of solutions for the inverse geometrical model in double support
Satisfied
Level 5
Other constraints and criterion

Fig. 3. Sequence of constraints to satisfy before the step can be defined

5 Simulation Results

Figures 4–7 are devoted to a chosen motion velocity for a biped which equals 0.3 m/s. Figure 4 shows that the needed torques for this trajectory are inside the template, motor torque/velocity, given by the manufacturer. The normal components of the ground reactions as functions of time, during one step are presented in Fig. 5. The constraint of unilateral contact on the leg tip 2 is active because the fixed limit 20 N is reached in the tip of leg 2 during the double-support phase. The double-support phase begins after time 0.93 s.

Figure 6 shows as functions of time the evolutions of joint variables δ_1, δ_2, δ_3 and δ_4 in single-support phase and double-support phase. Let us remark that the discontinuities in the graphes mark the limit between the single-support phase and the double-support phase. These discontinuities are not due to an impact (only an impactless motion is considered). These discontinuities appear in the graphes of Figs. 5–7 because the role of both legs are exchanged at the beginning of the double-support phase. Figure 7 shows the behavior of the variable α, which is monotone as expected. The discontinuity at the end of the single-support phase (time 0.93 s) is due to the exchange of the role of both legs.

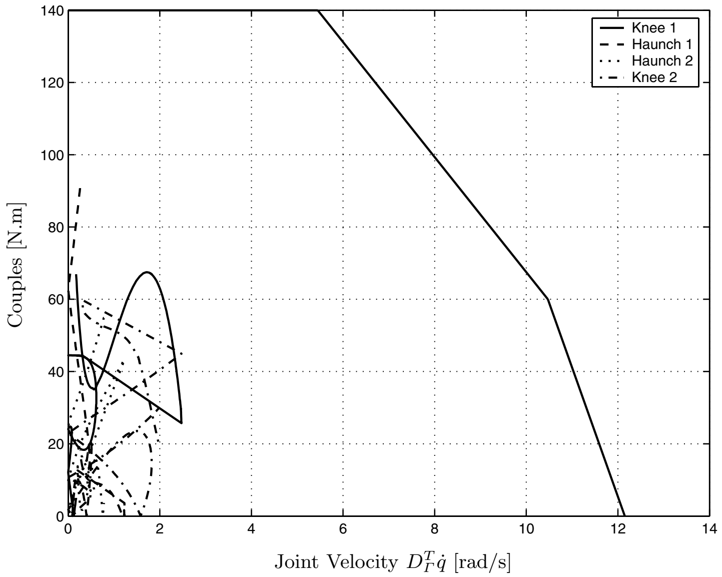


Fig. 4. Velocity versus torque for knee i and haunch i, ($i = 1, 2$) are inside the template, motor torque/velocity, defined by the limit values 140 N.m and 12 rad/s

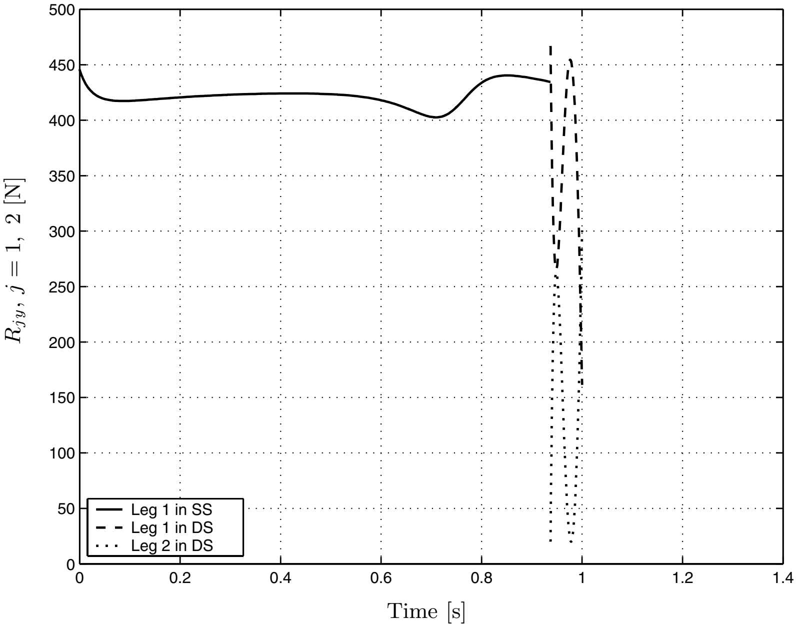


Fig. 5. Normal components in the stance leg tips

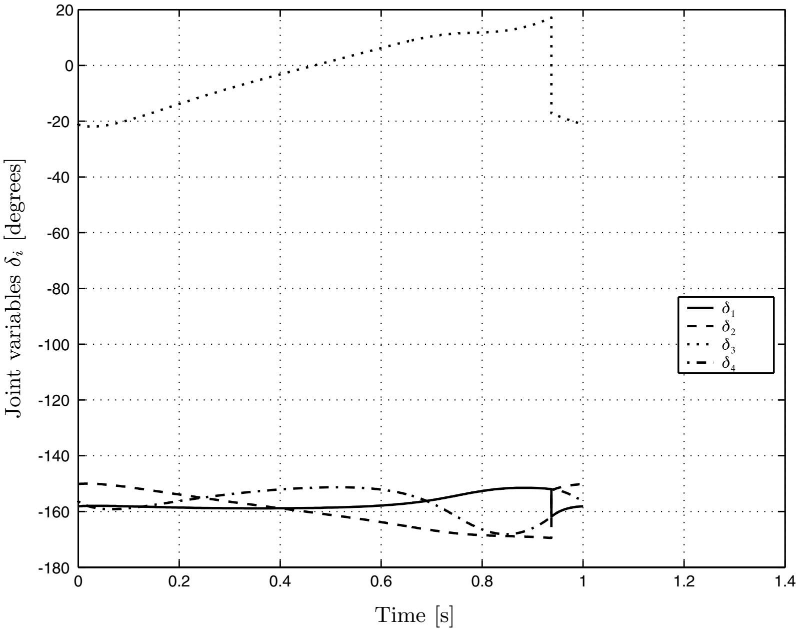


Fig. 6. Evolution of joint variables δ_1, δ_2, δ_3 et δ_4

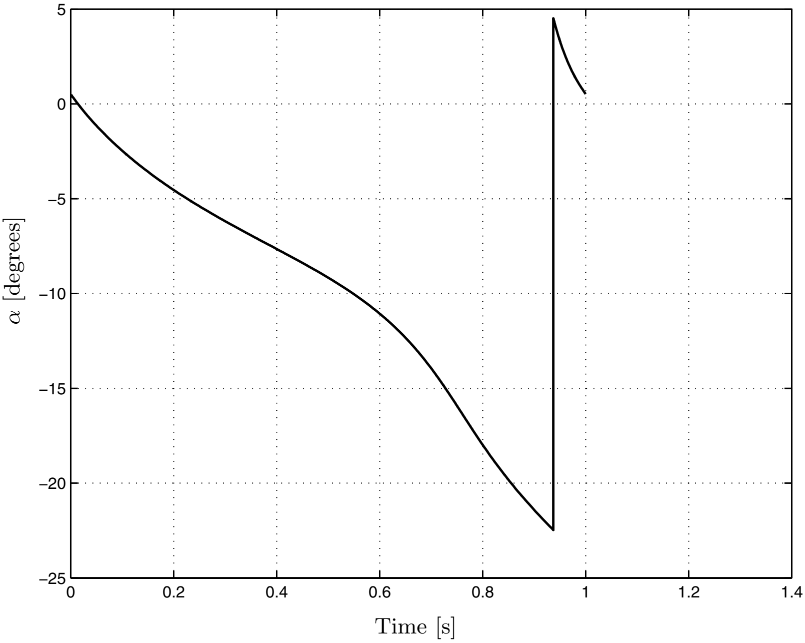


Fig. 7. Evolution of α in function of time

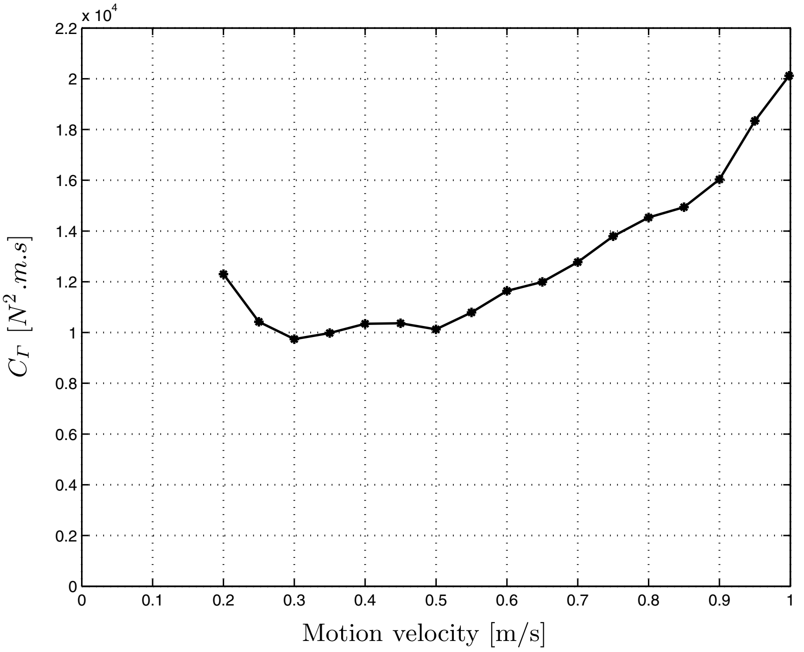


Fig. 8. C_Γ in function of several motion velocities for the biped

In conclusion, for the velocity 0.3 m/s of the biped an optimal motion is feasible according to the constraints. Other velocities of walk for the biped have been tested with success. In Fig. 8 discrete values of criterion C_{Γ} are presented versus the velocity of motion. The evolution of discrete criterion C_{Γ} versus velocity of motion is more regular if the optimal walks are obtained without taking into account Coulomb friction. This is due to the fact that the convergence for the case with friction is not very good, since torques are not smooth. For superior velocities a running gait is more appropriate, (see for example numerical experiments in the paper [18]).

6 Conclusion

An optimization process is proposed to design optimal bipedal gaits for a five-link biped. The walking gaits are composed of single-support phases and double-support phases, but with no impact. The criterion minimized is the integral of the square of the torques. A sequential procedure is done, taking into account the constraints according to their importance realizing a walk step. Coulomb frictions, which are nonlinear and discontinuous functions, are taken into account because their contribution cannot be neglected. A possible improvement would be to do a piecewise linear approximation of the Coulomb friction around the discontinuity point of the friction force for a null joint velocity. Currently the main drawback of the optimization method we used is that it is not exactly adapted to our problem. Our problem is a semi-infinite problem, that is an optimization problem with constraints that must be satisfied over an interval. We have then adapted our problem by considering the constraints over an interval only at their most constraining point. The optimization problem we then solve is with non-smooth constraints. But we obtained convergence even if *NPSOL* was not designed to cope with such non-smooth problems. To solve our problem, we would like to consider an optimization algorithm that can take into account a variable number of constraints in the future. Indeed, the number of maximum and minimum where we considered the semi-infinite constraints can change during the optimization process. We hope also to experiment on prototype *Rabbit* these reference trajectories and to extend also this work to a walking biped with more degrees of freedom.

References

[1] M. Vukobratovic and D. Juricic. Contribution to the synthesis of biped gait. In *Proc. of the IFAC Symp. Technical and Biological on Control*, Erevan, USSR, 1968.

[2] V. V. Beletskii and P. S. Chudinov. Parametric optimization in the problem of bipedal locomotion. *Izv. An SSSR. Mekhanika Tverdogo Tela [Mechanics of Solids]*, (1):25–35, 1977.

[3] A. M. Formal'sky. *Locomotion of Anthropomorphic Mechanisms.* Nauka, Moscow [In Russian], 1982.

[4] I. E. Sutherland and M. H. Raibert. Machines that walk. *Scientific American*, 248:44–53, 1983.

[5] K. Hirai, M. Hirose, and T. Haikawa. The development of honda humanoid robot. In *Proc. of the IEEE Conf. on Robotics and Automation*, pp. 1321–1326, 1998.

[6] K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, and T. Isozumi. Humanoid robot hrp-2. In *Proc. of the IEEE Conf. on Robotics and Automation*, pp. 1083–1090, 2004.

[7] A. A. Grishin, A. M. Formal'sky, A. V. Lensky, and S. V. Zhitomirsky. Dynamic walking of a vehicle with two telescopic legs controlled by two drives. *Int. J. of Robotics Research*, 13(2):137–147, 1994.

[8] F. Zonfrilli, M. Oriolo, and T. Nardi. A biped locomotion strategy for the quadruped robot sony ers-210. In *Proc. of the IEEE Conf. on Robotics and Automation*, pp. 2768–2774, 2002.

[9] J. Pratt, C. M. Chew, A. Torres, P. Dilworth, and G. Pratt. Virtual model control: an intuitive approach for bipedal locomotion. *International Journal of Robotics Research*, 20(2):129–143, 2001.

[10] C. Chevallereau, G. Abba, Y. Aoustin, F. Plestan, E.R. Westervelt, C. Canudas de Wit, and J. W. Grizzle. Rabbit: a testbed for advanced control theory. *IEEE Control Systems Magazine*, 23(5):57–79, 2003.

[11] S. Miossec and Y. Aoustin. A simplified stability study for a biped walk with under and over actuated phases. *International Journal of Robotics Research*, 24(7):537–551, 2005.

[12] M. W. Spong and M. Vidyasagar. *Robots Dynamics and Control.* John Wiley, 1989.

[13] S. Miossec. *Contribution à l'Étude de la Marche d'un bipède.* PhD thesis, Université Nantes France, 2004.

[14] A. M. Formal'sky. *Ballistic Locomotion of a Biped. Design and Control of Two Biped Machines.* Human and Machine Locomotion, Ed. A. Morecki and K. Waldron. Springer-Verlag, 1997.

[15] Y. Aoustin and A. M. Formal`sky. Control design for a biped: Reference trajectory based on driven angles as functions of the undriven angle. *Journal of Computer and Systems Sciences International*, 42(4):159–176, 2003.

[16] P. H. Channon, S. H. Hopkins, and D. T. Pham. Derivation of optimal walking motions for a bipedal walking robot. *Robotica*, 10(3):165–172, 1992.

[17] C. Chevallereau, A. Formal'sky, and D. Djoudi. Tracking of a joint path for the walking of an under actuated biped. *Robotica*, 22(1):15–28, 2003.

[18] C. Chevallereau and Y. Aoustin. Optimal reference trajectories for walking and running of a biped. *Robotica*, 19(5):557–569, 2001.

[19] L. Roussel, C. Canudas de Wit, and A. Goswami. Generation of energy optimal complete gait cycles for biped. In *Proc. of the IEEE Conf. on Robotics and Automation*, pp. 2036–2042, 2003.

[20] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. *User's guide for NPSOL 5.0.*

Performing Open-Loop Stable Flip-Flops – An Example for Stability Optimization and Robustness Analysis of Fast Periodic Motions

K. Mombaur

IWR, Universität Heidelberg, Im Neuenheimer Feld 368, 69120 Heidelberg,
Germany
katja.mombaur@iwr.uni-heidelberg.de

Summary. For fast motions in biomechanics and robotics, stability and robustness against perturbations are critical issues. The faster a motion the more important it is to exploit the system's natural stability properties for control. The stability of a periodic motion can be measured in terms of the spectral radius of the monodromy matrix. We optimize this stability criterion for a given robot topology, using special purpose optimization methods and leaving the model parameters, actuator inputs, trajectory start values and cycle time free to be determined by the optimization. This approach allows us to create simulations of robots that can move stably without any feedback. In order to analyze the robustness of a resulting periodic motion, we propose two methods, the first of which relies on forward simulations using perturbed start data and parameters while the second is based on the pseudospectra of the matrix. As a new example for a fast open-loop stable motion that has been produced by stability optimization, we present a biped gymnastics robot performing repetitive flip-flops (i.e. back handsprings). A similar model has previously been shown capable of performing open-loop stable running motions and repetitive somersaults.

1 The Role of Open-Loop Stability in Fast Motions

For fast walking and running motions, stability is a crucial property. Maintaining static stability throughout the motion is obviously not possibly for fast walking and running robots with just one or two legs. In contrast, those robots need to be allowed to tip and fall and swing, but the resulting motions have to be overall dynamically stable. Dynamically stable motions are characterized by the fact that they persist even in the presence of perturbations which always exist in a real world environment. The more complex the system becomes and the faster a motion gets, the more difficult is the task of stabilizing the motion by means of appropriate feedback control systems.

A comparison of the speed of running motions in biology and robotics still shows a huge gap: despite all technological process in recent years, the world's

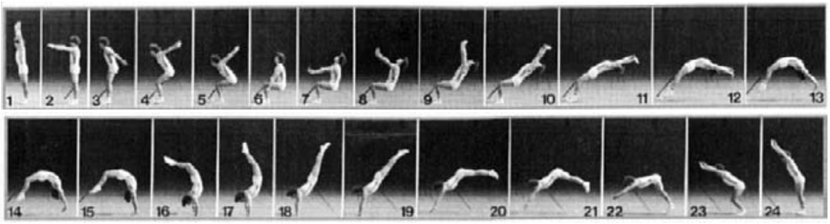


Fig. 1. A flip-flop (back handspring) motion in gymnastics (taken from [9], with permission)

fastest robots are still far behind the world's fastest track runners which run the 100 meters at average speed of nearly 37 km/h. One explanation for this is that biological feedback and signal processing of a running person based on senses and neurons is still far more sophisticated than any technical feedback system implemented on a robot. However, it seems even more important, that fast biological motions exploit the natural stability properties of the system much better than robots which typically rely on the traditional approach of trajectory playback. In biology, fast motions seem to be to some extent inherently stable, or self-stabilizing, which significantly reduces the online feedback effort.

In robotics, the idea of exploiting natural stability properties has been introduced mainly through the field of passive-dynamic walking (compare e.g. the work of McGeer [12, 13] or Ruina and coworkers [8]). Passive-dynamic robots walk down inclined slopes in an amazingly natural looking manner, without any motors, powered only by gravity, and are fully open-loop stable without any feedback.

There are different approaches to transfer the ideas of passive-dynamics to actuated walking and running motions (e.g. Collins et al. [6]). There only exist very few entirely open-loop stable actuated robots today, e.g. the one-legged hopping robots of Ringrose [21] and Wei et al. [24], and they rely mainly on simple stabilization measures like a large curved foot in combination with a low center of mass. There are also more complex robots that combine an exploitation of self-stabilizing effects and feedback control, see e.g. the works of Buehler [4] or Pratt [20]. Very fast open-loop stable multi-legged robots have been built inspired by cockroaches (Cham et al. [5]).

The idea of open-loop stability or exploitation of natural stability has been addressed by different authors of this symposium, e.g.:

- Martijn Wisse presented his robot Denise which is a quasi-passive walking robot with a very natural gait based on very little actuation and feedback
- Heiko Wagner and Peter Giesl explore the open-loop stability present in biological systems especially in the muscle actuators
- John Schmitt describes the SLIP model which can be passively stable and for which the basin of stability can be enlarged by very simple feedback.

We approach the issue of producing fast open-loop stable motions from an optimal control perspective. The objective of this paper is twofold:

- to summarize a systematic procedure based on numerical optimization that we developed [14, 15], to produce robot configuration and fast periodic motions that are fully open-loop stable, and to analyze their robustness. These numerical methods are meant to be used already in the design phase of the robots because they help to determine adequate geometric and inertial properties;
- to present one new specific example optimized with these methods: a two-legged robot performing repetitive flip-flops i.e. back handsprings.

This paper presents an extension of the work done in cooperation with Bock, Schlöder and Longman [17, 16]. Altogether this research shows that very different types of open-loop stable motions can be produced using these numerical techniques.

Since robustness is another crucial characteristic of a motion – besides stability – we also present different methods that allow one to numerically asses the robustness of a solution, i.e. to quantitatively determine the size of the tolerated perturbations.

In detail, the following steps are necessary in order to produce such a purely open-loop stable robot configuration and motion:

- Step 1: Choose the basic robot configuration and motion
- Step 2: Establish a mathematical model of the robot motion
- Step 3: Determine open-loop stable solutions by means of optimization of all model parameters and free input variables
- Step 4: Analyze robustness of solution.

The individual steps will be described in the following sections of this paper for the specific example of the biped robot performing repetitive flip-flops. A special focus is on the stability optimization in step 3, where the underlying numerical methods and the results of these methods for the robot under investigation will be presented.

2 First Step: Choosing the Robot Configuration and Motion – The Flip-Flop Example

In a first step the basic characteristics of the robot configuration and the motion to be stabilized need to be picked. In detail, that means that one has to select

1. the robot topology:
   i.e. determine the number of bodies, their basic shapes, the types of connections (or joints) between those bodies, the types of passive elements (like springs or dampers), the number and types of actuators etc.

2. the motion of the robot:
   i.e. describe the gait to be performed (like walking, running, trot, gallop etc.) or any other type of periodic motion (like jumping with flip etc.); this also includes a determination of the logical order of phases and of the natural phase-separating conditions etc.
3. the free quantities in the model:
   i.e. choose those model parameters, actuator inputs and characteristics of the trajectory that qualify for a modification by the optimization procedures.

In the present paper, we investigate a periodic flip-flop – or back handspring motion of a planer robot trying to mimic sequences of the human gymnastics movement (Fig. 1). Going along the items listed above, the following selections have been made:

1. The robot consists of a large trunk (in a bird-like horizontal orientation), two thin telescopic legs with point feet and two arms which are assumed to be identical to the legs in shape and size (compare Fig. 2). Since both arms and both legs act exactly in parallel for the planar motion considered, this original model can be substituted by a model with just one leg (pointing down from the trunk) and just one arm (pointing up, see Fig. 4). This model is in fact almost identical to the biped robot model that we have investigated in earlier publications and that we have proven capable of open-loop stable periodic running motions [17] and open-loop stable periodic somersaults with alternating single leg contacts [16] (also compare Fig. 3). The only difference is that one leg now has to be pointed upwards in order to be interpreted as an arm. Each leg/arm is connected to the trunk by a hinge driven by a torque and a parallel torsional spring-damper element. The lower part of the leg/arm is assumed to be massless, and the two parts of each leg/arm are connected by an actuated spring-damper element (series elastic actuator).

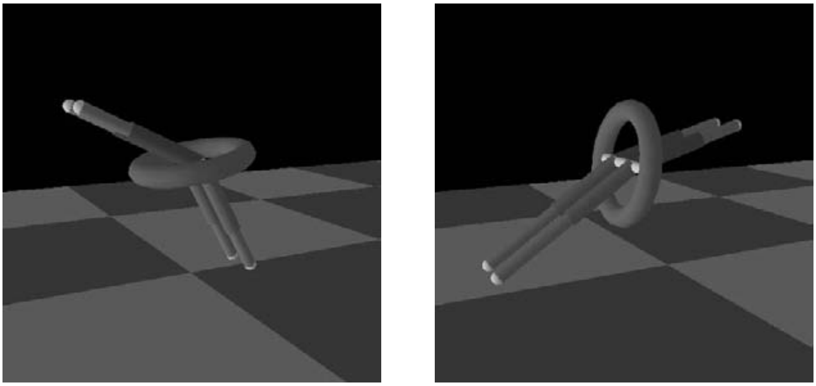


Fig. 2. A robot model performing flip-flop motions

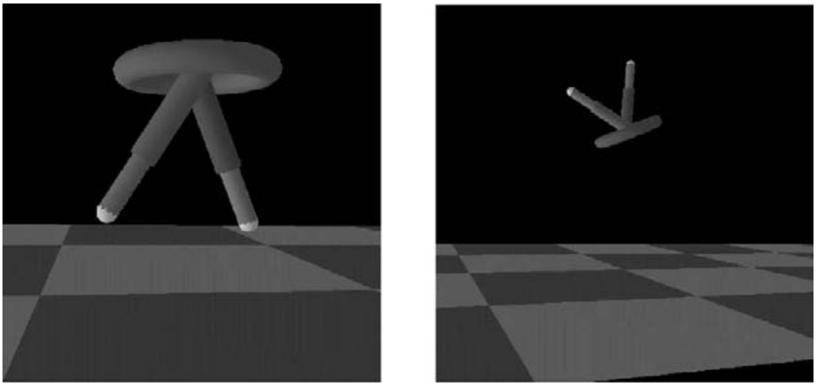


Fig. 3. Two-legged robot capable of periodic running and somersault motions

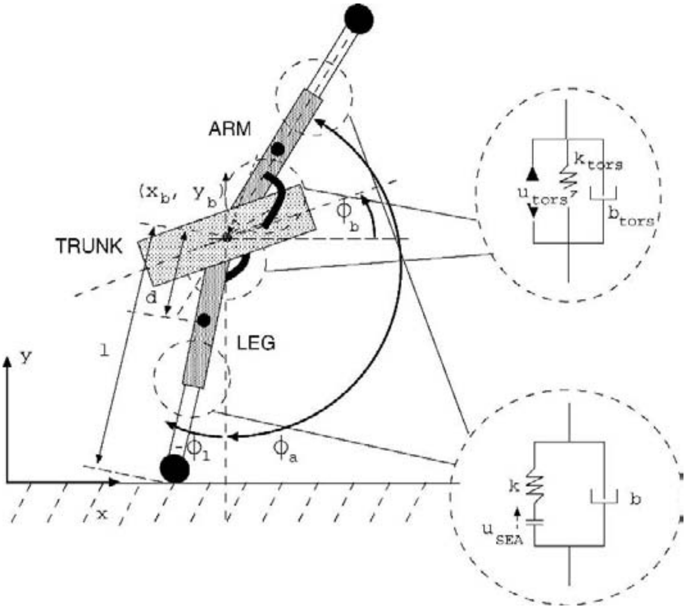


Fig. 4. Robot model used for flip-flop computations with one arm and one leg

2. As shown in Fig. 1, for the human flip-flop motion the following phases can be identified: foot contact phase – flight phase – hand contact phase – flight phase. Since our robot model is characterized by several symmetries not present in the human gymnast (legs and arms are identical, as well as the associated actuators, and the trunk is symmetric with respect to its horizontal plane), we can assume that foot and arm contacts have identical effects and reduce the model to half a physical cycle, i.e. just consider two phases: a contact phase followed by a flight phase. The natural condition

ending flight and starting the contact phase is satisfied if the respective foot or hand reaches zero height with a negative vertical velocity. The contact phase ends if the spring in the telescoping leg/arm regains its normal length after compression.

3. In order to optimize the open-loop stability of the flip-flip robot motion the following quantities are left free to be modified by the optimization procedures:
   - Most of the model parameters i.e. of the design variables that can be a priori chosen for a robot at design time and are fixed throughout the motion. Model parameters for this robot model are trunk mass and inertia m_b and Θ_b, leg/arm mass and inertia m_l and Θ_l, distance between centers of mass of trunk and leg d, leg rest length l_0, torsional spring and damper constants k_{tors} and b_{tors}, rest location of torsional spring $\Delta\phi$, and translational spring and damper constants k and b. For the computations presented here m_b and l are fixed for scaling reasons, the rest is left free. Note that all quantities correspond to the substitute model (Fig. 4).
   - All time-varying system inputs, i.e. the time histories of the forces and torques produced by motors and other types of actuators and acting on the model are also left free to be determined by optimization. Out of the four actuators described above – two torques u_{tors} between the hip and the leg and arm, respectively, and two series elastic actuators u_{SEA} in the telescopic part of the leg/arm – only three are active in the half cycle considered.
   - There are a few remaining free quantities describing the trajectory of the robot, like the initial values (at the beginning of a cycle) of all positions and velocities and the period time. The rest of the trajectory is implicitly determined by the dynamics and free quantities described above, like the parameters and the input histories.

   Even though all these quantities are free in optimization, they are subject to physically reasonable bounds.

3 Second Step: Establishing a Mathematical Model of the Robot Motion

After the general selections described in the previous section have been made, a detailed mathematical model of the robot and the motion has to be set up. From a mathematical perspective, gait models result in hybrid systems which include continuous motion phases with highly complex nonlinear differential equations and discrete "phases" (of duration zero) with sudden – discontinuous – changes in the state variables of the system[1]. The number of degrees of freedom, the number of state variables, as well as the number of free control

[1] State variables x for a mechanical system are all position and velocity variables used for description

variables $u(t)$ (i.e. actuator inputs in physical terms) can be different in each phase. Typically, the points of phase change do not take place at given times but only depend implicitly on time via the states of the system.

In order to fully describe the motion of the flip-flop robot, we have to establish the following sets of equations:

- differential equations for the leg contact phase
- differential equations for the flight phase
- discrete equations for discontinuity "phase" after touchdown.

We use the same set of state variables for the description of all phases, namely $q = (x_b, y_b, \phi_b, \phi_l, \phi_a)^T$, and the corresponding velocities $\dot{q}$, where x_b and y_b are the position coordinates of the trunk center of mass in the vertical plane, and ϕ_b, ϕ_l and ϕ_a are the orientations of trunk, leg and arm. During the flight phase, these position variables correspond to the five degrees of freedom; and the reduction during contact phase is handled by additional constraints.

The coordinates of the centers of mass of the leg and arm (x_l, y_l) and (x_a, y_a) can be eliminated using the distance parameter d and the respective angle by ϕ_l or ϕ_a.

The motion in the flight phase is described by the following set of ordinary differential equations:

$$\begin{pmatrix} m & 0 & 0 & m_l d\cos\phi_l & m_l d\cos\phi_a \\ 0 & m & 0 & m_l d\sin\phi_l & m_l d\sin\phi_a \\ 0 & 0 & \theta_b & 0 & 0 \\ m_l d\cos\phi_l & m_l d\sin\phi_l & 0 & \theta_l + m_l d^2 & 0 \\ m_l d\cos\phi_a & m_l d\sin\phi_a & 0 & 0 & \theta_l + m_l d^2 \end{pmatrix} \begin{pmatrix} \ddot{x}_b \\ \ddot{y}_b \\ \ddot{\phi}_b \\ \ddot{\phi}_l \\ \ddot{\phi}_a \end{pmatrix} =$$

$$\begin{pmatrix} m_l d(\sin\phi_l \dot{\phi}_l^2 + \sin\phi_a \dot{\phi}_a^2) \\ -m_l d(\cos\phi_l \dot{\phi}_l^2 + \cos\phi_a \dot{\phi}_a^2) - mg \\ \sum_{i=1}^{2} \Big(u_{tors,i} - k_{tors}(\phi_b - \phi_{l_i} - \varDelta\phi) - b_{tors}(\dot{\phi}_b - \dot{\phi}_{l_i})\Big) \\ -u_{tors,1} - m_l g d\sin\phi_l + k_{tors}(\phi_b - \phi_l - \varDelta\phi) + b_{tors}(\dot{\phi}_b - \dot{\phi}_l) \\ -u_{tors,2} - m_l g d\sin\phi_a + k_{tors}(\phi_b - \phi_a - \varDelta\phi) + b_{tors}(\dot{\phi}_b - \dot{\phi}_a) \end{pmatrix} \tag{1}$$

where m is the total mass $m = m_b + 2m_l$ and $u_{tors,1}$ and $u_{tors,2}$ are the torques between trunk and leg or arm, respectively. For compactness of notation, we use $\phi_{l_1} = \phi_l$ and $\phi_{l_2} = \phi_a$ in the third line of eqn. (1).

The leg length l is fixed to l_0 during the major part of the flight phase (since the foot is massless) and depends on the other coordinates during the contact phase (with the leg) as follows:

$$l = \frac{y_b}{\cos\phi_l} \Rightarrow \tag{2}$$

$$\dot{l}_i = \frac{\dot{y}_b}{\cos\phi_l} + y_b \frac{\sin\phi_l}{\cos^2\phi_l}\dot{\phi}_l \ . \tag{3}$$

(equivalently for arm contact).

The series elastic actuator (compare Pratt et al. [19]) active in the contact leg $u_{SEA,l} \geq 0$ actively changes the spring's length

$$\Delta l = \frac{y_b - r}{\cos\phi_l} + r - u_{SEA,l} - l_0 \tag{4}$$

$u_{SEA,l}$ is equal to zero at touchdown and must be positive at liftoff to compensate for the energy loss in the damper.

In order to model foot contact, we use a constraint-based approach instead of contact forces: friction is assumed to be large enough to guarantee instantaneous stopping of the contact point without sliding. This would reduce the degrees of freedom of the system by two, but at the same time the previously fixed leg length becomes variable and changes under the influence of the SEA spring-damper forces. In sum, this leads to reduction of degrees of freedom by one which is described by the additional kinematic constraint in velocity space

$$\dot{x}_b + (y_b + y_b \tan^2\phi_l\ )\ \dot{\phi}_l + \tan\phi_l\ \dot{y}_b = 0\ . \tag{5}$$

The corresponding equation on the acceleration level is used to establish the differential-algebraic equations of index 1 describing the contact phase:

$$\begin{pmatrix} m & 0 & 0 & m_l d\cos\phi_l & m_l d\cos\phi_a & 1 \\ 0 & m & 0 & m_l d\sin\phi_l & m_l d\sin\phi_a & tan\phi_l \\ 0 & 0 & \theta_b & 0 & 0 & 0 \\ m_l d\cos\phi_l & m_l d\sin\phi_l & 0 & \theta_l + m_l d^2 & 0 & y_b(1+\tan^2\phi_l) \\ m_l d\cos\phi_a & m_l d\sin\phi_a & 0 & 0 & \theta_l + m_l d^2 & 0 \\ 1 & \tan\phi_l & 0 & y_b(1+\tan^2\phi_l) & 0 & 0 \end{pmatrix} \begin{pmatrix} \ddot{x}_b \\ \ddot{y}_b \\ \ddot{\phi}_b \\ \ddot{\phi}_l \\ \ddot{\phi}_a \\ \lambda \end{pmatrix} =$$

$$\begin{pmatrix} m_l d(\sin\phi_l\dot{\phi}_l^2 + \sin\phi_a\dot{\phi}_a^2) + (F_k + F_d)\sin\phi_l \\ -m_l d(\cos\phi_l\dot{\phi}_l^2 + \cos\phi_a\dot{\phi}_a^2) - mg - (F_k + F_d)\cos\phi_l \\ \sum_{i=1}^{2}\left(u_{tors,i} - k_{tors}(\phi_b - \phi_{l_i} - \Delta\phi) - b_{tors}(\dot{\phi}_b - \dot{\phi}_{l_i})\right) \\ -u_{tors,1} - m_l g d\sin\phi_l + k_{tors}(\phi_b - \phi_l - \Delta\phi) + b_{tors}(\dot{\phi}_b - \dot{\phi}_l) \\ -u_{tors,2} - m_l g d\sin\phi_a + k_{tors}(\phi_b - \phi_a - \Delta\phi) + b_{tors}(\dot{\phi}_b - \dot{\phi}_a) \\ -2\cdot\cos^{-2}\phi_l\dot{\phi}_l(\dot{y}_b + y_b\tan\phi_l\dot{\phi}_l) \end{pmatrix} \tag{6}$$

with spring and damper forces F_k and F_d

$$F_k = k\ (\frac{y_b}{\cos\phi_l} - l_0 - u_{SEA,1}) \tag{7}$$

$$F_d = b\ (\frac{\dot{y}_b}{\cos\phi_l} + y_b\frac{\tan\phi_l}{\cos\phi_l}\dot{\phi}_l)\ . \tag{8}$$

The solution of these equations must lie on the invariant described by the velocity equation (5); this is guaranteed by including this equation in the computations of the discrete phase (see below).

Before addressing this discrete phase, we briefly state all phase switching conditions:

- Phase change from contact phase to flight pase, i.e. liftoff, occurs when the spring length in the leg is equal to the (modified) rest length

$$s_{liftoff} = l_0 + u_{SEA,1} - \frac{y_b}{\cos\phi_l} = 0 \tag{9}$$

  and, at the same time, the trunk has a positive vertical speed

$$c_{liftoff} = \dot{y}_b > 0\ . \tag{10}$$

- Touchdown, i.e. phase change from flight phase to the following contact phase (arm contact) occurs when the height of the prospective contact point is equal to zero

$$s_{touchdown} = y_b - l_0 \cos\phi_a = 0\ . \tag{11}$$

  The vertical speed of the contact point at touchdown must be negative:

$$c_{touchdown} = \dot{y}_b + l_0 \sin\phi_a \dot{\phi}_a < 0\ . \tag{12}$$

Discrete phases are used to describe sudden changes of the state variables due to collisions etc.; in the model the time of such an event is assumed to be zero. The lift-off phase transition is assumed to be smooth, so no discrete phase is inserted at this point. However, touchdown is generally non-smooth. The velocity of the contact point is instantaneously set to zero, and the discontinuity propagates to all other parts of the system. A set of five algebraic equations is used to determine the five unknown velocities after impact with the arm:

- non-sliding ground contact combined with spring-damper action:

$$\dot{x}_{contact} = \dot{x}_b + l_0 \cos\phi_a \dot{\phi}_a + \dot{y}_b \tan\phi_a + y_b \dot{\phi}_a \tan^2\phi_a = 0 \tag{13}$$

- conservation of angular momentum of trunk about hip:

$$H_{trunk,hip} = \Theta_b \dot{\phi}_b = const\ . \tag{14}$$

- conservation of angular momentum of the leg (remaining in free swing phase) about the hip

$$H_{swingleg,hip} = (\Theta_l + m_l d^2)\dot{\phi}_l = const\ .$$

- conservation of angular momentum of full robot about prospective contact point

$$\begin{aligned} H_{robot,contact} = {} & \Theta_b \dot{\phi}_b - m_b(y_b - y_c)\dot{x}_b + m_b(x_b - x_c)\dot{y}_b \\ & + \Theta_l \dot{\phi}_l - m_l(y_l - y_c)\dot{x}_l + m_l(x_l - x_c)\dot{y}_l \\ & + \Theta_l \dot{\phi}_a - m_l(y_a - y_c)\dot{x}_a + m_l(x_a - x_c)\dot{y}_a = const\ . \end{aligned} \tag{15}$$

with

$$x_c = x_b + l_0 \sin \phi_a \tag{16}$$

$$y_c = y_b - l_0 \cos \phi_a \ . \tag{17}$$

- conservation of translational momentum in direction of prospective stance arm (considering spring-damper-force)

$$m(\dot{x}_b \sin \phi_a - \dot{y}_b \cos \phi_a) - F_{kd} = const \ . \tag{18}$$

The model presented so far is identical (except for notational details) for all robots of the same family, the running robot, the somersault robot and the flip-flop robot.

However, using additional inequality constraints, it is possible to distinguish one type of motion from the other and impose the desired type. For flip-flops these constraints are e.g.

- at foot contact, the foot is pointing down and forward while the arm is pointing up and backwards
- leg and arm are always on different sides of the trunk which must be assured by appropriate collision avoidance constraints
- leg and arm have a negative rate during the whole cycle (i.e. are continuously rotating backwards)

Also the periodicity constraints (i.e. coupled equality constraints) applied to the model are different for each type of motion. In order to obtain the desired periodic flip-flop with symmetric leg and arm contact half-cycles, the following modified periodicity constraints have to be imposed:

$$\begin{aligned}
y_b(0) &= y_b(T) \\
\phi_b(0) &= \phi_b(T) - \pi \\
\phi_l(0) &= \phi_a(T) - 2\pi \\
\phi_a(0) &= \phi_l(T) \\
\dot{x}_b(0) &= \dot{x}_b(T) \\
\dot{y}_b(0) &= \dot{y}_b(T) \\
\dot{\phi}_b(0) &= \dot{\phi}_b(T) \\
\dot{\phi}_l(0) &= \dot{\phi}_a(T) \\
\dot{\phi}_a(0) &= \dot{\phi}_l(T)
\end{aligned} \tag{19}$$

Additionally, the following inequality constraints need to be satisfied for any type of motion:

- clearance of swing leg and arm (height of lowest point larger than zero)
- contact avoidance of all bodies.

4 Numerical Stability Optimization Methods for Step 3

We address the task of step 3 to determine open-loop stable solutions by means of numerical optimization. A problem like this can not be solved by standard optimization or optimal control methods due to

- the complexity of the robot gait models described above, namely the non-smoothness in the dynamics (i.e. the hybrid system property)
- the difficulties produced by the choice of the optimization criterion stability (which is another source of non-smoothness of an even more complicated type, see below).

In our previous research we therefore have developed numerical methods for the optimization of open-loop stability of general hybrid periodic systems. The method described here depends on a two-level approach splitting the problem into periodic gait generation and stabilization of the periodic system (for details see [14, 15]). This approach has already been applied to many types of problems and robot examples and has always delivered stable solutions [14, 17, 16].

Figure 5 gives an overview of the two-level procedure. The tasks to be performed in each level will be presented in the next two paragraphs.

4.1 Solution of Periodic Optimal Control Problem

The task of the lower level – or inner optimization loop is to find actuator patterns, initial values and cycle time leading to a periodic trajectory while the set of parameters is fixed by the outer loop for each inner loop computation. The choice of those variables is governed by energy consumption considerations (in terms of actuator inputs u). We also have imposed a lower bound on

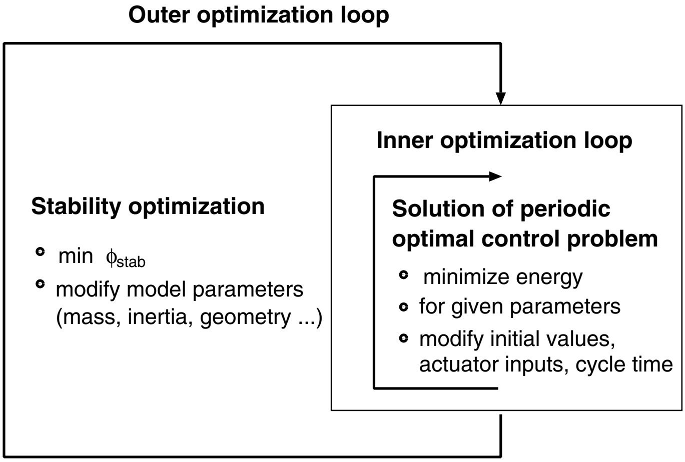

Outer optimization loop
Inner optimization loop
Stability optimization
min ϕ_{stab}
modify model parameters (mass, inertia, geometry ...)
Solution of periodic optimal control problem
minimize energy
for given parameters
modify initial values, actuator inputs, cycle time

Fig. 5. Sketch of two-level stability optimization procedure

the trunk forward speed at all points, and bounds on the leg inclination angle at touchdown and liftoff instants. Together with the equations of motion, the periodicity constraints and phase switching conditions, box constraints on all variables etc., this leads to a multi-phase optimal control problem of the following form:

$$\min_{x,u,T} \int_0^T ||u||_2^2 dt \tag{20}$$

$$\begin{aligned}
\text{s. t.} \qquad \dot{x}(t) &= f_j(t, x(t), u(t), p) \quad \text{or DAE} && (21)\\
x(\tau_j^+) &= h(x(\tau_j^-)) && (22)\\
g_j(t, x(t), u(t), p) &\geq 0 && (23)\\
\text{for} \quad & t \in [\tau_{j-1}, \tau_j] \ , \\
& j = 1, \ldots, n_{ph}, \ \tau_0 = 0, \tau_{n_{ph}} = T \\
r_{eq}(x(0), .., x(T), p) &= 0 && (24)\\
r_{ineq}(x(0), .., x(T), p) &\geq 0 \ . && (25)
\end{aligned}$$

We solve this problem using a variant of the optimal control code MUSCOD (Bock & Plitt [3], Leineweber [10]) suited for periodic gait problems. It is based on

- a direct method for the solution of the optimal control problem (also termed a "first discretize then optimize method":
  instead of using arbitrary (i.e. infinite dimensional) control functions $u(t)$, we restrict the controls to a discretized space described by a finite set of parameters. For numerical efficiency, we use functions with local support, in this case piecewise constant functions on a grid with m intervals.
- a multiple shooting state parameterization:
  the basic idea of this technique is to split the long integration interval $[0, T]$ into many smaller ones and to introduce the values of the state variables x at all those grid points as new variables s_{ij} (compare Fig. 6). The original boundary value problem is thus transformed into a set of initial value problems with corresponding continuity conditions between the integration intervals. For numerical reasons the multiple shooting grid is chosen identical to the control grid described above. The multiple shooting approach is very favorable for a number of reasons. The phase order and switching structure, which is generally known for walking problems, can be easily prescribed in this context. The rough knowledge that one usually has about the trajectory in the case of walking and running can be exploited in the generation of starting data for the multiple shooting points. Since the integration intervals are much shorter than in the original problem, the chances for finding a solution of the initial value problem obtaining sufficiently accurate derivatives increase significantly, even if all values are still far from the final solution.

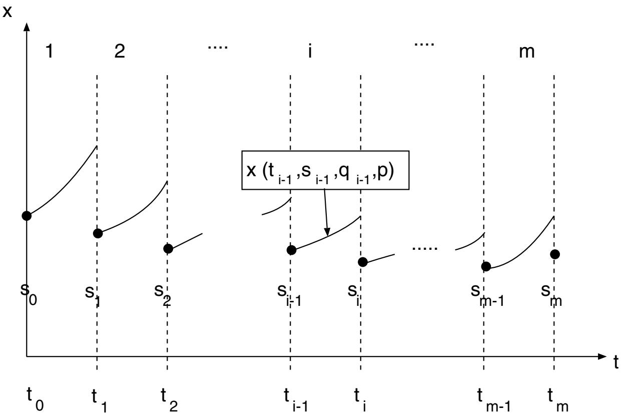


Fig. 6. Multiple Shooting state parameterization

Using these discretization techniques, the original infinite dimensional optimal control problem is transformed into a finite dimensional nonlinear programming problem which is of large dimension but very structured and can therefore be solved efficiently by a tailored SQP algorithm exploiting the structure of the problem (also compare Leineweber et al. [10] [11]).

The treatment of the dynamical model equations is not part of the discretized optimal control problem; this task must however be handled in parallel in order to provide the required information for the evaluation of objective functions, continuity constraints and the derivatives thereof. For this task, fast and reliable integrators are used that also include a computation of sensitivities based on the techniques of internal numerical differentiation (for details see Bock [2]).

4.2 Stability Optimization of Periodic Solution

In the outer loop of the optimization procedure the open-loop stability of the periodic optimal problem solution is optimized by adjusting the model parameters. Stability is defined in terms of the spectral radius of the Jacobian C of the Poincaré map – also termed the monodromy matrix – associated with the periodic solution (see e.g. [22]). If the spectral radius is smaller than one, the solution is asymptotically stable, and if it is larger than one, the solution is unstable. We have proven that this criterion based on linear theory and typically applied to simple smooth systems can also be used to demonstrate the stability of solutions of a nonlinear multiphase system with discontinuities (Mombaur et al. [15]).

The computation of the monodromy matrix corresponding to a periodic solution of the hybrid system can be performed efficiently by reusing sensitivity information computed during the optimal control problem solution. The following steps are required:

a) For continuous model phases, a simple chain rule multiplication of the sensitivity matrices on the individual multiple shooting intervals must be performed:

$$C_{q,v}(t_0, t_m) = C_{q,v}(t_{m-1}, t_m) \cdot ... \cdot C_{q,v}(t_1, t_2) \cdot C_{q,v}(t_0, t_1) \ . \tag{26}$$

b) At the points of discontinuity t_s – like the touch-down or lift-off of the robot – an "update" of the sensitivity information has to be performed. This update term has to take into account that these discontinuity points are not fixed in time, but implicitly depend on the states and parameters and will therefore experience a time shift under the influence of a perturbation of these variables. It is computed according to the following formula (compare Bock [2], von Schwerin et al. [23]):

$$U_{q,v} = (\Delta f - J_t - J_{q,v} f_{left}(t_s)) \cdot \frac{1}{\dot{s}} (s_q, s_v)^T + I + J_{q,v} \ . \tag{27}$$

f, q, and v are defined as in Sect. 2, and s is the relevant switching function with partial derivatives s_q and s_v and total derivative with respect to time $\dot{s}$. Subscripts *left* and *right* always denote quantities before and after the switching point, respectively. I is the identity matrix. J is the state variable jump function with the partial derivatives J_t and $J_{q,v}$. Δf is the right hand side change. The full monodromy matrix including the update becomes

$$C_x := C_{q,v}(t_0, t_m) = C_{q,v}(t_s, t_m) U_{q,v} C_{q,v}(t_0, t_s) \tag{28}$$

c) If the models include non-periodic variables, a subsequent reduction to the periodic subspace is necessary to compute the matrix to be used in optimization.

The eigenvalues of this matrix which is quadratic and nonsymmetric can be computed using a standard QR algorithm as available in Lapack (Anderson et al. [1]); these eigenvalues may be real or complex.

We use the spectral radius as objective function of our optimization

$$\min_p |\lambda(C(p))|_{max}, \tag{29}$$

with the intention to decrease it below one.

This is a difficult optimization criterion for different reasons:

- The maximum eigenvalue function of the non-symmetric matrix C is non-differentiable and possibly even non-Lipschitz at points where multiple eigenvalues coalesce.
- The determination of the matrix C involves the computation of first order sensitivities of the discontinuous trajectories (see above).
- The function is non-convex and typically has several local minima.

Any gradient-based optimization method would thus require second order derivatives of the trajectory which are extremely hard to compute, especially due to the discontinuities in the dynamics. For all reasons mentioned, a direct search method has proven to be a very good choice for the solution of this outer loop optimization problem. Direct search methods are optimization methods that solely use function information and neither compute nor explicitly approximate derivatives. We have implemented a modification of the Nelder-Mead algorithm [18] which is based on a polytope with $n+1$ vertices for optimization in n-dimensional space. According to the function information collected at its vertices the polytope expands in directions promising descent and contracts in bad directions. In contrast to the original method, we allow for multiple expansions in a promising direction, we use a different direction of contraction, and we only apply full polytope shrinking after multiple one-dimensional contractions. In addition, we consider the different nature of optimization variables by appropriate scaling of the initial polytope, we use a modified termination criterion, and we rely on a restart procedure as globalization strategy. In contrast to the original Nelder-Mead method, our algorithm can directly handle box constraints on the optimization variables not requiring a penalty function. Although there is no theoretical convergence proof for this direct search method in the case of non-smooth systems, it has delivered excellent results with spectral radii below one in all the computational examples that we have applied it to so far.

5 Results of Step 3: Open-Loop Stable Repetitive Flip-Flops

Applying these stability optimization methods to the model of the periodic flip-flop established in step 2, leads to the open-loop stable motion visualized in Fig. 7. It is characterized by a maximum eigenvalue of magnitude 0.807.

The robot is characterized by the following model parameters: $m_b = 2.0$, $\Theta_b = 1.0$, $m_l = 0.848$, $\Theta_l = 0.0174$, $d = 0.24$, $l_0 = 0.5$, $k_{tors} = 10.07$, $\Delta\phi = 0.969$, $b_{tors} = 7.41$, $k = 701.1$, and $b = 23.16$ (all in ISO units). Note that all parameter values are given for the substitute model; going back to the original model with two legs and two arms requires cutting in half m_l, Θ_l, and all spring and damper constants.

Figures 8 and 9 show the state variable histories and actuator inputs for this open-loop stable solution. The initial values of the trajectory are $x_b(0) = 0.0$, $y_b(0) = 0.439$, $\phi_b(0) = 0.2$, $\phi_l(0) = 0.5$, $\phi_a(0) = 4.273$, $\dot{x}_b(0) = 5.745$, $\dot{y}_b(0) = -1.549$, $\dot{\phi}_b(0) = -9.659$, $\dot{\phi}_l(0) = -8.598$, $\dot{\phi}_a(0) = -8.804$.

The cycle time is $T = 0.3\,\text{s}$ with phase times $T_{flight} = 0.15\,\text{s}$ and $T_{contact} = 0.15\,\text{s}$ (implicating that both phase times reach their lower limits as specified).

The direction associated with the non-periodic variable x_b is eliminated from the monodromy matrix, such that the matrix considered for stability computations has dimension nine. The periodic mapping described by this

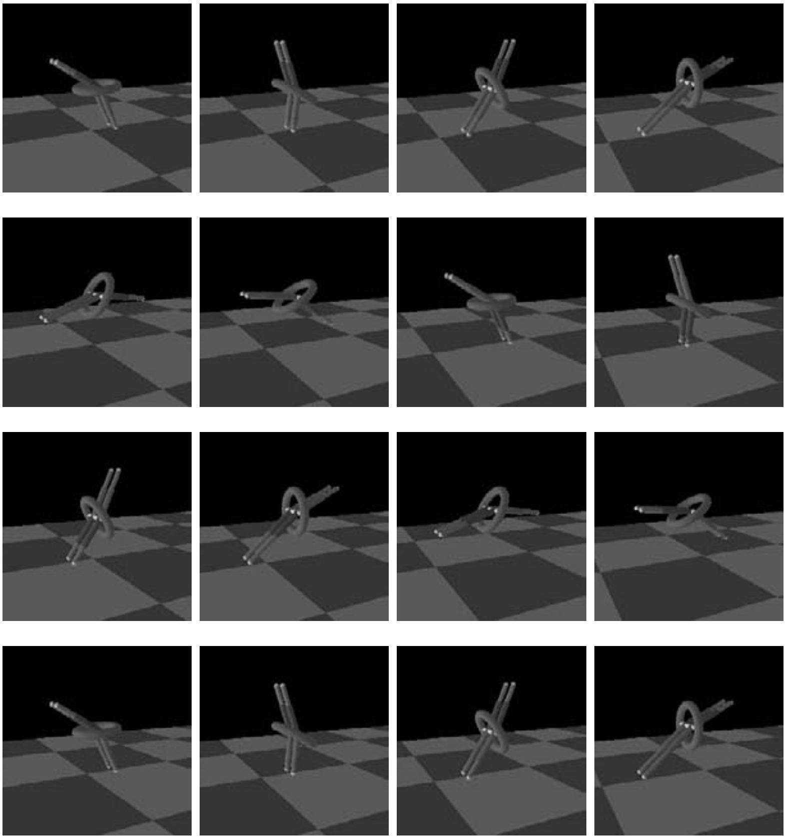


Fig. 7. Open-loop stable flip-flops produced by optimization

matrix also includes the shift of arm and leg and of trunk orientation expressed in the modified periodicity constraints (19). The computations resulted in the nine eigenvalues, by magnitude:

$$\begin{aligned}
|\lambda_1| &= 0.807 \\
|\lambda_2| &= 0.389 \\
|\lambda_3| &= 0.225 \\
|\lambda_4| &= 0.797 \\
|\lambda_5| &= 0.644 \\
|\lambda_6| &= 0.665 \\
|\lambda_7| &= 2 \cdot 10^{-6} \\
|\lambda_8| &= 0 \\
|\lambda_9| &= 0
\end{aligned}$$

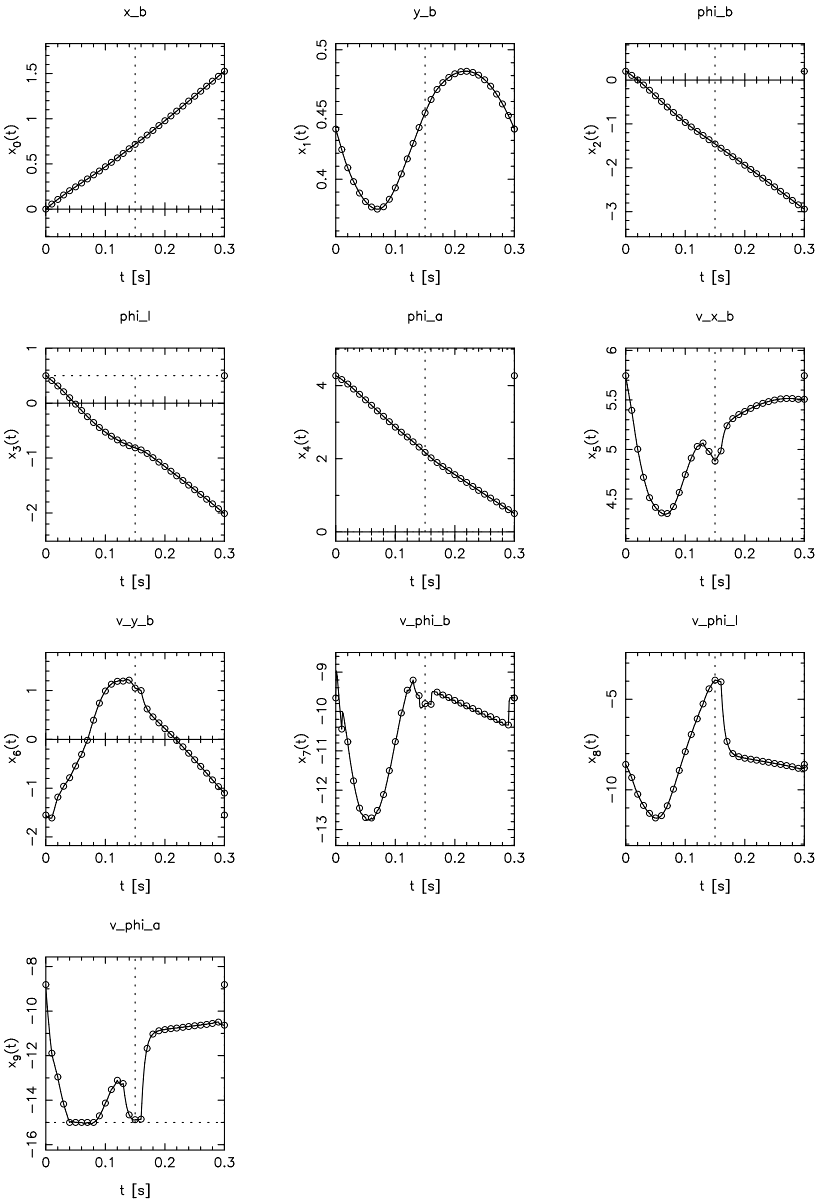


Fig. 8. State variable trajectories of open-loop stable flip-flop solution

The two eigenvalues of zero come from the fact that the degrees of freedom of the robot are reduced from five to four during the contact phase (i.e. from ten to eight independent directions in state space). This also implies that perturbations associated with this lost degree of freedom are naturally damped

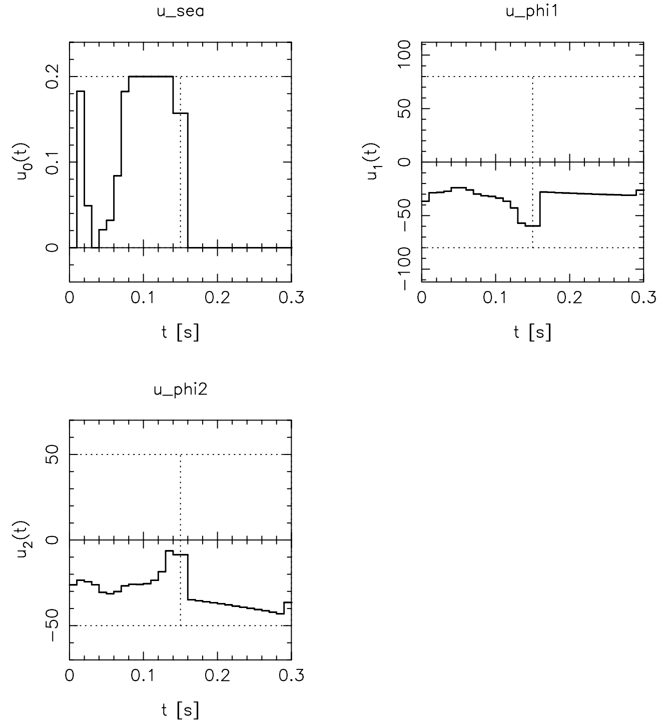


Fig. 9. Actuator inputs producing flip-flop motion in Fig. 8

out which is represented by a rank reduction by two of the monodromy matrix, and thus by two zero eigenvalues. Another eigenvalue happens to be very small for this particular solution, but it is not analytically or numerically zero in the general case.

6 Fourth Step: Robustness Analysis of Solution

A maximum eigenvalue with a magnitude below one guarantees stability of the solution against small perturbations in the initial value. But how small is small? The absolute size of the maximum eigenvalue only gives some information about the speed of decay over time of the perturbations applied, not about the absolute size of the perturbations that are possible before the robot falls down. An analysis of the stability margins of a solution, i.e. of its robustness, is also very important in order to asses the quality of a solution. Stability of a solution is a prerequisite for robustness, but stability does not necessarily imply robustness.

There is no straightforward way to numerically compute the robustness of a solution including all nonlinear effects in terms of a single function, but we present two quick numerical tests that can help to evaluate the robustness of a given solution.

6.1 Test 1: Simulation of Perturbed Solutions

We can determine stability margins numerically by applying one-dimensional perturbations to the initial values of the trajectory and simulating the resulting behavior of the system checking if it stumbles or if it returns to the periodic motion. This test includes all nonlinear effects associated with the system and the motion including:

- nonlinearities of the model equations
- closeness to constraints, especially to phase-switching constraints (perturbations may cause new phase switching structures to appear)

As an example, we investigate here the effect of perturbations of the initial vertical velocity $\dot{y}(0)$. The boundary of the basin of attraction of the stable solution is above +30% of the reference value. Figure 10 shows the recovery process after a perturbations of +30%.

6.2 Test 2: Pseudospectra of Monodromy Matrix

Another completely different way of assessing the robustness of a solution is to look at the pseudospectra of the computed monodromy matrix. The pseudospectra can be defined in terms of the spectra of all nearby, i.e. perturbed matrices:

$$\Lambda_\epsilon(C) = \{z \in \mathbb{C} : z \in \Lambda(C+E) \text{for some E with} ||E|| < \epsilon\} \tag{30}$$

where Λ denotes the spectrum of a matrix. For more information on pseudospectra, including other, equivalent definitions and useful tools, see the Pseudospectra Gaitway by Embree and Trefethen [7]. Pseudospectra capture the fact that nonsymmetric matrices, i.e. matrices without an orthogonal basis of eigenvectors may exhibit transients or other types of effects which are different from the asymptotic behavior predicted by the eigenvalues. The pseudospectrum is used to define measures of robust stability and can help to qualitatively compare different solutions. However, one has to keep in mind that this analysis only includes information about the sensitivity of the spectral radius with respect to matrix entries and not about the sensitivities of the matrix entries with respect to the free optimization variables.

The pseudospectrum associated with the most stable solution of the flip-flop robot is given in Fig. 11.

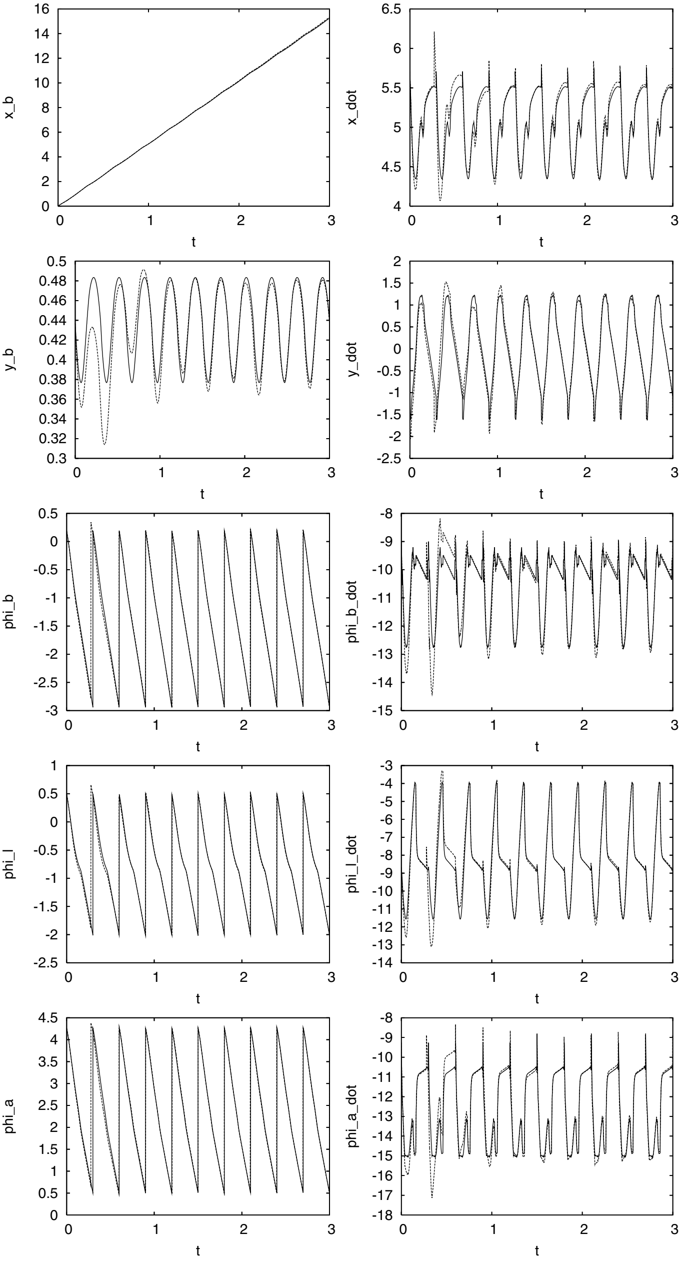


Fig. 10. Effect of a perturbation (of $\dot{y}(0)$ by +30%) on most stable trajectory of flip-flop robot (stable base solution is shown by *solid line*, perturbed solution by *dashed line*)

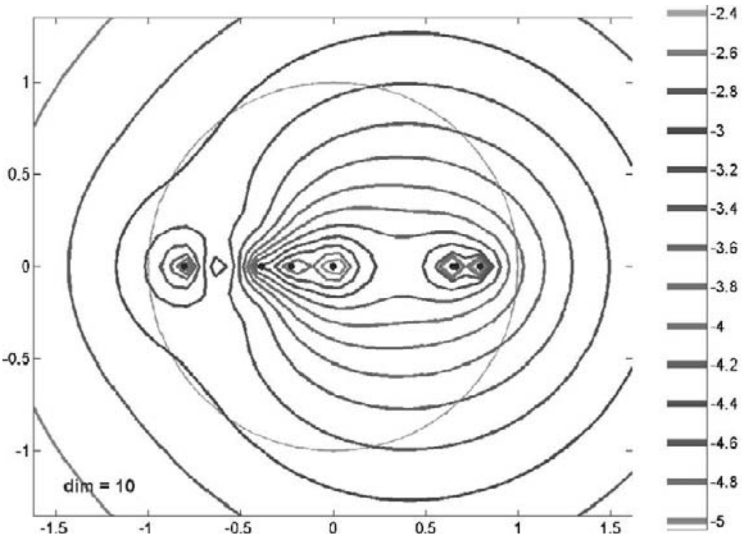


Fig. 11. Pseudospectra of monodromy matrix corresponding to most stable flip-flop solution computed with Eigtool [7]

7 Conclusions

We have presented an animated biped robot that is capable of performing repetitive open-loop stable flip-flop motions. This adds a new type of motion to the range of periodic motions that are possible without any feedback. Producing these simulations was only possible by means of special purpose numerical stability optimization methods that also have been briefly described in this paper. In addition, two numerical tests for the robustness of the solution have been presented.

Acknowledgments

I would like to thank the anonymous reviewer of [16] who suggested to investigate the open-loop stability of flip-flops as a next example. Financial support by the Margarete von Wrangell program of the State of Baden-Württemberg is gratefully acknowledged.

References

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostruchov, and D. Sorensen. *Lapack Users' Guide.* SIAM, 1995.

[2] H. G. Bock. Randwertproblemmethoden zur Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen. In *Bonner Mathematische Schriften 183*. Universität Bonn, 1987.

[3] H. G. Bock and K.-J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings of the 9th IFAC World Congress, Budapest*, pp. 242–247. International Federation of Automatic Control, 1984.

[4] M. Buehler. Dynamic locomotion with one, four and six-legged robots. *Journal of the Robotics Society of Japan*, 20(3):15–20, April 2002.

[5] J. G. Cham, S. A. Bailey, J. E. Clark, R. J. Full, and M. R. Cutkosky. Fast and robust: Hexapedal robots via shape deposition manufacturing. *International Journal of Robotics Research*, 21(10-11):869–882, 2002.

[6] S. H. Collins, A. L. Ruina, R. Tedrake, and M. Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307:1082–1085, 2005.

[7] Mark Embree and Nick Trefethen. Pseudospectra gaitway. *web.comlab.ox.ac.uk/ projects/pseudospectra/index.html*, 2005.

[8] M. Garcia, A. Ruina, M. Coleman, and A. Chatterjee. Some results in passive-dynamic walking. In *Biology and Technology of Walking*, pp. 268–275. Euromech Colloquium 375, Munich, 1998.

[9] Kurt Knirsch. *Lehrbuch des Gerät- und Kunstturnens*, vol. 1. Knirsch-Verlag, 1997.

[10] D. B. Leineweber, I. Bauer, H. G. Bock, and J. P. Schlöder. An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization – part I: theoretical aspects. *Comput. Chem. Engng*, 27:157–166, 2003.

[11] D. B. Leineweber, A. Schäfer, H. G. Bock, and J. P. Schlöder. An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization – part II: software aspects and applications. *Comput. Chem. Engng*, 27:167–174, 2003.

[12] T. McGeer. Passive dynamic walking. *International Journal of Robotics Research*, 9:62–82, 1990.

[13] T. McGeer. Passive dynamic biped catalogue. In R. Chatila and G. Hirzinger, editors, *Proceedings of the 2nd International Symposium of Experimental Robotics, Toulouse*. Springer-Verlag, New York, 1991.

[14] K. D. Mombaur. *Stability Optimization of Open-loop Controlled Walking Robots*. PhD thesis, University of Heidelberg, 2001. www.ub.uni-heidelberg.de/archiv/1796. VDI-Fortschrittbericht, Reihe 8, No. 922, ISBN 3-18-392208-8.

[15] K. D. Mombaur, H. G. Bock, J. P. Schlöder, and R. W. Longman. Open-loop stable solution of periodic optimal control problems in robotics. *ZAMM – Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 85(7):499–515, July 2005.

[16] K. D. Mombaur, H. G. Bock, J. P. Schlöder, and R. W. Longman. Self-stabilizing somersaults. *IEEE Transactions on Robotics*, 21(6), Dec. 2005.

[17] K. D. Mombaur, R. W. Longman, H. G. Bock, and J. P. Schlöder. Open-loop stable running. *Robotica*, 23(01):21–33, January 2005.

[18] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[19] G. A. Pratt and M. M. Williamson. Series Elastic Actuators. In *Proceedings of IROS*, pp. 399–406, Pittsburgh, 1995.

[20] J. E. Pratt. *Exploiting Inherent Robustness and Natural Dynamics in the Control of Bipedal Walking Robots.* PhD thesis, Massachusetts Institute of Technology, 2000.

[21] R. P. Ringrose. Self-stabilizing running. Technical report, Massachusetts Institute of Technology, 1997.

[22] J. La Salle and S. Lefschetz. *Stability by Liapunov's Direct Method.* Academic Press, 1961.

[23] R. v. Schwerin, M. J. Winckler, and V. H. Schulz. Parameter estimation in discontinuous descriptor models. In D. Bestle and W. Schiehlen, editors, *IUTAM Symposium on Optimization of Mechanical Systems*, pp. 269–276. Kluwer Academic Publishers, 1996.

[24] T. E. Wei, G. M. Nelson, R. D. Quinn, H. Verma, and S. L. Garverick. Design of a 5-cm monopod hopping robot. In *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 2828–2823, 2000. http://www.fluggart.-cwru.edu/icra2000/icra2000.html.

Achieving Bipedal Running with RABBIT: Six Steps Toward Infinity

B. Morris[1], E.R. Westervelt[2], C. Chevallereau[3], G. Buche[4],
and J.W. Grizzle[1]

[1] Control Systems Laboratory, EECS Department, University of Michigan, Ann Arbor, Michigan 48109-2122, USA
{MorrisBJ,Grizzle}@umich.edu

[2] Locomotion and Biomechanics Laboratory, Department of Mechanical Engineering, The Ohio State University, Columbus, Ohio 43210, USA
Westervelt.4@osu.edu

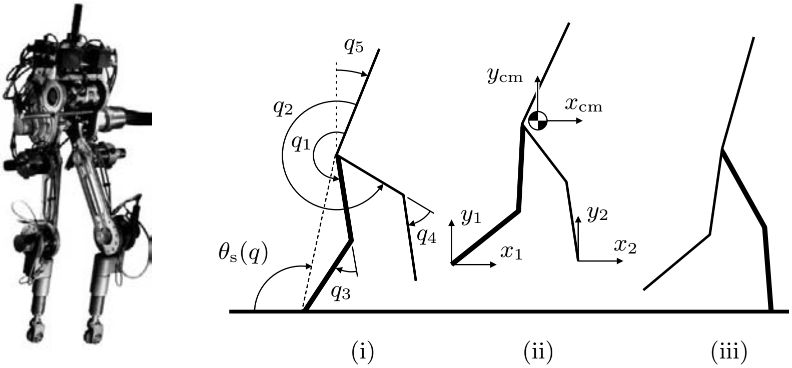
[3] IRCCyN, Ecole Centrale de Nantes, UMR CNRS 6597, BP 92101, 1 rue de la Noë, 44321 Nantes, cedex 03, France
Christine.Chevallereau@irccyn.ec-nantes.fr

[4] LAG, Ecole Nationale d'Ingénieurs Electriciens de Grenoble, BP 46, 38402 St Martin d'Hères, France
Gabriel.Buche@inpg.fr

Summary. This paper develops a class of bipedal running controllers based on the hybrid zero dynamics (HZD) framework and discusses related experiments conducted in September 2004 in Grenoble, France. In these experiments, RABBIT, a five-link, four-actuator, planar bipedal robot, executed six consecutive running steps. The observed gait was remarkably human-like, having long stride lengths (approx. 50 cm or 36% of body length), flight phases of significant duration (approx. 100 ms or 25% of step duration), an upright posture, and an average forward rate of 0.6 m/s. A video is available at [7, 17]. In the time allotted for experiments, stability of the gait could not be validated. To put the results into context, background information on hybrid robot modeling, control philosophy, and gait optimization techniques accompany final experimental observations. An additional discussion about some unmodeled dynamic and geometric effects that contributed to implementation difficulties is given.

1 Introduction

Designed and built as a platform to explore legged locomotion, RABBIT, a five-link, four-actuator, planar, bipedal robot [3, 2] (see Fig. 1(a)), has provided a means to test conceptually new approaches to underactuated, active, dynamic walking and running. Since March 2003, RABBIT has been used to experimentally verify a mathematical framework for the systematic design,



(a) RABBIT (b) Phases of running and coordinate conventions.

Fig. 1. RABBIT and the different phases of running with coordinate conventions labeled. In (b), the robot is shown (i) at the end of the stance phase; (ii) during flight; and (iii) at the beginning of the stance phase just after landing. To avoid clutter, the coordinate conventions have been spread out over the stance and flight phases. Angles are positive in the *counter clockwise* direction. The stance leg is indicated in bold

analysis, and optimization of controllers that induce stable walking gaits in N-link planar bipedal robots with one degree of underactuation [8, 20]. The next challenge for RABBIT is to achieve stable running.

In previous experiments, walking controllers for RABBIT acted by enforcing *virtual constraints*, which are holonomic constraints used to coordinate link movements throughout a gait. The stability of such walking motions were analyzed on the basis of the *hybrid zero dynamics* (HZD) *of walking*, with the conclusions of theory supported by experimental results [18]. Recent work in [5] extends the method of virtual constraints and the notion of an HZD to encompass the analysis of running in robots such as RABBIT. The developed control strategy is hybrid with both continuous and event-based actions and leads to the deliberate creation of an *HZD of running* and an accompanying stability test.

This paper summarizes a first attempt to use RABBIT to experimentally validate the theory of stable running presented in [5]. To facilitate implementation, the controller hypotheses of [5] are slightly relaxed, leading to controllers that are easier to design but which still lend the closed-loop system to a reduced-dimensionality stability test. Both discrete and continuous actions of the modified hybrid control law are discussed in detail. To put the results into context, background information on hybrid robot modeling, control philosophy, and gait optimization techniques accompany final experimental observations. An additional discussion about some unmodeled dynamic and geometric effects that contributed to implementation difficulties is given.

1.1 Related Work on Running Machines

Nearly twenty years ago, Raibert developed a series of 2D and 3D running machines that are more commonly called hoppers [16]. These hoppers employed a three part controller regulating hopping height, foot touchdown angle, and body attitude correction. The successful use of such a controller is tied to the morphology of the hopper: a massive body with significant inertia at the end of an actuated, compliant, prismatic leg.

Recently, there have been a number of successful demonstrations of running in robots with a different morphology: bipeds with revolute knees. A variety of control approaches were used. In late 2003, both Iguana Robotics and Sony announced (separate) experimental demonstrations of running in robots with revolute knees. Iguana Robotics' controller was based on central pattern generators (CPGs) and Sony's was based on the zero moment point (ZMP). In early 2004, running was announced for HRP-2LR [13] using a controller based on a technique of *resolved momentum.* In December 2004, Honda's robot, ASIMO, achieved running at 3 km/h (0.8 m/s) with a 50 ms flight phase using a controller based on *posture control.* A year later, ASIMO ran at a new top speed of 6 km/h.

1.2 Outline

The remainder of the paper is a self-contained description of the theoretical development and hardware modifications leading up to one example of an experiment in which RABBIT took six consecutive running steps.[5] Echoing [5, Sects. III and IV], Sect. 2 develops an open-loop model for RABBIT.[6] Section 3 discusses a modification of the control law given in [5] that is based on relaxed hypotheses. Philosophy and motivation of the modified control law are given in Sect. 3.1 with a detailed development of the hybrid controller in Sects. 3.2 to 3.6. The resulting closed-loop model of RABBIT and its stability properties are discussed in Sects. 3.7 and 3.8. Beyond this, Sect. 4 outlines a method for the design of stable gaits using constrained nonlinear optimization and includes a numerical example. Section 5 presents results from the first experimental implementation of running on RABBIT and a discussion outlining a number of possible reasons why stable running was not observed. Conclusions are drawn in Sect. 6.

[5] In a number of experiments, RABBIT achieved five or six running steps before tracking errors exceeded software bounds. One example is examined in detail.

[6] The content of Sect. 2 is based entirely on [5, Sects. III and IV] and is included for completeness.

2 Modeling

2.1 Assumptions and Terminology

RABBIT is modeled as a planar robot with five rigid massive links connected by (four) actuated, frictionless, revolute joints. The model is subdivided into two legs with identical physical properties, and a torso. The legs each consist of two links, a thigh and a shank, and are connected to each other and the single-link torso at the hips. Let $q_b := (q_1, q_2, q_3, q_4)'$ be the vector of actuated body coordinates, q_5 be the unactuated coordinate of the robot's absolute orientation, and x_{cm} and y_{cm} be the cartesian coordinates that give the horizontal and vertical positions of the robot's center of mass. See Fig. 1(b) for a depiction of the robot's morphology and coordinate convention.

The robot is said to be in *flight phase* when neither leg is in contact with the ground, and in *stance phase* when one leg is in stationary contact with the ground. During stance, the leg contacting the ground is called the *stance leg* and the other is called the *swing leg.* The transition from stance to flight is called *takeoff* and the transition from flight to stance is called *landing.* In this context, (steady-state) *running* is defined as a sequence of alternating stance and flight phases that is symmetric with respect to the left and right legs stride-to-stride.[7]

2.2 Dynamics of Flight and Stance

In the flight phase, the robot has 7 DOF with generalized coordinates $q_{\mathrm{f}} := (q_b', q_5, x_{\mathrm{cm}}, y_{\mathrm{cm}})'$. The equations of motion for this phase may be derived using the method of Lagrange and written in the following form:

$$D_{\mathrm{f}}(q_{\mathrm{b}})\ddot{q}_{\mathrm{f}} + C_{\mathrm{f}}(q_{\mathrm{b}}, \dot{q}_{\mathrm{f}})\dot{q}_{\mathrm{f}} + G_{\mathrm{f}}(q_{\mathrm{f}}) = B_{\mathrm{f}} u \ , \tag{1}$$

where D_{f} is the inertia matrix, the matrix C_{f} contains Coriolis and centrifugal terms, and G_{f} is the gravity vector. Introducing the state vector $x_{\mathrm{f}} := (q_{\mathrm{f}}', \dot{q}_{\mathrm{f}}')'$, the model (1) is expressed as

$$\dot{x}_{\mathrm{f}} = f_{\mathrm{f}}(x_{\mathrm{f}}) + g_{\mathrm{f}}(x_{\mathrm{f}})u \ . \tag{2}$$

The state space is taken as $\mathcal{X}_{\mathrm{f}} = T\mathcal{Q}_{\mathrm{f}} = \{x_{\mathrm{f}} = (q_{\mathrm{f}}', \dot{q}_{\mathrm{f}}')' \mid q_{\mathrm{f}} \in \mathcal{Q}_{\mathrm{f}},\ \dot{q}_{\mathrm{f}} \in I\!R^7\}$, where the configuration space $\mathcal{Q}_{\mathrm{f}}$ is a simply-connected, open subset of $I\!R^7$ corresponding to physically reasonable configurations of the robot.

In the stance phase, the stance leg end is fixed and, therefore, x_{cm} and y_{cm} are no longer independent coordinates. Accordingly, the robot in stance

[7] The chosen definition of running is fundamental to the following model and controller development. Other authors have defined running based on the motion of the center of mass or the reaction force profile on the stance leg, for example see [14].

phase has 5 DOF with generalized coordinates $q := (q_b', q_5)'$. Similar to the flight phase, the equations of motion may be written as

$$D_{\mathrm{s}}(q_{\mathrm{b}})\ddot{q} + C_{\mathrm{s}}(q_{\mathrm{b}}, \dot{q})\dot{q} + G_{\mathrm{s}}(q) = B_{\mathrm{s}} u \ . \tag{3}$$

Note that (3) may be obtained by subjecting (1) to the constraint that one leg end is in contact with the ground. Choosing the state vector $x_{\mathrm{s}} := (q', \dot{q}')'$, the model (3) may be expressed as

$$\dot{x}_{\mathrm{s}} = f_{\mathrm{s}}(x_{\mathrm{s}}) + g_{\mathrm{s}}(x_{\mathrm{s}})u \ . \tag{4}$$

The state space is taken as $\mathcal{X}_{\mathrm{s}} = T\mathcal{Q}_{\mathrm{s}} = \{x_{\mathrm{s}} = (q', \dot{q}')' \mid q \in \mathcal{Q}_{\mathrm{s}},\ \dot{q} \in \mathbb{R}^5\}$, where the configuration space $\mathcal{Q}_{\mathrm{s}}$ is a simply-connected, open subset of $\mathbb{R}^5$ corresponding to physically reasonable configurations of the robot.

2.3 Transitions

Landing, the transition from the flight phase to the stance phase, is modeled as a rigid impact. During this instantaneous event impulsive reaction forces from the ground bring the velocity of the tip of the advancing leg to zero without causing it to rebound or slip. In addition, at the moment of landing, the robot's configuration remains unchanged, but joint velocities change instantaneously [11]. The post-impact joint velocities[8] are given by a function [5, Eq. (21)],

$$\dot{q}^{+} = \tilde{\mathit{\Delta}}(q_{\mathrm{f}}^{-}, \dot{q}_{\mathrm{f}}^{-}) \ . \tag{5}$$

Since the gait is assumed to be symmetric from stride to stride (with respect to the left and right legs) a state relabeling matrix R is used to swap leg definitions (redefine the coordinates) at landing. The flight-to-stance transition operator, including state relabeling, is therefore defined as

$$x_{\mathrm{s}}^{+} = \mathit{\Delta}_{\mathrm{f}}^{\mathrm{s}}(x_{\mathrm{f}}^{-}) := \begin{bmatrix} R\,q^{-} \\ R\tilde{\mathit{\Delta}}(q_{\mathrm{f}}^{-}, \dot{q}_{\mathrm{f}}^{-}) \end{bmatrix} \ . \tag{6}$$

This transition operator is applied when the end of the advancing leg touches the ground, that is, when $y_2 = 0$ (see Fig. 1(b)). Define the function, $H_{\mathrm{f}}^{\mathrm{s}} : T\mathcal{Q}_{\mathrm{f}} \to \mathbb{R}$ by $H_{\mathrm{f}}^{\mathrm{s}}(x_{\mathrm{f}}) = y_2$, so that $H_{\mathrm{f}}^{\mathrm{s}}(x_{\mathrm{f}}) = 0$ characterizes the transition hypersurface surface $\mathcal{S}_{\mathrm{f}}^{\mathrm{s}}$ within $T\mathcal{Q}_{\mathrm{f}}$.

The transition from stance to flight is also modeled as an instantaneous event, but one on which positions and velocities are unchanged,

[8] The terms $x_{\mathrm{f}}^{-} := (q_{\mathrm{f}}^{-'}, \dot{q}_{\mathrm{f}}^{-'})'$ and $x_{\mathrm{s}}^{+} := (q^{+'}, \dot{q}^{+'})'$ refer to the system state just before and just after the landing event. The terms $x_{\mathrm{s}}^{-} := (q^{-'}, \dot{q}^{-'})'$ and $x_{\mathrm{f}}^{+} := (q_{\mathrm{f}}^{+'}, \dot{q}_{\mathrm{f}}^{+'})'$ refer to the system state just before and just after the takeoff event. The addition of the superscript "*" (such as x_{f}^{+*}) indicates reference to the value *at steady-state*, i.e., on the periodic orbit.

$$x_{\mathrm{f}}^{+} = \Delta_{\mathrm{s}}^{\mathrm{f}}(x_{\mathrm{s}}^{-}) := \begin{bmatrix} \begin{bmatrix} q^{-} \\ f_{\mathrm{cm}}(q^{-}) \end{bmatrix} \\ \begin{bmatrix} \dot{q}^{-} \\ \frac{\partial}{\partial q} f_{\mathrm{cm}}(q^{-})\, \dot{q}^{-} \end{bmatrix} \end{bmatrix} . \tag{7}$$

where $f_{\mathrm{cm}}(q) := (x_{\mathrm{cm}}(q), y_{\mathrm{cm}}(q))'$ gives the location of the center of mass. The transition from stance into flight is treated as a control decision because the flight phase is initiated (at will) by accelerating the stance leg off the ground. The transition hypersurface, $\mathcal{S}_{\mathrm{s}}^{\mathrm{f}}$, is a level set of a function $H_{\mathrm{s}}^{\mathrm{f}}(x_{\mathrm{s}}) : T\mathcal{Q}_{\mathrm{s}} \to I\!R$ that is chosen by the control designer.

2.4 A Hybrid, Open-Loop Model of Running

The stance and flight dynamic models may be represented compactly, along with their transition models, as a discrete-event system with two charts (terminology taken from [9]). This open-loop hybrid model is specified by charts Σ_{f} and Σ_{s} where for $(i, j) \in \{(\mathrm{f}, \mathrm{s}), (\mathrm{s}, \mathrm{f})\}$, $\Sigma_i = \{\mathcal{X}_i, \mathcal{F}_i, \mathcal{S}_i^j, \mathcal{T}_i^j\}$,

1. $\mathcal{X}_i$ is a *state manifold*, which is 10 dimensional in stance and 14 dimensional in flight;
2. $\mathcal{F}_i$ is a *flow on the state manifold*, a differential equation describing the in-phase motion on $\mathcal{X}_i$;
3. $\mathcal{S}_i^j$ is a *switching hypersurface*, a hypersurface of $\mathcal{X}_i$ corresponding to a transition from one state manifold to another; and
4. $\mathcal{T}_i^j$ is a *transition map* giving initial conditions for the next continuous phase.

In this notation, the open-loop hybrid model is

$$\Sigma_{\mathrm{f}} : \begin{cases} \mathcal{X}_{\mathrm{f}} = T\mathcal{Q}_{\mathrm{f}} \\ \mathcal{F}_{\mathrm{f}} : (\dot{x}_{\mathrm{f}}) = f_{\mathrm{f}}(x_{\mathrm{f}}) + g_{\mathrm{f}}(x_{\mathrm{f}})u \\ \mathcal{S}_{\mathrm{f}}^{\mathrm{s}} = \{x_{\mathrm{f}} \in T\mathcal{Q}_{\mathrm{f}} \mid H_{\mathrm{f}}^{\mathrm{s}}(x_{\mathrm{f}}) = 0\} \\ \mathcal{T}_{\mathrm{f}}^{\mathrm{s}} : x_{\mathrm{s}}^{+} = \Delta_{\mathrm{f}}^{\mathrm{s}}(x_{\mathrm{f}}^{-}) \end{cases} \tag{8a}$$

$$\Sigma_{\mathrm{s}} : \begin{cases} \mathcal{X}_{\mathrm{s}} = T\mathcal{Q}_{\mathrm{s}} \\ \mathcal{F}_{\mathrm{s}} : (\dot{x}_{\mathrm{s}}) = f_{\mathrm{s}}(x_{\mathrm{s}}) + g_{\mathrm{s}}(x_{\mathrm{s}})u \\ \mathcal{S}_{\mathrm{s}}^{\mathrm{f}} = \{x_{\mathrm{s}} \in T\mathcal{Q}_{\mathrm{s}} \mid H_{\mathrm{s}}^{\mathrm{f}}(x_{\mathrm{s}}) = 0\} \\ \mathcal{T}_{\mathrm{s}}^{\mathrm{f}} : x_{\mathrm{f}}^{+} = \Delta_{\mathrm{s}}^{\mathrm{f}}(x_{\mathrm{s}}^{-}). \end{cases} \tag{8b}$$

3 Control Methodology

3.1 Summary and Philosophy

The overall philosophy of HZD control is to use the freedom available in feedback design to achieve a parameterized family of closed-loop systems whose stability analysis is analytically tractable. This allows the use of numerical optimization to search among the family of closed-loop systems to find those that yield a desired behavior, such as stable running at a pre-determined speed with upper bounds on peak actuator power and the coefficient of static friction between the leg end and the ground.

Parameterization is achieved through the use of virtual constraints in both the stance and flight phases. Perfect enforcement of virtual constraints results in low-dimensional surfaces that are invariant under the differential equations of the closed-loop model[9] and are also invariant under the transition maps.[10] To achieve invariance at landing, a deadbeat action is incorporated in the flight phase controller that steers the robot to land in a pre-determined configuration, while respecting conservation of angular momentum about the robot's center of mass. This hybrid controller creates a one DOF HZD that allows the stability of a running motion to be analyzed in closed form on the basis of a one-dimensional Poincaré map.

In the first running experiment attempted on RABBIT, there was not sufficient time[11] to implement completely the controller of [5]. The controller that was implemented used virtual constraints in both the stance and flight phases, but the deadbeat action of the flight phase controller was not implemented to regulate the final configuration of the robot at touchdown. Instead, to account for the changing configuration of the robot at touchdown, the transition controller of [19] was adopted[12]. Key points of the related analysis are highlighted in Sect. 3.8.

3.2 Preliminaries on Virtual Constraints

Since RABBIT has four independent actuators (two at the hips and two at the knees), four virtual constraints may be imposed in both the stance and

[9] "Invariant" in this sense means that if the differential equation is initialized on the constraint surface, then its solution remains on the constraint surface until a transition occurs.

[10] "Invariant" in this sense means that if the solution is on the flight phase (resp. stance phase) constraint surface at touchdown (resp. takeoff), then after transition the solution will be contained in the stance phase (resp. flight phase) constraint surface.

[11] A total of two weeks were available to perform the experiments.

[12] The transition controller of this paper takes into account the joint angles of the robot at touchdown but not the joint angular velocities. As a result a true HZD of running is not created, and the resulting analysis of Sect. 3.8 (based on [8]) is modified accordingly.

flight phases. To define them, consider a function pair $\{\theta(q),\ h_d(\theta)\}$, where $\theta : \mathcal{Q} \to I\!R$ is a scalar function of the configuration variables, and $h_d : I\!R \to I\!R^4$ is a function giving the desired configuration of the actuated joints as a function of θ. The virtual constraints are expressed as outputs of (8),

$$y = h(q) := q_b - h_d \circ \theta(q)\ , \tag{9}$$

which are zeroed by the action of a state feedback controller. The design of such a controller is a well-understood, standard problem of nonlinear control [12].

For purposes of design, the virtual constraints are parameterized [5]. For notational convenience, the stance phase and flight phase virtual constraints will be parameterized separately by a_{s} and a_{f}, respectively. These parameter sets, which lie in the parameter spaces $\mathcal{A}_{\mathrm{s}} := I\!R^{n_{\mathrm{s}}}$ and $\mathcal{A}_{\mathrm{f}} := I\!R^{n_{\mathrm{f}}}$, may be updated at takeoff and landing events but are otherwise constant. With this notation, the virtual constraints for stance and flight are, respectively,[13]

$$y = q_b - h_{d,\mathrm{s}}[a_{\mathrm{s}}](\theta_{\mathrm{s}}(q)) \tag{10a}$$

$$y = q_b - h_{d,\mathrm{f}}[a_{\mathrm{f}}](\theta_{\mathrm{f}}[a_{\mathrm{f}}](q_{\mathrm{f}}))\ . \tag{10b}$$

3.3 Stance Phase Control

The controller for the stance phase acts by updating the parameters a_{s} and by enforcing the virtual constraints (10a). As a result of enforcing the virtual constraints, in stance phase, the robot behaves as an unactuated 1 DOF system whose properties may be tuned by choosing different constraint parameters. Apart from different boundary conditions on the virtual constraints, this control is identical to the walking controllers developed in [18, 20]. The stance phase parameter vector, a_{s}, may be expressed as

$$a_{\mathrm{s}} := (a'_{\mathrm{s},0},\ a'_{\mathrm{s},1}, \ldots,\ a'_{\mathrm{s},m_{\mathrm{s}}-1},\ a'_{\mathrm{s},m_{\mathrm{s}}},\ \theta_{\mathrm{s}}^-,\ \theta_{\mathrm{s}}^+)'\ , \tag{11}$$

where $m_{\mathrm{s}} \geq 3$, $a_{\mathrm{s},i} \in I\!R^4$ for $i \in \{0, 1, \ldots, m_{\mathrm{s}} - 1, m_{\mathrm{s}}\}$, and $\theta_{\mathrm{s}}^-, \theta_{\mathrm{s}}^+ \in I\!R$. Note that $n_{\mathrm{s}} = 4\,(m_{\mathrm{s}} + 1) + 2$. The terms θ_{s}^- and θ_{s}^+ are the values of the function $\theta_{\mathrm{s}}(q)$ evaluated at the end and the beginning of the stance phase. In [18, 20], h_d is expressed in terms of Bézier polynomials. Here, a slightly different class of polynomials[14] is used that satisfy the following:

$$\begin{aligned} h_{d,\mathrm{s}}[a_{\mathrm{s}}](\theta_{\mathrm{s}}^+) &= a_{\mathrm{s},0} & \tfrac{d}{d\theta_{\mathrm{s}}} h_{d,\mathrm{s}}[a_{\mathrm{s}}](\theta_{\mathrm{s}}^-) &= a_{\mathrm{s},m_{\mathrm{s}}-1} \\ \tfrac{d}{d\theta_{\mathrm{s}}} h_{d,\mathrm{s}}[a_{\mathrm{s}}](\theta_{\mathrm{s}}^+) &= a_{\mathrm{s},1} & h_{d,\mathrm{s}}[a_{\mathrm{s}}](\theta_{\mathrm{s}}^-) &= a_{\mathrm{s},m_{\mathrm{s}}}\ . \end{aligned} \tag{12}$$

[13] Terms that are constant during the continuous phases of motion, and potentially updated at phase transitions, will be considered *parameters* and enclosed in square brackets.

[14] Any class of smooth functions satisfying these properties may be used to define virtual constraints.

The stance phase virtual constraints are imposed on the dynamics by using a control $u_{\mathrm{s}} : \mathcal{X}_{\mathrm{s}} \times \mathcal{A}_{\mathrm{s}} \to I\!R^4$ that drives (10a) to zero *in finite time*. The specific assumptions are as in [8, 20].

3.4 Flight Phase Control

The development of the flight phase controller is similar to that of the stance phase controller. The key difference is the choice of θ_{f} in (10b) to be a function of the position of the center of mass. The flight phase parameter vector, a_{f}, is defined as

$$a_{\mathrm{f}} := (a'_{\mathrm{f},0},\ a'_{\mathrm{f},1}, \dots,\ a'_{\mathrm{f},m_{\mathrm{f}}-1},\ a'_{\mathrm{f},m_{\mathrm{f}}},\ x^+_{\mathrm{cm,f}},\ \dot{x}^+_{\mathrm{cm,f}},\ T_{\mathrm{f}})' \ , \tag{13}$$

where $m_{\mathrm{f}} \geq 3$, $a_{\mathrm{f},i} \in I\!R^4$ for $i \in \{0, 1, \dots, m_{\mathrm{f}}-1, m_{\mathrm{f}}\}$, and $x^+_{\mathrm{cm,f}}, \dot{x}^+_{\mathrm{cm,f}}, T_{\mathrm{f}} \in I\!R$. Note that $n_{\mathrm{f}} = 4\,(m_{\mathrm{f}}+1)+3$. The terms $x^+_{\mathrm{cm,f}}$, $\dot{x}^+_{\mathrm{cm,f}}$, and T_{f} are, respectively, the horizontal position of the center of mass at the beginning of the flight phase, the horizontal velocity of the center of mass at the beginning of the flight phase, and the estimated[15] duration of the flight phase. The flight phase virtual constraints (10b) are given by

$$\theta_{\mathrm{f}}[a_{\mathrm{f}}](q_{\mathrm{f}}) := \frac{1}{T_{\mathrm{f}}} \left(\frac{x_{\mathrm{cm}} - x^+_{\mathrm{cm,f}}}{\dot{x}^+_{\mathrm{cm,f}}} \right) \ , \tag{14}$$

and $h_{d,\mathrm{f}}[a_{\mathrm{f}}]$, which, as in the stance phase, is a smooth, vector-valued function that satisfies

$$\begin{aligned} h_{d,\mathrm{f}}[a_{\mathrm{f}}](0) &= a_{\mathrm{f},0} & \tfrac{d}{d\theta_{\mathrm{f}}} h_{d,\mathrm{f}}[a_{\mathrm{f}}](1) &= a_{\mathrm{f},m_{\mathrm{f}}-1} \\ \tfrac{d}{d\theta_{\mathrm{f}}} h_{d,\mathrm{f}}[a_{\mathrm{f}}](0) &= a_{\mathrm{f},1} & h_{d,\mathrm{f}}[a_{\mathrm{f}}](1) &= a_{\mathrm{f},m_{\mathrm{f}}} \ . \end{aligned} \tag{15}$$

For a given stride, let t_{f} denote the elapsed time within the flight phase. By conservation of linear momentum, $\dot{x}^+_{\mathrm{cm,f}}$ is constant during flight, which implies $t_{\mathrm{f}} = (x_{\mathrm{cm}} - x^+_{\mathrm{cm,f}})/\dot{x}^+_{\mathrm{cm,f}}$. As a result, $\theta_{\mathrm{f}} = t_{\mathrm{f}}/T_{\mathrm{f}}$ is a valid substitute for (14), and for this reason, the given flight phase virtual constraints are said to be *time scaled*. Flight phase virtual constraints are enforced using any smooth state feedback controller $u_{\mathrm{f}} : \mathcal{X}_{\mathrm{f}} \times \mathcal{A}_{\mathrm{f}} \to I\!R^4$ that drives (10b) to zero exponentially quickly.

Note that finite-time convergence is not used in the flight phase. A finite-time controller is used in the stance phase to render the stance phase constraint surface finite-time attractive so that the analysis of running will be similar to that of walking [8]. For further discussion of this point, refer to Sect. 3.8.

[15] Calculation of T_{f} requires the height of the center of mass at landing, $y^-_{\mathrm{cm,f}}$, to be known *a priori*, which is only possible if the virtual constraints are exactly enforced throughout the flight phase.

3.5 Transition Control: Landing

In the event that landing occurs with the state of the robot not satisfying the virtual constraints, the control parameters of the subsequent stance phase, a_{s}, are updated to ensure that the configuration of the robot satisfies $q_b - h_{d,\mathrm{s}}[a_{\mathrm{s}}](\theta_{\mathrm{s}}^{+}) = 0$. The parameter updates are governed by the differentiable function $w_{\mathrm{f}}^{\mathrm{s}} : \mathcal{S}_{\mathrm{f}}^{\mathrm{s}} \to \mathcal{A}_{\mathrm{s}}$, such that for $a_{\mathrm{s}} = w_{\mathrm{f}}^{\mathrm{s}}(x_{\mathrm{f}}^{-})$,

$$\begin{aligned} a_{\mathrm{s},0} &= q_b^{+} \\ a_{\mathrm{s},1} &= a_{\mathrm{s},1}^{*} \\ &\vdots \\ a_{\mathrm{s},m_{\mathrm{s}}-1} &= a_{\mathrm{s},m_{\mathrm{s}}-1}^{*} \\ a_{\mathrm{s},m_{\mathrm{s}}} &= a_{\mathrm{s},m_{\mathrm{s}}}^{*}\ . \end{aligned} \qquad \begin{aligned} \theta_{\mathrm{s}}^{+} &= \theta_{\mathrm{s}}(q^{+}) \\ \theta_{\mathrm{s}}^{-} &= \theta_{\mathrm{s}}^{-*} \end{aligned} \tag{16}$$

In the above, q^{+} is calculated using $\Delta_{\mathrm{f}}^{\mathrm{s}}(x_{\mathrm{f}}^{-})$, and the terms θ_{s}^{-*} and $a_{\mathrm{s},i}^{*} \in I\!R^4$, $i \in \{1, \ldots, m_{\mathrm{s}} - 1, m_{\mathrm{s}}\}$ are constant parameters chosen during design.

If the stance phase finite-time controller can satisfy the virtual constraints (10a) before the liftoff event occurs, and the parameter updates obey (16), then the stance phase will terminate with $q_b - h_{d,\mathrm{s}}[a_{\mathrm{s}}](\theta_{\mathrm{s}}^{-}) = 0$, or equivalently, with $q^{-} = q^{-*}$.

3.6 Transition Control: Takeoff

At takeoff, the parameters of the flight phase virtual constraints, a_{f}, are updated so that the duration of the planned motion of the robot is equal to the estimated flight time. Parameter updates are governed by a continuously differentiable function $w_{\mathrm{s}}^{\mathrm{f}} : \mathcal{S}_{\mathrm{s}}^{\mathrm{f}} \to \mathcal{A}_{\mathrm{f}}$, such that for $a_{\mathrm{f}} = w_{\mathrm{s}}^{\mathrm{f}}(x_{\mathrm{s}}^{-})$,

$$\begin{aligned} a_{\mathrm{f},0} &= a_{\mathrm{f},0}^{*} \\ a_{\mathrm{f},1} &= a_{\mathrm{f},1}^{*} \\ &\vdots \\ a_{\mathrm{f},m_{\mathrm{f}}-1} &= a_{\mathrm{f},m_{\mathrm{f}}-1}^{*} \\ a_{\mathrm{f},m_{\mathrm{f}}} &= a_{\mathrm{f},m_{\mathrm{f}}}^{*} \end{aligned} \qquad \begin{aligned} x_{\mathrm{cm,f}}^{+} &= (f_{\mathrm{cm}}(q^{-}))_1 \\ \dot{x}_{\mathrm{cm,f}}^{+} &= \left(\frac{\partial f_{\mathrm{cm}}}{\partial q}(q^{-})\,\dot{q}^{-}\right)_1 \end{aligned} \tag{17}$$

$$T_{\mathrm{f}} = \frac{\dot{y}_{\mathrm{cm,f}}^{+}}{g} + \frac{\sqrt{(\dot{y}_{\mathrm{cm,f}}^{+})^2 - 2g(y_{\mathrm{cm,f}}^{-*} - y_{\mathrm{cm,f}}^{+})}}{g}\ .$$

where g is the magnitude of the acceleration of gravity and $y_{\mathrm{cm,f}}^{-*}$ is the height of the center of mass at the end of the flight phase, on the limit cycle. The terms $a_{\mathrm{f},i}^{*} \in I\!R^4, i \in \{0, 1, \ldots, m_{\mathrm{f}} - 1, m_{\mathrm{f}}\}$ are parameters chosen during design. Initiation of the takeoff event is a control decision, designated to occur when $\theta_{\mathrm{s}}(q) = \theta_{\mathrm{s}}^{-}$. In the closed-loop model the switching hypersurface is $\mathcal{S}_{\mathrm{s}}^{\mathrm{f}} = \{(x_{\mathrm{s}}, a_{\mathrm{s}}) \in \mathcal{X}_{\mathrm{s}} \times \mathcal{A}_{\mathrm{s}} \mid H_{\mathrm{s}}^{\mathrm{f}}(x_{\mathrm{s}}, a_{\mathrm{s}}) = 0\}$ where $H_{\mathrm{s}}^{\mathrm{f}}(x_{\mathrm{s}}, a_{\mathrm{s}}) := \theta_{\mathrm{s}}(q) - \theta_{\mathrm{s}}^{-}$.

3.7 Resulting Closed-Loop Model of Running

To form the closed-loop model of running, the state space of the open-loop model, (8), is enlarged to include the parameters of the flight and stance phases. Define the augmented state spaces $\bar{\mathcal{X}}_{\mathrm{f}} := T\mathcal{Q}_{\mathrm{f}} \times \mathcal{A}_{\mathrm{f}}$ and $\bar{\mathcal{X}}_{\mathrm{s}} := T\mathcal{Q}_{\mathrm{s}} \times \mathcal{A}_{\mathrm{s}}$ with elements given by $\bar{x}_{\mathrm{f}} := (q_{\mathrm{f}}', \dot{q}_{\mathrm{f}}', a_{\mathrm{f}}')'$ and $\bar{x}_{\mathrm{s}} := (q', \dot{q}', a_{\mathrm{s}}')'$. The closed-loop dynamics may then be written as

$$\bar{f}_{\mathrm{f}}(\bar{x}_{\mathrm{f}}) := \begin{bmatrix} f_{\mathrm{f}}(x_{\mathrm{f}}) + g_{\mathrm{f}}(x_{\mathrm{f}}) u_{\mathrm{f}}(x_{\mathrm{f}}, a_{\mathrm{f}}) \\ 0_{n_{\mathrm{f}} \times 1} \end{bmatrix} \tag{18a}$$

$$\bar{f}_{\mathrm{s}}(\bar{x}_{\mathrm{s}}) := \begin{bmatrix} f_{\mathrm{s}}(x_{\mathrm{s}}) + g_{\mathrm{s}}(x_{\mathrm{s}}) u_{\mathrm{s}}(x_{\mathrm{s}}, a_{\mathrm{s}}) \\ 0_{n_{\mathrm{s}} \times 1} \end{bmatrix} . \tag{18b}$$

The zero vectors reflect that the virtual constraint parameters do not change during the continuous phases of running. The impact maps, in which the parameters are updated, are modified to include the parameter update laws, $w_{\mathrm{f}}^{\mathrm{s}}$ and $w_{\mathrm{s}}^{\mathrm{f}}$:

$$\bar{\Delta}_{\mathrm{f}}^{\mathrm{s}}(\bar{x}_{\mathrm{f}}^-) := \begin{bmatrix} \Delta_{\mathrm{f}}^{\mathrm{s}}(x_{\mathrm{f}}^-) \\ w_{\mathrm{f}}^{\mathrm{s}}(x_{\mathrm{f}}^-) \end{bmatrix} \tag{19a}$$

$$\bar{\Delta}_{\mathrm{s}}^{\mathrm{f}}(\bar{x}_{\mathrm{s}}^-) := \begin{bmatrix} \Delta_{\mathrm{s}}^{\mathrm{f}}(x_{\mathrm{s}}^-) \\ w_{\mathrm{s}}^{\mathrm{f}}(x_{\mathrm{s}}^-) \end{bmatrix} . \tag{19b}$$

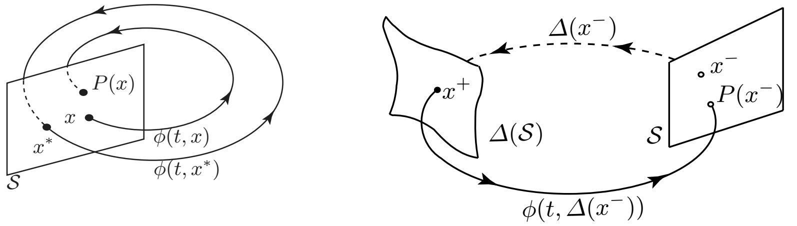
The closed-loop hybrid model is then

$$\Sigma_{\mathrm{cl,f}} : \begin{cases} \bar{\mathcal{X}}_{\mathrm{f}} = T\mathcal{Q}_{\mathrm{f}} \times \mathcal{A}_{\mathrm{f}} \\ \bar{\mathcal{F}}_{\mathrm{f}} : (\dot{\bar{x}}_{\mathrm{f}}) = \bar{f}_{\mathrm{f}}(\bar{x}_{\mathrm{f}}) \\ \bar{S}_{\mathrm{f}}^{\mathrm{s}} = \{(x_{\mathrm{f}}, a_{\mathrm{f}}) \in \bar{\mathcal{X}}_{\mathrm{f}} \mid H_{\mathrm{f}}^{\mathrm{s}}(x_{\mathrm{f}}) = 0\} \\ \bar{\mathcal{T}}_{\mathrm{f}}^{\mathrm{s}} : \bar{x}_{\mathrm{s}}^+ = \bar{\Delta}_{\mathrm{f}}^{\mathrm{s}}(\bar{x}_{\mathrm{f}}^-) \end{cases} \tag{20a}$$

$$\Sigma_{\mathrm{cl,s}} : \begin{cases} \bar{\mathcal{X}}_{\mathrm{s}} = T\mathcal{Q}_{\mathrm{s}} \times \mathcal{A}_{\mathrm{s}} \\ \bar{\mathcal{F}}_{\mathrm{s}} : (\dot{\bar{x}}_{\mathrm{s}}) = \bar{f}_{\mathrm{s}}(\bar{x}_{\mathrm{s}}) \\ \bar{S}_{\mathrm{s}}^{\mathrm{f}} = \{(x_{\mathrm{s}}, a_{\mathrm{s}}) \in \bar{\mathcal{X}}_{\mathrm{s}} \mid H_{\mathrm{s}}^{\mathrm{f}}(x_{\mathrm{s}}, a_{\mathrm{s}}) = 0\} \\ \bar{\mathcal{T}}_{\mathrm{s}}^{\mathrm{f}} : \bar{x}_{\mathrm{f}}^+ = \bar{\Delta}_{\mathrm{s}}^{\mathrm{f}}(\bar{x}_{\mathrm{s}}^-). \end{cases} \tag{20b}$$

3.8 Existence and Stability of Periodic Orbits

The Poincaré return map is a well-known tool for determining the existence of periodic orbits and their stability properties (see Fig. 2). For its use in hybrid systems, see for example [6, 8, 10, 15]. Its application to periodic orbits of (20) can be carried out using the results in [8] and a construction presented in



(a) Poincaré return map for ODE systems

(b) Poincaré return map for systems with impulse effects

Fig. 2. Illustration (**a**) shows the geometric interpretation of a Poincaré return map $P : \mathcal{S} \to \mathcal{S}$ for an ordinary differential equation (non-hybrid) system as event-based sampling of the solution near a periodic orbit. The Poincaré section, $\mathcal{S}$, may be any co-dimension one (hyper) C^1-surface that is transversal to the periodic orbit. Illustration (**b**) shows the geometric interpretation of a Poincaré return map for a system with impulse effects. A periodic orbit exists when $P(x^-) = x^-$

[5, Thm. 1]. Since thorough development of these ideas would consume more space than is available, the main ideas are only sketched.

The first step is to construct a system with impulse effects (that is, a single-chart hybrid model) that has the same Poincaré map as (20). Following [5, Eq. (62)], define

$$\Sigma_{\mathrm{cl}} : \begin{cases} \dot{\bar{x}}(t) = \bar{f}_{\mathrm{s}}(\bar{x}(t)) & x^-(t) \notin \bar{S} \\ \bar{x}^+(t) = \bar{\Delta}(\bar{x}^-(t)) & \bar{x}^-(t) \in \bar{S}, \end{cases} \tag{21}$$

where $\bar{S} := \bar{S}_{\mathrm{s}}^{\mathrm{f}}$, $\Delta := \bar{\Delta}_{\mathrm{f}}^{\mathrm{s}} \circ P_{\mathrm{f}}$, and P_{f} is the flow of the closed-loop flight phase model (see [5, Eq. (59)]). In words, this system consists of the differential equation of the closed-loop stance phase model of (20) and a generalized impact map $\bar{\Delta}$ that includes the transition map from stance to flight, the flight phase dynamics, and the impact map from flight to stance. The generalized impact map is the result of event-based sampling of the solution of (20) at takeoff events.

Because the virtual constraints in the stance phase are achieved with a continuous finite-time controller [1], the reduction technique of [8, Thm. 2] is applicable. Because the parameter updates in the stance phase can be computed in terms of the state of the robot at takeoff, the analysis of periodic orbits can be reduced to the computation of a one-dimensional restricted Poincaré map, ρ, having $\mathcal{S}_{\mathrm{s}}^{\mathrm{f}}$ as a Poincaré section.

4 Design of Running Motions with Optimization

4.1 Optimization Parameters

To design a running motion, a numerical routine is used to search the parameter spaces $\mathcal{A}_{\mathrm{s}}$ and $\mathcal{A}_{\mathrm{f}}$ for a set of parameters that results in a desirable gait (periodic orbit of (20)). Common requirements on the gait are achieved by incorporating constraints into the numerical search. Such constraints address actuator limits, allowable joint space, and unilateral ground-contact forces. The constraints also ensure steady-state running at a certain speed and overall efficiency of the gait. For the experiments reported here, the gait was designed using an optimization approach that combined the ideas of [4] and [20]; the optimization was performed directly on the parameters of the virtual constraints in order to *simultaneously* determine a periodic running motion and a controller that achieves it. This is in contrast with the approach of [5] where virtual constraints are designed by regression against optimal, pre-computed, periodic trajectories.

Virtual constraints are assumed to be identically satisfied on the periodic orbit, which has two consequences: first, the integration of the closed-loop system dynamics can be performed using the stance and flight phase zero dynamics (see [5] for details), resulting in short computation times; and second, the virtual constraint parameters, a_{s} and a_{f}, are not independent. Once the independent parameters have been identified, standard numerical optimization routines may be used to search for desirable gaits. The implementation of such a procedure is outlined in the following subsections.

4.2 Boundary Conditions of the Virtual Constraints

The transition maps of takeoff and landing can be used to identify redundancies between the virtual constraint parameter vectors a_{s} and a_{f}. Given the state corresponding to the end of the limit-cycle stance phase, $x_{\mathrm{s}}^{-*} = (q^{-*}, \dot{q}^{-*})$, the state at the beginning of the subsequent flight phase may be computed as $x_{\mathrm{f}}^{+*} = (q_{\mathrm{f}}^{+*}, \dot{q}_{\mathrm{f}}^{+*}) = \mathit{\Delta}_{\mathrm{s}}^{\mathrm{f}}(x_{\mathrm{s}}^{-*})$. For both x_{s}^{-*} and x_{f}^{+*} to satisfy the virtual constraints of their respective phases, the following relations must hold,

$$\begin{aligned} a^*_{\mathrm{s},m_{\mathrm{s}}-1} &= \dot{q}^{-*}_{b,\mathrm{s}}/\dot{\theta}^{-*}_{\mathrm{s}} & \quad a^*_{\mathrm{f},0} &= q^{+*}_{b,\mathrm{f}} \\ a^*_{\mathrm{s},m_{\mathrm{s}}} &= q^{-*}_{b,\mathrm{s}} & \quad a^*_{\mathrm{f},1} &= \dot{q}^{+*}_{b,\mathrm{f}}\, T^*_{\mathrm{f}} \ , \end{aligned} \tag{22}$$

which are derived by applying (12), (14), (15), and (17) to (10). These are the boundary conditions associated with the liftoff event of the periodic orbit. The state of the robot at the beginning of the stance phase, $x_{\mathrm{s}}^{+*} = (q^{+*}, \dot{q}^{+*})$, can be related to the state at the end of the previous flight phase, $x_{\mathrm{f}}^{-*} = (q_{\mathrm{f}}^{-*}, \dot{q}_{\mathrm{f}}^{-*})$, by the landing map, $x_{\mathrm{s}}^{+*} = \mathit{\Delta}_{\mathrm{f}}^{\mathrm{s}}(x_{\mathrm{f}}^{-*})$, to yield the following additional design constraints,

Table 1. Independent and dependent terms used in optimization. The choice of the independent terms is non-unique and depends on the specific optimization procedure. The parameters below correspond to the algorithm in Sect. 4.3, which is one straightforward method to ensure the boundary conditions of the virtual constraints are met

Terms of Optimization	
Independent	Dependent
$x_{\mathrm{f}}^{-*} \in \mathbb{R}^{14}$	$\theta_{\mathrm{s}}^{+*} \in \mathbb{R}$
$a^*_{\mathrm{s},2}, \ldots, a^*_{\mathrm{s},m_{\mathrm{s}}} \in \mathbb{R}^4$	$a^*_{\mathrm{s},0},\, a^*_{\mathrm{s},1} \in \mathbb{R}^4$
$\theta_{\mathrm{s}}^{-*} \in \mathbb{R}$	$x_{\mathrm{s}}^{+*} \in \mathbb{R}^{10}$
$a^*_{\mathrm{f},2}, \ldots, a^*_{\mathrm{f},m_{\mathrm{f}}-2} \in \mathbb{R}^4$	$a^*_{\mathrm{f},0},\, a^*_{\mathrm{f},1} \in \mathbb{R}^4$
	$a^*_{\mathrm{f},m_{\mathrm{f}}-1},\, a^*_{\mathrm{f},m_{\mathrm{f}}} \in \mathbb{R}^4$
	$x^{+*}_{\mathrm{cm,f}},\, \dot{x}^{+*}_{\mathrm{cm,f}},\, T^*_{\mathrm{f}} \in \mathbb{R}$
	$x_{\mathrm{f}}^{-} \in \mathbb{R}^{14}$

$$\begin{aligned} a^*_{\mathrm{s},0} &= q^{+*}_{b,\mathrm{s}} & \qquad a^*_{\mathrm{f},m_{\mathrm{f}}-1} &= \dot{q}^{-*}_{b,\mathrm{f}}\, T^*_{\mathrm{f}} \\ a^*_{\mathrm{s},1} &= \dot{q}^{+*}_{b,\mathrm{s}} / \dot{\theta}^{+*}_{\mathrm{s}} & \qquad a^*_{\mathrm{f},m_{\mathrm{f}}} &= q^{-*}_{b,\mathrm{f}} \quad . \end{aligned} \tag{23}$$

The update law presented here enforces fewer boundary conditions than the update law of [5]. The extra boundary conditions associated with takeoff are already satisfied by (22), but those of landing are not met by (23); they are more difficult to satisfy due to conservation of angular momentum in the flight phase. The main theoretical result of this paper is that invariance of the flight and stance phase constraint surfaces over the landing event is not a necessary condition for achieving provably stable running. As noted earlier, relaxing this condition makes running motions significantly easier to design.

4.3 Optimization Algorithm Details

Trial gaits for the running experiments were generated using the constrained nonlinear optimization routine `fmincon` of MATLAB's Optimization Toolbox. Three quantities are involved in optimization: J, a scalar cost function to be minimized on the periodic orbit, EQ, a vector of equality constraints, and $INEQ$, a vector of inequality constraints. The following is a description of the optimization procedure that was implemented. The independent and dependent terms[16] of optimization are given in Table 1. Note that when the optimizer terminates with the constraints satisfied, x_{s}^{+*} will be a point located on a closed-loop periodic orbit and the virtual constraints will be given by (11) and (13).

[16] "Terms" is used to describe those variables used in optimization; these are different from the *parameters* of the virtual constraints.

Algorithm

1. Select $x_{\mathrm{f}}^{-*} = (q_{\mathrm{f}}^{-*}, \dot{q}_{\mathrm{f}}^{-*})$, the state corresponding to the end of the flight phase.
2. Using the flight-to-stance transition function, $\Delta_{\mathrm{f}}^{\mathrm{s}}$, calculate $x_{\mathrm{s}}^{+*} = (q^{+*}, \dot{q}^{+*})$, the state corresponding to the beginning of the subsequent stance phase.
3. Calculate θ_{s}^{+*} by (16) and $a_{\mathrm{s},0}^{*}$, $a_{\mathrm{s},1}^{*}$ by (23).
4. Select $a_{\mathrm{s},2}^{*}, \ldots, a_{\mathrm{s},m_{\mathrm{s}}}^{*}$, and θ_{s}^{-*} to complete the stance phase parameter vector a_{s}.
5. Using parameters a_{s} and the initial condition x_{s}^{+*}, integrate the equations of motion of stance and apply the stance-to-flight transition operator, $\Delta_{\mathrm{s}}^{\mathrm{f}}$, to obtain $x_{\mathrm{f}}^{+*} = (q_{\mathrm{f}}^{+*}, \dot{q}_{\mathrm{f}}^{+*})$.
6. Calculate $a_{\mathrm{f},0}^{*}$, $a_{\mathrm{f},1}^{*}$ by (22); $a_{\mathrm{f},m_{\mathrm{f}}-1}^{*}$, $a_{\mathrm{f},m_{\mathrm{f}}}^{*}$ by (23); and $x_{\mathrm{cm,f}}^{+*}$, $\dot{x}_{\mathrm{cm,f}}^{+*}$, and T_{f}^{*} by (17).
7. Select $a_{\mathrm{f},2}^{*}, \ldots, a_{\mathrm{f},m_{\mathrm{f}}-2}^{*}$ to complete the flight phase parameter vector a_{f}.
8. Using parameters a_{f}, and initial condition x_{f}^{+*}, integrate the equations of motion of flight to obtain x_{f}^{-}.
9. Evaluate J, EQ, and $INEQ$.
10. Iterate Steps 1 to 8 until J is (approximately) minimized, each entry of EQ is zero, and each entry of $INEQ$ is less than zero.

4.4 An Example Running Motion

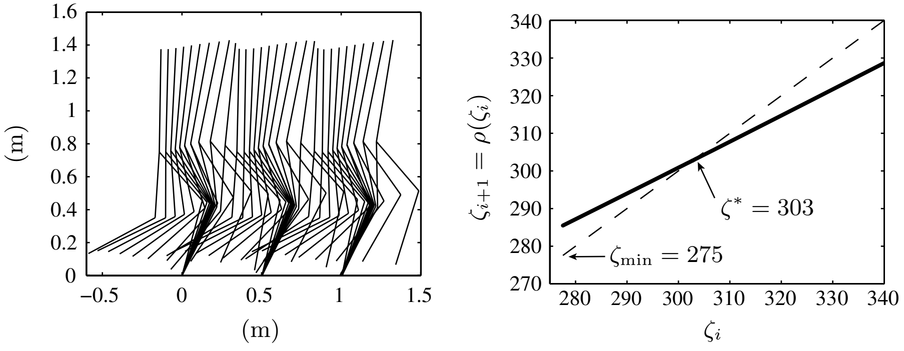
A sample running gait designed by the above algorithm is now presented. A stick diagram of this motion is given in Fig. 3(a). The stability analysis outlined in Sect. 3.8 was applied to the resulting running motion. Figure 3(b) gives the restricted Poincaré map, which indicates that the motion is locally exponentially stable. The gait was designed to minimize the integral of torque squared per distance traveled, with the following constraints:

Equality Constraints, *EQ*

- error associated with finding a fixed point $||x_{\mathrm{f}}^{-} - x_{\mathrm{f}}^{-*}||$
- deviation from the desired running rate
- required frictional forces at the leg ends are zero just before takeoff and just after landing (to prevent slipping at these transitions)

Inequality Constraints, *INEQ*

- magnitude of the required torque at each joint less than 100 Nm
- knee angles to lie in $(0°, -70°)$ and hip angles to lie in $(130°, 250°)$ (see Fig. 1(b) for measurement conventions)
- minimum height of the swing foot during stance greater than 7 cm



(a) Stick diagram

(b) Poincaré map

Fig. 3. Stick diagram and Poincaré map for the example running motion (rate 0.58 m/s). Poincaré map constructed by evaluating $\zeta := (\sigma_{\mathrm{s},1}^{-})^2/2$ at the end of successive stance phases, where $\sigma_{\mathrm{s},1}^{-}$ is the angular momentum about the stance leg end just before liftoff. The fixed point, $\zeta^* = 303$, is located at the intersection of ρ and the identity map $\zeta_i = \zeta_{i+1}$, and corresponds to an equilibrium running rate of 0.58 m/s. The slope of the graph at ζ^* is $d\rho/d\zeta \approx 0.67$, indicating exponential stability

- required coefficient of friction of the stance phase less than 0.7
- flight time greater than or equal to 25% of total gait duration
- landing foot impacts the ground at an angle of approach less than 45° from vertical
- joint angular velocities less than 5 rad/s

5 Experiment

5.1 Hardware Modifications to RABBIT

Prior to the experiment reported here, only walking experiments had been performed with RABBIT. To prepare for the task of running, four hardware modifications were made.

The first modification was the inclusion of prosthetic shock absorbers in the shanks. It was speculated that with shock absorbers the landing would cause less wear and tear on the harmonic drive gear reducers that form RABBIT's hip and knee joints. The inclusion of shock absorbers added approximately 5 cm to each shank.

The second modification was the installation of force sensitive resistors into RABBIT's point feet. These devices allowed for more accurate measurement of the touchdown time than did the previously installed mechanical contact switches. Since these sensors suffer from significant drift, their signals were numerically differentiated to make easier the detection of impact events.

The last two modifications were the bolting of aluminum u-channel stock along each thigh and the widening of the hips. Both of these changes were made to help prevent flexing of the legs in the frontal plane. Significant flexing was witnessed during the first several experimental trials of running. This problem was more pronounced in running than in walking because of the greater impact forces associated with landing. On several occasions RABBIT "tripped itself" during a stance phase of running when the swing leg passed by the stance leg (the legs knocked against each other). This came about because RABBIT was designed to have its legs close together to better approximate a planar biped.

5.2 Result: Six Running Steps

After completing hardware modifications and successfully reproducing previous walking experiments, running experiments were conducted. A number of experimental trials resulted in RABBIT taking several human-like[17] running steps. One such trial, which was an implementation of the example running motion of Sect. 4.4, will be discussed here.

For this experiment, motion was initiated by an experimenter who pushed the robot forward, into the basin of attraction of a walking controller that induced walking with an average forward walking rate of 0.8 m/s. RABBIT then achieved stable walking, followed by a transition to running in a single step, followed by 6 running steps. After the sixth step, the experiment was terminated by the control software when the tracking error limit of 0.3 radians was exceeded for the stance knee angle. Examination of collected data suggests that tracking error resulted from actuator saturation[18]. Data also show the swing leg extremely close to the ground at the moment the experiment was terminated, suggesting the swing leg may have, in fact, struck the ground contributing additional tracking error.

A plot of estimated[19] foot height is given in Fig. 4. Average stride duration for the steps was 431 ms. Flight times, observed as those portions of Fig. 4 where neither leg is at zero height, lasted an average of 107 ms (25% of the stride). Videos of the experiment and many additional data plots are available at [7, 17].

[17] A human-like gait is considered to be characterized by an upright posture, a torso leaning slightly forward, and a long step length.

[18] See [18] for a description of the PD controllers used to enforce the virtual constraints.

[19] When RABBIT is in flight, there is no accurate way to determine hip height. A sensor was mounted to record boom pitch angle, but due to flexing of the boom, these data were inaccurate. During the stance phase this lack of sensing is not a problem because the end of the stance leg is always at zero height.

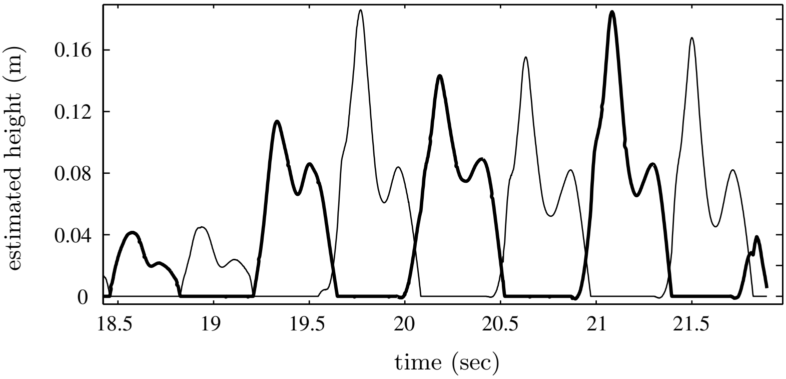


Fig. 4. Estimated height of the point feet with RABBIT's left foot indicated in bold. Flight phases occur when neither foot is at zero height

5.3 Discussion

Several problematic issues related to RABBIT's hardware did not appear until running was attempted. (For a discussion of general implementation issues of walking including unmodeled effects of the boom, gear reducers, and an uneven walking surface see [18].) Future running experiments–whether on RABBIT or another, similar mechanism–should take into account the following issues.

Boom Dynamics

The perturbing effects of the boom were found to be much more significant during flight phases than during stance phases. When RABBIT is modeled as a planar system, an analysis of the three-dimensional mechanics shows that the contribution of the boom to the center of mass dynamics is significant. Specifically, q_5 is no longer, in general, a cyclic variable during flight. However, if boom masses are appropriately distributed, the parabolic motion of the center of mass, as modeled in a planar system, is recovered. Unfortunately, this special mass distribution was impossible because RABBIT does not have a counterweight system.

Walking Surface

The walking surface was also a source of problems. This surface–consisting of rubber over elevated plywood supported on the edges by a wood frame–was originally built to provide a uniform, level surface. Although the surface appears uniform, walking experiments demonstrated otherwise. It was found that the surface has “fast” and “slow” areas corresponding to varying floor stiffness and coefficient of friction.

Limited Joint Space

For safety, RABBIT's joints have hard stops that limit its joint space, which, for example, prevent the shank from contacting the thigh. Although the available joint space was sufficient for walking, it became a significantly limiting factor in the design of running gaits. These hard stops prevented the swing leg from being folded close to the hip, which is a natural and desirable motion that minimizes the leg's rotational inertia.

6 Conclusion

A novel approach to the control of running in planar bipeds and its first experimental implementation on RABBIT have been presented. The control law is hybrid, consisting of continuous actions in the stance and flight phases, and discrete actions at the transitions between these phases. In the stance and flight phases, the controller coordinates the relative motions of the robot's links by imposing virtual constraints at the actuated joints. At the transition from stance to flight, the controller adjusts the virtual constraints for the flight phase as a function of estimated flight duration to ensure that the former swing leg is advanced properly to take up its role as the next stance leg. At the transition from flight to stance, the controller updates the virtual constraints of the stance phase to account for the orientation of the robot at landing. For the nominal periodic running motion, the parameters of the virtual constraints are determined by numerical optimization in order to meet actuator power limits, friction bounds, joint limits, etc. For running experiments, RABBIT's mechanical and electrical systems were modified: shock absorbers were added to the shanks; the ground contact sensors were improved; the stiffnesses of legs in the frontal plane were increased; and the hips were widened.

The main theoretical result of this paper was the development of a running controller that is based on the HZD methodology, but easier to design and implement while still resulting in a reduced-dimensionality stability test. The main experimental result of this paper was the physical realization of six consecutive running steps with a human-like gait and identification of hardware difficulties of running with RABBIT that were not present in walking.

Acknowledgments

The work of J.W. Grizzle and B. Morris was supported by NSF grant ECS-0322395. The work of E.R. Westervelt was supported by NSF grant CMS-0408348.

References

[1] S. P. Bhat and D. S. Bernstein. Finite-time stability of continuous autonomous systems. *SIAM J. Contr. Optim.*, 38:751–66, 2000.

[2] G. Buche. ROBEA Home Page. `http://robot-rabbit.lag.ensieg.inpg.fr/English/`.

[3] C. Chevallereau, G. Abba, Y. Aoustin, F. Plestan, E. R. Westervelt, C. Canudas de Wit, and J. W. Grizzle. RABBIT: a testbed for advanced control theory. *IEEE Control Systems Magazine*, 23(5):57–79, October 2003.

[4] C. Chevallereau and Y. Aoustin. Optimal reference trajectories for walking and running of a biped robot. *Robotica*, 19(5):557–69, September 2001.

[5] C. Chevallereau, E. R. Westervelt, and J. W. Grizzle. Asymptotically stable running for a five-link, four-actuator, planar bipedal robot. *International Journal of Robotics Research*, 24:431–64, 2005.

[6] B. Espiau and A. Goswani. Compass gait revisited. In *Proc. of the IFAC Symposium on Robot Control, Capri, Italy*, pp.839–46, September 1994.

[7] J. W. Grizzle. Jessy Grizzle's publications. `http://www.eecs.umich.edu/~grizzle/papers/robotics.html`.

[8] J. W. Grizzle, G. Abba, and F. Plestan. Asymptotically stable walking for biped robots: Analysis via systems with impulse effects. *IEEE Transactions on Automatic Control*, 46:51–64, January 2001.

[9] J. Guckenheimer and S. Johnson. Planar hybrid systems. In *Hybrid Systems II, Lecture Notes in Computer Science*, pp. 203–25. Springer-Verlag, 1995.

[10] I. A. Hiskens. Stability of hybrid limit cycles: application to the compass gait biped robot. In *Proc. of the 40th IEEE Conf. Dec. and Control, Orlando, FL*, December 2001.

[11] Y. Hürmüzlü and D. B. Marghitu. Rigid body collisions of planar kinematic chains with multiple contact points. *International Journal of Robotics Research*, 13(1):82–92, 1994.

[12] A. Isidori. *Nonlinear Control Systems: An Introduction.* Springer-Verlag, Berlin, third edition, 1995.

[13] S. Kajita, T. Nagasaki, K. Kaneko, K. Yokoi, and K. Tanie. A hop towards running humanoid biped. In *Proc. of the 2004 IEEE International Conference on Robotics and Automation*, pp. 629–35, 2004.

[14] T. A. McMahon, G. Valiant, and E. C. Frederick. Groucho running. *Journal of Applied Physiology*, 62(6):2326–37, June 1987.

[15] S. G. Nersesov, V. Chellaboian, and W. M. Haddad. A generalization of Poincaré's theorem to hybrid and impulsive dynamical systems. *Int. J. Hybrid Systems*, 2:35–51, 2002.

[16] M. H. Raibert. *Legged robots that balance.* MIT Press, Cambridge, MA, 1986.

[17] E. R. Westervelt. Eric Westervelt's publications. `http://www.mecheng.osu.edu/~westerve/publications`.

[18] E. R. Westervelt, G. Buche, and J. W. Grizzle. Experimental validation of a framework for the design of controllers that induce stable walking in planar bipeds. *International Journal of Robotics Research*, 23(6):559–82, 2004.

[19] E. R. Westervelt, J. W. Grizzle, and C. Canudas de Wit. Switching and PI control of walking motions of planar biped walkers. *IEEE Transactions on Automatic Control*, 48(2):308–12, February 2003.

[20] E. R. Westervelt, J. W. Grizzle, and D. E. Koditschek. Hybrid zero dynamics of planar biped walkers. *IEEE Transactions on Automatic Control*, 48(1):42–56, January 2003.

Velocity-Based Stability Margins for Fast Bipedal Walking

J.E. Pratt[1] and R. Tedrake[2]

[1] Florida Institute for Human and Machine Cognition
jpratt@ihmc.us
[2] Massachusettes Institute of Technology
russt@mit.edu

Summary. We present velocity-based stability margins for fast bipedal walking that are sufficient conditions for stability, allow comparison between different walking algorithms, are measurable and computable, and are meaningful. While not completely necessary conditions, they are tighter necessary conditions than several previously proposed stability margins. The stability margins we present take into consideration a biped's Center of Mass position and velocity, the reachable region of its swing leg, the time required to swing its swing leg, and the amount of internal angular momentum available for capturing balance. They predict the opportunity for the biped to place its swing leg in such a way that it can continue walking without falling down. We present methods for estimating these stability margins by using simple models of walking such as an inverted pendulum model and the Linear Inverted Pendulum model. We show that by considering the Center of Mass location with respect to the Center of Pressure on the foot, these estimates are easily computable. Finally, we show through simulation experiments on a 12 degree-of-freedom distributed-mass lower-body biped that these estimates are useful for analyzing and controlling bipedal walking.

1 Introduction

“How stable is your robot?” is a fundamental yet challenging question to answer, particularly with fast moving legged robots, such as dynamically balanced bipedal walkers. With many traditional control systems, questions of stability and robustness can be answered by eigenvalues, phase margins, loop gain margins, and other stability margins. However, legged robots are nonlinear, under-actuated, combine continuous and discrete dynamics, and do not necessarily have periodic motions. These features make applying traditional stability margins difficult.

In this paper we define stability for a biped simply as whether or not the biped will fall down. We focus on velocity-based stability since we believe that regulating the velocity of the Center of Mass is the most challenging subtask for human-like bipedal walking. Regulating velocity is a challenging subtask due to the extended period during a natural gait that the Center of Mass velocity is underactuated (the actuators cannot produce an arbitrary acceleration on the Center of Mass). For example, once the body has traveled far enough away from the foot, the only course of action that can stabilize the Center of Mass velocity is to take a step. Other requirements such as regulating virtual leg length and body orientation, and swinging the swing leg, can be met through traditional control system techniques since these subtasks are fully actuated during the majority of the gait.

We argue that the most crucial thing for regulating velocity in bipedal walking is the ability to place the foot of the swing leg in a proper location that allows for maintaining or reducing velocity on subsequent steps. A sufficient condition for being able to maintain or reduce velocity is the ability to eventually come to a stop. Thus we define stability margins that estimate the likelihood that a biped can "Capture" its kinetic energy and stop over a given number of steps. We define a Capture Point as a point that can be stepped to in order to stop. We propose the "N-Step Capture Stability Margin" which gives an indication of the degree to which a biped can stop in N steps. For most practical purposes if a walking biped cannot stop within several steps, it is probably close enough to falling to consider it unstable. Hence being, say, 10-Step Capturable is a sufficient, and close to necessary, stability condition.

Exact computation of these stability margins is difficult since the dynamics of bipedal walking is complex. However, we *can* compute these margins for simplified walking models. These simplified walking models give useful approximations to the real values of the margins, which we demonstrate through control experiments on a simulated 12 degree-of-freedom lower-body biped with distributed mass. The robot can recover from being pushed by stepping to a Capture Point computed from the simplified walking models. It can step to desired foothold locations by guiding the Capture Point to the desired stepping point and stepping once the Capture Point reaches it.

2 Stability Definition

In this paper, we define stability for a biped in terms of whether or not the biped will fall down. However, the concept of falling down is difficult to precisely define. For example, sitting down on the floor and slipping down onto the floor might result in the exact same trajectories and end state but one is considered falling and the other is considered sitting, with the only difference being intent.

For the purposes of this paper, to eliminate complications rising from such concepts as intent, let us define a fall as follows.

Definition 1 (Fall). *When a point on the biped, other than a point on the feet of the biped, touches the ground.*

This definition allows for such things as holding a hand railing to not be considered a fall, by pushing the burden onto a precise definition of the ground and of the feet. However, in this paper, we only consider situations in which the ground is everything that is not part of the biped and the feet are the two commonly known appendages at the end of the legs.

Turning now to the question of whether the biped will fall, consider the deterministic closed-loop dynamics of the biped in the general form $\dot{\mathbf{x}} = f(\mathbf{x}, t)$, where $\mathbf{x} \in \Re^N$ is the state vector. Define a subset $F \subset \Re^N$ which includes all configurations of the robot for which some part of the robot other than the feet is touching the ground. The basin of attraction of F, which we will call the "Basin of Fall", defines all of the states of the robot that eventually lead to a fall:

Definition 2 (Basin of Fall). *Subset of state space that leads to a fall.* $B \subset \Re^N, x(t) \in B \Rightarrow \exists \varDelta t \geq 0$ *s.t.* $\mathbf{x}(t + \varDelta t) \in F$

The complement of the Basin of Fall is the Viability Kernel of Wieber [32]. Whether or not the robot will fall down, and hence whether or not the robot is stable, can now be precisely defined.

Definition 3 (Stable). *A biped is stable if and only if the state of the robot is not inside the Basin of Fall, B.*

Note that for a real-world biped in a non-deterministic environment, the Basin of Fall may be the entire state-space, as all bipeds will eventually fall given enough time. Also, note that for a biped that has regions of chaotic gait, the Basin of Fall may be Uncomputable as determining whether some states are in the Basin of Fall may be Undecideable [26]. In addition, for the system to be deterministic, the dynamics, f, and the state vector, $\mathbf{x}$, must contain full information about the environment such as the ground profile. Encoding the entire environment for all time is prohibitive in general.

Therefore, to ensure computability, non-zero volume of stable states, and feasible definition of the state and environment, one may wish to consider the state of the biped to be stable if it does not lead to a fall after some reasonable finite amount of time, τ, and define the Time-Limited Basin of Fall as follows:

Definition 4 (Time-Limited Basin of Fall). *Subset of state space that leads to a fall within a finite amount of time, τ.* $B_\tau \subset \Re^N, x(t) \in B_\tau \Rightarrow \exists \varDelta t, 0 \leq \varDelta t \leq \tau$ *s.t.* $\mathbf{x}(t + \varDelta t) \in F$

For human-like walking, approximately 1 minute is a reasonable horizon to consider since it is highly unlikely that a biped would be in a state where a fall is inevitable but the biped can stagger around before the fall for a whole minute.

Computing the Basin of Fall, B, is conceptually simple, but computationally expensive. Given an accurate closed-loop dynamic model and a discretized state space, one can use dynamic programming to determine B. Let $V(\mathbf{x})$ be the value function encoding the time until the biped falls. Initialize V to zero for states that are falling states and infinity for all other states. Then use the update rule $V(\mathbf{x_n}) = \min(V(\mathbf{x_n}), V(\mathbf{x_{n+1}}) + \Delta t)$ where state x_n dynamically leads to state x_{n+1} and Δt is the time taken to transition from state x_n to x_{n+1}. In the worst case N^2 updates would have to occur, where N is the number of discrete states. If the dynamics can be simulated backward in time, then only N updates need be made if the states are visited recursively, starting from falling states. In a companion paper [28], we will use similar ideas to explore stochastic stability margins for legged locomotion.

A stability margin that logically follows from the previous discussion is the distance to the Basin of Fall, called the Viability Margin by Wieber [32].

$$M_{Viability}(\mathbf{x}) = \begin{cases} \min_{\mathbf{x'} \in B} \left(|\mathbf{x} - \mathbf{x'}|\right) \text{ if } \mathbf{x} \notin B \\ -\min_{\mathbf{x'} \notin B} \left(|\mathbf{x} - \mathbf{x'}|\right) \text{ if } \mathbf{x} \in B \end{cases} \tag{1}$$

By definition the Viability Margin is both a necessary and sufficient indicator of stability and thus would be a good stability margin for a biped. However, it does have drawbacks. It does not take into consideration what disturbances we would like the biped to be robust to, since distance in state space does not necessarily correlate to real-world disturbances. It does not take into consideration the dynamics of the system in moving from the current state to the nearest boundary state, since Cartesian nearness does not necessarily correlate with dynamic nearness. It assigns importance to each degree of freedom based upon its units of measure. It is difficult to compute. And finally, it does not give us much insight into why a biped is stable or why it falls. Some of these problems could be mitigated by such things as weighting the state variables based on importance, for each state computing the magnitude of a given disturbance required to transition the biped into a Basin of Fall state, etc. However, these additions would only add to the computational complexity of computing this margin.

Additionally, perfect dynamic models are impossible to attain for real systems, and to compute the entire Basin of Fall during experimental trials would require an infeasible number of trials, many of which could result in damage to the biped.

Due to these limitations, instead of attempting to compute the entire Basin of Fall, in this paper we endeavor to develop heuristic stability margins that approximate whether a state is in the Basin of Fall. These margins, such as the N-Step Capture Margin, give an indication of the ability of the biped to come to a stop within a given number of steps. Previous stability margins, such as the static stability margin, are similar heuristic approximations, but are typically too conservative. In the next section we discuss desirable characteristics for heuristic bipedal walking stability margins.

3 Desirable Characteristics of Stability Margins

An ideal stability margin for a biped would act as a fortune teller. It would tell us when the biped is going to fall down next, what the cause will be, and how it can be prevented. If the biped is not going to fall down, the margin would indicate the closest the biped will be to falling down in the next step or so, at what point during the gait this occurs, and how much extra disturbance it could handle.

While such omniscience is infeasible for anything but the simplest systems, some reasonable characteristics we may desire for stability margins include:

- Necessary. If the stability margin is outside an acceptable threshold of values, the robot will fall down.
- Sufficient. If the stability margin is inside the acceptable threshold of values, the robot will not fall down.
- Comparable. Two control algorithms should be comparable for stability based on their relative stability margins.
- Measurable and Computable. One should be able to measure the relevant state variables and estimate the stability margin on-line in order to use it for control purposes.
- Meaningful. The stability margin should answer relevant questions as to why the robot fell. It should correlate with the degree of robustness to disturbances, such as noise, terrain irregularities, and external forces or impulses.

The Viability Margin is necessary, sufficient, and allows comparisons. However, its main drawback is that it is very difficult to compute. Various heuristic stability margins, which are much easier to compute, have been used in analyzing and controlling bipeds. In the next section we review some of the margins that are commonly used for bipedal walking and discuss how well they achieve these desirable characteristics. In the subsequent sections we introduce some heuristic stability margins which we believe more accurately measure the stability of a biped as defined by Definition 3.

4 Review of Stability Margins for Bipedal Walking

While there have been many proposed ways to define stability for a bipedal walking robot, we argue that many of these do not adequately address the desired characteristics described above. Here we review eigenvalues of Poincare return maps [14], the Zero Moment Point (ZMP) criterion [30], the foot rotation indicator (FRI) [7], and change of angular momentum [1, 20] as potential stability margins in light of the desired characteristics.

4.1 Eigenvalues of Poincare Return Maps

For a periodic system, trajectories in state space will cycle and return each cycle to a point in a slice of state space. Deviations from a periodic limit cycle will return to the cycle, and for small deviations, typically follow a linear relation,

$$X_{n+1} = KX_n \; , \tag{2}$$

where X is the vector of deviations from the fixed point that the limit cycle passes through and K is a linear return matrix. One of the eigenvalues of K, corresponding to the evolution of the orbit, will be 1.0. If the magnitude of the remaining eigenvalues of K are all less than one, then the limit cycle is stable. Thus, the magnitude of the largest eigenvalue of the return map, disregarding the eigenvalue corresponding to the orbit, is a suitable stability margin for a periodic system. Measuring the eigenvalues of Poincare return maps is commonly used for analyzing Passive Dynamic Walking robots [16, 8, 27, 2, 4, 29] and was used by Miura and Shimoyama [17] to analyze their Biper robots.

However, using eigenvalues of Poincare return maps assumes periodicity and is valid only for small deviations from a limit cycle. While most bipedal systems, including humans, indeed seem to be periodic, there is nothing about the bipedal walking problem that requires periodicity. In particular, a biped's motion is not periodic when walking over discontinuous rough terrain, or when abruptly changing speed or direction of travel. In addition, large disruptions from a limit cycle, such as when being pushed, cannot be analyzed using this technique as it assumes small deviations. Therefore, while eigenvalue magnitudes of Poincare return maps may be sufficient for analyzing periodic bipedal walking, they are not sufficient for analyzing nonperiodic motions, and are not necessary for analyzing bipedal walking in general.

4.2 Zero Moment Point (ZMP) and Foot Rotation Indicator (FRI)

The Zero Moment Point (ZMP) is the location on the ground where the net moment generated from the ground reaction forces has zero moment about two axes that lie in the plane of the ground [30]. The ZMP when used in control algorithm synthesis for bipedal walking robots typically is computed analytically based upon desired trajectories of the robot's joints. As long as the ZMP lies strictly inside the support polygon of the foot, then these desired trajectories are dynamically feasible. If the ZMP lies on the edge of the support polygon, then the trajectories may not be dynamically feasible.

During playback of the desired joint trajectories, the actual ZMP is measured from force sensors in the foot or by observing accelerations of all the joints [10]. Then deviations between the precomputed and actual ZMP are typically used to modify the joint trajectories [34, 9]. The ZMP is equivalent to the Center of Pressure (CoP) but is commonly used to mean the analytically computed point based on the state and acceleration of the robot whereas the CoP is commonly used to mean the point measured from ground reaction forces.

The Foot Rotation Indicator (FRI) point [7] is the point on the ground where the net ground reaction force would have to act to keep the foot stationary given the state of the biped and the accelerations of its joints. If the foot is stationary, then the FRI, the ZMP, and the CoP are all the same point. If the foot is experiencing rotational acceleration, then the ZMP and CoP are on an edge of the support polygon, and the FRI is outside the support polygon. Therefore the FRI is a more general form of the ZMP and provides both a positive and negative margin when used for control and analysis purposes.

The ZMP stability margin is the distance from the ZMP to the nearest edge of the convex hull of the support polygon. In its typical use it measures how much room for error there is in achieving the desired trajectories of the robot. If the ZMP reaches the edge of the support polygon, then the desired Center of Mass trajectory may no longer be dynamically feasible.

Maintaining the ZMP inside the support polygon is not a necessary condition for stable walking. During toe-off in human walking and in walking of the robot Spring Flamingo [23], the ZMP stability criterion is violated, the FRI point lies outside the foot, and the foot rotates. For bipeds with point feet [11, 24, 5, 31], and Passive Dynamic Walkers with curved feet, when on one support foot, the ZMP, FRI, and CoP have little value as they are all simply the location of the foot, and the ZMP criterion is always violated. Maintaining the ZMP inside the support polygon is also not a sufficient condition for stable walking since a biped can fall down while its ZMP remains in the center of its foot. As an example, if all the joint torques are set to zero, the robot will collapse while the ZMP stays inside the foot.

The ZMP criterion does become a sufficient criterion for stable walking when coupled with a particular class of control algorithms, typically those that rely on stiff tracking of predetermined joint trajectories [13, 34, 9, 19]. One way to state the ZMP criterion is: *Given desired state variable trajectories that are consistent with the dynamics and that predict the ZMP staying inside the support polygon, a trajectory tracking control algorithm can stably track those trajectories as long as the ZMP does indeed remain inside the support polygon.*

In these terms the ZMP criterion is a sufficient criterion for stable walking and has become a very powerful tool for trajectory generation and verification of the dynamic feasibility of trajectories during execution. However, its utility has perhaps led to its overuse, resulting in the majority of bipedal robots relying heavily on prerecorded trajectories and stiff joint control to achieve those trajectories. Such stiff joint control of prerecorded trajectories typically leads to poor robustness to pushes and unknown rough terrain, relies on a flat-footed gait, and makes it difficult to incorporate natural dynamic mechanisms that have shown their utility in Passive Dynamic Walkers [16, 8, 27, 2, 4, 29], and a growing number of powered bipeds [22, 3, 33].

Since the ZMP criterion is a sufficient condition for stable walking only when coupled with a particular class of control system, as pointed out by

Abdallah and Goswami [1] the ZMP criterion should be viewed as a controllability criterion that assesses the dynamic feasibility of a desired trajectory through state-space.

It is important to note that the ZMP, CoP, and FRI are not functions of the state variables of the robot, but rather require knowledge of either the joint torques and/or the joint accelerations. Thus, they themselves cannot be viewed as state variables, and thus are not good candidates for stability margins that are a function of the state of the robot. Instead, for this paper and in previous work [23, 22], we view the Center of Pressure as a control input that can be arbitrarily and instantaneously changed, within the limits of actuator bandwidth, force limits, and the limitation that it stays inside the support polygon.

4.3 Angular Momentum

Motivated by observations that humans appear to regulate angular momentum about the Center of Mass when standing, walking, and running, researchers [1, 20] have suggested that angular momentum about the Center of Mass (refered to as spin angular momentum) of a biped should be minimized throughout a motion. They argue that the spin angular momentum should be used when needed to balance, such as when lunging to prevent a fall after being pushed, or windmilling one's arms when standing on a balance beam. The amount of angular momentum that is available is limited by joint angle limits, joint speed limits, and joint power limits. By minimizing angular momentum when not required, the biped has a reserve it can draw on when necessary.

Minimizing spin angular momentum is not a necessary condition for stable walking. One can walk while violently thrashing his or her upper body mass around. It is not graceful, nor efficient, and perhaps grace and efficiency are the primary reasons humans do not usually walk in this fashion. Minimizing spin angular momentum is also not a sufficient condition for stable walking, as a biped can fall over while maintaining an angular momentum of zero. Therefore, we believe angular momentum about the Center of Mass, taken purely on its own, is not a good indicator of whether a biped will fall and hence not a good stability margin by itself.

However, the reserve in spin angular momentum that can be utilized to help recover from a push or other disturbance is important, as there is a coupling between angular momentum rate change and linear momentum rate change, and hence speed. This coupling arises from the fact that the net angular momentum rate change of the biped, *about the Center of Pressure,* is only modified by gravity (Fig. 1). Since the ground reaction force, by definition, acts through the Center of Pressure it does not affect the angular momentum about this point. The angular momentum dynamics about the Center of Pressure can therefore be written as

$$\dot{H}_{tot} = mgl \sin\theta_1 \ , \tag{3}$$

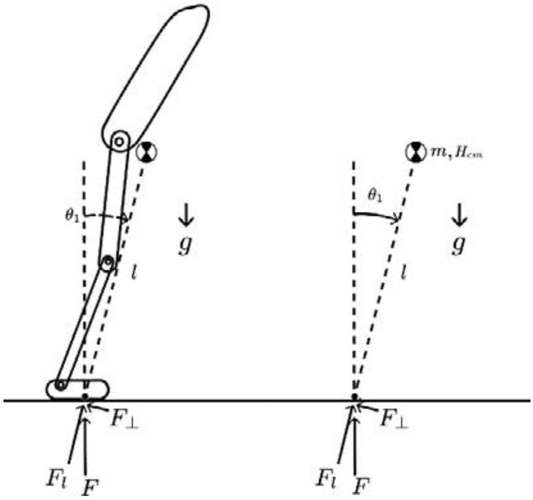


Fig. 1. Bipedal model for sagittal plane dynamics. l is the virtual leg length from the Center of Pressure to the Center of Mass. θ_1 is the angle from vertical to the virtual leg. F is the ground reaction force, with component F_l along the virtual leg and $F_\perp$ perpendicular to it. m and H_{cm} are the mass and angular momentum about the Center of Mass of the biped. g is the gravitational acceleration constant

where H_{tot} is the total angular momentum about the Center of Pressure, m is the mass of the robot, g is the gravitational acceleration, l is the distance from the Center of Pressure to the Center of Mass, and θ_1 is the angle from the Center of Pressure to the Center of Mass with vertical being zero. The total momentum about the Center of Pressure consists of the angular momentum of the Center of Mass rotating about the Center of Pressure, plus the spin angular momentum about the Center of Mass:

$$H_{tot} = H_0 + H_{cm} = ml^2\dot{\theta}_1 + H_{cm} \ . \tag{4}$$

Differentiating, we get

$$\dot{H}_{tot} = mgl\sin\theta_1, = ml^2\ddot{\theta}_1 + 2ml\dot{l}\dot{\theta}_1 + \dot{H}_{cm} \ . \tag{5}$$

The first term, $ml^2\ddot{\theta}_1$ is the acceleration of the Center of Mass penduluming around the Center of Pressure. The second term, $2ml\dot{l}\dot{\theta}_1$ encodes the coupling of distance to Center of Mass and rotational velocity (which makes a person rotate faster when they pull their legs in on a tire swing). Since only gravity has an influence on $\dot{H}_{tot}$, equation 5 demonstrates the coupling between acceleration of spin angular momentum about the Center of Mass, $\dot{H}_{cm}$, and acceleration of the Center of Mass about the Center of Pressure, $\ddot{\theta}_1$. A clockwise acceleration of internal inertia will create a counterclockwise acceleration of the Center of Mass penduluming over the Center of Pressure and vice versa. That is what leads to the counterintuitive strategy of lunging your upper body forward to prevent falling forward.

Since spin angular momentum is useful for catching balance in this way, having a reserve is useful. However, instead of using the reserve of spin angular momentum as a stability margin by itself, in Sect. 7 we propose using it as it relates to reducing the velocity of the biped. In the next section we justify our focus on velocity for analyzing bipedal stability.

5 Bipedal Walking and Stability

There are five subtasks that a biped performs while walking.

T1 Maintain body orientation within a reasonable bound.
T2 Maintain virtual leg length within a reasonable bound.
T3 Swing the swing leg.
T4 Transfer support from one support leg to the other.
T5 Regulate Center of Mass velocity.

For human-like bipedal walking, subtasks T1 through T3 are achievable through traditional control means as long as the support leg is firmly planted and joint torque limits that prevent foot slippage are observed. While there are occasions during the gait cycle where these tasks are not fully controllable, such as perhaps at the end of toe-off, we argue that there is plenty of opportunity during a normal bipedal walking gait to control orientation, virtual leg length, and leg swing using traditional techniques that depend on full controllability and full observability.

Subtask T4, transfer of support, is difficult when attempted using traditional high-gain joint position control techniques due to the over-constrained kinematics that the resultant closed loop kinematic structure presents [25]. However, when low impedance force control techniques are used [23], the system is no longer over-constrained and smooth transfer of support can occur relatively easily.

We contend that the most difficult subtask in bipedal walking from a controllability point of view is subtask T5, regulating the Center of Mass velocity. It is this subtask that makes bipedal walking both an interesting and challenging problem. It is the most difficult subtask as the degrees of freedom that contribute to the velocity vector are under-actuated from a continuous dynamics point of view. Once the Center of Mass projection on the ground moves even a small distance out of the support polygon of the feet, then a significant amount of angular acceleration of internal mass, such as windmilling the arms or lunging the upper body, must occur to bring the Center of Mass back. Beyond a certain distance, the Center of Mass cannot be brought back at all and the biped must take a step. Thus velocity can be regulated only through a combination of the continuous dynamics and the discrete dynamics.

It is this lack of actuation and the requirement of taking a step to continue walking that leads many to describe walking as a sequence of controlled falling.

We can illustrate the difficulty of regulating velocity versus orientation and leg length by looking at the dynamics of walking during single support. To simplify the discussion, we consider only sagittal plane dynamics (Fig. 1), but our discussion extends to 3D dynamics. The dynamics of the rotation of the mass about the Center of Pressure (5) is

$$\dot{H}_{tot} = mgl\sin\theta_1 = ml^2\ddot{\theta}_1 + 2ml\dot{l}\dot{\theta}_1 + \dot{H}_{cm} \tag{6}$$

where l is the virtual leg length from the Center of Pressure to the Center of Mass, θ_1 is the angle from vertical to the virtual leg, m and H_{cm} are the mass and angular momentum about the Center of Mass of the biped, and g is the gravitational acceleration constant.

As discussed in Sect. 4, humans tend to regulate their angular momentum about the Center of Mass, and thus the magnitude of $\dot{H}_{cm}$ is kept relatively low. Suppose $\dot{H}_{cm} = 0$. Since the virtual leg length, l is always positive, then equation 6 can be rewritten as

$$\ddot{\theta}_1 = -c_1\dot{l}\dot{\theta}_1 + c_2\sin\theta_1 \tag{7}$$

where c_1 and c_2 are always positive. If $\dot{\theta}_1$ and θ_1 have the same sign, then the magnitude of θ_1 must always be increasing, assuming $|\dot{l}| < \infty$ and $-\pi < \theta_1 < \pi$. In physical terms, this means that if the Center of Mass is moving away from the Center of Pressure, it cannot be stopped without either moving the location of the Center of Pressure (instantaneoulsy changing the value of θ_1), or by accelerating internal inertia. However, once the Center of Mass is beyond the support polygon of the foot, changing the Center of Pressure alone cannot stop further motion of the Center of Mass. At that point, only accelerating internal inertia ($\dot{H}_{cm} \neq 0$), or taking a step can prevent a fall. Due to joint range of motion, velocity, and torque limits, the amount of opportunity for using angular momentum to catch balance is limited. Therefore, during a large portion of a human-like walking gait, the only way to prevent a fall is to take another step.

Turning to the dynamics of the virtual leg length from the Center of Pressure to the Center of Mass we have

$$m\ddot{l} = ml\dot{\theta_1}^2 - mg\cos\theta_1 + F_l \tag{8}$$

where F_l is the component of the ground reaction force along the line from the Center of Pressure to the Center of Mass and is typically positive since the total ground reaction force must be positive and lie inside the friction cone. F_l is a function of the leg actuator forces and as long as the leg isn't straight can be arbitrarily and instantaneously set to any positive value, assuming ideal force-source actuators, as long as it doesn't result in slipping on the ground. Since mg is typically larger than $ml\dot{\theta_1}^2$ on the Earth at typical walking speeds, one can achieve a large range of negative $\ddot{l}$, and an even larger range of positive $\ddot{l}$, bounded only by the strength of the leg. Therefore, during a large percentage of the walking gait, the virtual leg length, l is fully controllable.

Turning to the dynamics of the angular momentum about the Center of Mass we have

$$\dot{H}_{cm} = lF_{\perp} \tag{9}$$

where $F_{\perp}$ is the component of the ground reaction force perpendicular to the line from the Center of Pressure to the Center of Mass. $F_{\perp}$ is a function of the leg actuator forces and can be arbitrarily and instantaneously set to any value, assuming ideal force-source actuators, as long as it doesn't result in slipping on the ground. Since a large range of positive and negative values of $\dot{H}_{cm}$ can be achieved during a large portion of the walking gait, angular momentum about the Center of Mass is fully controllable.

Note that in this discussion we do not break out the components of H_{cm} due to their complexity, nor do we consider the motion of internal degrees of freedom. We assume that all the internal degrees of freedom have an actuator associated with them, and only consider their net effect on the walking motion as it couples through H_{cm}. Achieving any orientation of internal body parts, or swinging the swing leg along a trajectory in internal coordinates is thus achievable with traditional control means.

Note that in this discussion we use $\dot{\theta}_1$ as our velocity variable (and could use $\dot{\theta}_2$ for a 3D analysis). However, any two variables that are independent of virtual leg length velocity, $\dot{l}$ could be used. Except for a fall, the virtual leg should never lie in the horizontal plane. Therefore, $\dot{x}$ and $\dot{y}$ would also be a suitable choice in the above analysis. As long as we choose a definition of a velocity vector that along with the virtual leg vector spans the 3D space, the above analysis can be modified to show that velocity is not fully controllable for a large percentage of a human-like walking gait.

We have shown that virtual leg length and body orientation are fully controllable during a large portion of a natural gait, with the main limitation in their control being the requirement that the foot doesn't slip on the ground. In contrast, velocity is not controllable with continuous dynamics once the Center of Mass has moved away from the support polygon. For human-like walking, velocity is controllable only through a combination of continuous dynamics and discrete dynamics (i.e. taking a step). This lack of controllability is why we consider velocity regulation the most challenging part of walking and why we focus on stability margins were velocity is central.

6 Capture Points and Capture Regions

A key capability required for robust 3D walking is the ability to place the swing leg foot at an appropriate position, such that the Center of Mass can come to rest over the foot. We call such a point a "Capture Point". Before defining a Capture Point, we define a Capture State and a Safe Feasible Trajectory:

Definition 5 (Capture State). *State in which the kinetic energy of the biped is zero and can remain zero with suitable joint torques.*

Definition 6 (Safe Feasible Trajectory). *Trajectory through state space that is consistent with the robot's dynamics, is achievable by the robot's actuators, and does not contain any Falling States.*

Note that the Center of Mass must lie above the support polygon in a Capture State. We now define a Capture Point:

Definition 7 (Capture Point). *For a biped in state x, a Capture Point, p, is a point on the ground where if the biped covers p, either with its stance foot or by stepping to p in a single step, and then maintains its Center of Pressure to lie on p, then there exists a Safe Feasible Trajectory that ends in a Capture State.*

Note that since height, swing, orientation, and velocity are dynamically coupled, the location of a Capture Point is dependent on the trajectory through state-space before and after swinging the leg and thus is not a unique point. Therefore, there exists a Capture Region on the ground such that if the Center of Pressure is placed inside this region, then the biped can come to a stop for some height and orientation trajectory.

Definition 8 (Capture Region). *The set of all Capture Points.*

The size of the Capture Region is highly dependent on how fast the swing leg can swing to a Capture Point before the biped has accelerated and the point is no longer a Capture Point. The faster the swing leg, the larger the area of the Capture Region. Therefore, all else being equal, a biped with a faster swing leg will have a higher margin of stability than one with a slower swing leg.

If constraints are placed on the subsequent height, swing, orientation, and internal motion of the robot, then the size of the Capture Region will be reduced. For example, we can define a momentum-regulated capture region as the set of all Capture Points that exist when $\dot{H}_{cm}$ is regulated during the subsequent motion. With enough constraints, the Capture Region may be reduced to a single point, or perhaps even vanish.

Note that our definition of Capture Point also implies that the swing leg *can* reach the point. Let us define an **Unreachable Capture Point** as a point that is not a Capture Point, but would be if the swing leg did not have kinematic constraints. The **Unreachable Capture Region** is then the set of all Unreachable Capture Points and there is no intersection between the Unreachable and Reachable Capture Regions.

While being able to stop in one step implies stability by definition 3, it is overconservative. Being able to stop in two steps is a less conservative estimate of stability and hence we consider Two-Step Capture Points.

Definition 9 (Two-Step Capture Point). *A point on the ground, p, such that if the biped swung its swing leg to cover p with its foot and maintained its Center of Pressure to lie on p, then there exists a Safe Feasible Trajectory, such that at some state along the trajectory, there exists a Capture Point.*

Definition 10 (Two-Step Capture Region). *The set of all Two-Step Capture Points.*

We can now define an N-Step Capture Point recursively:

Definition 11 (N-Step Capture Point). *A point on the ground, p, such that if the biped swung its swing leg to cover p with its foot and maintained its Center of Pressure to lie on p, then there exists a Safe Feasible Trajectory, such that at some state along the trajectory, there exists an N-1-Step Capture Point.*

Definition 12 (N-Step Capture Region). *The set of all N-Step Capture Points.*

If an N-Step Capture point exists, then we say that the biped is "N-Step Capturable". As N approaches ∞, the N-step Capture Region converges to the area on the ground that the foot can be placed at without resulting in an eventual fall.

Note that the above definitions require the maintenance of the Center of Pressure at a Capture Point, which is not necessary nor typical in walking. More general definitions would allow the Center of Pressure to move inside the foot. However, we conjecture that such definitions would result in equivalent Capture Regions, i.e. if a biped can stop by taking a step and then moving its Center of Pressure around its foot, then the biped can stop by taking the same step and maintaining the Center of Pressure at a single point inside the foot. We chose to maintain the Center of Pressure at a Capture Point in our definitions for computational reasons since doing so reduces the potential action space by two degrees-of-freedom. However, in controlling a biped, there is no such requirement.

The stability margins we propose in the next Section will give a measure of how much opportunity there is to stop based on the ability to reach a Capture Region with the swing leg. Knowing where the leg can swing to will be important and so we define the Reachable Region as follows.

Definition 13. *Reachable Region: Region on the ground that can be reached kinematically by any point on the bottom of the swing foot.*

7 Proposed Stability Margins

We now propose some velocity-based stability margins for bipedal walking that are based on Capture Regions. Because the Capture and Reachable Regions depend on the current state of the robot, $\mathbf{x}$, these margins are scalar functions of $\mathbf{x}$.

- **Zero-Step Capture Margin.** $M(\mathbf{x}, 0)$: Maximum distance from points in the Capture Region to their closest edge of the support polygon if the Capture Region and support polygon intersect. Otherwise, the negative distance from the Capture Region to the support polygon.
- **One-Step Capture Margin.** $M(\mathbf{x}, 1)$: Maximum distance from points in the Capture Region to their nearest boundary of the Reachable Region if the Capture Region is non-empty. Otherwise, the negative distance from the Unreachable Capture Region to the Reachable Region.
- **N-Step Capture Margin.** $M(\mathbf{x}, N)$: Maximum distance from points in the N-Step Capture Region to their nearest boundary of the Reachable Region if the N-Step Capture Region is non-empty. Otherwise, the negative distance from the Unreachable N-Step Capture Region to the Reachable Region.

Note that the Zero-Step Capture Margin is equivalent to the traditional static stability margin when velocity is negligible and angular momentum is not used for capturing balance. For a moving biped, it is more appropriate than the traditional margin, since the Center of Pressure needs to be placed near the Capture Point rather than the ground projection of the Center of Mass to stop the biped.

The above margins assume that any dynamically feasible trajectory through the state space is acceptable. Variants of the above margins can be defined that impose constraints on the trajectories, control system, etc. For example, angular-momentum-regulated versions of these margins such as the **Angular-Momentum-Regulated N-Step Capture Margin**, $M(\mathbf{x}, N, |\dot{H}_{cm}| < \dot{H}_{cm_{max}})$, can be defined as above, but where the Capture Region is computed with the assumption that angular momentum is limited for capturing balance. These margins are useful for determining whether the robot is Capturable without having to rely on drastic measures such as lunging or windmilling.

All of the above margins are in terms of how close the biped is to being able to stop. Determining if a biped is able to stop over a number of steps is appropriate, since if a biped cannot slow down and eventually stop, then its state is likely in the Basin of Fall, except for some metastable border states on the boundary of the Basin of Fall. We conjecture that in the limit as N approaches infinity, N-Step Capturable implies stability as defined by Definition 3. For most practical purposes, if the biped cannot stop in several steps, it is probably close enough to falling to consider it unstable. Therefore, though still not a completely necessary criterion, being say 10-step Capturable should be a suitable stability criterion.

The above stability margins are all in terms of Cartesian distance of various regions on the ground to the Reachable Region. This distance is related to the ability to get the swing leg to the region and also the time before the region moves and is no longer reachable if it isn't stepped to rapidly. One could also define other margins in terms of the area of a given region. These margin would give an indication of the degree of accuracy required in placing the swing leg in the region. For example,

- **N-Step Area-Based Capture Margin.** $M_{area}(\mathbf{x}, N)$: Area of the N-Step Capture Region if the N-Step Capture Region is non-empty. Otherwise, the negative distance from the Unreachable N-Step Capture Region to the Reachable Region.

We have now suggested a number of potential stability margins that can be used for analyzing and controlling bipedal walking. In terms of the desirable characteristics listed in Sect. 3, the margins listed above are sufficient conditions for stability, allow comparisons between different walking algorithms, and are meaningful. While none of them, except for boundary cases requiring extreme computationally complexity, are tight necessary conditions, they are tighter necessary conditions than previously proposed stability margins. These margins are measurable and computable. For $N = 1$, estimates with low computational complexity are easy to derive, as described in the next section. As N increases, the computational requirements likely increase, but may be reasonable for small N.

Deciding which margin is the most desirable and what degree of stability is required depends on tradeoffs between performance and safety. For example, if the biped is walking carefully over stepping stones, then having a large One-Step Area-Based Capture Margin may be desirable. If the biped is a gymnast on a balance beam, then being Angular-Momentum-Regulated One-Step Capturable is important since lunging or windmilling to prevent a fall results in deductions. And if the biped is an Olympic Race Walker, being able to stop in a small number of steps is less important than speed, so a small degree of ∞-Step Capture Margin is all that may be required.

In the next section we will discuss methods for estimating Capture Point locations and some of these stability margins.

8 Estimating Stability Margins

In this section we describe some methods for estimating our proposed stability margins.

8.1 Estimating One-Step Capture Points

We can derive estimates of the location of Capture Points by using inverted pendulum models that approximate walking. Using a constant length inverted pendulum model (Fig. 2, left side) and equating initial and final energy, we have $\frac{1}{2}mv^2 + mgh_0 = mgl$, where m is the mass, v is the velocity of the mass, g is the gravitational constant, h_0 is the initial height above the ground, and l is the virtual leg length. Let $\hat{s}$ be the unit vector parallel to the ground that

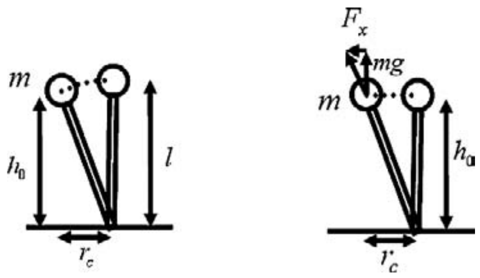


Fig. 2. Simple models for estimating the location of Capture Points

points in the same direction as the Center of Mass velocity. The Capture Point will lie somewhere on the line defined by the projection of the Center of Mass on the ground and $\hat{s}$: $\mathbf{x}_c = r_c\hat{s}$, where $\mathbf{x}_c$ is the location of the Capture Point with respect to the ground projection of the Center of Mass and r_c is the distance from the Center of Mass projection to the Capture Point. Because r_c, l, and h_0 form a right triangle, we have $r_c = \sqrt{l^2 - h_0^2}$. Solving for r_c we get

$$r_c = v\sqrt{\left(\frac{h_0}{g} + \frac{v^2}{4g^2}\right)} \tag{10}$$

This model assumes that the leg length stays constant as the Center of Mass follows an arc, coming to rest above the Capture Point. Another model, referred to as the Linear Inverted Pendulum model [11, 12], assumes that the Center of Mass height stays constant (Fig. 2 – right side). Using this method to compute the location of a Capture Point results in an even simpler equation. In order to counter gravity, the vertical force on the mass must be mg. Because ground reaction forces can only act on the line between the Center of Pressure and the Center of Mass, similar triangles are formed and we have

$$\frac{F_x}{mg} = \frac{x}{h_0} \Longrightarrow F_x = \frac{mg}{h_0}x \tag{11}$$

where F_x is the horizontal force on the mass, and x is the distance from the mass to the Capture Point. Because the mass moves at a constant height, the energy absorbed while moving above the Capture Point will be the integral of the force times the displacement:

$$E = \int_0^{r_c} F\,dx = \frac{mg}{h_0}\int_0^{r_c} x\,dx = \frac{mg}{2h_0}r_c^2 \tag{12}$$

Equating initial and final energies, we have $\frac{1}{2}mv^2 = \frac{mg}{2h_0}r_c^2$. Solving for r_c we get

$$r_c = v\sqrt{\frac{h_0}{g}} \tag{13}$$

Equations (10) and (13) give estimates for two different Capture Points, one based on the Center of Mass following an arc, and one based on it following a straight horizontal line. Because the Center of Mass typically follows a flattened arc in bipedal walking, a more relevant Capture Point for a typical gait will likely fall between those two bounds. For parameter values typical in human walking, these two bounds are within 15% of each other.

The above estimates assumed that the swing leg could instantaneously arrive at the Capture Point. Since it may take some time for the swing leg to get to the Capture Point, we should estimate where the Capture Point will be when the swing leg arrives. If we have an estimate on the time remaining for swing, we can estimate the predicted Capture Point location using the Linear Inverted Pendulum model. Figure 3 illustrates the key variables in estimating the predicted Capture Point location. First we estimate the Center of Mass trajectory. From the previous analysis, $F_x = \frac{mg}{h_0}x$. Therefore, $\ddot{x} = \frac{g}{h_0}x$. Since this is a linear equation, it can be solved in closed form [11, 12]

$$x = C_0 e^{wt} + C_1 e^{-wt},\ \dot{x} = D_0 e^{wt} + D_1 e^{-wt} \tag{14}$$

where $w = \sqrt{\frac{g}{h_0}}$ and

$$\begin{aligned} C_0 &= \frac{1}{2}\left(x_0 + \frac{v_0}{w}\right),\ C_1 = \frac{1}{2}\left(x_0 - \frac{v_0}{w}\right), \\ D_0 &= \frac{1}{2}\left(wx_0 + v_0\right),\ D_1 = \frac{1}{2}\left(-wx_0 + v_0\right) \end{aligned} \tag{15}$$

The equations for y are identical, with the proper substitutions. Given the estimated swing time, we can estimate the location and velocity of the Center of Mass at the end of swing using Equation (15). Using Equation (13) we then can estimate the location of the Capture Point at the end of swing. As the leg is swinging, we can update the predicted Capture Point location and adjust the swing leg trajectory to land in the desired location.

The above estimates of Capture Points were for point mass models. We can estimate a Capture Region using the above equations to compute the center of the region and then models for using angular momentum to determine a region around the center. Suppose we have no limit in how fast we can change our angular momentum ($\dot{H}_{cm}$ is unbounded), but we do have a limit on the angular momentum about the Center of Mass that we have in “reserve”, $H_{reserve}$. Also suppose we have a limit on the duration, $\tau_{reserve}$ that the biped can achieve that angular momentum before running into joint limits and having to “pay it back” by stopping the rotation. We conjecture that once it has been determined to use reserve angular momentum to catch balance, immediately using all that is available as rapidly as possible is the strategy that results in the recovery from the worst conditions.

If we assume that the change in angular momentum is achieved instantaneously through an impulsive torque, then the change in rotational velocity

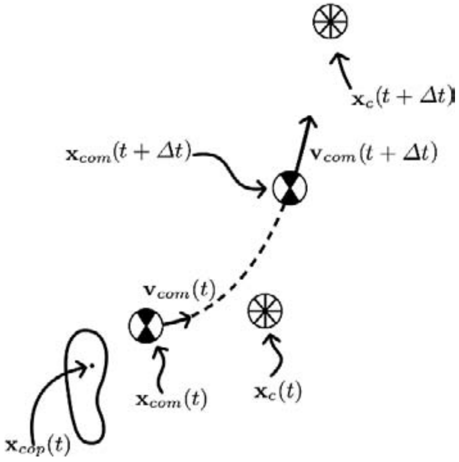

$\mathbf{x}_c(t+\Delta t)$
$\mathbf{x}_{com}(t+\Delta t)$
$\mathbf{v}_{com}(t+\Delta t)$
$\mathbf{v}_{com}(t)$
$\mathbf{x}_{com}(t)$
$\mathbf{x}_c(t)$
$\mathbf{x}_{cop}(t)$

Fig. 3. Evolution of the Center of Mass and a Capture Point from time t to $t+\Delta t$. $\mathbf{x}_{cop}(t)$ is the location of the Center of Pressure; $\mathbf{x}_{com}(t)$ and $\mathbf{v}_{com}(t)$ are the location and velocity of the Center of Mass; and $\mathbf{x}_c(t)$ is the location of the Capture Point

of the Center of Mass about the Center of Pressure can be determined by integrating Equation (6) for an infinitesimal amount of time. The result is

$$\Delta\dot{\theta}_1 = -\frac{1}{ml^2}H_{reserve} \qquad (16)$$

Given the new velocity, we can estimate the Center of Mass trajectory during the next $\tau_{reserve}$ seconds using Equation (14). At the end of this time, the angular momentum must be "paid back" before joint angle limits are violated. We can then repeat the above steps to estimate the final Center of Mass position and velocity.

To estimate the Capture Region, one can use a search, guessing points on the boundary and checking with the above procedure to see if the Center of Mass can be captured over that point with the reserves of angular momentum. After several points on the boundary are determined, then the Capture Region can be estimated with a best-fit curve.

8.2 Estimating N-Step Capture Points

Estimating N-Step Capture Points can be achieved through a brute-force search algorithm. Even with a large state space, the search can be relatively fast if N is small and a discrete step-to-step transition function, S is known,

$$\mathbf{x}_{n+1} = S(\mathbf{x}_n, \mathbf{x}_{step}, \mathbf{P}) \qquad (17)$$

where $\mathbf{x}_n$ is the state of the step n, $\mathbf{x}_{step}$ is the location of the point that the biped steps to, and $\mathbf{P}$ are parameters that govern the motion of the step. The

Linear Inverted Pendulum analysis described previously provides one such example of a discrete step-to-step transition function, with the time of step being the only input parameter.

Since it is likely that, on flat ground, an N-Step Capture Region is a connected, and perhaps even convex area on the ground, a search for the boundary of the Capture Region can be performed when considering the first $\mathbf{x}_{step}$. After the first step, then steps should only be considered that are likely optimal for stopping as quickly as possible. If a One-Step Capture Point exists, then the step should be there. If not, then the step should be as quickly as possible as far as possible toward the Unreachable Capture Region.

8.3 Quick Estimates on the Number of Steps Required for a Stop

We can estimate the approximate number of steps required for a biped to stop by estimating how much the biped can slow down each step using the Linear Inverted Pendulum model. The amount of energy absorbed during deceleration is $\frac{mg}{2h_0}r_{step}$ and the amount of energy returned during acceleration is $\frac{mg}{2h_0}r_{accel}$ where r_{step} is the step length from Center of Mass projection on the ground to the Center of Pressure, and r_{accel} is the distance from the trailing support leg's Center of Pressure to the Center of Mass projection on the ground. If swing can happen quickly enough, r_{accel} can be zero. However, when the swing leg takes long enough to swing that exchange of support cannot be achieved before the Center of Mass passes over the trailing Center of Pressure, then $r_{accel} \neq 0$. r_{accel} can be estimated with (14), given the required swing time. With the net change in energy we can then determine the velocity on the next step,

$$v_{n+1}^2 = v_n^2 - \frac{g}{h_0}r_{step}^2 + \frac{g}{h_0}r_{accel}^2 \tag{18}$$

We can iterate on Equation (18) to estimate the number of steps required to stop the biped. Note that a biped will always be able to slow down if the swing leg is fast enough to have a non-symmetric stance in which the Center of Pressure is in front of the Center of Mass more than it is behind it ($r_{step} > r_{accel}$). When the swing leg takes too long to swing ($r_{step} < r_{accel}$) the biped necessarily continues to speed up and eventually falls down.

If $r_{accel} = 0$, then the number of steps required to stop is $N = \frac{h_0}{gr_{step}^2}v^2$. If $r_{accel} \neq 0$ then $N > \frac{h_0}{gr_{step}^2}v^2$. For typical human parameters at fast walking speeds, $\frac{h_0}{gr_{step}^2}v^2 \approx 1$ meaning a human at top speed usually requires 1 or 2 steps to come to a stop but typically not more. This estimate is easily verified through simple experiments in which one person walks fast while another calls out "stop" at a random time.

Note that a biped with its mass concentrated in its body, and no limit to the torques at its joint, should be able to walk extremely fast along a straight line trajectory. The only limit is how quickly it can swing its leg and transfer support. But with massless legs, it should be able to swing and transfer instantaneously. Along a flattened arc trajectory, the speed would be limited by the requirement of the centripetal acceleration of the Center of Mass being less than the acceleration of gravity. At higher velocities, the ground reaction force falls below zero and the biped becomes airborne. For humans walking at Earth's gravity, this limitation is not the limiting factor in top speed [22]. However, at the gravity of the moon, this limitation is relevant and explains why astronauts preferred to hop on the moon rather than walk.

9 Standing Balance Strategies

Using our proposed stability definitions we can now propose an algorithm for selecting an appropriate strategy to regain balance after being disturbed when standing. The algorithm below is for the reflex phase of balance recovery [1]. Once the robots velocity is captured, than a recovery phase can begin to restore the posture of the robot.

1. if Momentum-Regulated Zero-Step Capturable, then use the Center of Pressure to maintain balance.
2. else if Zero-Step Capturable, then use angular momentum to capture balance by lunging or windmilling.
3. else if Momentum-Regulated One-Step Capturable, then take a step to a Momentum-Regulated Capture Point and use the Center of Pressure to maintain balance after the step.
4. else if One-Step Capturable, then take a step to a Capture Point and use angular momentum to capture balance after the step.
5. else if N-Step Capturable, take a sequence of steps to capture balance.
6. else take a running step and/or fall.

The first two strategies have been well documented with human walking and are often refered to as the "ankle strategy" and the "hip strategy" [15]. We prefer to call them the "Center of Pressure strategy" and the "Angular Momentum Strategy". All of the strategies are easily observable in easily performable balance experiments. Using the stability margins proposed in this paper, it should be possible to hypothesize what amount of disturbance is required to cause a human to switch from one strategy to another one.

10 Control Algorithms

10.1 Control Algorithms for Push Recovery

The balance strategies listed above can be easily implemented in a control algorithm for push recovery. For such an algorithm, it is not important to accurately compute the various stability margins. What is more important is to rapidly determine which balance recovery method to attempt. If a step is required, reaction speed is critical and therefore, determining a good place to step is more appropriate than deliberating on the optimal place to step.

In a simulation study, we developed an algorithm that attempts to stay balanced on one foot. The simulated robot uses modulation of the Center of Pressure in order to keep the Capture Point, which is estimated every control cycle using the Linear Inverted Pendulum model, inside the foot. The commanded Center of Pressure location, $\mathbf{x}_{CoP}$ is computed using a linear controller on the desired Capture Point location, $\mathbf{x}'_{capture}$,

$$\mathbf{x}_{CoP} = \mathbf{x}_{capture} + k(\mathbf{x}'_{capture} - \mathbf{x}_{capture}) \tag{19}$$

where k is the controller gain parameter. If the computed Center of Pressure location falls outside the foot, then it is changed to the nearest point on the foot's edge that lies on the line from the desired Capture Point to the actual Capture Point.

When a significant push occurs, the Capture Point moves outside of the foot. The biped approximates how long it will take to swing the leg to the point and uses that time in the estimate of where the Capture Point will be after swing. If this Capture Point is reachable, then the biped moves its Center of Pressure to the point on the foot nearest the Capture Point to minimize further acceleration and then takes a step to the Capture Point to regain balance. If this Capture Point is unreachable, then the algorithm assumes that if a 2-Step or N-Step Capture Region does exist, that it is near the boundary of the Reachable Region nearest the Unreachable Capture Point. Therefore, the biped will take a step as far as possible towards the Capture Point and then determine a new Capture Point to attempt to step to.

This algorithm works fairly well, validating the Capture Point estimates. When the robot is pushed in such a way that a Capture Point is reachable, the robot does regain balance on that step most of the time. This indicates that even if the estimated Capture Point is not a Capture Point, its margin of error is within the size of the foot.

10.2 Control Algorithms for the Stepping Stones Problem

Now consider the problem of taking steps to desired locations. We refere to this problem as the One-Step Stepping Stones problem, since it is similar to crossing a pond over stones considering only one step at a time.

In a simulation study, we developed an algorithm for one-step stepping stones that uses the Center of Pressure on the support foot to guide the Capture Point out of the foot toward the desired stepping location when told to take a step. The swing leg then swings to the desired stepping location and is loaded once the estimated Capture Point becomes coincident with the desired step location. Figure 4 shows time elapsed snapshots of the simulated biped using this algorithm. Note that since the robot steps to a Capture Point,

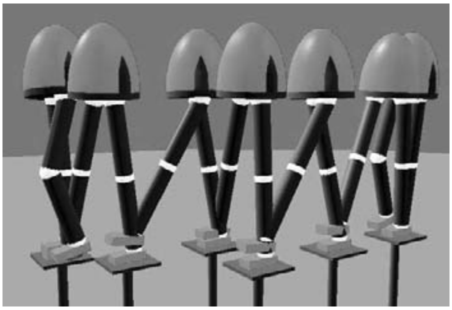


Fig. 4. Time elaspsed snapshots of simulated 12 degree-of-freedom lower-body biped walking over discrete steps while remaining One-Step Capturable. Snapshots are taken at one second intervals

at all times it is One-Step Capturable. If the robot is not told to step to the next desired location, it will stop and balance on one leg.

Since the desired footholds are staggered, the motion is not periodic and hence Poincare return maps could not be used to demonstrate stability of this simulation. Since there is a toe-off phase, the ZMP criterion is violated on every step and is therefore not applicable to this simulation.

Even though the robot is One-Step Capturable throughout the simulation, the motion is dynamic, fairly natural-looking, moderate speed, and includes toe-off and double-support phases. If the robot is told the next step in advance, it does not need to stop over the current step and can continue at a moderate velocity. Therefore, being One-Step Capturable does not imply stopping or even imply slow walking. In this example the small performance degradation is well worth it in order to get the safety margin that being One-Step Capturable provides.

In current work we are investigating how speed can be increased by looking two or three steps ahead and guiding the capture point on a path that curves by the first stepping stone and then on to the second stepping stone as the first step is being taken.

11 Discussion and Future Work

11.1 Probabilistic Stability Margins

In this paper, we have defined stability assuming a deterministic system. However, bipeds should be considered nondeterministic, since ground variations, sensor noise, and external disturbances are impossible to precisely model.

Most stability margins handle nondeterminism by relating to the tolerance to a particular unknown disturbance. This is the case for phase margins and gain margins in traditional linear control and is the case for many margins for bipedal walking, such as the static stability margin and the margins introduced in this paper.

These margins typically give comparative indications of robustness to terrain, noise, and disturbances (the larger the margin, the greater the disturbance that can be tolerated). They sometimes are an indication of the magnitude of the largest single disturbance that can be tolerated. However, they usually do not indicate the probability of instability given a particular disturbance distribution. In a companion paper [28], we explore stochastic stability margins for legged locomotion.

11.2 Proofs of Conjectures

In this paper we have stated several conjectures without proof. While these conjectures seem logical, it would be very useful to prove them since they are relied on for both the theory and the computational implementation in this paper. These conjectures are:

- In the limit as N approaches infinity, N-Step Capturable implies stability as defined by Definition 3.
- The Capture Region remains the same whether or not the constraint is enforced that the Center of Pressure remains at a Capture Point during a step.
- For flat terrain, the Capture Region will be a connected, and perhaps even convex, area on the ground.
- Once it has been determined to use reserve angular momentum to catch balance, immediately using all that is available as rapidly as possible is the strategy that results in the recovery from the worst conditions.

11.3 Generalization

In this paper we considered bipedal walking only. For a multi-legged robot or animal, similar definitions should apply. The difference would be that instead of requiring the foot to cover a Capture Point, the resultant convex hull of the support polygon would need to cover a Capture Point. For running, the difference would be in the Reachable Region of the swing foot. The Reachable Region would be larger, and in addition to being kinematically constrained, it would also be constrained by the leg strength and take-off velocities.

References

[1] Muhammad Abdallah and Ambarish Goswami. A biomechanically motivated two-phase strategy for biped upright balance control. IEEE International Conference on Robotics and Automation (ICRA), 2005.

[2] Michael J. Coleman. *A Stability-Study of a Three-Dimensional Passive-Dynamic Model of Human Gait.* PhD thesis, Cornell University, 1998.

[3] Steven H. Collins, Andy Ruina, Russ Tedrake, and Martijn Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307:1082–1085, February 18 2005.

[4] Steven H. Collins, Martijn Wisse, and Andy Ruina. A three-dimensional passive-dynamic walking robot with two legs and knees. *International Journal of Robotics Research*, 20(7):607–615, July 2001.

[5] E.R. Dunn and R.D. Howe. Foot placement and velocity control in smooth bipedal walking. pages 578–583. IEEE International Conference on Robotics and Automation (ICRA), 1996.

[6] Elena Garcia, Joaquin Estremera, and Pablo Gonzalez de santos. A classification of stability margins for walking robots. In *Proceedings of the International Conference on Climbing and Walking Robots (CLAWAR)*, 2002.

[7] A. Goswami. Postural stability of biped robots and the foot rotation indicator (FRI) point. *International Journal of Robotics Research*, 18(6), 1999.

[8] A. Goswami, B. Espiau, and A. Keramane. Limit cycles and their stability in a passive bipedal gait. pages 246–251. IEEE International Conference on Robotics and Automation (ICRA), 1996.

[9] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka. The development of Honda humanoid robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1321–1326, 1998.

[10] Qiang Huang, Kazuhito Yokoi, Shuuji Kajita, Kenji Kaneko, Hirohiko Arai, Noriho Koyachi, and Kazuou Tanie. Planning walking patterns for a biped robot. *IEEE Transactions on Robotics and Automation*, 17(3):280–289, June 2001.

[11] S. Kajita and K. Tani. Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode. volume 2, pp. 1405–1411. IEEE International Conference on Robotics and Automation (ICRA), 1991.

[12] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kazuhito Yokoi, and Hirohisa Hirukawa. The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. pages 239–246. IEEE International Conference on Intelligent Robots and Systems (IROS), 2001.

[13] T. Kato, A. Takanishi, H. Jishikawa, and I. Kato. The realization of the quasi-dynamic walking by the biped walking machine. In A. Morecki, G. Bianchi, and K. Kedzior, editors, *Fourth Symposium on Theory and Practice of Walking Robots*, pp. 341–351. Warsaw: Polish Scientific Publishers, 1983.

[14] Hassan K. Khalil. *Nonlinear Systems.* Prentice Hall, 3rd edition, December 2001.

[15] A.D. Kuo and F. E. Zajac. Human standing posture: multijoint movement strategies based on biomechanical constraints. *Progress in Brain Research*, 97:349–358, 1993.

[16] Tad McGeer. Passive walking with knees. pages 1640–1645. IEEE International Conference on Robotics and Automation (ICRA), 1990.

[17] H. Miura and I. Shimoyama. Dynamic walk of a biped. *International Journal of Robotics Research*, 3(2):60–74, 1984.

[18] Jun Morimoto and Christopher Atkeson. Minimax differential dynamic programming: An application to robust biped walking. *Advances in Neural Information Processing Systems*, 2002.

[19] Hun ok Lim and Atsuo Takanishi. Waseda biped humanoid robots realizing human-like motion. pages 525–530. IEEE International Workshop on Advanced Motion Control, 2000.

[20] Marko Popovic, Amy Englehart, and Hugh Herr. Angular momentum primitives for human walking: Biomechanics and control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004.

[21] Marko Popovic, Ambarish Goswami, and Hugh Herr. Ground reference points in legged locomotion: Definitions, biological trajectories and control implications. *International Journal of Robotics Research*, 24(12):1013–1032, Dec 2005.

[22] Jerry Pratt. *Exploiting Inherent Robustness and Natural Dynamics in the Control of Bipedal Walking Robots.* PhD thesis, Computer Science Department, Massachusetts Institute of Technology, 2000.

[23] Jerry Pratt and Gill Pratt. Intuitive control of a planar bipedal walking robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1998.

[24] Jerry Pratt, A. Torres, Peter Dilworth, and Gill Pratt. Virtual actuator control. IEEE International Conference on Intelligent Robots and Systems, 1996.

[25] C.L. Shih and W.A. Gruver. Control of a biped robot in the double-support phase. *IEEE Transactions on Systems, Man and Cybernetics*, 22(4):729–735, 1992.

[26] Michael Sipser. *Introduction to the Theory of Computation.* Course Technology, second edition, 2005.

[27] Adam C. Smith and Matthew D. Berkemeier. Passive dynamic quadrupedal walking. In *Proceedings of the 1997 IEEE Conference on Robotics and Automation*, vol. 1, pp. 34–39. IEEE, April 1997.

[28] Russ Tedrake and Jerry E. Pratt. Probabilistic stability in legged systems and the mean first passage time (FPT) stability margin. *In progress*, 2005.

[29] Russ Tedrake, Teresa Weirui Zhang, Ming-fai Fong, and H. Sebastian Seung. Actuating a simple 3D passive dynamic walker. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 5, pp. 4656–4661, New Orleans, LA, April 2004.

[30] Miomir Vukobratovic and Branislav Borovac. Zero-moment point – thirty five years of its life. *International Journal of Humanoid Robotics*, 1(1):157–173, 2004.

[31] E. R. Westervelt, J. W. Grizzle, and D. E. Koditschek. Zero dynamics of underactuated planar biped walkers. *IFAC-2002, Barcelona, Spain*, pp. 1–6, Jul 2002.

[32] Pierre-Brice Wieber. On the stability of walking systems. In *Proceedings of the International Workshop on Humanoid and Human Friendly Robots*, 2002.

[33] Martijn Wisse. *Essentials of dynamic walking; Analysis and design of two-legged robots.* PhD thesis, Technische Universiteit Delft, 2004.

[34] J. Yamaguchi, N. Kinoshita, A. Takanishi, and I. Kato. Development of a dynamic biped walking system for humanoid – development of a biped walking robot adapting to the humans' living floor. pages 232–239. IEEE International Conference on Robotics and Automation (ICRA), 1996.

Nonlinear Model Predictive Control and Sum of Squares Techniques

T. Raff, C. Ebenbauer, R. Findeisen, and F. Allgöwer

Institute for Systems Theory and Automatic Control, University of Stuttgart, Germany
{raff,ce,findeise,allgower}@ist.uni-stuttgart.de

Summary. The paper considers the use of sum of squares techniques in nonlinear model predictive control. To be more precise, sum of squares techniques are used to solve at each sampling instant a finite horizon optimal control problem which arises in nonlinear model predictive control for discrete time polynomial systems. The combination of nonlinear model predictive control and sum of squares techniques is motivated by the successful application of semidefinite programming in linear model predictive control. The advantages and disadvantages of applying sum of squares techniques to nonlinear model predictive control are illustrated on a small example.

1 Introduction

Model predictive control (MPC) is a popular control strategy, especially suitable for the control of multivariable systems subject to input and state constraints [1, 26, 16]. The basic idea of model predictive control is to determine the control input by repeatedly solving at each time instant a finite horizon optimal control problem. In general one distinguishes between linear and nonlinear model predictive control. In linear model predictive control, the optimal control problem is usually solved for a linear system subject to linear input and state constraints and a quadratic performance index. In contrast to linear model predictive control, nonlinear model predictive control (NMPC) allows the consideration of a nonlinear system, possible nonlinear input and state constraints, and a possible nonquadratic performance index in the optimal control problem setup. Model predictive control, especially linear model predictive control, is widely used in industry [24]. One reason why linear model predictive control is more successful than nonlinear model predictive control is that the optimal control problem with linear constraints and a quadratic performance index is a convex quadratic optimization problem for discrete time linear systems. The key advantage in this case is that any local minimum is also a global minimum and that efficient numerical algorithms exist which guarantee to converge to the global minimum [5]. In nonlinear model predictive control, the finite horizon optimal control problem is in general a nonconvex optimization problem. Nonconvex optimization problems are often hard to solve and rather a local minimum than a global minimum is obtained [16].

The purpose of this paper is to solve the nonlinear and in general nonconvex optimization problem that stems from the finite horizon optimal control problem in nonlinear model predictive control for discrete time polynomial systems via sum of squares techniques [15, 20, 25]. Sum of squares techniques are based on the sum of squares decomposition which is an efficient method to check if a multivariate polynomial can be decomposed into a sum of squares since it can be solved via semidefinite programming [5]. The advantages and disadvantages of this combination are discussed on a small example. Furthermore, an introductory review of nonlinear model predictive control, sum of squares techniques, and polynomial optimization is given and an extension of the results presented in [25] to the class of implicit discrete time polynomial systems is introduced.

The remainder of the paper is organized as follows: In Sect. 2, nonlinear model predictive control is reviewed and in Sect. 3 an overview of the sum of squares decomposition of multivariate polynomials and polynomial optimization is given. In Sect. 4, polynomial optimization is applied to nonlinear model predictive control for discrete time polynomial systems. Finally, conclusions are stated in Sect. 5.

1.1 Notation

A polynomial p in $x = [x_1, \ldots, x_n]$ is a finite linear combination of monomials, i.e., $p(x) = \sum_\alpha c_\alpha x^\alpha = \sum_\alpha c_\alpha x_1^{\alpha_1} \ldots x_n^{\alpha_n}$, where $c_\alpha \in \mathbb{R}$ and $\alpha = [\alpha_1, \ldots, \alpha_n], \alpha_i \in \mathbb{N}_0$. The degree of p is defined as $d = \sum_{i=1}^{n} \alpha_i$. The set of all polynomials in $x = [x_1, \ldots, x_n]$ with real coefficients is written as $\mathbb{R}[x]$. A polynomial vector field $f : \mathbb{R}^n \rightarrow \mathbb{R}^n,\ f(x) = [f_1(x), \ldots, f_n(x)]^T$ is a vector field with $f_i(x) \in \mathbb{R}[x]$. A polynomial p is called positive definite, if $p(0) = 0$ and $p(x) > 0, \forall x \in \mathbb{R}^n \setminus \{0\}$ and positive semidefinite if $p(x) \geq 0, \forall x \in \mathbb{R}^n$.

2 Nonlinear Model Predictive Control

In this part of the paper, the basic idea and the mathematical setup of nonlinear model predictive control is introduced in Sect. 2.1 and Sect. 2.2 respectively. Furthermore, Sect. 2.3 provides a review of nonlinear model predictive control schemes that asymptotically stabilize nonlinear systems subject to input and state constraints.

2.1 Basic Idea of Model Predictive Control

The motivation of model predictive control is to stabilize a system by solving the infinite horizon optimal control problem

$$\min_{u_{\cdot|k}} \quad \sum_{i=k}^{\infty} F(x_{i|k}, u_{i|k}) \tag{1a}$$

$$\text{s.t.} \quad x_{i+1|k} = f(x_{i|k}, u_{i|k}), \quad x_{k|k} = x_k\ , \tag{1b}$$

$$u_{i|k} \in \mathcal{U},\ x_{i|k} \in \mathcal{X}\ . \tag{1c}$$

In the infinite horizon optimal control problem, a system (1b) with the initial condition x_k is asymptotically stabilized while a performance index (1a) subject to input and state constraints (1c) is minimized. The performance index (1a) often stems from economical and ecological considerations of the considered control problem. However, problem (1) is in general very difficult to solve. For unconstrained linear systems with a quadratic performance index, the closed solution of the infinite horizon optimal control problem can be determined. In this case, the solution of (1) is the well-known linear quadratic regulator. One possibility to obtain a computationally tractable approximation of (1), i.e., for constrained nonlinear systems with a nonquadratic performance index, is to consider the finite horizon optimal control problem

$$\min_{u_{\cdot|k}} \quad \sum_{i=k}^{k+N-1} F(x_{i|k}, u_{i|k}) \tag{2a}$$

$$\text{s.t.} \quad x_{i+1|k} = f(x_{i|k}, u_{i|k}), \quad x_{k|k} = x_k\ , \tag{2b}$$

$$u_{i|k} \in \mathcal{U},\ x_{i|k} \in \mathcal{X}\ , \tag{2c}$$

where N is the so called prediction horizon. The basic idea of model predictive control is the following one: At each time instant k, the finite horizon optimal control problem (2) is solved on-line with the current measurement of the system state x_k as an initial condition. To obtain a feedback, only the first control action of the optimal control sequence obtained from the finite horizon optimal control problem (2) is applied to the system (2b). These steps, i.e., solving the optimal control problem (2), applying the first control action to the system, and updating the initial condition by the current measurement of the system state, are repeated continuously in model predictive control. Due to online optimization in model predictive control, this control strategy is suitable to control linear as well as nonlinear multivariable systems with input and state constraints. However, the repeated solution of the finite horizon optimal control problem (2) leads to stability problems of the closed loop system [3]. Therefore, different approaches have been developed to achieve closed loop stability. An introductory overview of model predictive control schemes with guaranteed closed loop stability is given in the next sections.

2.2 Setup of Nonlinear Model Predictive Control

Before discussing different nonlinear model predictive control schemes with guaranteed closed loop stability, the mathematical setup is introduced. Consider the stabilization problem for nonlinear discrete time systems described by the difference equations

$$x_{k+1} = f(x_k, u_k) \tag{3}$$

subject to the input and state constraints

$$u_k \in \mathcal{U},\ x_k \in \mathcal{X}\ , \tag{4}$$

where $x_k \in \mathbb{R}^n$ is the state and $u_k \in \mathbb{R}^m$ the control input of the system. Furthermore, assume that the equilibrium point to be stabilized is at the origin, i.e., the vector field f satisfies $f(0,0) = 0$. The set $\mathcal{X}$ is a closed subset of $\mathbb{R}^n$ and the set $\mathcal{U}$ a compact subset of $\mathbb{R}^m$, both containing the origin. Suppose that the full state x_k is available at time instant k. Then the finite horizon optimal control problem that has to be solved at each time instant k in nonlinear model predictive control is given by

$$\begin{aligned}
\min_{\nu} \quad & E(x_{k+N|k}) + \sum_{i=k}^{k+N-1} F(x_{i|k}, u_{i|k}) \\
\text{s.t.} \quad & x_{i+1|k} = f(x_{i|k}, u_{i|k}), \quad x_{k|k} = x_k \\
& u_{i|k} \in \mathcal{U}, \quad i \in [k, k+N-1] \\
& x_{i|k} \in \mathcal{X}, \quad i \in [k, k+N-1] \\
& x_{k+N|k} \in \mathcal{E}\ ,
\end{aligned} \tag{5}$$

where

$$J(\nu, x_k) = E(x_{k+N|k}) + \sum_{i=k}^{k+N-1} F(x_{i|k}, u_{i|k}) \tag{6}$$

is the cost function, ν the control sequence $\nu = [u_{k|k}, \ldots, u_{k+N-1|k}]^T$, N the length of the prediction and the control horizon, and $x_{i|k}$ the predicted system state at time instant i that is obtained by applying the control input sequence $u_{k|k}, \ldots, u_{k+i-1|k}$ to the system (3) from the initial condition x_k. Furthermore, the stage cost $F : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}$ is a positive definite function and satisfies $F(0,0) = 0$. The terminal cost $E : \mathbb{R}^n \rightarrow \mathbb{R}$ satisfies $E(0) = 0$ and $\mathcal{E}$ is the terminal region. The optimal solution to the finite horizon optimal control problem (5) is denoted by $\nu^\star = [u^\star_{k|k}, \ldots, u^\star_{k+N-1|k}]^T$. The actual control action of the nonlinear model predictive controller is the first element of $\nu^\star$, i.e., $u_k = u^\star_{k|k}$. The finite horizon optimal control problem (5) differs from the optimal control problem (2) because of the terminal cost E and the terminal region $\mathcal{E}$. In the next section a summary is given on how to choose the terminal cost E and the terminal region $\mathcal{E}$ in order to achieve closed loop stability.

2.3 Stability of Nonlinear Model Predictive Control

A wide variety of approaches have been developed to achieve asymptotic closed loop stability [1, 9, 16]. All these approaches are based, implicitly or explicitly, on three ingredients: A terminal cost E, a terminal region $\mathcal{E}$, and a locally stabilizing control law $u_k = \varphi(x_k)$. Many of these approaches are covered by the following theorem [16]:

Theorem 1. *The closed loop system is asymptotically stable if the optimal control problem (5) is feasible at the first time instant and the following assumptions are satisfied for a terminal cost E, a terminal region $\mathcal{E}$, and a locally stabilizing control law $u_k = \varphi(x_k)$:*

[A1] $E(x_k) > 0,\ \forall x_k \in \mathbb{R}^n \setminus \{0\}$
[A2] $\mathcal{E} \subseteq \mathcal{X},\ 0 \in \mathcal{E}$
[A3] $\varphi(x_k) \in \mathcal{U},\ \forall x_k \in \mathcal{E}$
[A4] $f(x_k, \varphi(x_k)) \in \mathcal{E},\ \forall x_k \in \mathcal{E}$
[A5] $E(f(x_k, \varphi(x_k))) - E(x_k) \leq -F(x_k, \varphi(x_k)), \forall x_k \in \mathcal{E}.$

A proof of Theorem 1 can be found, e.g., in [16]. In the following, a number of approaches are described based on Theorem 1 which ensure closed loop stability.

Zero Terminal State Constraint. An approach to guarantee closed loop stability is to choose $E(x_k) = 0$, $\mathcal{E} = \{0\}$, and $\varphi(x_k) = 0$ [14], i.e., one obtains the so called zero terminal state constraint

$$x_{k+N|k} = 0\ . \tag{7}$$

If the optimal control problem (5) is feasible at the first time instant, all assumptions of Theorem 1 are satisfied and closed loop stability is achieved. The advantage of this method is the conceptual simplicity. The disadvantage of the zero terminal constraint is that it makes the resulting optimal control problem (5) in general difficult to solve and a long prediction horizon is needed to obtain feasibility at the first time instant.

Terminal Region. Another approach, called dual mode model predictive control [17], is to choose $E(x_k) = 0$, $\mathcal{E} = \{x_k \in \mathbb{R}^n | x_k^T P x_k \leq \alpha\}$, and $u_k = \varphi(x_k) = Kx_k$ as a locally stabilizing linear state feedback. The positive definite matrix P of the terminal region $\mathcal{E}$ and the linear feedback matrix K are determined by a particular procedure in such a way that the assumptions of Theorem 1 are satisfied [17]. The dual mode refers to two different controllers that are applied in different regions of the state space depending on the state being inside or outside the terminal region $\mathcal{E}$. If the state is outside the terminal region $\mathcal{E}$, the model predictive controller is used to steer the state inside the terminal region $\mathcal{E}$. If the state is inside the terminal region $\mathcal{E}$, the linear feedback $u_k = Kx_k$ is applied. Hence, closed loop stability is achieved by switching between the model predictive controller and the linear controller. From a computational point of view, this approach is more attractive than the zero terminal state constraint (7) since the terminal state is only required to lie in a region around the origin. A drawback of this approach is the switching between the model predictive controller and the linear controller which makes the implementation complicated.

Terminal Cost and Terminal Region. In the so-called quasi infinite horizon model predictive control scheme [6], the terminal cost E is chosen as $E(x_k) = x_k^T P x_k$, the terminal region $\mathcal{E}$ as $\mathcal{E} = \{x_k \in \mathbb{R}^n | x_k^T P x_k \leq \alpha\}$, and $u_k = \varphi(x_k) = K x_k$ as a locally stabilizing linear state feedback. The terminal cost E, the terminal region $\mathcal{E}$, and the linear feedback $u_k = \varphi(x_k) = K x_k$ are calculated off-line by a procedure described in [6] so that the assumptions of Theorem 1 are satisfied. The underlying idea of the quasi infinite horizon model predictive control scheme is to approximate the infinite horizon optimal control problem (1) by the terminal cost E. Due to this approximation it can be shown that this scheme has a better performance in comparison with the zero terminal constraint model predictive controller or the dual model model predictive controller. Furthermore, the terminal region $\mathcal{E}$ of the quasi infinite horizon model predictive controller is better suited for computational purposes than a zero terminal state constraint.

Note that not all nonlinear model predictive control schemes with guaranteed closed loop stability are reviewed. For a detailed and more rigorous treatment of model predictive control schemes with guaranteed stability for discrete time and continuous time systems, see e.g. [16] and the references quoted therein. In summary, model predictive control is suitable to control linear and nonlinear multivariable systems subject to state and input constraints. Furthermore, closed loop stability can be guaranteed by adding, e.g., a terminal cost E or a terminal region $\mathcal{E}$ to the finite horizon optimal control problem (2). The price to pay in model predictive control is that one has to solve at each time instant a finite horizon optimal control problem, i.e., in general a nonlinear and possible nonconvex optimization problem. Typically, specially tailored optimizers [2, 8] are used to solve the optimization problem in nonlinear model predictive control which enable to use this control strategy for practical control problems. For discrete time polynomial systems, the resulting optimization problem is a polynomial optimization problem if the objective function and the constraints are polynomial functions. One possible solution approach to solve polynomial optimization problems is to translate the original optimization problem to a semidefinite program via the theory of moments [15] or the theory of sum of squares polynomials [20]. This solution approach is in the following applied to solve the finite horizon optimal control problem in nonlinear model predictive control for discrete time polynomial systems. In the next section, a review of sum of squares techniques and polynomial optimization is given and in Sect. 4 polynomial optimization is applied to nonlinear model predictive control.

3 Sum of Squares Techniques

In this section, an introduction to the sum of squares decomposition of multivariate polynomials and polynomial optimization is given. The background of the sum of squares decomposition is presented in Sect. 3.1. Furthermore, the concept of polynomial optimization is introduced in Sect. 3.2. Finally, Sect. 3.3 gives an example to polynomial optimization.

3.1 Sum of Squares

The sum of squares decomposition is a method to check if a multivariate polynomial can be decomposed into a sum of squares. Since a sum of squares polynomial is always positive semidefinite, one can use the sum of squares decomposition as a sufficient condition to check if a multivariate polynomial is positive semidefinite. The question whether a polynomial in several variables can be written as sum of squares has a long history and goes back to Hilbert [12]. From a computational point of view, one can show that the general problem of checking if a polynomial is positive semidefinite is a NP-hard problem, i.e., the problem is hard to solve. Today, questions about positive polynomials are investigated in the field of real algebraic geometry. In the following, some facts about sum of squares polynomials and some computational aspects of the sum of squares decomposition are summarized. More details about sum of squares decomposition and its application in control theory can be found in [19, 29, 22].

Definition 1. *A polynomial p of degree $d = d_1 + ... + d_n$, $d_i \in \mathbb{N}_0$ with real coefficients c_d, i.e., $p(x) = \sum_{d_1+...+d_n \leq d} c_d x_1^{d_1} ... x_n^{d_n}$, is a* sum of squares (SOS) *if there exists a finite number of polynomials p_i such that p can be written as*

$$p(x) = \sum_i p_i^2(x) \ . \tag{8}$$

Hilbert proved that not every positive semidefinite polynomial can be written as a sum of squares. However, Artin's result in 1927, which was the answer to Hilbert's 17th problem posed in 1900, states that every positive semidefinite polynomial is a sum of squares of rational functions [22]. The next theorem, which is central for computational purposes, gives the answer to the question, when a polynomial is a sum of squares [7]:

Theorem 2. *A polynomial p of degree $2d$ has a sum of squares decomposition if and only if there exists a positive semidefinite matrix Q such that*

$$p(x) = m^T(x) Q m(x) \ , \tag{9}$$

where m is the vector of all monomials in $x_1, \ldots, x_n$ of degree less or equal to d, i.e., $m(x) = [1, x_1, x_2, \ldots, x_n, x_1 x_2, \ldots, x_n^d]^T$. There exist $\binom{n+d}{n}$ such monomials.

This representation theorem, which is based on the so called "Gram matrix" method, tells that all sum of squares polynomials can be parameterized by the set (convex cone) of positive semidefinite matrices. The next theorem [27], which basically extends Artin's result, gives a link between sum of squares polynomials and positive definite polynomials.

Theorem 3. *If p is a positive definite polynomial, then $\|x\|^r p(x)$ is a sum of squares polynomial for a sufficiently large integer r.*

This theorem is helpful if positive definiteness of a polynomial must be checked with the help of a sum of squares decomposition. Furthermore, it is often useful and sufficient to check positive semidefiniteness of a polynomial p on a subset in $\mathbb{R}^n$ constrained by the set of points $\{x \in \mathbb{R}^n : \; p_1(x) \geq 0\}$, where p_1 is a polynomial. In the theory of quadratic forms, this is usually done via Finsler's lemma or the S-procedure [4]. An analogous type of argument does the same for polynomials (cf. e.g. [18]):

Lemma 1. *Let p_1 be a polynomial. A polynomial p is positive semidefinite on the set $\{x \in \mathbb{R}^n : \; p_1(x) \geq 0\}$, if there exists a positive semidefinite polynomial q such that $p(x) - q(x)p_1(x)$ is positive semidefinite.*

It should be mentioned that there also exist representation theorems for (strictly) positive polynomials on compact semialgebraic sets, i.e., compact sets defined by polynomial inequalities [13, 22]. These representation theorems [13, 22] play a decisive role in the polynomial optimization theory [15, 20] which is outlined in Sect. 3.2.
However, what makes the above results especially appealing from a computational point of view is the fact that these results can be efficiently and reliably solved on a computer, namely, with the help of semidefinite programming. In particular, in [19], the gap between the Gram matrix method and semidefinite programming was bridged by showing that the existence of a sum of squares decomposition of a polynomial can be decided by solving a semidefinite program. Semidefinite programming, which may be viewed as a generalization of linear programming, is a convex optimization problem with a linear objective function under linear matrix inequality constraints. Semidefinite programs have very nice properties in optimization theory and practice [30, 5] and are very successful in linear control design and combinatorial optimization [4]. Combining the sum of squares decomposition and semidefinite programming, solutions for the following system of affine polynomial inequalities can be obtained [19, 29]:

$$\begin{aligned} \sum_{i=1}^{N_j} q_i(x) p_{ij}(x) &\geq r_j(x) \\ \sum_{i=1}^{N_j} \tilde{q}_i(x) \tilde{p}_{ij}(x) &= \tilde{r}_j(x) \; , \end{aligned} \tag{10}$$

where $j = 1, \ldots, M$, p_{ij}, $\tilde{p}_{ij}$, r_j, and $\tilde{r}_j$ are given polynomials and q_i and $\tilde{q}_i$ are unknown (to be determined) polynomials, for which the degrees have to be specified. This system of polynomial equations and inequalities can be solved by a sum of squares relaxation, namely by solving

$$\begin{aligned} &\sum_{i=1}^{N_j} q_i(x) p_{ij}(x) - r_j(x) = \text{SOS} \\ &\sum_{i=1}^{N_j} \tilde{q}_i(x) \tilde{p}_{ij}(x) - \tilde{r}_j(x) = 0 \ , \end{aligned} \tag{11}$$

where SOS stands for an arbitrary sum of squares polynomial. If one would like to enforce positive definiteness instead of semidefiniteness, e.g., $p(x)q(x) > 0$, then this can be done for example via $p(x)q(x) - r(x) = \text{SOS}$, where r is a positive definite polynomial, e.g. $r(x) = \varepsilon x^T x$, $\varepsilon > 0$. Note, that for example with the freely available Matlab toolbox SOSTOOLS [21] the equations and inequalities of type (11) can be easily solved.

In summary, it should be noted that the sum of squares decomposition is a relaxed condition, i.e., a sufficient condition for positive semidefiniteness. Hence there may exist solutions to the above polynomial system of equations and inequalities, although no solution can be found by using the sum of squares decomposition. Furthermore, the degrees of the unknown polynomials q_{ij} and $\tilde{q}_{ij}$ must be specified a priori. In the next section, the sum of squares techniques are applied to solve polynomial optimization problems.

3.2 Polynomial Optimization

In recent years, major advances in polynomial optimization have been made. Two approaches based on semidefinite programming have been developed. One approach uses the theory of moments [15] and the other approach the theory of sum of squares [20, 28]. These methods are strongly related because the theory of moments is, in some sense, dual to the theory of nonnegative polynomials. Both approaches build a sequence of semidefinite programs of increasing size. The optima of these semidefinite programs converge monotically to the global optimum for the original nonconvex polynomial optimization problem. In the following, the basic idea of polynomial optimization via the sum of squares approach is outlined. Consider the optimization problem

$$\min_{x \in K} \quad p_0(x) \ , \tag{12}$$

where $p_0(x) \in \mathbb{R}[x]$ and $K = \{x \in \mathbb{R}^n : p_i(x) \leq 0, \ i = 1, \ldots, m\}$ is a compact set described by multivariate polynomial inequalities, i.e., $p_i(x) \in \mathbb{R}[x], \ i = 1, \ldots, m$. Furthermore, some p_i satisfies, on its own, the condition that $\{x \in \mathbb{R}^n : p_i(x) \leq 0\}$ is compact. Under these assumptions, a representation theorem [23] can be established as follows:

Theorem 4. *If the assumptions as stated above are satisfied, then any polynomial p_0 strictly positive on the set K can be written as*

$$p_0(x) = q_0(x) - \sum_{i=1}^{m} q_i(x)p_i(x) \ , \tag{13}$$

where q_i, $i = 0, \ldots, m$, are all sum of squares polynomials.

Unfortunately, the degrees of the sum of squares polynomials q_i, $i = 0, \ldots, m$, are unknown and must be chosen sufficiently high. However, Theorem 4 plays a central role since it allows to consider the following equivalent polynomial optimization problem [20, 28]

$$\begin{aligned} \max \quad & t \\ \text{s.t.} \quad & p_0(x) - t + \sum_{i=1}^{m} q_i(x)p_i(x) \geq 0, \quad \forall x \end{aligned} \tag{14}$$

where p_i, $i = 0, \ldots, m$, are given polynomials and q_i, $i = 1, \ldots, m$, are unknown sum of squares polynomials. The basic idea of the optimization problem (14) is shown in Fig. 1 for an unconstrained polynomial optimization problem in one variable. Instead of minimizing the polynomial p_0 directly, t is maximized while satisfying the inequality in (14).

Since the unknowns t, q_i, $i = 1, .., m$, enter affine in the polynomial optimization problem (14), it can be solved using the sum of squares decomposition of multivariate polynomials introduced in the previous subsection. If the degrees of the sum of squares polynomials q_i, $i = 1, \ldots, m$, have not been chosen sufficiently high, then t is a lower bound for $p^\star$, where $p^\star$ is the global optimum of the original polynomial optimization problem (12). However, increasing the degrees of the sum of squares polynomials q_i, $i = 1, \ldots, m$, the lower bound t converges arbitrarily close to $p^\star$. The price to pay for improving the lower bound t by increasing the degrees of the sum of squares polynomials is that the number of decision variables and the size of the semidefinite program of the polynomial optimization problem (12) grow exponentially. The theory of moments [15] relates to the theory of nonnegative polynomials [20, 28] by considering the dual problem of (14). It can be shown that the dual problem of (14) is the semidefinite program obtained by solving the

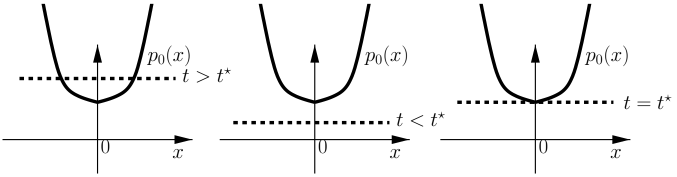


Fig. 1. Basic idea of polynomial optimization

original polynomial optimization problem (12) using the theory of moments. What remains to be discussed is the detection of the global optimum and the extraction of the optimal solution. In [11], a rank condition is proposed to detect the global optimum. The extraction of the optimal solution is based on a Cholesky decomposition and on an eigenvalue problem. For a more thorough discussion of polynomial optimization, consult [11, 15, 20] and the references quoted therein. Note, that with the freely available Matlab software toolbox GloptiPoly [10] polynomial optimization problems of type (12) can be easily solved. In the following, a small example is given to illustrate the possibilities of polynomial optimization before applying polynomial optimization to nonlinear model predictive control in Sect. 4.

3.3 Example

Consider the polynomial optimization problem

$$\begin{aligned} \min_{x_1,x_2} \quad & 2x_1^4 - 4x_1^2 + x_1 + x_2^2 + 4 \\ \text{s.t.} \quad & -\frac{3}{2} \le x_1 \le \frac{3}{2}, \ -\frac{3}{2} \le x_2 \le \frac{3}{2} , \end{aligned}$$

where the objective function $p_0(x) = 2x_1^4 - 4x_1^2 + x_1 + x_2^2 + 4$ has one local and one global minimum as shown in Fig. 2. Using GloptiPoly [10] to solve the semidefinite programming relaxations described above, the computed lower bound is $t = 0.9705$ and the extracted solutions are $x_1 = -1.0575$ and $x_2 = 0$. In this example, the lower bound $t = 0.9705$ is equal to the global minimum of the original problem. This example illustrates that the global minimum of a

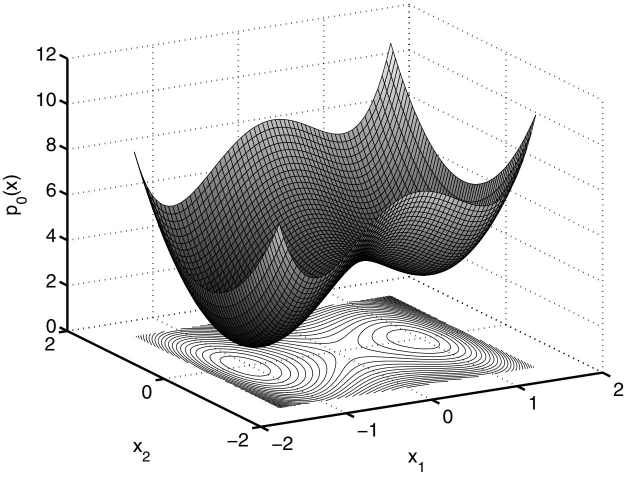


Fig. 2. Nonconvex polynomial objective function p_0

nonconvex polynomial optimization problem can be efficiently computed via semidefinite programming.

4 Sum of Squares in Nonlinear Model Predictive Control

In this section, sum of squares techniques and polynomial optimization are applied to nonlinear model predictive control for the class of discrete time polynomial systems. In Sect. 4.1 and Sect. 4.2, it is shown that the finite horizon optimal control problem in nonlinear model predictive control can be formulated as a polynomial optimization problem and can be solved with the polynomial optimization techniques presented in the Sect. 3. Furthermore, the applicability of polynomial optimization in nonlinear model predictive control is shown in Sect. 4.3.

4.1 Polynomial Optimization Formulation of NMPC

In the following it is shown that the finite horizon optimal control problem (5) in nonlinear model predictive control can be formulated as a polynomial optimization problem for the class of discrete time polynomial systems. In the following it is assumed that the system (3) is polynomial, i.e., $f : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ is a polynomial vector field and $f(0,0) = 0$. Furthermore, the stage cost $F : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}$ is assumed to be a polynomial map that satisfies $F(0,0) = 0$. The terminal penalty term $E : \mathbb{R}^n \rightarrow \mathbb{R}$ is assumed to be also a polynomial map with $E(0) = 0$ and the terminal region $\mathcal{E}$ can be described by a polynomial inequality. In order to formulate the finite horizon optimal control problem (5) as a polynomial optimization problem, the predicted states $x_{k+1|k}, \ldots, x_{k+N|k}$ must be computed from the future inputs $u_{k|k}, \ldots, u_{k+N-1|k}$ and the initial condition x_k. To write (5) in a compact form, the vector

$$\xi = \Big[x_{k+1|k},\, x_{k+2|k},\, \cdots, x_{k+N|k}\Big]^T \tag{15}$$

for the predicted states is introduced. Iterating system (3) yields

$$\xi = H(\nu, x_k) \tag{16}$$

with

$$H(\nu, x_k) = \begin{bmatrix} h_1(u_{k|k}, x_k) \\ h_2(u_{k|k}, u_{k+1|k}, x_k) \\ \vdots \\ h_N(u_{k|k}, \ldots, u_{k+N-1|k}, x_k) \end{bmatrix} = \begin{bmatrix} f(x_k, u_{k|k}) \\ f(x_{k+1|k}, u_{k+1|k}) \\ \vdots \\ f(x_{k+N-1|k}, u_{k+N-1|k}) \end{bmatrix}$$

and $\nu = [u_{k|k}, \ldots, u_{k+N-1|k}]^T$. Note, that $h_i,\ i = 1, \ldots, N$, are multivariate polynomial functions in ν for a fixed initial condition x_k. Using (16), the cost function (6) can be rewritten as

$$\begin{aligned} J(\nu, x_k) &= F(x_k, u_{k|k}) \\ &+ \ldots \\ &+ F(h_{N-1}(u_{k|k}, \ldots, u_{k+N-2|k}, x_k), u_{k+N-1|k}) \\ &+ E(h_N(u_{k|k}, \ldots, u_{k+N-1|k}, x_k)) \\ &= p_0(\nu) \ . \end{aligned} \tag{17}$$

Hence, $J(\nu, x_k)$ is a multivariate polynomial function in ν for a fixed state x_k, i.e., $p_0(\nu) \in \mathbb{R}[\nu]$. Next, the constraints of the optimization problem (5) must be expressed in terms of polynomial inequalities. For simplicity, suppose that the constraints are given as

$$\begin{aligned} \mathcal{X} &:= \{x_k \in \mathbb{R}^n | x_{min} \leq x_k \leq x_{max}\} \\ \mathcal{U} &:= \{u_k \in \mathbb{R}^m | u_{min} \leq u_k \leq u_{max}\} \ , \end{aligned} \tag{18}$$

where $x_{min}, x_{max}, u_{min}, u_{max}$ are constant vectors. Therefore, the constraints can be written as

$$\begin{aligned} L \begin{bmatrix} H(\nu, x_k) \\ 1 \end{bmatrix} &\leq 0 \ , \\ M \begin{bmatrix} \nu \\ 1 \end{bmatrix} &\leq 0 \ , \end{aligned} \tag{19}$$

where

$$L = \begin{bmatrix} I_1 & 0 & \ldots & 0 & -x_{max} \\ -I_1 & 0 & \ldots & 0 & x_{min} \\ 0 & I_1 & \ldots & 0 & -x_{max} \\ 0 & -I_1 & \ldots & 0 & x_{min} \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 0 & \ldots & I_1 & -x_{max} \\ 0 & 0 & \ldots & -I_1 & x_{min} \end{bmatrix} , \quad M = \begin{bmatrix} I_2 & 0 & \ldots & 0 & -u_{max} \\ -I_2 & 0 & \ldots & 0 & u_{min} \\ 0 & I_2 & \ldots & 0 & -u_{max} \\ 0 & -I_2 & \ldots & 0 & u_{min} \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 0 & \ldots & I_2 & -u_{max} \\ 0 & 0 & \ldots & -I_2 & u_{min} \end{bmatrix} ,$$

and I_1 and I_2 are identity matrices of appropriate dimensions. Therefore, (19) defines $2(n+m)N$ polynomial inequality constraints $p_i(\nu) \in \mathbb{R}[\nu],\ i = 1, \ldots, 2(n+m)N$, for a fixed state x_k. Combining (17), (19), and the fact that the terminal region $\mathcal{E}$ is described by an additional polynomial inequality, e.g.,

$$p_{2(n+m)N+1}(\nu) \leq 0 \ , \tag{20}$$

one obtains the following theorem [25]:

Theorem 5. *The finite horizon optimal control problem (5) can be formulated as a polynomial optimization problem of the form*

$$\min_{\nu \in K} \quad p_0(\nu) \tag{21}$$

for discrete time polynomial systems, if $K = \{\nu \in \mathbb{R}^{m \cdot N} : p_i(\nu) \leq 0,\ i = 1, \dots, 2(n+m)N+1\}$ *is a compact set described by the multivariate polynomial inequalities* $p_i(\nu) \in \mathbb{R}[\nu],\ i = 1, \dots, 2(n+m)N+1$.

4.2 Polynomial Optimization Formulation of NMPC for Implicit Discrete Time Polynomial Systems

In the following it is briefly outlined that polynomial optimization can also be applied to solve the finite horizon optimal control problem in nonlinear model predictive control for implicit discrete time polynomial systems, i.e., for systems of the form

$$f(x_{k+1}, x_k, u_k) = 0 \; . \tag{22}$$

The basic idea in this case is to introduce the vector

$$W(\xi, \nu) = \begin{bmatrix} f(x_{k+1|k}, x_k, u_{k|k}) \\ f(x_{k+2|k}, x_{k+1|k}, u_{k+1|k}) \\ f(x_{k+3|k}, x_{k+2|k}, u_{k+2|k}) \\ \vdots \\ f(x_{k+N|k}, x_{k+N-1|k}, u_{k+N-1|k}) \end{bmatrix} , \tag{23}$$

where the variables $\xi = \left[x_{k+1|k},\, x_{k+2|k},\, \cdots, x_{k+N|k}\right]^T$ are considered now as independent variables. Therefore, the cost function (6) can be rewritten as

$$\begin{aligned} J(\nu, x_k) &= F(x_k, u_{k|k}) \\ &\quad + \dots \\ &\quad + F(x_{k+N-1|k}, u_{k+N-1|k}) \\ &\quad + E(x_{k+N|k}) \\ &= p_0(\xi, \nu) \; , \end{aligned} \tag{24}$$

where $p_0(\xi, \nu)$ is a multivariate polynomial function in the variables ξ and ν for a fixed state x_k, i.e., $p_0(\xi, \nu) \in \mathbb{R}[\xi, \nu]$. In the same way one can proceed writing the state and input constraints and the terminal region as polynomial inequalities in ξ and ν. Therefore, one has $2(n+m)N+1$ polynomial inequalities of the form

$$p_i(\xi, \nu) \leq 0, \quad i = 1, \dots, 2(n+m)N+1 \tag{25}$$

with $p_i(\xi,\nu) \in \mathbb{R}[\xi,\nu],\ i = 1,\dots,2(n+m)N+1$, for a fixed state x_k. Finally, one has to restore the relation between $x_{k+i|k}$ and $x_{k+j|k}$ by introducing the constraint

$$\|W(\xi,\nu)\|^2 = 0 \ , \tag{26}$$

where $\|x\| = (\sum_{i=1}^{n} x_i^2)^{\frac{1}{2}}$ denotes the Euclidian norm of $x \in \mathbb{R}^n$. Note, that equality (26) can be rewritten as two polynomial inequalities of the form

$$\begin{aligned} p_{2(n+m)N+2}(\xi,\nu) &= \|W(\xi,\nu)\|^2 \leq 0 \\ p_{2(n+m)N+3}(\xi,\nu) &= \|W(\xi,\nu)\|^2 \geq 0 \ . \end{aligned} \tag{27}$$

Therefore, the finite horizon optimal control problem for implicit discrete time polynomial systems can be solved via the polynomial optimization techniques presented in Sect. 3. This result is summarized in the next theorem.

Theorem 6. *The finite horizon optimal control problem (5) can be formulated as a polynomial optimization problem of the form*

$$\min_{(\xi,\nu)\in K} \quad p_0(\xi,\nu) \tag{28}$$

for implicit discrete time polynomial systems of the form (22), if $K = \{(\xi,\nu) \in \mathbb{R}^{n\cdot N} \times \mathbb{R}^{m\cdot N} : p_i(\xi,\nu) \leq 0,\ i = 1,\dots,2(n+m)N+3\}$ *is a compact set described by the multivariate polynomial inequalities* $p_i(\xi,\nu) \in \mathbb{R}[\xi,\nu],\ i = 1,\dots,2(n+m)N+3$.

Due to Theorem 5 and Theorem 6, the polynomial optimization techniques presented in Sect. 3 can be applied to nonlinear model predictive control for discrete time polynomial systems. In the next section, the applicability of sum of squares techniques in nonlinear model predictive control is demonstrated on a small example.

4.3 Example

In the following, polynomial optimization is used to solve the finite horizon optimal control problem in nonlinear model predictive control for discrete time polynomial systems as stated in Theorem 5. Consider the stabilization problem of the second order polynomial system [6] described by the difference equations

$$x_{k+1} = f(x_k, u_k) \ ,$$

where

$$x_k = [x_k^{(1)}, x_k^{(2)}]^T ,$$

$$f(x_k,u_k)=\begin{bmatrix} x_k^{(1)}+0.1x_k^{(2)}+0.05(1+x_k^{(1)})u_k \\ 0.1x_k^{(1)}+x_k^{(2)}+0.05(1-4x_k^{(2)})u_k \end{bmatrix} .$$

This system is unstable and its linearization around the origin is stabilizable but not controllable. Furthermore, the input has to satisfy the constraint $\mathcal{U} := \{u_k \in \mathbb{R} | -2 \le u_k \le 2\}$. To stabilize the system, a nonlinear model predictive controller is designed based on the finite horizon optimal control problem (5) with a quadratic stage cost

$$F(x_k,u_k) = x_k^T \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} x_k + 0.1u_k^2 ,$$

a prediction and control horizon of the length $N = 4$, a terminal penalty term $E(x_k) = x_k^T P x_k$, and a terminal region $\mathcal{E} = \{x_k \in \mathbb{R}^2 | x_k^T P x_k \le \alpha\}$. Using the quasi infinite horizon method [1, 6] to ensure stability of the closed loop system, the matrix P and the parameter α are chosen as

$$P = \begin{bmatrix} 4.5 & 1.8 \\ 1.8 & 4.5 \end{bmatrix} , \quad \alpha = 0.3 .$$

The resulting polynomial MPC finite horizon optimal control problem was solved using GloptiPoly [10]. Figures 3 and 4 show the initial response of the closed loop system with an initial condition $x_0 = [-0.3, -0.9]^T$. It can be seen that the closed loop trajectories converge to the origin without violating the input constraints. Hence, this example demonstrates that in principle the polynomial optimization techniques based on semidefinite programming

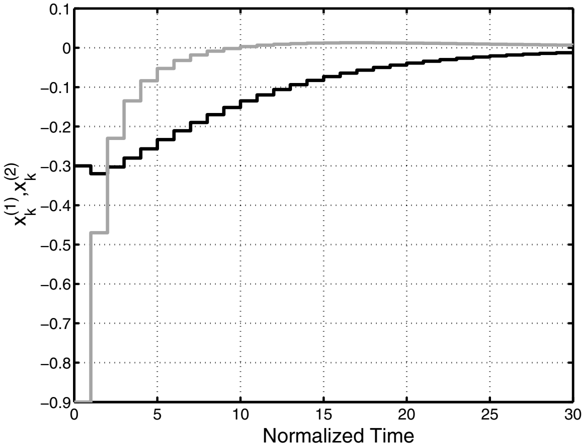


Fig. 3. Closed loop state trajectories $x_k^{(1)}$ (*black*) and $x_k^{(2)}$ (*gray*)

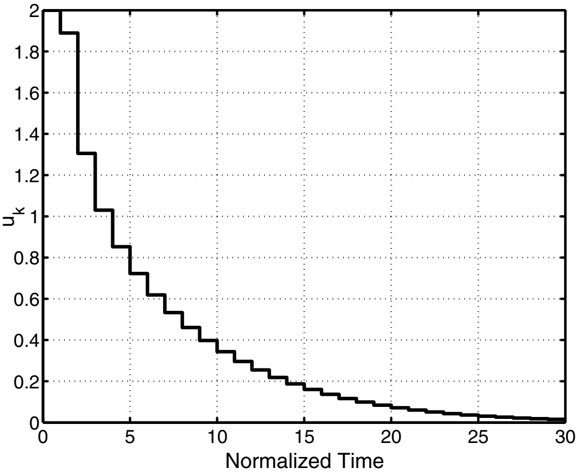


Fig. 4. Control input u_k

can be used to solve the finite horizon optimal control problem in nonlinear model predictive control for discrete time polynomial systems. Finally, some computational properties of the resulting polynomial optimization problem are summarized in Table 1. This table shows some optimization parameters of the polynomial optimization problem obtained using Matlab 6.5 and GloptiPoly on a Linux PC with a 3 GHz Intel Pentium 4 processor and 1.5 GB RAM. From Table 1 it can be seen that for a prediction horizon N greater than $N = 4$ the size of the semidefinite program (SDP) increases strongly. The reason for this increase is that the problem size of polynomial optimization problems growths exponentially with the number of variables [11, 15]. Therefore, these techniques can be only applied to small and medium size problems. In order to make these techniques applicable to larger polynomial optimization problems and thus to model predictive control for real world application, it is necessary to develop numerical methods which exploit the system structure, e.g., sparsity and symmetry of polynomial optimization problems.

Summarizing, the key advantage of applying sum of squares techniques in nonlinear model predictive control is that a global minimum of a possible nonconvex finite horizon optimal control problem is obtained via semidefinite programming. This property can avoid unacceptable performance degradation

Table 1. Optimization parameters

Prediction horizon	$N = 3$	$N = 4$	$N = 5$
Number of variables	3	4	5
SDP decision variables	455	495	3003
Relaxation	6	4	5

in nonlinear model predictive control. The main disadvantage of this approach is that it is computationally very demanding using currently available algorithms and therefore only applicable to small and medium size problems.

5 Conclusion

In this paper it was shown that sum of squares techniques and polynomial optimization can be applied to solve finite horizon optimal control problems in model predictive control for the class of discrete time polynomial control systems. Polynomial optimization problems can be solved via semidefinite programming based on the theory of sum of squares polynomials and on the theory of moments. The key advantage of this approach is that a global minimum of a possible nonconvex optimal control problem in model predictive control is obtained via semidefinite programming. This property can avoid unacceptable performance degradation in model predictive control. The applicability of polynomial optimization in nonlinear model predictive control was demonstrated on an example. However, this example also shows that a straightforword application of sum of squares techniques in model predictive control is only applicable to small and medium size polynomial optimization problems, because polynomial optimization problems are computationally very demanding using currently available algorithms.

References

[1] F. Allgöwer, T.A. Badgwell, J.S. Qin, J.B. Rawlings, and S.J. Wright. Nonlinear Predictive Control and Moving Horizon Estimation – An Introductory Overview. In *Advances in Control: Highlights of European Control Conference 1999*, pp. 391–449. Springer, 1999.

[2] L.T. Biegler. Efficient Solution of Dynamic Optimization and NMPC Problems. In *Nonlinear Model Predictive Control, Birkhäuser*, pp. 219–244, 2000.

[3] R.R. Bitmead, M. Gevers, I.R. Petersen, and R.J. Kaye. Monotonicity and Stabilizability Properties of Solutions of the Riccati Difference Equation: Propositions, Lemmas, Theorems, Fallacious Conjectures and Counterexamples. *Systems and Control Letters*, 5:309–315, 1985.

[4] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory.* SIAM, 1994.

[5] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[6] H. Chen and F. Allgöwer. A Quasi-Infinite Horizon Nonlinear Model Predictive Control Scheme with Guaranteed Stability. *Automatica*, 34:1205–1218, 1998.

[7] M.D. Choi, T.Y. Lam, and B. Reznick. Sums of Squares of Real Polynomials. In *Proceedings of Symposia in Pure Mathematics 58*, pp. 103–126, 1995.

[8] M. Diehl, R. Findeisen, Z.K. Nagy, H.G. Bock, J.P. Schlöder, and F. Allgöwer. Real-time Optimization and Nonlinear Model Predictive Control of Processes

Governed by Differential-Algebraic Equations. *Journal of Process Control*, 4:577–585, 2002.

[9] R. Findeisen, L. Imsland, F. Allgöwer, and B.A. Foss. State and Output Feedback Nonlinear Model Predictive Control: An Overview. *European Journal of Control*, 9:190–206, 2003.

[10] D. Henrion and J.B. Lasserre. {*GloptiPoly*} – *Global Optimization over Polynomials with Matlab and SeDuMi*, 2003. Available from http://www.laas.fr/ henrion/software/gloptipoly/.

[11] D. Henrion and J.B. Lasserre. Detecting Global Optimality and Extracting Solutions in GloptiPoly. In D. Henrion and A. Garulli, editors, *Positive Polynomials in Control*. Springer, 2004.

[12] D. Hilbert. *Gesammelte Abhandlungen*. Springer Verlag, 1932.

[13] T. Jacobi and A. Prestel. Distinguished Representations of Strictly Positive Polynomials. *Journal für reine und angewandte Mathematik*, 532:223–235, 2001.

[14] S. Keerthi and E. Gilbert. Optimal Infinite-Horizon Feeback Laws for a General Class of Constrained Discrete-Time Systems: Stability and Moving-Horizon Approximations. *Journal of Optimization Theory and Applications*, 57:265–293, 1988.

[15] J.B. Lasserre. Global Optimization with Polynomials and the Problems of Moments. *SIAM Journal on Optimization*, 11:796–817, 2001.

[16] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained Model Predictive Control: Stability and Optimality. *Automatica*, 36:789–814, 2000.

[17] H. Michalska and D.Q. Mayne. Robust Receding Horizon Control of Constrained Nonlinear Systems. *IEEE Transactions on Automatic Control*, 38:1623–1633, 1993.

[18] A. Papachristodoulou and S. Prajna. On the Construction of Lyapunov Functions using the Sum of Squares Decompostion. In *Proccedings of 41th IEEE Conference on Decision and Control, USA*, pp. 3482–3487, 2002.

[19] P.A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization.* PhD thesis, California Institute of Technology, 2000.

[20] P.A. Parrilo and B. Sturmfels. Minimizing Polynomial Functions. In S. Basu and L. Gonzalez-Vega, editors, *Algorithmic and Quantitative Real Algebraic Geometry*, vol. 60, pp. 83–99. DIMACS, 2000.

[21] S. Prajna, A. Papachristodoulou, and P. A. Parrilo. {*SOSTOOLS*} – *Sum of Squares Optimization Toolbox for Matlab*, 2002. Available from http://www.cds.caltech.edu/sostools.

[22] A. Prestel and C.N. Delzell. *Positive Polynomials. From Hilbert's 17th Problem to Real Algebra.* Springer Verlag, 2000.

[23] M. Putinar. Positive Polynomials on Compact Semi-Algebraic Sets. *Indiana University Mathematical Journal*, 42:969–984, 1993.

[24] S.J. Qin and T.A. Badgwell. A Survey of Industrial Model Predictive Control Technology. *Control Engineering Practice*, 11:733–764, 2003.

[25] T. Raff, R. Findeisen, C. Ebenbauer, and F. Allgöwer. Model Predictive Control for Discrete Time Polynomial Control Systems: A Convex Approach. In *Proceedings of the 2nd IFAC Symposium on System, Structure and Control, Mexico*, pp. 158–163, 2004.

[26] J.B. Rawlings. Tutorial Overview of Model Predictive Control. *IEEE Control Systems Magazine*, 20:38–52, 2000.

[27] B. Reznick. Uniform Denominators in Hilbert's Seventeenth Problem. *Mathematische Zeitschrift*, 220:75–97, 1995.

[28] N.Z. Shor. Class of Global Bounds of Polynomial Functions. *Tekhnicheskaya Kibernetika*, 6:9–11, 1987.

[29] B. Sturmfels. *Solving Systems of Polynomial Equations.* American Mathematical Society, 2002.

[30] M.H. Wright. The Interior-point Revolution in Optimization: History, Recent Developments, and Lasting Consequences. *Bulletin of the American Mathematical Society*, page Electronically published, 2004.

Comparison of Two Measures of Dynamic Stability During Treadmill Walking

M. Schablowski[1] and H.J. Gerner[1]

Stiftung Orthopädische Universitätsklinik Heidelberg
Matthias.Schablowski@ok.uni-heidelberg.de

Summary. Stability of biped walking is an important characteristic of legged locomotion. Whereas clinical investigations often relate increased variability to decreased stability, there are only few studies examining stability aspects directly. On the other hand, various papers from the field of robotics are dedicated to the question: how can the stability of legged locomotor systems be quantified? Particularly, when it comes to realizing fast motions in robots, the question of maintaining dynamic stability is of utmost importance. The current paper presents a theoretical comparison of several measures for dynamic stability – namely Floquet multipliers and Local Divergence Exponents (LDE). The sensitivity of these parameters to changes in speed of human treadmill locomotion is investigated. Experimental results show that two different types of stability with respect to speed dependence seem to exist. Short term LDE and Floquet multipliers consider the stability over a period of one stride, which seems to be optimal at intermediate walking speeds. Long term LDE quantify stability of movement trajectories over multiple strides. This type of stability decreases with speed and may be one reason for changing gaits from walking to running at a certain speed value.

1 Introduction

Stability of biped walking is an important characteristic of legged locomotion [1]. Studies which investigate this issue from a clinical point of view mainly target the question: how is an upright gait maintained in the presence of internal and external perturbations [2, 3, 4]?. Particularly, the effect of macroscopic perturbations challenging the musculoskeletal system's capabilities of balance control is studied. The measures being used for determining the degree of stability in this context are commonly based on the vertical projection of the center of mass onto the base of support [5].

When analyzing steady-state locomotion in the absence of external large scale perturbations, a different notion of stability comes into play. Here, the question is: how stably does the movement system reproduce the trajectories of the state-variables from one cycle of motion to the other? This issue can be investigated by using tools from the theory of nonlinear dynamical systems like Floquet multipliers [6] and Lyapunov exponents [7, 8]. Both of these measures quantify the effect of microscopic perturbations on the movement trajectories. Microscopic perturbations arise from extrinsic circumstances like irregular surfaces as well as from internal sources like inaccuracies during movement generation. Analyzing the impact of such microscopic perturbations during movement generation is essential for a deeper understanding of neuromuscular control in the context of nonlinear dynamical system theory [9].

The present study investigates dynamic stability of treadmill walking in healthy subjects with special consideration of the influence of walking speed. In particular, for the first time a comparison of two similar measures of dynamic stability – Floquet multipliers and Local Divergence Exponents – for this type of movement is undertaken.

2 Methods

The following subsections briefly review the mathematical basis of the measures of dynamic stability used here and the experimental setup used for data acquisition. Specifically, an adaption of the calculation of Floquet-Multipliers to treadmill walking and the method used for the estimation of Lyapunov exponents are explained.

2.1 Dynamic Stability Measures

Modified Floquet-Multipliers

Let γ be a periodic orbit of some flow Φ_t in $\Re^n$ arising from a nonlinear flow $f(x)$. We then first take a local cross section $\Sigma \subset \Re^n$ of dimension n-1. Σ defines a hypersurface which needs not to be planar but has to be chosen such that the flow is everywhere transverse to it [10]. Further on, only those points where the Φ_t intersects Σ are considered ("Poincaré map"). Thereby it is assumed that the oscillation period T_p is fixed. The Floquet-Multipliers represent a measure of how the intersecting points vary from one period of oscillation to the next.

For application of Floquet multipliers to gait analysis one starts from a discrete state space description of the human movement system [11]:

$$\mathbf{x}(k) = [q_1(k), \ldots, q_S(k), \dot{q}_1(k), \ldots, \dot{q}_S(k)]^T \tag{1}$$

For investigating dynamic stability of gait, possible state variables q and $\dot{q}$ are primarily the joint angles and the trajectories of selected points on the body surface. The number of state variables S depends on the measurement system at hand and the desired degree of differentiation of the model. Applying a Poincaré mapping to equation (1) leads to the following formulation:

$$\mathbf{x}_{n+1} = \mathbf{P}_p(\mathbf{x}_n) \tag{2}$$

$\mathbf{P}_p$ represents a nonlinear vector function, which maps the state space vector $\mathbf{x}$ at point in time p of the oscillation period n to the state space vector at the same point in time during period $(n+1)$ (p is arbitrary, yet fixed during the oscillation period). In the following only strictly periodic solutions of equation (2) are considered:

$$\mathbf{x}_p^* = \mathbf{P}_p(\mathbf{x}_p^*) \tag{3}$$

For investigating the stability of the equilibrium point $\mathbf{x}^*$ against infinitesimal perturbations $\delta\mathbf{x}$, $\mathbf{P}_p$ is linearized around $\mathbf{x}^*$ as follows:

$$\mathbf{J}_p = \frac{\boldsymbol{\partial}\mathbf{P}_\mathbf{p}}{\boldsymbol{\partial}\mathbf{x}}(\mathbf{x}_p^*) \tag{4}$$

$$\boldsymbol{\delta}\mathbf{x}_{n+1} = \mathbf{J}_p \delta\mathbf{x}_n \tag{5}$$

Here $\mathbf{J}_p$ is the Jacobian (also called Floquet or monodromy) matrix [10]. Due to the linearization, now, instead of the nonlinear vector function $\mathbf{P}_p$ the linear matrix $\mathbf{J}_p$ needs to be estimated from the measurement data (see below). Stability of the equilibrium point $\mathbf{x}_p^*$ of the linear system (5) is determined through the Eigenvalues $\beta_s^p, s = 1, \ldots, 2S$ of $\mathbf{J}_p$. The Eigenvalues β_s^p are also called Floquet Multipliers. $\mathbf{x}_p^*$ is stable if $|\beta_s^p| < 1$ for all s. In case of stability the Eigenvalue with the maximum absolute value $|\beta_{max}^p|$ dominates the system behavior.

Floquet multipliers were introduced into the field of gait analysis in [12] and were applied to the investigation of dynamic stability of post-polio gait [11, 6]. In these studies, overground walking was investigated and in particular the transition into the dynamical equilibrium was analyzed. For this purpose, the first four steps starting from stand were analyzed under the assumption that dynamic equilibrium was achieved after the fourth step. In this case, Floquet multipliers quantify stabilization of gait after a very large perturbation (the resting position).

During treadmill walking however, dynamic stability against infinitesimal deviations from dynamic equilibrium is of interest. Therefore, a different definition of the dynamic equilibrium needs to be assumed. In this paper, this equilibrium is defined as the mean value of the state space vector $\mathbf{x}$ at point in time p over all strides of a particular trial:

$$x_{p,s}^* = \frac{1}{N} \sum_{n=0}^{N-1} x_s(p + n \cdot T_p), \qquad s = 1, \ldots, 2S \tag{6}$$

Deviations from $\mathbf{x}_p^*$ for all N strides are calculated by:

$$\delta\mathbf{x}_n = \mathbf{x}(p + n \cdot T_p) - \mathbf{x}_p^*, \qquad n = 0, \ldots, N-1 \tag{7}$$

The elements $J_{p,ns}$ of the Jacobian $\mathbf{J}_p$ now are estimated from a system of linear equations:

$$\delta x_{n+1} = \sum_{s=1}^{2S} J_{p,ns} \cdot \delta x_{ns}, \qquad n = 1, \ldots, N-1 \tag{8}$$

The $N-1$ equations of system (8) describe the linear evolution of the deviations from stride n to stride $n+1$. For this reason, the minimum number of strides that need to be measured is determined by the dimension of the state space model. A necessary condition for a unique solution of equation (8) is [1]:

$$N \geq 2S + 1 \tag{9}$$

This relation is not mentioned explicitly in [6], however, condition (9) is fulfilled implicitly by the experimental protocol (the first 3 strides from 16 trials are accumulated to a measurement series of 48 strides, the 4th stride is the equilibrium [6]).

In theory, the Floquet multipliers are independent from the point in time p when the Poincaré map is being calculated. In experimental practice however, there are nevertheless differences when considering different points in time during the gait cycle. For this reason the authors in [6] use the times of maximum knee flexion, because these, according to the authors, lead to the most consistent results. These differences result from slight errors in the detection of the points in time from experimental data but also from the fact human gait is not strictly periodic.

Therefore, in the current paper a modified version of the Floquet multipliers is introduced which is based on the dynamic equilibrium during treadmill walking as defined in equation (3). The model of the movement system consists of the three sagittal joint angles for the left and the right leg and the corresponding temporal derivatives (dimension of model $2S = 12$). The derivatives at points in time k are estimated as follows:

$$\dot{q}(k) = (q(k+1) - q(k-1))/2 \tag{10}$$

In order to eliminate the dependence on the point in time when the Poincaré maps are calculated, the Floquet multipliers were modified by taking the average values of the Floquet multipliers over all K points of the normalized gait cycle (α_{str}):

$$\alpha_{str} = 1/K \sum_{p=1}^{K} \beta^{p}_{max} \tag{11}$$

Local Divergence Exponents

Lyapunov exponents quantify the development of trajectories in state space after an initial infinitesimal perturbation. They are a measure for the sensitivity of a system to such perturbations and express the degree of predictability

[1] A sufficient condition is that the determinant of the system (8) must be different from 0.

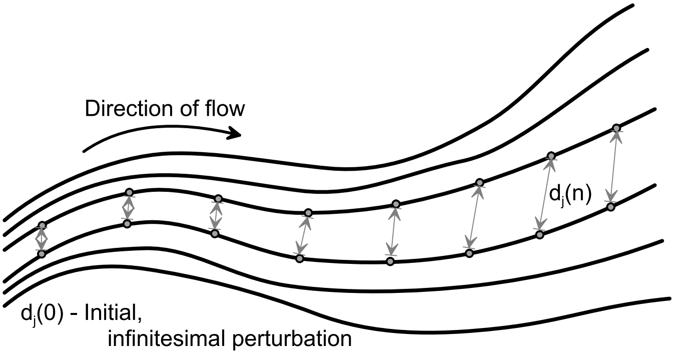


Fig. 1. Development of initial perturbations with the system dynamics and its quantification through local divergence exponents (adapted from [13])

of a dynamical system (see Fig. 1). For a complete description of the effects of a perturbation, the system development needs to be described in all directions of the reconstructed state space yielding multiple Lyapunov exponents. The number of these exponents is equal to the dimension of the embedding space and altogether they set up the so called Lyapunov spectrum. In practice, calculation of the full spectrum is difficult and analysis is restricted to the largest Lyapunov exponent λ_{max}, which dominates the behavior of the system. In this paper, the algorithm described in [14] using the modification from [13] is applied. In contrast to other implementations [15], this algorithm uses all the points of a time series and thus allows for reliable estimates for short measurements. In order to eliminate dynamical correlations, a Theiler window of one stride duration was applied during reconstruction of the state space [13, 16].

The maximum finite time Lyapunov exponent λ_{max} is defined as

$$d(t) = d_0 e^{\lambda_{max} t} \tag{12}$$

where $d(t)$ is the mean distance between neighboring trajectories at time t. d_0 is the initial distance between the reference points $\mathbf{x}_j$ and their nearest neighbors $\mathbf{x}_k$. The nearest neighbor is calculated as follows [14]:

$$d_0 = \min_{\mathbf{x_k}} \|\mathbf{x}_j - \mathbf{x}_k\|, \qquad j, k = 1, \ldots, M \ , \ j \neq k \tag{13}$$

Here, M is the number of reference trajectories. This definition of the Lyapunov exponent λ_{max} is valid only in the dual limit $t \rightarrow \infty$ and $d_0 \rightarrow 0$. For practical application with finite sampling times, the λ_{max} are approximated by their finite estimated values λ^*, which are calculated as follows [13, 14]:

$$\ln d_j(i) \approx \lambda^* \cdot i \cdot \mathit{\Delta} t + \ln d_0 \tag{14}$$

In equation (14) $d_j(i)$ designates the j_{th} pair of nearest neighbors after i discrete time steps, which is an expression for local divergence. Since Lyapunov

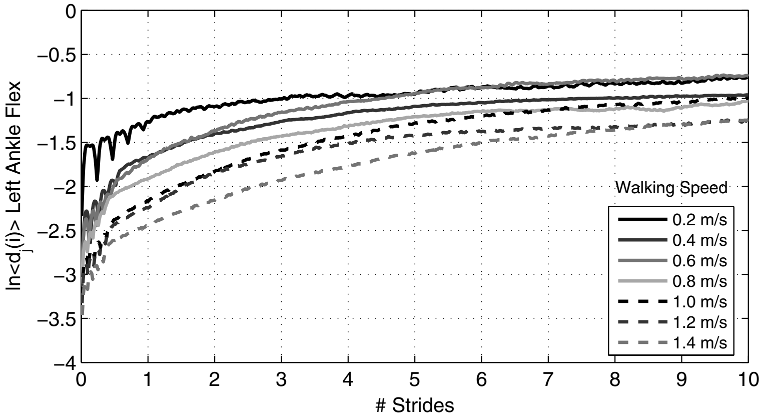


Fig. 2. Local divergence for the left ankle flexion angle of one subject at different walking speeds ($v_{TM} = 0.2, \ldots, 1.4\,\mathrm{m/s}$)

exponents are a measure of the mean changes of d_0 over time, the $d_j(i)$ are averaged over all M reference trajectories [14]:

$$\overline{d}(i) = \frac{1}{\Delta t}\langle \ln d_j(i)\rangle \qquad (15)$$

Here, $\langle \cdot \rangle$ denotes the mean over all values of j. The mean local divergence $\overline{d}(i)$ over the number of strides for the left ankle joint angle of one subject is depicted in Fig. 2. The line coding in the graph corresponds to seven individual walking speeds between 0.2 und 1.4 m/s. In order to allow for direct comparison of the curves belonging to different speeds, the x-axis was normalized to the duration of one stride as described in [17]. The oscillations during the first stride are caused by differences in the density of points when following trajectories in the different directions of the state space.

λ^* is derived from these curves [13, 14]. In order to distinguish the estimated values λ^* from the theoretically defined Lyapunov exponents λ_{max}, they are also called local divergence exponents [13]. The estimation depends on selecting a range with constant slope in the curves in Fig. 2 which is difficult to realize with sufficient accuracy in experimental data [14].

Therefore, an approach first described by [13] was used here: two different time scales were analyzed, where short term LDE (denoted as λ_{ST}) were calculated over the first stride and long term LDE (λ_{LT}) were computed for the strides 4–10 (Fig. 3).

Using these procedures, local scaling exponents were calculated for a series of measurements obtained from an experimental protocol described in the next subsection.

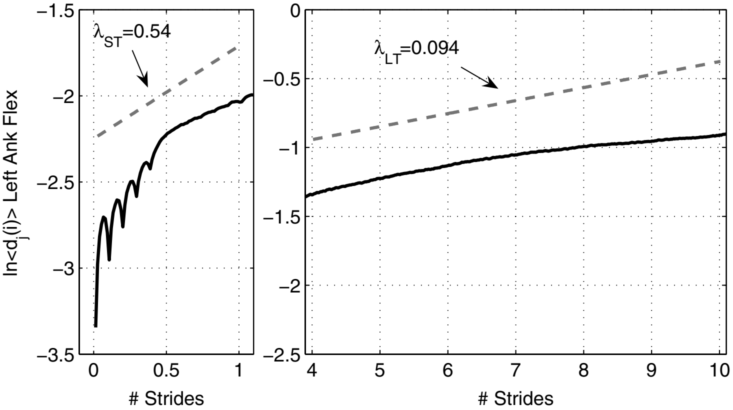


Fig. 3. Linear regression for deriving local divergence exponents from the curves in Fig. 2 [13]

Theoretical Comparison of the Two Measures

Both, the LDE as well as the Floquet multipliers represent geometrical measures, which quantify the development of two neighboring trajectories with the system dynamics. Regarding their application to the investigation of dynamic stability of human gait [13, 18, 6, 12, 11], however, there are several important differences to note:

Peridocity: Floquet multipliers are based on the assumption of a strictly periodical system, whose attractor is analyzed at specific points in time [12]. The LDE do not require the system to be periodic.

Linearization: Floquet multipliers are the Eigenvalues of a linearized transformation matrix describing the system dynamics between selected periodic points in time. The LDE do not rely on any kind of linearization.

Modelling: Floquet multipliers are based on an a priori model of the human movement system [6], which determines the dimension of the state space. The embedding dimension which is used for calculating the LDE is derived from experimental data during further analysis. Therefore, Floquet multipliers inherently describe dynamic stability of the entire movement system as it is captured by the model. The LDE on the other hand quantify stability of the individual joint angle trajectory.

2.2 Experimental Protocol of Gait Data Acquisition

Using a commercial motion analysis system (Motion Analysis, Santa Rosa, CA) kinematic data from 10 subjects with no prior history of gait disorders were acquired. Three-dimensional trajectories of retroflective markers placed on anatomical landmarks as prescribed in [19] were recorded and transformed into joint angles [20]. For each subject, seven trials of 90 s each with fixed

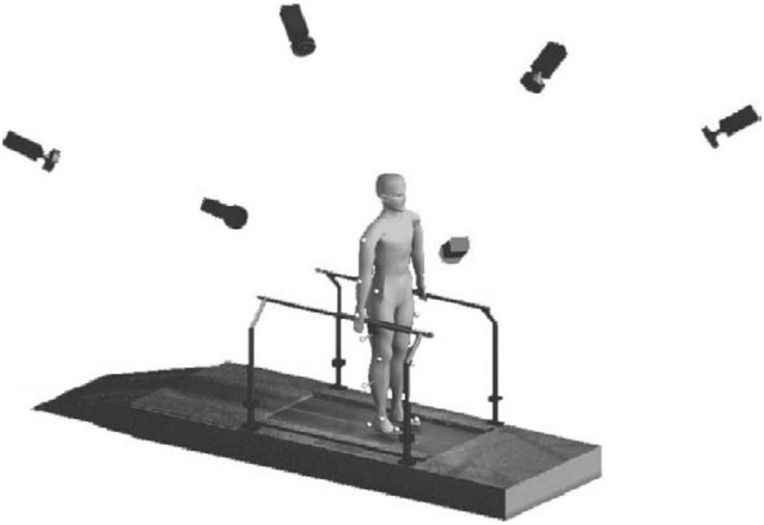


Fig. 4. Experimental setup for acquisition of gait data during treadmill walking

speeds ranging from $v_{TM} = 0.2, 0.4, \ldots, 1.4\,\mathrm{m/s}$ were recorded. Additionally, three trials at slow, medium and fast walking speeds were performed. In order to avoid systematic errors, the order of speed was randomized. Whereas there are reference data for walking overground [21, 22, 23], here, for the first time a normative base of kinematic movement data is provided for treadmill walking. Parts of these results were published in [24].

3 Results

3.1 Sagittal Plane Kinematics

The mean joint angles for the seven fixed walking speeds are displayed in Fig. 5. Assuming symmetry between the left and right side of the body, the data from both sides were averaged to yield one single time series per subject and speed. The most prominent changes with increasing speed are the increasing range of motion and a phase shift of the extreme values in the curves. This corresponds to the results described in [23]. On the basis of these joint angle curves, the two stability measures introduced in the previous section were calculated.

3.2 Local Dynamic Stability

Floquet-Multipliers

In Fig. 6 the modified Floquet multipliers α_{str} as defined in the previous section for all trials are plotted over walking speed (black squares). The solid

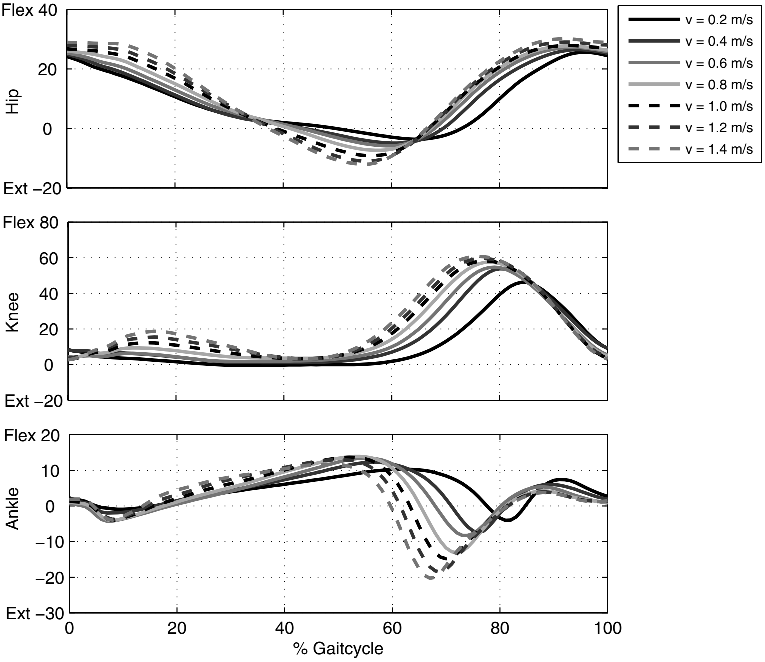


Fig. 5. Changes in the sagittal plane kinematics of the lower limb joints with varying walking speed

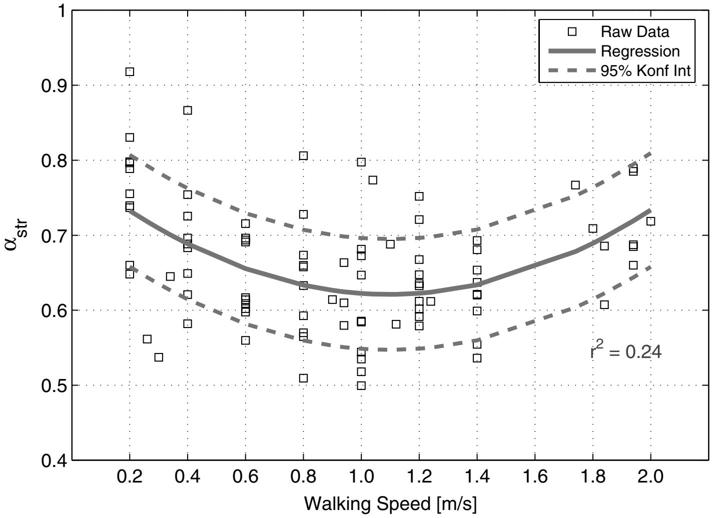


Fig. 6. Dynamic stability of treadmill walking as described by the averaged Floquet multiplier (α_{str}) plotted over walking speed

line delineates the quadratic regression curve over the range of speeds covered by the measurements. The regression explains 24% of variance in the raw data ($r^2 = 0.24$). Hence, the relationship is not very strong, yet nevertheless significant ($p < 0.05$) indicating a certain dependence of this stability measure on walking speed with a weak local minimum at intermediate speeds.

Local Divergence Exponents

The course of local dynamic stability over walking speed for the sagittal joint angles of the lower limbs is depicted in Fig. 7. The long term LDE (λ_{LT}) are shown in the left column, the short term values (λ_{ST}) are displayed in the right column. Quadratic regression curves are shown; the r^2-values give the proportion of explained variance in the data. The individual graphs contain the accumulated values for both the left and the right side joint angles.

In general, the regression curves indicate that the two estimates of local dynamic stability depend on walking speed. However, there appears to exist a significant difference in speed dependence on the two time scales: the λ_{ST} express a minimum at intermediate speeds for the hip joint and no clear tendency in the two remaining joints. The λ_{LT}, which describe the long term stability, continuously increase towards larger speed values.

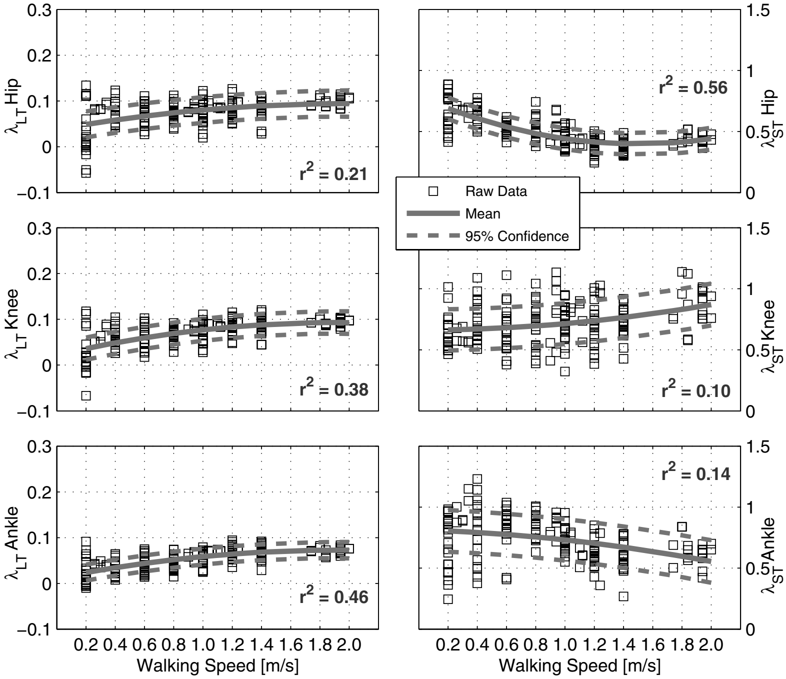


Fig. 7. Speed dependence of local divergence exponents including quadratic regression curves

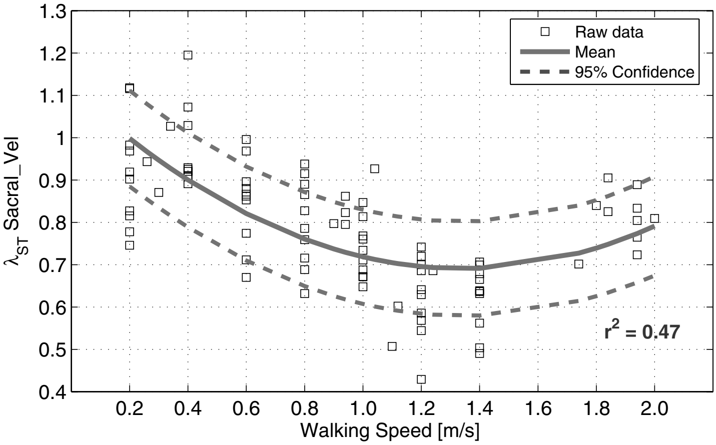


Fig. 8. Speed dependence of short term stability expressed in normal walking

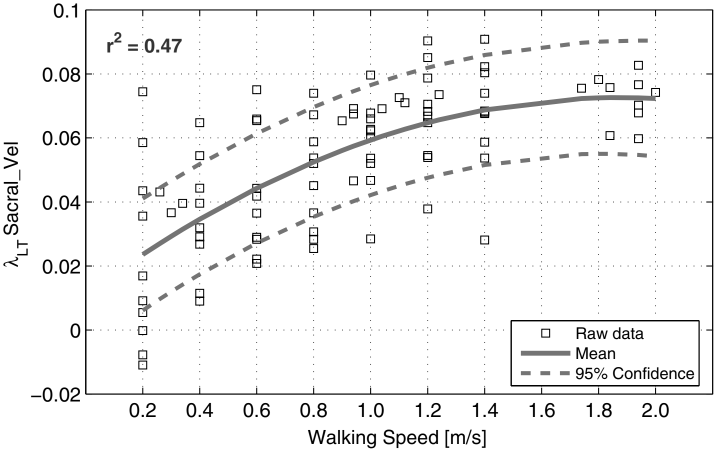


Fig. 9. Speed dependence of long term stability expressed in normal walking

In order to further investigate these different tendencies and to allow for comparison with the results obtained in [25], the local divergence exponents here were also calculated for the time derivatives of the sacral marker position on both time scales. The main reason for using these velocity data instead of the position is to eliminate drift effects in the absolute marker positions during walking on the treadmill [25].

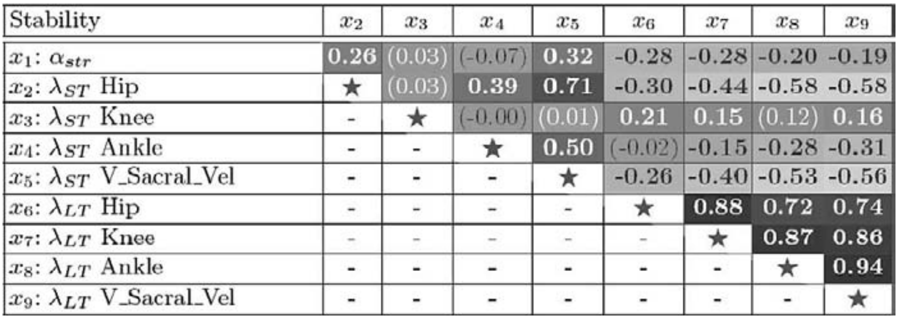
The course of the regression curves in the Figs. 8 and 9 confirms the differences in speed dependence for the two different time scales: whereas the speed dependence of short term stability expresses a distinct minimum at intermediate speeds (≈ 1.3 m/s), the long term values show a clear increase towards higher speeds. The percentage of variance explained in both cases reaches almost 50%. Since the basic dynamic properties of the system are in principal not dependent on the measurement modalities chosen for analysis, this supports the idea of two different effects of speed on stability of walking.

3.3 Experimental Comparison of the Two Measures

The two measures for dynamic stability applied here are both based on a geometrical point of view with respect to the system dynamics. However, as shown in Sect. 2.1, there are several differences in the way the measures are defined. The fact that they both aim at quantifying dynamic stability of walking calls for an analysis of the relation between their values.

Table 1 gives a list of the linear correlations between the modified Floquet multipliers introduced in Sect. 2.1 and the Local Divergence exponents. The individual cells of the table contain the correlation coefficients coded as gray values. Darker gray cells contain large positive correlations, lighter gray cells display negative values. Statistically significant coefficients are printed in bold font, non significant correlations stand in parentheses.

Table 1. Linear Correlations between the different measures for dynamic stability



Stability	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
x_1: α_{str}	**0.26**	(0.03)	(-0.07)	**0.32**	**-0.28**	**-0.28**	**-0.20**	**-0.19**
x_2: λ_{ST} Hip	★	(0.03)	**0.39**	**0.71**	**-0.30**	**-0.44**	**-0.58**	**-0.58**
x_3: λ_{ST} Knee	-	★	(-0.00)	(0.01)	**0.21**	**0.15**	(0.12)	**0.16**
x_4: λ_{ST} Ankle	-	-	★	**0.50**	(-0.02)	**-0.15**	**-0.28**	**-0.31**
x_5: λ_{ST} V_Sacral_Vel	-	-	-	★	**-0.26**	**-0.40**	**-0.53**	**-0.56**
x_6: λ_{LT} Hip	-	-	-	-	★	**0.88**	**0.72**	**0.74**
x_7: λ_{LT} Knee	-	-	-	-	-	★	**0.87**	**0.86**
x_8: λ_{LT} Ankle	-	-	-	-	-	-	★	**0.94**
x_9: λ_{LT} V_Sacral_Vel	-	-	-	-	-	-	-	★

Leaving out $\lambda_{ST} Knee$ there are essentially three blocks containing correlations with different signs: the features x_1 to x_5 are weakly positively correlated with each other and negatively with the features x_6 to x_9. The latter are strongly correlated to each other. The absolute value of the correlation coefficients increases significantly from the first to the third block.

4 Discussion

The results presented in the previous section comprise two measures of dynamic stability of walking. Theoretical comparison of the two measures as described in Sect. 2 illustrates differences which arise from the mathematical definitions of the two measures. However, the difference in speed dependance shown in Sect. 3 demonstrates that these theoretical differences are not fully replicated by the experimental application. Whereas theory suggests grouping the measures according two their mathematical definition – Floquet versus Lyapunov –, the experiments performed here emphasize the importance of the time scale on which the measures are based.

This finding is supported by the data listed in Table 1 as follows: since block $x_1 - x_5$ mainly consists of the λ_{ST} which are calculated over the first stride, it makes sense to assume that this block indicates short term stability. α_{str} is positively correlated with $\lambda_{ST}Hip$ and $\lambda_{ST}V\_Sacral\_Vel$. This coincides well with the fact that this measure is based on the assumption of a strictly periodic system (see Sect. 2.1) and consequently measures short term stability as well. The second block which consists of the λ_{LT} describes some sort of long term stability that is captured over several strides. The differences in modeling between α_{str} and λ seem to make no difference here. From this, it may be argued that capturing the full kinematics of the entire movement system may be not necessary to describe dynamic stability of walking.

Both the measures describing short term stability of periodic walking motion over one stride (α_{str} and λ_{ST}) exhibit a tendency towards maximal stability at intermediate speeds, whereas long term stability seems to decrease continuously towards faster speeds[2]. The minima in α_{str} and $\lambda_{ST,Hip}$ agree with the results from [26] where a maximum in stability for the head marker trajectory at the preferred walking speed was found. The average self-selected speed of the subjects analyzed here was around 1.02 m/s which does not coincide with the location of the minimum in λ_{ST} for the hip (≈ 1.4 m/s). This together with the fact that the quadratic speed dependency is only weak corresponds well with the conclusion of the authors in [26] that (short term) stability is only one criterion among others determining the self selected walking speed.

The significant increase in λ_{LT} with walking speed indicates increasing instability during faster gaits. This corresponds well with the interpretation that the local divergence exponents quantify the capability of the locomotor system to compensate for slight irregularities from stride to stride. This capability is increasingly challenged at higher gait speeds which leads to decreased stability of motion. The highest speed values adopted by the subjects in this study are close to the walk-run-transition (mean value $v_{TM} = 1.89 \pm 0.08$ m/s, compared e.g. to $v_{TM} = 2.17$ m/s in [27]). According to the theory of nonlinear dynamical systems, this transition can be interpreted as switching to a different system attractor. Therefore, following the reasoning illustrated in [28], it can be hypothesized that this increasing instability is an important factor in changing gaits [28, 29, 27].

In contradiction to the results obtained here, the authors in [25] found a continuous increase on both time scales. The authors investigated the derivative with respect to time of the three dimensional trajectory of a

[2] Larger values of λ indicate decreasing stability.

marker in the area of the sacral vertebral column. A potential reason for the difference in the results is the smaller range of speeds investigated there ($v_{pref}, v_{pref} \pm 20\%, v_{pref} \pm 40\%$).

The correlation between λ_{LT} and walking speed which was shown here also supports the results described in [7]. The authors demonstrated that older subjects show significantly higher values in their local divergence exponents compared to a younger reference group. This was interpreted as an indicator of increased instability of walking with age. At the same time, the older group showed significantly lower walking speeds, a fact that was not examined in more detail by the authors. The dependence of the local divergence exponents on walking speed, which was found here, underlines the explanation found by the authors that, in fact, aging leads to increased walking instability. This increase is most probably not only a consequence of the difference in walking speed [7].

An interesting question is why the speed dependence of the sacral marker velocities is much more pronounced than that of the joint angles. There are two possible explanations for this: on the one hand, the differences could be related to the way the stability measures are determined. There are several marker trajectories connected in a nonlinear way during the the calculation of the joint angles, which might lead to slight nonstationarities. On the other hand, it can be hypothesized that there are different control mechanisms involved in controlling the position/velocity of the sacral marker in comparison to the joint angles.

The fact that virtually all values of λ have positive values and as such indicate chaos does not necessarily mean that, gait itself is chaotic [13]. As pointed out in [30] stochastic noise can indicate chaos where in fact there is none. A sound investigation of this question can be undertaken using the method of surrogate data [30].

5 Conclusion

The results presented here complement data from previous studies [25]. Two different mechanisms regarding dynamic stability of locomotion seem to exist: the minimum in λ_{ST} at intermediate speed values suggests that short term stability may serve as an optimization criterion in the locomotor system. The increased local dynamic stability (λ_{LT}) towards slower speeds may explain the tendency towards slower speeds in many gait pathologies. On the other hand, the increasing instability at higher speeds may be one reason for the transition from walking to running.

Acknowledgment

The author would like to thank all the subjects who volunteered in this study.

References

[1] R. Full, T. Kubow, J. Schmitt, P. Holmes, and D. Koditschek, “Quantifying dynamic stability and maneuverability in legged locomotion,” *Integrative and Comparative Biology*, vol. 42, pp. 149–157, 2002.

[2] D. Krebs, D. Goldvasser, J. Lockert, L. Portney, and K. Gill-Body, “Is base of support greater in unsteady gait?” *Physical Therapy*, vol. 82, no. 2, pp. 138–47, Feb 2002.

[3] M. Sliwinski, S. Sisto, M. Batavia, B. Chen, and G. Forrest, “Dynamic stability during walking following unilateral total hip arthroplasty,” *Gait and Posture*, vol. 19, no. 2, pp. 141–7, Apr 2004.

[4] M. Hahn and L. Chou, “Can motion of individual body segments identify dynamic instability in the elderly?” *Clinical Biomechanics*, no. 8, pp. 737–44, Oct 2003.

[5] M. Vukobratovic and J. Stepanenko, “On the stability of anthropomorphic systems,” *Mathematical Biosciences*, vol. 15, pp. 1–37, 1972.

[6] Y. Hurmuzlu, C. Basdogan, and D. Stoianovici, “Kinematics and dynamic stability of the locomotion of post-polio patients,” *Journal of Biomechanical Engineering*, vol. 118, no. 3, pp. 405–11, Aug 1996.

[7] U. H. Buzzi, N. Stergiou, M. J. Kurz, P. A. Hageman, and J. Heidel, “Nonlinear dynamics indicates aging affects variability during gait.” *Clinical Biomechanics*, vol. 18, no. 5, pp. 435–443, Jun 2003.

[8] J. Dingwell et al., “Local dynamic stability versus kinematic variability of continuous overground and treadmill walking,” *Journal of Biomechanical Engineering*, vol. 123, no. 1, pp. 27–32, Feb 2001.

[9] D. Sternad, “Debates in dynamics: A dynamical systems perspective on action and perception,” *Human Movement Science*, vol. 19, pp. 407–423, 2000.

[10] J. Guckenheimer and P. Holmes, *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields*, 3rd ed., ser. Applied Mathematical Sciences, F. John, J. Marsden, and L. Sirovich, Eds. New York ; Berlin ; Heidelberg ; Tokyo [u.a.]: Springer-Verlag, 1990, vol. 42.

[11] Y. Hurmuzlu, C. Basdogan, and J. Carollo, “Presenting joint kinematics of human locomotion using phase plane portraits and poincare maps,” *Journal of Biomechanics*, vol. 27, no. 12, pp. 1495–9, Dec 1994.

[12] Y. Hurmuzlu and C. Basdogan, “On the measurement of dynamic stability of human locomotion,” *Journal of Biomechanical Engineering*, vol. 116, no. 1, pp. 30–36, 1994.

[13] J. Dingwell et al., “Nonlinear time series analysis of normal and pathological human walking,” *Chaos*, vol. 10, no. 4, pp. 848–863, Dec 2000.

[14] M. Rosenstein, J. Collins, and C. DeLuca, “A practical method for calculating largest lyapunov exponents from small data sets,” *Physica D*, vol. 65, pp. 117–134, 1993.

[15] A. Wolf, J. Swift, H. Swinney, and J. Vastano, “Determining lyapunov exponents from a time series,” *Physica D*, vol. 16, pp. 285–317, 1985.

[16] J. Theiler, “Spurious dimensions from correlation algorithms applied to limited time-series data,” *Physical Reviews A*, vol. 34, pp. 2427–2432, 1986.

[17] J. Dingwell, “Variability and nonlinear dynamics of continous locomotion: Applications to treadmill walking and diabetic peripheral neuropathy,” PhD Thesis, The Penyslvania State University, 1998.

[18] J. Dingwell et al., "Increased variability of continuous overground walking in neuropathic patients is only indirectly related to sensory loss," *Gait and Posture*, vol. 14, no. 1, pp. 1–10, Jul 2001.

[19] M. Kadaba, H. Ramakrishnan, and M. Wootten, "Measurement of lower extremity kinematics during level walking," *Journal of Orthopaedic Research*, vol. 8, no. 3, pp. 383–92, May 1990.

[20] Motion Analysis, *OrthoTrak5.0 – Gait Analysis Software Reference Manual*, 3617 Westwind Blvd., Santa Rosa, CA 95403, 2001. [Online]. Available: http://www.motionanalysis.com/

[21] L. Larsson, P. Odenrick, B. Sandlund, P. Weitz, and P. Öberg, "The phases of the stride and their interaction in human gait," *Scandinavian Journal of Rehabilitation Medicine*, vol. 12, no. 3, pp. 107–12, 1980.

[22] T. Öberg, A. Karsznia, and K. Öberg, "Basic gait parameters: reference data for normal subjects, 10-79 years of age." *Journal of Rehabilitation Research and Development*, vol. 30, no. 2, pp. 210–223, 1993.

[23] ——, "Joint angle parameters in gait: reference data for normal subjects, 10–79 years of age." *Journal of Rehabilitation Research and Development*, vol. 31, no. 3, pp. 199–213, Aug 1994.

[24] M. Schablowski, "Speed dependence of local dynamic stability of treadmill walking," in *Proceedings of the 3rd International Symposium on Adaptive Motion in Animals and Machines (AMAM)*, Technische Universität Illmenau, 2005.

[25] C. Marin and J. Dingwell, "Local dynamic stability improves at slower walking speeds," in *Proceedings fo the 28th Annual Meeting of the American Society of Biomechanics*. ASB, 2004.

[26] K. Holt, S. Jeng, R. Ratcliffe, and J. Hamill, "Energetic cost and stability during human walking at the preferred stride frequency," *Journal of Motor Behavior*, vol. 27, no. 2, pp. 164–178, 1995.

[27] F. Diedrich and W. Warren, "Why change gaits? dynamics of the walk-run transition," *Journal for Experimental Psychology: Human Perception and Performance*, vol. 21, no. 1, pp. 183–202, Feb 1995.

[28] J. Brisswalter and D. Mottet, "Energy cost and stride duration variability at preferred transition gait speed between walking and running," *Canadian Journal of Applied Physiology*, vol. 21, no. 6, pp. 471–80, Dec 1996.

[29] L. Li, "Stability landscapes of walking and running near gait transition speed," *Journal of Applied Biomechanics*, vol. 16, no. 4, Nov 2000.

[30] J. Theiler, S. Eubank, A. Longtin, B. Galdrikian, and J. Farmer, "Testing for nonlinearity in time series: The method of surrogate data," *Physica D*, vol. 58, pp. 77–94, 1992.

Simple Feedback Control of Cockroach Running

J. Schmitt

Oregon State University, Corvallis, Oregon
schmitjo@engr.orst.edu

Summary. The spring loaded inverted pendulum model (SLIP) has been shown to accurately model sagittal plane locomotion for a variety of legged animals. Tuned appropriately, the model exhibits passively stable periodic gaits using either fixed leg touchdown angle or swing-leg retraction protocols. In this work, we investigate the relevance of the model in insect locomotion and develop a simple feedback control law to enlarge the basin of stability and produce stable periodic gaits for both the point mass and rigid body models. Control is applied once per stance phase through appropriate choice of the leg touchdown angle. The control law is unique in that stabilization is achieved solely through direct observation of the leg angle and body orientation, rather than through feedback of system positions, velocities, and orientation.

1 Introduction

The spring loaded inverted pendulum (SLIP) [1, 2, 3, 4] has emerged as a template of locomotion dynamics in the vertical plane for a large number of animals. In the model, the combination of legs animals use during each stance phase is idealized as a single effective leg represented by an elastic spring. Experimental research on animal locomotion has shown that running animals use multiple legs as one [5], and that the resulting body motion is well represented by this simple template [1, 2, 6, 7]. For suitable model parameters, both the rigid body and point mass models produce self-stabilizing periodic gaits, with or without the inclusion of simple feedforward control methodologies [8, 9, 10].

Much of the previous research on the SLIP model has utilized a fixed angle leg reset policy, where the leg touchdown angle remains constant for each stance phase. While this leg touchdown protocol produces stable periodic gaits for the point mass and rigid body models, the basin of stability for the rigid body model remains quite small and stability is only achieved for suitably tuned model parameters [8]. Additionally, use of a similar fixed angle leg reset protocol in a three dimensional spatial SLIP model produces only unstable periodic gaits [11]. In this work, we investigate the performance of the two dimensional SLIP model in modeling insect locomotion and, looking forward to developing simple stabilizing controllers for the spatial SLIP model, construct a simple feedback control law to expand the basin of stability for periodic orbits of both the point mass and rigid body SLIP models.

The paper is structured as follows. In Sect. 2, we briefly review the rigid body vertical plane SLIP formulation of [8]. While we primarily focus on the point mass model, extensions to motions of the rigid body model are also considered in Sect. 4.2. In Sect. 3, we investigate the stability of periodic gaits of the SLIP model with model parameters similar to those of the cockroach *Blaberus discoidalis*. We find that the fixed angle leg reset policy utilized in previous works produces predominantly unstable periodic gaits for these model parameters. These results, in conjunction with the stability results of the spatial SLIP model for a similar leg touchdown protocol, prompt an investigation into the stability properties of alternate leg touchdown protocols. Specifically, we begin by analyzing a leg touchdown protocol that places the next leg down at the same angle, relative to the inertial frame, that the previous leg was lifted. Neglecting gravity during the stance phase, we show that such a protocol produces neutrally stable period two as well as period one orbits. Since one of the eigenvalues of the period one orbits is negative one, continuity arguments suggest that leg placement protocols between these extremes will produce stable periodic gaits. We therefore introduce a simple, adaptive leg touchdown angle control law in Sect. 4.1 that connects both this leg touchdown protocol as well as the fixed angle leg reset policy, and show that the control law improves the stability properties of periodic gaits. We briefly consider the stability of periodic orbits of the rigid body model under a similar control law in Sect. 4.2, and numerically show that while inclusion of this control law produces stable gaits, periodic gaits are only necessarily achieved by incorporating delay feedback control into the control law.

2 Review of SLIP Model Formulation

The SLIP model, illustrated in Fig. 1, consists of a rigid body of mass m and moment of inertia I. A pair of legs are attached at a frictionless pin joint P in the body, displaced a distance d from the center of mass, where d can take either sign. While the SLIP model has primarily been used in investigating the motions of larger animals, in this work we examine its relevance in insect locomotion, in particular the locomotion of the cockroach *Blaberus discoidalis*. Cockroaches run in an alternating tripod gait, with three legs down during each stance phase. Experiments have shown that the forces produced by these legs during the stance phase are well represented by a single effective leg. Since the mass of the legs of the insect comprise less than 6% of the total mass, we therefore model the tripod of legs by a single, massless effective leg represented by a linear, elastic spring of nominal length l.

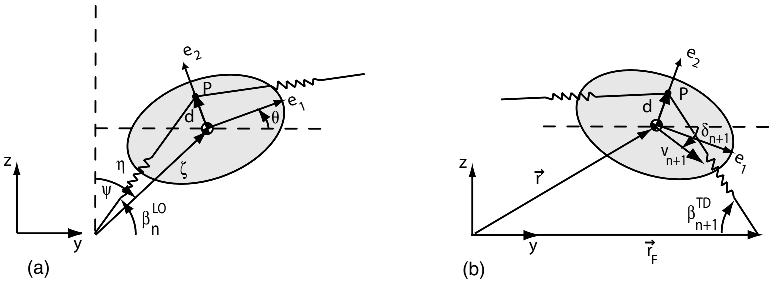


Fig. 1. SLIP rigid body model formulation, illustrating coordinate systems and relevant quantities at (**a**) leg lift-off and (**b**) leg touchdown

If both legs are attached at the same point P in the body, a full stride consists of a stance and flight phase, since left and right stance phases are indistinguishable. The stance phase begins when the leg, extended at its nominal length l, touches the ground at an angle β_n^{TD} with respect to the inertial frame. Superscripts of TD and LO denote values at touchdown and lift-off respectively, whereas subscripts identify the specific stance phase. The foot placement remains fixed during the stance phase and is represented by a moment free pin joint. Under the influence of gravity and its own momentum, the body moves forward in the y direction during the stance phase, compressing and extending the elastic leg. When the leg returns to its nominal length, it is lifted from an angle β_n^{LO} with respect to the inertial frame and a flight phase ensues. Simple ballistic dynamics govern the flight phase, and the next stance phase begins when the leg touches the ground, placed at an angle β_{n+1}^{TD} with respect to the inertial frame. While feedforward control is required to place each leg in anticipation of the next stance phase, no energy is required to move the leg to the prescribed position since the leg has no mass. As a result, the system is passive and energy is globally conserved, since no impacts or impulses occur.

The equations of motion for the stance phase are derived in [8] and summarized here, implemented with a linear spring leg:

$$\begin{aligned} m\ddot{\zeta} &= m\zeta\dot{\psi}^2 - mg\cos(\psi) - k(1-\frac{l}{\eta})(\zeta + d\cos(\theta+\psi)) \\ 2m\zeta\dot{\zeta}\dot{\psi} + m\zeta^2\ddot{\psi} &= mg\sin(\psi) + k(1-\frac{l}{\eta})d\zeta\sin(\theta+\psi) \qquad (1) \\ I\ddot{\theta} &= k(1-\frac{l}{\eta})d\zeta\sin(\theta+\psi) \end{aligned}$$

where η, ζ, ψ, θ, and k denote the leg length, distance from the foot placement to the center of mass, angle ζ makes with the vertical inertial axis, body rotation, and spring stiffness respectively. The flight phase dynamics are governed by simple ballistic dynamics which may be integrated to yield

$$y(t) = y^{LO} + \dot{y}^{LO}t,\ z(t) = z^{LO} + \dot{z}^{LO}t - \frac{1}{2}gt^2,\ \theta(t) = \theta^{LO} + \dot{\theta}^{LO}t\ . \quad (2)$$

The composition of the stance and flight phase dynamics result in a piecewise-holonomic system. While systems with piecewise-holonomic constraints can display asymptotic stability [12], they are often best described in terms of partial asympototic stability. In these cases, the corresponding periodic motions typically exhibit some neutral eigendirections (with eigenvalue = 0 or Floquet multiplier = 1) often associated with conserved quantities or symmetries such as energy conservation or rotational invariance. As in previous analyses for the conservative horizontal and vertical plane models [8, 13], perturbations to stable gaits in the direction of the eigenvector(s) of these conserved quantities or symmetries do not grow nor decay, but result in the attainment of a new gait.

3 Periodic Gaits, Stability and Control

Many prior SLIP model analyses focus on gait properties for a fixed angle leg reset model, where the leg is reset to its original touchdown angle at the beginning of each stance phase. For model parameter ranges consistent with larger animals, these studies illustrate the ability of the model to produce passively stable gaits. While we initially examine gaits produced for a fixed angle leg reset model with parameters set to those typical of the roach *Blaberus discoidalis*, we subsequently investigate simple feedback control laws that prescribe the leg touchdown angle and enhance stability.

In all cases, simulations are developed and performed using the Runge-Kutta integrator ode45 available in Matlab. As in previous work [8, 14, 15], we determine periodic orbits and their stability through use of a Poincaré map [16], with a Poincaré section defined at the instant of leg touchdown. Fixed points of the mapping represent periodic gaits in the continuous system, and are identified by a Newton-Raphson iteration. As illustrated in Fig. 1, we simplify the stability analysis through the definition of our Poincaré map in terms of variables of a new coordinate frame $(v, \delta, \theta, \dot{\theta}, \beta)$, where v is the center of mass velocity, δ is the velocity heading angle measured clockwise from the inertial horizontal axis, θ is the body rotation and $\dot{\theta}$ is the angular velocity.

Stability of fixed points of the mapping is determined by examining the eigenvalues of the linearization of the Poincaré map about the fixed point. If any eigenvalue is greater than unity the periodic orbit is unstable and if all non-unity eigenvalues remain within the unit circle, the periodic orbit is stable.

3.1 Fixed Angle Leg Reset Gaits

We begin by investigating the point mass SLIP model with parameters similar to those of the cockroach *Blaberus discoidalis*: spring stiffness (k) of 20 N/m,

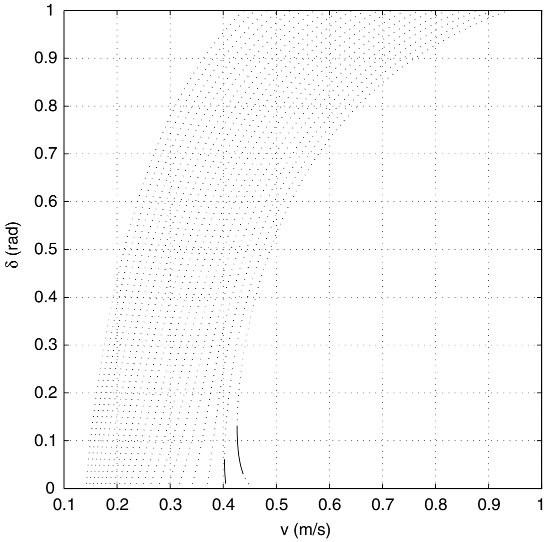


Fig. 2. Point mass SLIP gait families $(v_n^{TD}, \delta_n^{TD})$ for $\beta_n^{TD} = 1.1 - 1.3$. Gait stability is indicated by *dotted* and *solid lines* for unstable and stable gaits respectively. Model parameters used in calculating the gaits are similar to those of *Blaberus discoidalis*, as described in the text. Gait families for increasing β_n^{TD} are obtained as one moves from *right* to *left* in the plot

leg length (l) of 0.015 m, body mass (m) of 0.0025 kg, and a leg touchdown angle (β_n^{TD}) between $1.1 - 1.3$ radians. These parameter values are chosen to produce reasonable leg compressions (less than 50%) during the stance phase, as well as to match experimental stride length and frequency results [17]. While the theoretical results neglect gravity during the stance phase so that angular momentum is conserved, gravity is included in both stance and flight phases in all simulations conducted in this work. In this section, we consider only the point mass SLIP model with $d = 0$; extensions to the rigid body model with $d \neq 0$ and an adaptive control law are presented in Sect. 4.2.

Using Newton-Raphson routines in conjunction with the model simulation, we obtain a one parameter family of periodic gaits, depending upon the body touchdown velocity, v. The gait family is initially obtained for a nominal value of $\beta_n^{TD} = 1.20$, which best matches the experimental stride length and frequency results. Gait families for values of $\beta_n^{TD} = 1.1 - 1.3$ are subsequently obtained to see if changes in gait stability occur for these alternate values, even though stride length and frequency results do not necessarily match those observed experimentally. As illustrated in Fig. 2, almost all periodic gaits obtained over this range of leg touchdown angles are unstable. A small range of stable periodic gaits exist for leg touchdown values between $1.1 - 1.11$, although the minimum touchdown speeds obtained in these instances exceed the preferred operating speed of 0.25 m/s of *Blaberus discoidalis.*

3.2 Alternate Leg Angle Touchdown Protocols

The stability properties of the SLIP model with parameters similar to those of *Blaberus discoidalis*, in conjunction with the instability results of the fixed angle leg reset policy in the three dimensional spatial SLIP model, prompt an investigation into alternate leg touchdown protocols. A problem with the fixed angle leg reset policy is the propensity for the model to trip. During each flight phase, the body must attain a height sufficient to place the next leg down at the constant touchdown angle. Many initial conditions produce gaits that do not satisfy this condition, leading to gaps in the mapping where no periodic gaits can exist, as found in [8]. A simple means of eliminating these gaps is to place the next leg down at the previous leg liftoff angle, $\beta_{n+1}^{TD} = \beta_n^{LO}$. Even in the absence of a flight phase, a model utilizing this leg touchdown protocol on a level surface will not stumble, although the forward velocity may be insufficient to compress the elastic spring and move the body forwards past the foot placement point.

Properties of Period Two Gaits with $\boldsymbol{\beta_{n+1}^{TD} = \beta_n^{LO}}$

Implementing a leg touchdown protocol where the new leg touchdown angle equals the previous leg lift-off angle primarily produces asymmetric period two gaits, $\beta_{n+2}^{TD} = \beta_n^{TD} \neq \beta_{n+1}^{TD}$, although period one gaits also exist, due to the symmetry of leg touchdown and lift-off angles in such gaits, $\beta_n^{TD} = \beta_n^{LO}$. We construct a Poincaré map, through the use of conservation laws, to determine necessary conditions for periodic orbits, as well as stability properties of those orbits. Analytically, as in previous studies [11, 8], we assume that leg forces dominate gravity forces during the stance phase. As in [8], we use a mixed approximation in computing the stance map, where we concurrently neglect the effect of gravity during stance to retain conservation of angular momentum, but retain the effect of gravity in computing the velocity mapping to enforce energy conservation. The variations in the leg angle swept during stance, $\Delta\psi$, therefore reduce to those examined in the lateral leg spring model [14, 13], and the analyses conducted therein apply.

We compute the touchdown velocity mapping directly from energy conservation, with the zero potential energy level defined at the initial touchdown height as

$$\frac{m(v_n^{TD})^2}{2} = \frac{m(v_{n+2}^{TD})^2}{2} + mgl(\sin(\beta_{n+2}^{TD}) - \sin(\beta_n^{TD})) \tag{3}$$

$$v_{n+2}^{TD} = \sqrt{(v_n^{TD})^2 + 2gl(\sin(\beta_n^{TD}) - \sin(\beta_{n+2}^{TD}))}\ . \tag{4}$$

We compute the velocity heading angle mapping using conservation of linear momentum, conservation of angular momentum and conservation of energy. Since the next leg touchdown angle equals the previous leg lift-off angle in this protocol, conservation of energy between the lift-off and touchdown conditions necessarily yields $v_{n+1}^{TD} = v_n^{LO}$ and $v_{n+2}^{TD} = v_{n+1}^{LO}$. Conservation of linear momentum during the flight phase

$$v_{n+1}^{TD}\cos(\delta_{n+1}^{TD}) = v_n^{LO}\cos(\delta_n^{LO}) \tag{5}$$

$$v_{n+2}^{TD}\cos(\delta_{n+2}^{TD}) = v_{n+1}^{LO}\cos(\delta_{n+1}^{LO}) \tag{6}$$

therefore yields $\delta_{n+1}^{TD} = -\delta_n^{LO}$ and $\delta_{n+2}^{TD} = -\delta_{n+1}^{LO}$. Since gravity is neglected during stance, angular momentum is conserved during each stance phase

$$ml^2 v_n^{TD}\sin(\beta_n^{TD} - \delta_n^{TD}) = ml^2 v_n^{LO}\sin(\beta_n^{LO} + \delta_n^{LO}) \tag{7}$$

$$ml^2 v_{n+1}^{TD}\sin(\beta_{n+1}^{TD} - \delta_{n+1}^{TD}) = ml^2 v_{n+1}^{LO}\sin(\beta_{n+1}^{LO} + \delta_{n+1}^{LO})\ . \tag{8}$$

Utilizing the relationships for the velocities, velocity heading angles and leg touchdown protocol developed above, however, we find that the magnitude of angular momentum remains constant across all stance phases for this protocol

$$ml^2 v_n^{LO}\sin(\beta_n^{LO} + \delta_n^{LO}) = ml^2 v_{n+1}^{TD}\sin(\beta_{n+1}^{TD} - \delta_{n+1}^{TD}) \tag{9}$$

$$ml^2 v_{n+1}^{LO}\sin(\beta_{n+1}^{LO} + \delta_{n+1}^{LO}) = ml^2 v_{n+2}^{TD}\sin(\beta_{n+2}^{TD} - \delta_{n+2}^{TD})\ . \tag{10}$$

As a result, we find

$$ml^2 v_{n+2}^{TD}\sin(\beta_{n+2}^{TD} - \delta_{n+2}^{TD}) = ml^2 v_n^{TD}\sin(\beta_n^{TD} - \delta_n^{TD}) \tag{11}$$

such that the velocity heading angle map can be expressed as

$$\delta_{n+2}^{TD} = \beta_{n+2}^{TD} - \sin^{-1}\left(\frac{v_n^{TD}}{v_{n+2}^{TD}}\sin(\beta_n^{TD} - \delta_n^{TD})\right)\ . \tag{12}$$

Stance phase geometry and the leg touchdown protocol produce a relationship for the leg touchdown angle mapping as

$$\begin{aligned}\beta_{n+2}^{TD} = \beta_{n+1}^{LO} &= \pi - \varDelta\psi_2 - \beta_{n+1}^{TD}\\ &= \pi - \varDelta\psi_2 - (\pi - \varDelta\psi_1 - \beta_n^{TD})\\ &= \varDelta\psi_1 - \varDelta\psi_2 + \beta_n^{TD}\end{aligned} \tag{13}$$

where $\varDelta\psi_1$ and $\varDelta\psi_2$ represent the leg angle swept during the first and second stance phase respectively. The full period two Poincaré map is therefore comprised by equations (4), (12), and (13), and a periodic orbit for this mapping requires $\varDelta\psi_1 = \varDelta\psi_2$. Ignoring gravity, the formulas for each $\varDelta\psi$ can be constructed from conservation of angular momentum and energy during the stance phase, as in [14], to yield

$$\Delta\psi_1 = \int_{\zeta_{b1}}^{l} \frac{2v_n^{TD} l \sin(\beta_n^{TD} - \delta_n^{TD}) d\zeta}{\zeta\sqrt{((v_n^{TD})^2 - k/m(\zeta - l)^2)\zeta^2 - l^2(v_n^{TD})^2 \sin^2(\beta_n^{TD} - \delta_n^{TD})}} \tag{14}$$

$$\Delta\psi_2 = \int_{\zeta_{b2}}^{l} \frac{2v_{n+1}^{TD} l \sin(\beta_{n+1}^{TD} - \delta_{n+1}^{TD}) d\zeta}{\zeta\sqrt{(a - k/m(\zeta - l)^2)\zeta^2 - l^2(v_{n+1}^{TD})^2 \sin^2(\beta_{n+1}^{TD} - \delta_{n+1}^{TD})}} \tag{15}$$

$$a = (v_{n+1}^{TD})^2 + 2gl(\sin(\beta_{n+1}^{TD}) - \sin(\beta_n^{TD})) \tag{16}$$

where ζ_{b1}, ζ_{b2} are the largest positive roots of the equation(s)

$$ml^2(v_n^{TD})^2 \sin^2(\beta_n^{TD} - \delta_n^{TD}) + k(\zeta_{b1} - l)^2\zeta_{b1}^2 - m\zeta_{b1}^2(v_n^{TD})^2 = 0 \tag{17}$$

$$\frac{ml^2(v_{n+1}^{TD})^2 \sin^2(\beta_{n+1}^{TD} - \delta_{n+1}^{TD}) + k(\zeta_{b2} - l)^2\zeta_{b2}^2}{m\zeta_{b2}^2[(v_{n+1}^{TD})^2 + 2gl(\sin(\beta_{n+1}^{TD}) - \sin(\beta_n^{TD}))]} = 1 \; . \tag{18}$$

Here, ζ_{b1}, ζ_{b2} represent the distance between the center of mass and the foot placement point when $\dot{\zeta} = 0$. For the point mass system, in which $\zeta = \eta$, this represents the state of maximal spring compression. For consistency, gravity is included in the energy calculation at leg touchdown, but not during the stance phase, in the formulation of $\Delta\psi_2$. Conservation of energy between the start of the first and second stance phases yields $a = (v_n^{TD})^2$, and from equations (7–9), we find $v_{n+1}^{TD} \sin(\beta_{n+1}^{TD} - \delta_{n+1}^{TD}) = v_n^{TD} \sin(\beta_n^{TD} - \delta_n^{TD})$. Substituting these values into the expression for $\Delta\psi_2$ in (15) shows that the integrands of (14-15) are equivalent. Similar arguments can be used to show that equations (17) and (18) are equal, resulting in $\zeta_{b2} = \zeta_{b1}$. By ignoring gravity during stance, we find that $\Delta\psi_1$ necessarily equals $\Delta\psi_2$ such that the period two mapping reduces to

$$v_{n+2}^{TD} = v_n^{TD} \tag{19}$$

$$\delta_{n+2}^{TD} = \delta_n^{TD} \tag{20}$$

$$\beta_{n+2}^{TD} = \beta_n^{TD} \; . \tag{21}$$

Therefore, under these assumptions, this leg touchdown protocol necessarily produces period two gaits. The eigenvalues of the mapping are all unity, indicating that these gaits are neutrally stable. Numerical simulations verify this result for a large number of gaits, even when gravity is included during the stance phase. The limited effect of gravity during the stance phase observed here is restricted to this particular leg touchdown protocol and model. Specifically, this leg touchdown protocol ensures, through conservation of energy and conservation of linear momentum, that orbits of the system are reflection-symmetric about the midpoint of the flight phase. As a result, while angular momentum is not conserved during either stance phase, the net angular impulse delivered by gravity during the first stance phase is counteracted by an equal and opposite net angular impulse during the second stance phase. For other leg touchdown protocols, especially those relating to period one orbits, gravity can have a significant effect on orbits and orbital stability, as discussed in [8, 18].

Properties of Period one Gaits with $\beta_{n+1}^{TD} = \beta_n^{LO}$

Neutrally stable period one gaits, with $\beta_{n+1}^{TD} = \beta_n^{TD} = \beta_n^{LO}$, exist as a subset of the family of period two gaits analyzed previously. The eigenvalues of the period two mapping in this case represent the square of each eigenvalue in the period one map, suggesting that eigenvalues in the associated period one mapping are ± 1. We pursue a stability analysis in this section to determine if one of the eigenvalues is negative one for this leg touchdown protocol. If an eigenvalue is negative one, then continuity suggests that a leg touchdown protocol exists between the extremes of $\beta_{n+1}^{TD} = \beta_n^{TD}$ and $\beta_{n+1}^{TD} = \beta_n^{LO}$ that produces stable periodic gaits.

The period one Poincaré map for the leg touchdown protocol $\beta_{n+1}^{TD} = \beta_n^{LO}$ is a composition of the stance and flight phase maps. The full stride map is constructed similarly to the period two map, using conservation of energy, conservation of linear momentum and conservation of angular momentum to yield

$$\begin{aligned} v_{n+1}^{TD} &= \sqrt{(v_n^{TD})^2 + 2gl(\sin(\beta_n^{TD}) - \sin(\Delta\psi + \beta_n^{TD}))} && (22) \\ \delta_{n+1}^{TD} &= \pi - \Delta\psi - 2\beta_n^{TD} + \delta_n^{TD} && (23) \\ \beta_{n+1}^{TD} &= \pi - \Delta\psi - \beta_n^{TD}\ . && (24) \end{aligned}$$

As in [8], consistency requires that we neglect gravity in computing the liftoff velocity in the heading angle map, since gravity is neglected in the computation of the leg sweep angle, $\Delta\psi$, during stance. Neglecting gravity in this computation yields $v_n^{LO} = v_n^{TD}$, resulting in the heading angle map presented above. A period one orbit therefore requires $\Delta\psi = \pi - 2\beta_n^{TD}$, which implies through the stance phase geometry that $\beta_n^{LO} = \beta_n^{TD}$. We construct the Jacobian of the Poincaré map evaluated at the fixed point as

$$\begin{bmatrix} 1 + b\frac{\partial\Delta\psi}{\partial v_n^{TD}} & b\frac{\partial\Delta\psi}{\partial\delta_n^{TD}} & b(2 + \frac{\partial\Delta\psi}{\partial\beta_n^{TD}}) \\ -\frac{\partial\Delta\psi}{\partial v_n^{TD}} & 1 - \frac{\partial\Delta\psi}{\partial\delta_n^{TD}} & -(2 + \frac{\partial\Delta\psi}{\partial\beta_n^{TD}}) \\ -\frac{\partial\Delta\psi}{\partial v_n^{TD}} & -\frac{\partial\Delta\psi}{\partial\delta_n^{TD}} & -1 - \frac{\partial\Delta\psi}{\partial\beta_n^{TD}} \end{bmatrix} \qquad (25)$$

where

$$b = \frac{gl\cos(\beta_n^{TD})}{v_n^{TD}}\ . \qquad (26)$$

We calculate the eigenvalues of the Jacobian, evaluated at the fixed point, in the Appendix as

$$\lambda_{1,2} = 1 \tag{27}$$

$$\lambda_3 = -1 - \frac{\partial \Delta\psi}{\partial \beta_n^{TD}} - \frac{\partial \Delta\psi}{\partial \delta_n^{TD}} + b\frac{\partial \Delta\psi}{\partial v_n^{TD}} \tag{28}$$

where $\Delta\psi$ is computed as in equation (14).

While ignoring gravity during the stance phase enables $\Delta\psi$ to be computed analytically, as in [14], it results in a complex expression in terms of incomplete elliptic integrals. As in [13], we instead utilize the Schwind-Koditschek approximation [18] to compute $\Delta\psi$ and the associated derivatives. Using this approximation, the computation of $\Delta\psi$ can be approximated by

$$\Delta\psi = \frac{2lv_n^{TD}\sin(\beta_n^{TD} - \delta_n^{TD})(l - \zeta_b)}{\hat{\zeta}\sqrt{((v_n^{TD})^2 - k/m(\hat{\zeta} - l)^2)\hat{\zeta}^2 - l^2(v_n^{TD})^2\sin^2(\beta_n^{TD} - \delta_n^{TD})}} \tag{29}$$

$$\hat{\zeta} = \frac{3\zeta_b + l}{4} \, . \tag{30}$$

While straightforward, computing the derivatives required in the eigenvalue expression is lengthy and left to the Appendix. Using the fixed points computed for the gait family with $\beta_n^{TD} = 1.2$, we numerically compute the third eigenvalue using equation (28) and the relationships for the derivatives detailed in the Appendix. This computation reveals that the third eigenvalue has a maximum deviation of 10% from -1 across all the gaits of the gait family. Purely numerical computation of the Jacobian and associated eigenvalues for each gait reveal that the third eigenvalue is equal to -1 in each case. The difference between the numerical results and our analytical approximation results from neglecting gravity during stance as well as the approximation used for the computation of $\Delta\psi$ and the associated derivatives. However, taken together, these results show that periodic gaits utilizing this leg touchdown protocol are neutrally stable, with one eigenvalue equal to negative one.

4 An Adaptive Control Law

The results of the analyses for the period two and period one orbits under the leg touchdown protocol $\beta_{n+1}^{TD} = \beta_n^{LO}$ are used to guide the construction of an adaptive control law for period one orbits.

4.1 Control of the Point Mass SLIP Model

We begin by deriving a general period one Poincaré map with varying leg touchdown angle, using conservation of energy, conservation of linear momentum during the flight phase, and conservation of angular momentum during the stance phase as

$$v_{n+1}^{TD} = \sqrt{(v_n^{TD})^2 + 2gl(\sin(\beta_n^{TD}) - \sin(\beta_{n+1}^{TD}))} \tag{31}$$

$$\cos(\delta_{n+1}^{TD}) = \sqrt{\frac{(v_n^{TD})^2 + 2gl(\sin(\beta_n^{TD}) - \sin(\beta_n^{LO}))}{(v_n^{TD})^2 + 2gl(\sin(\beta_n^{TD}) - \sin(\beta_{n+1}^{TD}))}} \cos(g(\cdots)) \tag{32}$$

$$\beta_{n+1}^{TD} = f(\beta_n^{TD}, \beta_n^{LO}, v_n^{TD}, \delta_n^{TD}) \tag{33}$$

where

$$g(\beta_n^{TD}, v_n^{TD}, \delta_n^{TD}) = \beta_n^{LO} + \sin^{-1}(\frac{v_n^{TD}}{v_n^{LO}} \sin(\delta_n^{TD} - \beta_n^{TD})) \tag{34}$$

$$\beta_n^{LO} = \pi - \Delta\psi - \beta_n^{TD} \tag{35}$$

$$v_n^{LO} = \sqrt{(v_n^{TD})^2 + 2gl(\sin(\beta_n^{TD}) - \sin(\beta_n^{LO}))}\ . \tag{36}$$

While a period one orbit of the mapping must satisfy $\beta_{n+1}^{TD} = \beta_n^{TD} = \beta_n^{LO}$ and $\Delta\psi = \pi - 2\beta_n^{TD}$, the stability of an orbit depends upon the leg touchdown protocol utilized in response to perturbations from the periodic orbit. Using the fixed angle leg reset policy ($\beta_{n+1}^{TD} = \beta_n^{TD}$) with model parameters similar to those of *Blaberus discoidalis* primarily produces unstable gaits with a single eigenvalue greater than unity, whereas using the leg touchdown protocol $\beta_{n+1}^{TD} = \beta_n^{LO}$ necessarily produces neutrally stable periodic gaits with an eigenvalue of -1. Continuity therefore suggests that the stability of the periodic gait should vary continuously for leg placement protocols between these two extremes. An adaptive control law that incorporates both of these leg touchdown protocols and satisfies the leg angle symmetry condition for a periodic orbit is given by

$$\beta_{n+1}^{TD} = \beta_n^{LO} + c(\beta_n^{TD} - \beta_n^{LO}) \tag{37}$$

where c is an arbitrary constant. Since gait symmetry requires $\beta_n^{TD} = \beta_n^{LO}$ for a periodic gait, c only changes the stability of a gait, since a periodic gait remains periodic for any value of c. In particular, we expect that the unstable gaits observed in the SLIP model will stabilize as c is decreased from unity (the fixed angle leg reset policy, $\beta_n^{TD} = \beta_n^{LO}$), since the unstable eigenvalue must enter the unit circle and tend towards -1 as c tends towards zero (the $\beta_{n+1}^{TD} = \beta_n^{LO}$ protocol).

We investigate the stability properties of periodic gaits utilizing this feedback control law using numerically computed periodic orbits and eigenvalues, with model parameters set to values similar to those of *Blaberus discoidalis*. We begin by analyzing the effect changing c has on the eigenvalues of a representative gait family, as illustrated in Fig. 3. As illustrated in the second panel, decreasing c from unity shifts the non-unity eigenvalue curve downwards, stabilizing an increasing number of gaits until almost all gaits are stabilized for $c = 0.1$. We note, however, that as c decreases, some gaits, while still stable, will be relatively less stable as the eigenvalue determining stability tends towards negative one. This is illustrated in the third panel for a single orbit,

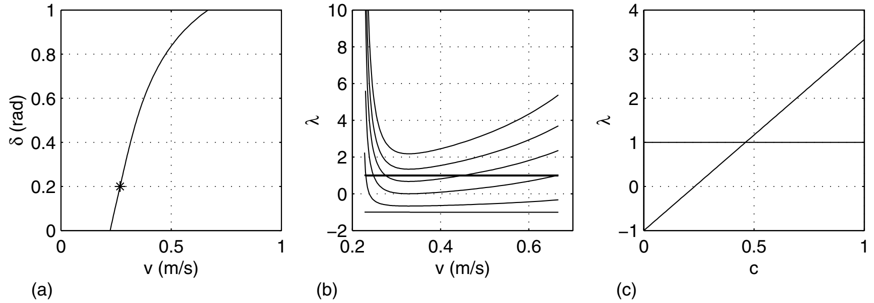


Fig. 3. (**a**) SLIP periodic gait family and stability for $\beta = 1.2$, with other parameters set to those of *Blaberus discoidalis*, as described in the text (**b**) Gait family eigenvalues for $c = 1, 0.7, 0.5, 0.3, 0.1, 0$, with curves moving downwards as c decreases. (**c**) Eigenvalue variation as c varies from 1 to 0 for the periodic gait (*) identified in (a)

denoted with a * in the first panel, where we observe that an almost linear eigenvalue variation occurs with changes in c. As c decreases from unity, the unstable gait becomes less unstable, stabilizes for $c < .46$, and becomes more stable until c reaches 0.23, at which point the eigenvalue begins to grow in magnitude as it approaches -1 at $c = 0$. As c passes through zero, we find that the eigenvalue passes through -1, indicating a flip bifurcation. Therefore, for $c < 0$, we find that the period one orbit once again becomes unstable.

We illustrate the performance of the control law, for $c = 0.3$, in Fig. 4. While a fixed angle leg reset policy is incapable of producing a stable periodic orbit with these model parameters, we see that the inclusion of this simple control law leads to stabilization to a periodic orbit within several stance phases. How quickly a new stable gait is obtained depends upon how close the eigenvalue is to zero, which depends upon c and the model parameters. We note that variations in β^{TD} utilized in stabilizing the orbit remain quite small.

It is important to clarify the definition of asymptotic stability that is being utilized in this context. Since energy conservation and translational invariance naturally produce unity eigenvalues, we apply the definition of asymptotic stability used by Coleman et al. [19] and Coleman and Holmes [20]. In this less restrictive definition of asymptotic stability, a periodic orbit is asymptotically stable if perturbations result in the convergence to a nearby periodic gait. Perturbations to a periodic gait for our system and control result in the attainment of a new, stable gait, due to the partial asymptotic stability of these gaits and the coupling of motions introduced through the inclusion of the control law.

Since the model remains conservative under this control law, we also illustrate the changes in gait stability for a constant energy surface, as illustrated in Fig. 5. The gait family illustrated therein is determined by choosing an en-

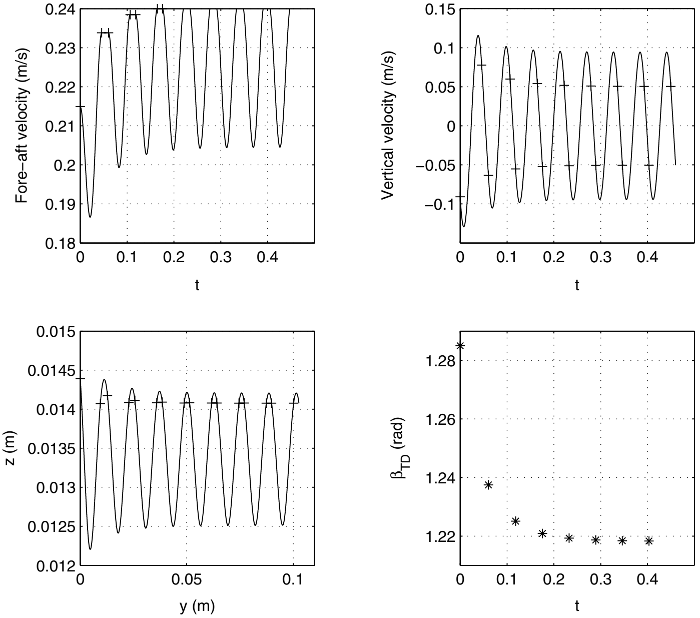


Fig. 4. Stabilization of the SLIP model to a periodic gait for $c = 0.3$, with parameters are set to values similar to the cockroach *Blaberus discoidalis*. *Plus signs* denote the start and end of the stance phase and stars denote the start of the stance phase

ergy value and varying β_n^{TD}, thereby implicitly determining v_n^{TD}, and finding the associated δ_n^{TD} that determines a periodic orbit by a Newton-Raphson iteration. The gaits for this constant energy represent the periodic orbits which the model can settle on, assuming no perturbations occur that change the energy of the system. As c decreases, we see that an increasing number of gaits stabilize, until all gaits are stable at $c = 0.1$. To be clear, however, the definition of asymptotic stability utilized in this work means that perturbations to a particular periodic orbit that do not result in a change in energy do result in the state asymptotically converging to a nearby periodic orbit of the gait family. Obviously, perturbations to a conservative system that result in a change in the total system energy necessarily result in the convergence to a new periodic gait that belongs to a different gait family. Perturbations that would typically be encountered in practice would tend to fall into this latter category rather than the former. In fact, Altendorfer [21] notes that the small range of touchdown speeds evidenced for a constant energy surface necessitates changes in the total system energy if the controlled system is to exhibit a useful range of touchdown velocities.

Achieving asymptotic stability in the more traditional sense to perturbations within the energy surface (i.e. returning to the original periodic orbit) requires a measure of knowledge of the periodic orbit that we wish to stabilize.

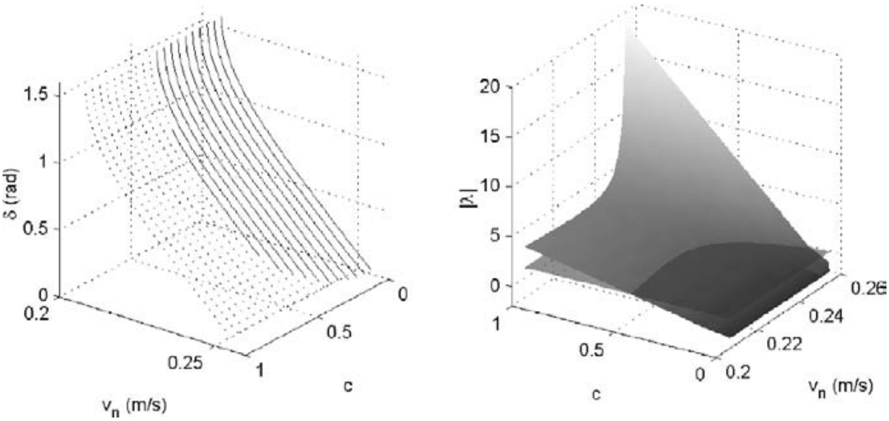


Fig. 5. Periodic gaits and stability for the SLIP model as a function of c, for a constant energy surface corresponding to an insect running at 0.25 m/s at a height of .014 m. All other parameters are similar to those of *Blaberus discoidalis*, as described in the text. (**a**) Periodic gaits for $c = 0.0 - 1.0$, with *dotted* and *solid lines* denoting unstable and stable periodic gaits respectively (**b**) Eigenvalue variation for the periodic gaits of panel (a)

The fixed angle leg reset gaits that exhibit this property for larger animals accomplish this naturally since the leg touchdown protocol defines a desired leg touchdown angle and therefore a desired touchdown velocity, assuming a constant energy. We note that we have constructed a control law similar in structure and content to that presented in this work which includes a dependence on a desired touchdown angle, β^{TD}_{des}. This alternate control law produces asymptotically stable periodic gaits in the more traditional sense, as explained above. The performance of this alternate control law will be investigated in detail in a future work.

Finally, returning to the original gait family plot of Fig. 2, we illustrate how this control law enlarges the basin of attraction, as illustrated in Fig. 6. As c decreases, an increasing number of periodic gaits on all gait family curves stabilize. Considering that perturbations to an orbit will typically occur transverse to the constant energy surface, this control law therefore enables the model to successfully recover from a large range of perturbations. As a result, implementing this control expands the stability basin of the SLIP model.

4.2 Control of the Rigid Body SLIP Model

Displacing the leg attachment point from the center of mass in the SLIP model couples the translational and rotational dynamics and introduces pitching. Use of the same control law formulation, with leg angles defined in the inertial frame, is examined briefly in this section. Preliminary simulations of the rigid body SLIP model suggest that the control law implementation stabilizes

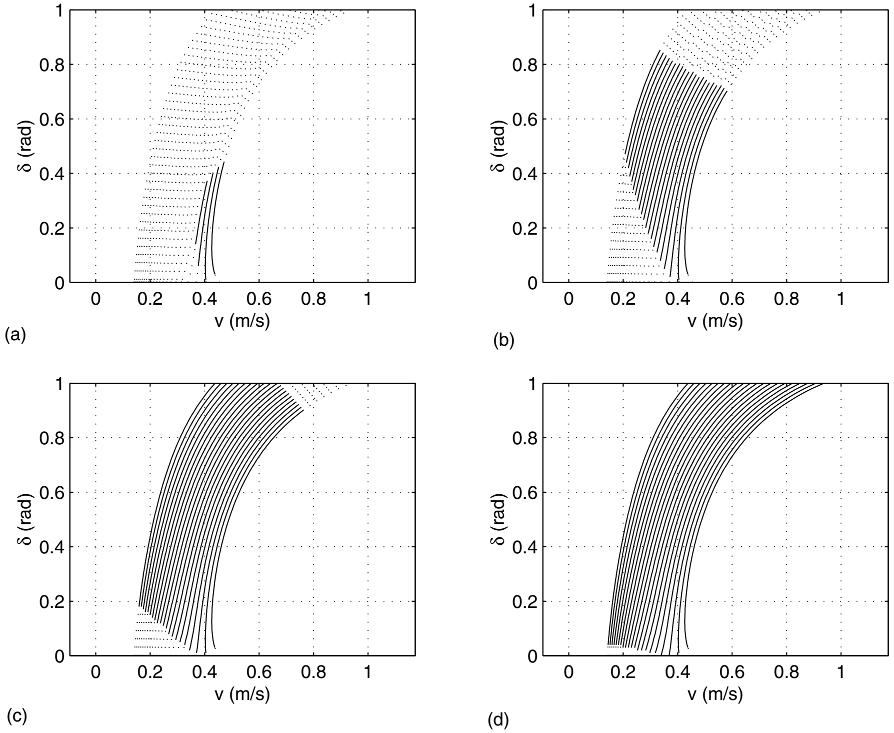


Fig. 6. SLIP gait families for $\beta_n^{TD} = 1.1 - 1.3$ for (**a**) $c = 0.7$ (**b**) $c = 0.5$ (**c**) $c = 0.3$ and (**d**) $c = 0.1$. Gait stability is indicated by *dotted* and *solid lines* for unstable and stable gaits respectively. Model parameters used in calculating the gaits are similar to those of *Blaberus discoidalis*, as described in the text

the system, but does not necessarily produce period one gaits, since the quasi-periodic gaits observed in [8] also appear, as illustrated in Fig. 7. However, the recurrent nature of quasi-periodic orbits enables the use of chaos control methods to enforce stabilization to period one gaits. Specifically, a delay feedback controller [22] is implemented to enforce stabilization to period one orbits. Delay feedback control incorporates the difference between the value of a state variable at one instant and one period delayed, such that the control effect vanishes when the periodic orbit is attained. In our system, delay feedback control is implemented by including a dependence in the control law on the pitch angle as follows

$$\beta_{n+1}^{TD} = \beta_n^{LO} + c(\beta_n^{TD} - \beta_n^{LO}) + c_2(\theta_n^{LO} - \theta_{n-1}^{LO}) \ . \tag{38}$$

The effect of the control law and the control law with delay feedback included is illustrated in Figs. 7 and 8. In both cases, the rigid body SLIP model is simulated from the same set of initial conditions ($v = 0.25$, $\delta = 0.2$, $\theta = 0$, $\dot{\theta} = -0.2$). Figure 7 illustrates that while the inclusion of the leg angle control does

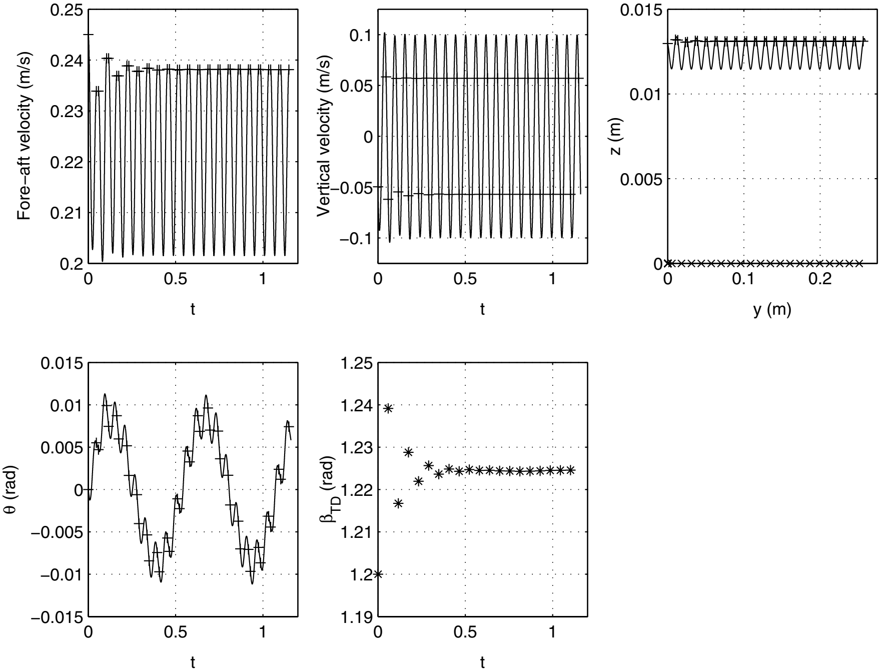


Fig. 7. Rigid body SLIP simulation with the standard leg angle feedback control implemented, with $d = 0.001$, $c = 0.1$, $I = 1.86 \times 10^{-7}$ and initial conditions as specified in the text. Model parameters used in calculating the gaits are similar to those of *Blaberus discoidalis*, as described in the text. *Plus signs* (+) denote the start and end of each stance phase, whereas *stars* (*) denote the start of each stance phase

stabilize the system, the resulting orbit is quasi-periodic rather than period one. While period one gaits may be obtained through the use of this control scheme, they are not necessarily achieved. Conversely, Fig. 8 illustrates the effect of the control law with delay feedback control included. As illustrated, the system not only stabilizes, but stabilizes to a period one orbit. Simulations initiated from a wide range of initial conditions provide similar results. As in previous analyses, the translational and rotational coupling present in the equations of motion, as well as in the control law, lead to the attainment of a new gait in response to perturbations. While further investigation is required to quantify the effects of these control laws on the stability of the rigid body SLIP model, these preliminary results suggest that the control laws presented here will expand the very small basin of attraction identified for this model in previous studies [8].

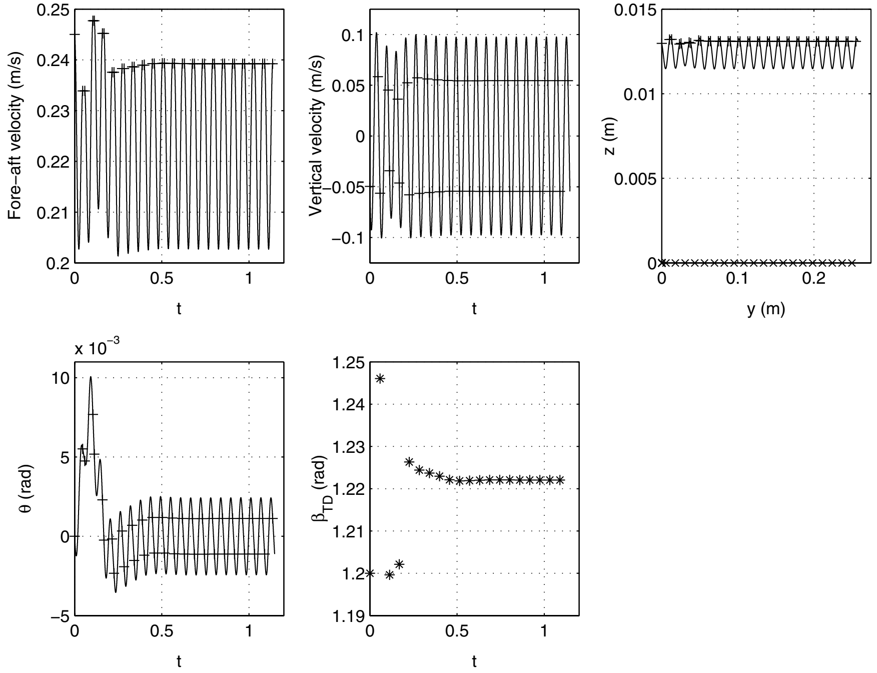


Fig. 8. Rigid body SLIP simulation with delayed feedback control, with $d = 0.001$, $c = 0.1$, $c_2 = 1.25$, $I = 1.86 \times 10^{-7}$ and initial conditions as specified in the text. Model parameters used in calculating the gaits are similar to those of *Blaberus discoidalis*, as described in the text. *Plus signs* (+) denote the start and end of each stance phase, whereas *stars* (*) denote the start of each stance phase

5 Conclusion

In this work, we investigate the applicability of the SLIP model to insect locomotion, specifically that of the cockroach *Blaberus discoidalis*. We find that unlike previous results obtained for the SLIP model when applied to larger animals, periodic gaits produced for these model parameters remain largely unstable over a wide range of leg touchdown angles. This, in conjunction with results indicating that a fixed angle leg reset policy employed in a spatial SLIP model produces only unstable gaits, prompts an investigation into alternate leg touchdown protocols and their effects on gait stability. In particular, we show that a leg angle touchdown protocol that places the next leg down at the previous leg lift-off angle, $\beta_{n+1}^{TD} = \beta_n^{LO}$, necessarily produces neutrally stable period two gaits, and is capable of producing neutrally stable period one gaits with an eigenvalue equal to negative one. Continuity arguments between this leg touchdown protocol and the fixed angle leg reset protocol are used to develop a feedback control law based on inertial leg touchdown angles that stabilizes these unstable periodic gaits. Numerically computed gait families verify that lowering the parameter c in this control law increases the number of stable periodic gaits, therefore expanding the basin of stability. Implementing the same control law in a rigid body SLIP formulation also produces stable gaits, but not necessarily periodic gaits, since quasi-periodic gaits also appear. Utilizing the pitch angle as delay feedback control in the control law forces stabilization to period one orbits.

The control laws developed in this work are unique in that they: a) do not require knowledge of pre-existing periodic orbits or linearization about those orbits to achieve control b) require relatively simple feedback measurements rather than full state feedback for implementation c) apply control once per stance phase rather than continuously during the stance phase and d) can adapt to perturbations by changing to a different stable periodic gait that is more suitable to the new environment. Qualitatively, it appears that the effectiveness of the control law results from implicit information about the system angular momentum that is present in a reading of the leg lift-off angle. Quantitative exploration of this hypothesis will be conducted in future work. Additionally, it appears that these control laws may also have applicability in both the horizontal plane and spatial SLIP models. Preliminary simulations utilizing two control laws specifying the leg placement angles of a point mass spatial SLIP model appear to, at least for some parameter ranges, produce stable periodic gaits. We plan to investigate these applications further in later works.

References

[1] R. Blickhan. The spring-mass model for running and hopping. *J. Biomechanics*, 11/12:1217–1227, 1989.

[2] R. Blickhan and R.J. Full. Similarity in multi-legged locomotion: bouncing like a monopode. *J. Comp. Physiol. A*, 173:509–517, 1993.

[3] G.A. Cavagna, N.C. Heglund, and C.R. Taylor. Mechanical work in terrestrial locomotion: two basic mechanisms for minimizing energy expenditure. *Am. J. Physiol.*, 233 (5):R243–R261, 1977.

[4] T.A. McMahon and G.C. Cheng. The mechanics of running: how does stiffness couple with speed? *J. Biomechanics*, 23 (suppl 1):65–78, 1990.

[5] R.J. Full and M.S. Tu. Mechanics of a rapid running insect: two-, four- and six-legged locomotion. *J. Exp. Biol.*, 156:215–231, 1991.

[6] C.T. Farley, J. Glashenn, and T.A. McMahon. Running springs: speed and animal size. *J. Exp. Biol.*, 185:71–86, 1993.

[7] R.J. Full, R. Blickhan, and L.H. Ting. Leg design in hexpedal runners. *J. Exp. Biol.*, 158:369–390, 1991.

[8] R. Ghigliazza, R.M. Altendorfer, P. Holmes, and D. Koditschek. A simply stabilized running model. *SIAM Journal on Applied Dynamical Systems*, 2(2):187–218, 2003.

[9] A. Seyfarth, H. Geyer, M. Gunther, and R. Blickhan. A movement criterion for running. *J. of Biomechanics*, 35:649–655, 2002.

[10] A. Seyarth, H. Geyer, and H. Herr. Swing-leg retraction: a simple control model for stable running. *J. Exp. Biology*, 206:2547–2555, 2003.

[11] J. Seipel and P. Holmes. Running in three dimensions: analysis of a point-mass sprung-leg model. *Intl. J. Robotics Research*, 24(8), 2005.

[12] A. Ruina. Non-holonomic stability aspects of piecewise holonomic systems. *Reports on Mathematical Physics*, 42(1/2):91–100, 1998.

[13] J. Schmitt and P. Holmes. Mechanical models for insect locomotion: Stability and parameter studies. *Physica D*, 156(1–2):139–168, 2001.

[14] J. Schmitt and P. Holmes. Mechanical models for insect locomotion: Dynamics and stability in the horizontal plane – Theory. *Biological Cybernetics*, 83(6):501–515, 2000.

[15] J. Schmitt and P. Holmes. Mechanical models for insect locomotion: Dynamics and stability in the horizontal plane – Application. *Biological Cybernetics*, 83(6):517–527, 2000.

[16] J. Guckenheimer and P. Holmes. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields.* Springer-Verlag, New York, NY, 1990.

[17] L.H. Ting, R. Blickhan, and R.J. Full. Dynamic and static stability in hexapedal runners. *J. Exp. Biol.*, 197:251–269, 1994.

[18] W.J. Schwind and D.E. Koditschek. Approximating the stance map of a 2 dof monoped runner. *J. Nonlinear Science*, 10(5):533–568, 2000.

[19] M. Coleman, A. Chatterjee, and A. Ruina. Motions of a rimless spoked wheel: a simple 3d system with impacts. *Dynamics and stability of systems*, 12(3):139–169, 1997.

[20] M. Coleman and P. Holmes. Motions and stability of a piecewise holonomic system: the discrete chaplygin sleigh. *Regul. Chaotic Dyn.*, 4:55–77, 1999.

[21] R. Altendorfer, R. Ghigliazza, P. Holmes, and D. Koditschek. Exploiting passive stability for hierarchical control. In *Proceedings of the Fifth International Conference on Climbing and Walking Robots (CLAWAR 2002)*, pp. 81–85, London, 2002. Professional Engineering Publishing Limited.

[22] K. Pyragas. Continuous control of chaos by self-controlling feedback. *Phys. Lett. A*, 170:421–428, 1992.

Appendix

The eigenvalues of the period one Poincaré map are determined from $\det(\lambda I - DP)$, where I is the identity matrix and DP is the Jacobian matrix of (25). Evaluating the determinant and simplifying yields

$$\begin{aligned}&(\ \lambda -1 - b\frac{\partial \Delta\psi}{\partial v_n^{TD}})(\lambda^2 + (\frac{\partial \Delta\psi}{\partial \beta_n^{TD}} + \frac{\partial \Delta\psi}{\partial \delta_n^{TD}})\lambda - 1 - \frac{\partial \Delta\psi}{\partial \beta_n^{TD}} - \frac{\partial \Delta\psi}{\partial \delta_n^{TD}}) + \\ &(\ \lambda -1)b\frac{\partial \Delta\psi}{\partial v_n^{TD}}\frac{\partial \Delta\psi}{\partial \delta_n^{TD}} + (\lambda - 1)(2 + \frac{\partial \Delta\psi}{\partial \beta_n^{TD}})b\frac{\partial \Delta\psi}{\partial v_n^{TD}} = 0\ . \qquad \text{(A-1)}\end{aligned}$$

Factoring a $(\lambda - 1)$ out of all terms yields

$$
\begin{aligned}
(\lambda -1)((\lambda-1-b\frac{\partial\Delta\psi}{\partial v_n^{TD}})(\lambda+1+\frac{\partial\Delta\psi}{\partial\beta_n^{TD}}+\frac{\partial\Delta\psi}{\partial\delta_n^{TD}})+\\
b\,\frac{\partial\Delta\psi}{\partial\delta_n^{TD}}\frac{\partial\Delta\psi}{\partial v_n^{TD}}+2b\frac{\partial\Delta\psi}{\partial v_n^{TD}}+b\frac{\partial\Delta\psi}{\partial\beta_n^{TD}}\frac{\partial\Delta\psi}{\partial v_n^{TD}})=0\ .
\end{aligned}
\tag{A-2}
$$

Expanding further and simplifying yields

$$
(\lambda-1)^2(\lambda+1+\frac{\partial\Delta\psi}{\partial\beta_n^{TD}}+\frac{\partial\Delta\psi}{\partial\delta_n^{TD}}-b\frac{\partial\Delta\psi}{\partial v_n^{TD}})=0 \tag{A-3}
$$

which yields the expression for the eigenvalues in (27–28).

From (28), it is clear that we need to calculate $\frac{\partial\Delta\psi}{\partial\beta_n^{TD}}$, $\frac{\partial\Delta\psi}{\partial v_n^{TD}}$, $\frac{\partial\Delta\psi}{\partial\delta_n^{TD}}$, in order to determine the value for the relevant eigenvalue that determines stability. The approximation of $\Delta\psi$ presented in (29–30) can be simplified further as

$$
\begin{aligned}
\Delta\psi &= \frac{p}{q} && \text{(A-4)}\\
p &= 128 l v_n^{TD}\sin(\beta_n^{TD}-\delta_n^{TD})(l-\zeta_b) && \text{(A-5)}\\
q &= (3\zeta_b+l)\sqrt{s} && \text{(A-6)}\\
s &= (16(v_n^{TD})^2-\frac{9k}{m}(\zeta_b-l)^2)(3\zeta_b+l)^2 && \text{(A-7)}\\
&\quad -256 l^2(v_n^{TD})^2\sin^2(\beta_n^{TD}-\delta_n^{TD})\ .
\end{aligned}
$$

Since $\Delta\psi$ is a function of ζ_b, in evaluating the required partial derivatives, we need $\frac{\partial\zeta_b}{\partial v_n^{TD}}$, $\frac{\partial\zeta_b}{\partial\delta_n^{TD}}$, $\frac{\partial\zeta_b}{\partial\beta_n^{TD}}$, all of which may be determined implicitly from (17) as

$$
\frac{\partial\zeta_b}{\partial v_n^{TD}} = \frac{m v_n^{TD}(\zeta_b^2-l^2\sin^2(\beta_n^{TD}-\delta_n^{TD}))}{\zeta_b(-m(v_n^{TD})^2+k(2\zeta_b-l)(\zeta_b-l))} \tag{A-8}
$$

$$
\frac{\partial\zeta_b}{\partial\delta_n^{TD}} = \frac{m l^2(v_n^{TD})^2\sin(2(\beta_n^{TD}-\delta_n^{TD}))}{2\zeta_b(-m(v_n^{TD})^2+k(2\zeta_b-l)(\zeta_b-l))} \tag{A-9}
$$

$$
\frac{\partial\zeta_b}{\partial\beta_n^{TD}} = -\frac{m l^2(v_n^{TD})^2\sin(2(\beta_n^{TD}-\delta_n^{TD}))}{2\zeta_b(-m(v_n^{TD})^2+k(2\zeta_b-l)(\zeta_b-l))}\ . \tag{A-10}
$$

Given these partial derivatives, we can calculate the expressions for the required derivatives of $\Delta\psi$ from the quotient rule

$$
\frac{\partial\Delta\psi}{\partial()} = \frac{q\dot{p}-p\dot{q}}{q^2} \tag{A-11}
$$

where () represents $v_n^{TD},\delta_n^{TD},\beta_n^{TD}$ and $\dot{()}$ denotes derivatives with respect to these variables, which can be expressed as

$$
\frac{\partial p}{\partial v_n^{TD}} = 128 l\sin(\beta_n^{TD}-\delta_n^{TD})((l-\zeta_b)-v_n^{TD}\frac{\partial\zeta_b}{\partial v_n^{TD}}) \tag{A-12}
$$

$$\frac{\partial p}{\partial \delta_n^{TD}} = -128 l v_n^{TD}(\cos(\beta_n^{TD} - \delta_n^{TD})(l - \zeta_b) + \frac{\partial \zeta_b}{\partial \delta_n^{TD}} \sin(\beta_n^{TD} - \delta_n^{TD})) \tag{A-13}$$

$$\frac{\partial p}{\partial \beta_n^{TD}} = 128 l v_n^{TD}(\cos(\beta_n^{TD} - \delta_n^{TD})(l - \zeta_b) - \frac{\partial \zeta_b}{\partial \beta_n^{TD}} \sin(\beta_n^{TD} - \delta_n^{TD})) \tag{A-14}$$

and

$$\frac{\partial q}{\partial v_n^{TD}} = 3\frac{\partial \zeta_b}{\partial v_n^{TD}}\sqrt{s} + \frac{(3\zeta_b + 1)(t\frac{\partial \zeta_b}{\partial v_n^{TD}} + w - u\sin(\beta_n^{TD} - \delta_n^{TD}))}{\sqrt{s}} \tag{A-15}$$

$$\frac{\partial q}{\partial \delta_n^{TD}} = 3\frac{\partial \zeta_b}{\partial \delta_n^{TD}}\sqrt{s} + \frac{(3\zeta_b + 1)(t\frac{\partial \zeta_b}{\partial \delta_n^{TD}} - x\frac{\partial \zeta_b}{\partial \delta_n^{TD}} + u v_n^{TD}\cos(\beta_n^{TD} - \delta_n^{TD}))}{\sqrt{s}} \tag{A-16}$$

$$\frac{\partial q}{\partial \beta_n^{TD}} = 3\frac{\partial \zeta_b}{\partial \beta_n^{TD}}\sqrt{s} + \frac{(3\zeta_b + 1)(t\frac{\partial \zeta_b}{\partial \beta_n^{TD}} - x\frac{\partial \zeta_b}{\partial \beta_n^{TD}} - u v_n^{TD}\cos(\beta_n^{TD} - \delta_n^{TD}))}{\sqrt{s}} \tag{A-17}$$

where

$$t = 6(3\zeta_b + l)(16(v_n^{TD})^2 - \frac{9k}{m}(\zeta_b - l)^2) \tag{A-18}$$

$$u = 512 l^2 v_n^{TD}\sin(\beta_n^{TD} - \delta_n^{TD}) \tag{A-19}$$

$$w = (32 v_n^{TD} - \frac{18k}{m}(\zeta_b - l)\frac{\partial \zeta_b}{\partial v_n^{TD}})(3\zeta_b + l)^2 \tag{A-20}$$

$$x = \frac{18k}{m}(\zeta_b - l)(3\zeta_b + 1)^2 \ . \tag{A-21}$$

Running and Walking with Compliant Legs

A. Seyfarth[1], H. Geyer[1], R. Blickhan[2], S. Lipfert[1], J. Rummel[1],
Y. Minekawa[1], and F. Iida[1,3]

[1] Locomotion Laboratory, University of Jena, Dornburger Str. 23, D-07743
Jena, Germany
www.lauflabor.de

[2] Science of Motion, University of Jena, Seidelstr. 20, D-07749 Jena, Germany

[3] Artificial Intelligence Laboratory, University of Zurich, Andreasstrasse 15,
CH-8050 Zurich, Switzerland

It has long been the dream to build robots which could walk and run with ease. To date, the stance phase of walking robots has been characterized by the use of either straight, rigid legs, as is the case of passive walkers, or by the use of articulated, kinematically-driven legs. In contrast, the design of most hopping or running robots is based on compliant legs which exhibit quite natural behavior during locomotion.

Here we ask to what extent spring-like leg behavior could be useful in unifying locomotion models for walking and running. In so doing, we combine biomechanical experimental and computer simulation approaches with theoretical considerations and simple legged robots.

We have found that (1) walking and running result from mechanical stability which corresponds to the experimentally observed gait dynamics, (2) running is a subset of stable movement patterns for high system energies, and (3) walking with knee flexion during stance can result from passive leg mechanics with elastic structures spanning the joints.

1 Introduction

What are the common design and control principles of legged locomotion?

On the one hand, we must consider the internal leg function: the number of leg segments, the arrangements of muscles, ligaments and other soft tissue within the leg and the appropriate muscle activation patterns in order to generate a desired leg behavior.

On the other hand, we need to integrate the leg into an encompassing control system – the global leg function – such that we can observe a “complete” movement pattern including legs and the supported body. This means that if we have simple, biologically meaningful models describing the leg behavior

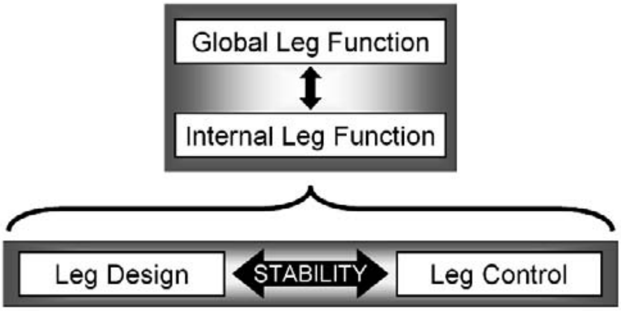


Fig. 1. Leg function is divided into global and internal leg functions. Global leg function describes the leg control based on a given internal leg function. The internal leg function addresses issues of design and control of a segmented leg itself. To identify control strategies we analyze the mechanical stability of selected leg designs at different levels of leg function

in a desired movement task, we can use these leg templates (i.e. simplified models, [7]) to derive the required leg control strategy.

A very simple description of the leg behavior is provided by the spring-mass model. Here, the force generated by the leg during the stance phase is assumed to be proportional to the amount of leg compression, i.e. the more the leg shortens the larger the corresponding leg force. Despite its simplicity this model is a very powerful tool to predict movement strategies or jumping performance [22]. It can help to better understand the role of leg segmentation or muscle function during fast movements. Furthermore, spring-like leg operation seems to ease required control action even in highly dynamic situations with reduced sensory perception.

In legged locomotion we observe a sudden change in leg behavior between walking and running. This gait transition occurs in most animals at about the same dimensionless speed at Froude number equal to approximately 0.4, which is calculated using the equation v^2/gl (where v is the forward speed, g is the gravitational acceleration, and l is the leg length, [14, 26, 8, 18]).

This suggests the possibility that common, underlying mechanical principles may exist for legged systems. If this is true, mechanical design methodologies could be derived which would yield systems equally capable of walking and running gaits, depending on the selected system condition (e.g. system energy) or movement strategy.

In order to change gait, either the mechanical system could change its behavior in an internal and self-organized fashion or the overlaying control strategy could initiate the gait transition. This might point to distinct movement primitives (i.e. programs) which could be used to select gaits. In order to control a given mechanical system we could apply simple feedback control approaches (e.g. a PD controller). In this paper we rather suggest a different approach. First, we build simplified mechanical models either on a computer (i.e. simulation models) or as physical models (i.e. robots) and explore their behavior for a variety of reasonable initial conditions and model parameters. Second, we compare the predictions of different proposed models with experiments of human or animal locomotion. Finally, we investigate the influence of the model parameters on the movement performance and try to estimate the best control strategy in order to improve system stability with the lowest possible sensory effort.

In this paper we describe a series of simple models of internal and global leg functions and compare them with the behavior of two experimental legged robots. We will start with two models addressing internal leg behavior, a three-segment model with elastic joints and a two-segment model with an extensor muscle. Afterwards, we present models describing the global leg function in spring-mass running and walking. Finally, we explore the behavior of the simple legged robots, imitating walking and running.

2 Internal Segmentation of the Leg

In order to operate in a spring-like fashion, a segmented leg must be able to compress and extend stably. With two segments, a leg would have no problem doing so as long as the internal leg joint is operating in a spring-like fashion as well. Leg compression directly translates into joint flexion, which in turn results in higher joint torque and consequently, in increased leg force. The only problem could be in the generation of linear leg spring behavior as observed in human and animal locomotion. Therefore, the joint spring should be nonlinear in terms of the torque-angle characteristics, i.e. being more compliant at low compressions and being stiffer at larger joint flexion.

If we extend the two-segment leg by one segment we obtain a three-segment leg similar to the human leg, with foot, shank and thigh. Now, leg compression can result in different outcomes. Let us assume both internal leg joints (e.g. ankle and knee) to be equally stiff and let us further assume completely symmetric leg geometry (equal segment lengths, equal initial joint angles). We would then expect a transformation of leg compression into equal flexion of the two joints. Interestingly, however, this is not the case (Fig. 2).

In the three segment leg, stability requires a local minimum of the mechanical system energy with respect to variations in the joint angle configuration (ankle and knee) and a given leg length (e.g. distance hip to toe). At a certain amount of leg compression however, this local minimum in system energy changes into a local maximum when loading both joints equally. Consequently, symmetric bending of both joints becomes unstable. Even perfect adjustment of the joint springs cannot prevent asymmetric joint behavior if a certain critical amount of leg compression is exceeded [25]. Here, one leg joint starts to extend whereas the second joint flexes rapidly. As a consequence, the leg force is not shared equally between the joints and the leg stiffness drops. This situation would lead to high local stresses at the flexing joint which could

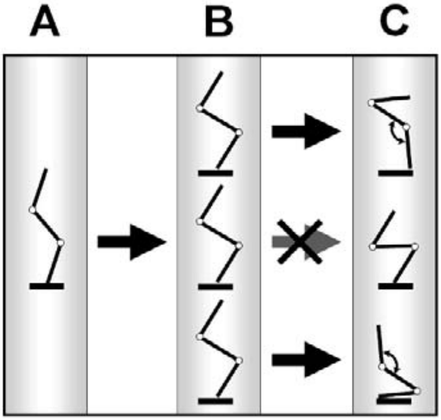


Fig. 2. Three-segment model with equal segment lengths (L1=L2=L3) and two joint springs of equal stiffness and equal static equilibrium angles (**A**). After a certain amount of leg compression (**B**) symmetric bending of both joints becomes unstable. Depending on infinitely small perturbations the two joints proceed by rotating asymmetrically, with one extending and the other flexing (**C**)

result in structural damage or failure in a real leg. To avoid this disastrous behavior, different measures can be introduced. One strategy is to steer the leg joint movements by a kinematic control approach, whereby the joint angle is constrained along a desired trajectory. This intervention may work at low energies through the use of a high-bandwidth controller to counteract the system dynamics. At faster leg movements however, a more systematic modification is required. Stability analysis of the three-segment leg reveals that different solutions exist to avoid this intrinsic instability (Fig. 3).

Strategy A: The joint spring characteristics are slightly non-linear, e.g. the joint stiffness increases with joint flexion. If one joint flexes more than the other, the increased joint stiffness compensates for the mechanical disadvantage caused by the joint's increased flexion. The model predicts a higher nonlinearity in joints whose configuration at static equilibrium is characterized by greater extension (e.g. knee compared to ankle). This prediction is confirmed by experimental results. The required nonlinearity of the joint torque characteristics might be provided by the nonlinear stress-strain characteristics of tendons and aponeuroses connecting the muscle to the skeleton as found in human running or jumping [13, 21].

Strategy B: The leg segmentation is asymmetric, i.e. the outer leg segments are different in length. The model predicts that stable leg operation can be achieved if the joint adjacent to the shorter outer segment (e.g. ankle in humans) is more flexed compared to the other leg joint. This finding is in agreement with human leg design.

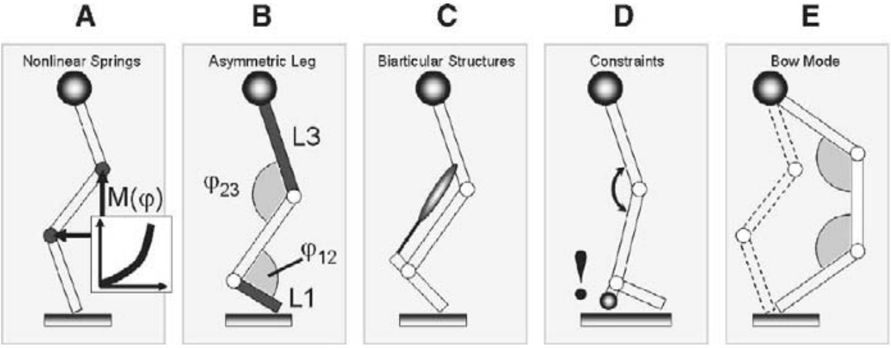


Fig. 3. Five strategies to avoid instability in an elastic three-segment leg: (**A**) nonlinear joint springs, (**B**) asymmetric leg segmentation and leg joint configuration, (**C**) biarticular elastic structures, (**D**) joint constraints, e.g. heel contact, and (**E**) operation in a bow configuration

Strategy C: The risk of out-of-phase joint function due to mechanical instability can be reduced by adding biarticular elastic structures. In the case of asymmetric leg segmentation (Strategy B) it is sufficient to have only one biarticular spring which flexes the more extended joint (knee) and extends the more flexed joint (ankle). The risk of over-extension of the ankle joint is largely avoided by the flexed ankle configuration (about 80–120 degrees) during human locomotion. A biarticular antagonist of the m.gastrocnemius is not required and does not exist in nature.

Strategy D: Even if all previous strategies (A–C) fail to prevent unstable leg operation, there is still another hard-built safety measure to avoid overextension of the more extended leg joint (the knee): the mechanical constraint of joint flexion due to a skeletal structure, e.g. the calcaneus with the heel pad. If the activity of the plantar flexors is not sufficient the heel strike prevents overextension of the knee which would have serious consequences in many athletic movements like running or long jumping.

Strategy E: Finally, there is a strategy to achieve stable leg operation by using a very different nominal leg configuration by swapping the leg joints (from a zigzag or Z-configuration) to the same side with respect to the leg axis (bow or C-configuration). The leg configuration is safe and simple to control at cost of reduced limb stiffness and reduced leg force. It can be found in the upper limbs of humans or in spiders.

All of these strategies guarantee parallel joint operation in a three segment leg and can be found in nature. It is important to realize that these measures are not exclusive and are often implemented in a highly redundant fashion. Similar to the air bags in our cars, the leg includes several design and control strategies to avoid mechanical instability potentially leading to serious damages of the musculo-skeletal system. Elastic joint behavior in itself does not guarantee stable leg operation during contact. However, for the identified leg design and control strategies (Fig. 3) the control of the highly nonlinear segmented leg could be simplified. The results demonstrate that spring-like leg operation can be a key for better understanding the architecture and function of biological legs. On the joint level, spring-like behavior (joint stiffness and nominal angle) can be adapted based on neuromuscular mechanisms [6], [10]. In turn, if all required measures are undertaken to guarantee stable leg operation, spring-like leg operation can result at various loading conditions.

3 Generation of Muscle Activity

Periodic movement patterns as observed in legged locomotion require a cyclic action of the muscles within the body. The time series of the corresponding muscle activation pattern could be the result of different mechanisms. One possibility is direct control of muscle activation by supraspinal commands. In this case the exact timing of all muscles would require a high processing (i.e. high-bandwidth) capacity in the brain. A different approach would be the generation of periodic movement patterns in rhythm generators located in the spinal cord (central pattern generators, [12]). Then the higher control could be reduced to coordinate these pattern generators. Another possibility is the generation of the required muscle activation based on the dynamics of the musculo-skeletal system and sensory feedback to the spinal cord [27]. Such a control strategy could further relax the neural control effort and could take advantage of positive side-effects of the muscle-reflex dynamics. In a simulation study [10], we asked which proprioceptive reflex loop would be capable of generating the required muscle stimulation *STIM(t)* for steady state hopping in place (Fig. 4). Leg geometry is reduced to two massless leg segments with one Hill-type extensor muscle spanning the leg joint. The body is represented by a point mass on top of the upper segment (Fig. 4). Muscle stimulation is assumed to be the sum of a given central command, *STIM0*, and a potential reflex contribution (amplified and time delayed sensory signal based on muscle length, muscle velocity or muscle force).

We found that steady state hopping is possible (1) with an optimized stimulation pattern *STIM0(t)* or (2) based on a constant *STIM0* and positive length or positive force feedback. The predicted maximum hopping height employing positive force feedback is 83% of that calculated using an optimal muscle stimulation pattern and turned out to be robust with respect to simulated external perturbations (e.g. changed ground properties, Fig. 4C). At moderate hopping heights, an almost spring-like leg operation is predicted (Fig. 4B). The simulation results indicate that the generation of the extensor muscle activity in hopping or running tasks could be facilitated by positive force feedback. Instead of giving the muscle a precisely timed activation pattern, the task is now executed based on a constant stimulation (*STIM0)* and proper integration of proprioceptive signals into the activation of the extensor-motoneuron. The control effort is therefore largely reduced and the

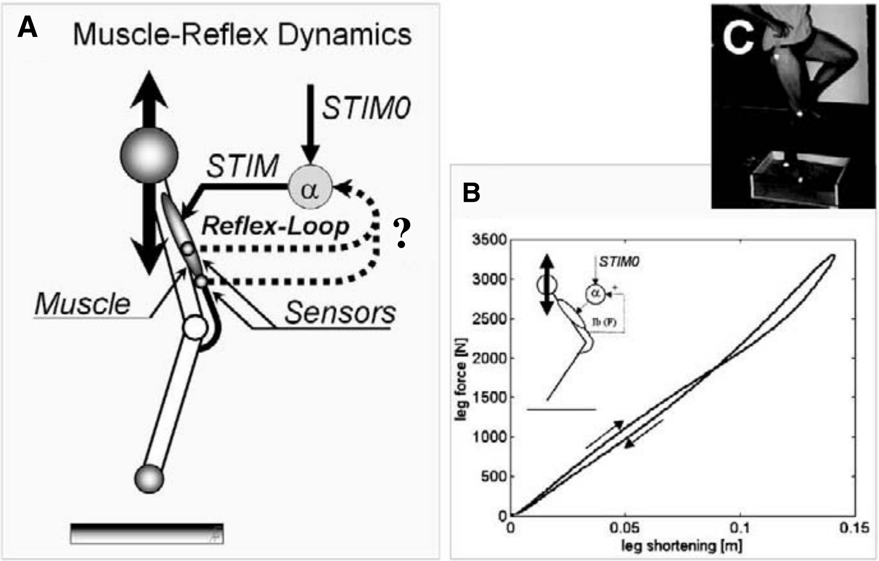


Fig. 4. (**A**) Two-segment leg model with one extensor muscle and proprioceptive feedback. The activity of the α-motoneuron driving the extensor muscle ($STIM$) is assumed to be the sum of a central command ($STIM0$) and a contribution of the reflex loop based on different sensory signals (muscle length, muscle velocity or muscle force). Muscle stimulation signals *STIM(t)* with and without sensory feedback for stable hopping in place are calculated. (**B**) The best hopping performance based on sensory feedback is achieved with positive force feedback and a constant bias signal *STIM0*. Here, spring-like leg behavior is found. (**C**) The muscle-reflex dynamics are robust with respect to environmental changes like hopping on a dissipative substrate (sand)

leg behavior is more robust with respect to perturbations. Furthermore, even with little or no passive compliance (as would be provided by tendons, for instance) the muscle-reflex dynamics produced spring-like leg behavior. In that respect leg stiffness is an emergent steady-state behavior in cyclic hopping (or running) based on the neural integration of sensory information and will therefore adapt to environmental changes detected by the sensory organs.

In the previous sections two simplified models for the internal leg function are introduced. Both models give new insights into how spring-like legs could be designed and controlled. It turned out that compliant leg operation is not just useful to store elastic energy; it also helps to make a segmented leg safe and robust when faced with external perturbations.

In the following two sections we will deal with the global leg function, i.e. how a spring-like leg can be controlled to obtain stable locomotion as observed in running or walking. We provide evidence that spring-like leg operation may also be useful in facilitating the global leg function, i.e. the method in which the body utilizes limbs for stable locomotion.

4 Running with Elastic Legs

The movement of a single leg during walking and running is characterized by a series of contact and swing phases. At lower speeds humans generally choose to walk whereas at higher speeds running is preferred. During the stance phase of running, the leg compresses until midstance and then extends until the leg leaves the ground. The force generated by the leg is approximately proportional to the amount of leg compression [4, 3]. This relation provides the basis for the concept that leg stiffness is the parameter describing that the ratio between leg force and leg compression remains constant. This concept leads to the spring-mass model which describes the movement of the center of mass based on a spring-like leg operation during the stance phase of running [1].

For certain combinations of leg stiffness k and leg angle of attack α_0 cyclic motion of the center of mass (COM) can be observed (Fig. 5A). Interestingly, cyclic movement can also be achieved for slightly different initial conditions.

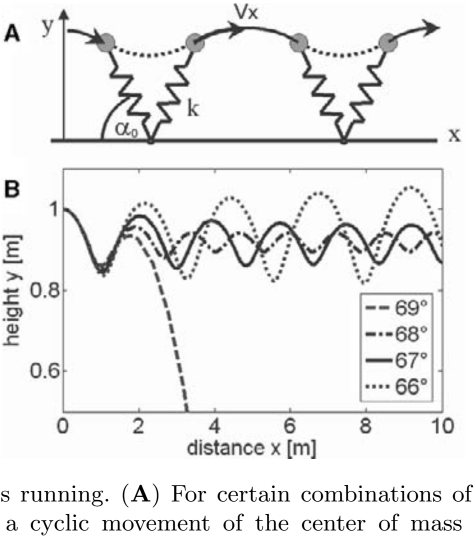


Fig. 5. Spring-mass running. (**A**) For certain combinations of leg stiffness k and angle of attack α_0 a cyclic movement of the center of mass can be found. (**B**) Different angles of attack can result in stable running patterns (here $\alpha_0 = 67^{\rm o}$, 68^o with leg stiffness $k = 20$ kN/m, body mass $m = 80$ kg, leg length $l = 1$ m). Steeper angles overshoot a step resulting in failure at the proceeding step ($\alpha_0 = 69^{\rm o}$) whereas flatter angles decelerate the movements until forward velocity goes to zero ($\alpha_0 = 66^{\rm o}$). With leg retraction, e.g. increasing α_0 prior to landing, the angular adjustment can vary much more without losing running stability (not shown here, for details see [24])

The movement of the center of mass approaches a steady state after a couple of steps without modifying the landing leg angle α_0 (Fig. 5B). However, even when modifying the angle of attack α_0 stable running can be observed, although the COM trajectory has adapted slightly. Hence, spring-mass running is self-stabilizing and robust with respect to changes in the initial conditions and model parameters (e.g. leg angle of attack, leg stiffness, system energy). The adjustment of leg stiffness and angle of attack is not unique. At a given running speed, different combinations are possible (Fig. 5B) and result in specific running styles; for instance, in terms of step length or step frequency. This prediction agrees with experimental results [23]. With increasing speed, the range of successful combinations of leg angle and leg stiffness is even increased.

In contrast, at low speed (less than 3 m/s) no stable running is predicted with the constant angle of attack control policy at any leg stiffness. However, animal or human running reveals that the leg angle is not kept constant prior to landing [5]. In fact, a backward rotation of the leg with respect to the body is observed. Introducing this early leg retraction we find an increased stability in spring-mass running at low speeds [24]. Even larger variations in internal or external conditions (e.g. a change in ground level of 50 percent leg lengths) can be managed when leg retraction is used.

In this section we introduced two global leg control strategies for running with compliant legs: constant angle of attack and leg retraction. We found that for a given system energy spring-mass running is stable for various combinations of leg stiffness k and angle of attack α_0. For higher running speeds and by employing leg retraction the region of stable running (leg stiffness and leg angle adjustment) is largely enhanced.

In a recent study, the spring-mass model for running was extended to a rigid body model in the vertical plane and analyzed based on an analytical approximation neglecting gravity during the stance phase [11]. The results support the identified strategies for stable running with a fixed angle of attack policy. Furthermore, it is argued that at high running speeds the domains of attraction become smaller (i.e. the system is less robust with respect to perturbations of the center of mass trajectory) leading to a demand for more elaborated control methods. One possible method could involve the adaptation of leg stiffness to flight time duration (e.g. due to muscle preactivation) similar to the strategy of leg retraction. Therefore, an integration of muscular mechanisms into the analysis of running stability would be helpful.

5 From Running to Walking

So far only a single leg was considered in the analysis of spring-mass running. As a next step we generalize the model to bipedal locomotion. Does the concept of spring-like leg operation hold only for running with single support phases? What happens to the system dynamics if more than one leg is in contact with the ground at the same time?

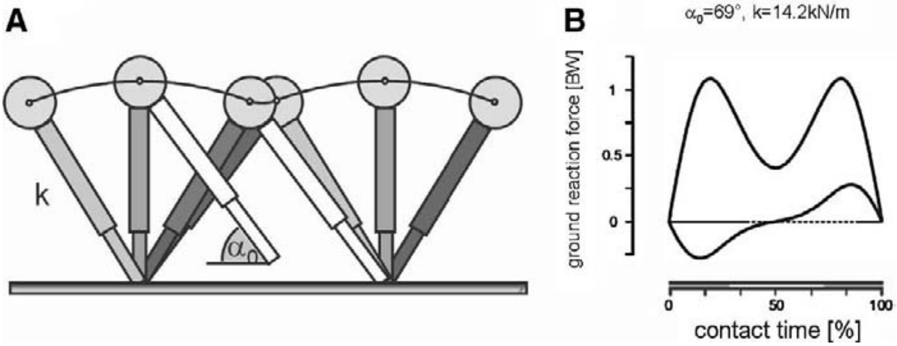


Fig. 6. Spring-Mass Walking. (**A**) For certain combinations of leg stiffness k and angle of attack α_0 a cyclic movement of the center of mass can be found. (**B**) The single leg force patterns (*upper line*: vertical force, *lower line*: horizontal force) resemble that found in human and animal walking [9]

In a previous simulation study we investigated the behavior of a bipedal spring-mass model [9]. This model consists of two massless springs (leg stiffness k) and a point mass representing the center of mass (COM). During the single support phase one leg spring remains in contact with the ground while the other is positioned at a constant angle of attack. If the COM reaches the corresponding landing height of the second leg before the first leg leaves the ground a double support phase occurs (Fig. 6A). Although slightly more complex than the previous running model, stable solutions can again be found using different combinations of leg stiffness k and angle of attack α_0. In contrast to the single leg model, stable solutions with double support phases can only be found for low system energies (forward speed lower than about 1.4 m/s). The single leg forces predicted by the bipedal spring-mass model are very close to the observed patterns in human and animal walking (Fig. 6B). The corresponding maximum walking speed is only slightly above the preferred walking speed observed in humans. This suggests higher control efforts at higher walking speeds. Hence, mechanical stability and therefore a relaxed control (rather than metabolic considerations) could be important criteria to explain the preferred walking speed.

At high energies (forward speed larger than about 3 m/s) the previously observed running pattern is found if the leg is allowed to contact the ground after take-off of the opposite leg. Thus, the bipedal spring-mass model can both stable walking and running on a single mechanical system. There is an energetic gap between both gaits which can only be accessed by a more complicated control strategy (e.g. leg retraction of the swing leg). The model suggests that walking and running are two natural and self-stabilizing behaviors of a simple mechanical system. This concept has strong implications on the design and control of legged systems in general. Leg compliance is not only a strategy to facilitate control; it might also be the origin of the existence of natural gaits. Furthermore, it resolves the issue of collision avoidance which was addressed recently in studies on inverted pendulum walking [20].

In the last sections we examined consequences of compliant leg operation on a global scale. We found that the two fundamental gaits of legged locomotion are natural behaviors of elastic legs attached to a common center of mass. In the next section simple legged robots are introduced. How does the behavior of a constructed leg compare to the conceptional models for legged locomotion presented in the last four sections?

6 Exploring Simple Legged Robots

Elastic leg behavior can elucidate design and control strategies used in legged systems. However, all models presented so far are based on computer simulations or analytical calculations. While the model-based predictions have been compared to results from biology, as discussed in Sects. 3, 4 and 5, it was decided that further validation under real-world conditions was necessary. Therefore, we built a series of very simple legged robots to better understand the pros and cons of our theoretical models, examining their validity and the underlying assumptions, and to examine any overlooked elements.

What should a simple legged robot look like? A good example can be found in the pioneering work of Raibert and his coworkers [19]. These robots are made of elastic legs (pogo sticks) which are controlled in such a manner as to regulate hopping height, body speed and pitch at desired values. We found the construction of a pogo stick leg still "delicate" from the mechanical point of view (e.g. due to constraint forces perpendicular to the leg axis) and decided consequently to start with a simple two-segment leg (Fig. 7A), instead.

6.1 Robot Testbed with Elastic Two-Segment Leg

Our goal is to explore the natural dynamics of a two-segment elastic leg during forward hopping. The movement of the robot (Fig. 7A) is constrained by a metal boom which allows the robot to only move in vertical and horizontal direction. Body pitch movements are not allowed to keep the mechanical system as simple as possible. The focus of this approach is to better understand how legged systems are organized. The movement of the leg is driven by a servo motor between the body and the upper leg segment (thigh). The motor introduces a sinusoidal oscillation defined by oscillator frequency *f*, angular amplitude A and offset angle O. Taking a maximum angular velocity ω_{MAX} of the motor (1 rotation/s) into account, the amplitude A can be calculated depending on the frequency f with

$$A = \omega_{MAX}/(2\pi f) \ . \qquad (1)$$

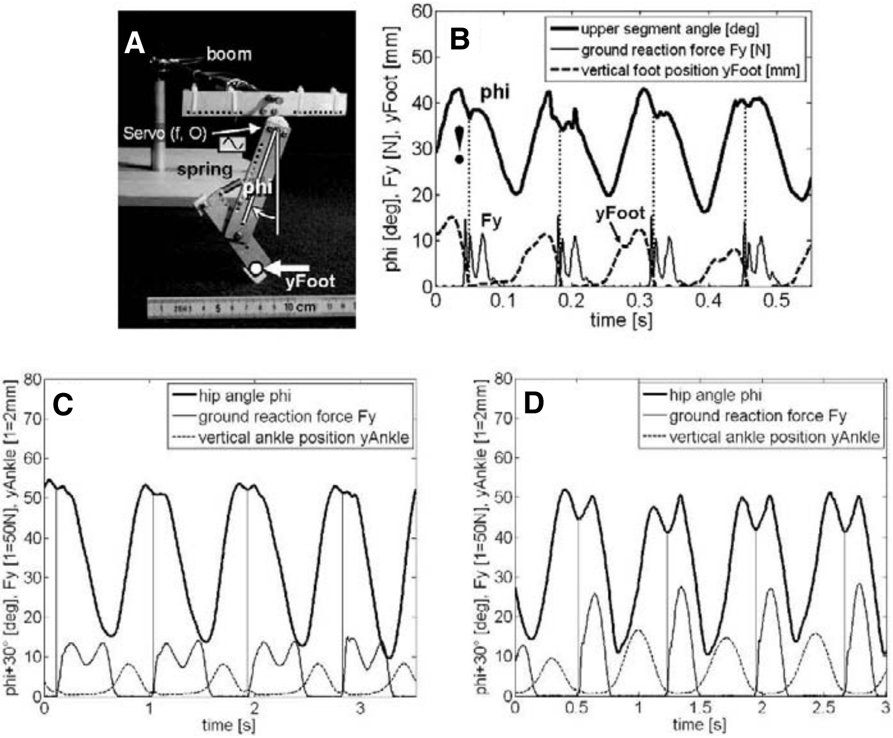


Fig. 7. (**A**) The "Fujubot" robot, with an elastic two-segment leg kinematically driven at a sinusoidal trajectory using a hip-mounted servo motor. (**B**) Experimental data for stable robot hopping. The angle of the upper segment (phi) is predefined by the motor control. At touch-down (*dotted vertical lines*) retraction of the upper segment is briefly interrupted, but recommences immediately afterwards. (**C**) Experimental data on human walking and (**D**) running at 2 m/s

This robot follows the rapidly prototyped, minimalist approach to design ("cheap design" robots, [16]), avoiding high-end or high-precision components and advanced control approaches. For instance, the upper leg segment does not follow the desired angular trajectory of the servo motor (Fig. 7B). Every time the leg hits the ground leg retraction is interrupted. This is due to the fact that the leg joint flexes shortly after landing impact. Comparing this observation with experimental data in human running and walking we find the same phenomenon. At a forward speed of 2 m/s the relative timing of protraction and retraction of the upper limb is very similar between walking and running. Both in the robot and in human running touch-down occurs shortly after the initiation of limb retraction. This is in agreement with the predicted role of leg retraction for stability [24].

To investigate the influence of an enforced leg retraction after touch-down, we implemented a higher torque motor using very stiff coupling of the motor

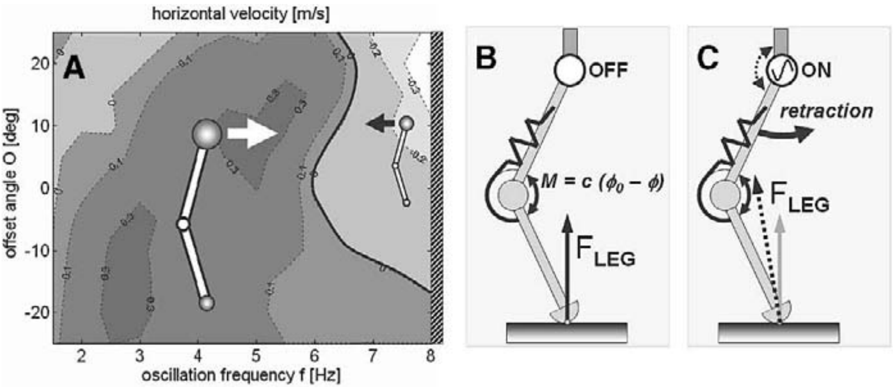


Fig. 8. (**A**) Dependency of hopping direction on control parameters (oscillation frequency *f*, offset angle *O*) of the servo motor. (**B**) With no hip actuation the leg force F_{LEG} merely depends on joint torque *M* and leg geometry. (**C**) Leg force is enhanced compared to (B) if the hip retracts actively (hip extension torque). The opposite is true if leg retraction is pointing to the left, in which case the leg force is reduced

to the hip joint. As a result the boom keeping the upper body upright broke. Consequently, a more compliant coupling was inserted between the motor and the hip joint, imitating the biological function of tendons in hip muscles.

The robot demonstrates a variety of behaviors depending on the selected oscillator frequency *f* and offset angle *O*. Surprisingly, at low oscillation frequencies, the hopping direction is not as expected, namely opposite to the leg joint (Fig. 8A). This movement could well be compared to that of a hopping bird. This behavior is not very sensitive to changes in the control parameters. At higher frequencies (above 6 Hz) a more human-like movement is observed. Then, the leg joint points forward (similar to a human knee). At an intermediate region hopping in place is observed with no substantial horizontal movement.

Why does the robot change its movement direction depending on the selected control frequency? To approach this question the effect of active limb retraction on the leg dynamics is considered in Figs. 8B and 8C. For simplicity, we focus on a static approach neglecting all dynamic effects, i.e. due to segmental accelerations, joint damping, or torques at the foot point. With no retraction (zero hip torque), leg force is directly dependent on limb configuration. This is a consequence of the rotational spring which relates joint torque to joint angle.

If the hip is actively contributing to limb retraction (Fig. 8C), leg force is increased or decreased depending on the geometrical relation between leg joint torque and hip torque. In bird-like hopping, leg force is reduced whereas in human-like hopping the force is increased. As a consequence of this increased (or decreased) leg force, the natural frequency of the hopping system is changed, i.e. an increased leg force to some extent imitates a stronger (stiffer) leg associated with a higher step frequency. This is a well known dependency for the spring-mass model.

Hence, with compliant legs movement direction can be encoded as a frequency signal. At this level, the detailed trajectories of the limb segments are not required to control the different directions of movement. Stable hopping is robust with respect to variations in the control parameters (oscillation frequency, offset angle). This is in line with the self-stabilizing mechanisms of spring-mass running. It remains to be investigated in detail how the control parameters influence the angle of attack and the effective leg stiffness. Furthermore, we found that elastic joint behavior is important in dealing with impacts (e.g. touch-down) avoiding damage to the actuator. A simple harmonic oscillation in the hip is sufficient to obtain stable hopping movement. The observed protraction and retraction plots of the upper limb are very similar to those observed in human walking and running. This encouraged us to build bipedal robots imitating the hip strategy of the one-legged hopping robot. Based on this experimental platform we will extend the concept of spring-like hopping to compliant walking.

6.2 Bipedal Robot

The bipedal spring-mass model indicates that compliant legs may facilitate stable running and walking. Does this mean that walking is just running with double support phases? We approach this question by comparing experimental data on human walking and running with the behavior of a simple bipedal robot. We will demonstrate that leg compliance is useful in generating stable walking movements. The analysis of the three-segment model (Sect. 1) indicates that elastic joint operation may lead to a synchronous operation of the leg joints. Such behavior can be found in human running (Fig. 9) where during stance both knee and ankle joint flex and extend in parallel. In walking this situation is not found: the knee joint extends during midstance and the ankle joint extends only at the late portion of the stance phase when the knee joint has returned to flexing. The desired function of a mechanical spring to store and release elastic energy does not seem to be fulfilled by a walking leg. The high level of coordination between knee and ankle joint in running allows an efficient push-off phase after midstance. The biarticular m.gastrocnemius transfers the rotational energy of the knee joint to the ankle which in turn is capable of generating a rapid leg extension. In walking this coupling is not found. The extension of the knee joint does not lead to a push-off phase because the ankle joint does not follow the knee extension. In the extended configuration the knee is not able to contribute to leg lengthening but it has a significant contribution to leg rotation. In fact, the effect of thigh retraction on leg retraction is reduced by knee extension and (later) supported by knee flexion. As a result the thigh is already protracting before the leg leaves the ground (Fig. 9A).

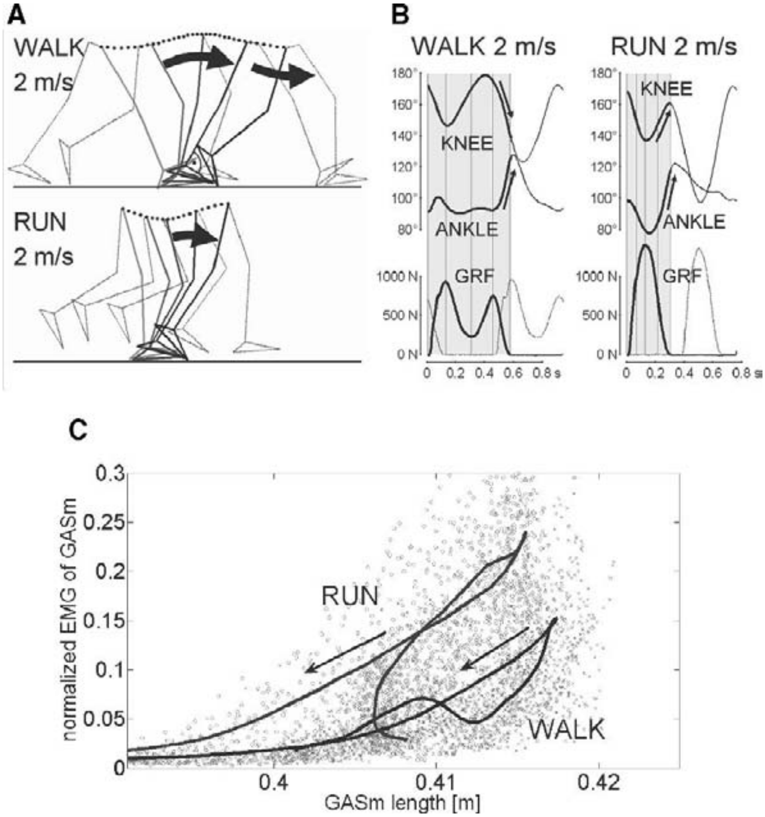


Fig. 9. (**A**) Leg kinematics in human walking and running at the preferred transition speed (2 m/s). (**B**) Knee and ankle joint kinematics and vertical ground reaction forces corresponding to (A). (**C**) Muscle activity vs. muscle length of m.gastrocnemius medialis (GASm) of one subject during walking (41 steps, circles) and running (45 steps, crosses) at 2 m/s. *Thick lines* represent mean tracings for walking and running. Muscle length is calculated based on knee and ankle angle data [15]

If the rotational energy during knee extension (at midstance in walking) is not used for push-off, how can this energy be reused for locomotion? With continuous thigh retraction during stance phase (e.g. due to active hip retraction) the m.gastrocnemius gets stretched and can use the knee rotation at a later time to contribute to ankle extension. Hence, the rotational energy of the knee during midstance could still be reused for push-off triggered by the amount of leg rotation during stance phase. Therefore, we hypothesize that in walking the nominal length of the biarticular m.gastrocnemius should be longer than when compared to running.

To test this hypothesis we analyze the activity of m.gastrocnemius medialis (Fig. 9C) and compare it to the estimated length of the muscle [15]. We find

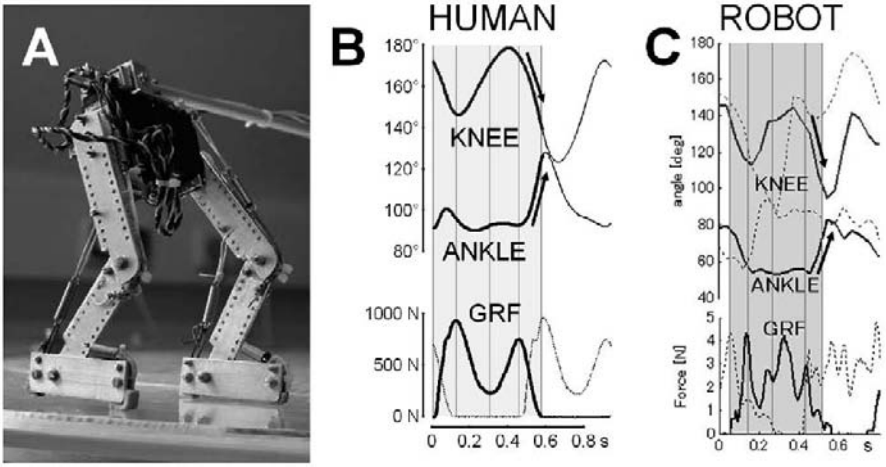


Fig. 10. (**A**) The "JenaWalker" bipedal robot, equipped with compliant legs. (**B**) Joint kinematics of knee and ankle joint and ground reaction forces (GRF) during human walking and (**C**) during robot walking. Dotted curves indicate opposite leg

that muscle activity is dependent on muscle length and gait. In walking the EMG is active at greater muscle length as compared to running. This indicates that walking could take advantage of elastic biarticular structures spanning knee and ankle joint as also suggested in a previous simulation study [2].

To test this idea the bipedal "JenaWalker" robot was developed (Fig. 10A, [17]). In order to investigate this gait-specific interplay between ankle and knee joint we decided to use three-segment legs. A biarticular spring was installed between knee and ankle joints, simulating the function of m.gastrocnemius. Furthermore, two additional "muscles" were required in the leg: one foot flexor (m.tibialis anterior) and one biarticular knee extensor and hip flexor (m.rectus femoris). This combination of elastic structures in a three-segment leg turned out to be sufficient to generate stable locomotion. In agreement with our experimental data on human walking we kept the hip control the same as in the hopping robot (i.e. a simple harmonic oscillation).

An example of the leg kinematics and the ground reaction forces for stable walking is given in Fig. 10C. The robot is able to reproduce the experimentally observed knee and ankle joint kinematics (Fig. 10B). It is important to note that no effort was made to optimize the leg kinematics or the ground reaction forces to fit biological data. The criterion used for optimization is a steady periodic movement pattern. The robot is able to walk, hop and run specified by a corresponding motor adjustment. For instance, walking is robust with respect to changing step frequencies. Running is only possible for high offset angles with the legs operating in front of the body. This is due to the fact that the leg cannot bend freely during the swing phase. With synchronous hip function, hopping was found.

The legged robots presented here demonstrate that stable walking and running can be observed with compliant legs and simple harmonic oscillations at the hip joint. The movements are robust although both design and control are very simple. In the three-segment leg, the function of knee and ankle joint is gait-specific. The biarticular muscles could play an important role in synchronizing the internal leg function depending on the selected gait. *Running* can be considered as a gait with fast leg compression and fast leg rotation. The leg rotation (retraction) is mainly provided by hip extension and knee flexion (plus late ankle extension). Due to synchronized operation of knee and ankle, the high joint angular velocities are slowed down in the biarticular m.gastrocnemius. *Walking* is a gait with slow leg compressions and slow leg rotation. Leg rotation is reduced by knee extension during midstance but supported by knee flexion at early and late stance phase. As a consequence, the upper limb can already start protraction while the leg is still on the ground. This allows the hip control to still follow a harmonic oscillation despite a duty factor larger 0.5. Due to the out-of-phase operation of knee and ankle, the slow joint movements are accelerated in the biarticular m.gastrocnemius. Further research is required to better understand these intersegmental dynamics in walking and running at different speeds and environmental conditions.

7 Conclusions

In this paper we summarized several simple biomechanical models and "cheap" design robots describing legged locomotion. The common design principle was to reduce the systems to a minimal configuration which allows for a systematic investigation of the underlying mechanisms of legged locomotion. The introduction of compliant structures and the search for self-stabilizing mechanisms revealed to be effective tools to identify natural movement patterns and relaxed control strategies.

It turned out that walking and running can be described as two natural movement patterns of one mechanical system with elastic legs. The organization of the segmented leg is largely supported by elastic structures spanning one or more joints. For properly designed legs, the control is largely simplified and could be reduced to an adjustment of hip oscillators in our legged robots. In the future, we aim to identify leg designs which are equally suited for both human-like walking and running. Therefore, a better understanding of the gait selecting mechanisms will be required.

References

[1] Blickhan, R. (1989) The spring-mass model for running and hopping. J. Biomech. 22, 1217–1227.

[2] van den Bogert, A. J. (2003) Exotendons for assistance of human locomotion. Biomedical Engineering Online, 2–17.

[3] Cavagna, G. A., Heglund, N. C. and Taylor, C. R. (1977) Mechanical work in terrestrial locomotion: two basic mechanisms for minimizing energy expenditure. Am. J. Physiol. 233, 243–261.

[4] Cavagna, G. A., Saibene, F. P. and Margaria, R. (1964) Mechanical work in running. J. Appl. Physiol. 19, 249–256.

[5] De Wit, B., De Clercq, D., and Aerts, P. (2000). Biomechanical analysis of the stance phase during barefoot and shod running. J. Biomech. 33, 269–278.

[6] Feldmann, A. (1966) Functional tuning of the nervous system during control of movement or maintenance of a steady state posture. II controllable parameters of the muscles. Biophysics 11: 565–578.

[7] Full, R. F. and Koditschek D. (1999) Templates and anchors: neuromechanical hypotheses of legged locomotion on land. J. Exp. Biol. Vol. 202, Issue 23, 3325–3332.

[8] Gatesy, S. and Biewener, A. (1991) Bipedal locomotion effects of speed, size and limb posture in birds and humans. J. of Zool. 224, 127–147.

[9] Geyer, H. (2005) Simple models of legged locomotion based on compliant limb behavior. PhD Thesis. University of Jena.

[10] Geyer, H., Seyfarth, A. and Blickhan, R. (2003) Positive force feedback in bouncing gaits? Proc. R. Soc. Lond. B 270, 2173–2183.

[11] Ghigliazza, R., Altendorfer, R., Holmes, P., and Koditschek, D. (2003) A simply stabilized running model. SIAM Journal on Applied Dynamical Systems, 2(2), pp. 187–218.

[12] Grillner, S. (1986) in Wenner-Gren International Symposium Series Vol. 45. Neurobiology of Vertebrate Locomotion (eds. Grillner, S., Stein, P. S. G., Stuart, D. G., Forssberg, F. and Herman, R. M.) 505–512 (Macmillan, London).

[13] Guenther, M. and Blickhan, R. (2002) Joint stiffness of the ankle and the knee in running. J. Biomech. 35, 1459–1474.

[14] Hayes, G. and Alexander, R. M. (1983) The hopping gaits of crows (corvidae) and other bipeds. J. Zool. Lond. 200, 205–213.

[15] Hof, A. L., van Zandwijk, J. P. and Bobbert, M. F. (2002) Mechanics of human triceps surae muscle in walking, running and jumping. Acta Physiol Scand 174, 17–30.

[16] Iida, F. (2005) Cheap design approach to adaptive behavior: walking and sensing through body dynamics. AMAM conference, Ilmenau.

[17] Iida, F., Minekawa, Y., Rummel, J., and Seyfarth, A. (2006) Toward human-like biped robot with compliant legs. Intelligent Autonomous Systems 9. Arai, T. et al. (Eds.). IOS Press, 820–827.

[18] Kram, R., Domingo, A., and Ferris, P. F. (1997) Effect of reduced gravity on the preferred walk-run transition speed. J. Exp. Biol. 200, 821–826.

[19] Raibert, M (1986) Legged robot that balance. MIT press.

[20] Ruina, A., Bertram, J. E. A., and Srinivasana, M. (2005) A collisional model of the energetic cost of support work qualitatively explains leg sequencing in walking and galloping, pseudo-elastic leg behavior in running and the walk-to-run transition. J. Theor. Biology, 237, 170–192.

[21] Seyfarth, A., Blickhan, R., and van Leeuwen, J. L. (2000) Optimum take-off techniques and muscle design for long jump. J. Exp. Biol. 203, 741–750.

[22] Seyfarth, A., Friedrichs, A., Wank, V., and Blickhan, R. (1999) Dynamics of the long jump. J. Biomech. 32, 1259–1267.

[23] Seyfarth, A., Geyer, H., Guenther, M., and Blickhan, R. (2002). A movement criterion for running. J. Biomech. 35, 649–655.

[24] Seyfarth, A., Geyer H., and Herr, H. (2003) Swing-leg retraction: a simple control model for stable running. J. Exp. Biol. 206, 2547–2555.

[25] Seyfarth, A., Guenther, M., and Blickhan, R. (2001). Stable operation of an elastic three-segmented leg. Biological Cybernetis 84, 365–382.

[26] Thorstensson, A. and Roberthson, H. (1987) Adaptations to changing speed in human locomotion: speed of transition between walking and running. Acta Physiol. Scand. 131, 211–214.

[27] Zehr, E. P. and Stein, R. B. (1999) What functions do reflexes serve during human locomotion? Prog. Neurobiol. 5, 185–205.

Self-stability in Biological Systems – Studies based on Biomechanical Models

H. Wagner[1] and P. Giesl[2]

[1] Biomechanics and Motor Control, WWU Münster, Horstmarer Landweg 62b, 48149 Münster, Germany
heiko.wagner@uni-muenster.de

[2] Zentrum Mathematik, TU München, Boltzmannstr. 3, 85747 Garching bei München, Germany
giesl@ma.tum.de

Summary. Mechanical properties of complex biological systems are non-linear, e.g. the force-velocity-length relation of muscles, activation dynamics, and the geometric arrangement of antagonistic pair of muscles. The control of such systems is a highly demanding task. Therefore, the question arises whether these mechanical properties of a muscle-skeletal system itself are able to support or guarantee for the stability of a desired movement, indicating self-stability. Self-stability of single joint biological systems were studied based on eigenvalues of the equation of motions and the basins of attraction were analysed using Lyapunov functions. In general, we found self-stability in single muscle contractions (e.g. frog, rat, cui), in human arm and leg movements, the human spine and even in the co-ordination of complex movements such as tennis or basketball. It seems that self-stability may be a general design criterion not only for the mechanical properties of biological systems but also for motor control.

1 Introduction

The basis for human and animal motion and locomotion are co-ordinated muscle contractions. Even for very simple movements, a huge number of muscles must be controlled. Therefore, we may ask how humans and animals are able to control such complex neuromusculoskeletal systems. Especially for humans, the easiest way to get an answer is to ask somebody. But we would not expect a meaningful answer because the motor control system acts almost without conscious control. The muscles must generate sufficient forces and moments at the joints. However, these forces and moments must be fine tuned in such a way that they can react upon sudden perturbations. This fine tuning may be guaranteed by mono- and poly-synaptic reflexes with negative and positive feedback-loops [5]. On the other hand, the mechanical properties of a musculoskeletal system itself may support or even guarantee for sufficient stability [7, 4]; in these cases motion or locomotion is self-stabilized [10, 2, 1]. Simple biomechanical models, e.g. single muscle contractions or one degree of freedom joint models, can be self-stabilized. But even if these subsystems are self-stabilized the global motion of the multi-body system may still be unstable. However, the global control of a locally self-stabilized system is easier compared to a locally unstable system. The purpose of this paper is to analyze and summarize the self-stabilizing properties of biological systems, i.e. single muscle contractions, single joint movements, and more complex arrangements like the human spine. This paper is a companion paper to Giesl and Wagner (this issue) where the mathematical details to analyse stability and basins of attraction of biomechanical models are given.

2 Single Muscle Quick Release Contractions

The first step in analyzing self-stability of musculoskeletal systems was to study whether mechanical properties of muscles themselves may provide self-stability. A simple method to investigate the self-stabilizing properties of dissected muscles is a quick-release experiment. In quick-release experiments dissected muscles are loaded with an external weight or force, which will be released suddenly [8]. Typically, after the release the muscles contracted and found a new equilibrium at shorter muscle lengths, indicating that the systems were stable (Fig. 1).

As a next step, the quick-release experiments were described and simulated by an equation of motion of a biomechanical model. Based on the equation of motion the stability could be analysed by the eigenvalues of the system. For the given experiments the eigenvalues were negative indicating self-stability. Furthermore, because of the simplicity of the model, the system could be

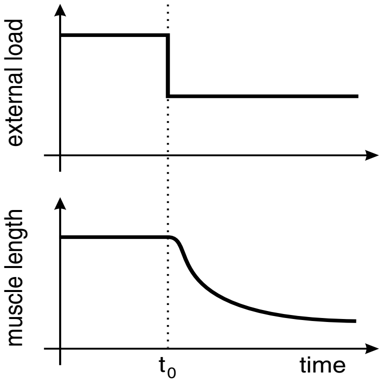


Fig. 1. Schematical representation of a quick-release experiment. At time t_0 the external load was released (*upper row*) and the muscle contracted until a new equilibrium was found between the external force and the muscle force

analysed analytically. From this it could be shown that the classical muscle properties, i.e. the force-velocity relation, force-length relation, are sufficient to provide self-stability. As the mechanical properties of dissected muscles support self-stability, the question arise whether muscles within a geometrical arrangement of joints are still able to achieve self-stability.

3 One Degree of Freedom Joint Models

In models consisting of a muscle and a simple one-degree of freedom rotational joint the interactions between the muscle and joint properties influence the stability of the system. In general, the inner muscle moment arm as well as the moment arm according to the external force vector depended on the flexion angle of joints. The individual shapes of these dependencies were influencing the mechanical stability of the system. We performed quick-release experiments with the elbow-joints of rats and cuis, while the extensor muscles were stimulated [8]. Here again, the systems found new equilibriums after the release of the external loads, indicating stability. A stability analysis based on a biomechanical model resulted in negative eigenvalues, indicating asymptotic stability likewise. For flexor muscles of an elbow joint the moment arms can be calculated from simple trigonometric assumptions, whereas the moment-arms of extensor muscles depends on individual geometrical arrangements. Based on an analytical analysis of the eigenvalues (Giesl and Wagner, this issue) it can be estimated that the derivative $\frac{\partial h_{ext}}{\partial \beta}$ of the geometric function h_{ext} with respect to the flexion angle β must be positive to support stability. This is guaranteed around elbow flexion angles below ca. 90° (Fig. 2).

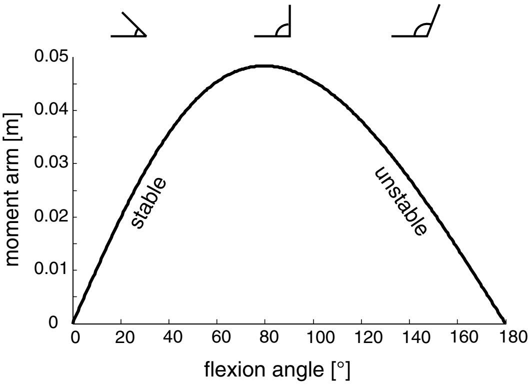


Fig. 2. Representation of the flexor moment arm at an elbow joint (length humerus = 0.27 m, length ulna = 0.26 m, insertion of the muscle at the ulna = 0.048 m). The slope of the curve indicates that stability is supported for flexion angles below ca. 90°

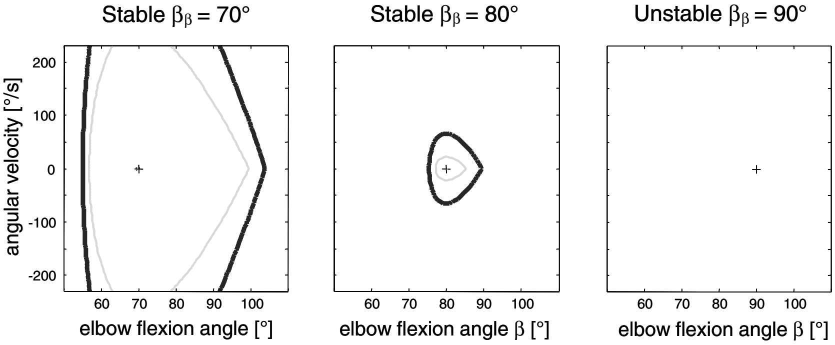


Fig. 3. The basins of attraction of equilibrium points in the phase-plane were calculated with Lyapunov-functions of the systems (Giesl and Wagner this issue). The basins of attraction (black lines – extensor activation 50% of maximum voluntary contraction) were reduced with increasing elbow angles (*left*: equilibrium at $\beta_0 = 70°$; *middle*: $\beta_0 = 80°$; *right*: $\beta_0 = 90°$). Furthermore, the basins of attraction were reduced with decreasing level of co-activation (*thin gray lines* – extensor activation 25% of maximum voluntary contraction)

Experiments with humans supported this result [6]. Here, we determined individual muscle properties of the flexor and extensor muscles. Then the subjects performed quick-release contractions [12]. As a result, the subjects found new equilibriums at lower elbow flexion angles, which was in accordance with the animal experiments. The stability analysis of the experiments resulted in negative eigenvalues for flexion angles below ca. 90°, indicating stability. Furthermore, basins of attraction were calculated based on the theory of Lyapunov functions. We found considerable large basins of attraction at low elbow flexion angles and unstable situations for more extended elbows (Fig. 3). Furthermore, the areas of the basins of attraction depended on the co-activation level of the antagonistic muscles.

Finally, we analysed simple vertical oscillations of a human leg model [10, 11]. Here additionally, the self-stability was supported by a moving center of rotation at the knee joint, as well as a co-activation of bi-articular muscles, i.e. rectus and biceps femoris muscles.

While introducing a joint it is much more complicated to achieve self-stability. Therefore, several solutions to support the stability could be found in biological systems, e.g. co-activation of bi-articular muscles, moving center of rotations.

4 Varying Center of Rotation Model

In the previous section we have discussed the stabilizing behaviour of simple biomechanical models of extremities, i.e. elbow joint and knee joint. These models have a well defined location of the instantaneous center of rotation. In the following we will discuss models with simple one-degree of freedom rotational joints and varying center of rotations. As an example, we may think of lateral flexions of the human lumbar spine. Here we can not define one single center of rotation but depending on the intermuscular co-ordination of local inter-vertebral muscles the center of rotation will vary between the lowest (L5-S1) and the highest functional unit (L1-L2). As a simplification we analyzed rotations at one functional unit, e.g. L1-L2, while the other units were assumed to be stiff. How interactions between different joints of models with more than one degree of freedom may influence the stability of the system cannot be answered with the present simplified model. Is it still possible to self-stabilize such a model with a single pair of antagonistic muscles? First, we described the antagonistic muscles with a Hill-type model including a force-velocity relation, but excluding a force-length relation [9]. As a result two stable areas existed; one around L5-S1 with negative attachment angles of the muscles, e.g. obliquus internus muscle, and another one more cranially for positive attaching angles of the muscles, e.g. obliquus externus or multifidus muscles. But it was impossible to stabilize every center of rotation with only one antagonistic muscular arrangement. Therefore, we improved the model and included a force-length relation such that the muscle was acting on the ascending limb. Now, it was possible to self-stabilize the system at every location of the center of rotation (Fig. 4). We calculated the minimum physiological cross-sectional area (PCSA) of the acting muscles that still can stabilize the system. The physiological cross-sectional area is nearly proportional to the maximum isometric force of a muscle, therefore, a low minimum value of PCSA indicates that only low muscular force is necessary to stabilize the system. For oblique muscle arrangements a minimum physiological cross-sectional area (PCSA) between 50 and 80 cm^2 was found, whereas, muscles acting in parallel to the spine were able to stabilize the system with only 7 cm^2. Introducing additional antagonistic muscles could not reduce this minimum value of the PCSA.

5 Discussion

The purpose of this paper was to analyze and summarize the self-stabilizing properties of biological systems. We tried to draw a line from single muscle contractions, single joint movements, to more complex arrangements like the human spine. Single muscle contractions could be stabilized based on typical shapes of bio-mechanical properties, i.e. the force-velocity relation and the force-length relation of skeletal muscles. It could be shown analytically, that the typical shape of the force-velocity relation was essential for the stabilisation of single muscle contractions [8].

Furthermore, if acting on the ascending limb of the force-length relation, the typical shape of the active and passive force-length relation supports the self-stability of muscles. Especially for muscle lengths above the optimum

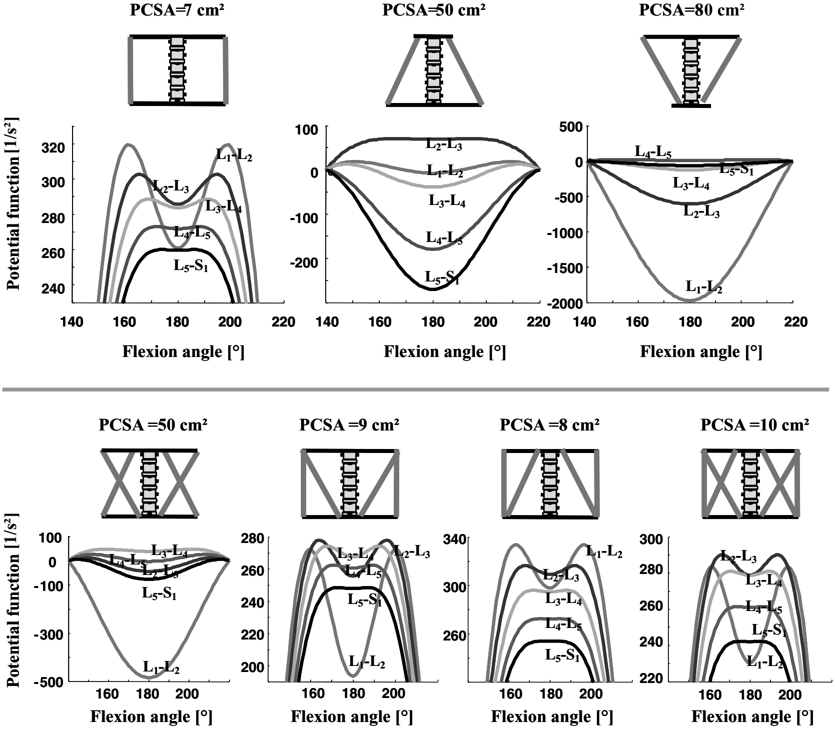


Fig. 4. Lyapunov-functions for different antagonistic muscle arrangements describing the lumbar spine. The centers of rotations were located at L1-L2, L2-L3, L3-L4, L4-L5, and L5-S1 [3]. For all situations a minimum PCSA for self-stability was calculated. The x-axis shows the flexion angle of the segment, while 180° represented the vertical position. The small icons represent the muscular arrangements of the different models and the minimum PCSA for each model is given at the top

length the passive properties were important. In sub-maximal contractions, the activation level of the muscle changes the slope of the force-velocity and force-length relation and thus changes the stability of the system. Therefore, sub-maximal co-ordination patterns in physiological motions and locomotion influences the self-stability of the system [11].

If the muscle was not dissected the geometrical arrangement of joints influenced the self-stability behaviour. The flexion angle of an elbow joint effected the stability of the system. Extending the elbow more than about 90° results in an unstable situation [6]. This simple geometrical dependency may influence simple movement tasks, e.g. imagine a waiter who should not spill the water in the glass while moving. But also throwing tasks are influenced by these geometric relations. Compare throwing a basketball with juggling. Whereas in the first case the basketball will be released with a nearly extended elbow joint, in the second case the juggling-ball leaves the hand with a more flexed elbow joint.

Especially the stabilisation of the human spine is a challenging task. Here the location of the center of rotation may vary depending on the activation pattern of intervertebral muscles. They are influencing the stiffness around the spine and therefore the location of the instantaneous center of rotation. However, the simulations support the assumption that even for this complicated situation the muscles can guarantee for the self-stabilizing function of the spine. Without changing the activation patterns of the trunk muscles, it seems to be possible to stabilize lateral flexions at different centers of rotations [9]. This analysis of the self-stabilizing behaviour of biological systems may influence different scientific areas, e.g. robotics and prosthetics, and it may hopefully have an effect for the medicine and physiotherapy. A profound understanding of the self-stabilizing properties of biological systems is important while investigating the motor control of complex movements of the whole body. Although the models analyzed here were very simple, we may assume that self-stability seems to be an important criterion for the evolution of humans and animals. If the basin of attraction of an equilibrium point or an envisioned trajectory is considerably large this may offer a great advantage for motor control systems. A self-stable system can be controlled much easier with simple reflexes compared to an unstable system. If the system is risking to move out off the basin of attraction a simple reflex may be sufficient to push it back into the stable basin. Especially for fast movements, which do not require a high precision, the neuronal system can be unburdened. Considering the control of legged robots this may reduce the requirements on the precision of the sensors and the controller systems.

6 Acknowledgement

We would like to thank A. Liebetrau. She calculated the numerical analyses of the human spine model with varying center of rotation.

References

[1] Blickhan R, Seyfarth A, Geyer H, Grimmer S, Wagner H, Günther M (accepted) Intelligence by mechanics. Proc. Roy. Soc. Lond. A.

[2] Blickhan R, Wagner H, Seyfarth A (2003). Brain or muscles? (S. G. Pandalai, Ed.) Recent Research Developments in Biomechanics. Transworldresearch, Trivandrum, India. 1:215–245.

[3] Bogduk, N (2000) Klinische Anatomie von Lendenwirbelsäule und Sakrum. Berlin, Springer.

[4] Dickinson M H, Farley C T, Full R J, Koehl M A, Kram R, Lehman S (2000) How animals move: an integrative view. Science 288(5463):100–106.

[5] Geyer H, Seyfarth A, Blickhan R (2003) Positive force feedback in bouncing gaits? Proc R Soc Lond B Biol Sci 270(1529):2173–2183.

[6] Giesl P, Meisel D, Scheurle J, Wagner H (2004) Stability analysis of the elbow with a load. J Theor Biol 228(1):115–125.

[7] Loeb G E, Brown I E, Cheng E J (1999) A hierarchical foundation for models of sensorimotor control. Exp Brain Res 126(1):1–18.

[8] Siebert T, Wagner H, Blickhan R (2003) Not all oscillations are rubbish: Forward simulation of quick-release experiments. JMMB 3(1):107–122.

[9] Wagner H, Anders C, Puta C, Petrovitch A, Mörl F, Schilling N, Witte H, Blickhan R (2005) Musculoskeletal properties support stability of the lumbar spine. Pathophysiology 12: 257–65.

[10] Wagner H, Blickhan R (1999) Stabilizing function of skeletal muscles: an analytical investigation. J Theor Biol 199(2):163–179.

[11] Wagner, H, Blickhan R (2003) Stabilizing Function of Antagonistic Neuromusculoskeletal Systems – an Analytical Investigation. Biol Cybern 89:71–79.

[12] Wagner H, Giesl P, Blickhan R (subm.) Musculoskeletal stabilization of arm movements – complex or real –. Journal of Mechanics in Medicine and Biology.

Holonomy and Nonholonomy in the Dynamics of Articulated Motion

P.-B. Wieber

INRIA Rhône-Alpes, 38334 St-Ismier Cedex, France
pierre-brice.wieber@inrialpes.fr

Summary. Walking, running or jumping are special cases of articulated motions which rely heavily on contact forces for their accomplishment. This central role of the contact forces is widely recognized now, but it is rarely connected to the structure of the dynamics of articulated motion. Indeed, this dynamics is generally considered as a complex nonlinear black-box without any specific structure, or its structure is only partly uncovered. We propose here to precise this structure and show in details how it shapes the movements that an articulated system might realize. Some propositions are made then to improve the design of control laws for walking, running, jumping or free-floating motions.

1 Introduction

Improving the technology of sensors, actuators, computing power, mechanical design, might still be necessary in order to achieve faster and more reliable motions than what can be observed today in humanoid robotics, but better control law designs will probably be one of the key points. The design of feedback laws heavily relies on the understanding that we have of the underlying dynamics, and there may still be room for improvement here.

Walking, running or jumping are special cases of articulated motions that strongly depend on contact forces for their accomplishment. This central role of the contact forces is widely recognized now, but it is rarely connected to the structure of the dynamics of articulated motion. Indeed, this dynamics is generally considered as a complex nonlinear black-box without any specific structure, or its structure is only partly uncovered. We propose here to precise this structure and show in details how it shapes the movements that an articulated system might realize.

The existence of a structure in the dynamics of articulated motion has often been recognized [4, 7, 21, 25, 29], but it has never been analyzed as thoroughly as in [18]. Even in a study as precise as what can be found in [21, 20], it is not completely clear that behind the “d’Alembertian wrench” of the system studied there lies in fact its Newton and Euler equations, as will be stated here. Most of all, the holonomy and nonholonomy of the kinetic momenta and their implications for the locomotion of articulated systems has been rarely if not never discussed outside the works of the authors of [18], with the only exception of space robotics [16, 19].

The analysis that we are going to propose here is therefore deeply inspired by what can be found in [18]. Now, this brilliant work has been made in the framework of Lie algebras, a very powerful framework for high-level analysis of dynamical systems, but which may hide somehow the details appearing in the “real” equations to the reader who doesn’t speak this language fluently. The main point of the present article is therefore to rederive these results without the use of Lie algebras. Doing so calls for an unusual way to derive the dynamics of articulated systems, through the use of Gauss’s principle.

We’re going therefore to present this principle and how it can be used to derive the dynamics of articulated systems in Sect. 2. This original way of deriving this dynamics will be helpful then in Sect. 3 in order to precise its inner structure. What this structure implies for the movements that articulated systems can realize will be studied then in Sect. 4, where nonholonomy makes its first appearance. Nonholonomy will be the main topic then of Sect. 5, where some implications of this phenomenon for the locomotion of articulated systems are put to light, and where propositions are made to make use of it in order to improve the design of control laws for walking, running, jumping or free-floating motions.

2 Gauss’s Principle and the Dynamics of Articulated Motion

2.1 Gauss’s Principle

Gauss’s principle, equivalent to d’Alembert’s one, can be seen as an extension of the principle of virtual work to the dynamical case. It states that the acceleration of a set of solids subject to some constraints deviates the least possible from the acceleration that it would have had without the constraints [23, 27]. This deviation is measured with the following kinetic metric:

$$\mathcal{D} = \sum_k \frac{1}{2}\left(\ddot{x}_k - \underline{\ddot{x}}_k\right)^T m_k \left(\ddot{x}_k - \underline{\ddot{x}}_k\right) + \frac{1}{2}\left(\dot{\omega}_k - \underline{\dot{\omega}}_k\right)^T \mathbb{I}_k \left(\dot{\omega}_k - \underline{\dot{\omega}}_k\right) , \tag{1}$$

with $\ddot{x}_k$ and $\dot{\omega}_k$ the translation and rotation accelerations of the k^{th} solid, m_k its mass, $\mathbb{I}_k$ its inertia matrix expressed at its center of mass, and $\underline{\ddot{x}}_k$ and $\underline{\dot{\omega}}_k$ the translation and rotation accelerations that it would have had without the constraints, that is the solutions of the classical Newton and Euler equations,

$$\begin{aligned} m_k\, \underline{\ddot{x}}_k &= f_k \ , \\ \mathbb{I}_k\, \underline{\dot{\omega}}_k - (\mathbb{I}_k\, \omega_k) \times \omega_k &= \tau_k \ , \end{aligned} \tag{2}$$

where f_k and τ_k are the forces and torques acting on this solid. Note that the Euler equation is expressed in a frame attached to the solid, as well as the velocity ω_k, reason why there is a gyroscopic term $(\mathbb{I}_k\,\omega_k) \times \omega_k$.

2.2 The Dynamics of Articulated Motion

Considering now a set of solids constrained to move together by a set of articulations, their dynamics can be computed with the help of Gauss's principle. The constraints induced by the articulations can be expressed implicitly by describing the positions of the different solids of the system in a compact way through a configuration vector $q \in \mathbb{R}^n$. Their velocities and accelerations can be related then to the vectors $\dot{q}$ and $\ddot{q}$ with the help of translation and rotation jacobians:

$$\begin{aligned} \dot{x}_k &= J_{tk}(q)\,\dot{q}\ , \\ \omega_k &= J_{Rk}(q)\,\dot{q} \end{aligned} \tag{3}$$

and

$$\begin{aligned} \ddot{x}_k &= J_{tk}(q)\,\ddot{q} + \dot{J}_{tk}(q,\dot{q})\,\dot{q}\ , \\ \dot{\omega}_k &= J_{Rk}(q)\,\ddot{q} + \dot{J}_{Rk}(q,\dot{q})\,\dot{q}\ . \end{aligned} \tag{4}$$

Introducing these relations in the definition (1) of the deviation $\mathcal{D}$ and solving the Newton and Euler equations (2) for $\underline{\ddot{x}}_k$ and $\underline{\dot{\omega}}_k$, the optimality condition for the minimization of this deviation turns into (we skip these calculations which are completely straightforward)

$$\frac{\partial \mathcal{D}}{\partial \ddot{q}} = M(q)\,\ddot{q} + N(q,\dot{q})\,\dot{q} - \mathcal{F} = 0\ , \tag{5}$$

with

$$M(q) = \sum_k J_{tk}^T\, m_k\, J_{tk} + J_{Rk}^T\, \mathbb{I}_k\, J_{Rk}\ , \tag{6}$$

$$N(q,\dot{q}) = \sum_k J_{tk}^T\, m_k\, \dot{J}_{tk} + J_{Rk}^T\, \mathbb{I}_k\, \dot{J}_{Rk} - J_{Rk}^T\, (\mathbb{I}_k\, J_{Rk}\, \dot{q}) \times J_{Rk}\ , \tag{7}$$

$$\mathcal{F} = \sum_k J_{tk}^T\, f_k + J_{Rk}^T\, \tau_k\ . \tag{8}$$

We end up therefore with a classical Lagrangian description of the dynamics of a system of articulated bodies with an inertia matrix $M(q)$, nonlinear dynamical effects $N(q,\dot{q})\,\dot{q}$ and generalized forces $\mathcal{F}$ acting on the system. Note that we're taking some liberties in (7) and in the following with the notation of the cross-product by considering that given a vector $v \in \mathbb{R}^3$, the notation $(v)\times$ means in fact multiplying by the classical anti-symmetric matrix

$$\begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}.$$

The point here is that even though the definition (6) of the inertia matrix is identical to what can be found in usual robotics textbooks [15, 26], it is not the case for the definition (7) of the nonlinear effects. Indeed, these nonlinear effects are generally presented through Christoffel symbols that completely hide the structure that can be seen here, and this structure is going to be very useful in analyzing the dynamics of articulated motion. For example, one can observe immediately with the definitions here that the matrix $\dot{M} - 2N$ is anti-symmetric.

3 Inner Structure of the Dynamics of Articulated Motion

3.1 The Structure of the Configuration Vector

In the case of locomoting or free-floating articulated systems, the configuration vector $q \in \mathbb{R}^n$ introduced in the previous section stitches in fact together three very distinct informations,

$$q = \begin{bmatrix} \hat{q} \\ x_0 \\ \theta_0 \end{bmatrix} , \tag{9}$$

where $\hat{q}$ describes the positions of the articulations of the system, and x_0 and θ_0 the position and the orientation of a reference frame attached to one solid of the system.

This structure of the configuration vector can be found then in all the kinematic and dynamic equations of the system, to begin with the translation and rotation jacobians that were introduced in (3). Indeed, if we consider the translation and rotation velocities $\dot{\hat{x}}_k$ and $\hat{\omega}_k$ of the k^{th} solid with respect to the reference frame that we have just introduced, they must be composed with the velocities $\dot{x}_0$ and ω_0 of this reference frame itself in order to obtain the total translation and rotation velocities $\dot{x}_k$ and ω_k of the solid. This is done through the following classical composition rules (remember that the rotation velocities ω are expressed in local frames):

$$\begin{aligned} \dot{x}_k &= R_0(\theta_0)\, \dot{\hat{x}}_k + \dot{x}_0 + (R_0(\theta_0)\, \omega_0) \times (x_k - x_0) , \\ R_k(q)\, \omega_k &= R_k(q)\, \hat{\omega}_k + R_0(\theta_0)\, \omega_0 \end{aligned}$$

with $R_0(\theta_0)$ and $R_k(q)$ the orientation matrices of the k^{th} solid and the reference frame with respect to the inertial frame. Now, if we use the fact that the velocities $\dot{\hat{x}}_k$ and $\hat{\omega}_k$ are solely related to the vector $\dot{\hat{q}}$,

$$\begin{aligned} \dot{\hat{x}}_k &= \hat{J}_{tk}(\hat{q})\, \dot{\hat{q}} , \\ \hat{\omega}_k &= \hat{J}_{Rk}(\hat{q})\, \dot{\hat{q}} , \end{aligned}$$

and the fact that the rotation velocity ω_0 of the reference frame can be related to the angular velocity $\dot{\theta}_0$,

$$\omega_0 = J_{R0}(\theta_0)\,\dot{\theta}_0 \ ,$$

these composition rules turn into (with shortened notations)

$$\begin{aligned} \dot{x}_k &= R_0\,\hat{J}_{tk}\,\dot{\hat{q}} + \dot{x}_0 - (x_k - x_0) \times R_0\,J_{R0}\,\dot{\theta}_0 \ , \\ \omega_k &= \hat{J}_{Rk}\,\dot{\hat{q}} + R_k^T R_0\,J_{R0}\,\dot{\theta}_0 \ , \end{aligned}$$

where we can observe that the translation and rotation jacobians that were introduced in (3) exhibit a structure corresponding exactly to the structure (9) of the configuration vector:

$$\begin{aligned} J_{tk} &= \begin{bmatrix} R_0\,\hat{J}_{tk} & \mathbb{1}_{3\times3} & -(x_k - x_0) \times R_0\,J_{R0} \end{bmatrix} \\ J_{Rk} &= \begin{bmatrix} \hat{J}_{Rk} & \mathbb{0}_{3\times3} & R_k^T\,R_0\,J_{R0} \end{bmatrix} \end{aligned} \qquad (10)$$

with $\mathbb{0}_{3\times3}$ and $\mathbb{1}_{3\times3}$ a zero and an identity matrix.

3.2 Back to Newton and Euler Equations

Replacing this structure of the jacobians J_{tk}^T and J_{Rk}^T in (6)–(8), we obtain a structure of the inertia and non-linear effects matrices and of the generalized forces that corresponds once again exactly to the structure (9) of the configuration vector:

$$M(q) = \sum_k \begin{bmatrix} \hat{J}_{tk}^T\,R_0^T\,m_k\,J_{tk} + \hat{J}_{Rk}^T\,\mathbb{I}_k\,J_{Rk} \\ m_k\,J_{tk} \\ J_{R0}^T\,R_0^T\big((x_k - x_0) \times m_k\,J_{tk} + R_k\,\mathbb{I}_k\,J_{Rk}\big) \end{bmatrix} , \qquad (11)$$

$$N(q,\dot{q}) = \\ \sum_k \begin{bmatrix} \hat{J}_{tk}^T\,R_0^T\,m_k\,\dot{J}_{tk} + \hat{J}_{Rk}^T\,\mathbb{I}_k\,\dot{J}_{Rk} - \hat{J}_{Rk}^T\,(\mathbb{I}_k J_{Rk}\,\dot{q}) \times J_{Rk} \\ m_k\,\dot{J}_{tk} \\ J_{R0}^T\,R_0^T\big((x_k - x_0) \times m_k\,\dot{J}_{tk} + R_k\,\mathbb{I}_k\,\dot{J}_{Rk} - R_k\,(\mathbb{I}_k J_{Rk}\,\dot{q}) \times J_{Rk}\big) \end{bmatrix} , \qquad (12)$$

$$\mathcal{F} = \sum_k \begin{bmatrix} \hat{J}_{tk}^T\,R_0^T\,f_k + \hat{J}_{Rk}^T\,\tau_k \\ f_k \\ J_{R0}^T\,R_0^T\big((x_k - x_0) \times f_k + R_k\,\tau_k\big) \end{bmatrix} . \qquad (13)$$

The dynamics (5) can be split therefore in three lines, each one with a very specific structure. Particularly interesting are the two last ones: with the help of relations (3) and (4), the line in the middle gives

$$\sum_k m_k \ddot{x}_k = \sum_k f_k$$

and the last line gives

$$J_{R0}^T R_0^T \sum_k (x_k - x_0) \times m_k \ddot{x}_k + R_k \mathbb{I}_k \dot{\omega}_k - R_k (\mathbb{I}_k \omega_k) \times \omega_k = \\ J_{R0}^T R_0^T \sum_k (x_k - x_0) \times f_k + R_k \tau_k$$

what, putting aside the multiplication by $J_{R0}^T R_0^T$, corresponds to an equality between the dynamical momentum of rotation of the whole system and the sum of all the torques applied to it, both expressed with respect to x_0 in an absolute reference frame. What appears here are therefore simply a Newton and an Euler equation for the whole system.

3.3 Forces Acting on the System

We're going to consider three different types of generalized forces, those which are most generally found acting on systems of articulated bodies: the gravity $\mathcal{F}_g$, the control forces $\mathcal{F}_u$ and the contact forces $\mathcal{F}_c$. The gravity forces and torques acting on each solid are simply $f_k = m_k g$ and $\tau_k = 0$, where g is simply the vector of the gravity field. Replacing this in (13) gives

$$\mathcal{F}_g = \sum_k \begin{bmatrix} \hat{J}_{tk}^T R_0^T m_k g \\ m_k g \\ J_{R0}^T R_0^T (x_k - x_0) \times m_k g \end{bmatrix}. \tag{14}$$

Now, we can observe from the structure (10) of the translation and rotation jacobians that we obviously have

$$J_{tk} \begin{bmatrix} 0 \\ g \\ 0 \end{bmatrix} = g \quad \text{and} \quad J_{Rk} \begin{bmatrix} 0 \\ g \\ 0 \end{bmatrix} = 0$$

so that a short inspection of (11) leads to the fact that

$$\mathcal{F}_g = M(q) \begin{bmatrix} 0 \\ g \\ 0 \end{bmatrix}.$$

This implies that the dynamics (5) of the system under the action of gravity can be written as:

$$M(q)\left(\ddot{q} - \begin{bmatrix} 0 \\ g \\ 0 \end{bmatrix}\right) + N(q,\dot{q})\,\dot{q} = 0\ .$$

Here lies the obvious observation that even on an articulated system, the action of gravity is nothing but a linear acceleration in the direction of the gravity field.

Concerning control forces, we'll consider that the only ones acting on the system are internal forces acting between the different solids of the system, coming from the action of muscles or actuators on the articulations of the system. In this case, the application of Newton's law of action and reaction leads us immediately to the conclusion that their sum (13) is of the form

$$\mathcal{F}_u \;=\; \begin{bmatrix} u \\ 0 \\ 0 \end{bmatrix}\ , \tag{15}$$

and we're not going to precise any deeper the structure of the vector u for the analysis undertaken here.

Concerning the contact forces between the system and its environment, very different models exist [3] and we will focus only on the fact that whatever the model, there are always similar constraints on their direction and amplitude due to unilaterality and limited friction. We will gather all these limitations in a vector inequality relating these forces to the position of the system:

$$\mathcal{A}(q,\mathcal{F}_c) \;\leq\; 0 \tag{16}$$

4 Motions that an Articulated System can Realize

4.1 With only Control Forces

If we introduce the center of mass x_G of the system,

$$x_G = \frac{1}{m}\sum_k m_k x_k \quad \text{with} \quad m = \sum_k m_k\ ,$$

the Newton equation of the whole system appears to be simply

$$m\,\ddot{x}_G \;=\; \sum_k f_k\ . \tag{17}$$

If we get rid of the multiplication by $J_{R0}^T\,R_0^T$ in the Euler equation and if we add to it

$$(x_0 - x_G)\times\sum_k m_k\ddot{x}_k \;=\; (x_0 - x_G)\times\sum_k f_k$$

in order to express the momentum of rotation and the sum of torques with respect to the center of mass instead of x_0, we obtain

$$\sum_k (x_k - x_G) \times m_k \ddot{x}_k + R_k \, \mathbb{I}_k \, \dot{\omega}_k - R_k \, (\mathbb{I}_k \, \omega_k) \times \omega_k \; = \; \sum_k \; (x_k - x_G) \times f_k + R_k \, \tau_k \; , \tag{18}$$

what is equal to

$$\frac{d}{dt}\left[\sum_k (x_k - x_G) \times m_k \dot{x}_k + R_k \, \mathbb{I}_k \, \omega_k\right] \; = \; \sum_k \; (x_k - x_G) \times f_k + R_k \, \tau_k \; ,$$

where the left hand side appears to be simply the derivative of the kinetic momentum of rotation of the system. This way, we can see that if the system is under the action of only the control forces (15), we have the obvious conservation of the kinetic momenta:

$$\begin{aligned} m \, \dot{x}_G &= \text{Constant} \; , \\ \sum_k (x_k - x_G) \times m_k \dot{x}_k + R_k \, \mathbb{I}_k \, \omega_k &= \text{Constant} \; . \end{aligned}$$

If the system starts with a zero velocity, these constants are zero and the first equation implies that whatever the control forces employed, the center of mass of the system will remain unmoved. The implications of the second equation are more subtle since it is a nonholonomic constraint, a relation between the velocities of the bodies of the system that doesn't imply a relation between their positions and orientations: it constrains the movements that the system can realize, but not the positions that it can reach.

We're going to spend more time in Sect. 5 on the many implications of this nonholonomy, but we can already stress that notwithstanding this conservation of the kinetic momentum of rotation, the position of the articulations $\hat{q}$ and the orientation of the system θ_0 can be controlled together to any desired value, with the only action of muscles or actuators on the articulations of the system: controlling the articulations of the system is enough to control also its orientation.

4.2 With Gravity Forces

If the system is under the action of the gravity forces (14) in addition to the control forces considered earlier, we can observe that the only modification to the movements of the system is that its center of mass will be linearly accelerated along the gravity vector instead of staying idle:

$$\ddot{x}_G \; = \; g \; .$$

The conservation of the kinetic momentum of rotation is unchanged, and so is the conclusion about its nonholonomy, and so is therefore the fact that

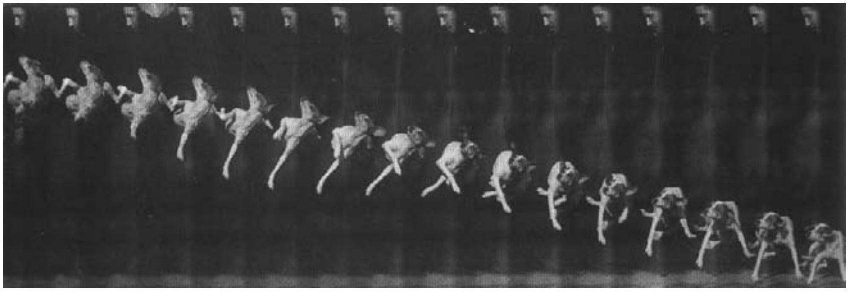


Fig. 1. Falling back on the feet thanks to nonholonomy [6]

controlling the articulations of the system is enough to control also its orientation. This can be observed in the most usual example of articulated system under the action of gravity and muscles, a cat falling which always manages to fall back on its feet. Figure 1 shows the similar case of a dog, and a close inspection of this stop-motion allows to understand how the rotation of the body is undertaken even though the kinetic momentum of rotation is kept unchanged to zero: limbs are moved back and forth with different lengths, and therefore with different inertial properties. We're going to focus again later on this very simple principle.

4.3 With Contact Forces

Of the three types of forces considered in Sect. 3.3, the contact forces appear therefore to be the only one able to generate movements of the center of mass of the system in any way other than falling down. The locomotion of articulated systems completely rely therefore on the availability of contact forces. Now, we have seen that these contact forces are limited because of unilaterality and limited friction, as has been resumed in the general inequality (16). Every movement undertaken by an articulated system has therefore to comply with these limitations.

Research in biped locomotion has been extensively focusing on this question, and in different ways, either focusing explicitly on the dynamic momenta that appear in the Newton and Euler equations (17)–(18) as in the Resolved Momentum Control approach [10, 11], in the Zero Moment Point analysis [8, 28] and in other similar works [9, 13, 22], or treating more globally the force allocation problem directly in the Lagrangian Equation (5) as in [7, 14, 29]. Since we have seen that this Lagrangian equation embeds explicitly the Newton and Euler equations of the system, we can conclude that these two ways of approaching the problem are exactly equivalent.

The contact forces are also the only ones which can have an effect on the momentum of rotation of the system and therefore potentially remove, within the bounds of the inequalities (16), all the dynamical constraints that existed on the movements of articulated systems. This point is less crucial

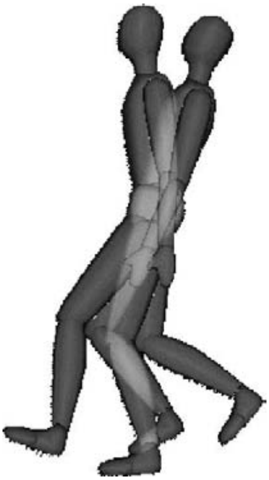


Fig. 2. 22 degrees of rotation of the whole body in the sagittal plane induced by 24 steps of the walking pattern recorded in [31] and replayed on a simple free-floating biomechanical model, what corresponds to a complete turn in 393 steps

though than the problem of moving the center of mass since we have seen that the nonholonomy of this momentum of rotation allows for a control of the orientation of the system through the control of its articulations. This is going to be the main topic of the next section.

5 Some Implications on the Locomotion of Articulated Systems

5.1 Nonholonomy of the Momentum of Rotation When Walking

We have seen in Sect. 4.2 that the nonholonomy of the momentum of rotation can be observed in the case of the very specific movements that cats and dogs realize in order to fall back on their feet. It is in fact a very general phenomenon that can be observed even when walking. When replaying for example the walking pattern recorded in [31] on a simple biomechanical model with no external forces and therefore with a kinetic momentum of rotation constantly equal to zero, we can indeed observe a rotation of the whole body happening in the sagittal plane (Fig. 2). Note that a general property of such nonholonomic constraints is that the outcome of the movement doesn't depend on its speed, but solely on its shape, so we can measure the nonholonomy here as an amount of rotation of the whole body for each step of walking, depending solely on the shape of this step, regardless of its actual speed. In this case, that gives $1/393^{rd}$ of a complete turn of the body per step accomplished.

A more accurate measurement of this phenomenon can be obtained on a robotic system such as the HRP-2 robot, for which the inertial properties and

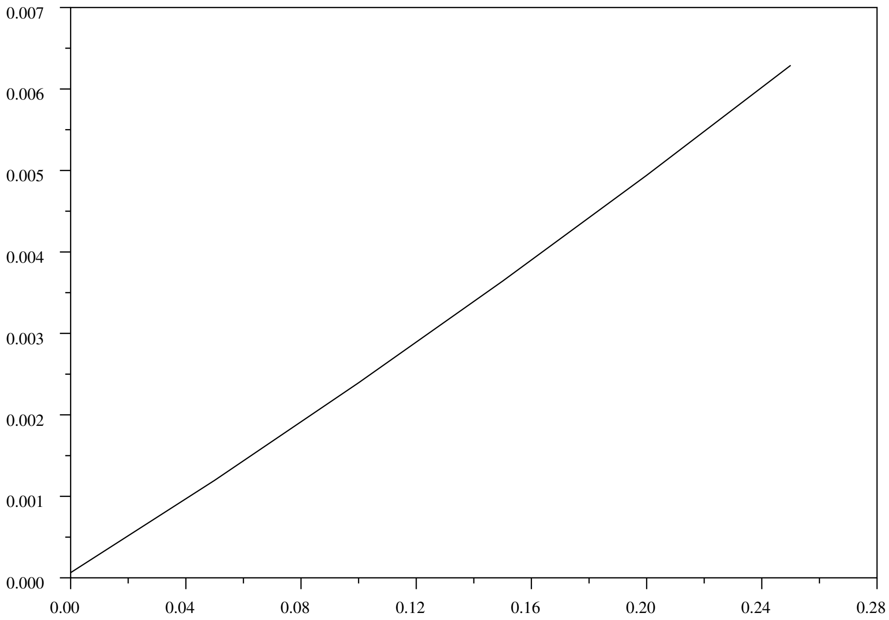


Fig. 3. Fraction of a complete turn in the sagittal plane of the whole body of the HRP-2 robot induced by a step of walking as a function of the height of this step (in meters), with up to $1/159^{th}$ of a complete turn for a 25 cm high step

the movements actually realized can be known with very good precision. We can accurately measure then the amount of rotation of the whole body as a function of the shape of the step, for example its height in Fig. 3, reaching here $1/159^{th}$ of a complete turn of the body for a 25 cm high step.

Note that this phenomenon is intrinsic to the movements of the legs when walking, back and forth with different lengths in order to avoid undesired contacts with the ground, and therefore back and forth with different inertial properties, just as what has been observed in the case of the dog falling in Fig. 1. Keeping the body upright when walking necessarily calls therefore for a second phenomenon in order to counterbalance this rotation. One can imagine the arms making the exact inverse of the movements of the legs, with different lengths when moving back and forth, but this is not what can be observed in natural walking patterns, without mentionning the cases when the arms don't even move back and forth, when holding a heavy object or when keeping arms crossed. The counterbalancing phenomenon that can be observed is in fact a non-zero mean kinetic momentum of rotation in the direction opposite to the rotation induced by the legs' movements (Fig. 4). With these observations in mind, the proposition made in [22] of controlling the kinetic momentum of rotation of a walking system to keep it to zero appears to be problematic: a zero kinetic momentum of rotation appears to be incompatible with walking,

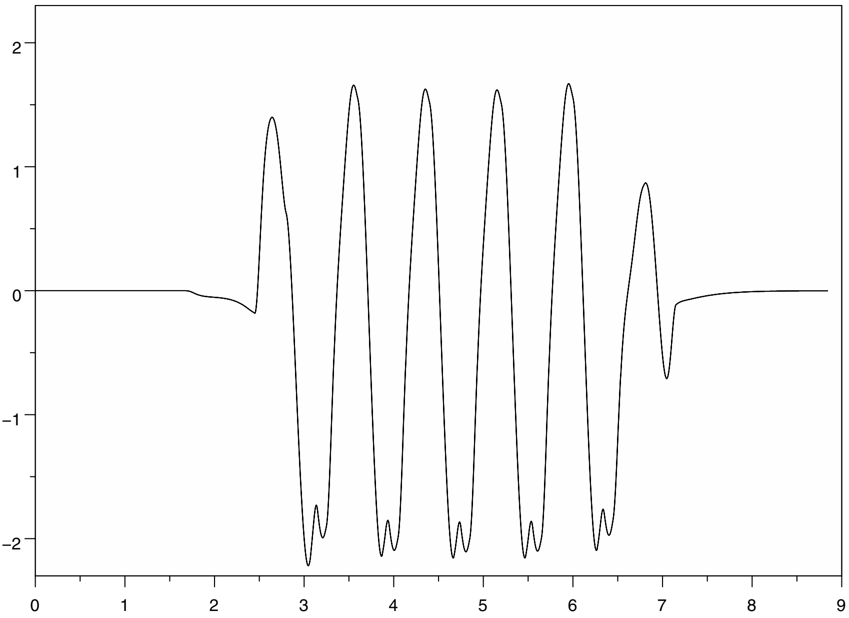


Fig. 4. Kinematic momentum of rotation of the HRP-2 robot in the sagittal plane (in kg.m^2.s^{-1}) as a function of time (in seconds) when executing 25 cm high steps on the ground

and designing a control law to do so will obviously lead to an error either in the tracking of the walking movements or in the tracking of the zero kinetic momentum, both potential sources of instability.

It could be tempting then to control this kinetic momentum of rotation to a non-zero value, following for example the framework presented in [11], but the choice of the desired value may not be easy to decide since it should be made according to the specific shape of the walking pattern being considered. On top of that, the kinetic momentum of rotation can be observed to vary strongly when walking (Fig. 4), so the choice of a constant value might be an unnecessary limitation. But most of all, as we have already observed in Fig. 2, the value of the kinetic momentum of rotation is only scarcely related to the actual rotations that the articulated system is going to realize: deciding a value for this momentum decides in fact almost nothing about the movement to come. It may not be sure therefore whether focusing specifically on the control of the kinetic momentum of rotation as proposed in [1, 9, 11, 13, 22] is the best option, after all. At least, it is incomplete in controlling the motion of an articulated system.

5.2 Some Hints for Improving the Control of Walking, Running, Jumping and Free Floating Motions of Articulated Systems

The amounts of rotation measured in the previous section, $1/393^{rd}$ to $1/159^{th}$ of a complete turn, might look small if not negligible, the main reason that led to the erroneous conclusion in [22] that they might be strictly zero. But we should not forget the example of the cat always falling back on its feet: nonholonomy can be a very precious tool in the control of the locomotion of articulated systems, not to be underestimated.

Of course, solely varying the height of the steps as in the previous section is not a serious solution to completely stabilize a walking movement, but this is an indication that varying the shape of the walking pattern can help improve this stability, especially if motions of the whole body are involved, for greater efficiency (the arms can be a precious source of inertial effects). At least, this can be a valuable addition to the methods already known for stabilizing walking motions such as varying the step lengths or the speed [30].

Useful in the case of walking, making use of this nonholonomy can become an absolute necessity in the case of running, jumping and free floating motions, being the only way to control the orientation of the system when contact forces are not available anymore. Now, it is well known as a side effect of a famous theorem due to R.W. Brockett [2] that the complete control of a system with nonholonomic constraints can't be realized with continuous time-invariant feedback control laws: discontinuous or time-varying control laws are a necessity in this case [5]. This explains why the use of this nonholonomy is out of reach of the control laws proposed in [12, 17, 25] which are all continuous and time-invariant.

Using this nonholonomy is a well established control method in space robotics [16, 19, 24], but the only control law making such a use of this nonholonomy that seems to have been proposed so far for humanoid locomotion is for running, in [4]. There, a time-varying control law is proposed in the flight phase by simply letting an additional degree of freedom in the design of the trajectories be used to control the orientation of the system. We can observe there that calling for discontinuous or time-varying control laws doesn't necessarily imply complex solutions: local modifications of the shape of the limb trajectories can be just enough, what can be made even easier from a computational point of view with the help of a library of precomputed motions.

6 Conclusion

The core of this short note has been a precise description of the inner structure of the dynamics of articulated systems of bodies, establishing in particular how the Newton and the Euler equations of the whole system are very simply and directly embedded inside its Lagrangian dynamics, implying an immediate equivalence between the approaches that focus on the former equations and the approaches that focus on the latter one. This whole analysis has been made possible in the first place thanks to an original derivation of this Lagrangian dynamics through the use of Gauss's principle.

Conclusions have been derived then concerning the holonomy of the Newton equation and the nonholonomy of the Euler equation, implying the necessity of contact forces for articulated systems to realize translations, but not rotations for which joint forces are enough. A specific analysis of this latter phenomenon has been undertaken then in the case of walking motions, and propositions have been finally made to make use of it for improving the design of control laws for stabilizing walking, running, jumping, and more generally every kind of articulated movements on the ground and in the air.

References

[1] M. Abdallah and A. Goswami. A biomechanically motivated two-phase strategy for biped upright balance control. In *Proceedings of the IEEE International Conference on Robotics & Automation*, 2005.

[2] R.W. Brockett. Asymptotic stability and feedback stabilization. In *Differential Geometric Control Theory*. Birkhauser, 1983.

[3] B. Brogliato. *Nonsmooth Impact Mechanics*. Springer-Verlag, 1996.

[4] C. Chevallereau, E.R. Westervelt, and J.W. Grizzle. Asymptotically stable running for a five-link, four-actuator, planar bipedal robot. *International Journal of Robotics Research*, 2005.

[5] A. De Luca and G. Oriolo. Modelling and control of nonholonomic mechanical systems. *Kinematics and Dynamics of Multi-Body Systems, CISM Courses and Lectures*, 1995.

[6] M. Frizot. *Etienne-Jules Marey : Chronophotographe*. Nathan, 2001.

[7] Y. Fujimoto, S. Obata, and A. Kawamura. Robust biped walking with active interaction control between foot and ground. In *Proc. of the IEEE International Conference on Robotics & Automation*, 1998.

[8] A. Goswami. Postural stability of biped robots and the Foot-Rotation Indicator (FRI) point. *International Journal of Robotics Research*, 1999.

[9] A. Goswami and V. Kallem. Rate of change of angular momentum and balance maintenance of biped robots. In *Proceedings of the IEEE International Conference on Robotics & Automation*, 2004.

[10] K. Harada, S. Kajita, K. Kaneko, and H. Hirukawa. An analytical method on real-time gait planning for a humanoid robot. In *International Conference on Humanoid Robotics*, 2004.

[11] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa. Resolved momentum control: Humanoid motion planning based on the linear and angular momentum. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots & Systems*, 2003.

[12] S. Kajita, T. Nagasaki, K. Kaneko, K. Yokoi, and K. Tanie. A running controller of humanoid biped hrp-2lr. In *Proceedings of the IEEE International Conference on Robotics & Automation*, 2005.

[13] T. Komura, H. Leung, S. Kudoh, and J. Kuffner. A feedback controller for biped humanoids that can counteract large perturbations during gait. In *Proceedings of the IEEE International Conference on Robotics & Automation*, 2005.

[14] K. Löffler, M. Gienger, and F. Pfeiffer. Sensor and control design of a dynamically stable biped robot. In *Proceedings of the IEEE International Conference on Robotics & Automation*, 2003.

[15] R.M. Murray, Z. Li, and S. Sastry. *A Mathematical Introduction to Robotic Manipulation.* CRC Press, 1994.

[16] Y. Nakamura and R. Mukherjee. Exploiting nonholonomic redundancy of space robots through hierarchical Lyapunov functions. *IEEE Transactions on Robotics and Automation*, 1993.

[17] V. Nunez and N. Nadjar-Gauthier. Control strategy for vertical jump of humanoid robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots & Systems*, 2005.

[18] J. Ostrowski and J. Burdick. Geometric perspectives on the mechanics and control of robotic locomotion. In *Proc. of the International Symposium on Robotics Research*, 1995.

[19] E. Papadopoulos. Nonholonomic behaviour in free-floating space manipulators and its utilization. *Nonholonomic Motion Planning, Xu, Y. and Kanade, T. (Eds.)*, 1993.

[20] J. Park. Principle of dynamical balance for multibody systems. *Multi-Body Systems and Dynamics*, 2005.

[21] J. Park, Y. Youm, and W.-K. Chung. Control of ground interaction at the zero-moment point for dynamic control of humanoid robots. In *Proceedings of the IEEE International Conference on Robotics & Automation*, 2005.

[22] M. Popovic, A. Hofmann, and H. Herr. Zero spin angular momentum control: definition and applicability. In *International Conference on Humanoid Robotics*, 2004.

[23] E.J. Routh. *Dynamics of a system of rigid bodies, part 1.* Dover Publications, 1905.

[24] V. Schulz, R. Longman, and H. Bock. Computer-aided motion planning for satellite mounted robots. *M 2 AS*, 1998.

[25] L. Sentis and O. Khatib. Control of free-floating humanoid robots through task prioritization. In *Proceedings of the IEEE International Conference on Robotics & Automation*, 2005.

[26] M.W. Spong and M. Vidyasagar. *Robot Dynamics and Control.* John Wiley & Sons, 1989.

[27] N. Tchï¿½aev. *Mï¿½anique rationnelle.* ï¿½itions Mir Moscou, 1993.

[28] M. Vukobratović, B. Borovac, D. Surla, and D. Stokić. *Biped Locomotion : Dynamics, Stability, Control and Application.* Springer-Verlag, 1990.

[29] P.-B. Wieber. Constrained dynamics and parametrized control in biped walking. In *Proceedings of the International Symposium on Mathematical Theory of Networks and Systems*, 2000.

[30] P.-B. Wieber and C. Chevallereau. Online adaptation of reference trajectories for the control of walking systems. Technical Report 5298, INRIA Rhône-Alpes, 2004.

[31] D.A. Winter. *Biomechanics and Motor Control of Human Movement, Second Edition.* John Wiley & Sons, New York, 1990.

Dynamic Stability of a Simple Biped Walking System with Swing Leg Retraction

M. Wisse[1], C.G. Atkeson[2], and D.K. Kloimwieder[2]

[1] Delft University of Technology
www.dbl.tudelft.nl
[2] Carnegie Mellon University
www.cs.cmu.edu/∼cga

Summary. In human walking, the swing leg moves backward just prior to ground contact, i.e. the relative angle between the thighs is decreasing. We hypothesize that this swing leg retraction may have a positive effect on gait stability, because similar effects have been reported in passive dynamic walking models, in running models, and in robot juggling. For this study, we use a simple inverted pendulum model for the stance leg. The swing leg is assumed to accurately follow a time-based trajectory. The model walks down a shallow slope for energy input which is balanced by the impact losses at heel strike. With this model we show that a mild retraction speed indeed improves stability, while gaits without a retraction phase (the swing leg keeps moving forward) are consistently unstable. By walking with shorter steps or on a steeper slope, the range of stable retraction speeds increases, suggesting a better robustness. An optimization of the swing leg trajectory of a more realistic model also consistently comes up with a retraction phase, and indeed our prototype demonstrates a retraction phase as well. The conclusions of this paper are twofold; (1) use a mild swing leg retraction speed for better stability, and (2) walking faster is easier.

1 Introduction

In human walking, the swing leg moves forward to maximal extension and then it moves backward just prior to ground contact. This backward motion is called “swing leg retraction”; the swing foot stops moving forward relative to the floor and slightly moves backward before touching the ground. In biomechanics it is generally believed that humans apply this effect (also called “ground speed matching”) in order to reduce heel strike impacts. However, we believe that there is a different way in which swing leg retraction can have a positive effect on stability; a fast step (too much energy) would automatically lead to a longer step length, resulting in a larger energy loss at heel strike. And conversely, a slow step (too little energy) would automatically lead to a shorter step length, resulting in less heel strike loss. This could be a useful stabilizing effect for walking robots.

The primary motivation to study swing leg retraction comes from our previous work on passive dynamic walking [15, 4, 19]. Passive dynamic walking [11] robots can demonstrate stable walking without any actuation or control. Their energy comes from walking downhill and their stability results from the natural dynamic pendulum motions of the legs. Interestingly, such walkers possess two equilibrium gaits, a "long period gait" and a "short period gait" [5, 12]. The long period gait has a retraction phase, and this gait is the only one that can be stable. The short period gait has no swing leg retraction. This solution is usually dismissed, as it never provides passively stable gaits.

More motivation stems from work on juggling and running, two other underactuated dynamic tasks with intermittent contact. The work on juggling [14] featured a robot that had to hit a ball which would then ballistically follow a vertical trajectory up and back down until it was hit again. The research showed that stable juggling occurs if the robot hand is following a well chosen trajectory, such that its upward motion is decelerating when hitting the ball. The stable juggling motion required no knowledge of the actual position of the ball. We feel that the motion of the hand and ball is analogous to that of the swing leg and stance leg, respectively. Also analogous is the work on a simple point-mass running model [16]. It was shown that the stability of the model was significantly improved by the implementation of a retraction phase in the swing leg motion. It has been suggested [13] that this effect also appears in walking.

In this paper we investigate the stabilizing influence of the swing leg retraction speed just prior to heel strike impact. We use a Poincaré map analysis of a simple point-mass model (Sect. 2). The results are shown in Sect. 3, including a peculiar asymmetric gait that is more stable than any of the symmetric solutions. Section 4 reports that the results are also valid for a model with a more realistic mass distribution. The discussion and conclusion are presented in Sects. 5 and 6.

2 Simulation Model and Procedure

The research in this paper is performed with an inverted pendulum model consisting of two straight and massless legs (no body) and a single point mass at the hip joint, see Fig. 1. Straight legged ("compass gait") models are widely used as an approximation for dynamic walking [7, 6, 9, 8, 5].

2.1 Stance Leg

The stance leg is modeled as a simple inverted pendulum of length 1 (m) and mass 1 (kg) (Fig. 1). It undergoes gravitational acceleration of 1 (m/s^2) at an angle of γ following the common approach to model a downhill slope

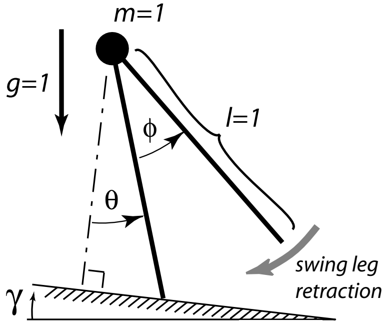


Fig. 1. Our inverse pendulum model, closely related to the "Simplest Walking Model" of [5]

in passive dynamic walking. It has one degree of freedom denoted by θ, see Fig. 1. The foot is a point and there is no torque between the foot and the floor. The equation of motion for the stance leg is:

$$\ddot{\theta} = \sin(\theta - \gamma) \tag{1}$$

which is integrated forward using a 4^{th} order Runge-Kutta integration routine with a time step of 0.001 (s).

2.2 Swing Leg

The swing leg is modeled as having negligible mass. Its motion does not affect the hip motion, except at the end of the step where it determines the initial conditions for the next step. A possible swing leg motion is depicted in Fig. 2 with a dashed line. As is standard with compass gait walkers, we ignore the brief but inevitable foot scuffing at midstance.

The swing leg motion at the end of the step is a function of time which we construct in two stages. First we choose at which relative swing leg angle ϕ (See Fig. 1) heel strike should take place, ϕ_{lc}. This is used to find a limit cycle (an equilibrium gait), which provides the appropriate step time, T_{lc}. Second, we choose a retraction speed $\dot{\phi}$. The swing leg angle ϕ is then created as a linear function of time going through the point $\{T_{lc}, \phi_{lc}\}$ with slope $\dot{\phi}$.

2.3 Transition

The simulation transitions from one step to the next when heel strike is detected, which is the case when $\phi = -2\theta$. An additional requirement is that the foot must make a downward motion, resulting in an upper limit for the forward swing leg velocity $\dot{\phi} < -2\dot{\theta}$ (note that $\dot{\theta}$ is always negative in normal walking, and note that swing leg *retraction* happens when $\dot{\phi} < 0$). In our simulation, we use a third order polynomial to interpolate between two

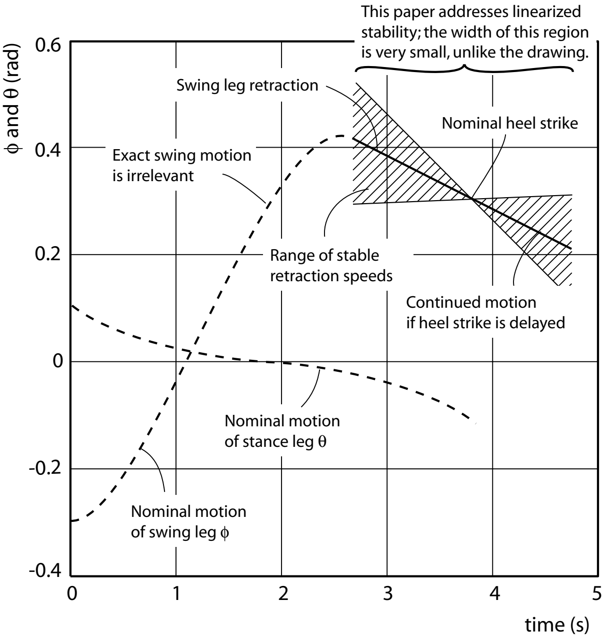


Fig. 2. The figure shows an example trajectory and it shows the stable region (*hatched area*) for retraction speeds. Only the swing leg trajectory around heel strike is important; the swing leg by itself has no dynamic effect on the walking motion other than through foot placement

simulation data points in order to accurately find the exact time and location of heel strike.

The transition results in an instantaneous change in the velocity of the point mass at the hip, see Fig. 3. All of the velocity in the direction along the new stance leg is lost in collision, the orthogonal velocity component is retained. This results in the following transition equation:

$$\dot{\theta}^{+} = \dot{\theta}^{-} \cos\phi \tag{2}$$

in which $\dot{\theta}^{-}$ indicates the rotational velocity of the old stance leg, and $\dot{\theta}^{+}$ that of the new stance leg. At this instant, θ and ϕ flip sign (due to relabeling of the stance and swing leg). Note that in Eq. (2) ϕ could equally well be replaced with 2θ.

The instant of transition is used as the start of a new step for the swing leg controller; in the case that a disturbance would make step n last longer than usual, then the start of the swing leg trajectory for step $n + 1$ is postponed accordingly. Thus, although the swing leg motion is a time-based trajectory independent of the state of the stance leg, it does depend on foot contact information for the start of the trajectory.

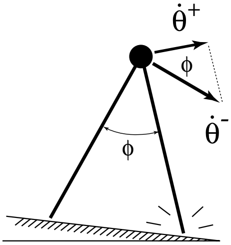


Fig. 3. At heel strike the velocity of the point mass is redirected. All velocity along the length of the new stance leg is lost, so that $\dot{\theta}^+ = \dot{\theta}^- \cos\phi$

2.4 Finding Limit Cycles

The model exhibits a limit cycle if the initial conditions of step $n+1$ are exactly equal to those of step n. For this model, the only independent initial conditions are the stance leg angle (θ) and its velocity ($\dot{\theta}$). The motion of the swing leg is fully trajectory controlled; we assume that it accurately follows the desired trajectory.

The first step of finding a limit cycle is to guess initial conditions that are near a hypothesized limit cycle, either through experience or by starting from a known limit cycle for similar parameter values. This provides initial guess $\{\theta_0, \dot{\theta}_0\}$. Then a Newton-Raphson gradient-based search algorithm is applied on the difference between $\{\theta_0, \dot{\theta}_0\}$ and the initial conditions of the next step, which we obtain through forward simulation. The search algorithm terminates when the norm of the difference is smaller than $1e^{-9}$. The search algorithm uses a numerically obtained gradient $\mathbf{J}$ which is also used for the stability analysis as described in the next paragraph. Note that this procedure can find unstable as well as stable limit cycles.

2.5 Poincaré Stability Analysis

The stability of the gait is analyzed with the Poincaré mapping method, which is a linearized stability analysis of the equilibrium gait. The Poincaré mapping method perturbs the two independent initial conditions and monitors the effect on the initial conditions for the subsequent step. Assuming linear behavior, the relation between the original perturbations at step n and the resulting perturbations at step $n+1$ is captured in the Jacobian matrix $\mathbf{J}$, as in:

$$\begin{bmatrix} \Delta\theta_{n+1} \\ \Delta\dot{\theta}_{n+1} \end{bmatrix} = \mathbf{J} \begin{bmatrix} \Delta\theta_n \\ \Delta\dot{\theta}_n \end{bmatrix} \tag{3}$$

If the magnitudes of both of the eigenvalues of $\mathbf{J}$ are smaller than 1, then errors decay step after step and the gait is stable. The eigenvalues could have imaginary parts, as was the case for the passive model [5], but in the model with trajectory control they have no imaginary parts.

2.6 Nominal Gait

We have chosen the steady passive gait with a slope of $\gamma = 0.004$ rad as a basis of reference for walking motions. For the passive model, γ is the only parameter, and for $\gamma = 0.004$ there exists only one unstable equilibrium gait (the "short period solution") and one stable equilibrium gait (the "long period solution"). We use the latter as our reference gait. The initial conditions and the step time of that gait are listed in Table 1.

Table 1. Initial conditions and step time for steady walking of the passive walking model at a slope of $\gamma = 0.004$ rad. Note that the initial velocity for the swing leg $\dot{\phi}$ is irrelevant for our study, because the swing leg motion is fully trajectory controlled

θ	0.1534 rad
$\phi = -2\theta$	0.3068 rad
$\dot{\theta}$	−0.1561 rad/s
$\dot{\phi}$	0.0073 rad/s
step time	3.853 s

3 Results

3.1 Nominal Limit Cycle

For the given gait of Table 1 on a given slope of $\gamma = 0.004$ rad, the only parameter that we can vary is the retraction speed $\dot{\phi}$; how fast is the swing leg moving rearward (or forward, depending on the sign) just prior to heel strike. This parameter has no influence on the nominal gait, but it does change the behavior under small disturbances as captured by the Poincaré stability analysis. Note that the swing leg will follow a fixed time-based trajectory independent of the disturbances on the initial conditions.

The stability results are shown in Fig. 4; the eigenvalues of $\mathbf{J}$ on the vertical axis versus the retraction speed $\dot{\phi}$ at the horizontal axis. A positive value for $\dot{\phi}$ indicates that the swing leg keeps moving forward. A value of zero means that the swing leg is being held at the heel strike value $\phi = 0.3068$ and so the foot comes down vertically. A negative value for $\dot{\phi}$ indicates the presence of a retraction phase.

Figure 4 shows that stable gaits emerge for retraction speeds of $-0.18 < \dot{\phi} < 0.009$, and that the fastest convergence will be obtained with $\dot{\phi} = -0.09$ since the maximum absolute eigenvalue is minimal at that point. In other words, swing leg retraction is not necessary for stable walking, but errors will definitely decay faster if the swing leg motion does include a retraction phase. Also, even though some forward swing leg motion ($\dot{\phi} > 0$) is allowable, this

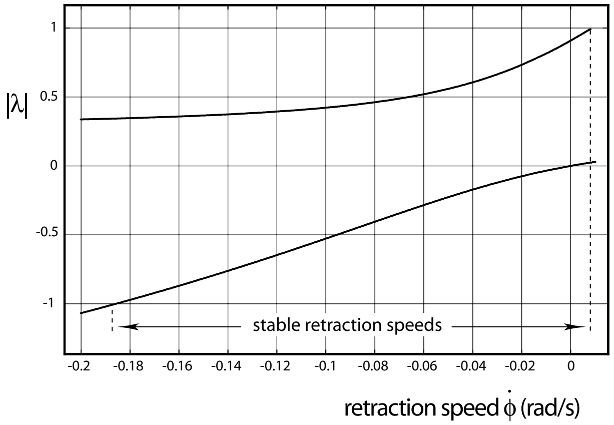


Fig. 4. The graph shows the range of stable retraction speeds. The stability is indicated with eigenvalues of **J** on the vertical axis. The walking motion is stable if the eigenvalues are between -1 and 1. The horizontal axis contains the retraction speed $\dot{\phi}$. A positive value for $\dot{\phi}$ indicates that the swing leg keeps moving forward, a value of zero means that the swing leg is being held at the heel strike value $\phi = 0.3068$ and so the foot comes down vertically. A negative value for $\dot{\phi}$ indicates the presence of a retraction phase. Stable gaits exist for retraction speeds between -0.18 and $+0.009$ rad/s. In words, this graph shows that relative hip angle should be decreasing around the instant that heel strike is expected

would make the walker operate very close to instability characterized by a rapidly growing eigenvalue.

An interesting data point is $\dot{\phi} = 0$. One of the eigenvalues there is zero ($\lambda_1 = 0$); any errors in the initial condition θ will be completely eliminated within one step, because it is certain that the step will end with $\phi = 0.3068$ as the swing leg will be held at that value until heel strike occurs. The other eigenvalue can also be calculated manually. Although the derivation is a little more involved, the result simply reads $\lambda_2 = \cos^2 \phi$. A system with $\dot{\phi} = 0$ is dynamically equivalent to the “Rimless Wheel” [10, 2].

3.2 The Influence of Step Length

We repeat the stability analysis of the previous subsection still using the same slope $\gamma = 0.004$ but varying the step length of the gait. For example, we choose a much faster and shorter step starting with $\theta_0 = -0.1317$. The limit cycle belonging to that value starts with $\dot{\theta}_0 = -0.17$ while the step time is 2 s (this is what we tuned for). The resultant eigenvalues are shown in Fig. 5. Clearly there is a much larger range of stable retraction speeds, at the cost of slightly slower convergence. A retraction speed of $\dot{\phi} = -0.71$ seems optimal, i.e. this is where we find the smallest value when taking the maximal absolute values of

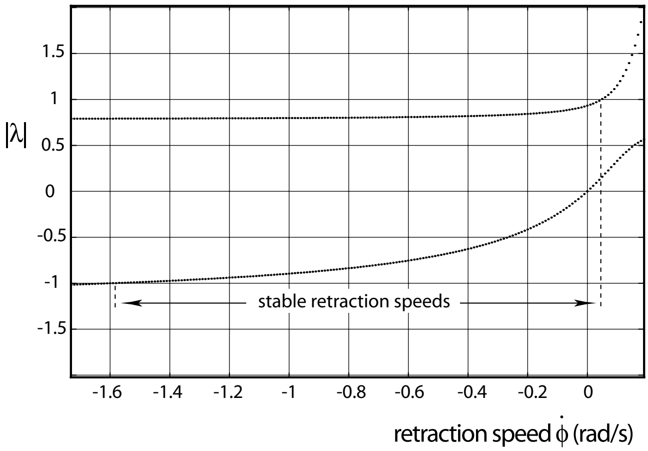


Fig. 5. A faster walking motion (T = 2 s) leads to a much larger range of stable retraction speeds

both of the eigenvalues. The absolute eigenvalues are 0.8, i.e. errors decrease 20% per step.

Figure 6 provides an overview of the effect of step length on stable range of retraction speeds and the optimal retraction speed and accompanying eigenvalues. The stable range decreases for larger step lengths until it is zero for $\phi = 0.3155$. Above that value no limit cycles exist, because the energy supply from gravity cannot match the large impact losses. Near this value, the walker is operating dangerously close to a state in which it does not have sufficient forward energy to pass the apex at midstance, resulting in a fall backward. The main conclusion from this graph is that it is wise to operate well away from a fall backward, i.e. walk fast!

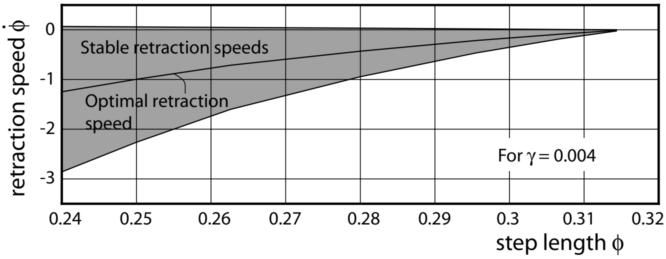


Fig. 6. Effect of step length on the range of stable retraction speeds for a floor slope of $\gamma = 0.004$. The *gray area* shows that shorter steps are better. The graph also shows the retraction speed with the smallest eigenvalues

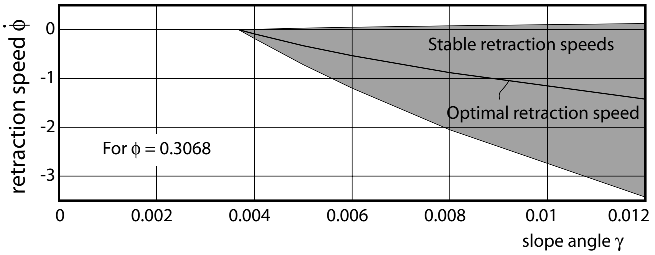


Fig. 7. Effect of floor slope on the range of stable retraction speeds for a step length of $\phi = 0.3068$. The *gray area* shows that steeper slopes (and thus faster steps) are better. The *graph* also shows the retraction speed with the smallest eigenvalues

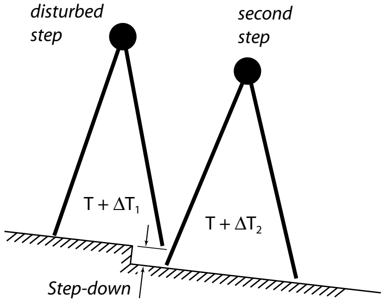


Fig. 8. A step-down of 0.1% of the leg length results in a slightly longer step time ($\Delta T_1 > 0$), followed by a significantly faster second step ($\Delta T_2 < 0$)

3.3 The Influence of the Slope Angle

The influence of the slope angle is similar to the that of the step length. A steeper slope provides more energy input and thus the resultant gait is faster, an effect similar to decreasing the step length. Figure 7 shows how the range of stable retraction speeds depends on the slope angle, for the nominal step length $\phi = 0.3068$.

3.4 Effect of Step-Down Disturbance on Step Time

During preliminary presentations of this work [18], we were asked whether the beneficial stability effect of swing leg retraction also holds in the case of a step-down in the floor. One might think that the step-down has the following adverse effect: the disturbed step takes longer than normal due to the extra step-down height (Fig. 8). The swing leg retraction then results in a *shorter*-than-normal steplength, while the extra energy should rather have been dissipated with a *longer*-than-normal steplength. Thus it seems that swing leg retraction worsens the effects of the step-down disturbance. However, we will show here that the opposite is true.

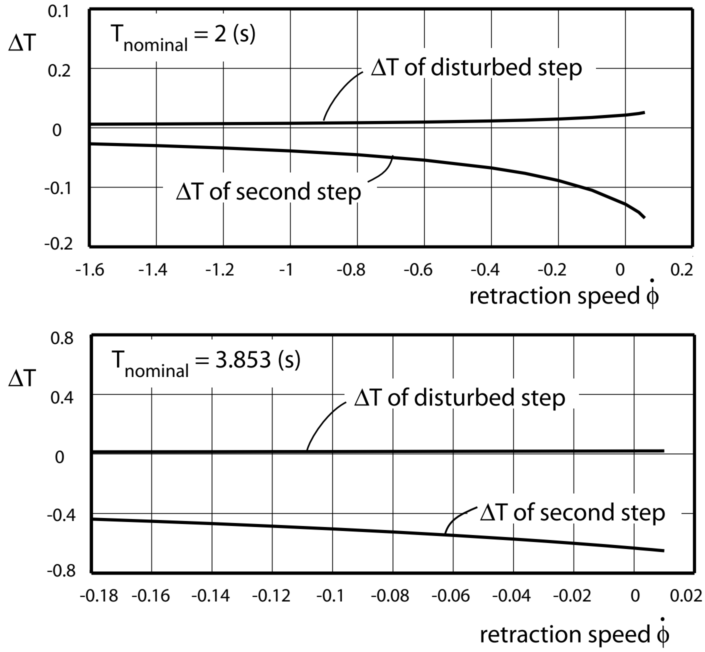


Fig. 9. Effect of a step-down of 0.1% of the leg length on the step time of the disturbed step and the step after that ("second step"). Both graphs show that the second step is much more disturbed than the disturbed step itself. The *top graph* shows that a fast walking gait results in a much smaller step time disturbance than a slow walking gait (*bottom graph*). Both graphs show that swing leg retraction is beneficial because it reduces the effect of a step-down on the step time of the second step

The most important gait characteristic for a disturbance analysis is the step time. For a simple 2D model, the step time is the most telling characteristic to predict how likely a forward or backward fall is. So, we must investigate the effect of the step-down disturbance on the step time. It turns out that the step after the disturbed step ("second step" in Fig. 8) suffers most, see Fig. 9. In this figure we show the effect of a step down of 0.1% of the leg length. The disturbed step itself has a slightly longer step time, and the second step has a significantly shorter step time. From this figure we draw two conclusions: (1) the step time is considerably less disturbed for the fast gait, so walking faster is better, and (2) the presence of a swing leg retraction phase actually *reduces* the step time disturbance ΔT, contrary to what one might expect. Note that this analysis was only done for the range of stable solutions from Figs. 4 and 5.

3.5 Asymmetric Gait is More Stable

The results in the previous sections show that the retraction speed can change the eigenvalues, but it doesn't ever seem to obtain eigenvalues of all zeros. The explanation is simple; the system uses one control input (the swing leg angle ϕ at heel strike) with which it must stabilize two states (stance leg angle θ and its velocity $\dot{\theta}$). For a discrete control situation like the one at hand, the controller needs at least two interactions with the system before a random disturbance can be eliminated. In other words: although one cannot obtain "deadbeat control" (all eigenvalues zero) within a single step, it should be possible to find a deadbeat controller for a succession of two steps. Here we present such a controller for our nominal situation of $\gamma = 0.004$ and $\phi = 0.3068$.

The previous solutions were all symmetric, i.e. the trajectory of the swing leg was the same each step. We found that a purposefully induced asymmetric gait can result in eigenvalues of all zeros. The swing leg trajectories (one for leg 1 and another for leg 2) are shown in Fig. 10. Leg 1 always goes to $\phi = 0.3068$ and does not have a retraction phase (i.e. the foot comes straight down). Leg 2 always follows a trajectory with a retraction speed of $\dot{\phi} = -0.125245$. If one calculates the eigenvalues over a series of steps of Leg 1 – Leg 2 – Leg 1 (or more), all eigenvalues are zero. This means that any disturbance will be completely eliminated after three steps of this asymmetric gait.

Preliminary research suggests that this "deadbeat" solution even pertains to large errors, although that requires a non-constant retraction speed. We intend to investigate such large-error solutions in the future. Note that in steady gait, the motion of the swing legs is asymmetric (one retracts and the other does not) but the step length and step time are still symmetric. Hence

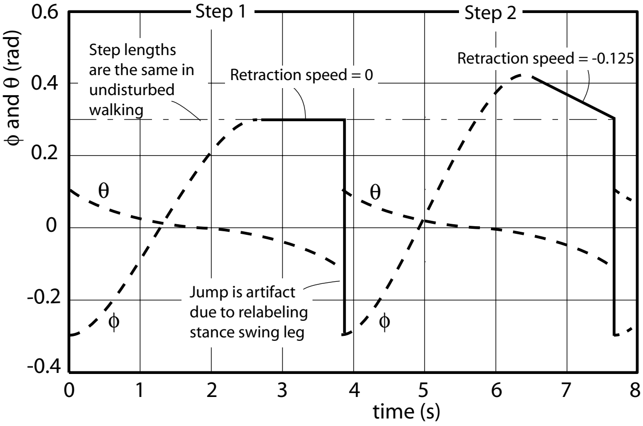


Fig. 10. An asymmetric gait can lead to two-step deadbeat control, i.e. to two eigenvalues of zero

this solution is conceptually entirely different from the "period-two" solutions found in a bifurcation analysis (e.g. [5]). Also note that due to the asymmetry, the eigenvalues cannot be divided up into "one-step" eigenvalues.

4 Automated Optimization on a More Realistic Mass Distribution also Results in Swing Leg Retraction

The theoretical results in the previous section are based on a point mass model for walking. One of the main assumptions there is that the mass of the swing leg is negligible. Obviously, in real walking systems this is not true. The reaction forces and torques from a non-massless swing leg will influence the walking motion. In our experience, the main effect is energy input. Driving the swing leg forward also pumps energy into the gait. A benefit of this effect is that a downhill slope is no longer required, but the question is whether it breaks the stabilizing effect of swing leg retraction. Or, even if it does still help stability, whether the stability gain outweighs the added energetic cost for accelerating the swing leg. We study these questions using a model with a more realistic mass distribution, based on a prototype we are currently experimenting with [1] (Fig. 11). The swing leg trajectory is optimized both for stability and for efficiency.

The model (Fig. 11) has the same topology as our initial model (Fig. 1). However, instead of a single point mass at the hip, the model now has a distributed mass over the legs, see Table 2. The swing leg follows the desired trajectory with reasonable accuracy using a PD controller on the hip torque:

$$T = k(\phi - \phi_{des}(t)) + d\dot{\phi} \tag{4}$$

with gain values $k = 1500$ and $d = 10$. The swing leg trajectory is parameterized with two knot points defining the start and the end of the retraction

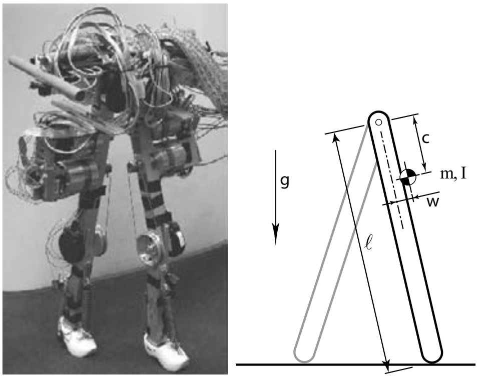


Fig. 11. Our current experimental biped and a straight-legged model with mass distribution based on the experimental robot, see Table 2

Table 2. Parameter values for a model with a more realistic mass distribution in the legs

gravity	g	9.81 m/s^2
floor slope	γ	0 rad
leg length	l	0.416 m
leg mass	m	3 kg
vertical position CoM	c	0.027 m
horizontal position CoM	w	0 m
moment of inertia	I	0.07 kgm^2

phase (Fig. 12). The trajectory before the first knot point is a third order spline which starts with the actual swing leg angle and velocity just after heel strike. The trajectory between the two knot points is a straight line. This parametrization provides the optimizer with ample freedom to vary the retraction speed, the nominal step length, and the duration of the retraction phase.

The optimization procedure is set up as follows. The model is started with manually tuned initial conditions, after which a forward dynamic simulation is run for 20 simulated seconds. The resulting motion is then rated for average velocity and efficiency:

$$cost = \Sigma_{20s}(w_T T^2 + w_v(\dot{x} - \dot{x}_{des})^2 + w_v \dot{y}^2) \tag{5}$$

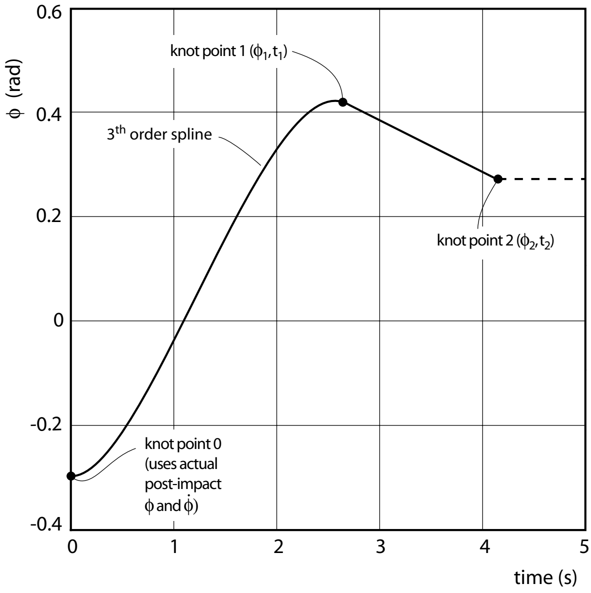


Fig. 12. The desired trajectory for the swing leg is parameterized with two knot points (four parameters)

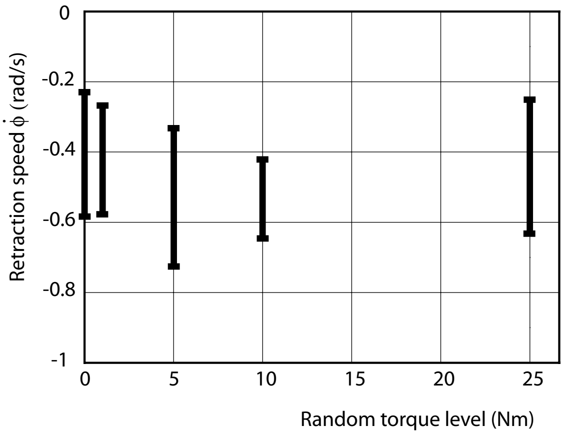


Fig. 13. Retraction speed $\dot{\phi}$ as a function of the level of random torque disturbances. A negative value for $\dot{\phi}$ indicates the presence of a retraction phase. The results are obtained with an optimization algorithm which was initialized with trajectories *with* a retraction phase for some runs and *without* one for others. Irrespective of the initialization and the level of disturbances, the optimization always settles into a trajectory *with* a retraction phase, i.e. $\dot{\phi}$ is always negative just prior to foot contact

with the weight for the torque penalty $w_T = 0.1$, the weight for the velocity penalty $w_v = 1$, and the desired forward velocity $\dot{x}_{des} = 0.3$, summed over a trial interval of 20 s. During the motion, random noise with uniform distribution is added to the hip torque. In this way, the model is indirectly rated for robustness; if the noise makes the walker fall, then the resultant average walking velocity is low and so the penalty for not achieving $\dot{x}_{des}$ is high.

A simulated annealing procedure optimized the cost function of (5) by adjusting the four knot point parameters for the swing leg trajectory. Figure 13 shows that for a wide range of noise levels and initialization values, the optimization procedure consistently settles into gaits with a retraction phase. These results fully concur with the theoretical results for the model with massless legs. Therefore, we conclude that the analysis is valid and the conclusion holds: a mild retraction speed is beneficial for the walking stability.

5 Discussion

This work was limited to a small error analysis. Our future work consists of analyzing the effect of the swing leg motion when under large disturbances, i.e. an analysis of the basin of attraction must be added to the present linearized stability analysis [19]. We also intend to investigate the effects of an increased model complexity by adding knees, feet, and an upper body.

Observations of the gait of our previously developed passive-based walking robots [17] show that they all walk with a retraction phase in the swing leg

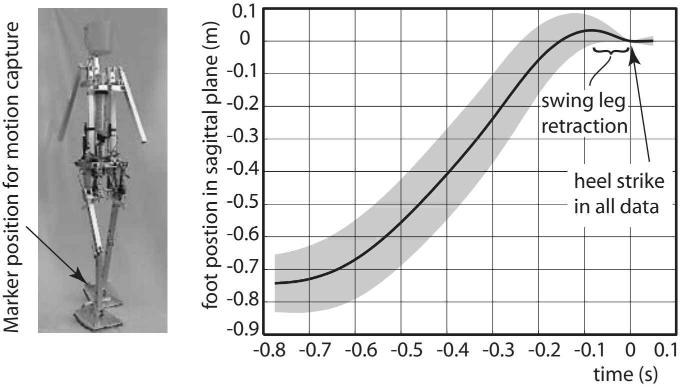


Fig. 14. *Left*: our prototype Denise. *Right*: the graph shows the clear existence of a retraction phase. The graphs shows a measurement of the motion of Denise's heel with respect to the floor. We measured over 150 steps from several trials. The data is time-synchronized using heel strike at the end as the reference, and the final position after heel strike is defined as 0 meters. Then we calculated the mean and standard deviation, which are shown in the graph

motion. For illustration, Fig. 14 shows the motion of the swing foot with respect to the floor, as measured with a motion capture system on our most recent prototype Denise [3]. The measurements (an average of over 150 steps) show that there exists a clear retraction phase just prior to heel strike.

6 Conclusion

In this paper we research the effect of swing leg retraction on gait stability. The conclusions are straightforward:

1. **Walk fast**; this decreases the sensitivity of the gait to the swing leg motion just prior to heel contact.
2. **Use mild swing leg retraction**; by moving the swing leg rearward just prior to heel contact, one avoids the highly unstable effects that occur when the swing leg is still moving forward at heel contact.

Acknowledgements

This research was supported by US National Science Foundation grants ECS-0325383 and CNS-0224419 and by the NWO, the Netherlands Organization for Scientific Research. The authors are grateful to Seiichi Miyakoshi for discussions about the concept of swing leg retraction. Also thanks to Brian Moyer at

Mark Redfern's lab of University of Pittsburgh for the motion capture measurements of robot Denise. Daan Hobbelen helped with the analysis of the step-down effects.

References

[1] S. O. Anderson, M. Wisse, C. G. Atkeson, J. K. Hodgins, G. J. Zeglin, and B. Moyer. Powered bipeds based on passive dynamic principles. In *Proc. of IEEE/RAS International Conference on Humanoid Robots*, pp. 110–116, Tsukuba, Japan, 2005.

[2] M. J. Coleman, A. Chatterjee, and A. Ruina. Motions of a rimless spoked wheel: A simple 3D system with impacts. *Dynamics and Stability of Systems*, 12(3):139–160, 1997.

[3] S. H. Collins, A. Ruina, R. L. Tedrake, and M. Wisse. Efficienct bipedal robots based on passive-dynamic walkers. *Science*, 307:1082–1085, February 18 2005.

[4] S. H. Collins, M. Wisse, and A. Ruina. A two legged kneed passive dynamic walking robot. *Int. J. of Robotics Research*, 20(7):607–615, July 2001.

[5] M. Garcia, A. Chatterjee, A. Ruina, and M. J. Coleman. The simplest walking model: Stability, complexity, and scaling. *ASME J. Biomech. Eng.*, 120(2):281–288, April 1998.

[6] A. Goswami, B. Thuilot, and B. Espiau. A study of the passive gait of a compass-like biped robot: symmetry and chaos. *Int. J. Robot. Res.*, 17(12):1282–1301, December 1998.

[7] V. T. Inman, H. J. Ralston, and F. Todd. *Human Walking.* Williams & Wilkinns, Baltimore, 1981. ISBN 0-683-04348-X.

[8] S. Kajita and K. Tani. Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode. In *Proc., IEEE Int. Conf. on Robotics and Automation*, pp. 1405–1411. IEEE, 1991.

[9] A. D. Kuo. Energetics of actively powered locomotion using the simplest walking model. *J. of Biomechanical Engineering*, 124:113–120, February 2002.

[10] T. McGeer. Powered flight, child's play, silly wheels, and walking machines. In *Proc., IEEE Int. Conf. on Robotics and Automation*, pp. 1592–1597, Piscataway, NJ, 1989.

[11] T. McGeer. Passive dynamic walking. *Int. J. Robot. Res.*, 9(2):62–82, April 1990.

[12] T. McGeer. Passive dynamic biped catalogue. In R. Chatila and G. Hirzinger, editors, *Proc., Experimental Robotics II: The 2nd International Symposium*, pp. 465–490, Berlin, 1992. Springer–Verlag.

[13] S. Miyakoshi, T. Yamamoto, G. Taga, and Y. Kuniyoshi. Bipedal walking simulation of a compass-like robot. In *Proc. of The Robotics Society of Japan*, pp. 1107–1108, 2000. (in Japanese).

[14] S. Schaal and C. G. Atkeson. Open loop stable control strategies for robot juggling. In *Proc., IEEE Int. Conf. on Robotics and Automation*, pp. 913–918, Piscataway, NJ, 1993. IEEE.

[15] A. L. Schwab and M. Wisse. Basin of attraction of the simplest walking model. In *Proc., ASME Design Engineering Technical Conferences*, Pittsburgh, Pennsylvania, 2001. ASME. Paper number DETC2001/VIB-21363.

[16] A. Seyfarth, H. Geyer, and H. Herr. Swing-leg retraction: A simple control model for stable running. *J. Exp. Biol.*, 206:2547–2555, 2003.

[17] M. Wisse. Three additions to passive dynamic walking; actuation, an upper body, and 3d stability. In *Proc., Int. Conf. on Humanoid Robots*, Los Angeles, USA, 2004. IEEE.

[18] M. Wisse, C. G. Atkeson, and D. K. Kloimwieder. Swing leg retraction helps biped walking stability. In *Proc. of IEEE/RAS International Conference on Humanoid Robots*, pp. 295–300, Tsukuba, Japan, 2005.

[19] M. Wisse, A. L. Schwab, R. Q. van der Linde, and F. C. T. van der Helm. How to keep from falling forward; elementary swing leg action for passive dynamic walkers. *IEEE Trans. on Robotics*, 21(3):393–401, 2005.

Lecture Notes in Control and Information Sciences

Editors: M. Thoma, M. Morari

Further volumes of this series can be found on our homepage:
springer.com

Vol. 340: Diehl, M.; Mombaur, K. (Eds.)
Fast Motions in Biomechanics and Robotics
460 p. 2006 [3-540-36118-9]

Vol. 339: Alamir, M.
Stabilization of Nonlinear Systems Using
Receding-horizon Control Schemes
325 p. 2006 [1-84628-470-8]

Vol. 338: Tokarzewski, J.
Finite Zeros in Discrete Time Control Systems
325 p. 2006 [3-540-33464-5]

Vol. 337: Blom, H.; Lygeros, J. (Eds.)
Stochastic Hybrid Systems
395 p. 2006 [3-540-33466-1]

Vol. 336: Pettersen, K.Y.; Gravdahl, J.T.;
Nijmeijer, H. (Eds.)
Group Coordination and Cooperative Control
310 p. 2006 [3-540-33468-8]

Vol. 335: Kozáowski, K. (Ed.)
Robot Motion and Control
424 p. 2006 [1-84628-404-X]

Vol. 334: Edwards, C.; Fossas Colet, E.;
Fridman, L. (Eds.)
Advances in Variable Structure and Sliding Mode
Control
504 p. 2006 [3-540-32800-9]

Vol. 333: Banavar, R.N.; Sankaranarayanan, V.
Switched Finite Time Control of a Class of
Underactuated Systems
99 p. 2006 [3-540-32799-1]

Vol. 332: Xu, S.; Lam, J.
Robust Control and Filtering of Singular Systems
234 p. 2006 [3-540-32797-5]

Vol. 331: Antsaklis, P.J.; Tabuada, P. (Eds.)
Networked Embedded Sensing and Control
367 p. 2006 [3-540-32794-0]

Vol. 330: Koumoutsakos, P.; Mezic, I. (Eds.)
Control of Fluid Flow
200 p. 2006 [3-540-25140-5]

Vol. 329: Francis, B.A.; Smith, M.C.; Willems,
J.C. (Eds.)
Control of Uncertain Systems: Modelling,
Approximation, and Design
429 p. 2006 [3-540-31754-6]

Vol. 328: Loría, A.; Lamnabhi-Lagarrigue, F.;
Panteley, E. (Eds.)
Advanced Topics in Control Systems Theory
305 p. 2006 [1-84628-313-2]

Vol. 327: Fournier, J.-D.; Grimm, J.; Leblond, J.;
Partington, J.R. (Eds.)
Harmonic Analysis and Rational Approximation
301 p. 2006 [3-540-30922-5]

Vol. 326: Wang, H.-S.; Yung, C.-F.; Chang, F.-R.
HąÞ Control for Nonlinear Descriptor Systems
164 p. 2006 [1-84628-289-6]

Vol. 325: Amato, F.
Robust Control of Linear Systems Subject to Uncertain
Time-Varying Parameters
180 p. 2006 [3-540-23950-2]

Vol. 324: Christoąędes, P.; El-Farra, N.
Control of Nonlinear and Hybrid Process Systems
446 p. 2005 [3-540-28456-7]

Vol. 323: Bandyopadhyay, B.; Janardhanan, S.
Discrete-time Sliding Mode Control
147 p. 2005 [3-540-28140-1]

Vol. 322: Meurer, T.; Graichen, K.; Gilles, E.D.
(Eds.)
Control and Observer Design for Nonlinear Finite
and Inąęnite Dimensional Systems
422 p. 2005 [3-540-27938-5]

Vol. 321: Dayawansa, W.P.; Lindquist, A.;
Zhou, Y. (Eds.)
New Directions and Applications in Control Theory
400 p. 2005 [3-540-23953-7]

Vol. 320: Steffen, T.
Control Reconąęguration of Dynamical Systems
290 p. 2005 [3-540-25730-6]

Vol. 319: Hofbaur, M.W.
Hybrid Estimation of Complex Systems
148 p. 2005 [3-540-25727-6]

Vol. 318: Gershon, E.; Shaked, U.; Yaesh, I.
HąÞ Control and Estimation of State-muliplicative
Linear Systems
256 p. 2005 [1-85233-997-7]

Vol. 317: Ma, C.; Wonham, M.
Nonblocking Supervisory Control of State Tree
Structures
208 p. 2005 [3-540-25069-7]

Vol. 316: Patel, R.V.; Shadpey, F.
Control of Redundant Robot Manipulators
224 p. 2005 [3-540-25071-9]

Vol. 315: Herbordt, W.
Sound Capture for Human/Machine Interfaces: Practical Aspects of Microphone Array Signal Processing
286 p. 2005 [3-540-23954-5]

Vol. 314: Gil', M.I.
Explicit Stability Conditions for Continuous Systems
193 p. 2005 [3-540-23984-7]

Vol. 313: Li, Z.; Soh, Y.; Wen, C.
Switched and Impulsive Systems
277 p. 2005 [3-540-23952-9]

Vol. 312: Henrion, D.; Garulli, A. (Eds.)
Positive Polynomials in Control
313 p. 2005 [3-540-23948-0]

Vol. 311: Lamnabhi-Lagarrigue, F.; Loría, A.; Panteley, E. (Eds.)
Advanced Topics in Control Systems Theory
294 p. 2005 [1-85233-923-3]

Vol. 310: Janczak, A.
Identiącation of Nonlinear Systems Using Neural Networks and Polynomial Models
197 p. 2005 [3-540-23185-4]

Vol. 309: Kumar, V.; Leonard, N.; Morse, A.S. (Eds.)
Cooperative Control
301 p. 2005 [3-540-22861-6]

Vol. 308: Tarbouriech, S.; Abdallah, C.T.; Chiasson, J. (Eds.)
Advances in Communication Control Networks
358 p. 2005 [3-540-22819-5]

Vol. 307: Kwon, S.J.; Chung, W.K.
Perturbation Compensator based Robust Tracking Control and State Estimation of Mechanical Systems
158 p. 2004 [3-540-22077-1]

Vol. 306: Bien, Z.Z.; Stefanov, D. (Eds.)
Advances in Rehabilitation
472 p. 2004 [3-540-21986-2]

Vol. 305: Nebylov, A.
Ensuring Control Accuracy
256 p. 2004 [3-540-21876-9]

Vol. 304: Margaris, N.I.
Theory of the Non-linear Analog Phase Locked Loop
303 p. 2004 [3-540-21339-2]

Vol. 303: Mahmoud, M.S.
Resilient Control of Uncertain Dynamical Systems
278 p. 2004 [3-540-21351-1]

Vol. 302: Filatov, N.M.; Unbehauen, H.
Adaptive Dual Control: Theory and Applications
237 p. 2004 [3-540-21373-2]

Vol. 301: de Queiroz, M.; Malisoff, M.; Wolenski, P. (Eds.)
Optimal Control, Stabilization and Nonsmooth Analysis
373 p. 2004 [3-540-21330-9]

Vol. 300: Nakamura, M.; Goto, S.; Kyura, N.; Zhang, T.
Mechatronic Servo System Control
Problems in Industries and their Theoretical Solutions
212 p. 2004 [3-540-21096-2]

Vol. 299: Tarn, T.-J.; Chen, S.-B.; Zhou, C. (Eds.)
Robotic Welding, Intelligence and Automation
214 p. 2004 [3-540-20804-6]

Vol. 298: Choi, Y.; Chung, W.K.
PID Trajectory Tracking Control for Mechanical Systems
127 p. 2004 [3-540-20567-5]

Vol. 297: Damm, T.
Rational Matrix Equations in Stochastic Control
219 p. 2004 [3-540-20516-0]

Vol. 296: Matsuo, T.; Hasegawa, Y.
Realization Theory of Discrete-Time Dynamical Systems
235 p. 2003 [3-540-40675-1]

Vol. 295: Kang, W.; Xiao, M.; Borges, C. (Eds)
New Trends in Nonlinear Dynamics and Control, and their Applications
365 p. 2003 [3-540-10474-0]

Vol. 294: Benvenuti, L.; De Santis, A.; Farina, L. (Eds)
Positive Systems: Theory and Applications (POSTA 2003)
414 p. 2003 [3-540-40342-6]

Vol. 293: Chen, G. and Hill, D.J.
Bifurcation Control
320 p. 2003 [3-540-40341-8]

Vol. 292: Chen, G. and Yu, X.
Chaos Control
380 p. 2003 [3-540-40405-8]

Vol. 291: Xu, J.-X. and Tan, Y.
Linear and Nonlinear Iterative Learning Control
189 p. 2003 [3-540-40173-3]

Vol. 290: Borrelli, F.
Constrained Optimal Control
of Linear and Hybrid Systems
237 p. 2003 [3-540-00257-X]

Vol. 289: Giarré, L. and Bamieh, B.
Multidisciplinary Research in Control
237 p. 2003 [3-540-00917-5]

Vol. 288: Taware, A. and Tao, G.
Control of Sandwich Nonlinear Systems
393 p. 2003 [3-540-44115-8]